LG

Life's Good

Getting Started with Niagara Framework® Software Manual

niagara4

PROPRIETARY DATA NOTICE

This document, as well as all reports, illustrations, data, information, and other materials are the property of LG Electronics U.S.A., Inc., and are disclosed by LG Electronics U.S.A., Inc., only in confidence.

⃠ ***Do not throw away, destroy, or lose this manual.***

Please read carefully and store in a safe place for future reference.
Content familiarity required for proper installation.

The instructions included in this manual must be followed to prevent product malfunction, property damage, injury, or death to the user or other people. Incorrect operation due to ignoring any instructions will cause harm or damage. A summary of safety precautions begins on page 6.

For more technical materials such as submittals, engineering databooks, and catalogs, visit www.lghvac.com.

SOM_Getting_Started_with_Niagara_Framework_05_17

For continual product development, LG Electronics U.S.A., Inc., reserves the right to change specifications without notice.

This document, as well as all reports, illustrations, data, information, and other materials are the property of LG Electronics U.S.A., Inc.

TABLE OF CONTENTS

SAFETY INSTRUCTIONS

The instructions below must be followed to prevent product malfunction, property damage, injury or death to the user or other people. Incorrect operation due to ignoring any instructions will cause harm or damage. The level of seriousness is classified by the symbols described below.

TABLE OF SYMBOLS

⚠DANGER	This symbol indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.
⚠WARNING	This symbol indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.
⚠CAUTION	This symbol indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury.
Note:	This symbol indicates situations that may result in equipment or property damage accidents only.
🛇	This symbol indicates an action that should not be performed.

This manual provides basic information about the Niagara 4 Framework and Workbench. Included are basic descriptions of the Workbench as well as reference information to help systems integrators and engineers get started with Niagara.

Refer to the LG MultiSITE VM3 Installation Manual for installation and mounting instructions of the controller.

⚠DANGER

🛇 **Do not use or store flammable gas or combustibles near the MultiSITE VM3 unit.**
There is risk of fire, explosion, and physical injury or death.

Disconnect power before installing or servicing the unit.
There is risk of physical injury or death due to electric shock.

🛇 **Do not touch any exposed outdoor unit wiring, terminals, or other electrical components with tools or exposed skin. Only qualified technicians should install, use, or remove this unit.**
Improper installation or use may result in fire, explosion, electric shock, physical injury and/or death.

SAFETY INSTRUCTIONS

⚠WARNING

All electric work must be performed by a licensed electrician and conform to local building codes or, in the absence of local codes, with the National Electrical Code, and the instructions given in this manual.
If the power source capacity is inadequate or the electric work is not performed properly, it may result in fire, electric shock, physical injury or death.

🛇 **Do not change the settings of the protection devices.**
If the pressure switch, thermal switch, or other protection device is shorted and forced to operate improperly, or parts other than those specified by LG are used, there is risk of fire, electric shock, explosion, and physical injury or death.

Dispose of any packing materials safely.
Packing materials, such as nails and other metal or wooden parts may cause puncture wounds or other injuries.

Tear apart and throw away plastic packaging bags so that children may not play with them and risk suffocation and death.

🛇 **Do not install the MultiSITE VM3 unit if it will be exposed to rain or other precipitation.** 🛇 **Do not install the unit in a location exposed to open flame or extreme heat.** 🛇 **Do not touch the unit with wet hands.**
There is risk of fire, electric shock, physical injury and/or death.

⚠CAUTION

Wear protective gloves when handling equipment.

Sharp edges may cause personal injury.

Note:

Disconnect power before installing or servicing the unit.

There is risk of equipment damage or degraded performance.

MultiSITE VM3 unit is for use with select LG air conditioning systems only. 🛇 Do not attempt to use this unit with any other type of system.

There is risk of equipment damage or degraded performance.

Clean up the site after all procedures are finished, and check that no metal scraps, screws, or bits of wiring have been left inside or surrounding the controller or indoor units.

🛇 Do not allow water, dirt, or animals to enter the controller.

There is risk of unit failure or degraded performance.

🛇 Do not spill water or other liquid on the inside of the controller. 🛇 Do not drop the controller into water. If the unit is immersed in water or other liquid, contact your local authorized LG distributor for support.

There is risk of unit failure or degraded performance.

Remove all power to controller before attaching (plug in) or detaching (unplug) any option module.

There is risk of possible equipment damage.

🛇 Do not remove the controller’s cover.

No configurable or user-serviceable items (such as jumpers or a battery) require cover removal. All items are accessible as switches and connectors on the unit’s top, bottom, and side, or behind the unit’s front access door or microSD card shutter.

This device is only intended for use as a monitoring and control device. 🛇 Do not use it for any other purpose.

There is risk of data loss or equipment damage.

Before removing or inserting the microSD card, disconnect all power to the controller and use static discharge precautions.

There is risk of equipment damage.

The MultiSITE VM3 unit is not compatible with a Power-Over-Ethernet (POE) network. 🛇 Do not connect the controller on a network segment which carries power.

The unit may fail.

CERTIFICATIONS

The MultiSITE VM3 controller has the following agency listings, compliances, and certifications:

UL-916, Energy Management Equipment - Edition 4

FCC Part 15, Class B - Federal Communications Commission, with FCC Part 15, Subpart C - WiFi

ICES-003, Class B - Industry Canada Interference-Causing Equipment Standard

RoHS 2 (Restriction of Hazardous Substances), Directive 2011/65/EU.

CE Declaration of Conformity (Council Directive 004-108-EC)

ACMA, complies with the requirements of the relevant ACMA Standards. This document covers mounting and wiring of the following products.

COMPLIANCE AND APPROVALS

Federal Communications Commission (FCC)

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

1. This device may not cause harmful interference, and

2. This device must accept any interference received, including interference that may cause undesired operation.

Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Canadian Department of Communications (DOC)

This device complies with Industry Canada License-exempt RSS standard(s). Operation is subject to the following two conditions: 1) this device may not cause interference, and 2) this device must accept any interference, including interference that may cause undesired operation of the device.

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (EIRP) is not more than that necessary for successful communication.

The device for operation in the band 5150–5250 MHz is only for indoor use to reduce the potential for harmful interference to co-channel mobile satellite systems.

Approved Antenna Listing

- ANT-DB1-RAF-RPS

Transmitter Module Listing

- Contains Transmitter Module FCC ID: W98-12977
- Contains Transmitter Module IC: 8339A-12977

To comply with FCC and Industry Canada RF exposure limits for general population /uncontrolled exposure, the antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

Product Documentation

This document is part of the Niagara technical documentation library. Released versions of Niagara software include a complete collection of technical information that is provided in both online help and PDF format.

Related documentation

Document Title	Description
AX to N4 Migration Guide	This document describes processes and considerations for users that want to migrate NiagaraAX stations to Niagara 4.
Hierarchies Guide	This document provides user information for working with the Hierarchies feature.
Histories Guide	This document covers the concepts of histories, history services, history components and plugins, and describes common histories related tasks.
Lexicon Guide	This document explains the concepts of lexicons, and describes how to use Workbench lexicon tools.
Relations Guide	This document explains the concepts of entity relationships, creating, editing, and tagging relations.
Scheduling Guide	This document explains the concepts of scheduling and describes how to add and configure different types of schedules. Also covered descriptions about linking and importing schedules.
Station Security Guide	This document introduces and provides procedures for using: secure communication (TLS/SSL), email security, user authentication (AuthenticationService), and component authorizations (Categories, Roles and some details about hierarchies and how they provide security).
Tagging Guide	This document describes tags, tag dictionaries, tag groups, direct and implied tags, and other concepts along with common tagging tasks.
Templates Guide	This document explains template concepts and includes common template tasks.
Web Charts Guide	This document describes and illustrates use of web charting tools.
Niagara 4 Installation Guide	This document describes how to install Niagara 4 on various platforms.
Niagara 4 Platform Guide	Describes Workbench views that are available when you have a platform connection to a Niagara 4 host. It also describes PlatformServices that are available in a Niagara 4 station.

Related documentation, continued.

Document Title	Description
Provisioning Guide	This document provides concepts and procedures related to using the Niagara provisioning tools.
Data Recovery Service Guide	This document describes how to use the data recovery service to manage station backups.
JACE® 8000 Backup and Restore Guide	This document describes how to make a backup and restore a backup using the USB port on a JACE 8000.
JACE NiagaraAX Install & Startup Guide	This document covers the initial software installation and configuration for a QNX-based JACE controller using Workbench. Applicable controllers include the JACE 3, 6, 7 series controllers.

CHAPTER 1 ABOUT THE NIAGARA FRAMEWORK

About this guide

This document provides basic information about the Niagara 4 Framework and Workbench. Included are basic descriptions of the Workbench as well as reference information to help systems integrators and engineers get started with Niagara.

In order to make the most of the information in this book, readers should have some training or previous experience with Niagara 4 or NiagaraAX software.

The following topics are covered in this chapter:

- About Niagara 4
- About control systems integration
- About Java
- About common networking and Internet protocols
- About programming for non-programmers
- About systems capabilities
- About component software design
- About the software architecture
- About Baja
- About Niagara building blocks
- Formats (BFormat)

LonWorks®

BACnet and ASHRAE are registered trademarks of ASHRAE. Microsoft, Excel, Internet Explorer, Windows, Windows Vista, Windows Server, and SQL Server are registered trademarks of Microsoft Corporation. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation. Tridium, JACE, Niagara Framework, NiagaraAX Framework, and Sedona Framework are registered trademarks, and Workbench, WorkPlaceAX, and AXSupervisor, are trademarks of Tridium Inc. All other product names and services mentioned in this publication that is known to be trademarks, registered trademarks, or service marks are the property of their respective owners.

Software frameworks provide a platform to allow businesses to more easily build their end-product offerings. The patented Niagara Framework is targeted at solving the challenges associated with managing diverse smart devices, unifying their data, and connecting them to enterprise applications. Examples of smart devices include: monitoring and control systems, sensors, metering systems, and embedded controls on packaged equipment systems.

Framework is something composed of parts fitted together and united; a structural frame; a basic structure (as of ideas); in object-oriented programming, a reusable basic design structure, consisting of abstract and concrete classes, that assists in building applications. Niagara Framework is a universal software infrastructure that allows companies to build custom, web enabled applications for accessing, automating, and controlling smart devices in real time over the Internet.

Niagara 4 is the fourth generation of the Niagara Framework. This UX framework provides an infrastructure which enables systems integrators and developers to build device-to-enterprise solutions, and Internet enabled control and monitoring products. The framework integrates diverse systems and devices (regardless of manufacturer or communication protocol) into a unified platform that can be easily managed in real time over the Internet (or intranet) using a late version HTML5-capable web browser. The framework supports “queryable” tags, the functionality on which many of the new features (search, tagging, relations, templates, and hierarchies) are based. The framework also includes a cutting-edge toolset that enables non-programmers to build rich applications in a drag-and-drop environment.

Niagara is fully scalable, meaning that it can be run on platforms spanning the range from small, embedded devices to enterprise class servers. Niagara is successfully applied globally in energy-services, building-automation, industrial-automation and M2M applications.

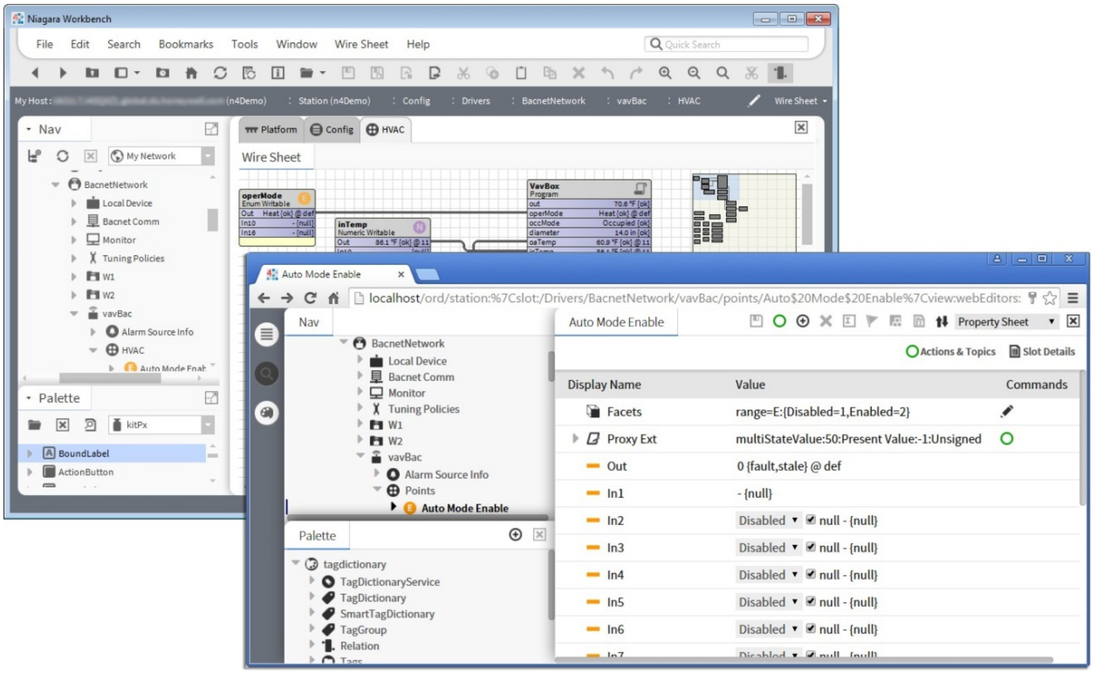
About Niagara 4

The user experience (UX) is at the heart of Niagara. The Niagara 4 “bajaux” framework is based on open web technologies, such as HTML5, CSS3, JSON, and Bajascript v2.0. The bajaux framework effectively provides an improved user experience in utility, ease of use, and efficiency, as well as the capability to produce feature-rich charts and dashboards. Niagara 4 is designed to be dynamic and responsive in order to accommodate changing usage circumstances and changes to individual systems over time. The improvements of this new framework can be summed up in the following categories: visualization improvements, tagging and templating, and advanced security.

Visualization improvements

The cutting-edge HTML5-based, bajaux user interface benefits the end-user as well as provides an improved developer experience. Building owners and facility managers who typically need to locate and visualize data in order to drive efficiencies can utilize the dashboard and web chart reporting capabilities. Developers can easily create dynamic, interactive applications and views using bajaux widgets which display across Niagara media, such as Workbench or in a modern web browser (no browser plug-in required). Search functionality integrated in Workbench enables quickly locating data to drop into other views. Optimized workflows for common tasks require fewer clicks and provide more intuitive interaction. Additionally, the new look and feel of the interface incorporates a clean and crisp design, with a subdued and focused use of color to emphasize important information.

Figure 1: Workbenchinterface (left) compared with web browser view (right)



localhost/ord/station:%7Cslot:/Drivers/BacnetNetwork/vavBac/points/Auto$20Mode$20Enable%7Cview:webEditors:

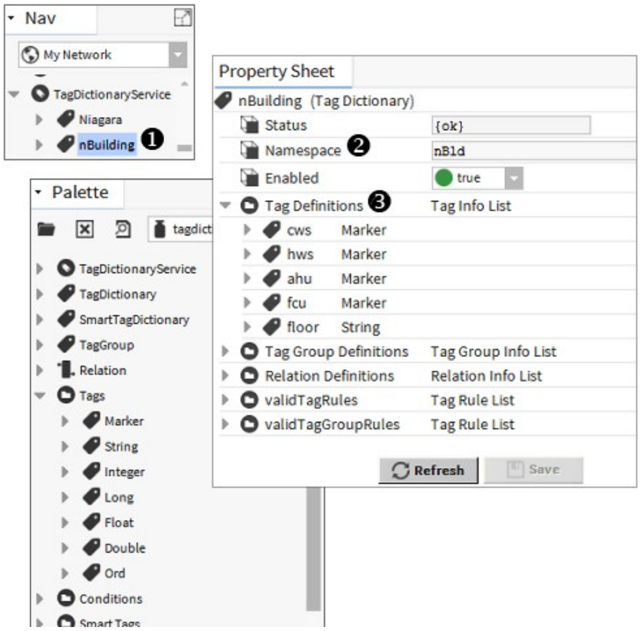
Display Name	Value	Commands
Facets	range=E:{Disabled=1,Enabled=2}	
Proxy Ext	multiStateValue:50:Present Value:-1:Unsigned	○
Out	0 {fault,stale} @ def	
In1	- {null}	
In2	Disabled ▾ ☑ null - {null}	
In3	Disabled ▾ ☑ null - {null}	
In4	Disabled ▾ ☑ null - {null}	
In5	Disabled ▾ ☑ null - {null}	
In6	Disabled ▾ ☑ null - {null}	
In7	Disabled ▾ ☑ null - {null}	

Tagging and templating

Systems integrators will find that metadata tagging at the component level enables a number of ways to find data, via search and navigation hierarchies, as well as providing ways to narrow results via filtering. Similarly, creating templates using pre-tagged devices results in built-in reusability which, of course, translates to shorter integration time. Using tag-based hierarchies, multiple navigation schemes can easily be created for different user roles. Also, there is no need to update Nav files every time new devices are added to a system.

In the following figure, callout 1 refers to Custom tag dictionary, 3 refers to tag definitions in Property Sheet. Tagdictionary palette appears on lower left.

Figure 2: Custom tag dictionary



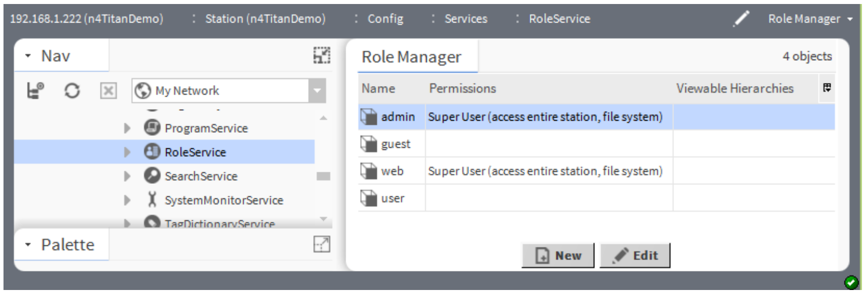
nBuilding (Tag Dictionary)

Status	{ok}
Namespace	nBld
Enabled	● true
Tag Definitions	Tag Info List
cws	Marker
hws	Marker
ahu	Marker
fcu	Marker
floor	String
Tag Group Definitions	Tag Group Info List
Relation Definitions	Relation Info List
validTagRules	Tag Rule List
validTagGroupRules	Tag Rule List

Advanced security

Developers and systems integrators can easily create secure systems using role-based access control and enhanced encryption features. Additional security enhancements include configurable authentication based on connection type, code signing which verifies that modules have not been modified and improved platform connection security. Also, security usability is greatly improved so that you do not have to be a cyber security expert to develop a secure system.

Figure 3: Role Manager view



Name	Permissions	Viewable Hierarchies
admin	Super User (access entire station, file system)	
guest		
web	Super User (access entire station, file system)	
user		

About control systems integration

Using the Niagara Framework, control systems integration enables you to do the following

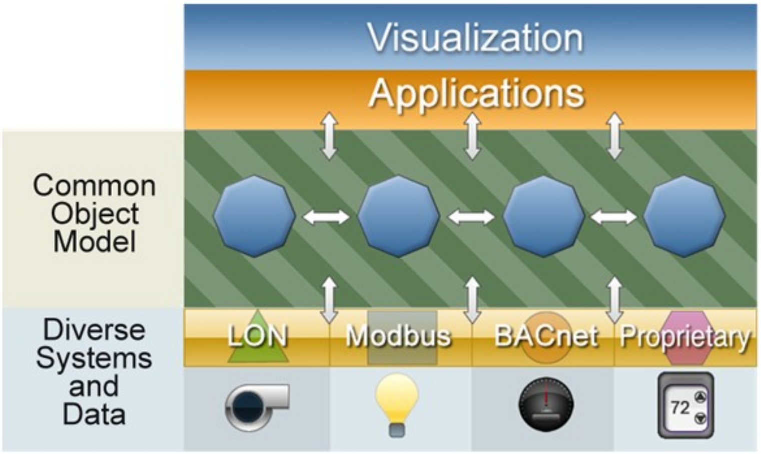
1. Connecting devices on a common communications media
2. Modeling those devices in software
3. Programming applications to use the information in those devices

Before a device, such as a chiller, VAV box, or temperature sensor, can be used, information from those devices must be pulled into the system software.

Niagara then models those devices and their data types in software through the common object model. This usually entails simplifying the device’s data types to make them easier to manipulate and control through the software.

The Niagara common object model is then used to build applications, with the goal being to provide non-programmers a means to program the system easily without developing raw code. The Niagara common object model is similar to a programming language in that there are a few key concepts that are used, but the real power is in the reusable libraries of applications and collections of objects that are available. Once you understand the key concepts and you can put them to work, you can use the objects to build control system solutions quickly and efficiently.

Figure 4: Niagara common object model



The Niagara common object model allows the framework to do the following:

- provide secure two-way communication between devices and the Internet
- send real-time device information across the Internet
- control devices in real-time across the Internet.

About Java

Much of the Niagara software is written in Java, which means that it is platform independent. Prior to Java, most software was written and compiled for a particular machine or operating system. If that software needs to run on some other processor, the program has to be compiled again. Java, on the other hand, compiles once. Niagara software runs on embedded JACE controllers using the QNX operating system and the IBM J9 Java Virtual Machine (JVM), and runs on Microsoft Windows desktop operating system platforms, as well as Linux and Solaris using the HotSpot JVM.

About Virtual Machines

It is possible to compile code once and run it on any platform due to a layer of software that exists between the machine and the software called the Java virtual machine (JVM). The Niagara Framework uses the Java VM as a common runtime environment across various operating systems and hardware platforms. The core framework scales from small embedded controllers to high end servers. The framework runtime is targeted at Java 8 Standard Edition, Compact 3 profile for runtime components. The user interface toolkit and programming tools are targeted at Java 8 Standard Edition.

There are a number of different virtual machines for different platforms on which the NRE is running, but the NRE itself, and all of its modules, are the same regardless of platform. The VM is responsible for defining how the software works with a given set of hardware, how it talks to a LonWorks adapter, how it talks to the communications port, how it interacts with the operating system, among other tasks.

About common networking and Internet protocols

Niagara is designed from the ground up to assume that there will never be any one "standard" network protocol, distributed architecture, or fieldbus. The design goal is to integrate cleanly with all networks and protocols. The framework standardizes what's inside the box, not what the box talks to.

The Niagara software suite implements a highly efficient adaptation of the Java component software model and current Internet technologies to provide true interoperability across a wide range of automation products. The object model can be used to integrate a wide range of physical devices, controllers, and primitive control applications including LonMark profiles, BACnet objects, and legacy control points. The architecture supports future enhancements by allowing legacy systems to be brought forward, where they can readily adopt new standards, solutions, and applications.

Enterprise-level software standards include Transmission Control Protocol/Internet Protocol (TCP/IP), eXtensible Markup Language (XML), Hyper Text Transfer Protocol Secure (HTTPS), Transport Layer Security (TLS), and others. These standards provide the foundation on which to build solutions that allow information to be shared between the control system and the enterprise information system.

About programming for non-programmers

Most features in the Niagara Framework are designed for dual use (for programmers as well as non-programmers). These features are designed around a set of Java APIs to be accessed by developers writing Java code. At the same time, most features are also designed to be used through high level graphical programming and configuration tools. This vastly increases the number of users capable of building applications on the Niagaraplatform.

About systems capabilities

Niagara is designed to run across a range of embedded systems, and to provide scalability to highly distributed systems.

About embedded systems capabilities

Niagara is one of the only software stacks designed to run across the entire range of processors from small embedded devices to server class machines. Niagara is targeted for embedded systems capable of running a Java VM. This excludes some very low-end devices that lack 32-bit processors or have only several megabytes of RAM, but opens up a wide range of processor platforms.

To speed time to market for partners designing smart devices, a reference design processor core has been developed. Known as the Niagara Processor Module (NPM), this platform can be licensed and makes it possible to quickly develop Niagara-based products.

About distributed systems capabilities

The framework is designed to provide scalability to highly distributed systems composed of tens of thousands of nodes running the Niagara software. Systems of this size span a wide range of network topologies and usually communicate over unreliable Internet connections. Niagara is designed to provide an infrastructure for managing systems of this scale.

About component software design

Niagara uses an architecture centered on the concept of “Component Oriented Development.” A component is a piece of self-describing software that can be assembled like building blocks to create new applications.

A component-centric architecture provides the following:

- Data normalization

Components provide a model used to normalize the data and features of different types of protocols and networks so that they can be integrated seamlessly.

- Graphic development tools

Applications can be assembled with components using graphical tools in the Niagara Workbench. This allows new applications to be built without requiring a Java developer.

- Application visibility

Components provide unsurpassed visibility into applications. Since components are self-describing, it is very easy for tools to look at how an application is assembled, configured, and to determine what is occurring at any point in time. This provides great value in debugging and maintaining applications.

- Software reuse

Components enable software reuse. Niagara supports custom development and extension of the framework. Niagara components are extensible and go beyond data and protocols to unify the entire development environment.

About the software architecture

The following images illustrate the Niagara software subsystems and the software processes and protocols, respectively.

Figure 5: Niagara software subsystems

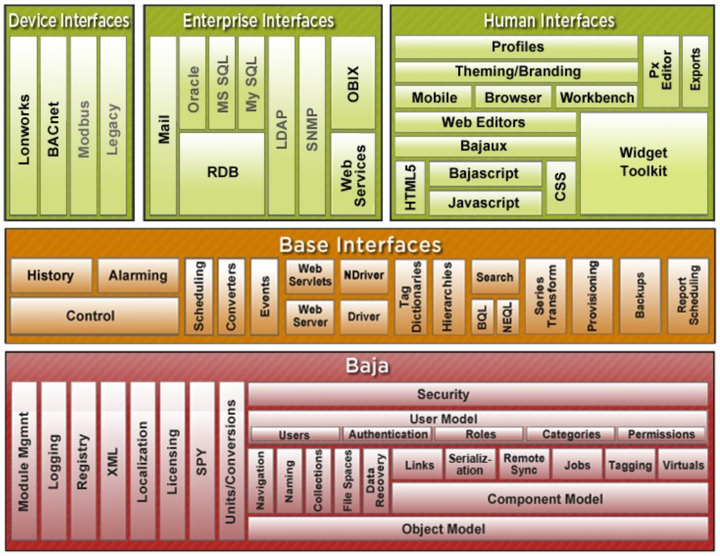
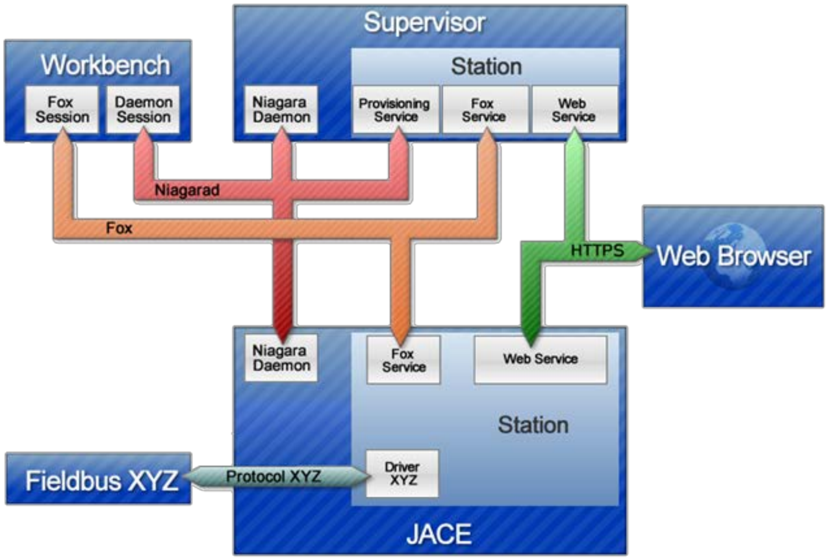


Figure 6: Niagara software processes and protocols



About Baja

The Baja (Building Automation Java Architecture) core framework is designed to be published as an open standard. This standard is being developed through Sun's Java Community Process as JSR 60. This JSR is still an ongoing effort, but it is important to understand the distinction between Baja and Niagara.

Fundamentally Baja is an open specification and the Niagara Framework is an implementation of that specification. As a specification, Baja is not a set of software, but rather purely a set of documentation.

The Baja specification will include:

- Standards for how Baja software modules are packaged
- XML schema for the component model;
- The component model and its APIs
- Historical database components and APIs
- Alarming components and APIs
- Control logic components and APIs
- Scheduling components and APIs
- BACnet driver components and APIs
- LonWorks driver components and APIs

Over time many more specifications for features will be added to Baja. But what is important to remember is that Baja is only a specification. Niagara is an implementation of that specification. Furthermore you will find a vast number of features in Niagara that are not included under the Baja umbrella. In this respect Niagara provides a superset of the Baja features.

About APIs

The API (Application Programming Interface) defines how software engineers access the capabilities of software like the Niagara Framework. Workbench is a Niagara API. Using it you create and edit the control logic for your job site.

Many features found in the Niagara framework are exposed through a set of Java APIs. In the Java world, APIs are grouped together into packages, which are scoped using DNS domain names. Software developed through the Java Community Process is usually scoped by packages starting with “java” or “javax.” It is important to understand the two types of APIs related to the framework.

- javax.baja

The APIs developed for Baja are grouped under javax.baja. These APIs are part of the open Baja specification and may be implemented by vendors other than Tridium. Using these APIs guarantees a measure of vendor neutrality and backward compatibility.

- com.tridium

Software which is proprietary and outside of the Baja specification is grouped under the com.tridium packages. The com.tridium packages contain code specific to how Niagara implements the Baja APIs.

The com.tridium code may or may not be documented. If com.tridium APIs are publicly documented, then Tridium encourages developers to use them, but does not guarantee backward compatibility. Undocumented com.tridium APIs should never be used by developers.

Note that some APIs have been developed under javax.baja even though they are not currently part of the Baja specification. These APIs may eventually be published through Baja, but are currently in a development stage.

About Niagara building blocks

This section describes some of the fundamental building blocks of the Niagara framework. It is important to understand these concepts and associated terminology in order to fully benefit from the use of the Niagara Workbench.

Concepts include the following:

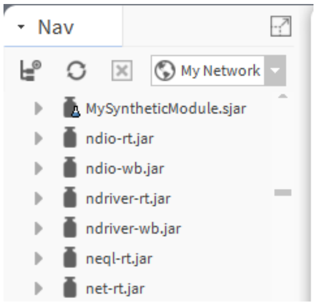
- Modules — the most basic unit of the software that comprises Niagara.
- Components — the primary building blocks of the Niagara framework.
- Presentation XML (Px) — an XML file format that defines how Niagara visualizes information (text, graphics, alarms, and so on) across diverse media, such as: Workbench, web browsers, and mobile devices.
- Stations — the main unit of server processing in the Niagara architecture. Defined by a single .bog file, a station “application” is launched as a single virtual machine (VM) on the host PC.
- ORDs — (Object Resolution Descriptor) is the Niagara universal identification system and is used throughout the framework.
- Views — a “visualization” of a component, such as: Property Sheet view, Wire Sheet view, etc.
- Lexicons — Niagara provides non-English language support by use of lexicons. Distributed as lexicon modules identified by two-digit Java locale codes, such as “fr” (French) as evidenced by this filename: ni-agaraLexiconFr.jar.

About modules

Modules are the smallest units of software in the Niagara architecture.

Major releases of the Niagara software are distributed along with a set of release modules (.jar extension), but as new modules are made available for that release, they may be distributed as independent revisions within that release. The following figure shows a partial list of modules, as displayed in the Nav side bar pane.

Figure 7: Module listing in the nav side bar tree



Note:

Don’t confuse modules with components. Components are used to build Niagara implementations, while modules make up the Niagara software itself.

Module characteristics

All modules are composed of a single Java ARchive (.jar) file that complies with PKZIP compression. Modules contain an XML manifest; they state their dependencies on other modules and their versions.

About jar files

Module .jar files are the mechanism for distributing Java modules. A .jar file is basically a compressed package whose components can be viewed with PKZIP or other archive viewing tool. Do not attempt to unzip a . jar file. The extracted file will not work. Module .jar files are the add-ins that are plugged into the software to give additional functionality. Inside a JAR file are the compressed software; its documentation; and in some cases, examples or pre-canned applications; and libraries.

There are different types of modules based on the applicable runtime profile (bajaux, doc, runtime):

- moduleName-rt.jar indicates a runtime module
- moduleName-wb.jar indicates a Workbench module
- moduleName-ux.jar indicates a bajaux web module
- moduleName-doc.jar indicates an online help module
- moduleName-se.jar indicates a Java SE compact-3 compliant module

Some modules, such as the Lon Works module, are distributed as multiple JAR files because there are so many different Lon Works devices. The core protocol is packaged in a JAR file called lonWorks.jar. There are individual .jar files for each LON device manufacturer (for example, lon_mfgName.jar). Every .jar file has its own version number.

About module versions

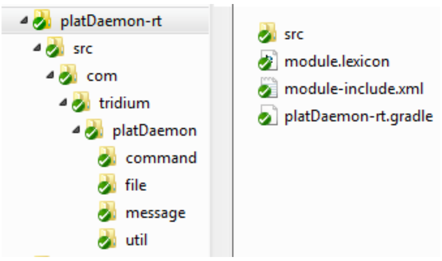
Niagara uses the module version number to make sure that you have the latest available module. Versions are specified as a series of whole numbers separated by a period, for example 4.0.16. To understand which version of a module is more recent, simply observe the module version number. For example, 4.0.16 is higher (more recent) than 4.0.8 because 4.0.16 > 4.0.8.

About directory structure

Every (.jar) module is managed in its own directory structure. The image below shows an example of a directory for the platDaemon-rt module.

All of the source code that is used to build the module’s .jar file is located under the src directory. During the build process the libJar directory is used to build up the image, which is then zipped up to create the module's .jar file.

Figure 8: Directory structure used to build the module’s jar



```
platDaemon-rt
  src
    com
      tridium
        platDaemon
          command
          file
          message
          util
```

```
src
module.lexicon
module-include.xml
platDaemon-rt.gradle
```

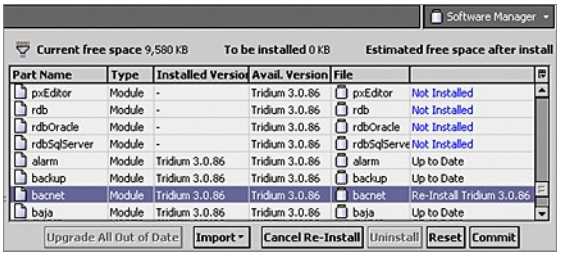
Benefits of modular software development

Modular software development provides several benefits, including the following:

- Improves tracking deployment and the assigning of version numbers

For example, if a new driver is available to be added to Niagara Framework, it can be packaged, delivered, and added in as a single module or with multiple modules, using the platform Software Manager view.

Figure 9: Software manager view



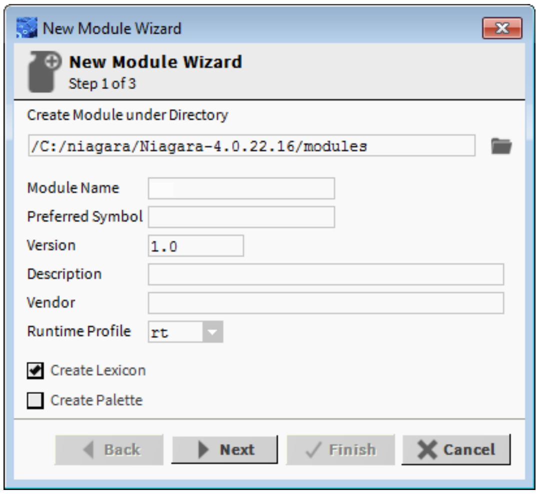
Current free space 9,580 KB   To be installed 0 KB   Estimated free space after install

Part Name	Type	Installed Versio	Avail. Version	File	
pxEditor	Module	-	Tridium 3.0.86	pxEditor	Not Installed
rdb	Module	-	Tridium 3.0.86	rdb	Not Installed
rdbOracle	Module	-	Tridium 3.0.86	rdbOracle	Not Installed
rdbSqlServer	Module	-	Tridium 3.0.86	rdbSqlServe	Not Installed
alarm	Module	Tridium 3.0.86	Tridium 3.0.86	alarm	Up to Date
backup	Module	Tridium 3.0.86	Tridium 3.0.86	backup	Up to Date
bacnet	Module	Tridium 3.0.86	Tridium 3.0.86	bacnet	Re-Install Tridium 3.0.86
baja	Module	Tridium 3.0.86	Tridium 3.0.86	baja	Up to Date

- Requires less space on the controller: Niagara’s modular development allows you to save space on your JACE by installing only necessary modules.
- Requires less space on the workstation: When you install the framework, you can choose the modules that you want to install with your application and omit any modules that you do not need. Later, you add modules using the platform Software Manager (as shown).
- Simplifies dependencies and secures files: A .jar’s runtime profile describes its contents based on which systems are able to use them, or on which content is appropriate for its configuration. This simplifies handling of dependencies and enables files to be secured via digital signature.

- Creates new modules: Software developers can use the New Module tool in Workbench to create new (.jar) modules and deploy them.

Figure 10: New module wizard



New Module Wizard

New Module Wizard
Step 1 of 3

Create Module under Directory
/C:/niagara/Niagara-4.0.22.16/modules

Module Name
Preferred Symbol
Version 1.0
Description
Vendor
Runtime Profile rt

☑ Create Lexicon
□ Create Palette

About components

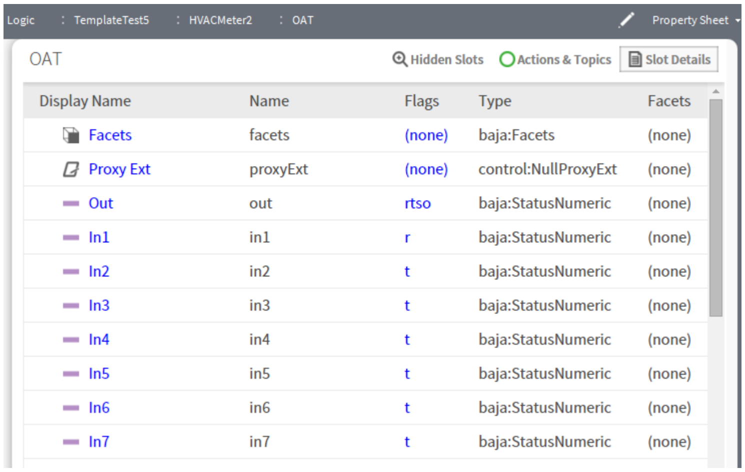
A component is the primary building block that you use to engineer an application using Niagara Workbench. As described in the section, “About component software design”, components provide many advantages for the application developer.

Components differ from modules in that components comprise an implementation of the framework, whereas modules comprise the software itself.

About slots

Niagara components are defined as a collection of “slots.” You can see all the slots that make up a component by viewing its slot details on either the Property Sheet view or AX Slot Sheet view.

Figure 11: Slot details for a single component



Display Name	Name	Flags	Type	Facets
Facets	facets	(none)	baja:Facets	(none)
Proxy Ext	proxyExt	(none)	control:NullProxyExt	(none)
Out	out	rtso	baja:StatusNumeric	(none)
In1	in1	r	baja:StatusNumeric	(none)
In2	in2	t	baja:StatusNumeric	(none)
In3	in3	t	baja:StatusNumeric	(none)
In4	in4	t	baja:StatusNumeric	(none)
In5	in5	t	baja:StatusNumeric	(none)
In6	in6	t	baja:StatusNumeric	(none)
In7	in7	t	baja:StatusNumeric	(none)

- Slot type: There are three types of slots:
  - Property: Property slots represent a storage location of another object.
  - Action: An action is a slot that specifies behavior that may be invoked either through a user command or by an event. Actions provide the capability to provide direction to components. They may be issued manually by the operator or automatically through links. Actions can be invoked in the Property Sheet view or by right-clicking on the component in the Nav tree.
  - Topic: Topics represent the subject of an event. Topics contain neither a storage location, nor a behavior. Rather a topic serves as a place holder for an event source.

- Slot name: Every slot is identified by a slot name that is unique within its type. Slot names must contain ASCII letters or numbers.
- Slot definition: Slots are either frozen or dynamic. A frozen slot is defined at compile time within a Type's Java class. That means that frozen slots are consistent across all instances of a specified Type – they don't change. Dynamic slots may be added, removed, renamed, and reordered during runtime – they can change. The power of the framework is in providing a consistent model for both frozen (compile time) slots and dynamic (runtime) slots.
- Flags: Slots have flags that allow modification of an object's presentation or behavior. For example, "read–only", "operator allowed", and "hidden", are some of the slot flags that may be used to restrict the presentation or behavior of an object.
- Facets: Facets contain metadata about an object. For example, "units of measurement" is a type of facet. Facets may be viewed in the slot sheet and edited from a component property sheet.

About master/slave components

Master components can be defined so that persistent properties are copied to slave components when they are changed. This allows you to change a property in one component that automatically updates the components that are linked to it as slaves. In this way a master component can update slave components in all the other stations in a system.

About point components

In any Niagara station, all real-time data are normalized within the station database as points, a special group of components. The following image shows several types of control points, as listed in the control module palette in Workbench.

Each type of point may be used for different purposes. When you engineer a job you may want to name a point, for example: a NumericWritable point named CondSetpoint. Points may be named and renamed but they retain their initial point type characteristics and their characteristic icon color.

Points serve as a type of shell, to which you may add point extensions. These extensions allow you to select only those functions that you need and thereby limit your point properties to just those that are necessary for your current application.

About component naming

In a station, components should be properly named using the following set of rules:

- Only alphanumeric (A-Z, a-z, 0-9) and underscore (_) characters are used. Spaces, hyphens, or other symbols characters (e.g: %, &, ., #, and so on) are illegal in component names. For further details, see the section, “Escaped names”, which covers using ASCII code in component names.
- The first character in the name must be a letter (not a numeral).
- The name must be unique for every component within the same parent component. Workbench automatically enforces this rule, via a popup error message.
- Naming is case-sensitive—for example, zone21 and Zone21 are unique names.

Note:

Case differences among names affect “name sorts” in table-based views, which order by ASCII code sequence, that is capital letters (A-Z) first, lower case (a-z) following.

To convey multiple-word names without using spaces, naming conventions, such as “CamelCase” and/or underscores are often used. For example:

- Floor1 or Floor_1
- ReturnAirTemp or Return_Air_Temp
- Zone201_SAT or Zone_201_SAT

About palettes

The palette provides a hierarchical view of available components. You may copy a component from the palette and paste it where you need it — on a Wire Sheet, Property Sheet, Px View, or in the Nav side bar pane. You can also create custom palettes that you associate with a module and use to hold a collection of frequently used components.

Escaped names

Workbench allows you to name components improperly, such as with spaces or other non-alphanumeric characters, without any warning.

Further, various drivers have learn features to automate the creation of points, some of which (by default) may also have such improper names—reflective of the native name of the source object. For example, a BACnet proxy point might have the default name Zone 6 RH%, which matches the actual (native) BACnet object’s name.

In any case, be aware that the actual component name has all illegal characters “escaped” using a $ character, along with the ASCII code for that character, in hexadecimal. The proxy point mentioned above, for example, results in the name Zone$206RH$25, where the $20 escapes the space and the $25 escapes the %. You can see these escaped names in the slot sheet of the component’s parent container. Or, with the component selected, look at its ord (shortcut Ctrl + L) to see its actual name. Other examples include the dash character “-” which is “$2d” and anytime you begin a name with a number, the “$3” is appended to the front of the name.

For the most part, this “escaped name” scheme is transparent to users. Whenever the name is displayed to the user, say in the Nav side bar, property sheet, wire sheet, or a point manger, the component’s name is “unescaped” by replacing the code (say, $20) with the actual ASCII character (say, a space). This way, the user sees Zone 6 RH% and so on. This is the component’s display name.

In some cases, escaped names lead to confusion. You should avoid them if possible (rename them). For example, if you add history extensions to escaped-named points, you see those escape codes listed for source points when accessing the History Ext Manager (although associated histories use the display names). Or, if you are building Px pages and manually typing in ords in Px widgets, you probably know source points by display names only. If you manually type in an ord without the actual (escaped) name, the widget binding fails with an error.

Note:

If this sounds too complicated, remember that drag-and-drop operations resolve escaped names without problems—for example, drag any point onto a Px page to get its proper ord.

About presentation

The Niagara framework provides a powerful presentation architecture based on XML and the Niagara component model.

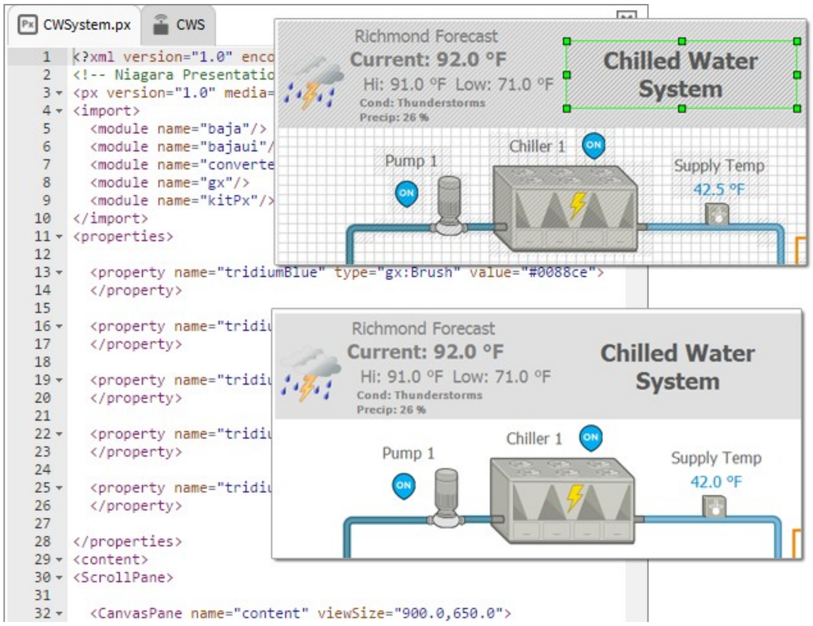
Presentation is a term that is used to describe how the software framework provides visualization of information across different types of media. The terms information, visualization, and media may comprise the following:

- information
  - real-time data
  - historical data
  - configuration data
  - alarm data
  - scheduling data
  - graphical data
  - textual data
- visualization
  - graphics (bitmap, JPG, PNG, SVG, vector)
  - videos
  - text documents
  - tables
  - charts
  - dashboards
  - input controls (actions, field editors, text fields, check boxes, trees)
- media
  - web browsers (HTML5, CSS3, JavaScript)
  - Workbench (Niagara stack)
  - mobile devices
  - csv
  - pdf
  - svg
  - xml
  - printed pages

About presentation xml (Px)

An XML file format is used to define a Niagara presentation. This file format is called “Px” for presentation XML and the term “Px” is commonly used to describe the Niagara presentation architecture. Presentation is a term that describes how Niagara visualizes information (text, graphics, alarms, and so on) across diverse media, such as Workbench, web browsers, mobile devices, and so on. Niagara uses “presentation xml” (Px) to accomplish this. The following image shows an example of a Px file in the Text Editor (Px source file), the Px Editor view, and as displayed in the Px Viewer.

Figure 12: Px file in Text Editor (left), Px Editor (top right), and Px Viewer



```
<?xml version="1.0" enco
<!-- Niagara Presentatio
<px version="1.0" media=
<import>
  <module name="baja"/>
  <module name="bajaui"/
  <module name="converte
  <module name="gx"/>
  <module name="kitPx"/>
</import>
<properties>

  <property name="tridiumBlue" type="gx:Brush" value="#0088ce">
  </property>

  <property name="tridiu
  </property>

  <property name="tridiu
  </property>

  <property name="tridiu
  </property>

  <property name="tridiu
  </property>

</properties>
<content>
<ScrollPane>

  <CanvasPane name="content" viewSize="900.0,650.0">
```

About presentation modules

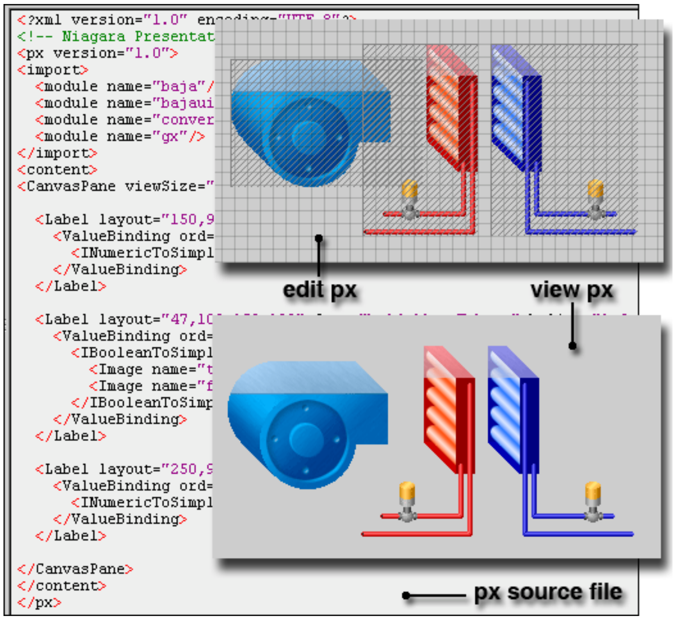
Niagara's presentation architecture is based on many modules and their associated public APIs:

- baja — the baja module defines the core component model upon which Niagara subsystems are built. This document describes enhancements to the baja component model which will be used by the presentation stack.
- gx — the gx module provides an API for drawing 2D graphics to a device. The gx module deals with drawing primitives: color, strokes, gradients, vector geometries (line, rectangle, ellipse, paths), bitmap images, fonts, and transforms.
- bajaui — the bajaui module provides the widget toolkit. Widgets are the basis for graphical composition, layout, user input, and data binding. The bajaui module builds upon gx to paint the widgets.
- bajaux — the bajaux module provides dynamic HTML5 functionality. Bajaux widgets, such as the Dashboard widget, enable you to add data, modify the view. They also allow you to save your customized version of the widget for subsequent viewing.
- PDF — the PDF implementation of the gx APIs.
- HTML — the HTML rendering and data binding engine.

About presentation xml (Px)

An XML file format is used to define a Niagara presentation. This file format is called "Px" for presentation XML and the term "Px" is commonly used to describe the Niagara presentation architecture. Presentation is a term that describes how Niagara visualizes information (text, graphics, alarms, and so on) across heterogeneous media, such as: Workbench, desktop browsers, handheld devices, and so on. Niagara uses "presentation xml" (Px) to accomplish this. The following image shows an example of a Px file in the Text Editor (Px source file), the Px Editor view, and as displayed in the Px Viewer.

Figure 13: Px file in Text Editor, Px Editor, and Px Viewer



```
<?xml version="1.0" en
<!-- Niagara Presentat
<px version="1.0">
<import>
  <module name="baja"/
  <module name="bajaui
  <module name="conver
  <module name="gx"/>
</import>
<content>
<CanvasPane viewSize="

  <Label layout="150,9
    <ValueBinding ord=
      <INumericToSimpl
    </ValueBinding>
  </Label>

  <Label layout="47,10
    <ValueBinding ord=
      <IBooleanToSimpl
        <Image name="t
        <Image name="f
      </IBooleanToSimp
    </ValueBinding>
  </Label>

  <Label layout="250,9
    <ValueBinding ord=
      <INumericToSimpl
    </ValueBinding>
  </Label>

</CanvasPane>
</content>
</px>
```

About presentation design philosophy

The presentation architecture is based on the following design principles:

- Component model is the core philosophy of every subsystem in Niagara is to build atop the component model. The presentation architecture is no exception. The design embraces a pure component model solution. The component model is used for the normalized representation of presentation, the “document object model” (DOM) of a Niagara presentation is always a “BComponent” tree.
- Unified visualization presentations include a unified approach to representing graphics, text, and input controls all within a single component model. This allows all visualizations to share a common file format as well as a rendering, layout, and input API.
- Unified media: the design goal of Px is to build a single presentation that can be used across multiple media. For example, given a Px file, it can be automatically rendered using the Workbench, as an HTML web page, or as a PDF file. All presentations are stored in the normalized component model as Px files.

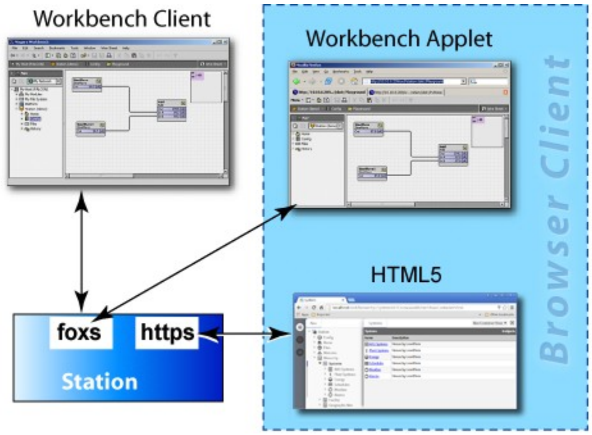
About presentation media

Niagara supports four primary target media.

- Workbench— the Niagara stack must be available to take full advantage of the presentation architecture. The desktop version of Workbench provides a standalone application that can render presentations with their full power. With the WbProfile feature, new custom applications can easily be built using Workbench and its presentation engine.
- Workbench applet — the applet allows the full Niagara stack to be downloaded to a client machine using a web browser. Once loaded to the client, you can run Workbench as an applet within an HTML page. In this scenario, the presentation engine will be available with the full feature set. The WbProfile feature may be used to customize how Workbench is decorated inside the HTML page.

- PDF — Adobe's PDF format is the standard way to export a presentation to print it. PDF provides explicit control for how a presentation is rendered on paper in various sizes. It is also a convenient file format for access via HTTP or email. The presentation architecture includes an engine for generating PDF files from Px files.
- HTML5 — provides enhanced capabilities for users and developers. A set of open web technologies (HTML5, CSS3, and JavaScript) provide a modern web interface using common standards. HTML5 views offer interactive functionality which allows you to edit properties and invoke commands right in the view. Other HTML5 functionality includes context sensitive menus, ability to add data to views dynamically. For the designer/developer, consistent rendering across media means you can develop a view once and it renders in both Workbench and Hx interfaces. The bajaux HTML5 widgets included by default provide interactive charting and dashboarding functionality. The bajaux widgets also integrate into the environment. For example, commands defined for a WebWidget render as added icons in the Workbench tool bar or in a modern HTML5-capable web browser.

Figure 14: Presentation media options



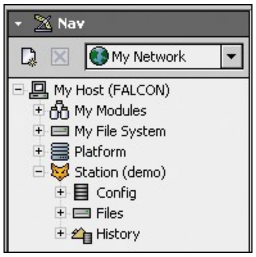
About stations

A station is the main unit of server processing in the Niagara architecture.

- A station database is defined by a single .bog file, for example: file:!stations/{name}/config.bog
- Stations are booted from their config.bog file into a single Virtual Machine (VM), or process, on the host machine.
- There is usually a one–to–one correspondence between stations and host machines (Supervisors or JACEs). However it is possible to run two stations on the same machine if they are configured to use different IP ports.

A station runs the components of the Niagara Framework and provides access for client browsers to view and control these components. The primary parts of a station include components and services. It is the combination of a database, a web server, and a control engine. The station either runs on a Web Supervisor PC or a JACE controller.

Figure 15: Station in nav tree

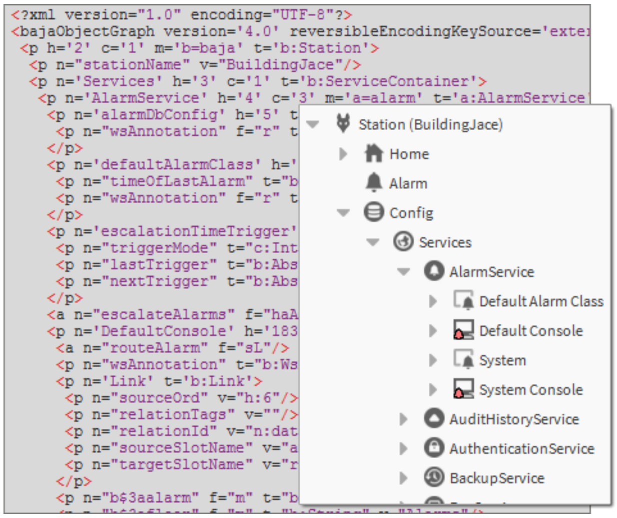


A system can be a single station or multiple stations depending on the size of the project and it is defined by a bog file.

About BOG files

A bog file (Baja object graph) contains Niagara components. It can be a complete database or any collection of components. A bog file is a special file that describes the components in a database. All views can be used on components in a bog file just as if they were in a station.

Figure 16: Sample bog file and Nav tree



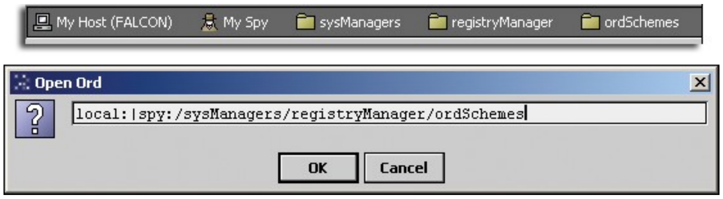
```
<?xml version="1.0" encoding="UTF-8"?>
<bajaObjectGraph version='4.0' reversibleEncodingKeySource='exte
 <p h='2' c='1' m='b=baja' t='b:Station'>
  <p n="stationName" v="BuildingJace"/>
  <p n='Services' h='3' c='1' t='b:ServiceContainer'>
   <p n='AlarmService' h='4' c='3' m='a=alarm' t='a:AlarmService
    <p n='alarmDbConfig' h='5' t
     <p n="wsAnnotation" f="r" t
    </p>
    <p n='defaultAlarmClass' h='
     <p n="timeOfLastAlarm" t="b
     <p n="wsAnnotation" f="r" t
    </p>
    <p n='escalationTimeTrigger'
     <p n="triggerMode" t="c:Int
     <p n="lastTrigger" t="b:Abs
     <p n="nextTrigger" t="b:Abs
    </p>
    <a n="escalateAlarms" f="haA
    <p n='DefaultConsole' h='183
     <a n="routeAlarm" f="sL"/>
     <p n="wsAnnotation" t="b:Ws
     <p n='Link' t='b:Link'>
      <p n="sourceOrd" v="h:6"/>
      <p n="relationTags" v=""/>
      <p n="relationId" v="n:dat
      <p n="sourceSlotName" v="a
      <p n="targetSlotName" v="r
     </p>
     <p n="b$3aalarm" f="m" t="b
```

About ORDs

An ORD is an “Object Resolution Descriptor”. The ORD is the Niagara universal identification system and is used throughout the Niagara Framework. The ORD unifies and standardizes access to all information. It is designed to combine different naming systems into a single string and has the advantage of being parsable by a host of public APIs.

An ORD is comprised of one or more queries where each query has a scheme that identifies how to parse and resolve to an object. ORDs may be displayed visually, as with the Open Ord locator or they may be entered in a text field, as shown in the Open ORD dialog box.

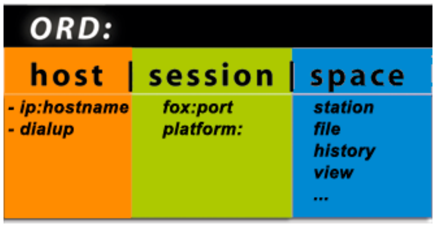
Figure 17: Open ORD and graphic locator system



local:|spy:/sysManagers/registryManager/ordSchemes

ORDs can be relative or absolute. A relative ORD takes the format of “slot:…”, such as “slot:AHU1/ Points/SpaceTemp”. The ORD is “relative” to the base ORD that contains slot: AHU1. An absolute ORD usually takes the general format of “host|session|space”, as illustrated below.

Figure 18: Absolute ORD typical structure



- host — identifies a machine usually by an IP address such as “ip:hostname”. For example “fox:” indicates a fox session to the host.
- session — identifies a protocol being used to communicate with the host.
- space — identifies a particular type of object. Common spaces are “module:”, “file:”, “station:”, “view:”, “spy:”, or “history:”

The local VM is a special case identified by “local:” which always resolves to BLocalHost.INSTANCE. The local host is both a host and a session (since no communication protocols are required for access).

Both a slot path and a handle scheme can name components within a ComponentSpace. So the ORD for a component usually involves both a space query and a path/handle.

ORD examples

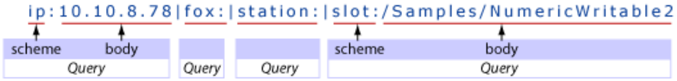
- ip:somehost|fox:|station:|slot:/MyService
- ip:somehost|fox:|station:|h:/42
- ip:somehost|fox:|file:/C:/dir/file.txt
- local:|file:!jre/lib/logging.properties
- local:|module://icons-ux/x16/cloud.png
- local:|spy:/

In Niagara you may view the complete list of installed ORD schemes at “spy:/sysManagers/registryManager/ordSchemes” (“local:|fox:|spy:/sysManagers/registryManager/ordSchemes”).

About schemes

An ORD is a list of one or more queries separated by the “|” pipe symbol. Each query is an ASCII string formatted as “<scheme>:<body>”.

Figure 19: Example ORD scheme and body



- scheme: the scheme name is a globally unique identifier which specifies, in Niagara, how to find a piece of code to lookup an object from the body string.
- body: the body string is formatted differently, according to the requirements of the scheme. The only rule is that it cannot contain a pipe symbol. Queries can be piped together to let each scheme focus on how to lookup a specific type of object. In general, absolute ords are in the following format: host | session | space.

Some examples are the following:

– ip:somehost|fox:|file:/dir/somefile.txt

In this example, the “ip” scheme is used to identify a host machine. The “fox” scheme specifies a session to that machine usually on a specific IP port number. Finally, the “file” scheme identifies an instance of a file within the “somehost” file system.

– ip:somehost|fox:1912|station:|slot:/Graphics/Home

In this example, the “ip” scheme is used to identify a host machine using an IP address. The “fox” scheme specifies a session to that machine usually on a specific IP port number. Finally, the “station” and “slot” schemes identify a specific component in the station database.

– local:|module://icons/x16/cut.png

This example illustrates a special case. The scheme “local” which always resolves to BLocalHost.INSTANCE is both a host scheme and a session scheme. It represents objects found within the local VM.

Types of schemes

A scheme is a globally unique identifier which specifies, in Niagara, how to find a piece of code to lookup an object.

- ip

The “ip” scheme is used to identify an Ip Host instance. Ords starting with “ip” are always absolute and ignore any base that may be specified. The body of a “ip” query is a DNS hostname or an IP address of the format “dd.dd.dd.dd”.

- fox

The “fox” scheme is used to establish a Fox session. Fox is the primary protocol used by Niagara for IP communication. A “fox” query is formatted as “fox:” or “fox:<port>”. If port is unspecified then the default 1911 port is assumed.

- file

The “file” scheme is used to identify files on the file system. All file ords resolve to instances of javax.baja.file.BIFile. File queries always parse into a FilePath. File ords include the following examples:

  - Authority Absolute: “//hostname/dir1/dir2”
  - Local Absolute: “/dir1/dir2”
  - Sys Absolute: “!defaults/system.properties”
    Sys absolute paths indicate files rooted under the Niagara installation directory identified via Sys.getBajaHome().
  - User Absolute: “^config.bog”
    User absolute paths are rooted under the user home directory identified via Sys.getUserHome(). In the case of station VMs, user home is the directory of the station database.
  - Relative: “myfile.txt”
  - Relative with Backup: “../myfile.txt”

- module

The “module” scheme is used to access BIFiles inside the module jar files. The module scheme uses the “file:” scheme's formatting where the authority name is the module name. Module queries can be relative also. If the query is local absolute then it is assumed to be relative to the current module. Module queries always parse into a FilePath:

  - module://icons/x16/file.png
  - module://baja/javax/baja/sys/BObject.bajadoc
  - module:/doc/index.html

- station

The “station” scheme is used to resolve the BComponentSpace of a station database.

- slot

The “slot” scheme is used to resolve a BValue within a BComplex by walking down a path of slot names. Slot queries always parse into a SlotPath.

- h

The “h” scheme is used to resolve a BComponent by its handle. Handles are unique String identifiers for BComponents within a BComponentSpace. Handles provide a way to persistently identify a component independent of any renames which modify a component's slot path.

- service

The “service” scheme is used to resolve a BComponent by its service type. The body of the query should be a type spec.

- spy

The “spy” scheme is used to navigate spy pages. The javax.baja.spy APIs provide a framework for making diagnostics information easily available.

- bql

The “bql” scheme is used to encapsulate a BQL query.

Types of space

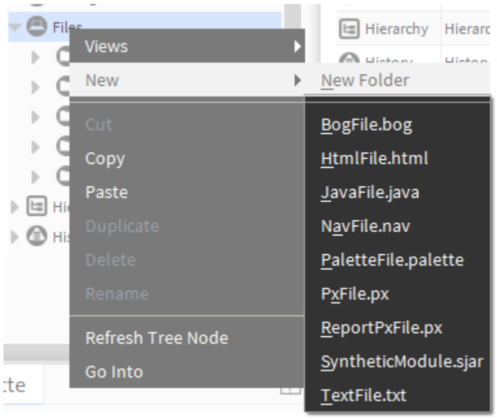
Space defines a group of objects that share common strategies for loading, caching, lifecycle, naming, and navigation. Following is a list of some of the different types of space:

- component
- file
- hierarchy
- history
- module
- orion
- station
- view

Types of files

In the file system, you may create and edit various types of files. Following is a list of some of the different types of files that reside in the file space:

Figure 20: Types of files available from the New menu



- BogFile.bog: Bog files are database files.
- HtmlFile.html: Html files are edited in the Text File Editor view and viewed in the Html View.
- JavaFile.java: Java files are edited in the Text File Editor view.
- NavFile.nav: Nav files are edited in the nav file editor and viewed in the nav tree. Refer the NiagaraAX Graphics Guide for more information about nav files.
- PaletteFile.palette: These files are custom collections of components that you create and save for viewing in the palette side bar.
- PxFile.px: Px files are edited in the Px view and are used to store graphic presentations that are available for viewing in the Px viewer and in a browser.
- ReportPxFile.px: The ReportPxFile is a regular Px file that has been “pre-loaded” with a ReportPane widget. The reporting functionality in helps you design, display, and deliver data to online views, web browser, and various other formats.
- TextFile.txt: Text files are edited and viewed in the text file editor.

About file naming

When working in a station, you often create various types of files, for example, a Px file if adding a new view, if creating a new Chart file, whenever exporting a view to pdf/txt/csv file, or if making a station backup .dist file. In addition, you often create new station file folders, and also copy graphic image files over to the station's file space.

Regardless of file types, whenever saving files (or before copying files to the station), it is strongly recommended that you restrict all characters in file and folder names to ones in the "original" set of ASCII characters in the BNF for Niagara file paths, namely:

a-z | A-Z | 0–9 | specials

specials= space | . | : | - | _ | $ | + | (|) | & | ‘ | ’ | @ | [|]

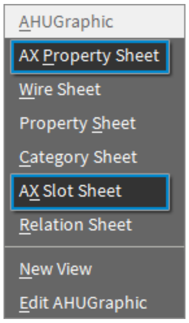
Otherwise, use of other characters, for example, a tilde (~) in file or folder names may result in obscure problems, particularly on JACE platforms (QNX-based hosts).

About views

There are many ways to visualize your system and its components. A "view" is a "visualization" of a component. One way to view a component is directly in the Nav tree side bar. In addition, you can right-click on an item and select one of its views to display in the view pane. You can see many different views of components. For example, a component that appears in the Nav tree may have a Wire Sheet view, a Property Sheet view, and a Px View, that all display in the view pane. Each component has a default view that appears whenever you activate a component (double-clicking, for example) without specifying a particular view.

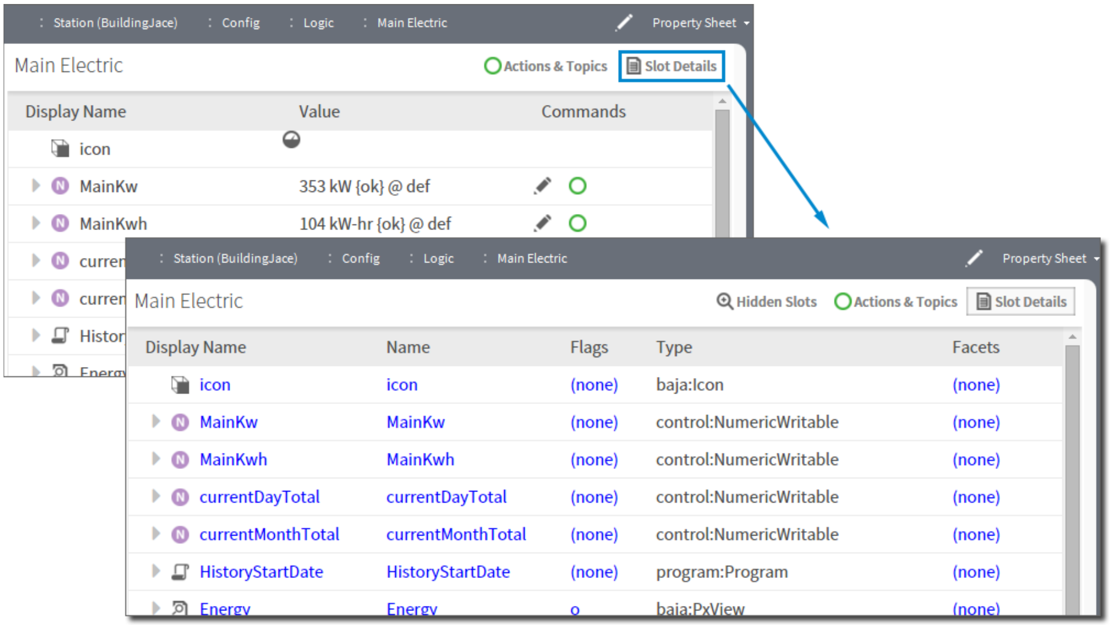
Two AX views are available under the Views selector menu and the right-click pop-up menu: the AX Property Sheet view, and AX Slot Sheet view. See the following figure.

Figure 21: AX views under Views selector



Optionally, you can use the Niagara 4 Property Sheet view which combines the functionality of the two views. When working in this view, simply click the Slot Details toggle command (upper right) to display and edit slot sheet details, as shown in the next figure. Enhanced functionality in this view also allows you to invoke actions and edit properties. Refer to the section "Plugin Guides" for a comprehensive list of views.

Figure 22: Property Sheet view with enhanced functionality



Display Name	Value	Commands
icon		
MainKw	353 kW {ok} @ def	
MainKwh	104 kW-hr {ok} @ def	
curren		
curren		
Histor		
Energ		

Display Name	Name	Flags	Type	Facets
icon	icon	(none)	baja:Icon	(none)
MainKw	MainKw	(none)	control:NumericWritable	(none)
MainKwh	MainKwh	(none)	control:NumericWritable	(none)
currentDayTotal	currentDayTotal	(none)	control:NumericWritable	(none)
currentMonthTotal	currentMonthTotal	(none)	control:NumericWritable	(none)
HistoryStartDate	HistoryStartDate	(none)	program:Program	(none)
Energy	Energy	o	baja:PxView	(none)

About lexicons

Niagara provides non-English language support by use of lexicons. Lexicons are identified by two-digit Java locale codes, such as "fr" (French) or "de" (German). All of the lexicons are distributed as modules (niagaraLexiconXx-rt.jar) included in the software installation.

Niagara 4.0 requires that custom lexicon files be compiled as a module (.jar).

Formats (BFormat)

A Baja Format (BFormat), is a class that returns object strings using a standardized formatting pattern language (script). A BFormat script consists of one or more calls chained together using the dot/period (.) operator. BFormat syntax requires that a percent (%) character begin and end each script. Like a wildcard or a variable, you embed BFormat script in static text strings. The system resolves the format (executes the calls contained in the embedded script) starting with the object that declares the format. Then the embedded calls (instructions) dynamically resolve to objects, and the system converts the final object into a string.

BFormats are very important for making templates (reusable portions of the data model Tree that require minimal configuration), when creating Px views, and to enable localization (foreign language support).

In Niagara, many properties that allow text entry support BFormat scripts. For example, a formula can refer to multiple points by using a Value Ord that contains BFormat script:
slot:/Drivers/NiagaraNetwork/%parent.name%/points/%name%

The system resolves the BFormat script in this example by substituting the name of the folder that contains the point for the %parent.name%, and the name of the point for %name%.

Call resolution sequence

BFormat scripts consist of one or more calls to methods that execute against system objects. Within a BFormat, the system resolves calls in this order:

1. Special calls:
    - time() returns the current time as a Niagara BAbsTime object
    - user() returns the current user’s name
    - lexicon(module:key) get the specified lexicon text
    - decodeFromString(module:type:escapedEncodedValue)
    - substring(to) on a string
    - substring(-fromEnd) on a string
    - substring(from,to) on a string
2. Java method get<call>(Context)
3. Java method get<call>()
4. Java method <call>()
5. Java method get(“<call>”)

Example scenarios

These example scenarios demonstrate how to configure text properties for which no default BFormat scripts exist.

Alarm extensions, history extensions, Px Widgets, and WeatherService provide good examples to illustrate the use of BFormat scripts.

BFormat example: naming points, VAV scenario

This example uses the BFormat %parent.parent% script to name points in alarm extensions.

This example involves a driver network of VAVs for 60 zones using 60 identical devices, each with identically named proxy points, but under a uniquely-named device component. For simplicity, assume that the devices are named: VAV1, VAV2, ...to VAV60.

Several proxy points in each zone require an alarm extension. You could manually rename the points for all 60 devices (a task that could take hours), or use BFormats to configure all VAV point names in a single operation.

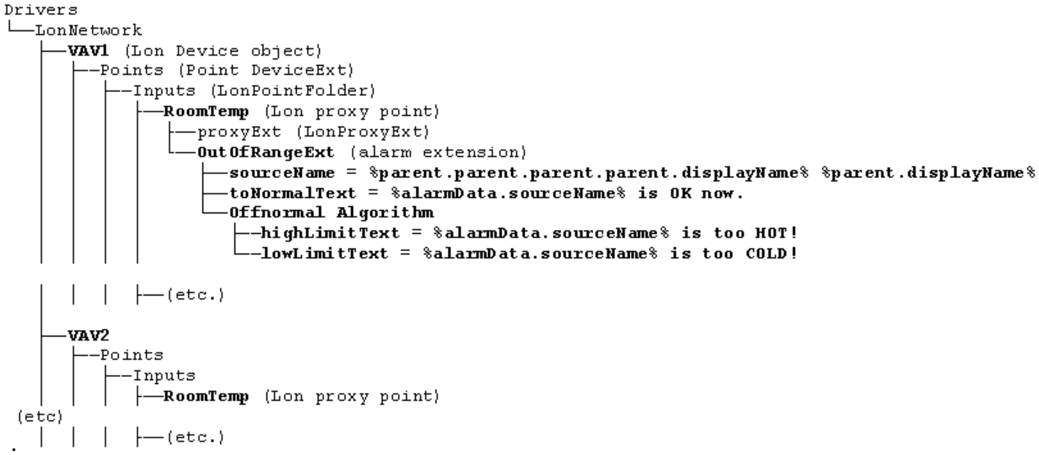
Manually renaming the points in each zone

This would involve entering unique names for all BFormat-capable properties under each alarm extension. In some cases, you may also have to modify related properties under the device's offNormalAlgorithm slot. Although not required, these changes would ensure that alarm records viewed in the alarm console provide unique source values (names) for each alarm. Without these unique names, only the station ords identify each VAV. For example, it can be difficult to isolate the zone and RoomTemp point for a specific alarm by looking for the point's ord.

Replicating properties using BFormats

This method will save you time. You begin by replacing the default values for several BFormat-capable (text) properties of one VAV's alarm extension such that generated alarms contain more useful source data, then you replicate the VAV application to the remaining 59 VAVs.

Figure 23: Example config structure and alarm extension property values using edited BFormat scripts



```
Drivers
└──LonNetwork
    ├──VAV1 (Lon Device object)
    │   ├─-Points (Point DeviceExt)
    │   │   ├─-Inputs (LonPointFolder)
    │   │   │   ├──RoomTemp (Lon proxy point)
    │   │   │   │   ├──proxyExt (LonProxyExt)
    │   │   │   │   └──OutOfRangeExt (alarm extension)
    │   │   │   │       ├──sourceName = %parent.parent.parent.parent.displayName% %parent.displayName%
    │   │   │   │       ├──toNormalText = %alarmData.sourceName% is OK now.
    │   │   │   │       └──Offnormal Algorithm
    │   │   │   │           ├─-highLimitText = %alarmData.sourceName% is too HOT!
    │   │   │   │           └─-lowLimitText = %alarmData.sourceName% is too COLD!
    │   │   │   │

    │   │   │   ├──(etc.)

    ├──VAV2
    │   ├─-Points
    │   │   ├─-Inputs
    │   │   │   ├──RoomTemp (Lon proxy point)
  (etc)
    │   │   │   ├──(etc.)
```

- VAV1, VAV2, etc. are Lon device objects.
- Points is the point device extension.
- Inputs is a Lon point folder.
- RoomTemp is a numeric Lon proxy point.
- proxyExt is a Lon proxy extension.
- OutOfRangeExt is a RoomTemp alarm extension.
- sourceName is a RoomTemp property. Instead of entering a value for this property on each of 60 property sheets, the following BFormat script resolves to a unique name for each VAV in the alarm console: %parent.parent.parent.parent.displayName% %parent,displayName%

This name contains two variable scripts separated by a space. The variable on the left resolves to four parent nodes in the tree structure to arrive at the device name (VAVn). The variable on the right resolves to the proxy point name, in this example, RoomTemp. In the alarm console the alarm source displays as VAVn RoomTemp, where n is a number between 1 and 60.

- toNormalText is a RoomTemp property that sets up a message when a point in alarm returns to normal. Instead of entering a value for this property on each of 60 property sheets, the following BFormat script customizes the message for each VAV:
  %alarmData.sourceName% is OK now.

This script resolves to the VAV name.

- OffnormalAlgorithm is the title of a group of properties on the property sheet.

- highLimitText is a RoomTemp property that sets up a message when the point returns a temperature above the defined limit:
  %alarmData.sourceName% is too HOT!

- lowLimitText is a RoomTemp property that sets up a message when the point returns a temperature below the defined limit.
  %alarmData.sourceName% is too COLD!

Note:

All the alarm text properties are relative to the alarm record component generated by an alarm, and not to the alarm extension responsible for generating the alarm. Alarm text properties in the alarm extension OffnormalAlgorithm include:

- toNormalText
- toOffNormalText

If the extension is an OutOfRangeExt, alarm text properties include:

- highLimitText
- lowLimitText

These OutOfRangeExt properties override any entry in the toOffNormalText property of the alarm extension parent.

The location of these text properties in the alarm record means that you cannot use the %parent.displayName% script for the alarm text properties, at least not if you expect to get any useful results. But because each alarm extension’s sourceName is now unique (using the technique above), you can reference it within alarm text type properties, along with any desired static text. Except here, the sourceName is an alarmData field, from the alarm record.

In the example, when RoomTemp in VAV1 triggers a high limit alarm, the alarm data message text displays:

VAV1 RoomTemp is too HOT!

When it returns to normal the alarm data message text displays:

VAV1 RoomTemp is OK now.

You could further modify the OffnormalAlgorithm high and low limit text properties to include the numerical (alarm) limit, using another alarmData field. For example, if highLimitText is set to this BFormat script: %alarmData.source% is above %alarmData.highLimit% degrees!, and the extension's highLimit is set to 74.5, upon a high limit alarm the message text displays VAV1 RoomTemp is above 74.5 degrees!. This technique may be useful if routing the alarm, for example, to a cellphone, where a minimum amount of alarm data text, including only the timestamp and the message text, are needed.

Note:

To see what alarmData fields are available for use in this manner, go to a station's alarm console and view the complete details (Alarm Record popup) for any one alarm.

BFormat example: naming histories

This example uses a technique other than the %parent.parent% script to name histories. This method may be called the folder-level-independent method, as explained in this topic.

History extension scenario

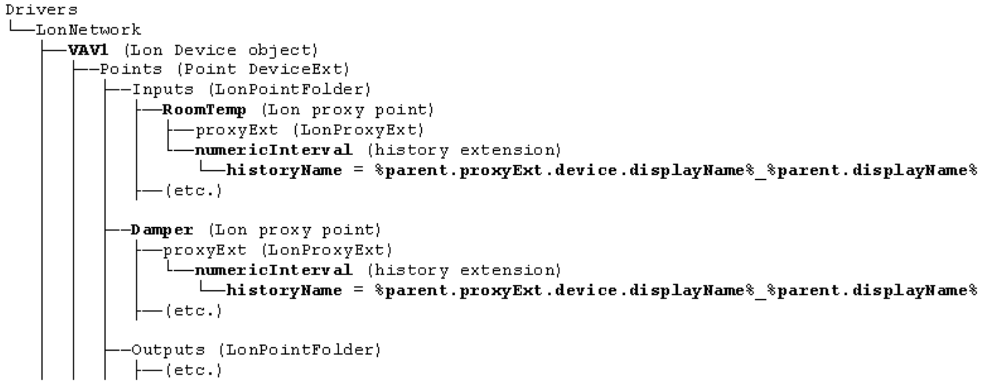
Consider a VAV network with replicated device applications. In each VAV zone, you need the histories of several proxy points, all identically named. For example:

- Each zone requires a numeric interval history on RoomTemp.
- Each zone requires a numeric interval history on Damper.

To manually configure these requirements you must either rename the parent point(s) or rename the history extensions' historyName to something unique, as duplicate history IDs are forbidden.

To automatically create unique histories names, before replicating this VAV application, you add a BFormat to the historyName in all history extensions, similar to editing the sourceName when naming points.

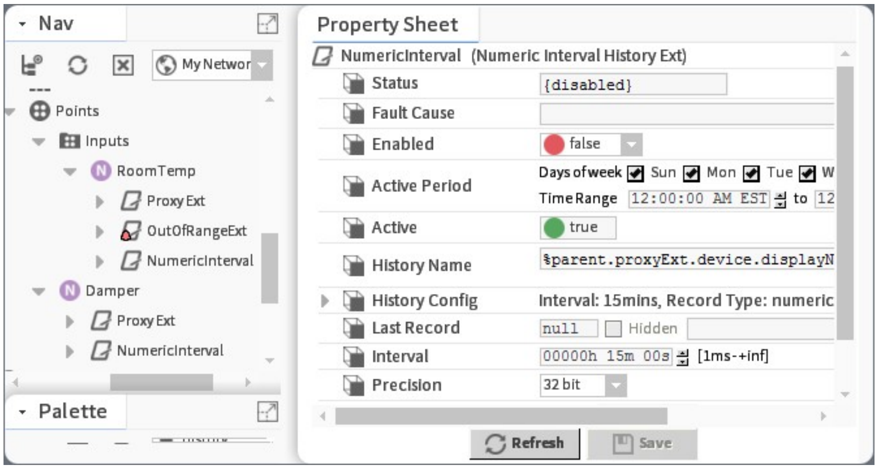
Figure 24: Example config structure and history extension property values using edited BFormat scripts



```
Drivers
└──LonNetwork
    ├──VAV1 (Lon Device object)
    │   ├──Points (Point DeviceExt)
    │   │   ├──Inputs (LonPointFolder)
    │   │   │   ├──RoomTemp (Lon proxy point)
    │   │   │   │   ├──proxyExt (LonProxyExt)
    │   │   │   │   └──numericInterval (history extension)
    │   │   │   │       └──historyName = %parent.proxyExt.device.displayName%_%parent.displayName%
    │   │   │   ├──(etc.)
    │   │   │
    │   │   ├──Damper (Lon proxy point)
    │   │   │   ├──proxyExt (LonProxyExt)
    │   │   │   │   └──numericInterval (history extension)
    │   │   │   │       └──historyName = %parent.proxyExt.device.displayName%_%parent.displayName%
    │   │   │   ├──(etc.)
    │   │   │
    │   │   ├──Outputs (LonPointFolder)
    │   │   │   ├──(etc.)
```

- VAV1 is a Lon device object.
- Points is the point device extension.
- Inputs is a Lon point folder.
- RoomTemp and Damper are numeric Lon proxy input points each with a proxyExt and a numericInterval history extension as child nodes.
- historyName is a numericInterval property on each history extension (double-click the numericInterval node to view its property sheet).

Figure 25: Lon device points showing a NumericInterval Property Sheet



NumericInterval (Numeric Interval History Ext)

Status	{disabled}
Fault Cause	
Enabled	false
Active Period	Days of week ☑ Sun ☑ Mon ☑ Tue ☑ W Time Range 12:00:00 AM EST to 12
Active	true
History Name	%parent.proxyExt.device.displayN
History Config	Interval: 15mins, Record Type: numeric
Last Record	null □ Hidden
Interval	00000h 15m 00s [1ms-+inf]
Precision	32 bit

Instead of entering a value for this property on each of 60 property sheets, the following BFormat script resolves to a unique name for each VAV:%parent.proxyExt.device.displayName%_%parent,displayName%

This name contains two variable scripts separated by an underscore.

- The variable script on the left of the underscore uses a special getDevice() method, where it resolves the proxy point's parent device name regardless of its folder depth under the Points extension. (In this example, proxy point RoomTemp is in an Inputs point subfolder, while the Damper proxy point is in the root of the Points extension).

Because the points are located in different folders, the %parent.parent% script would work for RoomTemp, but not for Damper. The folder-level-independent method, however, is more fault tolerant as a result of moving a proxy point, especially to change its hierarchy.

- The variable script to the right of the underscore resolves to the proxy point name (RoomTemp and Damper)

Given the tree structure of this network, the resulting histories appear as VAV1_RoomTemp, VAV1_Damper, and if replicated, VAV2_RoomTemp, VAV2_Damper, and so on.

Here is how this works:

%parent.proxyExt.device.displayName%, the parent, steps up one level to the Lon proxy point (say, RoomTemp).

The proxyExt is the slot name that walks back down the tree to a different child component, in this case to the LonProxyExt. The device calls the getDevice() method, and the displayName calls the getDisplayName() method.

This example assumes that no other components in the station also have a VAV component with a RoomTemp child, which also requires a history extension. Also, the example uses an underscore instead of a space even though spaces in object names are permitted (history being one type of object), they are escaped in the database using a "%20" string. This can be confusing in certain scenarios.

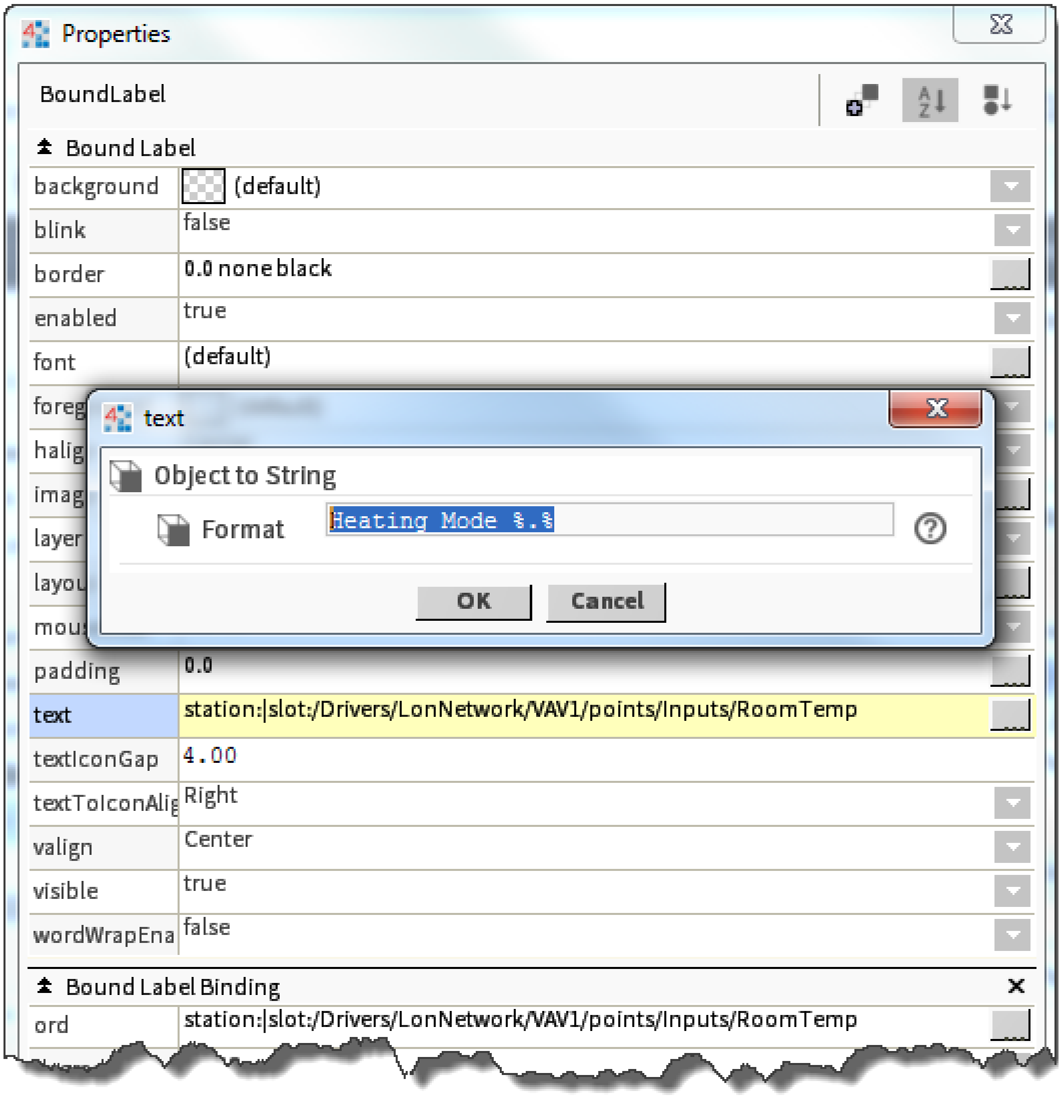
BFormat Px Widget examples

BFormat scripts can be used with Px widgets as demonstrated by this example.

Px widget scenarios

When you configure Px widget properties, especially for BoundLabel types with a binding to a component, double-clicking the Text property opens the Text window.

Figure 26: Example of adding static text in Text property of a BoundLabel



background	(default)
blink	false
border	0.0 none black
enabled	true
font	(default)
foreg	
halig	
imag	
layer	
layou	
mou	
padding	0.0
text	station:|slot:/Drivers/LonNetwork/VAV1/points/Inputs/RoomTemp
textIconGap	4.00
textToIconAlig	Right
valign	Center
visible	true
wordWrapEna	false

ord: station:|slot:/Drivers/LonNetwork/VAV1/points/Inputs/RoomTemp

Format: Heating Mode %.%

The ObjectToString, Format property determines the content of the displayed text. This property defaults to %.%, which displays the bound component's ord. In the example, a simple edit adds the words "Heating Mode:" in front of this default BFormat script.

For any point (or any component with an Out property), default text from the binding is identical to the out value displayed in the component's property sheet. The default text that is identical to the Out value includes facets, as well as the following information:

- If bound to a writable point, the Text Format contains three pieces of data, namely:

<value> <status> @priorityLevel>

where:

- <value> is
- <status> is {ok}, etc.
- @jpriorityLevel> is

For example: On {ok} @16 (a BooleanWritable) or 20% {ok} @12 (a NumericWritable)

- If bound to a read-only point, the Text Format provides two pieces of data, that is:

<value> <status>

For example:

Clean {ok} (a BooleanPoint) or 72.3 °F {ok} (a NumericPoint)

- If bound to a component that is not a point (there is no Out property), you must bind to a particular slot of that component, in order to display text other than its component type.

For example, if you drag a DegreeDays component to a Px page, the system displays the default text: Degree Days. However, if you change the binding's ord to <objectName>/clgDegDays, the system calculates and displays the cooling degree-days value (and status), such as: 5.0 {ok}

Text scripts for points

You can edit the Text Format property in any BoundLabel widget to include additional static text, and/or modify (or limit) the real-time data in the text.

The following table provides a few example BFormatting scripts and results for writable points.

Text (BFormat) script	Description	Example 1	Example 2 (script and result)
%out.value%	Value only (with facets).	On	AHU is %out.value% AHU is On.
%out.status%	Status including priority level, if writable point.	{ok} @ 16	Status of AHU is %out.status%. Status of AHU is {ok} @ 16.
%activeLevel%	Number only (1-16, def) for priority level, writable points only.	16	AHU is %out.value% at level %activeLevel%. AHU is On at level 16.
%status.flagsToString%	String value(s) for status flags set, without braces. If non: ok.	ok	AHU status is %status.flagsToString%. AHU status is ok.

The Example 1 column illustrates the resulting text if the Out script is On {ok} @ 16.

The Example 2 column shows the script and the resulting display when static text is added to the script.

More about text scripts

Object-to-String scripting is quite flexible when working with BoundLabel widgets. You are limited only by your understanding of Baja (see online Bajadoc in the Help system).

For the non-developer, these few simple rules may help:

- The BoundLabel widget must actually be bound to an object (using an ord). In other words, you cannot simply drag a BoundLabel from the kitPx palette to the Px page, edit the Text Format property, and get results. If needed, the BFormat script you use may be totally unrelated to the bound object. For example, you can bind to any object and enter a system-type call, as shown here:

  %time().toDateString% to produce text like “01-Nov-08”
- Relative to the bound object, you can use the parent technique to “walk up” the component tree of the text for a slot (or name), for example: %parent.parent.name% gives the name of the parent two levels up.

For example, a BoundLabel bound to a DiscreteTotalizerExt under a BooleanPoint, where you wish to display the (parent) point’s name and the number of times it has changed state since its last reset, could be achieved using this Text script:

%parent.displayName% had %changeOfStateCount% COS since last reset.

The result might be: ChWPump2 had 14 COS since last reset.

- In addition, relative to the bound object, you can also “walk down” the tree in a parallel path, using the slot name (versus name or displayName).

An example of this “walk down” method (via slot name) is in the history extension (historyName) example, along with the parent technique.

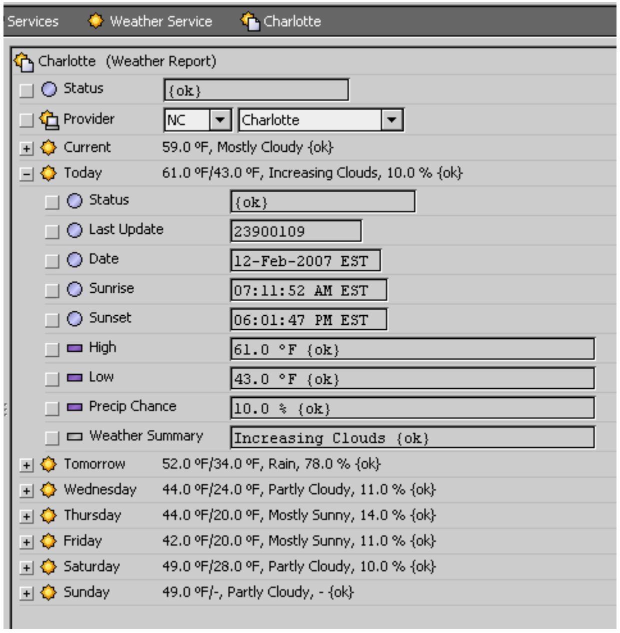
BFormat: WeatherService example

This example uses the WeatherService to show another way to use BFormat script.

An ord as a reference

The WeatherService can provide many pieces of information, including current conditions and forecasts.

Figure 27: Example Weather Report property sheet



Services › Weather Service › Charlotte

Charlotte (Weather Report)

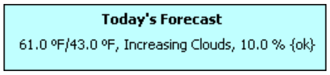
Status	{ok}
Provider	NC ▾ Charlotte ▾
+ Current	59.0 ºF, Mostly Cloudy {ok}
− Today	61.0 ºF/43.0 ºF, Increasing Clouds, 10.0 % {ok}
Status	{ok}
Last Update	23900109
Date	12-Feb-2007 EST
Sunrise	07:11:52 AM EST
Sunset	06:01:47 PM EST
High	61.0 °F {ok}
Low	43.0 °F {ok}
Precip Chance	10.0 % {ok}
Weather Summary	Increasing Clouds {ok}
+ Tomorrow	52.0 ºF/34.0 ºF, Rain, 78.0 % {ok}
+ Wednesday	44.0 ºF/24.0 ºF, Partly Cloudy, 11.0 % {ok}
+ Thursday	44.0 ºF/20.0 ºF, Mostly Sunny, 14.0 % {ok}
+ Friday	42.0 ºF/20.0 ºF, Mostly Sunny, 11.0 % {ok}
+ Saturday	49.0 ºF/28.0 ºF, Partly Cloudy, 10.0 % {ok}
+ Sunday	49.0 ºF/-, Partly Cloudy, - {ok}

To display this information on a graphic you create a bound label that references the WeatherService and the applicable property. For example, to display the forecast for today in Charlotte, NC, the referenced ord would be station:|slot:/Services/WeatherService/Charlotte/day0.

Default BFormat script

Instead of entering this ord, you could expand the WeatherService in the Nav tree to find this slot, then drag it to the Px page, where the Make Widget wizard automatically resolves to this ord. The default BFormat script of %.% for Text returns the information shown as the second line in the following image.

Figure 28: Default BoundLabel Text to Weather Report's Today property



Today's Forecast

61.0 ºF/43.0 ºF, Increasing Clouds, 10.0 % {ok}

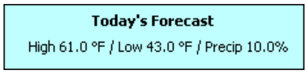
Modified BFormat script

To further control weather values, you can add additional BFormat scripts. For example, setting Text to the following:

High %High.value% °F / Low %Low.value% °F / Precip %PrecipChance.value%%%

returns the information shown in the second line of the following image.

Figure 29: Example of a modified BoundLabel Text to Weather Report's Today property



Today's Forecast

High 61.0 °F / Low 43.0 °F / Precip 10.0%

Note:

The system uses the percentage symbol (%) To delimit scripts in format Text fields. To display this symbol as text, enter two of them (%%).

BFormat errors

This topic covers a few example errors and considerations for dealing with errors.

Syntax errors

Not all attempts at customizing Format-type property values may be successful. If a syntax error causes the script to fail, an ERROR or err:<item>, where <item> is the script value appears in the produced text. For example:

- If you forget a % on BoundLabel Text entry, say: Fan is %out.value, the system displays: ERROR Fan is %out.value.
- Or, a script call to a misnamed slot might fail with a displayed error similar to: ChwPump2 had %err:control:DiscreteTotalizerExt:changeOfStates% COS since last reset.

You should test all modifications to BFormat scripts, to make sure you get the intended results.

Security breeches

To prevent a BFormat scripts from calling an object and, in the process, unintentionally changing the state of the object, Niagara maintains a BFormat blacklist of prohibited script call methods. By default, the methods listed here generate an error when any attempt is made to execute them via a BFormat:

- Any method that returns void. No exclusions are allowed.
- All synchronous action calls (that is, doAction()) calls that return a BValue . Exclusions are allowed.
- Any call to the submit (Context) method on any subclass of BJob.
- Any call to the clear () method on BDaySchedule and BEnumSetSehcdule.

BFormat default scripts

The system populates the text properties of some components (copied from palettes or originated from manager views) with default BFormat scripts. Other text properties default to empty.

The table below lists examples of the components with default scripts.

Component	Property	Default Script	Notes
Alarm extension for points, e.g. OutOfRangeExt, etc.	sourceName	%parent.displayName%	Suitable as is in many cases, such as where all parent points are uniquely named.
History extension for points, e. g. NumericInterval, etc.	historyName	%parent.name%	
Any network component's AlarmSourceInfo slot.	sourceName	%parent.displayName%	Often both properties work with these default values.
Any device-level component's AlarmSourceInfo slot.	sourceName	%parent.parent.displayName% %parent.displayName%	
EmailRecipient	subject	Niagara Alarm From %alarmData. sourceName%	The system provides much additional alarm data in the slot that contains the body of the email.

For a property value you can enter multiple BFormat scripts along with static text, as demonstrated by the defaults for a device's AlarmSourceInfo (sourceName) property in the table. A static space character separates the %parent.parent.displayName% from %parent.displayName%. The subject property of the EmailRecipient contains static text (Niagara Alarm From) ahead of the BFormat script (%alarmData. sourceName%).

BFormat scripts can save time by enabling the replication of applications, where you can achieve the desired result with minimal custom edits to property values. You may still custom format specific text as needed.

CHAPTER 2 ABOUT WORKBENCH

The following topics are covered in this chapter:

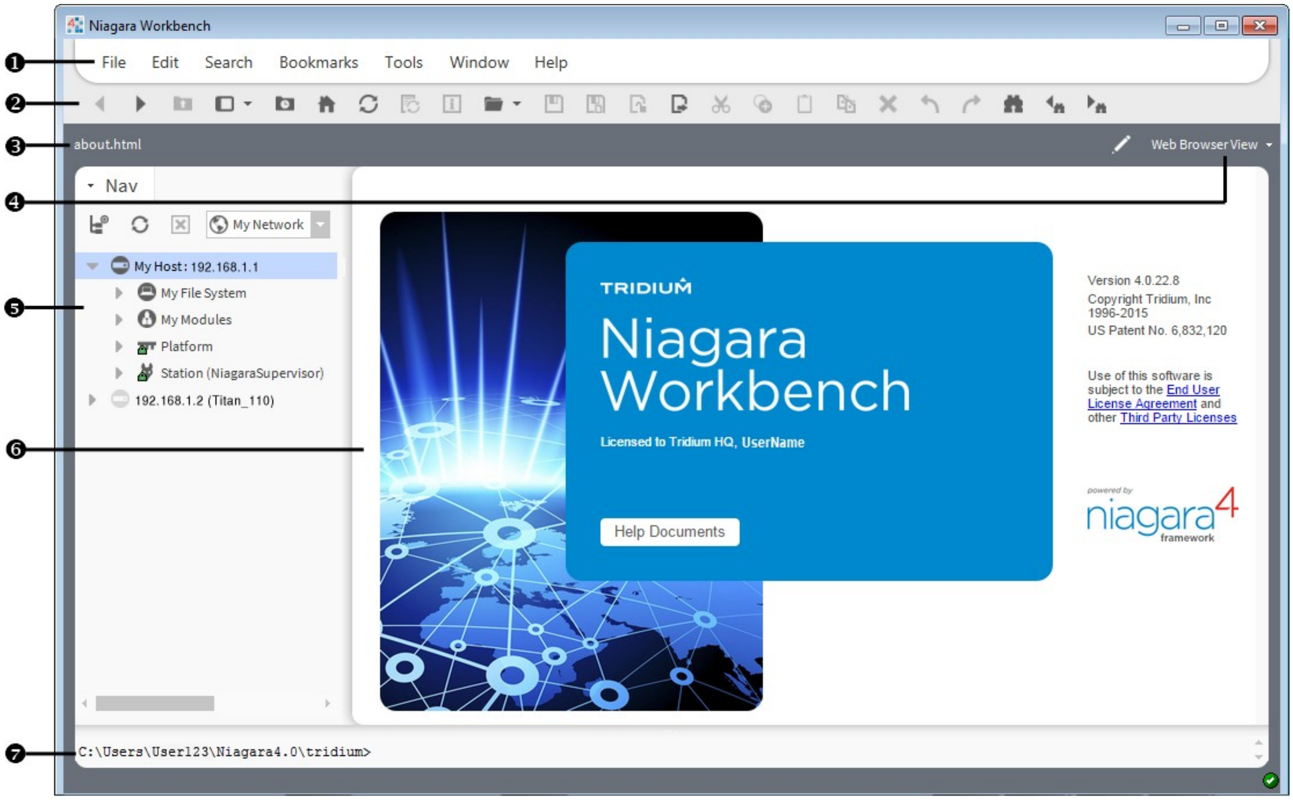
- Tour of the Workbench GUI
- Workbench window controls
- About the side bar panes
- About popup menus
- Table controls and options
- Controls and options for charts
- Customizing the Workbench environment

Workbench is the term for the Niagara graphical user interface. The following sections describe the general layout and many of the features of Workbench.

Tour of the Workbench GUI

When you start Workbench, the home window opens.

Figure 30: Workbench home window



Licensed to Tridium HQ, UserName

Version 4.0.22.8
Copyright Tridium, Inc
1996-2015
US Patent No. 6,832,120

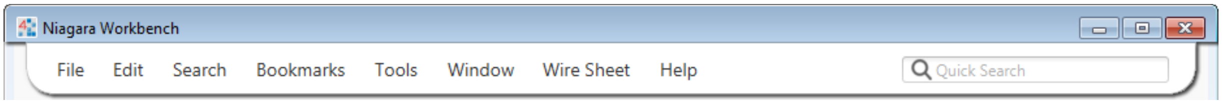
Use of this software is subject to the End User License Agreement and other Third Party Licenses

```
C:\Users\User123\Niagara4.0\tridium>
```

The home window is divided into seven areas:

❶ Menu bar — contains available program menus.

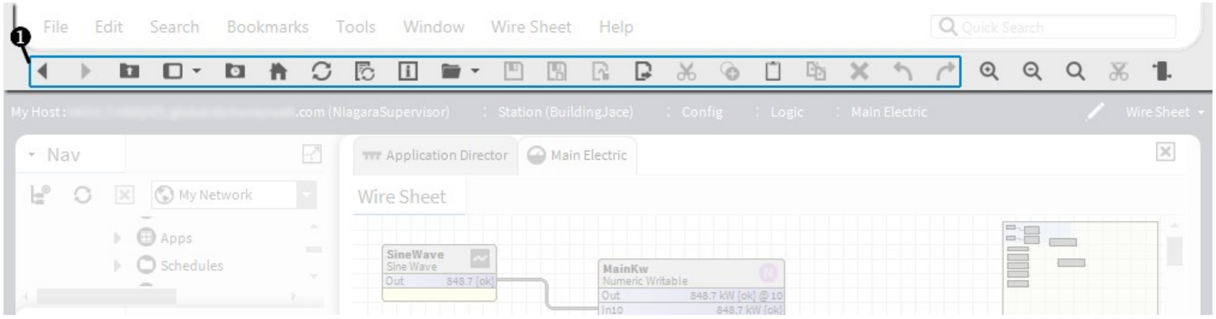
Figure 31: Menu bar



Many of the menus are context-sensitive and only appear when certain views are active. In a station connection, the Quick Search field displays on the right-side of the menu bar when the SearchService is installed on the connected station.

❷ Tool bar — contains icons for typical interaction with the interface plus icons specific to the view currently in use. Hovering the mouse pointer over an icon invokes a tool tip. The toolbar is the row of icons, just below the menu bar, that provides icons for actions affecting the objects that appear in the view pane. Usually, toolbar icons provide single–click access to many of the most commonly used features of the Workbench.

Figure 32: Toolbar

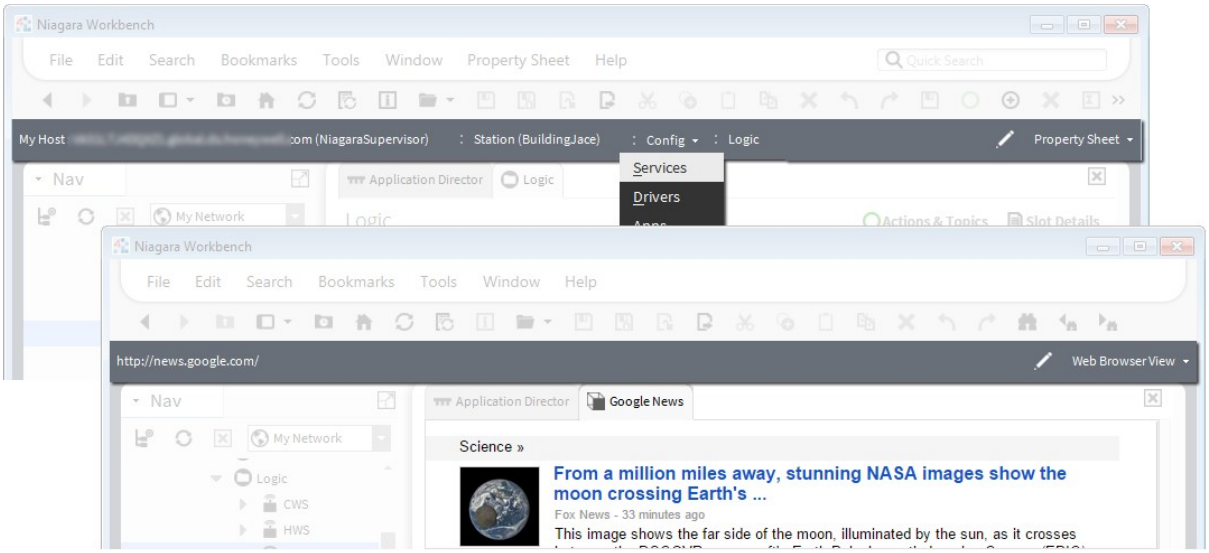


The primary icons (❶ in the above image) are always visible, Additional sets of icons are added to the toolbar when you select certain views. For example, when the Wire Sheet view is active, the Delete Links icon and Zoom icons are available.

When an icon is dimmed, it is unavailable. Hovering the mouse pointer over a toolbar icon invokes a popup description known as a “tool tip”.

❸ Path bar (locator bar) — located just below the toolbar, this area contains the path or Ord for the current view.

Figure 33: Path bar


Niagara Workbench
File Edit Search Bookmarks Tools Window Property Sheet Help
Quick Search
My Host [redacted] com (NiagaraSupervisor) : Station (BuildingJace) : Config : Logic
Property Sheet
Services
Drivers
Nav
My Network
Application Director
Logic
Actions & Topics
Slot Details
Niagara Workbench
File Edit Search Bookmarks Tools Window Help
http://news.google.com/
Web Browser View
Nav
My Network
Logic
CWS
HWS
Application Director
Google News
Science »
From a million miles away, stunning NASA images show the moon crossing Earth's ...
Fox News - 33 minutes ago
This image shows the far side of the moon, illuminated by the sun, as it crosses

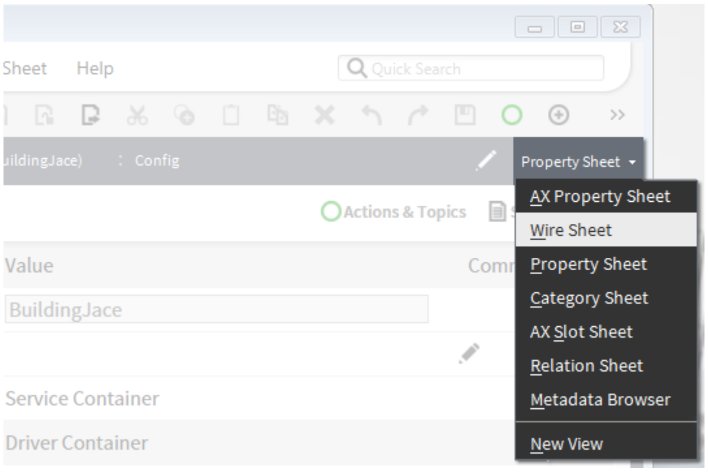
The left side of the path bar shows your current location (Ord or web address). The View selector appears on the right side.

The purpose of the path bar is to provide a graphical navigation field for selecting, displaying and entering destination references. The path bar serves several functions:

- It updates automatically each time you select a new view, so that it shows you the Ord of each view.
- The system displays an Ord in a graphical row of icons, so that when you hover the mouse pointer over an icon (or click on any icon) along that Ord, you can access any child node from the Ord's graphical drop-down list.
- The path bar functions much the same as a browser address field, permitting you to enter an Ord or a URL. Click the ✎ (Edit Path) icon to enter a different Ord or a web address (internal or external).

❹ View selector — a context sensitive menu with options that allow you to quickly display different views of the information that is currently in the view pane. This selector appears on the right side of the locator bar, just below the tool bar.

Figure 34: View selector



Due to our policy of continuous product innovation, some specifications may change without notification.

The options in the view selector differ, depending on the current view pane contents. For example, the view selector options that are available when you are viewing the platform in the view pane are different from the options that are available when you are displaying the Driver Manager view.

❺ Side bar pane — left-side area displays one or more side bars that you may select from the Windows menu. For example you might have the following open at the same time: Nav tree, Search side bar, and a module palette. For details, see the section, “About the side bar panes”.

❻ View pane — This pane, located on the right-side of the window, displays the currently selected view for the active tab. It is the largest display area below the locator bar. Features of the view pane include tabbed views and a thumbnail view.

You may add multiple tabs to the pane by using the tab feature. To change the selected view, do one of the following:

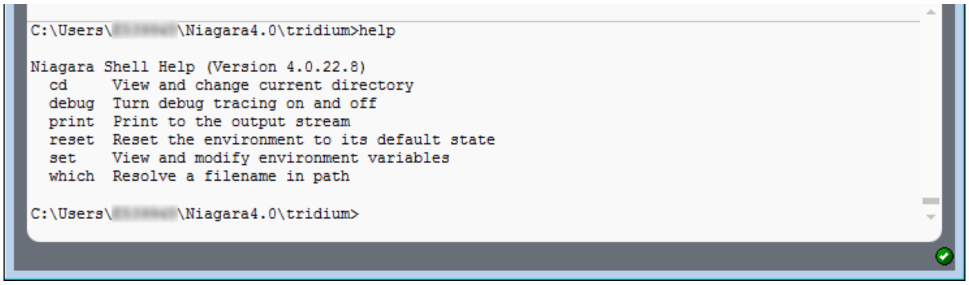
- Double-click on an item in the Nav tree.
- Select a view or action from a Nav tree palette popup menu.
- Select an option from a menu or submenu.
- Select an option from the locator bar.

The thumbnail view, when active, appears in the top right corner of the Wire Sheet window. It helps you find your way around the wire sheet.

❼ Console — bottom area provides access to a command line prompt without leaving the Workbench environment.

To hide or show the console, select Window→ Hide Console or Window→Console from the menu bar.

Figure 35: Console



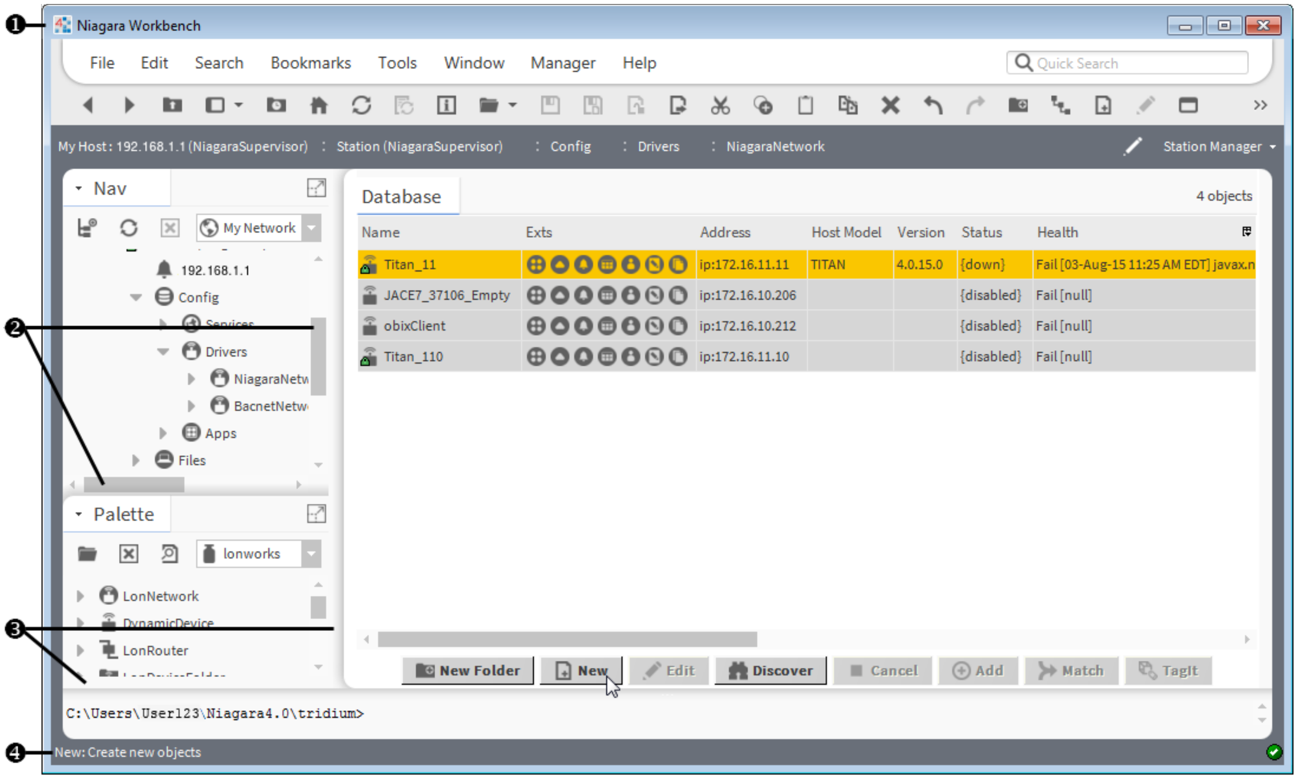
```
C:\Users\[redacted]\Niagara4.0\tridium>help

Niagara Shell Help (Version 4.0.22.8)
  cd     View and change current directory
  debug  Turn debug tracing on and off
  print  Print to the output stream
  reset  Reset the environment to its default state
  set    View and modify environment variables
  which  Resolve a filename in path

C:\Users\[redacted]\Niagara4.0\tridium>
```

The console has scroll bars on the right side and the window size may be adjusted by dragging the top border bar. From the console you may type in commands directly, including the help command for additional help.

Workbench window controls

The Workbench interface provides typical Windows-type controls plus other features unique to Niagara.

Figure 36: Window controls



Name	Exts	Address	Host Model	Version	Status	Health
Titan_11		ip:172.16.11.11	TITAN	4.0.15.0	{down}	Fail [03-Aug-15 11:25 AM EDT] javax.n
JACE7_37106_Empty		ip:172.16.10.206			{disabled}	Fail [null]
obixClient		ip:172.16.10.212			{disabled}	Fail [null]
Titan_110		ip:172.16.11.10			{disabled}	Fail [null]

```
C:\Users\User123\Niagara4.0\tridium>
```

New: Create new objects

❶ Title bar — has standard Windows title bar features, including windows name and icons to minimize, maximize, and close. Double-click the title bar to toggle between maximized and a sizable window.

You may create additional windows after starting Workbench. All have these basic features:

❷ Scroll bars — appear in window and window pane areas (side bar, view, or console) when some content portions are not visible. They are along the right and/or bottom portions of a window or pane.

Simply drag a scroll slider (highlighted area) to scroll quickly. Or, click an ending scroll arrow to move in incrementally.

❸ Border controls — as needed, drag any outside border to resize the entire window. Drag the inside border between the side bar and view areas, or (if shown) the console area to change their relative sizes.

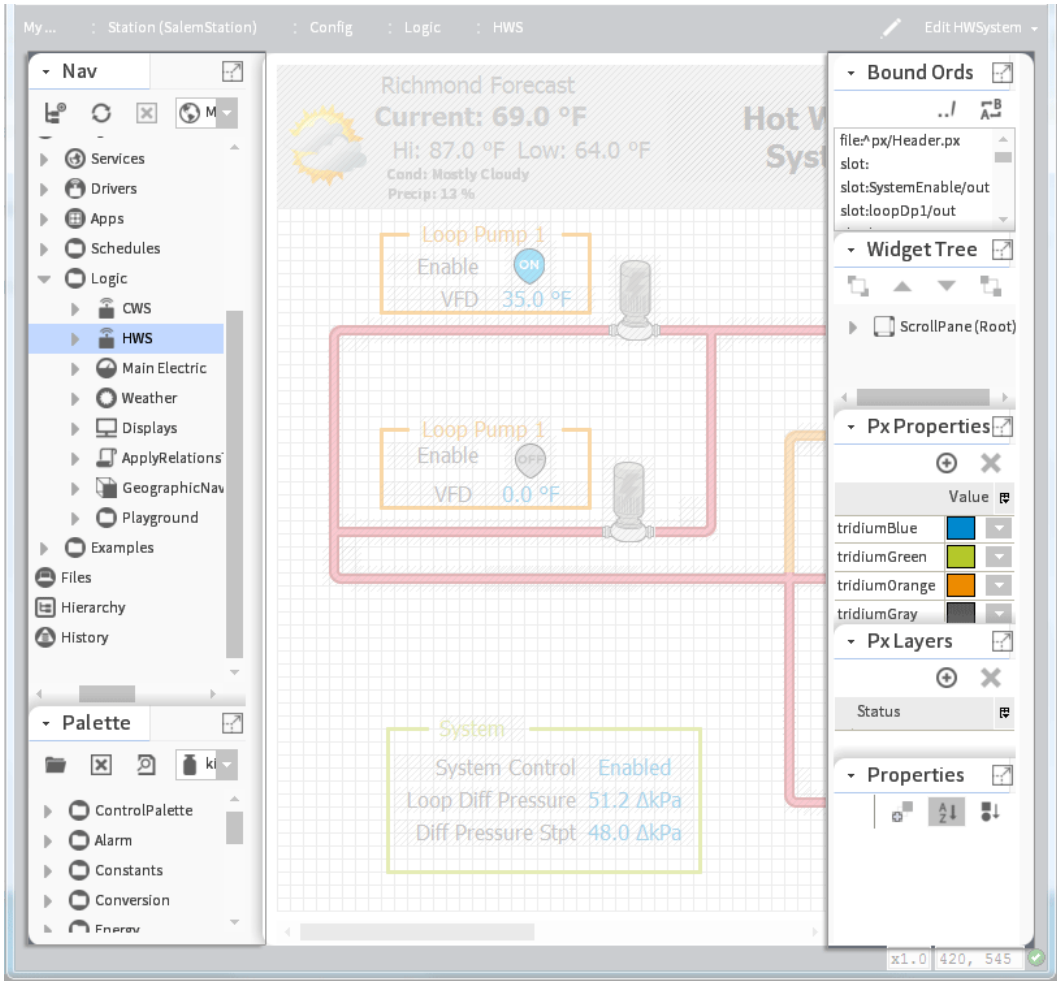
❹ Status bar — at the bottom left of the Workbench window is a status bar.

The status bar displays tool tips for icons in the toolbar and for buttons in views. When working in views other details are displayed in the status line as well.

About the side bar panes

Side bar panes are normally visible only if a side bar is open. All side bars display in the side bar pane and have some common features, such as the side bar title bar. When you close all side bars, the side bar pane collapses and the view pane and console pane (if open) expand to fill the window.

Figure 37: Side bar panes



Two types of side bar panes are the following:

- Primary side bar pane: The primary side bar pane is the first area on the left, below the locator bar.
- Px Editor side bar pane: This side bar pane appears on the right side of Workbench and is only available when the Px Editor is active.

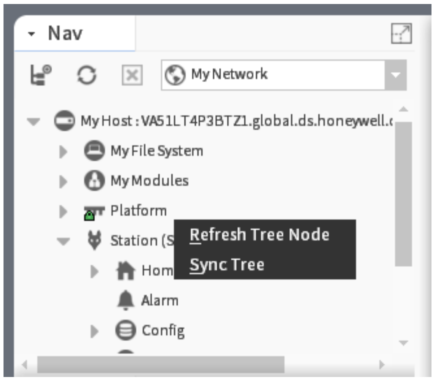
You use a popup menu to perform all operations that are available from inside the side bars (such as cutting or pasting). You use the icons in the title bar to open, close, and expand/restore the side bars. To resize side bar height and width, click and drag on the borders.

- For more information about the side bar title bar, see the section, “About the side bar title bar”.
- For more information about types of side bars see the section, “Types of side bars”.

About popup menus

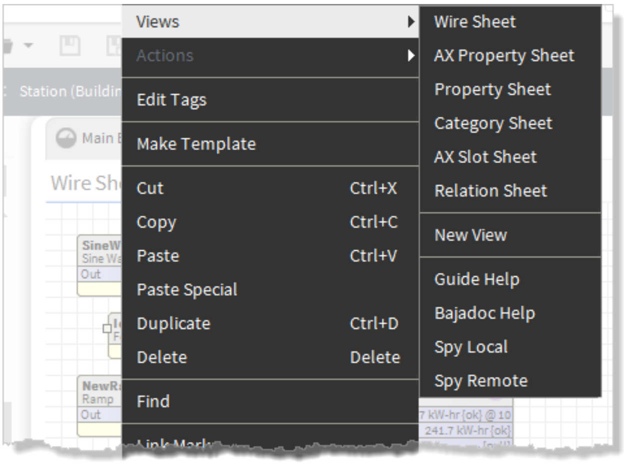
Workbench provides context-specific popup menus for controlling system options. Right-clicking on a component or a view opens these popup menus. The contents of each menu vary depending on the context.

Figure 38: Nav side bar popup menu with nothing selected



Right-clicking in the Nav tree side bar without selecting a component displays a very short (two-item) popup menu. Selecting a component in the Nav tree side bar displays a much longer popup menu.

Figure 39: Wire sheet popup menu



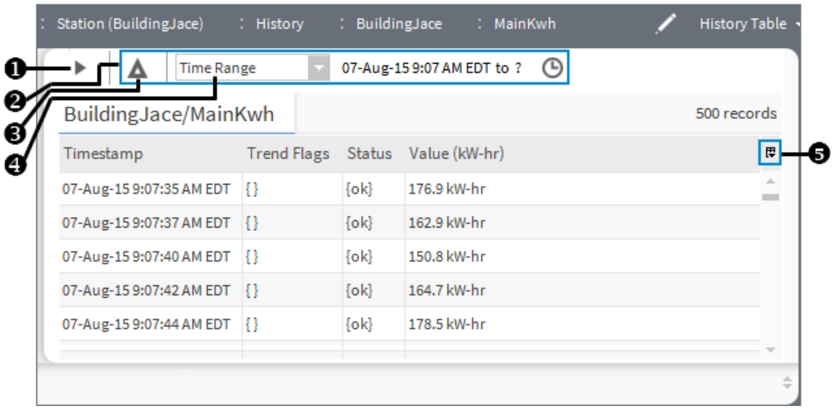
Right-clicking on the Wire sheet displays a popup menu with different options.

While most menu options are self-explanatory (Cut, Copy, Paste, etc.), all menu options are documented in the reference section of this guide.

Table controls and options

ManyWorkbench views present information in a table. All tables share similar features and controls.

Figure 40: Table controls and options



07-Aug-15 9:07 AM EDT to ?

BuildingJace/MainKwh 500 records

Timestamp	Trend Flags	Status	Value (kW-hr)
07-Aug-15 9:07:35 AM EDT	{}	{ok}	176.9 kW-hr
07-Aug-15 9:07:37 AM EDT	{}	{ok}	162.9 kW-hr
07-Aug-15 9:07:40 AM EDT	{}	{ok}	150.8 kW-hr
07-Aug-15 9:07:42 AM EDT	{}	{ok}	164.7 kW-hr
07-Aug-15 9:07:44 AM EDT	{}	{ok}	178.5 kW-hr

❶	Live Updates: The “play” icon, available for the History Table view, starts Live Updates (On Demand) updating. The icon changes to a “pause” icon while Live Updates is active.
❷	Data parameters: These controls include Delta (for history logging) and Time Range settings.
❸	Delta: Useful for history logging, displays value changes (delta) in your table.
❹	Time range: The dropdown option list has a variety of predefined time range options, including an option that allows you to restrict your data presentation to a particular date and time range that you specify.
	Title bar: Displays the name of the data collection on the left side of the title bar and in some tables (collection table, history table, alarm extension manager, and others) displays the total number of records in the table on the right side of the title bar.
	Column headings: Each column of data has a title that indicates the data type.
	Column boundaries: Each column has a movable column boundary that can be used to re-size the column using the mouse control. Stretch or shrink column width by dragging the column boundary, as desired. Use the Reset column widths menu item to reset all column widths to their default size.
❺	Table Options: Located in the upper right corner of the table, the dropdown list provides one or more of the following controls and options: • Reset column widths — sets all columns in the table to their default widths. This is useful if you manually changed widths of columns, and now some contents are hidden (even after scrolling). • Export — opens the Export dialog box where you can choose to export the table to PDF, text, HTML, or CSV (comma separated variable). • Context-sensitive menu items — additional context-sensitive menu items appear depending on the component that you are viewing.

Editing multiple rows in a table

Editing more than one table row at a time is sometimes called “batch editing,” or “batch processing.”

Step 1 Select the rows in the table to process using the Ctrl or Shift keys while you click the desired rows.

Step 2 To display the popup menu of available controls and actions, right-click.

Step 3 Select the control or action.

Step 4 If the system opens a window, make your selection and click OK.

Note:

Some actions, such as moving rows, or editing certain types of fields, are not appropriate for batch editing. In these cases, even though you can select multiple rows in the table, the action will be performed on only one (usually the top, or highest) selected record in the table.

Controls and options for charts

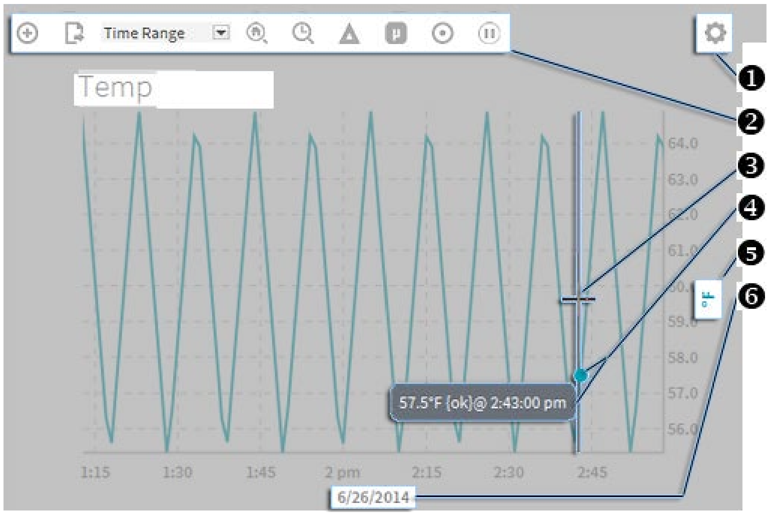
In Niagara 4 the Chart view is the default view for histories. While the History Chart view is a secondary view for legacy charts created an earlier release. Although the two views have a different look and feel, both offer many of the same controls and options.

Due to our policy of continuous product innovation, some specifications may change without notification.

Chart view

The Chart view (shown below) is the default view for History records in Workbench and in Hx, and a secondary view on schedules and Enum, Numeric, and Boolean points. Legacy charts, those created in earlier releases, are available as secondary History Chart views on History records.

Figure 41: Chart view description



❶ Settings icon — click to access chart Settings dialog

❷ Command bar — click icons to launch chart commands

❸ Cursor position indicator

❹ Data Value popup — displays when cursor is on a point

❺ Y-Axis label — default orientation of Y-axis for primary data

❻ X-Axis label — default orientation of X-axis. Once you have defined a specific Time Range for the chart, you can click this label to reopen the Time Range dialog to modify the range.

Data that can be rendered in a chart includes historical data, live historical data, live data, as well as schedules.

Although chart type is configurable via the Settings→Series tab dialog, the default chart type is determined by the type of data being presented. See examples in the table below.

Component type	Default chart type
Numeric histories and points	Render as lines with interpolation, display as a line chart.
Numeric schedules	Render as discrete lines with no interpolation, display as a line chart.
Boolean and Enum points	Render as shaded areas referred to as “swim lanes,” displayed as a shaded chart. Opacity of the swim lanes fill is based on the ordinal of the Enum.
Boolean and Enum schedules	Render as shaded areas referred to as “swim lanes,” displayed as a shaded chart. Opacity of the swim lanes fill is based on the ordinal of the Enum.

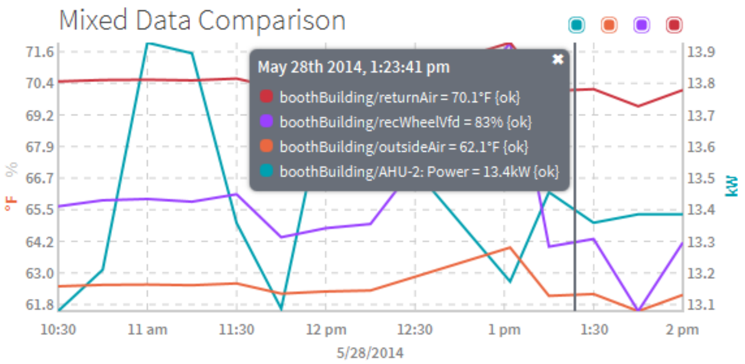
Different types of data (Numeric and Boolean or Enum) can be combined on the same chart. In that situation, the swim lanes representing Boolean and Enum data display with a dimmed opacity to allow you to more clearly view the lines representing the numeric data. Also, you can modify the default chart type of one or more components in a chart. For example, you can set a Boolean writable point to display bars while the data for another component plots a line.

The interactive Chart view allows you to make modifications while a chart is rendering. For example, while viewing a chart you can add one or more points, history records, schedules, or even containers of data. When adding data to a chart, the Y-axis automatically adjusts the units and can accommodate different units of measure by displaying multiple Y-axes. On a chart containing data with three or more different units of measure, such as that shown below, the display still shows dual Y-axes. You can switch the units displayed on the secondary Y-axis by clicking on the dimmed axis label.

For example, on the left-side Y-axis in the figure below, the dimmed % symbol indicates an alternate Y-axis with percent as the unit of measure. Clicking that % symbol switches the Y-axis units from displaying degrees to percent.

You can alternatively hide or show specific data or even completely remove data from a chart. Additionally, chart settings permit you to customize the appearance of a chart via selectable data colors and chart type per component, axis orientation, data source zooming, as well as permitting you to turn on or off the chart grid, background color, data popups, and status colors.

Figure 42: Multiple Y-axes accommodate data with different units of measure



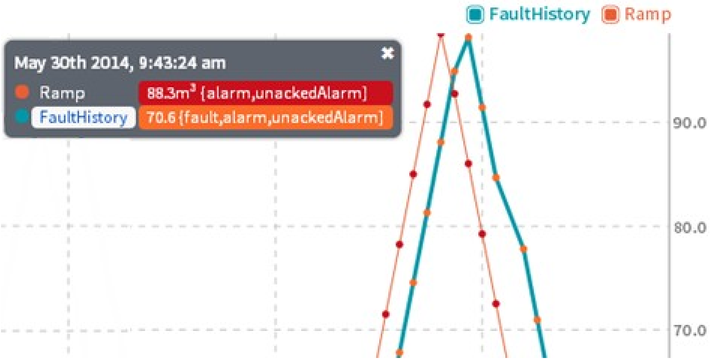
May 28th 2014, 1:23:41 pm

- boothBuilding/returnAir = 70.1°F {ok}
- boothBuilding/recWheelVfd = 83% {ok}
- boothBuilding/outsideAir = 62.1°F {ok}
- boothBuilding/AHU-2: Power = 13.4kW {ok}

Web charts utilize standard Niagara status colors to indicate current status. As shown in the chart below where the Status Coloring command is invoked, a red dot indicating Alarm status marks each plot in the Ramp line while an orange dot indicating Fault status marks each plot in the FaultHistory line. Also, status colors shown in the Fixed Data Popup dialog confirm the status of charted data.

Shade and Bar charts also display status colors. When enabled, if there is a non-ok status a color band at the top of the shaded area or bar indicates the status.

Figure 43: Line chart displaying status colors



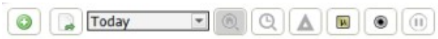
May 30th 2014, 9:43:24 am

Ramp	88.3m^3 {alarm,unackedAlarm}
FaultHistory	70.6 {fault,alarm,unackedAlarm}

Chart commands

Options in the Chart view Command Bar allow you to fine tune data presentation.

Figure 44: Command Bar



Most options in the Command Bar provide fine tuning for viewing purposes. Changes made with those options are of a temporary nature, not included when the chart is saved or exported. For example, if you turn on Time Zoom and Delta via buttons in the command bar and export the chart. When opened, the chart file displays default settings for those options. Exceptions are changes made with the Time Range, Sampling, and Status Coloring options, which are included on export or save.

Command Bar	Options	Description
Add Series		Add components to the chart. Select one or more components via File Chooser. Use Ctrl + Click to select multiple individual components or select a folder that contains multiple components.
Save		Available only when you open an existing chart file and make changes. Save — saves file (chart or csv format) to Station space: Files/charts/chartName.chart or Files/csv/chartName. csv
Export	Actions tab • File Name • Destination • File Type Options tab • View On Export • Status Column	Available in a new chart and when you open an existing chart file. • File Name — componentName (default) or type other name using normal file-naming conventions • Destination (Workbench) — StationExports file to station File space Files/charts/chartName.chart or Files/csv/chartName.csv • Destination (Web Browser) — – Download — exports file to operating system user's file space, for example: C: \Users\userName\Downloads\chartName.chart – Station— Exports file to station File space. For example, Files/charts/ chartName.chart – Print — launches browser Print dialog. Optionally, you can scroll to the bottom of the print dialog and click the link to "Print using system dialog" • File Type — chart (default) or csv • View On Export — Displays exported file • Status Columns — Available only for csv exports, exported file includes status data

Command Bar	Options	Description
 Time Range	• Auto (default) • Time Range • Today • Last 24 Hours • Yesterday • Week To Date • Last Week • Last 7 Days • Month To Date • Last Month • Year To Date • Last Year	Specifies time range for data display. Selecting the Time Range option launches a dialog where you can enter custom Start and End times for the range. Leave the End time field blank for live data to continue plotting on the chart.
Home Zoom	• On (default) • Off	Turns On/Off Home Zoom. On: Zooms to the X-axis of the primary data set. Note that if the primary data set is numeric, it zooms on the Y-axis.
Time Zoom	• On • Off (default)	Turns On/Off Time Zoom. On: Zooms X-axis to the time period specified by Time Range dropdown list. Off: Reverts to Home Zoom.
Delta	• On • Off (default)	Turns On/Off Delta On: Plots the rate of change between points. Off: Resumes plotting data points.
Sampling	• On • Off (default)	Turns On/Off Sampling. On: Sampling is enabled Off: Turns off sampling and disables autosampling behavior.
Status Coloring	• On • Off(default)	Turns On/Off data Status Coloring. On: Displays data points with status colors in a line chart and in shade or bar chart displays a status color band at the top of each bar. Off: Hides status coloring, data points/color bands.

Command Bar	Options	Description
⏸ Pause	• On / • Off (default)	Turns On/Off pause in live data plotting. On: Pauses live data plotting. No longer in live mode when paused. Off: Resumes live data plotting
■ Stop	• On / • Off (default)	Added in Niagara 41, this becomes visible only during data loading. Turns data chunking On/Off. On: Stops the data chunking process, halts data coming from the server. While stopped, the button displays a red border. Off: Reloads all of the data.

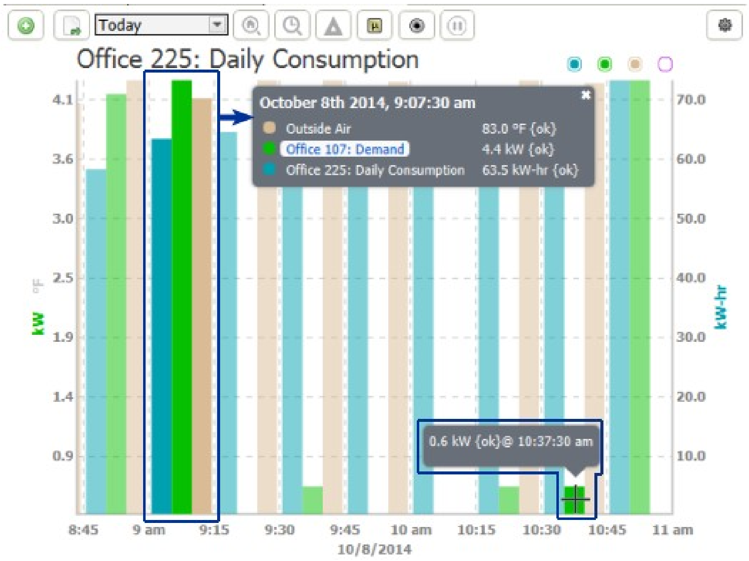
Chart settings

Options in the Chart view Settings dialog allow you to make data presentation changes that are of a persistent nature, meaning the changes are retained when the chart is exported or saved.

Series tab

Settings	Options	Description
Color	Color block assigned to each component	Change default data color by clicking color block and selecting different color via Color Picker.
Chart type	• Line • Discrete line • Shade • Bar	Line — plots a smooth line with interpolation. The default chart type for Numeric points and histories. Discrete line — plots a “stepped” line without interpolation. Shade — plots shaded areas, known as “swim lanes,” representing state change. The default chart type for Boolean and Enum points. Bar — plots vertical bars. Samples data into common intervals based on available width, When you have more than one component in a chart using bar chart type, they become a Bar Group, where the individual bars are adjacent (no space between). Clicking on a Bar Group selects the entire group and the values for all components in the group are shown in the Fixed Data Popup. While the mouseover Data Value Popup shows the value of a single component. See the figure below.

Figure 45: Daily Consumption



October 8th 2014, 9:07:30 am

Outside Air	83.0 °F {ok}
Office 107: Demand	4.4 kW {ok}
Office 225: Daily Consumption	63.5 kW-hr {ok}

0.6 kW {ok}@ 10:37:30 am

Axis tab

Settings	Options	Description
Y-Axis Orientation	- left - right (default)	Aligns Y-axis of primary data set to the left or right side of the chart.
Data Zoom Scope	- primary (default) - all	Sets the Data Zoom Scope to primary or all. Primary — zooms to the X-axis of the primary data set only. If the primary data set is numeric, it zooms on the Y-axis. All — changes the X-axis to accommodate all available data, including live data as it is recorded.
Show Grid	- true (default) - false	Turns on/off the chart grid. true — grid displays in chart behind data. false — grid does not display.
Background Area Color	- on - off (default)	Turns on/off the Background Area Color for the current theme. On — background area color displays in chart behind data. Off — background area color does not display.

Layers tab

Settings	Options	Description
Data Popup	- On (default)Displays - OffPauses	Enables/disables the Fixed Data popup. On — clicking on chart data displays the recorded date and time, as well as the name, value and status for each component in the chart at the point where you click. The persistent data popup remains visible until you close it.
Data Mouseover	- On (default) - Off	Enables/disables the mouseover Data Value popup. On — mouse position on chart data displays the recorded component value, status, and the time for that mouse position. Off — suspends display of mouseover data value popup.
Status Coloring	- On - Off (default)	Turns On/Off data status coloring. On — displays data points with status colors in a line chart and in a bar chart displays a status color band at the top of each bar. Off — hides status color data points/color bands.

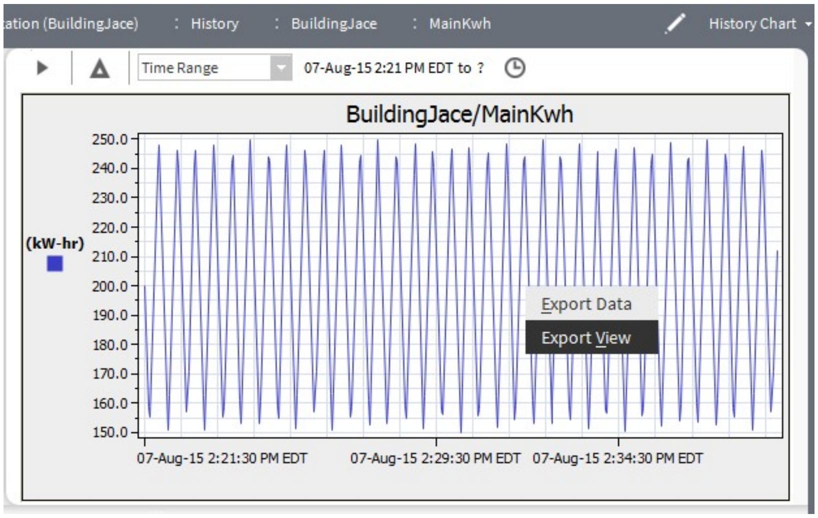
Sampling tab

Settings	Options	Description
Auto Sampling	• true (default) • false	Enables/disables automatic sampling optimizations. • true — automatically begins sampling if the focused data set exceeds 2500. • false — automatically stops sampling if the focused data set is below 2500.
Sampling	• true • false (default)	Enables/disables sampling for any size data set. true — turns on sampling false — turns off sampling ***Note:*** For performance reasons, sampling cannot be turned off once the focused data set exceeds 50,000. This threshold is configurable in the system.properties file.
Sampling Type	• average (default) • min • max • sum	average — samples average values for the selected rollup period. min — samples minimum values for the selected rollup period. max — samples maximum values for the selected rollup period. sum — samples the total of the values in the selected rollup period.
Sample Size	2500 (default)	Specifies the number of points in the data set to sample. Note that the default auto sampling size is configurable in the system.properties file.
Available Data Points	Read only	Displays the maximum number of points in the data set that are available to sample.
Sampling Period	Read only	Visible only once sampling has begun, displays the calculated average of the amount of time between each of the points that have been sampled.

History Chart view

Charts created in an AX release, are available as secondary History Chart views on history records. The History Chart view uses one or more of the controls and options described in the following list.

Figure 46: History Chart view controls and options



Data parameters

- ▲ Delta reporting option: This option is useful for history logging, when you want to display value changes (delta) in your report.
- Time Range ▾ Time range option list: This list has a variety of predefined time range options, including an option that allows you to restrict your data presentation to a particular date and time range that you specify.

▶ Live Updates

Click the icon to start Live (on demand) History plotting. The icon changes to a “pause” icon while Live History plotting is active.

Chart Title

This area of the chart displays the name of the chart. This title is editable in the chart builder view.

Y Axis

This displays units for the vertical axis.

X Axis

This displays units for the horizontal axis.

Charted Values

The color of the line and type of line is editable in the Chart Builder view.

Tool Tip

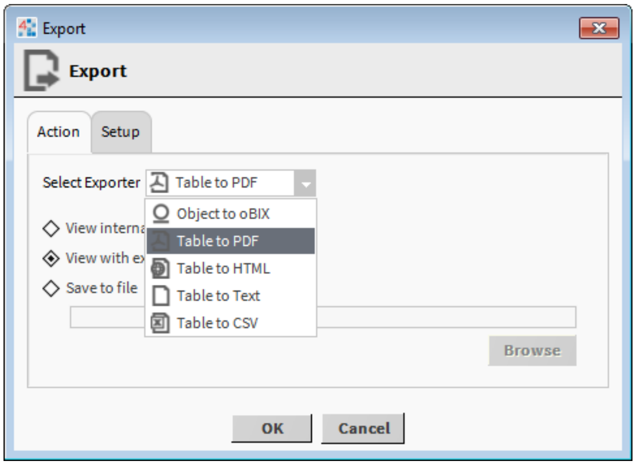
When you hover the mouse pointer over the history chart, a “tool tip” displays the date, time, and value of that location in the chart. The values are defined by chart axes and not the values of the actual data points.

Export popup menu

Right-clicking on the History Chart invokes the Export popup menu which contains two options:

Export Data opens the Export dialog box where you can choose to export the chart to oBIX, PDF, Text, HTML, or CSV (comma separated variable), as shown here.

Figure 47: Export Data options

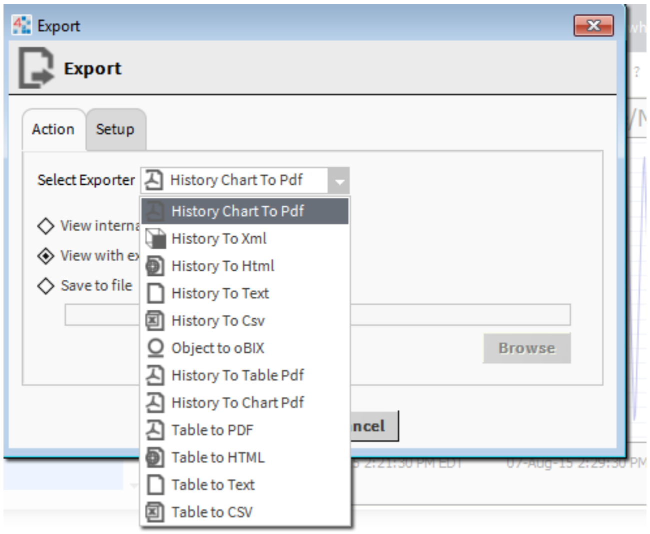


Note:

This export function works only with charts that are using a single history. Multiple histories do not export in a usable format.

Export View opens the Export dialog box where you can choose from a number of export options, as shown here.

Figure 48: Export View options



Customizing the Workbench environment

You can customize your Workbench interface as well as many of the settings in the Workbench environment. Some settings require an acknowledgement via the OK button, while others are invoked immediately and saved automatically on exiting Workbench. For example, if you exit Workbench with four tabbed windows in the view pane, Workbench displays the same four tabs the next time you open it.

Creating Additional Windows

Workbench allows you to create multiple fully functioning windows.

To create a duplicate of your current window, choose File:→New Window from the menu bar.

After you create multiple windows, you can then customize each individual window, as needed, to access different information, allowing you to see multiple concurrent views.

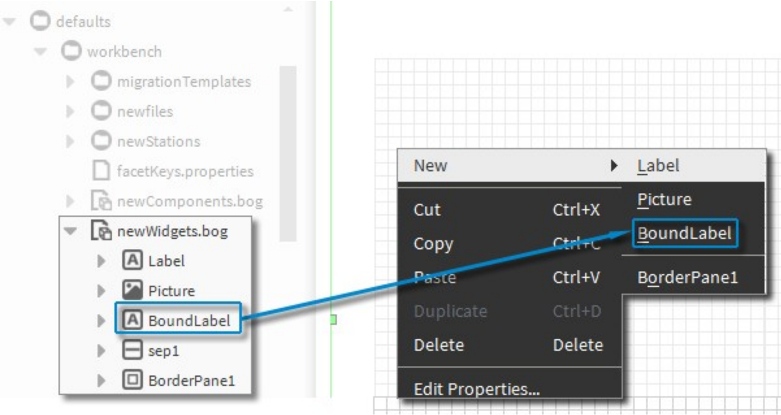
You can also copy and paste items from one window into another.

Editing options in the “new” popup menu

You can change the menu items that display on the context-sensitive popup menu by editing the files under the Workbench subfolder. For example, the following image shows the “newWidgets.bog” file that has been edited so that, when working in the PxEditor view, a “boundLabel” widget appears as one of the options in the popup menu under the New submenu.

Adding a widget to the newWidgets.bog file adds the item to the PxEditor New popup submenu.

Figure 49: PxEditor New popup submenu



Step 1 In the Nav tree under My File System, expand SysHome/ defaults/workbench/newWidgets. bog.

Step 2 Open the kitPx palette.

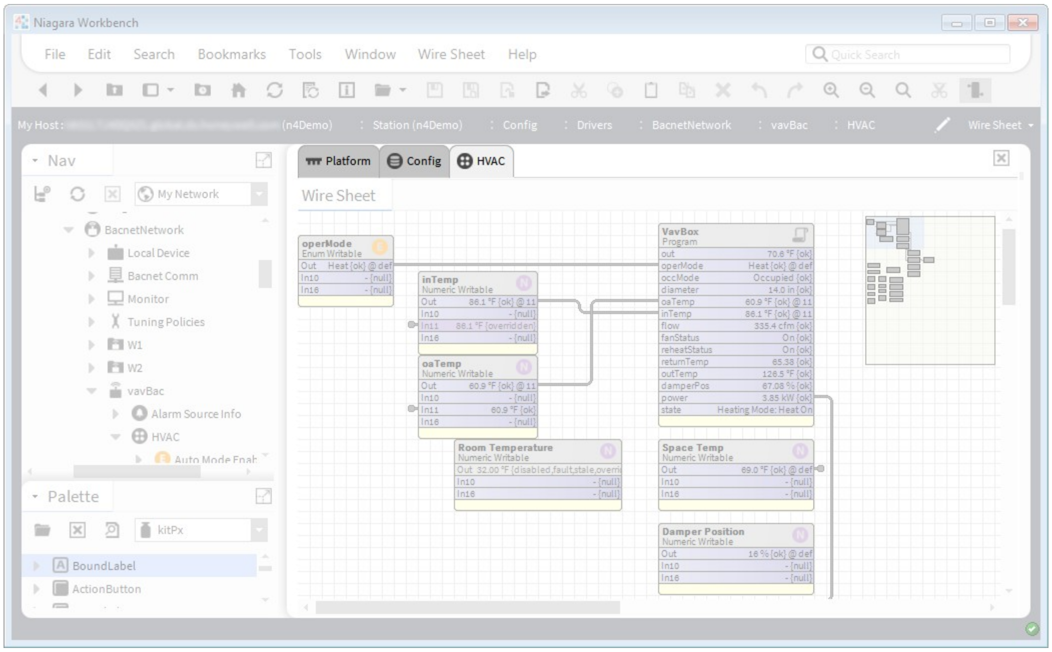
Step 3 Drag and drop the BoundLabel widget onto the newWidgets.bog file. The Bound Label option is immediately available in the New popup submenu.

Creating multiple tabs in the view pane

Workbench allows you to create multiple tabbed views within a single view pane.

As needed, you can use each tab to display a different view, component, or even host, all within the same window, as shown.

Figure 50: Multiple tabs in view



When you select a tab (make it active), the locator bar shows the current path and view. Also, the menu and tool bars update to show appropriate options for the current view.

Only one tab may be active at a time. In addition to simply clicking on a tab to make it active, you can select File → Next Tab from the menu bar to move to the next tab, moving left to right. Also, you can move to the last tab by choosing (from menu bar) File → Last Tab.

From the view of the active tab, you can copy items, select another tab, and then paste them into that view. You can also drag items into an active view from the (Nav or Palette) side bar.

Opening a new tab

You can use tabs to help organize and selectively display information in the view pane. Workbench allows you to create multiple tabbed views.

Step 1 Open a new tab in the view pane by any of the following methods:

- Use the keystroke combination: Click Ctrl + T.
- From the menu bar, select File→ New Tab.
- If tabs already exist, right-click on a tab and select New Tab from the popup menu. A new tab (identical to the previous one) opens in the view pane.

Closing a tab

Closing a tab removes the tab from the view pane.

Step 1 To close the active tab in the view pane, use any of the following methods:

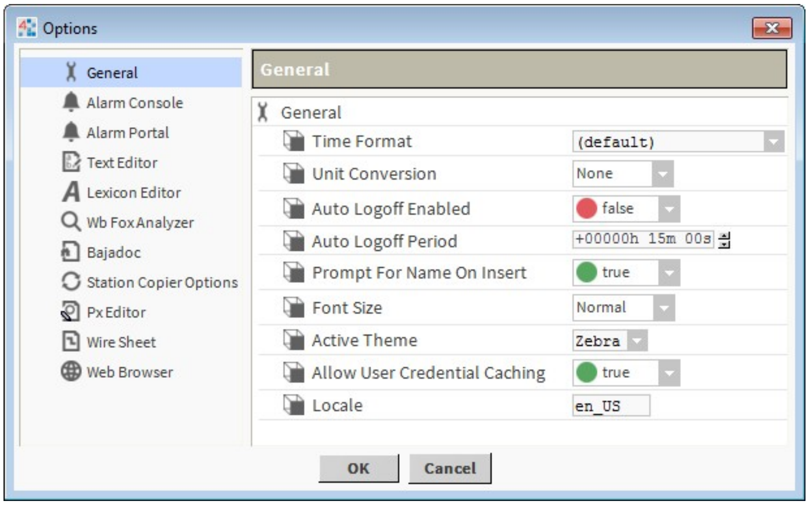
- Click ☒ (Close icon) located in the upper-right corner of the view, just below the View drop-down list.
- Right-click a tab and choose Close Tab to close that tab
- Right-click a tab and choose Close Other Tabs to close all tabs except that tab.
- From the menu bar, choose File→ Close Tab.

The currently active tab no longer appears.

Types of Workbench options

Use the Workbench options dialog box to customize your Workbench interface and to set other preferences. Open this dialog box by selecting Tools→ Options from the menu bar.

Figure 51: Options dialog box



Time Format	(default)
Unit Conversion	None
Auto Logoff Enabled	● false
Auto Logoff Period	+00000h 15m 00s
Prompt For Name On Insert	● true
Font Size	Normal
Active Theme	Zebra
Allow User Credential Caching	● true
Locale	en_US

The following options are available in the Options dialog box:

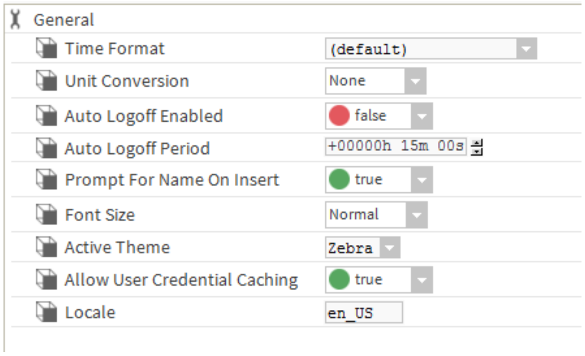
- General options
- Alarm Console options
- Alarm Portal options
- Text Editor options
- Lexicon Editor options
- Wb FoxAnalyzer options
- Bajadoc options
- Station Copier options
- Px Editor options
- Wire Sheet options
- Web Browser options

General options include the active Workbench theme.

General options

General options include settings for a variety of Workbench display and behavior options.

Figure 52: General Workbench options



General

Time Format	(default)
Unit Conversion	None
Auto Logoff Enabled	🔴 false
Auto Logoff Period	+00000h 15m 00s
Prompt For Name On Insert	🟢 true
Font Size	Normal
Active Theme	Zebra
Allow User Credential Caching	🟢 true
Locale	en_US

Note:

The Time Format and Unit Conversion parameters affect values that are displayed when connected to a station using Workbench—regardless of the User preferences (set under User Manager). The User preferences that are set under the User Manager are in effect when connected to a station by a browser.

- Time Format: Choose from a format option to set the way that time values are displayed by default.
- Unit Conversion: Choosing the English or Metric option converts values that are displayed in Workbench to the chosen unit type. Selecting None leaves units in the state that is assigned at the point facet.
- AutoLogoff Enabled: Setting this parameter to True will activate the AutoLogoff feature. When activated, the AutoLogoff feature will automatically log off a user from a platform when there is no activity detected in Workbench for the period specified in the AutoLogoff Period field. Setting this parameter to False will disable the Auto-Logoff feature.
- AutoLogoff Period: Time until Workbench logs off a user when AutoLogoffEnable is set to True.
- Prompt For Name On Insert: When set to True, Workbench displays a Name dialog, when a new item is added to the workspace.
- Font Size: Choose between Normal and Large font options for Workbench display.
- Active Theme: Choose between Zebra or Lucid “built-in” theme options for Workbench display. See the section, “About Workbench themes”.
- Allow User Credential Caching: If set to true (default), Workbench client access of a host (platform) or station allows a checkbox option to Remember these credentials in the popup Authentication dialog. If selected, the connection credentials are then cached and available in the Workbench Credentials Manager (Tools→ManageCredentials).

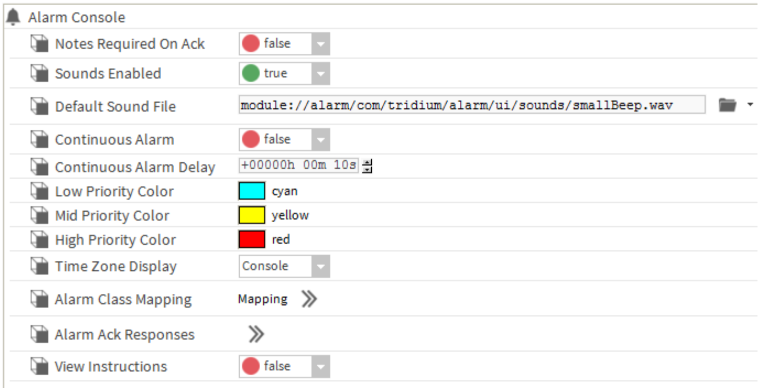
This is a convenience mechanism. However, for security best practices it is recommended to globally disable user credential caching by setting this property to false. This way that option is unavailable in the Authentication window. For related details, see the section, “Credentials manager”.

- Locale: Specifies the locale used by the Workbench VM, typically with a two-digit (ISO 639) code. Earlier the Workbench locale had to be specified in the !lib/system.properties file.

Alarm Console options

Alarm Console options allow you to customize both the appearance and behavior of the alarm console.

Figure 53: Alarm console options



Alarm Console

Notes Required On Ack	false
Sounds Enabled	true
Default Sound File	module://alarm/com/tridium/alarm/ui/sounds/smallBeep.wav
Continuous Alarm	false
Continuous Alarm Delay	+00000h 00m 10s
Low Priority Color	cyan
Mid Priority Color	yellow
High Priority Color	red
Time Zone Display	Console
Alarm Class Mapping	Mapping
Alarm Ack Responses	
View Instructions	false

Alarm console options include the following:

- Notes Required on Ack: This option, when set to true, causes the Notes dialog box to open whenever you initiate an alarm acknowledgement from the alarm console.
- Sounds Enabled: When set to true, causes a sound to accompany an alarm. You can also set this value under the Alarms menu in the Workbench main menu when the Alarm Console view is active.
- Default Sound File: Sets the path to the default sound file.
- Continuous Alarm: When set to true, causes an alarm to repeat continually, until it is acknowledged or cleared. This option works together with the Continuous Alarm Delay property. You can also set this value under the Alarms menu in the Workbench main menu when the Alarm Console view is active
- Continuous Alarm Delay: When Continuous Alarm is enabled, this property causes a “pause” time between in the continuous alarm sound. The continuous alarm is interrupted for a time equal to the value of this property.
- Alarm priority coloring: The following properties work together in combination to produce a range of colors that are assigned to the possible range of alarm priorities (1 - 255).
  - Low Priority Color
    Choose a color to designate a low priority alarm.
  - Mid Priority Color
    Choose a color to designate a mid priority alarm.
  - High Priority Color
    Choose a color to designate a high priority alarm.

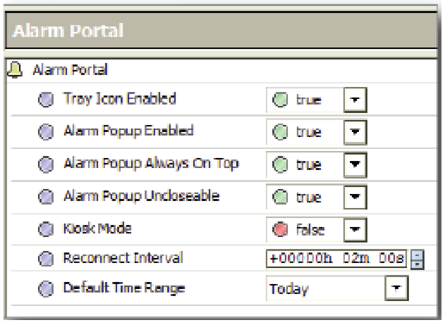
The highest priority (priority 1) is assigned the color of the High Priority Color setting. The Mid-Priority and Low Priority colors are assigned likewise, to Mid and Low Priority alarms. Alarm priorities that fall between these priority levels are assigned colors on a color-scale along a path defined by the three assigned colors.

- Time Zone Display: This property allows you to choose to display alarm record timestamp values in the time zone of the alarm console view (Console) or in the time zone of the alarm source (Source).
- Alarm Class Mapping: Provides a way for you to create alarm classes and map specific alarms to classes.
- Alarm Ack Responses: Use this property to create one or more text entries that you can use to populate the Notes dialog box when acknowledging an alarm. When the Notes Required on Ack is set to true, the Notes dialog box displays an additional option list containing any entries you create with this property. Use the button to open the associated Edit dialog box and add, edit, or remove response options, as desired. When the Notes Required on Ack is set to false, these Alarm Ack responses are not visible.
- View Instructions: When set to true, this property causes the alarm Instructions pane to display across the bottom of the Alarm Console. Instructions display in the pane for any single selected alarm that has associated instructions.

Alarm Portal options

Alarm Portal options allow you to customize both the appearance and behavior of the Portal Alarm Console.

Figure 54: Alarm portal options



Alarm Portal

Alarm Portal	
Tray Icon Enabled	true
Alarm Popup Enabled	true
Alarm Popup Always On Top	true
Alarm Popup Uncloseable	true
Kiosk Mode	false
Reconnect Interval	+00000h 02m 00s
Default Time Range	Today

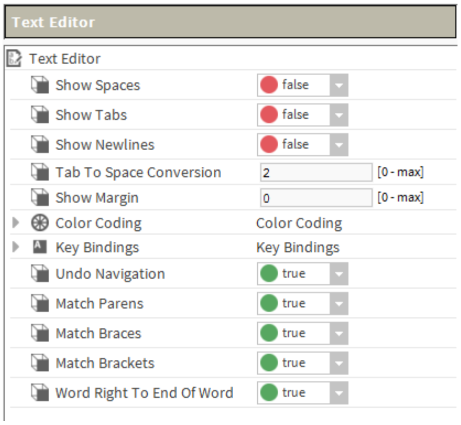
Alarm Portal options include the following:

- Tray Icon Enabled: When set to True, displays an alarm icon in the system tray when the alarm portal is active.
- Alarm Popup Enabled: When set to True, displays an alarm popup window when the alarm portal is active.
- Alarm Popup Always On Top: When set to True, causes the alarm popup window to stay on top of other windows when the alarm portal is active.
- Alarm Popup Uncloseable: When set to True, this makes the alarm popup window uncloseable if the alarm portal is active.
- Kiosk Mode: When set to true, the alarm portal opens in Kiosk Mode the next time it is started.
- Reconnect Interval: The alarm portal checks for “disconnected” alarm consoles. If a console is disconnected, a reconnect is attempted within the Reconnect Interval time.
- Default Time Range: Allows you to choose the default time range that displays in the Portal Alarm Console pane of the Alarm Console view (Console).

Text Editor options

These options offer a whole range of ways to customize the presentation and behavior characteristics of the text editor tool. Settings include text and symbol color coding options, as well as key bindings for shortcut keys. The text editor options properties are shown in the next figure.

Figure 55: Text editor options



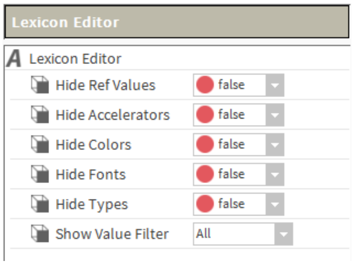
Text Editor

Text Editor		
Show Spaces	false	
Show Tabs	false	
Show Newlines	false	
Tab To Space Conversion	2	[0 - max]
Show Margin	0	[0 - max]
Color Coding	Color Coding	
Key Bindings	Key Bindings	
Undo Navigation	true	
Match Parens	true	
Match Braces	true	
Match Brackets	true	
Word Right To End Of Word	true	

Lexicon Editor options

These options offer ways to customize the default settings of the Lexicon Editor. The Lexicon Editor options are shown in the next figure, Lexicon Editor view. All of these properties may be overridden using the options that are available in the Lexicon Editor.

Figure 56: Lexicon editor options



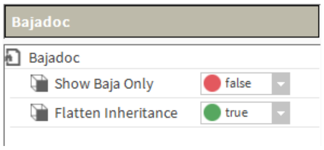
Lexicon Editor

Lexicon Editor
- Hide Ref Values ● false
- Hide Accelerators ● false
- Hide Colors ● false
- Hide Fonts ● false
- Hide Types ● false
- Show Value Filter All

Bajadoc options

Baja reference documentation includes both Java API details as well as Baja slot documentation.

Figure 57: Bajadoc options



Bajadoc

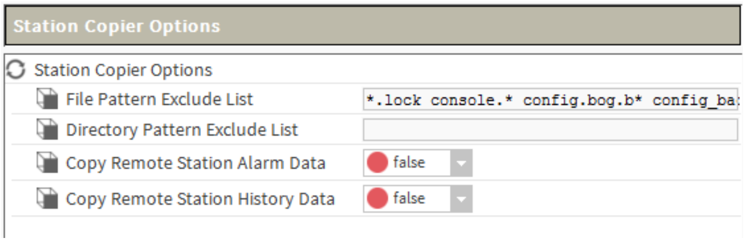
Bajadoc
- Show Baja Only ● false
- Flatten Inheritance ● true

- Show Baja Only: When set to True displays only the reference documentation for slots (Properties, Actions, and Topics). When set to False, documentation on the Java constructors, methods and fields is also displayed.
- Flatten Inheritance: This option allows you to flatten the inheritance hierarchy into a single set of documentation. When set to False only the Java members and Baja slots declared in the specified class are displayed. When set to True, all Java members and Baja slots inherited from super classes are also shown.

Station Copier options

This feature allows you to easily ignore critical data when copying stations to and from a platform. These properties allow you to configure station transfer options from a Workbench view. The properties include options to specify which directories and files to ignore when copying a station (by use of space delimited pattern filters). You can also use the property options to set default values for copying station alarm, history, job and critical data directories (hidden).

Figure 58: Station copier options



Station Copier Options

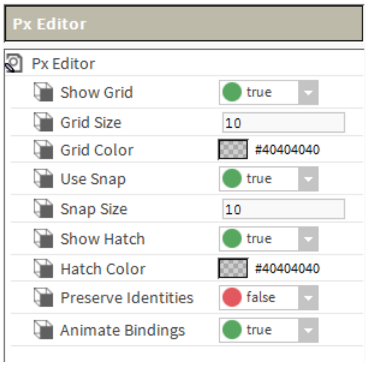
Station Copier Options	
File Pattern Exclude List	*.lock console.* config.bog.b* config_ba:
Directory Pattern Exclude List	
Copy Remote Station Alarm Data	● false
Copy Remote Station History Data	● false

- File Pattern Exclude List: Use this option to set a pattern filter that excludes any specified file types from being copied with the station.
- Directory Pattern Exclude List: Use this option to set a pattern filter that excludes any specified directories from being copied with the station. For example, if you wanted to exclude copying all directories in the station that begin with the letters “lighting”, then you could type “lighting*” in this field.
- Copy options: (Remote Station Alarm Data, Remote Station History Data): These properties allow you to choose to copy (true) or not copy (false) certain station data.

Px Editor options

These options offer ways to customize the presentation and behavior characteristics of the Px Editor view. The Px Editor options are shown and described in the following list:

Figure 59: Px editor options



Px Editor

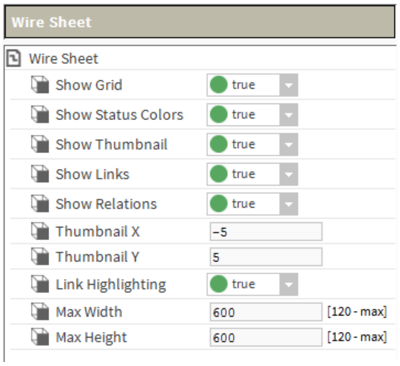
Px Editor	
Show Grid	● true
Grid Size	10
Grid Color	▨ #40404040
Use Snap	● true
Snap Size	10
Show Hatch	● true
Hatch Color	▨ #40404040
Preserve Identities	● false
Animate Bindings	● true

- Show Grid: This property sets the default condition of the Px editor grid. Select true to make the grid visible by default or select false to make the grid hidden by default. Either setting may be changed at any time using the PxEditor menu.
- Grid Size: This property sets the size of the grid in the Px editor.
- Grid Color: This property sets the color of the grid in the Px editor. Click in the color field to display the Color Choose dialog box. Use the Color Chooser to set the color that you want to assign to the grid.
- Use Snap: This property sets the default condition of the Snap feature in the Px editor. Select true to make objects snap to locations when they are at a distance equal to the Snap Size. Select false to disable the snap feature. Either setting may be changed at any time from the PxEditor menu.
- Snap Size: Set an integer value in this field to define the interval between successive snaps.
- Show Hatch: This property sets the default condition of the Px editor hatching that displays on objects on the Px editor canvas. Select true to make the hatching visible by default or select false to make the hatching hidden by default. Either setting may be changed at any time using the PxEditor menu.
- Hatch Color: This property sets the color of the hatching in the Px editor. Click in the color field to display the Color Choose dialog box. Use the Color Chooser to set the color that you want to assign to the Px editor hatching.
- Preserve Identities: When set to true, this property allows you to explicitly turn on support for encoding all names and handles on a Px page.
- Animate Bindings: This option, true by default, allows the Px Editor to display live binding data for widgets in Px Edit mode. If you set this option to false, then no data animation occurs in the Edit mode, although animation does occur, as expected, in View mode.

Wire Sheet options

Wire sheet options allow you to customize the appearance of the Wire Sheet view.

Figure 60: Wire sheet options



Wire Sheet

Wire Sheet

Show Grid	● true	
Show Status Colors	● true	
Show Thumbnail	● true	
Show Links	● true	
Show Relations	● true	
Thumbnail X	-5	
Thumbnail Y	5	
Link Highlighting	● true	
Max Width	600	[120 - max]
Max Height	600	[120 - max]

Wire sheet options include the following:

- Show Grid: When set to True displays a grid background on the wire sheet view.
- Show Status Colors: When set to True, this displays a different color for each status when it appears in the wire sheet.
- Show Thumbnail: When set to True, this provides a small “thumbnail” view of the whole wire sheet for orientation and navigation purposes.
- Thumbnail X: A field for setting the X axis for the default position of the thumbnail view.
- Thumbnail Y: A field is provided here for setting the Y axis for the default position of the thumbnail view.
- Link Highlighting: When set to True, this turns on link highlighting.
- Max Width: Use this property to specify a maximum size for the wire sheet width.
- Max Height: Use this property to specify a maximum size for the wire sheet height. For more details, see the section, “Link navigation”.

Web Browser options

This option allows you to customize the web browser by enabling or disabling the Web Development Tools.

About Workbench themes

Workbench themes options allow you to customize the appearance of the Workbench display. Generally speaking, a theme is a predefined collection of design elements that determine the way an application appears to the user. Within the Niagara Framework, a theme is a module that controls the appearance of Workbench on the local computer by defining fonts and colors, as well as the icons, used in the display. Choosing a different theme affects only the Workbench appearance, there is no other impact.

You can customize your Workbench display by selecting the theme that you prefer.

Types of themes

Workbench provides standard built-in themes as part of the default application environment.

Built-in themes

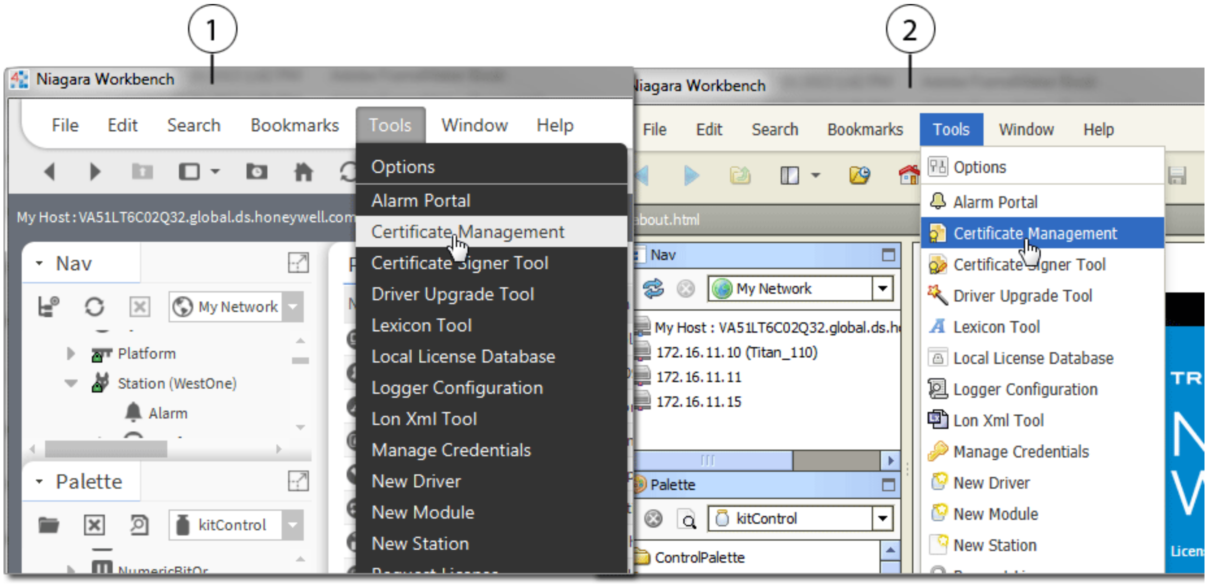
Following are three Workbench themes listed and illustrated below:

1. Distech: Standard theme that has look and feel common to other Distech Controls software products.

2. Zebra: This theme has a low contrast, grayscale coloring and is the default theme.

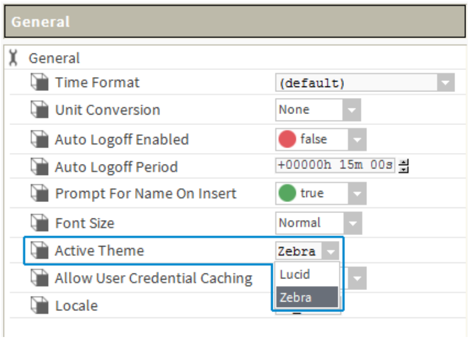
3. Lucid: Displays a blue and gray color scheme.

Figure 61: Workbench Differences between built-in themes: Zebra (left) and Lucid (right)



To change the Workbench theme, click Tools→Options to see the active theme in the General tab. Click the Active theme drop-down arrow and click on a different theme option to select it as shown above. In order for the theme change to take effect, you must exit Workbench and restart it.

Figure 62: Active Theme drop-down in General settings



General

General

Time Format	(default)
Unit Conversion	None
Auto Logoff Enabled	false
Auto Logoff Period	+00000h 15m 00s
Prompt For Name On Insert	true
Font Size	Normal
Active Theme	Zebra Lucid Zebra
Allow User Credential Caching	
Locale	

Note:

To save changes made in Workbench Options, such as a theme change, you must close/exit Workbench using File menu options or click the window’s close box. Closing Workbench in the console will not cause those changes to be saved.

CHAPTER 3 DATA AND CONTROL MODEL

The following topics are covered in this chapter:

- Wire Sheet object management
- About control points
- About point extensions
- About control triggers
- About point status
- About writable points
- About composites

In any Niagara station (Supervisor or controller) all real-time data are normalized within the station database as points, a special group of components. Each point represents one data item.

Wire Sheet object management

An object on a Wire Sheet view of a data model is a point component that represents a piece of data.

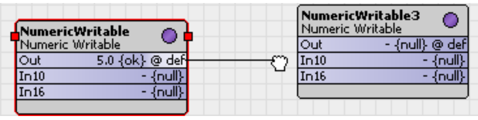
Basic object linking

Object links define the direction of data flow.

Step 1 Move the mouse over the In or Out node of a wire sheet object (glyph) until the area is highlighted and the mouse pointer changes to a white arrow.

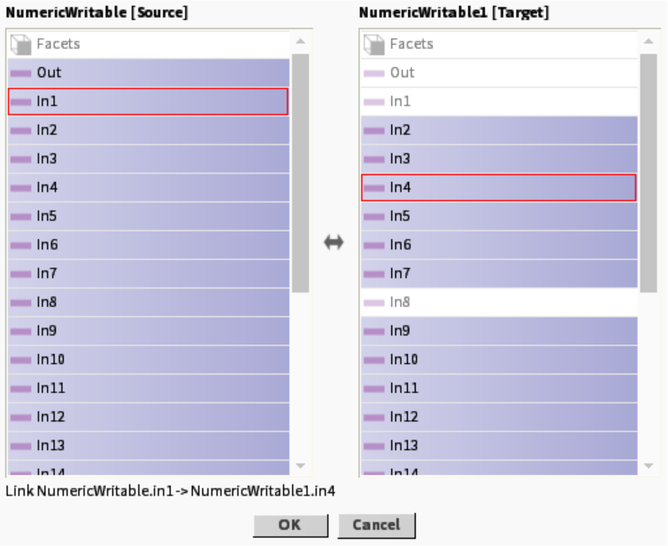
Step 2 Do one of the following, as shown in the next figure.

Figure 63: Basic object linking



- To complete a link, click and drag from the point of origin to the desired In or Out slot of the target object and release the mouse button.
- Drag a wire from the selected object to a vacant slot on another wire sheet object.

Figure 64: Basic object linking



Link NumericWritable.in1 -> NumericWritable1.in4

The Link window opens.

Step 3 To create the link using the Link window, select the desired slots and click OK.

To identify the objects (slots), their names display at the top of their respective columns. The object in the left column is the source object; the object in the right column is the target object.

You cannot select invalid (dimmed) slots when dragging from an In to an Out slot or when using this window.

Red rectangles surrounding each selection indicate the link.

A summary of the currently selected slots displays above the OK and Cancel buttons.

Continuous object linking

You can maintain a continuous link state as you drag connection wires to link to multiple target objects from a single source object. This allows you to link from a single source to as many targets as you want without having to click on the source object to re-initiate each link.

Step 1 Hold the Shift key any time you release the mouse button

If the mouse pointer is over a valid object node, a link is established or the Link dialog box opens to complete the link. This continued link state is indicated by the target end of the wire “sticking” to and moving with the mouse pointer.

Step 2 To deactivate the link state, release the Shift key and click anywhere or touch any key.

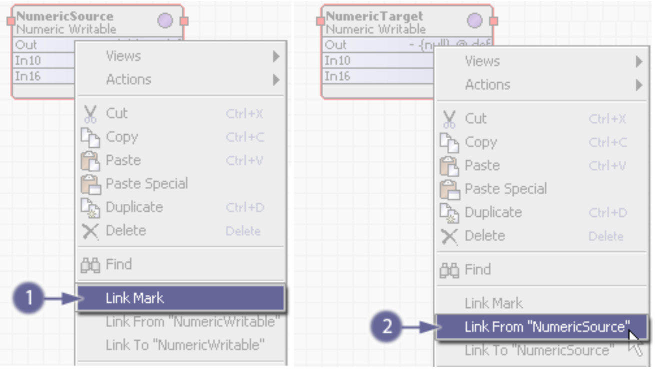
Creating multiple links at the same time

Link Mark is a feature that allows you to perform one-to-many, many-to-one, and many-to-many linking in a single operation.

Step 1 To define one or more selected objects as a link source or target, select the Link Mark command.

This command specifies that the selected objects are to be one side of the link operation. The names of the “marked” objects display as part of the Link Mark or Link From command in the popup menu.

Figure 65: Link Mark or Link From command



Step 2 To define one or more selected objects as a link source or link target, select the Link From command.

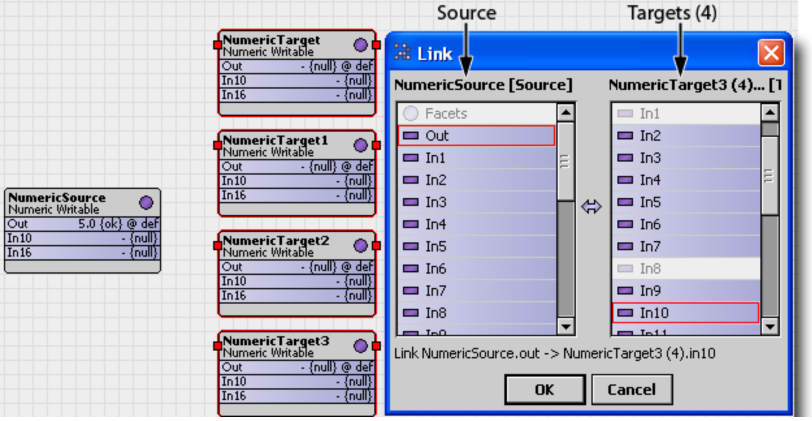
This command opens the Link dialog box with the “marked” object as the source object. You can still change source and target roles in the dialog box using the Reverse button ⇔.

Step 3 To define one or more selected objects as a link source or link target, select the Link To command.

This command opens the Link window with the “marked” object as the target object.

You can still change source and target roles in the dialog box using the Reverse button ⇔.

Figure 66: Source and target roles



Link NumericSource.out -> NumericTarget3 (4).in10

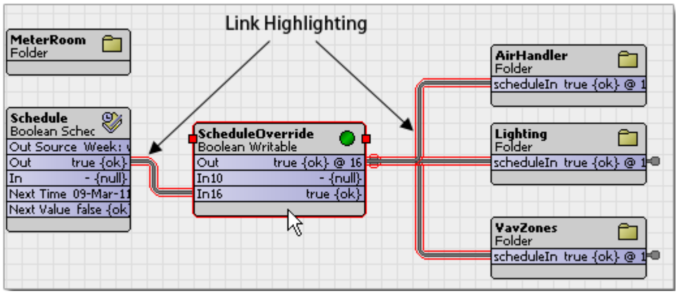
Viewing links

The link highlighting option (enabled by default) is available on the wire sheet. Link highlighting makes it easier to distinguish links on a crowded wire sheet.

Step 1 Select a component on the Wire Sheet.

All links associated with the selected object display with a colored outline around them.

Figure 67: Link highlighting



Note:

Highlighted links do not mean that the LINK is selected.

Step 2 Hold the shift key and select another component.

Subsequent link highlights display a different color highlighting (the system uses up to 20 colors before repeating).

Step 3 To customize the colors, edit the system.properties file (in the Nav tree lib folder, under MyFiles→ Sys Home).

You specify each color using the standard hexadecimal notation used for HTML color display.

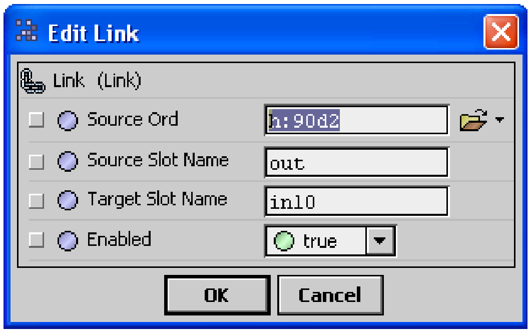
Editing links

You can edit a link directly from the wire sheet view using an Edit Link window, without having to go to the source component's Link Sheet view.

Step 1 To open the Edit Link window, right-click a single link (not multiple links or knob links) and select Edit Link from the popup menu. The Edit Link dialog box displays, as shown below.

Step 2 Use the property fields to change any of the following link property values: Source Ord, Source Slot Name, Target Slot Name, or Enabled and click OK.

Figure 68: Edit Link window.


Edit Link

Link (Link)

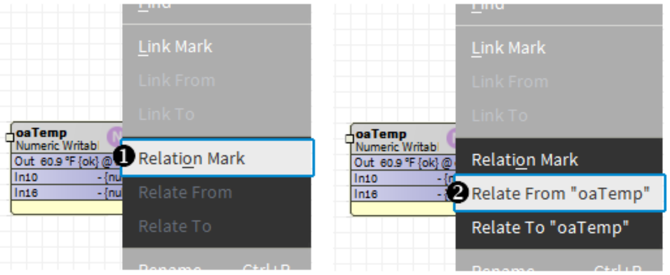
□ ○ Source Ord	h:90d2
□ ○ Source Slot Name	out
□ ○ Target Slot Name	in10
□ ○ Enabled	○ true

Organizing objects in a hierarchy

You add relations between objects for purposes of building hierarchies. Relation links are directional and tag-based. They indicate how an object relates to another object. Relation linking functions in the same manner as object linking.

Step 1 To define one or more selected objects as a relation source or relation target, select the Relate Mark command.

Figure 69: Relate Mark command



Step 2 Select the Relate From command to define one or more selected objects as a relation source or relation target.

This command opens the Relation dialog with the “marked” object as the source object. Select a tag from the dropdown list.

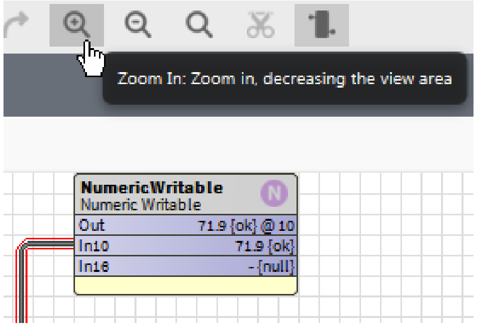
Step 3 Select the Relate To command to define one or more selected objects as a relation source or relation target.

This command opens the Relation dialog box with the “marked” object as the target object.

Zoom controls

Complex wire sheets can be difficult to manage. The zoom controls let you magnify and reduce the screen images as needed.

Figure 70: Zoom controls in the Wire Sheet view



Zoom controls (buttons) appear on the Workbench tool bar when the Wire Sheet view is active.

- To enlarge the view, click the plus magnifying glass.
- To reduce the view, click minus magnifying glass
- To reset the view to its original size, click the empty magnifying glass.

Keeping all objects within view

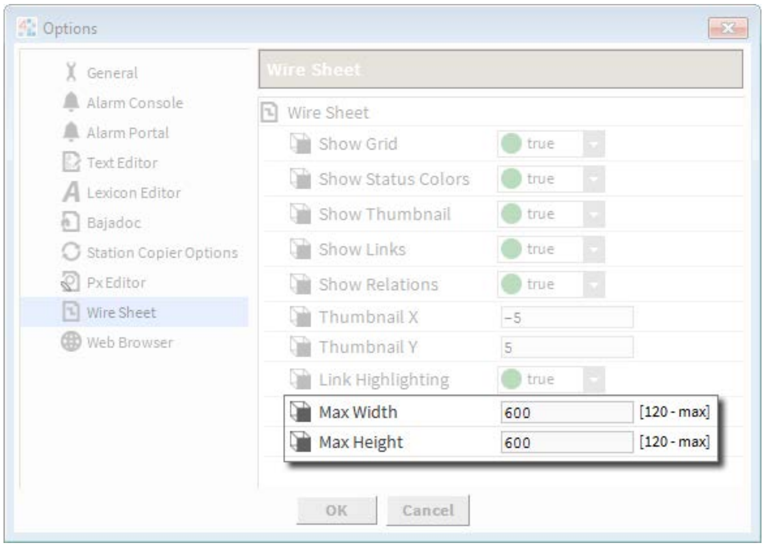
Having to scroll vertically and horizontally in the wire sheet view can become tiresome. You can configure this view to restrict its size so that all objects remain in view. If your model is complicated, you may need to configure multiple wire sheets.

Step 1 From the Workbench menu bar, click Tools→ Options.

The Options window opens.

Step 2 In the left pane of the Options window, click the Wire Sheet option.

Figure 71: Options window



Show Grid true

Show Status Colors true

Show Thumbnail true

Show Links true

Show Relations true

Thumbnail X -5

Thumbnail Y 5

Link Highlighting true

Max Width 600 [120 - max]

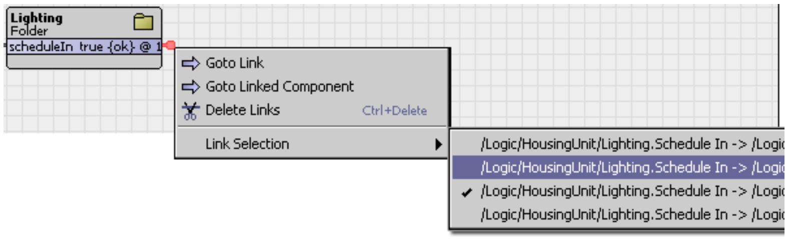
Max Height 600 [120 - max]

Step 3 Enter a desired maximum value (in pixels) in the Max Width and Max Height fields and click OK.

Link navigation

Link knobs provide easy navigation.

Figure 72: Link Selection menu


Lighting
Folder
scheduleIn true {ok} @ 1
Goto Link
Goto Linked Component
Delete Links Ctrl+Delete
Link Selection
/Logic/HousingUnit/Lighting.Schedule In -> /Logic
/Logic/HousingUnit/Lighting.Schedule In -> /Logic
✓ /Logic/HousingUnit/Lighting.Schedule In -> /Logic
/Logic/HousingUnit/Lighting.Schedule In -> /Logic

- Off-view linking

Double-clicking on the knob hyperlink at the opposite end of the link, displays the Wire Sheet view with the linked knob highlighted.

- Goto Link command

Right-clicking on the knob displays a GoToLink command on the popup menu. Selecting this command links to the component on the opposite end of the link, displaying that component's Wire Sheet view with the linked knob highlighted.

- Delete Links command

Selecting one or more off-view links (knobs) makes the Delete Links command available on the popup menu. This command removes all selected links and their associated off-view references. This is effective for multiple selected in-view and off-view (knob) links.

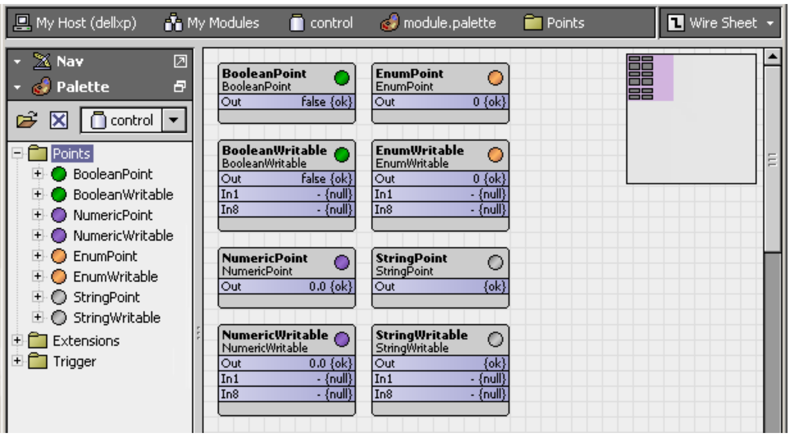
- Link selection

A Link Selection command is available from the popup menu when multiple links overlap on the wire sheet. Right-click on a link knob on the active wire sheet to select this command. Then choose the desired link from the secondary popup menu.

About control points

Control points are the foundation for all points in the station, including all proxy points.

Figure 73: Simple control points



The framework supports eight simple control point components. Each reflects a combination of a data (value) category and a point type. You can find these points in Points folder of the control palette.

Boolean Category	Numeric Category	Enum Category	String Category
BooleanPoint	NumericPoint	EnumPoint	StringPoint
BooleanWritable	NumericWritable	EnumWritable	StringWritable

The four categories apply to simple point components as well as to other components (for example, weekly schedules). These categories are the following:

- Boolean, which represents a binary value with only two states, such as off or on.
- Numeric, which represents an analog value, such as a temperature, level, rate or similar floating point number, or a varying count (integer). The system uses double-precision (64 bit) values.
- Enum, which represents an enumerated state (more than two), such as a multi-speed fan with states off, slow, and fast. Enums are often called multi-states or discretes. States typically derive from established integer value/state name pairs.
- String, which represents one or more ASCII characters (and if alpha-numeric), often with some literal meaning.

Each of the four point categories provides two point versions:

- A read-only version, which represents a data item that provides information and cannot be changed. Unlike the writable point version, there are no input type properties for read-only points. These four types are: BooleanPoint, NumericPoint, EnumPoint, and StringPoint.

Note:

As copied directly from the control palette, there is no application for read-only points. However, proxy points based upon read-only points, which are identical except for a non-null proxy extension and manner of creation, are both common and useful.

- A writable version, which represents a data item that can be changed, as well as read (usually by the station). These types are: BooleanWritable, NumericWritable, EnumWritable, and StringWritable.

An array of 16 InN inputs, each with a different priority level, is available to write a writable point's value. By default, the point's value can also be set with an operator-issued action (right-click command), available at priority levels 8 (override) and 1 (emergency). For more details, see the section "About point actions". Writable points also have other features. See, "About writable points".

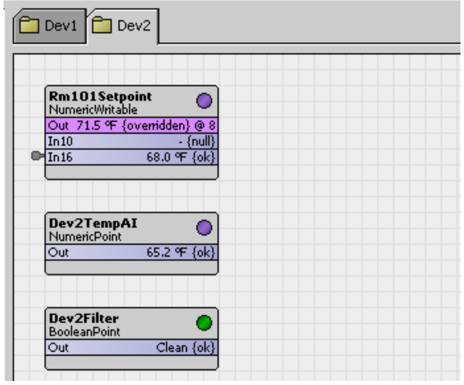
Other point components are found in both the kitControl palette. Briefly, these components include:

- Extensions, which expand a given point's functionality. As needed, you can add one or more extensions to a point, each as a child of that point. Extensions add functionality in a modular fashion.
- Time triggers provide periodic actions. These objects do not represent data, but, instead, they regularly fire a topic.
- Other control objects are found in various folders of the kitControl palette. They provide station control logic based on data obtained from points. Example objects include numeric math objects, Boolean logic objects, and a PID loop, among others.

About point properties

Each point has input properties and a single output property (Out) A point's Out property provides real-time information.

Figure 74: point Out provides real-time information



Rm101Setpoint
NumericWritable
Out 71.5 °F {overridden} @ 8
In10 - {null}
In16 68.0 °F {ok}

Dev2TempAI
NumericPoint
Out 65.2 °F {ok}

Dev2Filter
BooleanPoint
Out Clean {ok}

At a minimum, the Out property provides the following:

- A current value that conforms to one of four possible data categories.
- Facets, which define how the value displays. This information includes the value’s number of decimal places, engineering units, or text descriptors for Boolean/enum states.
- The current status of the data item, meaning the health and validity of the value. Status is specified by a combination of status flags, such as fault, overridden, alarm, and so on. If no status flag is set, status is considered normal and appears with the default status of {ok}.

Status flags are set in a number of ways.

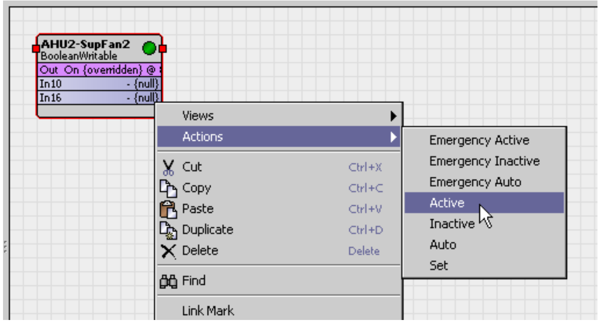
- The currently active priority level (from a 16-level priority scheme) for writable control points only. For more details on priority, see “About the priority scheme”.

The writable point’s priority appears at the end of the Out value. By default it is formatted as @ n, where n is a number from 1 to 16. If the fallback value is in effect, the value is formatted as @ def, for example: Off {ok} @ 1.

About point actions

Writable points have actions, which, by default appear in a right-click menu when you select a point in a Workbench view or in the Nav tree.

Figure 75: Actions available on right-click menu



An action is a slot that defines a behavior. Some other control objects and extensions also have actions. In the case of the four writable control points, default actions include the ability to:

- Override the point at priority levels 8 (override) and 1 (emergency override), where control can be independently set or “auto’ed” at either level. A level 8 override can be for defined (or custom) length duration, as specified in the action’s popup window.
- Set the value of the point’s Fallbackproperty.

Often, you modify a writable point’s default actions.

About override actions

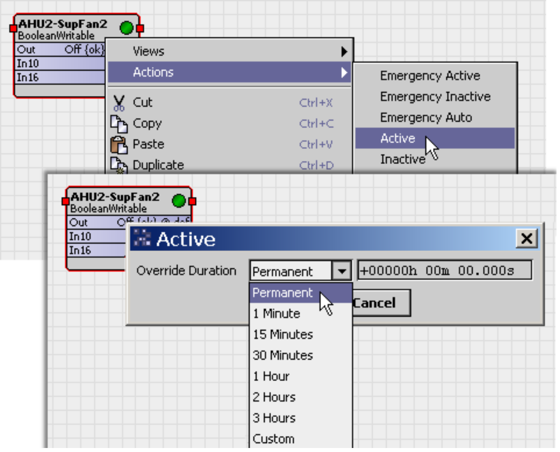
Whenever a writable point is controlled from an action issued at either override level, it has an “override” status. By default, override status color is magenta as shown below. For more point status details, see the section, “How status flags are set”.

A manual (level 8) override action to a point (not “auto”) prompts for override duration. See the following image.

Note:

Emergency overrides (level 1) do not have durations, these overrides are “permanent” (until auto’ed).

Figure 76: Override action duration (“argument”) dialog



Override Duration Permanent +00000h 00m 00.000s

By default, a Permanent override is the first choice in the drop-down list, the override value will remain effective until the next time this action is “auto’ed”. Other timed durations are available, including a Custom selection in which a user specifies a duration in hours, minutes, and seconds.

If needed, you can limit the maximum duration of manual override using facets.

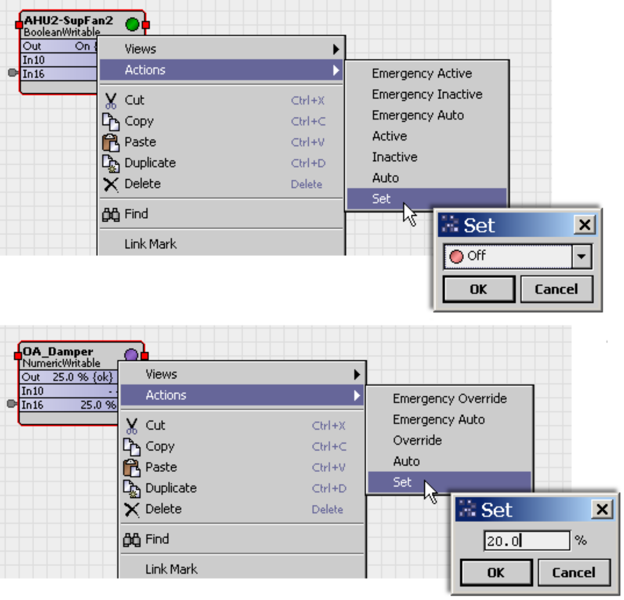
After clicking OK, the override action is issued to the point. If this is the highest effective priority level, the writable point operates under this control. If this is a timed override, the action is automatically “auto’ed” upon expiration of the override period.

About set (Fallback) action

Whenever a writable point has a null or invalid value at inputs In1—In16 (note this means that both override levels are currently “auto’ed”), the Out slot is set to the value of the Fallback property. For more details, see the section, “About the priority scheme”.

By default, an operator-level user can change the Fallback property, using the Set action. This produces a popup window that displays the current Fallback value.

Figure 77: Set action prompt to change writable point’s Fallback property



From the set action prompt, a Cancel leaves the current Fallback property unchanged. Otherwise, the Fallback property is set to the value entered (or currently displayed value).

Note that the set action prompt does not display (or accept) a null value for Fallback. However, a Fallback value of “null” can be entered from the point’s property sheet.

A common application for this feature is with NumericWritables used as setpoints, particularly under a NiagaraNetwork. As Niagara proxy points are always read-only points (not writable types), yet inherit any actions of the source point, this feature provides user access to setpoints via Px graphics without creating additional proxy points. In particular, this “set” action is designed to work well with SetPoint type widgets (found in the kitPx palette).

Note:

Each of the four constant kitControl components also provides a set action that works in a similar manner, including with kitPx widgets. However, a constant object (NumericConst, BooleanConst, etc.), has no priority inputs or Fallback property—the set action simply writes directly to the component’s current Out slot.

Modifying default actions

Unless all the defaults for actions of a writable point are acceptable (display names, all actions available, default user access), you may wish to modify action defaults. You can do this selectively from the slot sheet of the writable point.

Or, using facets you can limit the duration of manual overrides. See the section, “Maximum override duration facet”.

Following are some more details on slot sheet techniques:

- Display names — Change how the point’s popup action menu lists available actions
- Action access — Limit the actions that are made available, either by user level or for all users

Note:

As a “global” alternative to editing display names on slot sheets, you can edit the default values of lexicon keys, in this case for the control module for action type slots.

You can also modify the display name of the same action in multiple points in a single operation, using the (default) “Batch Editor” view of the station’s ProgramService (Config > Services > ProgramService).

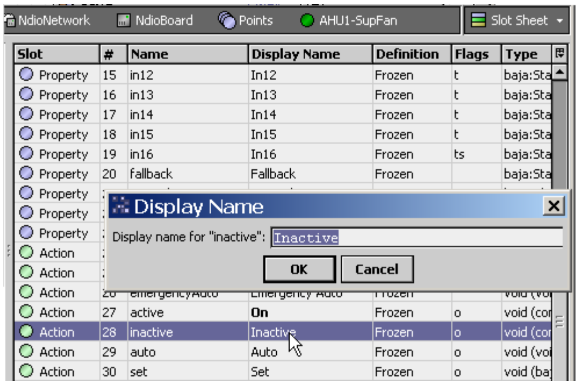
This is one of many examples of using the Batch Editor.

Also, starting in AX-3.6, you can use the Batch Editor to modify a config flag of the same slot on multiple components in a single operation.

Display names

By default, action display names are generic (“Emergency Inactive” and so on). You can change the display name for any action. From the slot sheet, click on an action’s Display Name for an editor. When you change a display name from defaults, it appears in listed in bold.

Figure 78: Editing action display names from slot sheet



Slot	#	Name	Display Name	Definition	Flags	Type
Property	15	in12	In12	Frozen	t	baja:Sta
Property	16	in13	In13	Frozen	t	baja:Sta
Property	17	in14	In14	Frozen	t	baja:Sta
Property	18	in15	In15	Frozen	t	baja:Sta
Property	19	in16	In16	Frozen	ts	baja:Sta
Property	20	fallback	Fallback	Frozen		baja:Sta
Property						
Property						
Property						
Action						
Action						
Action	26	emergencyAuto	Emergency Auto	Frozen		void (vo
Action	27	active	**On**	Frozen	o	void (co
Action	28	inactive	Inactive	Frozen	o	void (co
Action	29	auto	Auto	Frozen	o	void (vo
Action	30	set	Set	Frozen	o	void (ba

Display Name

Display name for "inactive": Inactive

OK Cancel

When a user invokes an action, the popup menu lists possible actions by more meaningful descriptors. For example, you could change the “set” action display name from “Set” to “Set Fallback.”

Action access

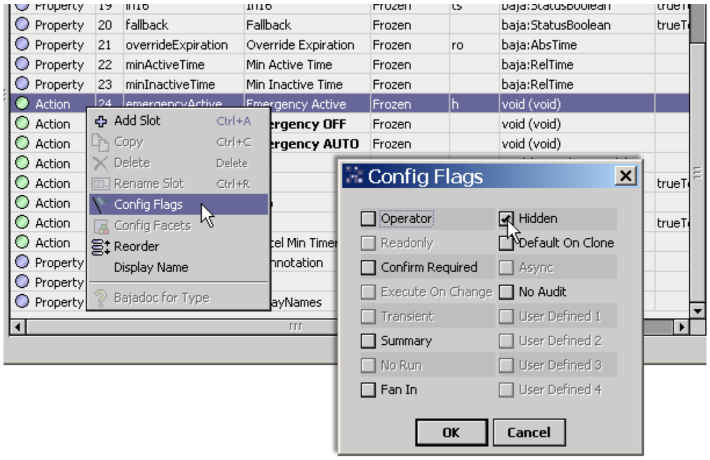
By default, for any writable point, all actions are available to any admin-level user, and all actions except emergency-level ones are available to an operator-level user. As needed, you can selectively “hide” actions (from any level user), or change default permissions for actions.

Note:

From a Px widget, you can also disable Px access to a bound writable point’s actions, by setting the “popupEnabled” binding property of the widget to false. In this case, access to the point’s actions would still be available from the point’s property sheet or in the wire sheet, unless otherwise changed from its slot sheet.

From the slot sheet, do this by editing the action’s config flags. Right-click the Action, and select Config Flags as shown in the next figure.

Figure 79: Editing config flags of action to “hide” or change permission level



Property	20	fallback	Fallback	Frozen		baja:StatusBoolean	trueT
Property	21	overrideExpiration	Override Expiration	Frozen	ro	baja:AbsTime	
Property	22	minActiveTime	Min Active Time	Frozen		baja:RelTime	
Property	23	minInactiveTime	Min Inactive Time	Frozen		baja:RelTime	
Action	24	emergencyActive	Emergency Active	Frozen	h	void (void)	
Action			ergency OFF	Frozen		void (void)	
Action			ergency AUTO	Frozen		void (void)	

Config Flags

☐ Operator
☑ Hidden
☐ Readonly
☐ Default On Clone
☐ Confirm Required
☐ Async
☐ Execute On Change
☐ No Audit
☐ Transient
☐ User Defined 1
☐ Summary
☐ User Defined 2
☐ No Run
☐ User Defined 3
☐ Fan In
☐ User Defined 4

In the Config Flags editor, you click to assign or remove config flags. For action slots, the following flags are most often changed:

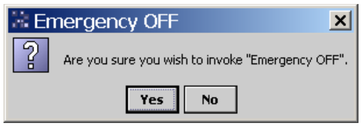
- Operator

If checked, only operator-level access is needed to invoke the action. If cleared, admin-level access is needed.

- Confirm Required

If checked, a second (confirmation) dialog appears after the action is invoked, before the action executes. An example confirmation dialog is shown below. By default, this flag is cleared.

Figure 80: Action confirmation dialog (from “Confirm Required” config flag)



Emergency OFF

Are you sure you wish to invoke "Emergency OFF".

- Hidden

If checked, the action does not appear (is hidden) from the action popup menu for any user. You may wish to do this selectively for some actions, for example, the “set” action for Fallback access. (Note that a user with admin-level rights to the point may still access the point’s slot sheet.)

As previously noted, starting in AX-3.6, the Batch Editor lets you modify a slot’s config flag on multiple points in one operation.

About point facets

Primarily, facets determine how the point’s value displays in the station. Examples include engineering units and decimal precision for numeric types, and descriptive value (state) text for boolean and enum types.

With the exception of proxy points (with possible defined device facets), point facets do not affect how the point’s value is processed.

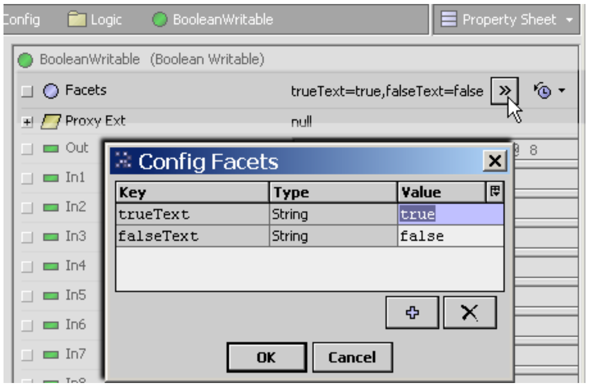
Note:

Besides control points, various other components have facets too. For example, many kitControl and schedule components have facets. Details about point facets apply to these components too, unless especially noted.

Accessing and editing facets

Facets is a frozen slot in all control points and objects in the kitControl module. As shown in the next figure, you modify facet values in a point’s property sheet, using the Config Facets dialog.

Figure 81: Point facets and edit dialog



Facets trueText=true,falseText=false

Proxy Ext null

Key	Type	Value
trueText	String	true
falseText	String	false

In this case a BooleanWritable’s default trueText facet is “true”—to modify you simply click to select, then type over with whatever text is needed. For example, change “true” to “On” and “false” to “Off”. When done click OK and then click Save to make the actual point change.

Optionally, in addition to modifying existing facets, you can add or remove facet values in a point. On many points you may only modify or provide new values for default facets. In the case of writable points, you can add a facet to limit override duration (see “Maximum override duration facet”).

Note:

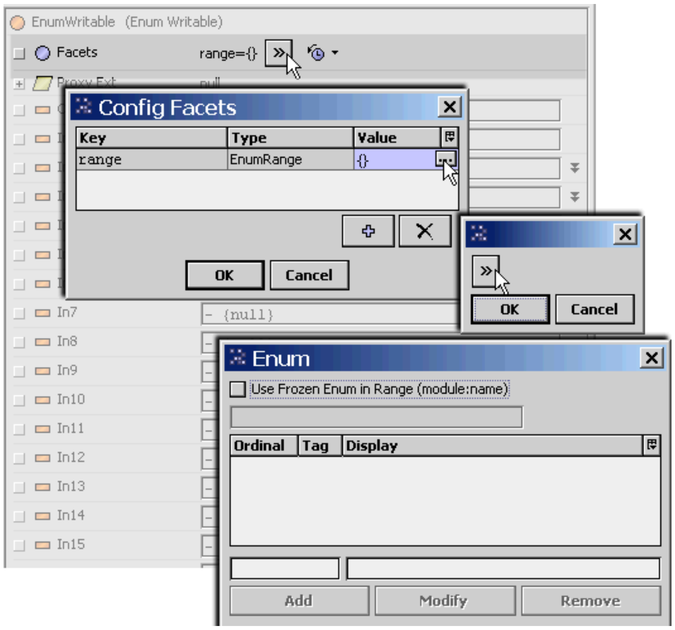
For string-type points (StringPoint, StringWritable), facets typically have little practical application. By default, the Facets slot is empty for string-type points.

Facets’ importance for enum points

Facets for enum-type components (EnumPoint, EnumWritable, EnumSchedule, etc.) define the operating range of the component, meaning its different possible enumerated states. Each state is defined by a pairing of an integer value-to-text, also known as ordinal-tag. Each ordinal must be a unique integer, and each tag must be unique text. By default, the point’s value displays using tag text.

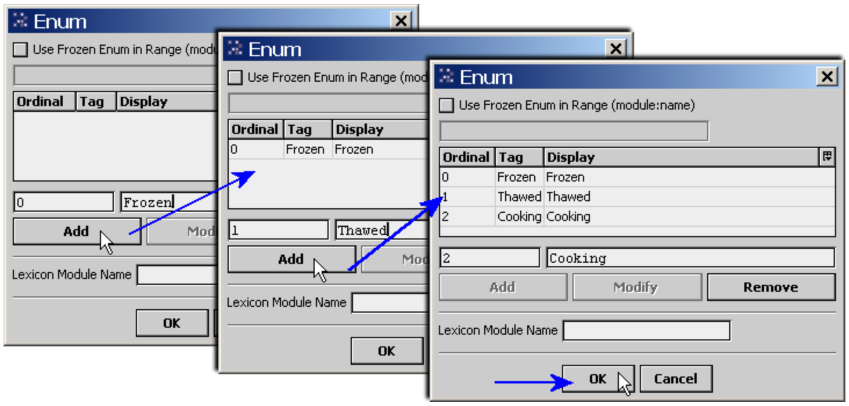
If you add an enum point from the control palette, its Facets slot has a blank range entry. Until you edit this facet and supply the ordinal-tag values, it can display only integer values. As shown in the next figure , a special Enum window appears when you edit range facets.

Figure 82: Producing Enum window for enum point “range” facets



In the Enum window when adding an entry, the Add button becomes available after you enter an integer value in the left side Ordinal box as shown below. Type in the associated text in the right side Tag box, and then click Add to add to the facet’s range. Click OK when done, also OK on any remaining pop-ups.

Figure 83: Type unique integer value in Ordinal box and associated text in Tag box

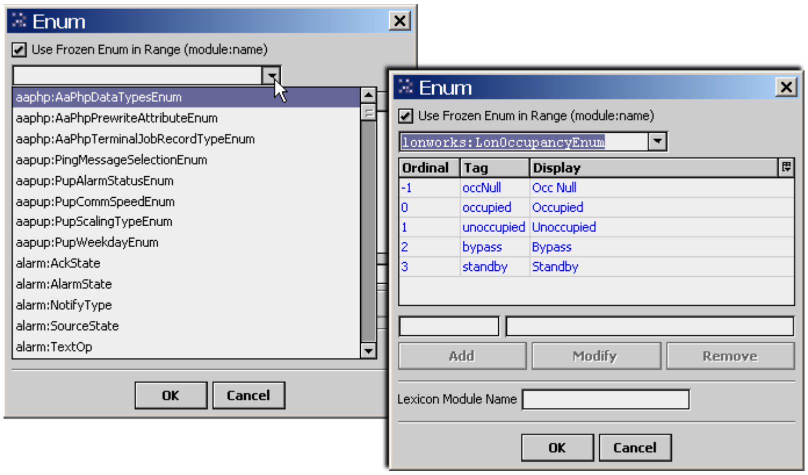


Ordinal	Tag	Display
0	Frozen	Frozen
1	Thawed	Thawed
2	Cooking	Cooking

If using lexicons, in the "Lexicon Module Name" field you can enter the module name of a configured lexicon (for example, control or kitControl) if Tag strings match lexicon "keys" in that lexicon file. In this case, enumerations will display the lexicon strings (values) for those ordinals instead of the tag text.

When defining range enumerations, instead of defining a custom one with your supplied ordinals and tags text, you can also select from well known "frozen" enumerations, as defined in various installed modules. A checkbox enables this and provides a drop-down list for you to select by module and enumeration type.

Figure 84: Frozen enum selection in Enum window



☑ Use Frozen Enum in Range (module:name)

aaphp:AaPhpDataTypesEnum
aaphp:AaPhpPrewriteAttributeEnum
aaphp:AaPhpTerminalJobRecordTypeEnum
aapup:PingMessageSelectionEnum
aapup:PupAlarmStatusEnum
aapup:PupCommSpeedEnum
aapup:PupScalingTypeEnum
aapup:PupWeekdayEnum
alarm:AckState
alarm:AlarmState
alarm:NotifyType
alarm:SourceState
alarm:TextOp

☑ Use Frozen Enum in Range (module:name)

lonworks:LonOccupancyEnum

Ordinal	Tag	Display
-1	occNull	Occ Null
0	occupied	Occupied
1	unoccupied	Unoccupied
2	bypass	Bypass
3	standby	Standby

Lexicon Module Name

Depending on the driver/network type, the Point Manager under a device may automate this facets range configuration when you add enum-type proxy points. For example, under a Lonworks device, if you add a EnumPoint for a Lonworks NVO that uses an enumerated SNVT, that point's facets will automatically be configured with the correct range values.

Note:

If an enum-type point receives an input value not included in its defined facets range, it displays the ordinal integer value for that input. This varies from the multistate objects used in r2 Niagara, which would display "Error" for any value not defined in its "stateText" entries.

Effect of facets on point actions

For some points with actions (see the section, "About point actions"), facets also affect availability in the point's action (command) menu.

- EnumWritable

Upon an override or emergency action, a secondary drop-down selection lists the possible enum values (in its range), using display tag text. This list appears ordered top-to-bottom by the tag associated with each ordinal, lowest-to-highest.

- NumericWritable

Upon an override or emergency action, an entry window permits a value only between the facets "min" and "max" values, inclusive. By default, these facets values for numeric-type points are min=-inf and max=+inf (no effective range checking for an action).

For example, you could use this facet's feature with a NumericWritable that sets a temperature control setpoint, by setting its facets min= 65 and max=85. After saving this change, any override or emergency action issued to that NumericWritable would need to fall within this range. Otherwise, a user would see a message showing the acceptable range, and be prompted to try again.

Note that Facets "min" and "max" values do not affect any received input values or proxied data, only what can be issued via an action.

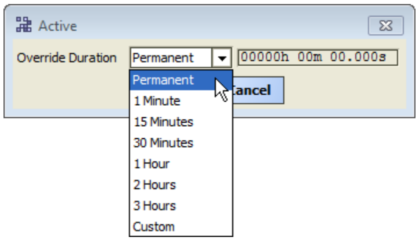
- Maximum override duration (for any writable point type)

Using facets you can also limit the maximum override duration of manual (level 8) override action invoked on writable control points. By default, the manual override of a writable point has no duration limits. For more details, see the section, "Maximum override duration facet".

Maximum override duration facet

Available for some time (but undocumented before), is the ability to limit the maximum override duration of an action invoked on a control point. By default, a manual (level 8) override of a writable point is unlimited in duration, thus the default "Permanent" label in the action menu.

Figure 85: Default override action menu for writable control point



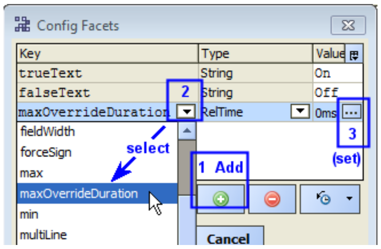
Override Duration: Permanent 00000h 00m 00.000s

If needed, change this by adding a "maxOverrideDuration" facet (choosing type baja:RelTime), with specified duration time, to either or both:

- Config, Sys Info property
- Any writable control point

Note that Override limits affects operator overrides (level 8) only, as emergency level overrides (level 1) are always unlimited in duration. In other words, an emergency level override lasts until an emergency level "auto".

Figure 86: Config Facets editor when picking maxOverrideDuration



Key	Type	Value
trueText	String	On
falseText	String	Off
maxOverrideDuration	RelTime	0ms

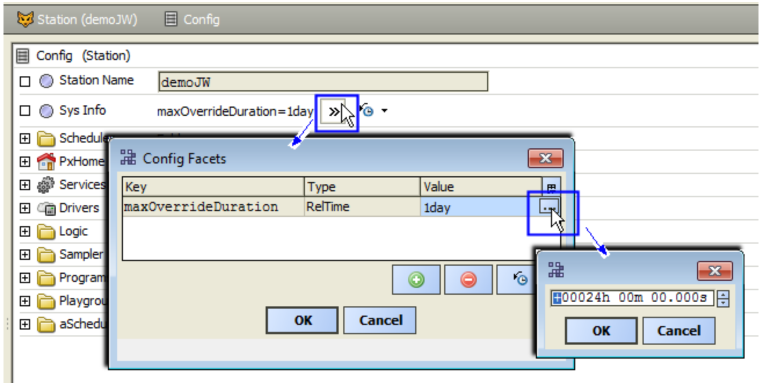
When a writable point is limited by a maxOverrideDuration facet, its action menu adjusts to show the allowable range.

For details about editing facets see the section, "Accessing and editing facets".

Config, Sys Info property

The Sys Info property of the station's root Config component has a facets control, shown in the Config component's property sheet.

Figure 87: Global maxOverrideDuration facet added to Sys Info property of station's Config component



Config (Station)

Station Name: demoJW

Sys Info: maxOverrideDuration=1day

Key	Type	Value
maxOverrideDuration	RelTime	1day

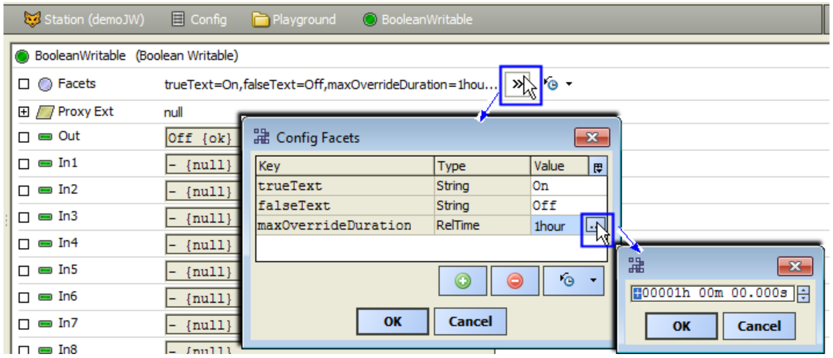
00024h 00m 00.000s

Adding this facet on the Sys Info property acts a global limit (station-wide) to a manual override action to all control points that do not have their own "maxOverrideDuration" facet.

Any writable control point

Each writable control point in the station can have a separately specified maximum override duration. If this facet is present, it overrides any global (Sys Info) maxOverrideDuration value.

Figure 88: Added maxOverrideDuration facet added at point level (overrides global setting)



BooleanWritable (Boolean Writable)

Facets: trueText=On,falseText=Off,maxOverrideDuration=1hou...

Proxy Ext: null

Out: Off {ok}

In1: - {null}

In2: - {null}

In3: - {null}

In4: - {null}

In5: - {null}

In6: - {null}

In7: - {null}

In8: - {null}

Key	Type	Value
trueText	String	On
falseText	String	Off
maxOverrideDuration	RelTime	1hour

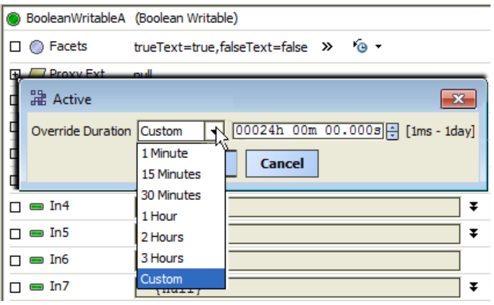
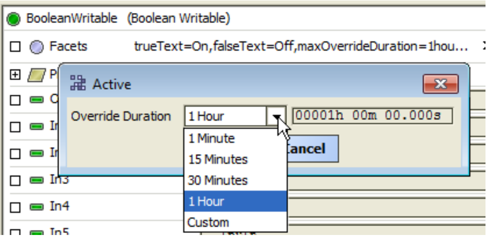
00001h 00m 00.000s

As shown, this maxOverrideDuration facet can be added along with any other facets in use by the control point. The example BooleanWritable point above already had configured facets for trueText and falseText.

Action menu examples

Example action menus from writable points with a maxOverrideDuration facet in effect are shown in the next figure.

Figure 89: Example override action menus affected by maxOverrideDuration facet

Global (Sys Info) level example	Point level example
	

BooleanWritableA (Boolean Writable)

Facets trueText=true,falseText=false

Proxy Ext null

Active

Override Duration: Custom; 00024h 00m 00.000s [1ms - 1day]

BooleanWritable (Boolean Writable)

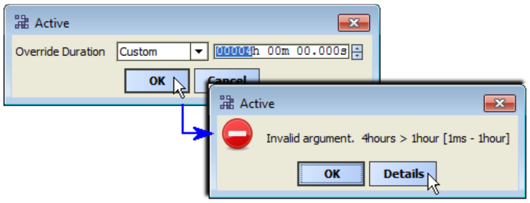
Facets trueText=On,falseText=Off,maxOverrideDuration=1hou...

Active

Override Duration: 1 Hour; 00001h 00m 00.000s

Note if a system user attempts to invoke a “Custom” override over the specified maxOverrideDuration limit, an error popup appears that shows the override duration range as shown below.

Figure 90: Custom override attempt over maxOverrideDuration limit produces error popup



Active

Override Duration: Custom; 00004h 00m 00.000s

Active

Invalid argument. 4hours > 1hour [1ms - 1hour]

As shown above, the allowable duration range appears in [brackets], in this case [1ms - 1hour].

About point extensions

As needed, you can add one or more extensions to a point, each as a child of that point. Extensions are a way to add functionality to a point (or extend it) in a modular fashion.

Extensions are found in several palettes, including alarm, control, history, and kitControl. Point extensions include the standard proxy extension (one for each point) and also optional extensions. See the section, “About the proxy extension”.

There are three main categories of optional extensions:

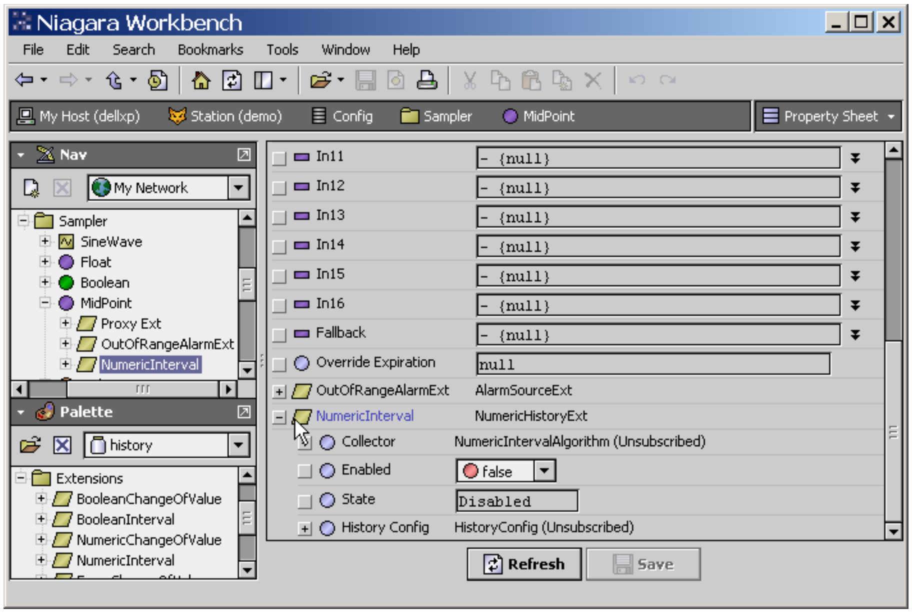
- Control extensions come from the control palette.
- Alarm extensions come from the alarm and kitControl palettes
- History extensions come from the history palette

You typically add an extension by either:

- dragging it into the property sheet of the point, or
- dropping it on the point’s icon in the Nav tree.

A point’s property sheet lists extensions below its normal (frozen) properties. You can expand each extension to view and modify its properties as shown below.

Figure 91: Extension expanded in a point’s property sheet



In11	- {null}
In12	- {null}
In13	- {null}
In14	- {null}
In15	- {null}
In16	- {null}
Fallback	- {null}
Override Expiration	null
OutOfRangeAlarmExt	AlarmSourceExt
NumericInterval	NumericHistoryExt
Collector	NumericIntervalAlgorithm (Unsubscribed)
Enabled	false
State	Disabled
History Config	HistoryConfig (Unsubscribed)

If a point has multiple extensions, they are processed in the same top-to-bottom order that they appear listed in that point’s property sheet. You can re-order extensions in a point from the top of the point’s property sheet, or on the point’s icon in the Nav sidebar, right-click and select Reorder.

Note:

If needed, you can also select and expose extension properties (for linking convenience) on the point’s glyph by using the Composite editor of the parent point. For more details, see the section, “About composites”.

About the proxy extension

Each point has a proxy extension (Proxy Ext), which is a frozen property. The proxy extension is important, it indicates how the point's data originates, including details specific to the parentage of the point's network and communications (driver).

A point's proxy extension is either:

- Null

For any point that you copy from the control palette (or add using the right-click menu), the proxy extension is simply null (NullProxyExt), an empty placeholder. The station itself originates the point's default Out value.

Also, many kitControl components also have a NullProxyExt, as they are based upon ControlPoints.

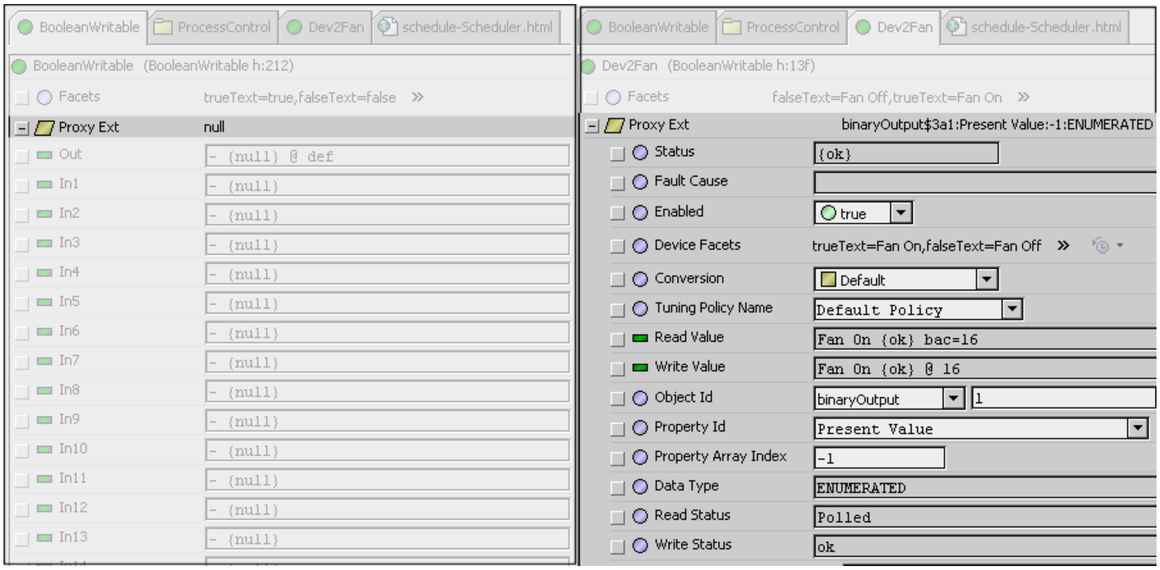
- <DriverType>

For any proxy point, meaning any of the 8 point types you create using the Points extension under a device represented in any of the network types, the proxy extension is <DriverType>ProxyExt. For example, a BooleanWritable proxy point under the Points container of a Bacnet Device has a proxy extension of BacnetBooleanProxyExt.

For any proxy point, its proxy extension contains information organized in child properties. Some properties may be unique to that specific driver type.

The following image compares property sheet views of the proxy extension properties for two Boolean-Writable points, on the left a control point, on the right a BACnet proxy point.

Figure 92: Proxy extension for control point (null) and a Bacnet proxy point.



BooleanWritable (BooleanWritable h:212)

Facets	trueText=true,falseText=false »
Proxy Ext	null
Out	- {null} @ def
In1	- {null}
In2	- {null}
In3	- {null}
In4	- {null}
In5	- {null}
In6	- {null}
In7	- {null}
In8	- {null}
In9	- {null}
In10	- {null}
In11	- {null}
In12	- {null}
In13	- {null}

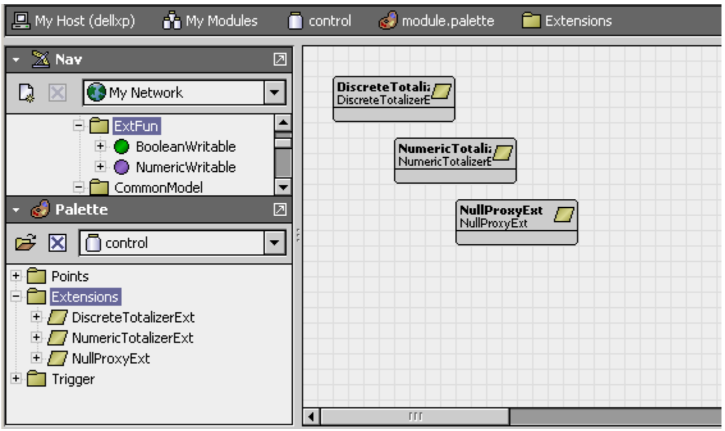
Dev2Fan (BooleanWritable h:13f)

Facets	falseText=Fan Off,trueText=Fan On »
Proxy Ext	binaryOutput$3a1:Present Value:-1:ENUMERATED
Status	{ok}
Fault Cause	
Enabled	true
Device Facets	trueText=Fan On,falseText=Fan Off »
Conversion	Default
Tuning Policy Name	Default Policy
Read Value	Fan On {ok} bac=16
Write Value	Fan On {ok} @ 16
Object Id	binaryOutput 1
Property Id	Present Value
Property Array Index	-1
Data Type	ENUMERATED
Read Status	Polled
Write Status	ok

About control extensions

Control extensions perform additional processing on a point’s received value. They are found in the Extensions folder of the control palette. As needed, you can add them to points along with alarm and history extensions.

Figure 93: Control extensions



There are relatively few types of control extensions. The following table lists all available control extension types and the applicable point parents.

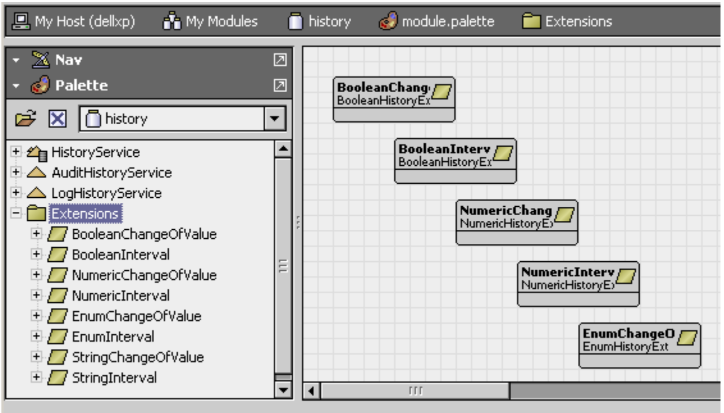
Control extension type (palette:Folder)	Applies to point types		What it does
	(read-only)	Writable	
DiscreteTotalizerExt (control:Extensions)	BooleanPoint	BooleanWritable	Accumulates runtime and change of state (COS) count. Extension actions permit resetting (zeroing) the runtime and COS count.
	EnumPoint	EnumWritable	
	—	any object with single Boolean Out, e.g. kitControl: Logic object “And”	
NumericTotalizerExt (control:Extensions)	NumericPoint	NumericWritable	Accumulates numeric total using hourly or minutely totalization. Extension has action to reset (zero) total.
	—	any object with single Numeric Out, e.g. kitControl: Math object “Add”	
ProxyExt (control:Extensions)	Standard for any point. See the section, “About the proxy extension”.		Provides methods to driver communications.

About history extensions

Add a history extension to any point for which you want to log historical data, that is, to collect a history. In a point history, each collected sample of slot “Out” has a date-timestamp, point status and value. Point history data is viewable in both a table and chart format.

Find the history extensions in palette history: Extensions.

Figure 94: History extensions



Each history extension has various properties in two major groups:

- Collector properties determine how or when data is collected, such as active time, and (if applicable) either change tolerance or the collection interval time.
- History Config properties determine how the system stores the data. These properties include history name, capacity, and full policy.

The following table lists all history extension types and the applicable point parents.

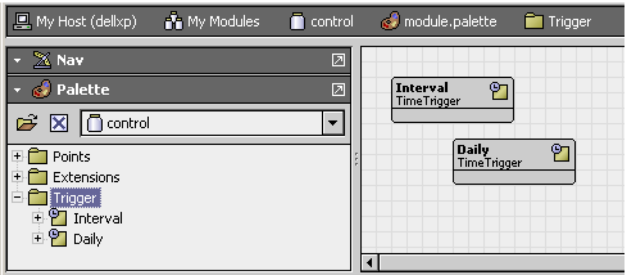
History extension type	Applies to point types		General description
	(read-only)	Writable	
BooleanChangeOfValue	BooleanPoint	BooleanWritable	Collects upon each change of Boolean value (state) or status.
	—	any object with single Boolean Out, e.g. kitControl: Logic object type “And”	
BooleanInterval	as above	as above	Collects upon a repeating time interval, as configured.
NumericChangeOfValue	NumericPoint	NumericWritable	Collects upon each change of value (outside a specified tolerance) or change of status.
	—	any object with single Numeric Out, e.g. kitControl: Math object type “Add”	
NumericInterval	as above	as above	Collects upon a repeating time interval, as configured.
EnumChangeOfValue	EnumPoint	EnumWritable	Collects upon each change of enumerated state or status.
	—	any object with single Enum Out	
EnumInterval	as above	as above	Collects upon a repeating time interval, as configured.
StringChangeOfValue	StringPoint	StringWritable	Collects upon each change of string value or status.
	—	any object with single String Out	
StringInterval	as above	as above	Collects upon a repeating time interval, as configured.

All history extensions have an Update History action available. This popup menu item provides a way to refresh the History Id after a rename. It applies the formatting property (as designated by enclosing % signs) of the Name Format field to the Id of the History Config Id. For example, if the History Name property under a history extension is set to %parent.name%, then the History Config Id is initially named based on this parent display name. However, if you rename the parent component, the History Config Id property does not automatically or immediately change. The Update History Id action invokes a renaming of the History Config Id based on the formatting property, so if the parent component (in this example case) is changed, the Update History Id action changes the Id property and, if different from the history name, it results in a change in the history name as well.

About control triggers

There are two “time triggers” in the palette control: Trigger. These objects do not represent data, but instead regularly fire a topic.

Figure 95: Control triggers



The two control trigger types are:

- Interval: Fires at a regular, repeating intervals specified in its Trigger Mode property. For example, every n minutes, n hours, or whatever combination needed.
- Daily: Fires once a day at a specific time and day of week, as specified in its Trigger Mode property. For example, at 00:00:00 (midnight) all days of week except Sunday.

Each type has even more configuration capabilities to further define the fire time.

How triggers are used

To use a trigger, you typically link it to a selected action of a point extension (e.g. control, alarm, and history) to automate an action. Often, you use a trigger as a child of a particular point (sibling to the linked extension). Or, you can have a trigger in the same container as multiple points, and link it to more than one point or point extension.

For example, a Daily trigger (defined for midnight) can be linked to the ResetElapsedActiveTime action of a DiscreteTotalizerExt extension of a BooleanWritable point. In this case, that point’s DiscreteTotalizerExt would only show runtime accumulated during the current day.

About related objects

Other objects with trigger functions are found in various palettes. The most closely related is the Trigger-Schedule object, found in the Schedule palette.

About point status

Along with point value, point status is available at point Out. Status reflects status flags, which may get set singly, or in combinations. Status flags are typed by a unique text string (for example: “alarm” or “down”), and many have associated status colors (a background and a foreground). For example, the default status color for alarm is white text on red background.

Status without any flags set is considered normal (text “ok”), and is without color indication.

Note:

For each installed lexicon, text strings for status flags (plus associated colors), are individually adjustable by editing default values in that host’s baja lexicon, using Workbench’s Lexicon Editor. For a default English installation, to change default status appearance settings, edit the en: baja lexicon. For more details, see the sections, “About lexicons” and “Lexicon Editor view”.

Types of status flags

Status is indicated by text on a colored background.

The next table lists the status types, illustrates the colors and explains what they mean.

Type	Default Colors, Example	Meaning
alarm	white text, red background 65.0 °F	Point currently has a value in an alarm range, as defined by property in its alarm extension.
fault	black text, orange background 208.8 °F	Originates from a proxy point only. Typically indicates a configuration or licensing error. If it occurs after normal operation, it may indicate a “native fault” in device, or the point’s parent device has a fault status.
overridden	black text, magenta background 72.0 °F	Current point control is from an action, meaning a user-invoked command at either priority level 8 (override) or priority 1 (emergency).
disabled	gray text, light gray background 79.0 °F	Originates from a proxy point only. Point (or its parent device or network) has been manually disabled (property enabled = false).
down	black text, yellow background 68.4 °F	Originates from a proxy point only. Driver communications to the parent device are currently lost, based upon the device status (Monitor) configuration for that network.
stale	black text, tan background 73.4 °F	Originates from a proxy point only. Driver communications have not received a requested response for this data item within the configured times (Tuning period).
null	(no color indication)	Current point control has entered a null state, vs. a specific value and priority level. Typical to fallback operation for a writable point. NOTE: If linking a null status Out to a “simple data” slot, the point’s “null value” is processed.
unackedAlarm	(no color indication)	Last point alarm event has not yet received user acknowledgment. Point’s alarm extension uses alarm class requiring acknowledgment.

Priority of status indication

Since status flags for a point or object can get set in combinations, status color indication uses a priority method. Among those 6 status flags with associated colors, priorities (and default colors) are:

1. disabled (dark gray): Proxy point origination only. Point may have other status flags set. Typically, you manually set and clear this status (unlike others). After disabled is set for a proxy point, it is no longer polled. Further status changes do not occur until disabled is cleared.

2. fault (orange): Typically proxy point origination only.

3. down (yellow): Proxy point origination only.

4. alarm (red): Point may have other status flags set.

5. stale (tan): Proxy point origination only.

6. overridden (magenta): Point may have other status flags set.

Note that Status types unackedAlarm and null do not affect the indicated status color.

How status flags are set

Status flags are set differently depending on the type of point or control object.

- Simple control point status
- Propagate Flags status option (linked Math and Logic objects)

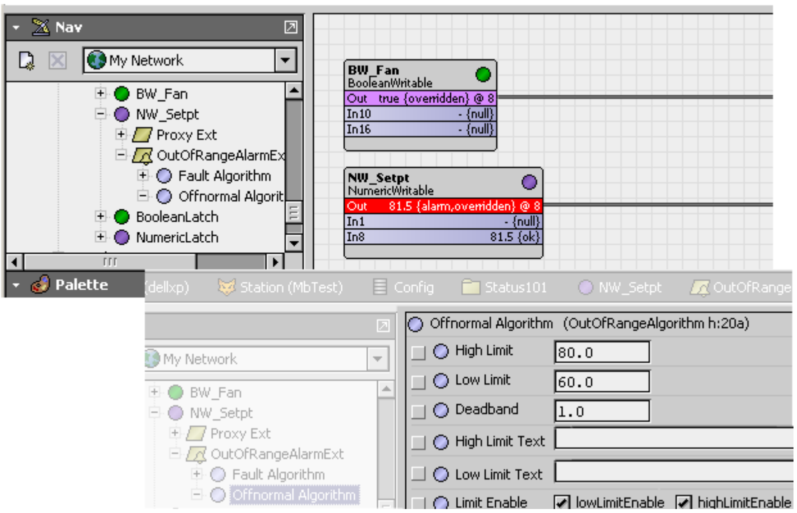
Simple control point status

For simple control points (NullProxyExt) the following status flags are the only ones set and cleared:

- alarm: Point is currently in an alarm condition as defined in its alarm extension.
- unackedAlarm: Point has alarm extension assigned to an alarm class requiring acknowledgment, but the last alarm event has not yet been acknowledged. Point may/may not be in alarm.
- override: Writable points only, during a user-initiated action at priority levels 8 (override) or 1 (emergency). The override flag clears when the action “times out,” or when a user issues an “auto” action at that same priority level.
- null: Writable point has “null” or otherwise “invalid” value at In1—In16, plus “null” configured as “Fallback” value.

The following image gives examples of override and alarm status in simple control points.

Figure 96: Status with simple control points



BW_Fan
BooleanWritable
Out true {overridden} @ 8
In10 - {null}
In16 - {null}

NW_Setpt
NumericWritable
Out 81.5 {alarm,overridden} @ 8
In1 - {null}
In8 81.5 {ok}

Offnormal Algorithm (OutOfRangeAlgorithm h:20a)
High Limit 80.0
Low Limit 60.0
Deadband 1.0
High Limit Text
Low Limit Text
Limit Enable ☑ lowLimitEnable ☑ highLimitEnable

Propagate Flags status option (linked Math and Logic objects)

By default, kitControl objects maintain independent status flags from input-linked points. However, as a configuration option in each math or logic-type kitControl object (kitControl palette folders “Math” and “Logic”), you can specify “out” status to propagate from input status.

The object’s Propagate Flags property allows you to select any combination of the following status types for propagation:

- disabled
- fault
- down
- alarm
- overridden

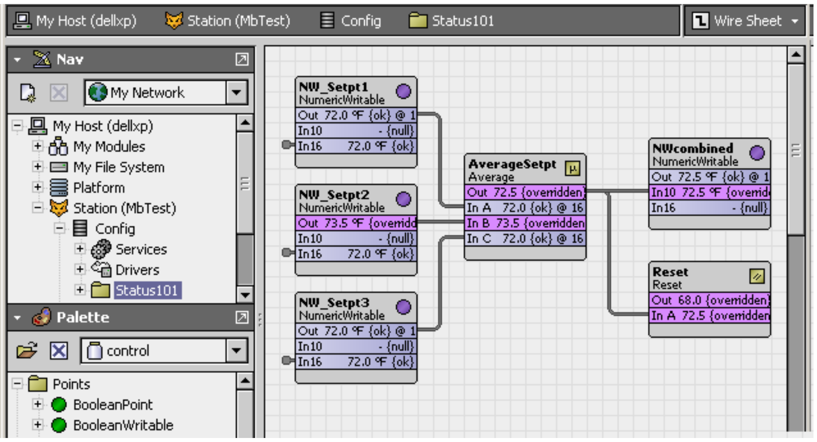
Note:

If the math or logic object has multiple inputs, and you set the propagateFlags property to select one or more of the statuses above, simple “OR” logic is used across all inputs for the propagation of each selected status.

Depending on usage, status propagation may be extensive. Note that four of the five status types (all except alarm and overridden) are “invalid status,” meaning they cause the output of the object (if linked) to be considered invalid at its destination target. See the section, “About isValid status check”.

As an example of status propagation, some number of NumericWritable points are used to establish set points, and you link them all to a Math: Average object for downstream zone control. In turn, the Average object feeds a Math: Reset object. Both math objects have “override” enabled in their “propagateFlags” property. A user issues an override (action) to one of the NumericWritable points, to override a setpoint.

Figure 97: Status propagation in linked writable points, kitControl objects



For the duration of the override, the linked Average object will also have an overridden status, as will the Reset object, and so on. However, note that the linked writable point (NWcombined) in this example does not have overridden status; status never propagates to any point.

Note:

Before using this feature in an actual job, you should test and evaluate results to be sure it has the desired effect. For example, if a Logic or Math object is exposed in a graphic and appears overridden, a user may (incorrectly) assume that they can right-click command (perform an action) on that kitControl object, based on status color indication.

About “isValid” status check

When linking an input-type slot of a writable point or kitControl object, the value received at the input is processed (evaluated) by that point or object only if it is valid.

A valid input is one with none of the following status flags set:

- down
- fault
- disabled
- null
- stale

If any of the above status bits are set at an input value that input value is not used.

- If a kitControl object with a single input, by default that object uses the last known valid received (at least until the input becomes valid again).
- If a kitControl object with a multiple inputs, only the valid inputs are evaluated.
- If a writable point, the priority scan continues. See the section, “About the priority scheme”.

About writable points

All writable points provide right-click actions. Override actions are evaluated within the priority scheme used by any writable point. In the case of the “set” action, the point’s “Fallback” property is modified. See “About point actions”. BooleanWritable points also offer built-in “minimum on/off” timers.

About the priority scheme

Each writable point uses a 16-level priority scheme, with corresponding inputs In1—In16, plus a “Fallback” property. Level 1 is the highest priority, and level 16 is the lowest.

The following topics further describe the priority scheme:

- Priority input scan
- Priority linking rules
- Priority level conventions

Due to our policy of continuous product innovation, some specifications may change without notification.

Priority input scan

For any writable point, the effective input value is determined by a priority scan, looking for a “non-auto” action at level 1 (emergency), then the value at the highest valid input, going from level 2, to 3, and so on to level 16. (At level 8, any “non-auto” action is evaluated as valid).

Like almost all control execution, this priority scan is event-driven, meaning it occurs when any input value changes. An input’s value typically comes from a link—however, for most inputs, you may enter a value directly in the point’s property sheet (as an alternative source).

A valid input is one with none of the following status bits set:

- down
- fault
- disabled
- null
- stale

If all 16 priority levels are evaluated without a valid input (and without an action at levels 1 and 8), then the fallback value is used.

You can configure the writable point’s Fallback property to be null, so that the point’s Out has a null status in this condition. Depending on the specific control sequence and usage of the writable point, this may be an effective solution.

However, by default, a set action exists on any writable point, which writes directly to the Fallback value. See the section “About set (Fallback) action”. If you want a writable point to always have a Fallback of null, go to its slot sheet, and set the Hidden config flag on the se” slot. Otherwise, the user can invoke a right-click command to set Fallback to any value. For more details, see “Modifying default actions”.

Priority linking rules

When linking to the priority inputs of a writable object, you must follow rules.

- Only one link per input (level).
- Levels 1 and 8 are unavailable for links. If a BooleanWritable, level 6 is also unavailable.

Priority levels 1 and 8 are reserved for actions (emergency and override). See the section, “About point actions”. Priority level 6 in a BooleanWritable is reserved for minimum on/off times. See the section, “About minimum On and Off times”.

Note:

Both rules vary from the r2 Niagara priority input scheme, where a single priorityArray input was used for a writable object (AnalogOutput, BinaryOutput, and so forth). That input could be linked to multiple priority type outputs, including those with duplicate priority levels and/or levels also used for object commands (emergency and manual).

Priority level conventions

The 16 priority levels used by writable points are modeled after corresponding BACnet priority levels. Priority levels confirm to the following conventions, from highest to lowest:

1. Emergency (Manual Life Safety)—Unlinkable input, but available as action (command).
2. Automatic Life Safety
3. User Defined
4. User Defined
5. Critical Equipment Control
6. Minimum On/Off (BooleanWritable meaning only, see the section, “About minimum On and Off times”.
7. User Defined

8. Override (Manual Operator)—Unlinkable input, but available as action (command).
9. Demand Limiting
10. User Defined
11. Temperature Override
12. Stop Optimization
13. Start Optimization
14. Duty Cycling
15. Outside Air Optimization
16. Schedule

Note:

Although priority levels are patterned after BACnet, there is no direct linkage to BACnet priorities, even with BACnet writable proxy points. Priority inputs of all writable points are strictly a station-centric implementation.

About minimum On and Off times

Each BooleanWritable point has built-in timers to specify minimum on and/or minimum off times. The respective point properties are “Min Active Time” and “Min Inactive Time.” Usage is optional, and both properties work independently. Typical usage is to prevent short-cycling of equipment controlled by the point.

Default property times for a BooleanWritable are all zeros (“00000h 00m 00s”), which effectively disables a timer. In either property, you can specify any value needed using of a mix of hours (h), minutes (m), and seconds (s) to enable that timer.

A minimum timer is triggered by a state change transition to active or inactive. This results in the new state value being stored in the point’s priority array (at priority level 6) for the duration of that timer. While a minimum timer is in effect, only input changes at a higher priority (5 or above) or an emergency action can affect the Out value.

For example, a BooleanWritable point controls a fan, with related properties set as follows:

- Min Active Time: 00000h 01m 30s: Specifies that once started, the fan must run at least 90 seconds.
- Min Inactive Time: 00000h 03m 5s: Specifies that once stopped, the fan must remain stopped at 3 minutes, 5 seconds.

Starting with the fan stopped at schedule level (priority 16), if a user gives it a manual override on (priority level 8), the fan will run for 90 seconds at priority level 6 (a higher level). After this period, the fan continues running at the override 8 level for the duration of the override.

During the initial 90 seconds, a different override action (off or auto) will be ineffective, as the higher priority level 6 remains in control. See the section, “Priority level conventions”.

Once stopped, the point’s minimum off time will keep the fan off at priority level 6 for the specified duration (in this example, 3 minutes and 5 seconds). During this period, only an emergency command or input change at In2—In5 can effect further change.

About composites

Currently, a composite is something that Workbench allows you do to virtually any component in a station, notably control points and objects, and even folders that contain control logic. When you make a composite, you expose slots of child components in the glyph (object shape) of that parent (composited) component. This can simplify linking and promote reuse of control logic.

Note:

- Composites have associated issues. See “Composite issues”. For now, you should avoid making folder composites in your control logic, and instead use the composite feature only at the point/object level to expose extension slots (if necessary).
- If you are familiar with Niagara r2, the composite concept is similar to “Bundle” or “Composite” objects, only more flexible. You can expose slots in containers many levels down, for example.

When you composite a component (say a control point, meaning its contents), you select specific slots in child components (say, properties and/or actions of its extensions) to be “exposed” in the “shape” of that point. Then, when looking at that point in the wire sheet view of its parent folder, you can see exposed properties of children as linkable slots (and/or available actions).

Some composite examples

A few simple examples of composites are the following:

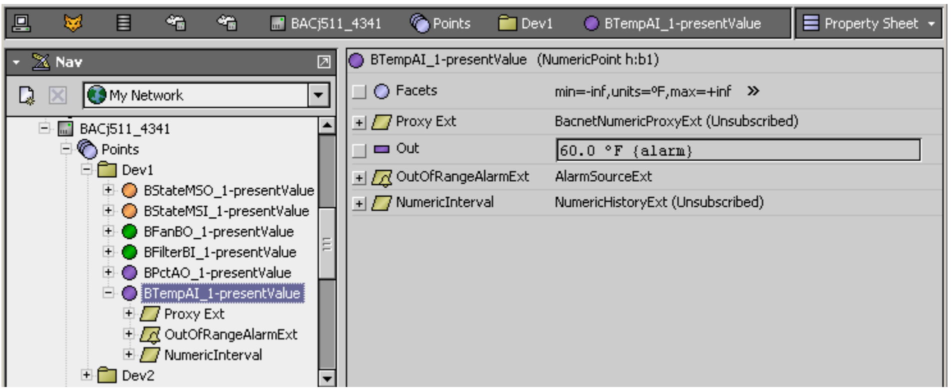
- Point-level composite
- Folder-level composite

Point-level composite

The following image shows a proxy NumericPoint representing a space temperature value that has two extensions:

- an alarm extension (OutOfRangeAlarmExt)
- a history extension (NumericInterval)

Figure 98: Property sheet of proxy point with two extensions



BACj511_4341 Points Dev1 BTempAI_1-presentValue Property Sheet

Nav

My Network

- BACj511_4341
  - Points
    - Dev1
      - BStateMSO_1-presentValue
      - BStateMSI_1-presentValue
      - BFanBO_1-presentValue
      - BFilterBI_1-presentValue
      - BPctAO_1-presentValue
      - BTempAI_1-presentValue
        - Proxy Ext
        - OutOfRangeAlarmExt
        - NumericInterval
    - Dev2

BTempAI_1-presentValue (NumericPoint h:b1)

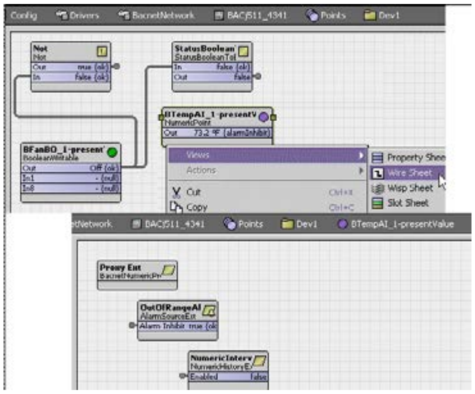
Facets	min=-inf,units=°F,max=+inf »
Proxy Ext	BacnetNumericProxyExt (Unsubscribed)
Out	60.0 °F {alarm}
OutOfRangeAlarmExt	AlarmSourceExt
NumericInterval	NumericHistoryExt (Unsubscribed)

In this example, when another system-related BooleanWritable (representing the system fan) has a false (Off) status, it is desired that this temperature point be:

- disabled from alarming, and
- disabled from continued history collection

To achieve these “disable” functions, you must link the controlling source fan out to a slot of each extension (visible in point’s wire sheet view, but not in the parent wire sheet for the point itself). Furthermore, other kitControl objects needed. Without making a composite, the necessary objects and links may appear.

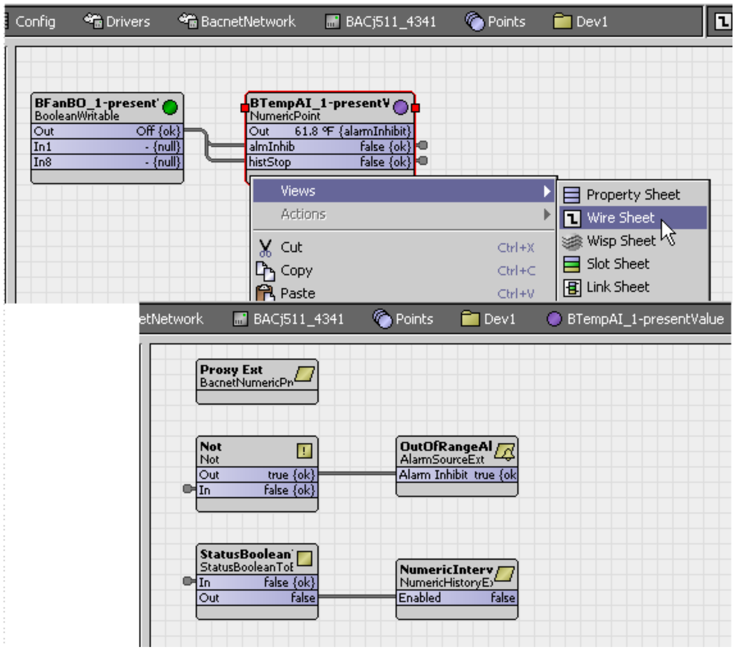
Figure 99: Proxy point example without composite



Notice that when looking at the proxy NumericPoint in the wire sheet of its parent folder, it is not apparent that this point has linked extensions.

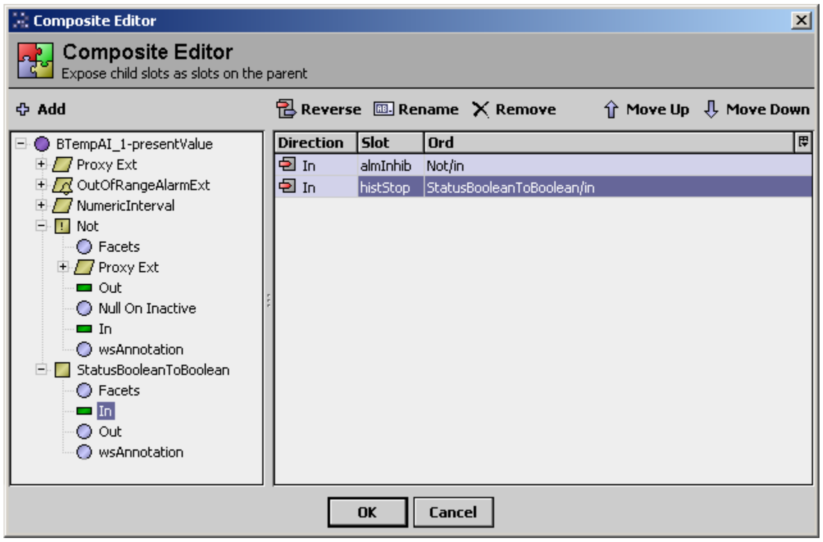
By making a composite of the NumericPoint (acting as the container for both the extensions plus the additional kitControl objects) you can simplify reuse and clarify available links. The following image shows the now-composited NumericWritable linked to the controlling BooleanWritable, and the wire sheet view of the NumericWritable that contains the needed kitControl objects.

Figure 100: Proxy point as composited container



In this example, the exposed input slots in the composite were renamed from “In” to “almInhib” and “histStop” respectively, to clarify what each does. When looking at the Composite Editor for this example NumericPoint, it appears.

Figure 101: Composite Editor for example NumericPoint

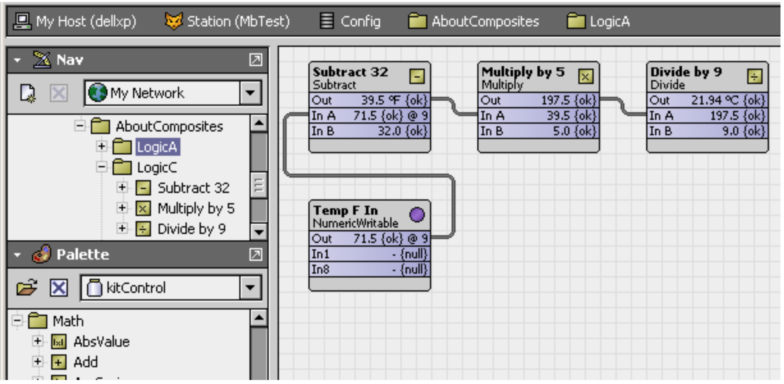


Direction	Slot	Ord
In	almInhib	Not/in
In	histStop	StatusBooleanToBoolean/in

Folder-level composite

This topic provides a simple example of three Math objects chained together.

Figure 102: Simple example of folder before compositing



The objects are located in a LogicA folder. Together, they perform a Celsius to Fahrenheit conversion. A NumericWritable is also shown linked to the first Math object to test.

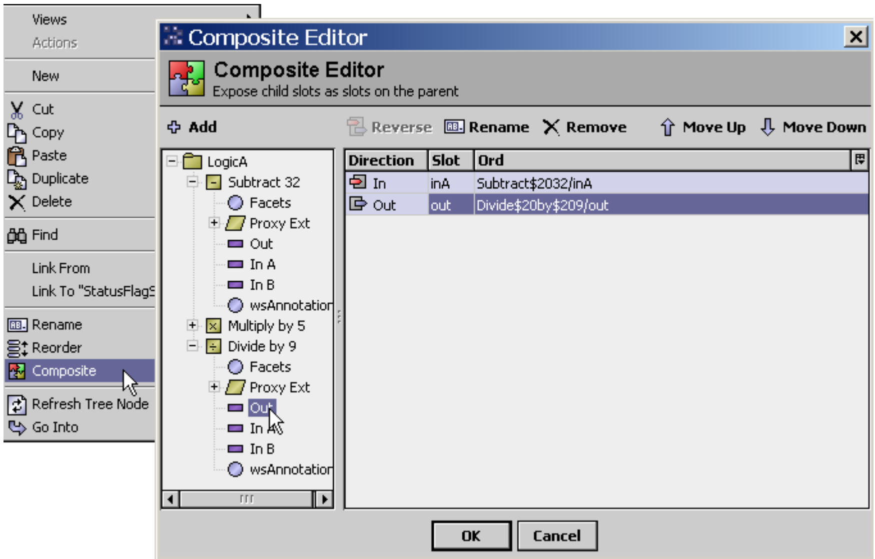
If this application was needed later, you could copy all tjree linked objects again and insert them in another (perhaps already crowded) wire sheet. However, the middle “Multiply” object reveals an intermediary result that is distracting.

Or, you could just create a new subfolder with only the three linked objects and then link directly to the child objects as needed. However, it would not be obvious from the parent’s wire sheet that links to children in that folder were established.

It would be better to encapsulate this into a single object with only a single input (degrees F) and single output (degrees C). You can do something like this by compositing the parent container, in this example, folder FolderA.

In this case, you would want to delete the link from the test NumericWritable first, and then open the Composite Editor for the parent component FolderA. The Composite Editor lets you expand the tree of all contained components and “expose” those items of interest.

Figure 103: Launching and using the Composite Editor

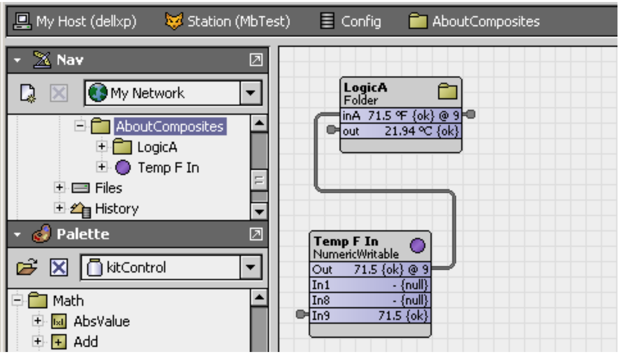


Direction	Slot	Ord
In	inA	Subtract$2032/inA
Out	out	Divide$20by$209/out

In this example, only the “In A” of the first math object and the “Out” of the third math object is selected to be exposed. The Composite Editor provides a tree pane showing slots of points and objects (by clicking the expand controls), and a slot is exposed by simply double-clicking it. Other controls in the editor are available to rename, remove, reorder and reverse exposed items, but are not used here.

After clicking OK to perform the composite, the item composited (in this example, LogicA) shows exposed slots when viewed in its parent’s wire sheet. The following image shows the test NumericWritable now linked to the composited LogicA folder.

Figure 104: Example LogicA folder showing exposed input and output



You could later reopen this folder’s Composite Editor and rename the exposed properties differently, perhaps “inDegF” and “outDegC” to make the purpose of the composited folder clearer. This would not affect the three (child) math objects in any way.

Composite issues

Although composites can simplify linking and make understanding logic flow easier, they always introduce performance issues. Perhaps the biggest issue is that once you link a dynamic value to a composite, for example the “out” of a proxy point, it essentially “pinned” into the “subscribed” state. This means that proxy point will always be polling (regardless of any other usage).

In addition, each item exposed in a composite represents a link, where each link consumes some small amount of station resources. If used excessively, composites could noticeably reduce the total capacity of the station.

CHAPTER 4 ABOUT WORKBENCH TOOLS

The following topics are covered in this chapter:

- Alarm portal
- Bacnet Service
- Security management
- Lexicon Tool
- Local License Database
- Logger Configuration tool
- Lon Xml Tool
- Lonworks Service
- Credentials manager
- NDIO to NRIO Conversion Tool
- New Driver wizard
- New Module wizard
- New Station wizard
- Request License
- Resource Estimator
- Time Zone Database Tool
- Todo List
- Workbench Job Service
- Workbench Service Manager

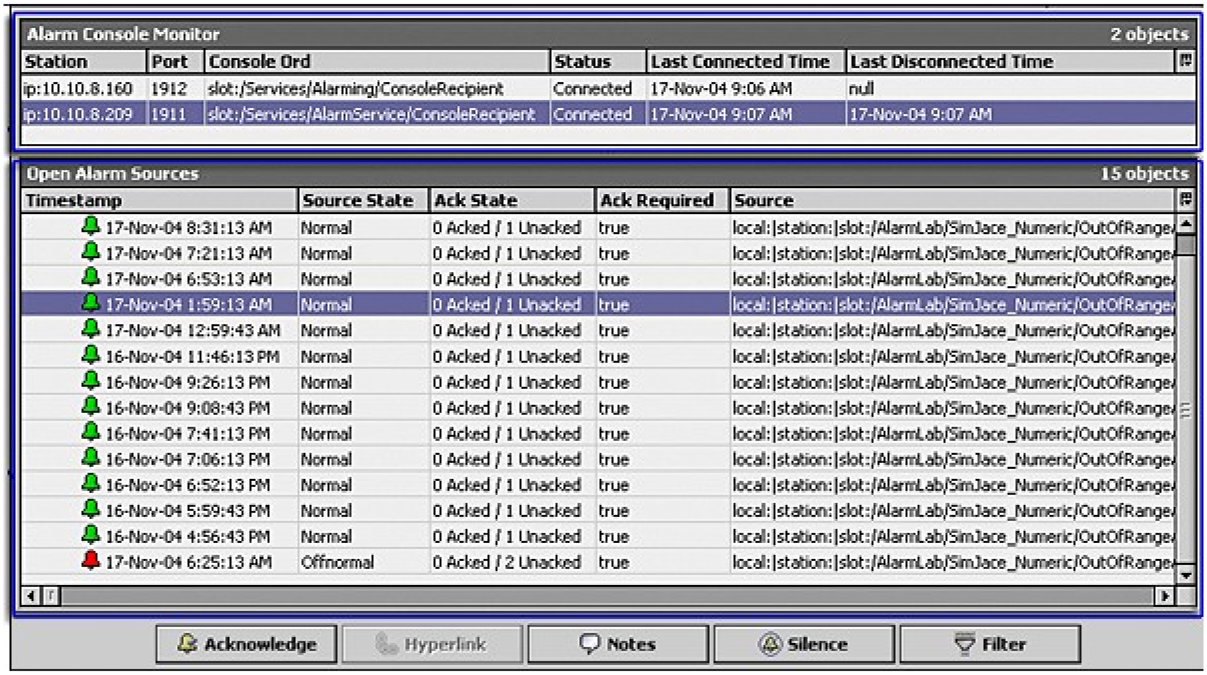
This section describes some of the standard tools available in Workbench. These tools provide views that are designed to facilitate specific tasks – from managing credentials to monitoring alarms. All of the tools described are available from the Tools menu. However, navigational links to some them also appear in the Nav side bar and in the Nav Container views.

When you first open Workbench, the My Tools node will not be present in the Nav tree. Therefore, if you configure a tool, such as the BACnet service tool, all the configuration data are lost when you close the current session. Tools, such as the Alarm Portal, BACnet service, Lonworks service, and the Workbench Job Service are available when you do not have a station open. This is provided as a convenience and may only be useful in a limited number of situations. For example, you would probably use the Job Service, BACnet service, and Lonworks Service under the Services node of a specific station.

Alarm Portal

The Alarm Portal tool allows you to view and acknowledge alarms from many different stations using a single viewer (portal).

Figure 105: Alarm portal



Alarm Console Monitor — 2 objects

Station	Port	Console Ord	Status	Last Connected Time	Last Disconnected Time
ip:10.10.8.160	1912	slot:/Services/Alarming/ConsoleRecipient	Connected	17-Nov-04 9:06 AM	null
ip:10.10.8.209	1911	slot:/Services/AlarmService/ConsoleRecipient	Connected	17-Nov-04 9:07 AM	17-Nov-04 9:07 AM

Open Alarm Sources — 15 objects

Timestamp	Source State	Ack State	Ack Required	Source
17-Nov-04 8:31:13 AM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
17-Nov-04 7:21:13 AM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
17-Nov-04 6:53:13 AM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
17-Nov-04 1:59:13 AM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
17-Nov-04 12:59:43 AM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 11:46:13 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 9:26:13 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 9:08:43 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 7:41:13 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 7:06:13 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 6:52:13 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 5:59:43 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
16-Nov-04 4:56:43 PM	Normal	0 Acked / 1 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange
17-Nov-04 6:25:13 AM	Offnormal	0 Acked / 2 Unacked	true	local:|station:|slot:/AlarmLab/SimJace_Numeric/OutOfRange

The alarm portal has two resizable panes:

- The top pane, Alarm Console Monitor, lists the stations displayed in the lower pane, Open Alarm Sources.

To add alarm consoles from different stations, right click in the Alarm Console Monitor (top pane) and choosing the Add Alarm Console menu selection to initiate the Add Alarm Console wizard.

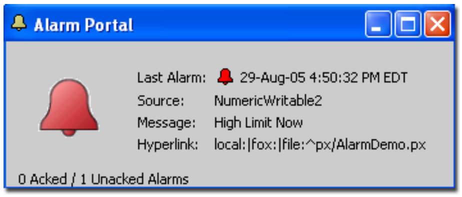
To view individual alarm record to see all information on the alarm, select an alarm and double-click it to see the Viewing Alarm Record.

Available commands include the following

To see alarm details, double-click on any alarm listed in the Open Alarm Sources pane.

The alarm portal tool, when enabled, also places the alarm icon in your system tray and an alarm popup window.

Figure 106: Alarm portal popup window



Last Alarm: 29-Aug-05 4:50:32 PM EDT
Source: NumericWritable2
Message: High Limit Now
Hyperlink: local:|fox:|file:^px/AlarmDemo.px

0 Acked / 1 Unacked Alarms

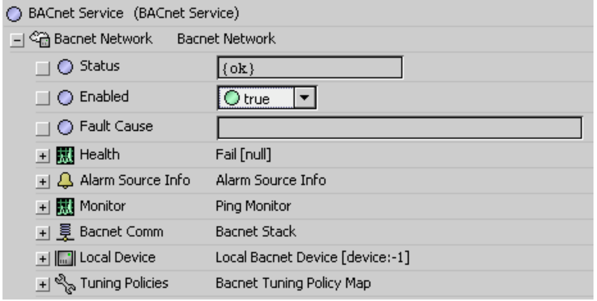
To set options for the alarm portal, click the Tools→Options menu, as described in the section, “Alarm Portal options”.

Bacnet Service

Following is an example of the Bacnet Service property sheet. Select Tools→ BACnet Service from the Workbench main menu to add this service to the My Tools node of the Nav tree.

The default view is the property sheet view.

Figure 107: Bacnet service property sheet view



BACnet Service (BACnet Service)	
Bacnet Network	Bacnet Network
Status	{ok}
Enabled	true
Fault Cause	
Health	Fail [null]
Alarm Source Info	Alarm Source Info
Monitor	Ping Monitor
Bacnet Comm	Bacnet Stack
Local Device	Local Bacnet Device [device:-1]
Tuning Policies	Bacnet Tuning Policy Map

Note:

You can use this Workbench Tools service to configure a local Bacnet device in the same way as you would under a station. However, any configuration that you do using the Workbench Bacnet Service (under the My Tools node) will be lost when you close Workbench. Use Bacnet Service under a station to save your Bacnet Service settings to a database.

Security management

Two tools are available in Workbench to manage and sign PKI (Public Key Infrastructure) certificates.

You use the Certificate Management view to secure communication within a NiagaraNetwork. Certificates secure TLS connections to the host.

You use the Certificate Signer Tool to sign intermediate digital certificates.

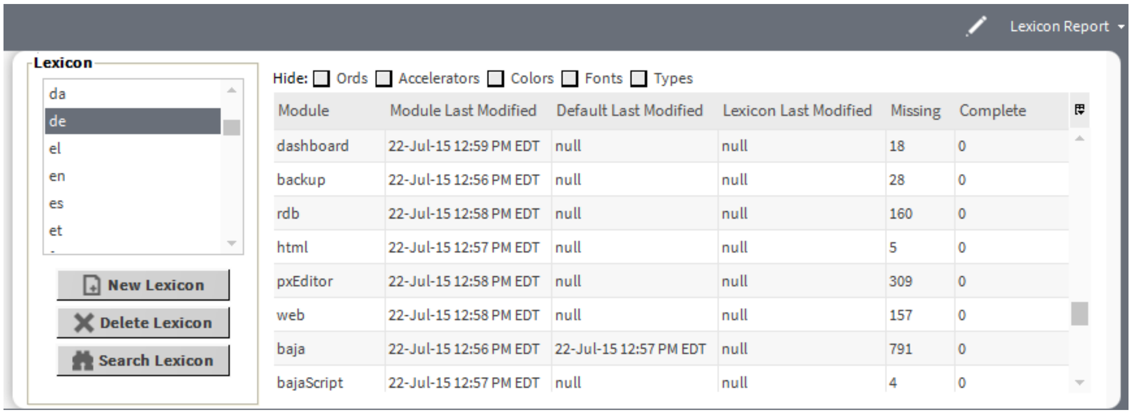
Lexicon Tool

This tool controls the text of the user interface. You can use it to translate the user interface. See the section, “About lexicons.

Lexicon Report view

The Lexicon Report view is available via the Workbench Tools menu, by selecting Lexicon Tool. On the left side of this view, the Lexicon pane shows a list of all module-based and file-based lexicon locales installed on your Workbench PC. As needed, you click one of these lexicons to select it. Selecting a lexicon shows various status parameters about each lexicon (broken down by module) in the table columns on the right side.

Figure 108: Lexicon report view



Hide: ☐ Ords ☐ Accelerators ☐ Colors ☐ Fonts ☐ Types

Module	Module Last Modified	Default Last Modified	Lexicon Last Modified	Missing	Complete
dashboard	22-Jul-15 12:59 PM EDT	null	null	18	0
backup	22-Jul-15 12:56 PM EDT	null	null	28	0
rdb	22-Jul-15 12:58 PM EDT	null	null	160	0
html	22-Jul-15 12:57 PM EDT	null	null	5	0
pxEditor	22-Jul-15 12:58 PM EDT	null	null	309	0
web	22-Jul-15 12:58 PM EDT	null	null	157	0
baja	22-Jul-15 12:56 PM EDT	22-Jul-15 12:57 PM EDT	null	791	0
bajaScript	22-Jul-15 12:57 PM EDT	null	null	4	0

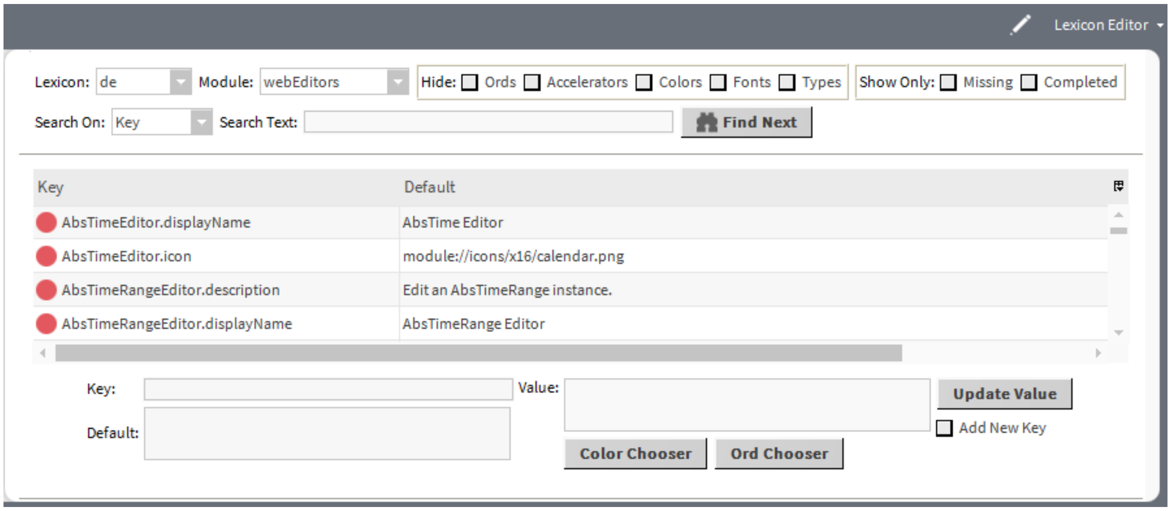
Check boxes at the top of report view allow you to “hide” values that could affect “missing” status counts.

This view lets you add (new) lexicons, delete unwanted lexicons, or search all available lexicon property values that contain a given text. Additionally, you can double-click any module row in the table to launch the Lexicon Editor view, which shows the contents of that lexicon file.

Lexicon Editor view

The Lexicon Editor view lets you view and edit the contents of lexicon files installed on your Workbench PC. Typically, you access the editor from the Lexicon Report view with a lexicon selected on the left side, by double-clicking a module in the table on the right side. The lexicon editor displays a table listing the defined keys for that module, with columns for mapping to a localized value override for the default value.

Figure 109: Lexicon editor view



Lexicon: de Module: webEditors Hide: □ Ords □ Accelerators □ Colors □ Fonts □ Types Show Only: □ Missing □ Completed

Search On: Key Search Text:

Key	Default
● AbsTimeEditor.displayName	AbsTime Editor
● AbsTimeEditor.icon	module://icons/x16/calendar.png
● AbsTimeRangeEditor.description	Edit an AbsTimeRange instance.
● AbsTimeRangeEditor.displayName	AbsTimeRange Editor

Key: Value:

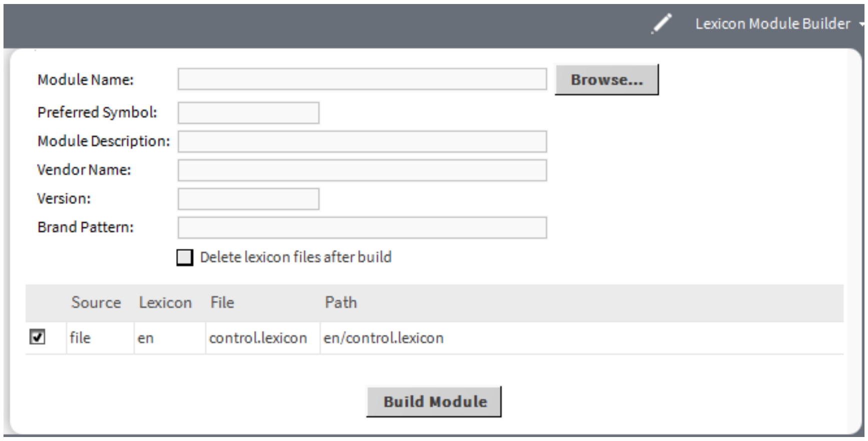
Default: □ Add New Key

At the top of this view, you can search for specified text on the lexicon property Key, the property Default, or the custom Value. Clicking any row in the table populates the Key field. At the bottom of the view, clicking the Add New Key button enables the Key field, as well. The Value text field allows you to modify the value of the selected key (or add a value for a new key). Starting in AX-3.7 the Value field has associated Color Chooser and Ord Chooser buttons. These options allow you to browse for a specific color or ORD element. Update the table with the Value field entry by clicking the Update Value button. Write changes to the lexicon file by clicking the Save button in the Niagara toolbar.

Lexicon Module Builder

The Lexicon Module Builder view allows you to bundle multiple lexicon files into a module for ease of distribution. The lexicons available for use are those lexicon files (modulename.lexicon) located in the !lexicon folder. You can build new lexicon modules or replace existing ones.

Figure 110: Lexicon module builder view



Module Name:
Preferred Symbol:
Module Description:
Vendor Name:
Version:
Brand Pattern:

☐ Delete lexicon files after build

	Source	Lexicon	File	Path
☑	file	en	control.lexicon	en/control.lexicon

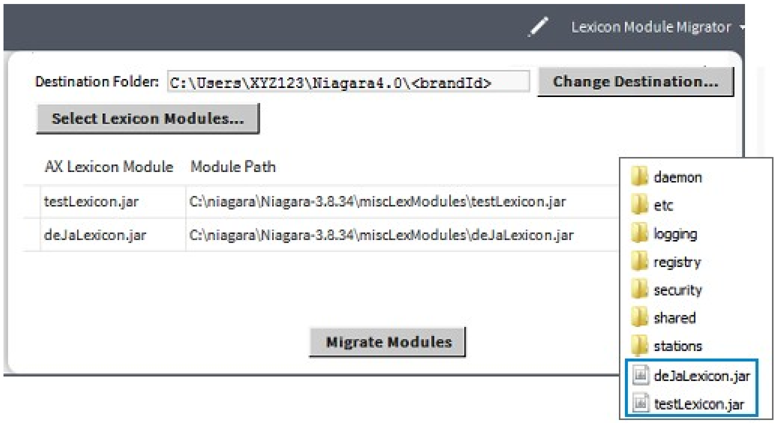
In this view there are text fields for defining module information, a Browse button that allows you to locate existing lexicon modules, a selection table of available lexicon files that can be selected for inclusion, a check box to delete source files after the build is completed, and a Build Module button to initiate the build.

On completion of the build, you will see a confirmation message indicating that the module was constructed and placed in the !modules folder.

Lexicon Module Migrator

The Lexicon Module Migrator view (shown) allows you to migrate AX-3.8 lexicon modules to a Niagara 4.0 installation.

Figure 111: Lexicon Module Migrator view and migrated lexicon modules in user home folder



Destination Folder: C:\Users\XYZ123\Niagara4.0\<brandId>

AX Lexicon Module	Module Path
testLexicon.jar	C:\niagara\Niagara-3.8.34\miscLexModules\testLexicon.jar
deJaLexicon.jar	C:\niagara\Niagara-3.8.34\miscLexModules\deJaLexicon.jar

You can select modules to migrate from anywhere in the PC file system. By default, the destination for migrated lexicon files is the Workbench user home. However, you can change the destination by selecting an alternate location.

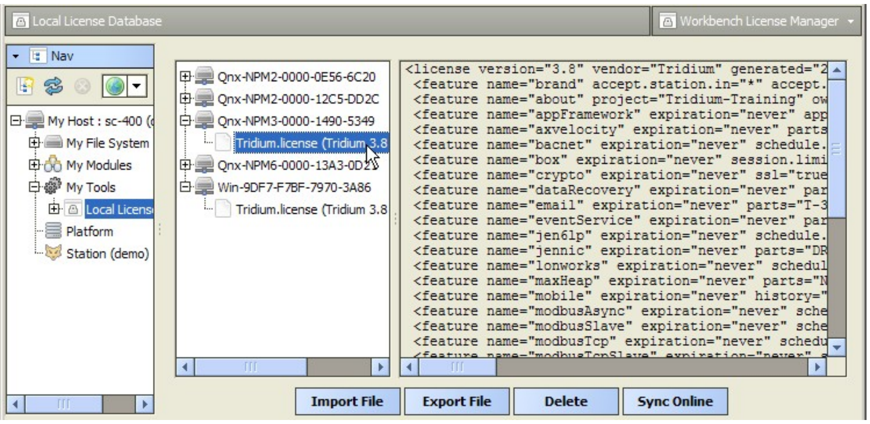
Note:

For any modules that are new in Niagara 4.0, such as webEditors, etc., you need to build new lexicon modules using the Lexicon Module Builder view.

Local License Database

The EC-Net 4 Pro License Manager view is available via the Workbench Tools menu, by selecting Local License Database.

Figure 112: Workbench License Manager



```
<license version="3.8" vendor="Tridium" generated="2
 <feature name="brand" accept.station.in="*" accept.
 <feature name="about" project="Tridium-Training" ow
 <feature name="appFramework" expiration="never" app
 <feature name="axvelocity" expiration="never" parts
 <feature name="bacnet" expiration="never" schedule.
 <feature name="box" expiration="never" session.limi
 <feature name="crypto" expiration="never" ssl="true
 <feature name="dataRecovery" expiration="never" par
 <feature name="email" expiration="never" parts="T-3
 <feature name="eventService" expiration="never" par
 <feature name="jen6lp" expiration="never" schedule.
 <feature name="jennic" expiration="never" parts="DR
 <feature name="lonworks" expiration="never" schedul
 <feature name="maxHeap" expiration="never" parts="N
 <feature name="mobile" expiration="never" history="
 <feature name="modbusAsync" expiration="never" sche
 <feature name="modbusSlave" expiration="never" sche
 <feature name="modbusTcp" expiration="never" schedu
```

As shown, this view lets you browse and manage the contents of your “local license database”.

This view provides a two-pane window into all the license files and parent “host ID” folders, where

- Left pane provides tree navigation, where you can expand folders and click (to select) license files.
- Right pane shows the text contents of any selected license file.

Buttons at the bottom of this view provide a way to manage the contents of your local license database, and are described as follows:

- Import File: Always available, this allows you to add license file(s) from a local license file or license archive (.lar) file.
- Export File: Always available, this allows you to save all licenses (or any selected licenses) locally, as a license archive file.
- Delete: This allows you to delete licenses from your Niagaralocal license database.
- Sync Online: Typically available if you have Internet connectivity. This lets you update all licenses (or any selected licenses) in your local license database with the most current versions, via the online licensing server.

Logger Configuration tool

The Workbench Logger Configuration tool allows you to manage log settings for the local Workbench. Using the Logger Configuration view you can add and remove log categories and change the applied severity level, which determines the amount of data that is displayed.

Logging is an effective tool for troubleshooting and debugging station communications problems. Each driver and device installed on the connected station has a log category based on its type.

In Niagara 4 logging is handled by the Java.util.logging (JUL) utility which supports multiple concurrent log handlers and provides hierarchical logging support. Also in systems without a Workbench connection, this capability allows you to enable/disable loggers using a web browser.

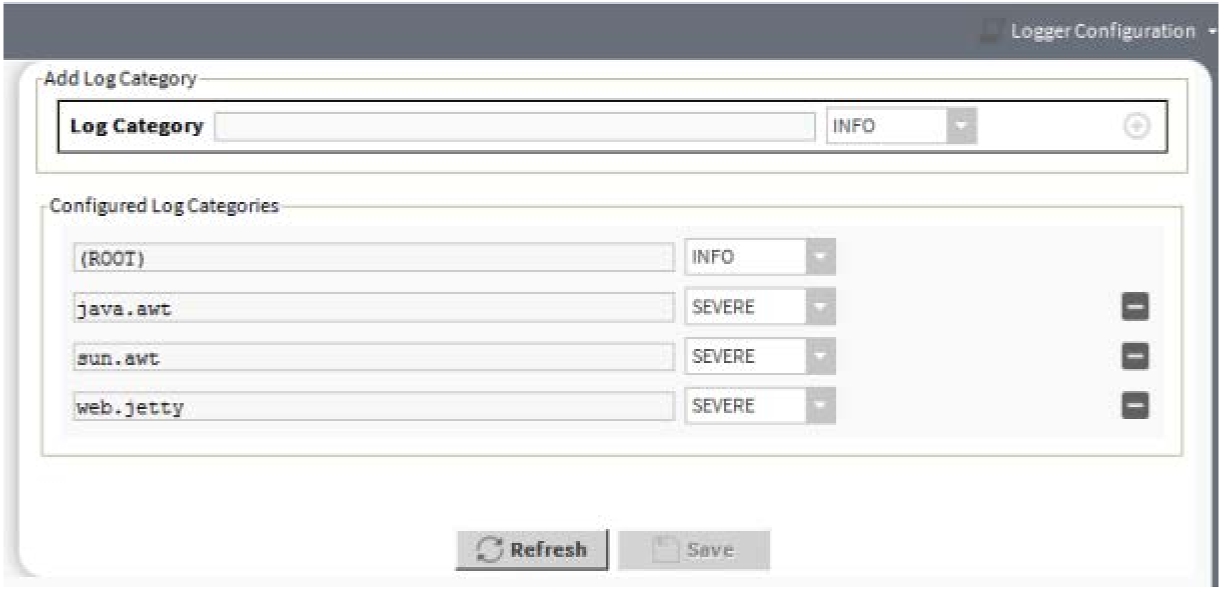
Logging configuration settings for local or Workbench logs are stored in the !logging/logging.properties file which exists in two locations, your Workbench User Home and the daemon User Home, and are maintained by the framework.

The Logger Configuration tool has seven severity levels which differ from the number of log levels used in NiagaraAX. Log levels used are mapped as shown in the following table.

NiagaraAX Log Levels	Niagara 4 Log Levels
	Off
Error	Severe
Warning	Warning
Message	Info
	Config
Trace	Fine
	Finer
	Finest
	All

The Logger Configuration view is the main view for the Workbench Logger Configuration tool, as well as for the station DebugService.

Figure 113: Logger Configuration view



Add Log Category

Log Category: INFO

Configured Log Categories

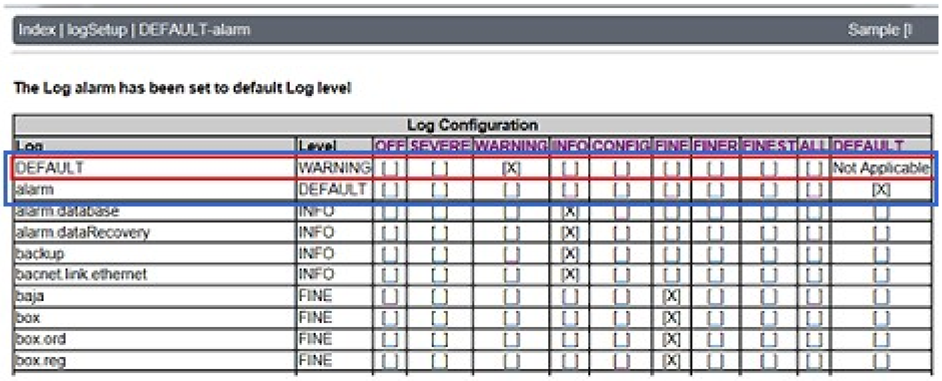
(ROOT)	INFO
java.awt	SEVERE
sun.awt	SEVERE
web.jetty	SEVERE

Logs for the connected station are visible in the platform Application Director view. While logs for Workbench are visible in a Niagara console started with Workbench.

The logs are persisted each time they are changed, since the changes are saved to the logging.properties file.

The “ROOT” log category is essentially a default log level. If any log is set to severity log level "default" then it uses the same configured severity level selected for the Root of that particular log. For example, the following diagram setting alarm to the default log level means that the alarm log level has been set to warning (the configured level of default row).

Figure 114: Alarm log



The Log alarm has been set to default Log level

Log Configuration											
Log	Level	OFF	SEVERE	WARNING	INFO	CONFIG	FINE	FINER	FINEST	ALL	DEFAULT
DEFAULT	WARNING	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	Not Applicable
alarm	DEFAULT	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[X]
alarm.database	INFO	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
alarm.dataRecovery	INFO	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
backup	INFO	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
bacnet.link.ethernet	INFO	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
baja	FINE	[ ]	[ ]	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]
box	FINE	[ ]	[ ]	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]
box.ord	FINE	[ ]	[ ]	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]
box.reg	FINE	[ ]	[ ]	[ ]	[ ]	[ ]	[X]	[ ]	[ ]	[ ]	[ ]

Both Local (Workbench) Spy and Remote (Station) Spy can be accessed via Workbench by right-clicking the station in the Nav tree. Only Remote (Station) Spy can be accessed via a web browser using the url:
http (s)://<ip address>:<port number>/ord?spy:

Configuring settings for local Workbench logs

Use the Logger Configuration tool in Workbench Tools to modify logging settings for local Workbench logs. For example, you can add and remove log categories, or change the severity level for an existing log in order to collect more data.

Note:

The procedures to configure log settings of a station are identical except that you must access the Logger Configuration view via the DebugService in the connected station.

Adding a new log category

1. In Workbench, select Tools→Logger Configuration. The Logger Configuration view appears.

2. Click in the Add Log Category field and enter the new category name. Or, enter only the first letter of a name to see a list of existing logs (whose names start with that letter) to choose from.

3. Click the severity level dropdown list and select the desired log level.

4. Click ⊕ (add) to add the new category.

5. Click Save.

Removing an existing log category

1. In Niagara, select Tools→Logger Configuration. The Logger Configuration view appears.

2. Click ⊟ (remove button), located to the right of the log category you wish to remove. The log category is immediately removed.

3. Click Save.

Changing the severity level of an existing log category

1. In Workbench, select Tools→Logger Configuration. The Logger Configuration view appears.

2. Click the current severity level and select the desired level from the dropdown list.

3. Click Save.

Viewing logged data

You can view logged data for stations and for Workbench.

Viewing logged data for the station

To view logged data for the station perform either of the following:

- For a station connection in Workbench, open the platform Application Director view.
- For a browser connection to the station, view “stdout” on the Spy menu

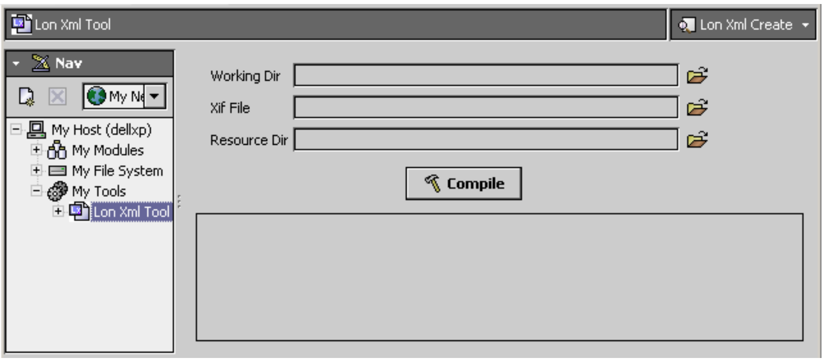
Viewing logged data for Workbench

To view logged data for Workbench, open a Niagara console started with Workbench.

Lon Xml Tool

The Lon Xml Tool is available from the Tools menu. The Lon Xml Create view helps you make your own Lon Xml (.lnml) file for a device, using the source .xif file and (if necessary) other resource files, as available from the device's manufacturer. The tool's dialog provides the necessary input fields.

Figure 115: Lon Xml Create tool view



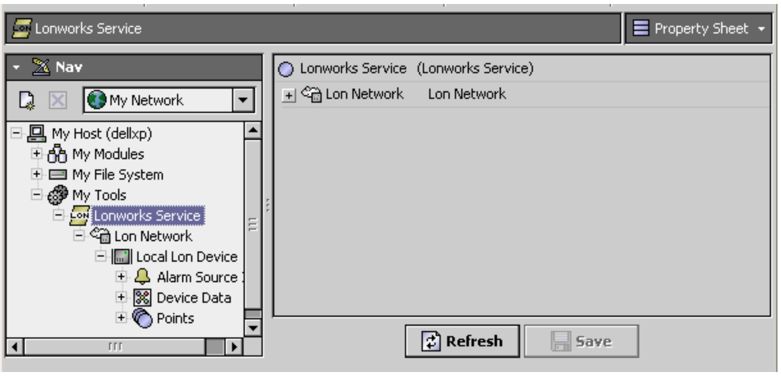
Refer to the Lonworks Guide for details about using this view and about the following:

- Need for custom Lon Xml files
- Lon Xml file overview
- Lon Xml creation
- Storing .lnml files
- Differential temperatures and lnml file edits

Lonworks Service

The Lonworks Service is applicable if your Workbench PC has a Lonworks adapter. You can view it by selecting Lonworks Service from the Tools menu. It provides the identical "LonNetwork" access as if you had a local (PC) station running.

Figure 116: Lonworks Service tool in Workbench



For example, you can use the Lon Device Manager to discover, add, and upload Lon devices, examine nvs and ncis as LonComponents, and even perform writes and do other configuration (make bindings between devices, commission devices, etc.).

Note:

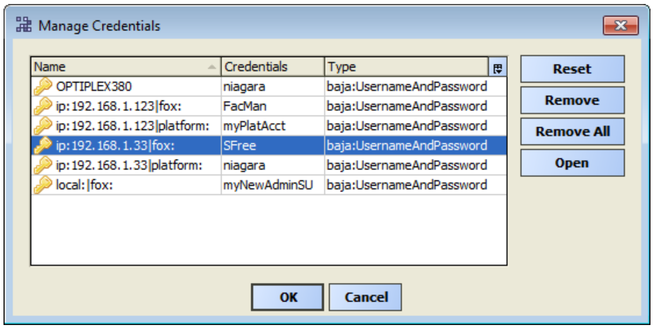
- Any configuration you perform is not stored (persisted) in a station database, as this is “station-less” access (modeled completely in RAM). When you close Workbench, all modeling is lost, consider the Lonworks Service like a “hand held device” in this respect.

For this reason, do not use this tool to access a Lonworks network already managed by a station, but instead open that station, and access its LonNetwork.

Credentials manager

The Manage Credentials tool, or Credentials Manager provides a window to access and open any previously “remembered” (cached) connections from your Workbench, including both platform and station connections.

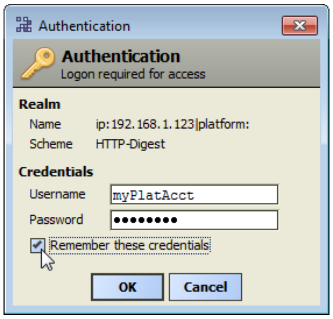
Figure 117: Manage Credentials dialog (credentials manager)



Name	Credentials	Type
OPTIPLEX380	niagara	baja:UsernameAndPassword
ip:192.168.1.123|fox:	FacMan	baja:UsernameAndPassword
ip:192.168.1.123|platform:	myPlatAcct	baja:UsernameAndPassword
ip:192.168.1.33|fox:	SFree	baja:UsernameAndPassword
ip:192.168.1.33|platform:	niagara	baja:UsernameAndPassword
local:|fox:	myNewAdminSU	baja:UsernameAndPassword

You can also remove or reset any cached credentials. Note that your credentials cache is populated whenever you have the login (Authentication) dialog option “Remember these credentials” (check box) enabled.

Figure 118: Authentication (Remember credentials)



Realm

Name ip:192.168.1.123|platform:

Scheme HTTP-Digest

Credentials

Username myPlatAcct

Password ••••••••

☑ Remember these credentials

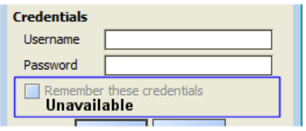
The available caching of credentials is a convenience feature, such that you can simply open the platform or station later by entering only the IP address, or simply clicking on that host’s dimmed platform or station in the Nav tree of Workbench, or going to the credentials manager. Then, the login authentication dialog has the cached credentials already entered, and you simply click OK.

If you want tighter security for any platform or station connection from your Workbench PC, you should clear this check box whenever opening (logging on) a platform or station. Furthermore, you should remove any related entries using the credentials manager. This way, that platform or station connection always requires full entry of both user name and password.

Note:

You can globally disable user credentials caching in Workbench via Tools→Options, in the General menu. When Allow User Credentials Caching is set to false, the Remember these credentials checkbox remains unavailable (dimmed) in any Authentication dialog.

Figure 119: Authentication dialog



For related details, see the section, “General options”.

NDIO to NRIO Conversion Tool

Starting in Niagara 4.2, you can use the NDIO to NRIO Conversion Tool to easily convert all NDIO objects to NRIO objects. This is necessary when replacing a legacy JACE model with a JACE-8000 which only supports NRIO. While NRIO and NDIO objects have similar configuration properties, they are not interchangeable. So you cannot simply move objects from NDIO modules to NRIO modules, you must run the conversion tool.

The NDIO to NRIO Conversion Tool is available when you select Tools→NDIO to NRIO Conversion Tool from the menu bar.

When the conversion is run, the following actions occur:

- Each NDIO Network is converted to an NRIO Network. Any instance of "NDIO" in the name (case insensitive) is replaced with "NRIO" (preserving case) and the name is appended with "_converted"
- Each NDIO type object under an NDIO network is converted into an NRIO type object
- All dynamic properties and relevant static properties are copied to the new objects
- Any NDIO device with more than 8 universal inputs, 4 digital outputs, or 4 analog outputs (NDIO-34) is split into multiple (1- to 3-) NRIO-16 devices, depending on the number of points. Points are split among each new device so no device exceeds the maximum amount of points, and addresses are updated so they are in the allowed range. The points in each new device retain their previous folder structure.
  - Universal inputs (UI) 1-8, analog outputs (AO) 1-4, and digital outputs (DO) 1-4 are moved to device 1. Any miss-configured points with an address less than 1 are moved to device 1.
  - Any non-NDIO point objects are copied to device 1 only

    – UI 9-16, AO 5-8, and DO 5-8 are moved to device 2

    – DO 9-10 are moved to device 3
- Any component under an NDIO network with a name that contains "NDIO" (case insensitive) are replaced by "NRIO" (preserving case).
- Any links to/from the NDIO objects are updated to reference the new NRIO objects.
- If running N4 or later, any relations to/from the NDIO objects are updated to reference the new NRIO objects. For usage details, see the procedure “Converting NDIO modules to NRIO”.

Converting NDIO modules to NRIO

The NDIO to NRIO Conversion Tool converts all NDIO objects to NRIO objects. This is necessary when replacing a legacy JACE with a JACE-8000 which does not support NDIO.

Prerequisites:

- The appropriate conversion module must be installed on your PC:
  - N4 and later installations require: nrioConversion-wb
  - AX-3.8U2 installations require: nrioConversion
- An existing station that has an NDIO network configured with points (regardless of whether or not the host supports NDIO).

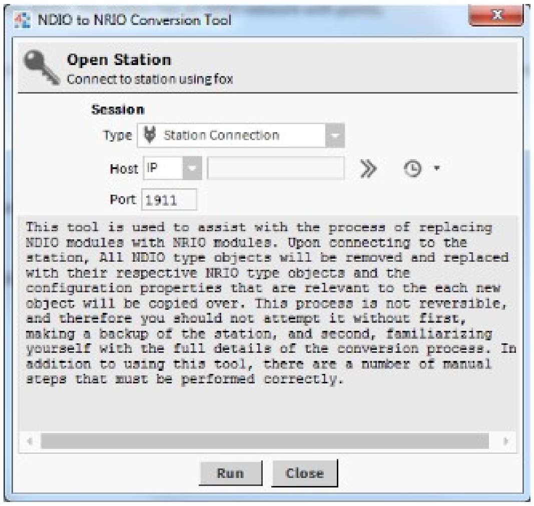
Note:

This process is not reversible, therefore you should not attempt it without first making a backup of the station and familiarizing yourself with the full details of the conversion process.

Step 1 Backup the JACE station containing the NDIO module(s) to be converted.

Step 2 Click Tools→ NDIO to NRIO Conversion Tool. The NDIO to NRIO Conversion Tool dialog displays, as shown here.

Figure 120: NDIO to NRIO Conversion Tool



Type: Station Connection
Host: IP
Port: 1911

```
This tool is used to assist with the process of replacing
NDIO modules with NRIO modules. Upon connecting to the
station, All NDIO type objects will be removed and replaced
with their respective NRIO type objects and the
configuration properties that are relevant to the each new
object will be copied over. This process is not reversible,
and therefore you should not attempt it without first,
making a backup of the station, and second, familiarizing
yourself with the full details of the conversion process. In
addition to using this tool, there are a number of manual
steps that must be performed correctly.
```

Step 3 Enter the following connection values:

- Change to a secure Station TLS Connection type, if applicable.
- Host: <IP address>

Step 4 Click Run.

- If already connected to the station you are not prompted for credentials.
- If not connected, an Authentication dialog displays. Enter your credentials and click OK to continue.

Note:

When running NiagaraAX, the initial NDIO to NRIO Conversion Tool dialog prompts you to enter your credentials regardless of whether or not you are connected to the station.

Step 5 Click Close.

Step 6 Check the NRIO Network to confirm that all NDIO objects were converted.

Note:

Any NDIO-34 module that has more points than an NRIO-16 module can accommodate will be split into multiple devices. In this case, it may make sense to move points between devices.

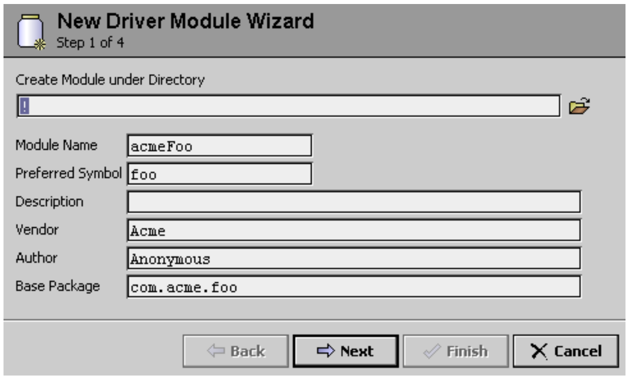
Step 7 Open the NRIO Device Manager for one of the new NRIO Networks. Use the Discover and Match functions to match the converted NRIO devices to the connected ones. Repeat this step for each network.

The conversion process is now complete.

New Driver wizard

The New Driver Module Wizard is available when you select Tools→ New Driver from the main menu. The first of four wizard dialog boxes is shown below.

Figure 121: New driver wizard



New Driver Module Wizard
Step 1 of 4

Create Module under Directory

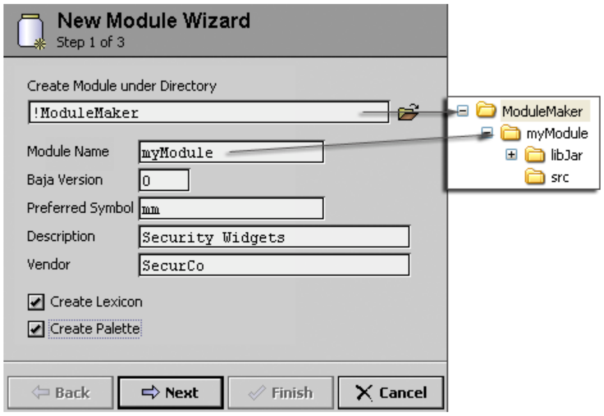
Module Name	acmeFoo
Preferred Symbol	foo
Description	
Vendor	Acme
Author	Anonymous
Base Package	com.acme.foo

This wizard is provided as a convenient way to perform some of the steps necessary for creating a driver module. The four wizard dialog boxes allow driver developers to specify some basic driver module options and finish with the wizard creating a set of folders and files under a station directory that you assign during the wizard process.

New Module wizard

The New Module Wizard is available when you select Tools→ New Module. The wizard presents a series of dialog boxes that help you create and save a new module.

Figure 122: New module wizard creates folders under working directory



New Module Wizard
Step 1 of 3

Create Module under Directory: !ModuleMaker
Module Name: myModule
Baja Version: 0
Preferred Symbol: mm
Description: Security Widgets
Vendor: SecurCo

☑ Create Lexicon
☑ Create Palette

This wizard creates a set of folders and files under a station directory that you assign during the wizard process. After creating these files, you need to finish the process, as described in “Working with modules”.

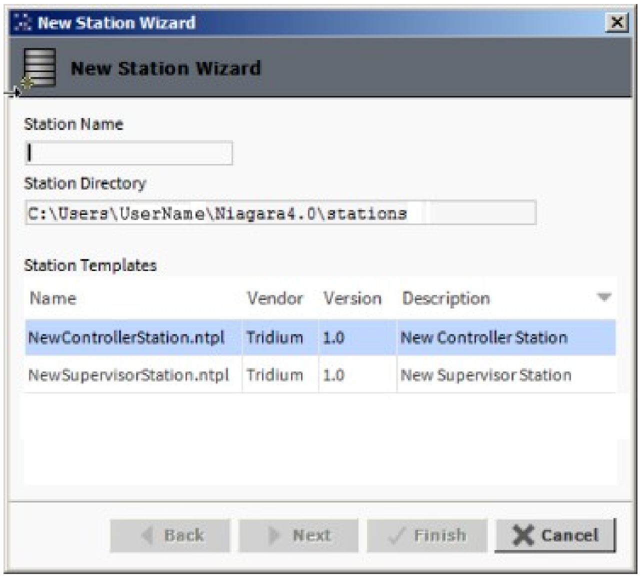
New Station wizard

The New Station Wizard available in the Tools menu uses templates to define the starting contents of a new station. By default, the wizard provides two station templates from which you can choose to create either a new controller (JACE) station or a new Supervisor station. Additionally, the list of templates shown in the wizard dialog automatically includes any user-defined station templates that are available. The wizard configures the new station with services and components as determined by the selected template.

Note:

At present, there is no difference between the two default station templates (aside from the names), they create identical stations. However, the configuration of these templates may vary in futureNiagara distributions or an OEM may distribute different default templates.

Figure 123: New Station tool



Station Name

Station Directory

C:\Users\UserName\Niagara4.0\stations

Station Templates

Name	Vendor	Version	Description
NewControllerStation.ntpl	Tridium	1.0	New Controller Station
NewSupervisorStation.ntpl	Tridium	1.0	New Supervisor Station

When creating a new station, if you enter a station name identical to that of an existing station, the wizard alerts you that the station exists and prompts whether or not you wish to delete the existing station. The functionality gives you an opportunity to change the new station name, if you wish, or confirm that you intend to delete the existing station.

The wizard also performs entry validation on the password fields in the second dialog, alerting you to errors. By default, the system requires “strong passwords”. In order to comply, values entered in the Admin Password and Admin Password Confirm fields must be identical and meet the following password criteria:

- at least 10 characters
- at least 1 digit
- at least 1 lowercase character
- at least 1 uppercase character

The options for which action to take when you click Finish are determined by whether and how you have made platform connections to localhost before. One or more of the following options are presented:

- Open it in user home: Selected by default, on completion the station is created in your Niagara User Home directory and a property sheet view of the station’s config.bog displays.

At this point the new station exists only in your User Home directory (your Niagara User Home) and not in the User Home of the local Niagaraplatform daemon.

- Copy it to the platform for “localhost” with Station Copier: On completion, the station is created in your Workbench User Home directory. Then you are prompted to login to make a local platform connection. After you login, the Station Copier starts the transfer (copy). After the station is copied, the Application Director opens with the new station visible in the daemon’s User Home.

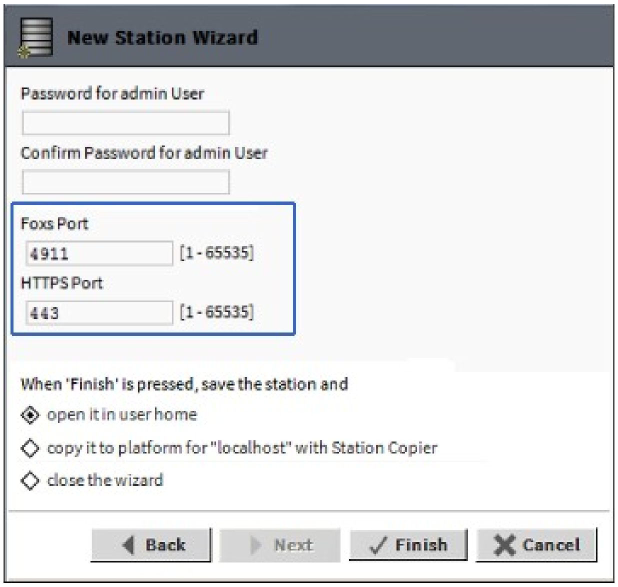
At this point the new station exists in two locations: on your local host (in your Workbench User Home) and also in the Niagara platform daemon User Home.

Note:

If you make any changes to the station now (while it is running on the platform daemon User Home), it is a good idea to copy the station back to your Workbench User Home so that you have a local copy of the updated station. This is useful if you plan to install it on any remote platform.

- Close the wizard: On completion, the station is created in your Workbench User Home directory, the wizard closes, and an alert dialog appears notifying you of successful station creation.

Figure 124: Default station templates contain configurable Foxs and HTTPS Port parameters



Password for admin User

Confirm Password for admin User

Foxs Port
4911 [1 - 65535]

HTTPS Port
443 [1 - 65535]

When 'Finish' is pressed, save the station and

● open it in user home

○ copy it to platform for "localhost" with Station Copier

○ close the wizard

In cases where the selected station template contains exposed (configurable) parameters, those are included as editable fields on the second wizard dialog, such as the port parameters indicated in the above image. This gives you the opportunity to modify those values as needed to complete the configuration for your new station.

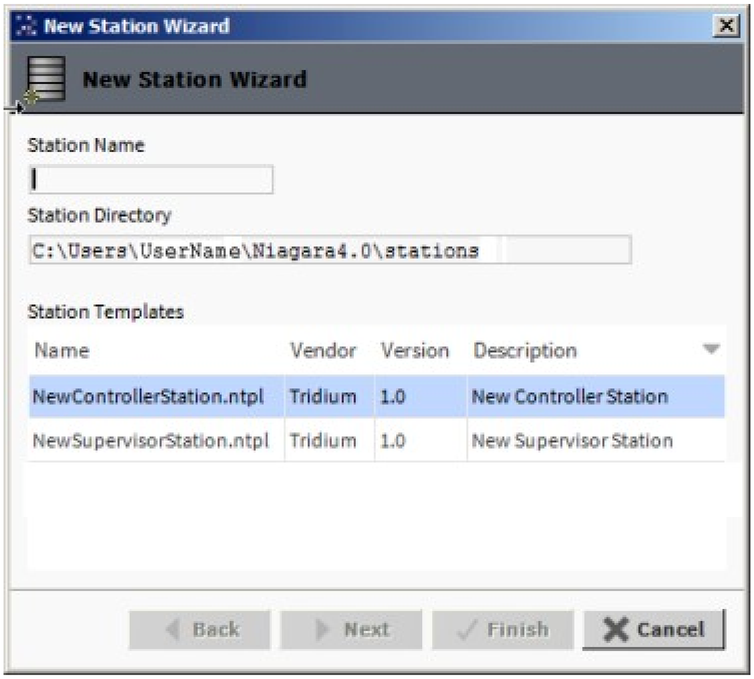
Both of the default station templates (controller, supervisor) have these two port configuration parameters. However, you can make a station template that provides different parameters or none at all. Whether the wizard dialog displays alternative parameters, additional parameters, or no parameters is determined by the station template. If there are parameters they display in this space. When there are more parameters than the space allows, a scroll pane appears.

Creating a new station

Use the New Station tool in Workbench Tools to create a new controller station or Supervisor station. The new station is automatically configured with appropriate services.

Step 1 In Workbench, select Tools→New Station. The New Station Wizard opens.

Figure 125: New Station Wizard



Station Name

Station Directory

C:\Users\UserName\Niagara4.0\stations

Station Templates

Name	Vendor	Version	Description
NewControllerStation.ntpl	Tridium	1.0	New Controller Station
NewSupervisorStation.ntpl	Tridium	1.0	New Supervisor Station

Step 2 In the New Station Wizard do the following:

a. Enter the Station Name

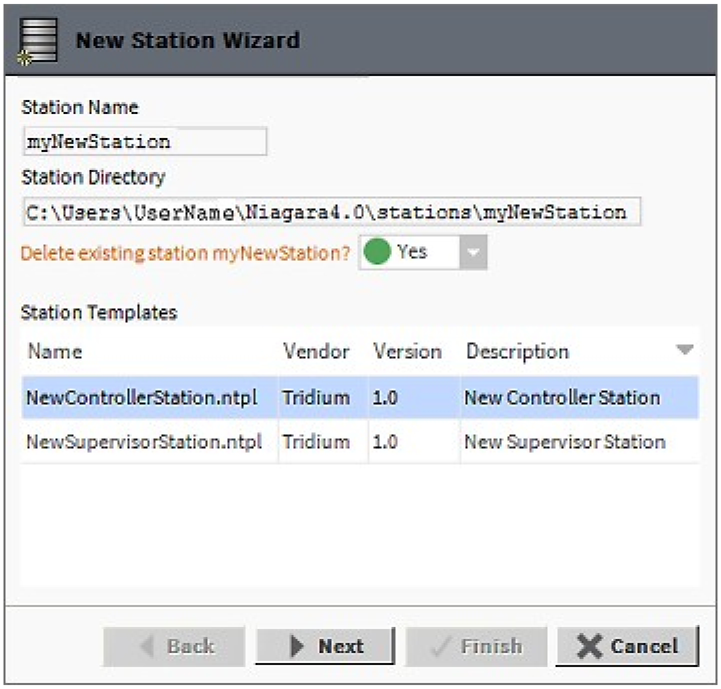
The station name field is case-sensitive and must begin with a letter. Best practice is to the keep station name short, use a station “display name” if a longer name with spaces or other characters is required.

Note:

The Station Name text that you enter is automatically appended to the read-only Station Directory field to create a directory of the same name.

Also, if you enter a duplicate station name you are prompted about whether to delete the existing station. You can either delete the existing station or provide a different station name.

Figure 126: New Station Wizard



New Station Wizard

Station Name: myNewStation

Station Directory: C:\Users\UserName\Niagara4.0\stations\myNewStation

Delete existing station myNewStation? Yes

Station Templates

Name	Vendor	Version	Description
NewControllerStation.ntpl	Tridium	1.0	New Controller Station
NewSupervisorStation.ntpl	Tridium	1.0	New Supervisor Station

b. Select a Station Template type.

The Station Templates table contains the default new station templates provided in Workbench as well as any user defined templates.

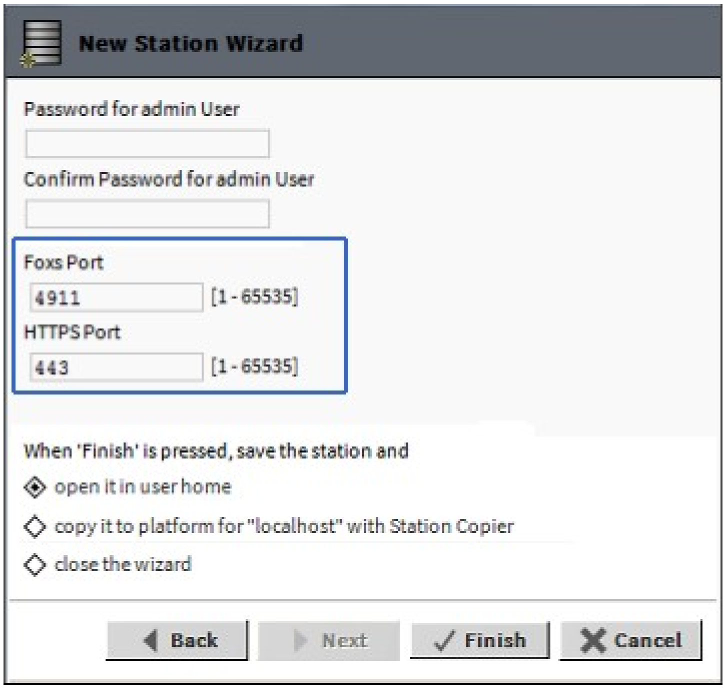
c. Click Next.

Step 3 The second dialog prompts for the admin user password, configuration information relevant to the station template, and allows you choose an action to take once the station creation is completed.

a. Enter Admin Password and Admin Password Confirm.

b. Modify the Foxs and HTTPS port numbers as needed.

Figure 127: New Station Wizard



New Station Wizard

Password for admin User

Confirm Password for admin User

Foxs Port

4911 [1 - 65535]

HTTPS Port

443 [1 - 65535]

When 'Finish' is pressed, save the station and

◆ open it in user home

◇ copy it to platform for "localhost" with Station Copier

◇ close the wizard

If the new Station Template selected in Step 1 contains any exposed configurable properties, such as the ports indicated in the above image, this dialog presents those properties allowing you to modify the values as needed.

c. Select an option for the preferred action on completion.

d. Click Finish.

The New Station Wizard closes. If this is the default option, open it in user home, is selected (as shown in the above image) a Property Sheet view of the new station config.bog file displays.

Creating a station template

Create a user-defined station template from a configured station that contains everything needed for the initial starting point of a new station. Once created, the station template is added to the list of selectable templates presented in the Workbench New Station Wizard.

Prerequisites:

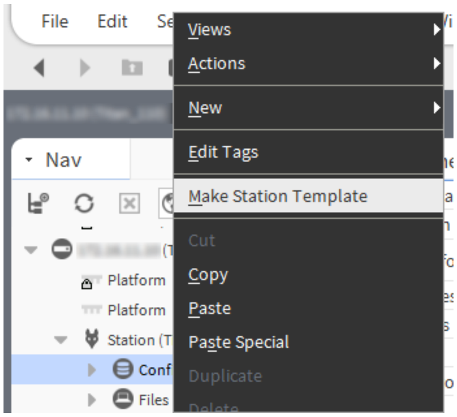
- An existing, fully-configured Niagara station suitable for use in a generic new station template.

Station templates are automatically saved in the stationTemplates sub-directory of your Workbench User Home. User-defined station templates stored in this location are automatically included in the Station Templates table shown in the Workbench New Station Wizard and are available for selection when using the tool.

This procedure is one the Systems Integrator might perform on a fully configured “basic” station for purposes of reuse and standardization.

Step 1 In the Nav tree, right-click on the station’s Config node and select Make Station Template to open the Template view.

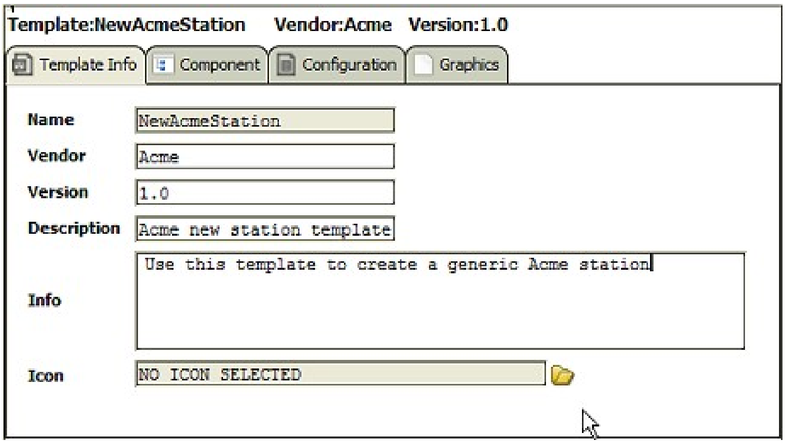
Figure 128: Make Station Template



Step 2 In the Template view, click each of the tabs to edit the station template as needed.

a. On the Template Info tab, fill-in desired information about the template, as shown in the next figure.

Figure 129: Template view



Template:NewAcmeStation Vendor:Acme Version:1.0

Name	NewAcmeStation
Vendor	Acme
Version	1.0
Description	Acme new station template
Info	Use this template to create a generic Acme station
Icon	NO ICON SELECTED

b. On the Component tab, make desired changes for the default station template.

c. On the Configuration tab, add configurable parameters as needed. For example, to add the Foxs Port:

i. In the left pane, expand Services→FoxService→Foxs Port

ii. Double-click Public Server Port to add it to the right pane.

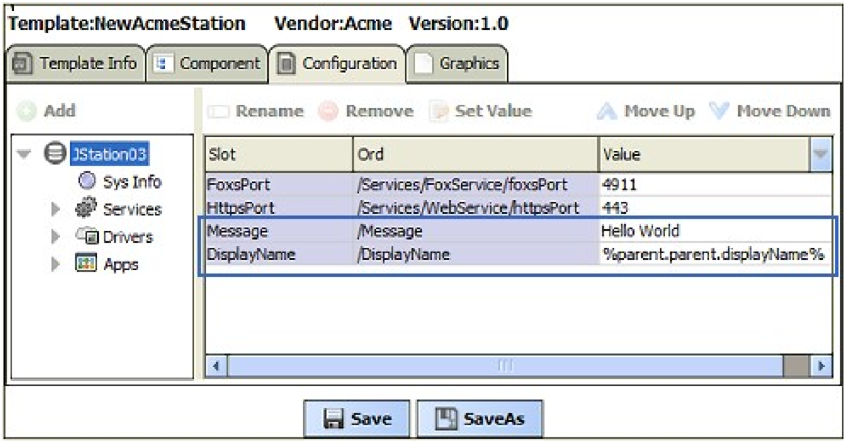
iii. Change the default value as needed and click Set Value.

iv. Click Rename and change the name to “Foxs Port”.

Note:

The value entered into the stationTemplate for the exposed property becomes the default value upon creating a new station.

Figure 130: Template view



Template:NewAcmeStation Vendor:Acme Version:1.0

Slot	Ord	Value
FoxsPort	/Services/FoxService/foxsPort	4911
HttpsPort	/Services/WebService/httpsPort	443
Message	/Message	Hello World
DisplayName	/DisplayName	%parent.parent.displayName%

Note:

Any exposed component properties defined in the template displays as configurable parameters in the New Station Wizard when this template is used.

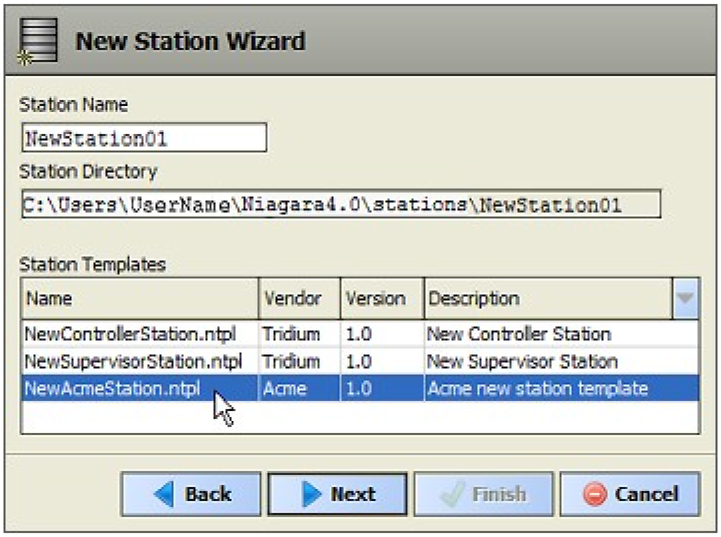
d. On the Graphics tab, make desired changes for the default station template.

Step 3 When finished modifying the template, click Save.

The new station template is saved in the ~stationTemplates sub-directory of your Workbench User Home.

Also, the station template is immediately available for inclusion in the Workbench New Station Wizard, where your template appears as a selectable option in the Station Templates table.

Figure 131: Workbench New Station Wizard Station Templates



New Station Wizard

Station Name
NewStation01

Station Directory
C:\Users\UserName\Niagara4.0\stations\NewStation01

Station Templates

Name	Vendor	Version	Description
NewControllerStation.ntpl	Tridium	1.0	New Controller Station
NewSupervisorStation.ntpl	Tridium	1.0	New Supervisor Station
NewAcmeStation.ntpl	Acme	1.0	Acme new station template

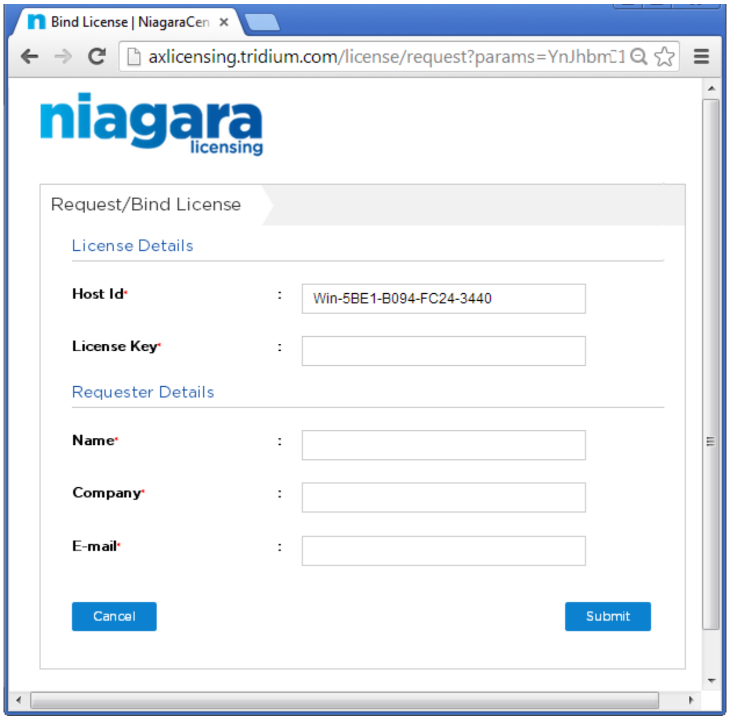
Note:

It is not necessary to restart Workbench in order for the user-defined station template to appear in the New Station wizard. The file simply needs to be saved to the ~stationTemplates directory.

Request License

The Request License option on the Tools menu simply opens a “Bind License form” in your Workbench PC’s default browser. By default, the only pre-filled field in this form is the host ID of your PC.

Figure 132: License request form in browser (from Workbench, Tools > Request License)



Host Id* : Win-5BE1-B094-FC24-3440

License Key* :

Name* :

Company* :

E-mail* :

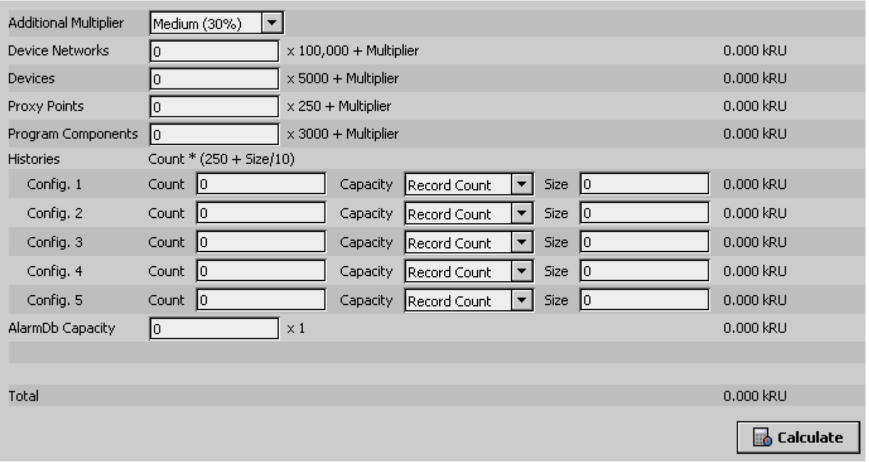
Typically, your Workbench PC is already licensed. Otherwise, you would not be able to successfully start Workbench, and then select Request License from the Tools menu.

However, you could use this as quick method to request a license for another PC on which you have installed Niagara. In that case, you could substitute (type in) the host ID for the other PC in this form, along with other pertinent information.

Resource Estimator

The Resource Estimator tool is available from the Tools menu. Use it as a worksheet to help you estimate the total number of station resources that you will use in a projected station implementation. The resource estimator view is shown in the next figure.

Figure 133: Resource estimator view



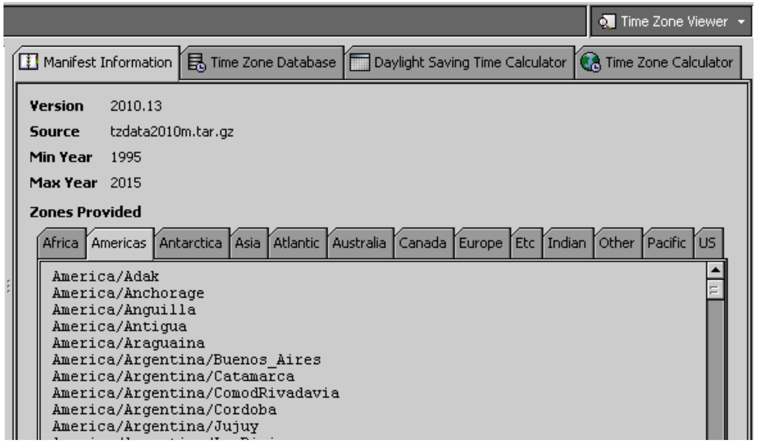
Additional Multiplier	Medium (30%)					
Device Networks	0	x 100,000 + Multiplier				0.000 kRU
Devices	0	x 5000 + Multiplier				0.000 kRU
Proxy Points	0	x 250 + Multiplier				0.000 kRU
Program Components	0	x 3000 + Multiplier				0.000 kRU
Histories	Count * (250 + Size/10)					
Config. 1	Count 0	Capacity	Record Count	Size	0	0.000 kRU
Config. 2	Count 0	Capacity	Record Count	Size	0	0.000 kRU
Config. 3	Count 0	Capacity	Record Count	Size	0	0.000 kRU
Config. 4	Count 0	Capacity	Record Count	Size	0	0.000 kRU
Config. 5	Count 0	Capacity	Record Count	Size	0	0.000 kRU
AlarmDb Capacity	0	x 1				0.000 kRU
Total						0.000 kRU

Time Zone Database Tool

The Time Zone Database Tool, available in the Tools menu, provides several ways to explore the local timezones.jar on the Workbench host. This jar file is the “historical time zone database”.

If the Workbench host is running a station (for example, a Supervisor), this is also the time zone database used by that station. Using this tool, you can see what time zones are available, see the past and current behaviors for any time zone, and in some cases the future behaviors as well.

Figure 134: Time Zone Database Tool in Workbench



Version 2010.13
Source tzdata2010m.tar.gz
Min Year 1995
Max Year 2015

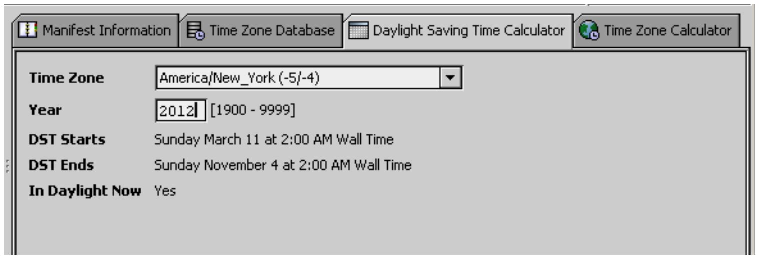
Zones Provided

```
America/Adak
America/Anchorage
America/Anguilla
America/Antigua
America/Araguaina
America/Argentina/Buenos_Aires
America/Argentina/Catamarca
America/Argentina/ComodRivadavia
America/Argentina/Cordoba
America/Argentina/Jujuy
```

Daylight Savings Time Calculator

This tab in the Time Zone Database Tool shows when the DST changeover occurs for any year.

Figure 135: Daylight Savings Time Calculator tab in Time Zone Database Tool



Time Zone America/New_York (-5/-4)
Year 2012 [1900 - 9999]
DST Starts Sunday March 11 at 2:00 AM Wall Time
DST Ends Sunday November 4 at 2:00 AM Wall Time
In Daylight Now Yes

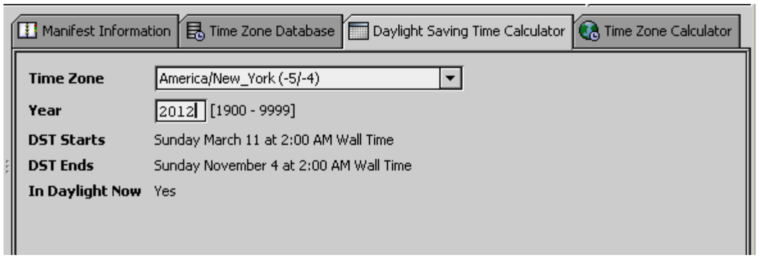
To use, select a time zone and type in a year in the Year field.

Press Enter to update the "DST Starts" and "DST Ends" time fields.

Time Zone Calculator

This tab in the Time Zone Database Tool lets you compare the time between any two time zones.

Figure 136: Time Zone Calculator tab in Time Zone Database Tool



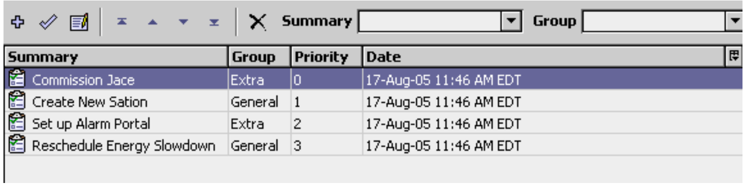
Time Zone America/New_York (-5/-4)
Year 2012 [1900 - 9999]
DST Starts Sunday March 11 at 2:00 AM Wall Time
DST Ends Sunday November 4 at 2:00 AM Wall Time
In Daylight Now Yes

To use, select a source time zone and a target time zone, and specify a source time. Initial default is the current time.

Todo List

The Todo List is available when you select it from the Tools menu. This tool is provided to help you with organizing, prioritizing and tracking tasks in Workbench. The Todo list view is shown in the next figure.

Figure 137: Todo list view



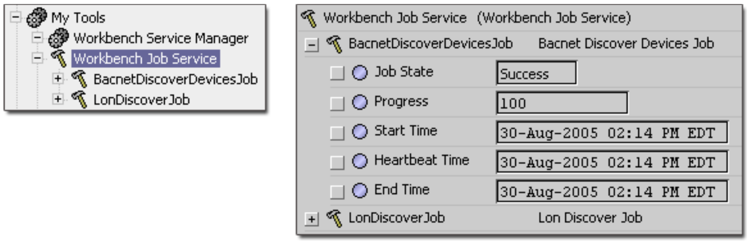
Summary	Group	Priority	Date
Commission Jace	Extra	0	17-Aug-05 11:46 AM EDT
Create New Sation	General	1	17-Aug-05 11:46 AM EDT
Set up Alarm Portal	Extra	2	17-Aug-05 11:46 AM EDT
Reschedule Energy Slowdown	General	3	17-Aug-05 11:46 AM EDT

The Todo list is a tabular view with standard table controls and options, as described in "Table controls and options". You can use this tabular list to create new lists or edit, group and rearrange existing lists and items in your lists. In addition, you can use the filter fields at the top of the display to filter what you see in the table, based on your summary description or group.

Workbench Job Service

The Workbench Job Service view, shown below, is available when you select it from the Tools menu. This job service tool keeps track of all Workbench jobs. These are jobs that are not initiated under a specific station, but initiated by the Workbench environment.

Figure 138: Job service nav tree and property sheet view



- My Tools
  - Workbench Service Manager
  - Workbench Job Service
    - BacnetDiscoverDevicesJob
    - LonDiscoverJob

Workbench Job Service (Workbench Job Service)

BacnetDiscoverDevicesJob	Bacnet Discover Devices Job
Job State	Success
Progress	100
Start Time	30-Aug-2005 02:14 PM EDT
Heartbeat Time	30-Aug-2005 02:14 PM EDT
End Time	30-Aug-2005 02:14 PM EDT
LonDiscoverJob	Lon Discover Job

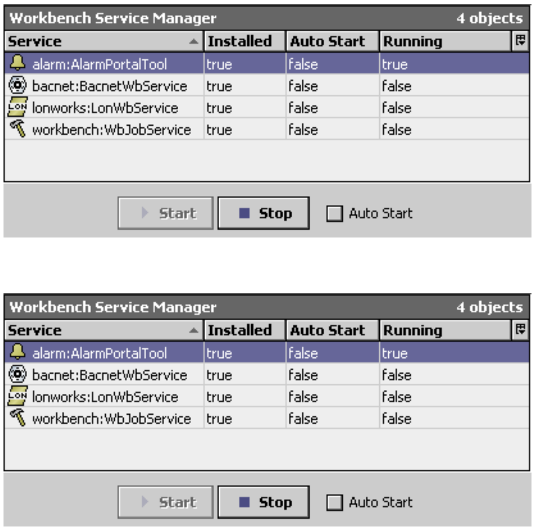
Note:

Workbench jobs are jobs that are initiated and run under the Workbench, not under a station. Jobs that are run under a station are monitored and displayed in the Station Job Service under the station Services node in the nav side bar.

Workbench Service Manager

The Workbench Service Manager view is available when you select it from the Tools menu. The view, shown below, is used to manage the life cycle and configuration of all services.

Figure 139: Workbench service manager view



Workbench Service Manager — 4 objects

Service	Installed	Auto Start	Running
alarm:AlarmPortalTool	true	false	true
bacnet:BacnetWbService	true	false	false
lonworks:LonWbService	true	false	false
workbench:WbJobService	true	false	false

Workbench Service Manager — 4 objects

Service	Installed	Auto Start	Running
alarm:AlarmPortalTool	true	false	true
bacnet:BacnetWbService	true	false	false
lonworks:LonWbService	true	false	false
workbench:WbJobService	true	false	false

The Workbench service manager is a tabular view with standard table controls and options, as described in “Table controls and options”. From this view you can Start, Stop, or configure any of the listed services to Auto-Start.

CHAPTER 5 COMPONENT GUIDES

The following topics are covered in this chapter:

- Components in alarmRdb module
- Components in backup module
- Components in baja module
- Components in chart module
- Components in control module
- Components in converters module
- Components in crypto module
- Components in file module
- Components in help module
- Components in net module
- Components in program module
- Components in timesync module
- Components in web module
- Components in workbench module

The Component Guides provide summary information on common components.

Component Reference Summary

Summary information is provided on components in the following modules:

- alarm
- alarmRdb
- backup
- baja
- chart
- control
- converters
- crypto
- email
- file
- flr
- help
- history
- net
- onCall
- program
- schedule
- sms
- timesync
- web
- workbench

Components in alarmRdb module

- rdbAlarmService

alarmRdb-RdbAlarmService

RdbAlarmService provides a means for using an RDBMS to store alarm records instead of using the default alarm database (^\alarm\alarm.adb). This is different than exporting histories to an RDBMS. When you use the rdbAlarmService, it replaces the use of the station's default alarm database.

Components in backup module

This module provides the BackupService component.

backup-BackupService

The BackupService provides for complete configuration backups to a WorkbenchWorkbench PC or a browser PC (user with Wb web profile). By default, the BackupService is included when you create a new station using the New Station wizard. The target host (JACE, Supervisor) must have the backup module installed.

The default view of a station's BackupService is the BackupManager, which provides a Backup button to manually initiate a backup. A backup automatically performs a local station save first, and is run as a standard station Job. This means each backup provides a progress bar, and upon completion, a popup notification. Under the station's JobService, any backup appears as a "Fox Backup."

About a backup dist

A backup is saved as a dist file (a zipped format) including minimally the station's config.bog, current station console output (.txt file), and backup.log file. If other station file types and subfolders are not excluded (in the BackupService configuration), the backup dist file contains them too, for example, files of type: px, nav, html, jpg, png, and so forth.

Also, the backup dist contains the zipped "nre-config" for that host (including license and certificates files), as well as pointers to the installed "nre-core," OS, and JVM, each by version. Essentially, a backup dist provides a "configuration snapshot" of the entire JACE platform in zipped "dist" file format. This allows for a complete image restoration.

Note:

- Be careful to keep backup dist files in a secure location. They have always contained sensitive information, for example a station’s config.bog file. They also may contain sensitive host platform information. In update releases (AX-3.7U1), this includes unique “key ring” files used for client password encryption.

Not included in a backup is runtime data that is actively managed by the station, such as the alarm and history databases. This data should be “backed up” using standard alarm routing and history archiving to a Supervisor host.

The default backup destination depends on your station connection, as either:

- Workbench(Fox) — !backups: A subdirectory backups under your Niagara release directory. If you have not previously made station backups, this directory is automatically created.
- Browser access (Wb Web Profile) — !backups: A niagara\wbapplet\backups folder under your Windows user profile location, for example:
  - Windows 7: C:\Users\John\niagara\wbapplet\backups
  - Windows XP: C:\Documents and Settings\John\niagara\wbapplet\backups

If you have not previously made station backups, this directory is automatically created.

The default name for a backup file uses a format of: backup_stationName_YYMMDD_HHMM.dist

For example, backup_demo_130412_1429.dist for a backup made of station “demo” on April 12, 2013 at 2:29 pm.

Restoring a backup

To restore a backup dist from Workbench, you open a platform connection to the JACE, and then use the platform Distribution File Installer to install a backup dist file.

- Restoring from a backup may be necessary if the host failed in some manner, to allow recovery (to the same hardware, or to replacement hardware).
- Another usage is to install the same backup dist file on multiple hosts, such that each host (typically JACE) has the identical configuration, including station database. When performing these “replicated” host installations of a backup dist, the platform Distribution File Installer allows you to choose if TCP/IP settings from the backup dist should be restored (the default is to not). Typically you do not, as TCP/IP settings must be unique on each host.

BackupService configuration

Configuration lets you define the file types and/or folders not included in a station backup. The service's property sheet provides the following properties for configuration:

Property	Value	Description
Enabled [general]	true or false	Activates and deactivates use of the function.
Exclude Files	string of names (each delimited by a semi-colon (;)	Specifies file types to exclude from the backup dist, either by name or extension. By default, the following files are excluded: *.hdb;*.adb;*.lock;*backup*;console.*;config. bog.b*;config_backup*
Exclude directories	string of Ords with Ord Chooser control	Specifies station subdirectories to exclude from the backup dist, using relative ORD syntax. An Ord Chooser control provides a Directory Chooser window in which to select station subfolders. By default, the following subfolders are excluded: file:^history, file:^alarm
Offline Exclude Files	string of names (each delimited by a semi-colon (;)	Specifies file types to exclude from the backup dist, when the station is stopped on the source host, either by name or extension. By default, the following files are excluded: *.lock; *backup*;console.*;config.bog.b*;config_backup* Note that History (*.hdb) and alarm (*.adb) files are backed up, unlike with a running backup.
Offline Exclude Directories	string of Ords with Ord Chooser control	Specifies station subdirectories to exclude from the backup dist, when the station is stopped on the source host. Directories are specified using relative Ord syntax. An Ord chooser control provides a Directory Chooser dialog in which you can select station subfolders. By default, no directories are excluded, unlike with a running backup.

baja-FoxBackupJob

This component appears as a child of the JobService and displays the following properties relative to the save job:

- Job State

Which of the following states the backup job is in currently: unknown, running, canceling, canceled, success, failed.

- Progress

A percentage 0) of progress toward completing the job.

- Start Time

Displays the time that the job started.

- Heart Beat Time

Displays the time of the last indication that the job is alive.

- End Time

Displays the time that the job ends.

Note that all jobs in a station are cleared upon a station restart.

Components in baja module

The following components are in baja module:

- Category
- CategoryService
- Component
- DataFile
- Directory
- FileSystem
- Folder
- Format
- IpHost
- Job
- JobService
- LocalHost
- Module
- ModuleSpace
- Permissions
- PermissionsMap
- PxView
- ServiceContainer
- Spy
- Station
- StationSaveJob
- UnrestrictedFolder
- User
- UserPasswordConfiguration
- UserPrototypes
- UserService
- UserServicePasswordConfiguration
- Vector
- VirtualComponent
- VirtualGateway
- WsTextBlock
- ZipFile

baja-Category

Each Category represents a custom logical grouping, and has a unique category index number. You can assign components, files, and histories to one or more categories. All categorizable components have a CategorySheet view, which shows you that component’s stored category memberships (CategoryMask).

Categories reside under the station’s CategoryService, which has views CategoryBrowser and CategoryManager. Categories play an integral role in station security, where you can give users permissions for some (or all) categories.

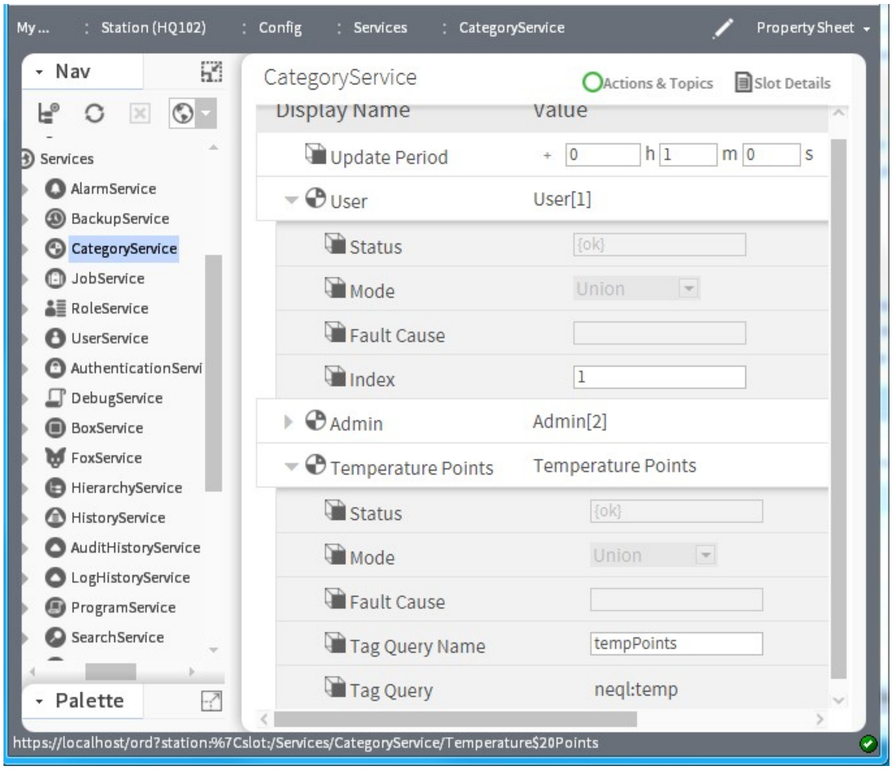
baja-CategoryService

This service is the station container for all categories, which represent logical groupings to which you can assign components, files, and histories. It is located in a station's Services container.

The default view of this service, the Category Browser, lets you centrally assign different objects to categories, using an expandable tree view of the station. The CategoryService also provides a Category Manager view, for you to create, edit and delete categories. Categories play an integral role in station security, where you can give users permissions for some (or all) categories. By default, the CategoryService is included when you create a new station using the New Station wizard.

Primary properties

Figure 140: CategoryService property sheet



Display Name	Value
Update Period	+ 0 h 1 m 0 s
User	User[1]
Status	{ok}
Mode	Union
Fault Cause	
Index	1
Admin	Admin[2]
Temperature Points	Temperature Points
Status	{ok}
Mode	Union
Fault Cause	
Tag Query Name	tempPoints
Tag Query	neql:temp

https://localhost/ord?station:%67Cslot:/Services/CategoryService/Temperature$20Points

In addition to being the container for child categories, the CategoryService has only one slot, Update Period.

Property	Value	Description
Update Period	hours minutes seconds	Sets the interval at which the system automatically assigns ancestor permissions. The default value is one (1) minute. If you assign a zero value, the system disables this feature.

User, Admin and additional basic category properties

Property	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • {ok} indicates that the component is polling successfully. • {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. • {disabled} indicates that the Enable property is set to false. • fault indicates another problem.
Mode		
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Index	integer	Sequential number that identifies the property in the station.

Tagged category properties

Property	Value	Description
Tag Query Name	text	A descriptive name to represent the results of the search.
Tag Query	NEQL	A NEQL query. This property is required when Type is Tagged Category.

baja-Component

Component is the required base class for all Baja component classes. The Component is available in the baja module.

Using Containers

Containers allow you to logically group components. The current container is the component that contains components in the display window. A container may be selected as the current container by one of the following methods:

- Double-click the component in the Nav tree.
- Right-click the component in the Nav tree (which brings up a menu) and select a view.
- Right-click the component in a wire sheet (which brings up a menu) and select a view.

Container components include the following:

- Component can be used as a general container for components. It allows you to place components and links in a container.
- Page is a special component used to created a map of name/Ref pairs as dynamic slots.

baja-DataFile

DataFile represents a data file in the file system of a session.

baja-Directory

Directory is used to represent directories in File space implementations.

baja-FileSystem

FileSystem is a File space for the local machine's file system.

baja-Folder

Folder is a special container designed to store components. The Folder is available in the baja palette.

baja-Format

Format (or “BFormat”) is used to format objects into strings using a standardized formatting pattern language. The format string is normal text with embedded scripts denoted by the % percent character (use %% to insert a real %). A script is one or more calls chained together using the . dot operator. Calls are mapped to methods using reflections.

Given call “foo”, the order of reflection mapping is:

- special call (see below)
- getFoo(Context)
- getFoo()
- foo(Context)
- foo()
- get("foo")

The following special functions are available to use in a script:

- time() calls Clock.time() to get current time as an AbsTime
- lexicon(module:key) gets the specified lexicon text

Examples of formats:

- "hello world"
- "my name is %displayName%"
- "my parent's name is %parent.displayName%"
- "%value% {%status.flagsToString%} @ %status.priority%"
- "%time().toDateString%"
- "%lexicon(bajaui:dialog.error)%"

For related details, see the section, “Formats (BFormat)”.

baja-IpHost

IpHost is used to represent a host machine that is identified with an IP address. The hostname of an IpHost is either a name resolvable via DNS or is a raw IP address.

- A blue square indicates active connection(s) from Workbench, for example, Fox (station) or platform.
- A red square indicates no active connections from Workbench.

baja-Job

A Job is used to manage a task that runs asynchronously in the background, but requires user visibility. Some example jobs include:

- StationSaveJob — From a station save, either initiated manually or from the auto-save function

- FoxBackupJob — From a station backup (dist) to a remote PC, see the section “backup-BackupService”. Many drivers also have various job types too. For example, the NiagaraDriver includes a StationDiscovery-Job and NiagaraScheduleLearnJob.

Every job finishes displaying one of the following status descriptors:

- Success — Job completed successfully.
- Canceled — Job canceled by user.
- Failure — Job failed to complete.
- Completed — Job completed.

Also, if you have the station open in Workbench, you see a momentary “notification popup” in the lower right corner of your display.

You can monitor and cancel a job from within the particular view where you initiated it, or centrally from the JobServiceManager view of a station’s JobService. You can also open a Jobs side bar to see all jobs in all opened stations.

Regardless of how you access jobs, note the following:

- To see details on a job, click the button next to its status descriptor. A popup Job Log dialog displays all the interim steps about the job, including timestamps and relevant messages.
- To dispose of a job, click the close (“X”) button to remove it from the station.

Note:

All jobs in a station are cleared upon a station restart.

baja-JobService

The JobService contains Jobs that were executed by different processes in the station. Each job appears as a child component. By default, the JobService is included when you create a new station using the New Station wizard. The default view of the JobService is the JobServiceManager.

Note:

All jobs in a station are cleared upon a station restart.

baja-LocalHost

LocalHost represents the root of the Baja local Host namespace. The LocalHost is available in the baja Module.

baja-Module

Module encapsulates a Baja software module which is packaged and delivered as a JAR file with a "meta-inf/ module.xml" description. Modules are the basic unit of software deployment in the Baja architecture. Module names must be one to 25 ASCII characters in length and globally unique. Modules are available in the Workbench ModuleSpace (named “My Modules” under “My Host”).

For related details, see the section “About modules”.

baja-ModuleSpace

ModuleSpace is the container for modules, which are keyed by their module name. In Workbench this is MyModules under My Host.

baja-Permissions

Permissions are a property used to define the permissions given to a user’s PermissionsMap.

baja-PermissionsMap

This component defines the security permissions to grant to a user.

Permissions are added as dynamic properties with the name matching a Category and the value an instance of permissions.

baja-PxView

PxView a dynamic view which may be added to components as a property or by overriding getAgents(). PxViews store the view contents in an XML file with a px extension. The view itself is defined as a tree of bajaui:Widgets.

baja-ServiceContainer

ServiceContainer (Services) is the container used, by convention, to store a station's services. The Service Manager is its primary view. A ServiceContainer is included in any station created using the New Station tool.

baja-Spy

Spy is an object wrapper for an instance of Spy, with an available "SpyViewer" interface to view diagnostic information about the running station.

baja-Station

Station (Config) represents the component configuration of a station in the Baja framework. The Station is available in a fox connection to a Niagara host, along with its file space (Files) and histories (History).

Station Save

Save the current state of the system to persistent storage. You should regularly backup your station or stations using the Save Action on the station.

baja-StationSaveJob

This component appears as a child of the job service and displays the following properties relative to the save job:

- Job State: indicates which of the following states the save job is in currently: unknown, running, canceling, canceled, success, failed.
- Progress: indicates a percentage 0) of progress toward completing the job.
- Start Time: displays the time that the job started.
- Heart Beat Time: displays the time of the last indication that the job is alive.
- End Time: displays the time that the job ends

For related information, refer to the following:

- baja-Job
- baja-JobService

Note:

All jobs in a station are cleared upon a station restart.

baja-SyntheticModuleFile

(AX-3.7 and later) SyntheticModuleFile (synthetic module) is a synthetic Java archive (.sjar) that allows the creation of memory-resident modules and types programmatically at station runtime. Synthetic modules differ from "standard" .jar (non-synthetic) modules in a number of ways, and have a special default "Synthetic Module File View" for editing contents.

baja-UnrestrictedFolder

UnrestrictedFolder is a special container designed to store objects for use on a palette. Normal isParentLegal() checks are not applied to UnrestrictedFolders. The UnrestrictedFolder is available in the baja palette.

baja-User

User is the component that represents a station connection, typically a specific person who needs to access the system. Users are children of the station's UserService. User components are also children of the UserService's UserPrototypes container, to allow "centralized user" support.

baja-UserPasswordConfiguration

UserPasswordConfiguration (Password Configuration) is a child container under each User component (and UserPrototype component) in an AX-3.7 or later station. It contains properties to specify periodic password expiration for the user and to require a password change (reset) upon the user’s next station login. The station’s UserService also has a related Password Configuration child container.

Note:

These components are not available in a new station until it is started and saved.

baja-UserPrototypes

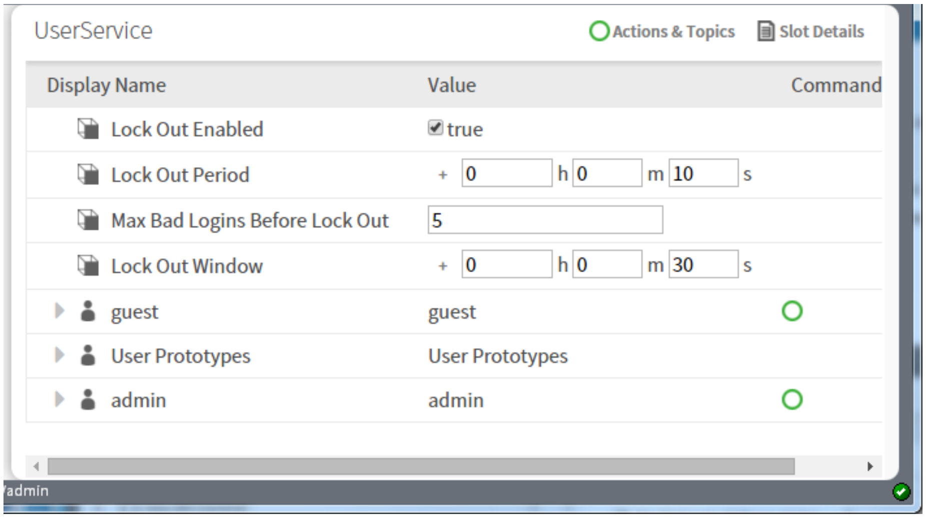
UserPrototypes is a frozen container on a station’s UserService. It contains a single frozen “Default Prototype” user, as well as any additional users needed for support of “centralized users” in the station’s NiagaraNetwork.

baja-UserService

This service manages all system users: human and machine. You access it by right-clicking UserService and clicking Views→Property Sheet.

The User Manager is its primary view of this service. The UserService is available in the baja module. By default, the UserService is included when you create a new station using the New Station wizard.

Figure 141: UserService property sheet

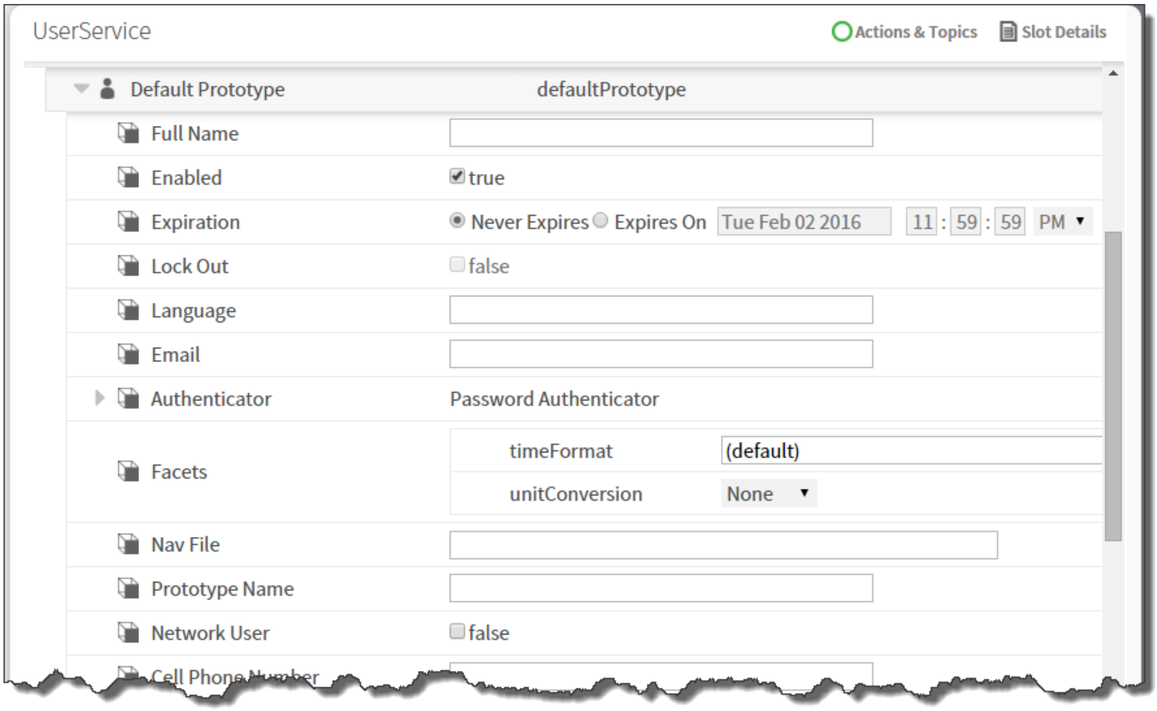


Display Name	Value	Command
Lock Out Enabled	☑ true	
Lock Out Period	+ 0 h 0 m 10 s	
Max Bad Logins Before Lock Out	5	
Lock Out Window	+ 0 h 0 m 30 s	
guest	guest	○
User Prototypes	User Prototypes	
admin	admin	○

Property	Value	Description
Lock Out Enabled	true or false	If enabled (true), a number of consecutive authentication failures temporarily disable login access to the user account, for the duration of the lock out period (next property). Using lock out makes it difficult to automate the guessing of passwords. Note that each user has a Clear Lock Out action.
Lock Out Period	hours minutes seconds (default is 10 seconds)	If lock out is enabled, this property defines the period of time a user account is locked out before being reset. While locked out, any login attempt (even a valid one) is unsuccessful. ***Note:*** The default Lock Out value guards against an automated, brute-force password attack, where a computer application issues hundreds of login attempts a second. The 10 second latency thwarts such an attack, as the attacker must wait 10 seconds after each five unsuccessful login attempts. If deemed necessary, you can adjust to guard against human attack.
Max Bad Logins Before Lock Out	number from 1 — 10 (default is 5)	If lock out is enabled, in conjunction with the Lock Out Window, this property specifies the number of consecutive failed user login attempts that trigger a lock out after a window of time.
Lock Out Window	hours minutes seconds (default is 30 seconds)	If lock out is enabled, and a user fails to log in successfully before the Max Bad Logins Before Lock Out window (period) expires, the user is locked out for the Lock Out Period duration. The system enforces changes to lock out properties the next time the user logs in. For example, if Max Bad Logins Before Lock Out is set to 5, user ScottF fails to log in four times within the Lock Out Window, and an admin-level user changes Max Bad Logins Before Lock Out to 3, the change does not lock ScottF out. User ScottF still has one more chance to log in before getting locked out. If ScottF's fifth attempt to log in fails, the system locks him out the next time he attempts to log in because five failed attempts is greater than or equal to the Max Bad Logins Before Lock Out of 3.
User Prototypes	multiple properties	See the next section.

Default Prototype

Figure 142: Prototype user properties



UserService

Default Prototype defaultPrototype

Full Name

Enabled ☑ true

Expiration ● Never Expires ○ Expires On Tue Feb 02 2016 11 : 59 : 59 PM

Lock Out □ false

Language

Email

Authenticator Password Authenticator

Facets timeFormat (default)

unitConversion None

Nav File

Prototype Name

Network User □ false

Cell Phone Number

The following table lists the properties to configure for each user.

Property	Value	Description
Full Name	text	The user’s name.
Enabled	true (default)	Unchecked (false) disables this user. Disabled users cannot access the system.
Expiration	radio buttons	• Never Expires: permits this user to always log on. • Expires On [date and time]: allows this user to log on until the expiration date and time.
Lock Out	false (default)	Checked enables a user may log on. Unchecked (true) prohibits this user from logging on.
Language		Identifies the language to use.
Email	email address	Defines the user’s email address.
Authenticator	additional properties	Manages user password.
Facets	timeFormat and unitConversion	Configures the time format and units to use when this user logs in to the system.
Nav File	file:^nav/NavFile.nav	Identifies the file to use for displaying a customized navigation tree.
Prototype Name	text	The name of the prototype used to create this user.

Property	Value	Description
Network User	false (default)	When unchecked (true), this user account can be synchronized to other stations on the network.
Cell Phone	number	The user's mobile phone number.
Authentication Scheme Name	drop-down list	Identifies the authentication scheme used to verify this user.
Roles	radio buttons	The checked box identifies the user's role.
Allow Concurrent Sessions	true (default)	When checked, allows you to connect multiple sessions. When unchecked, a new session invalidates the old session.
Default Web Profile		See "Default Web Profile".
Mobile Web Profile		See "Mobile Web Profile".

Authenticator—Password Config

Property	Value	Description
Force Reset at Next Login	true or false (default)	Causes the system to request that the user create a new password the next time they log in.
Expiration	radio button	Allows you to configure a password change for a specific date and time.

Default Web Profile

These properties configure the information available to a specific user who accesses the system through a browser. By default all information is visible.

Property	Value	Description
Type Spec	drop-down list	Identifies the type of • hx (default) • axvelocity • mobile • Workbench
Type Spec (type of profile)	drop-down list	Identifies the type of profile: • BasicHxProfile • DefaultHxProfile • HTML5HxProfile (default) • HandHeldHxProfile
Hx Theme	drop-down list	Selects the look of the user interface: • Zebra • Lucid
Enable Hx Workbench Views	yes (default)	Removing the check mark disables the views.

Property	Value	Description
Enable Nav Tree Side Bar	yes (default)	Removing the check mark disables the Nav tree.
Enable Search Side Bar	yes (default)	Removing the check mark disables the use of the search side bar.
Enable Palette Side Bar	yes (default)	Removing the check mark disables the use of the palette side bar.
Enable Nav File Tree	yes (default)	
Enable Config Tree	yes (default)	
Enable Files Tree	yes (default)	
Enable Histories Tree	yes (default)	
Enable Hierarchies Tree	yes (default)	

Mobile Web Profile

Property	Value	Description
Type Spec (type of spec)	drop-down list	• Hx • Mobile
Type Spec (type of profile)	drop-down list	Identifies the type of profile: • BasicHxProfile • DefaultHxProfile • HTML5HxProfile (default) • HandHeldHxProfile
Mobile Nav File	ord	Identifies the location of the file that defines the mobile nav tree for this user.
Hx Theme	drop-down list	Selects the look of the user interface: • Zebra • • Lucid

baja-UserServicePasswordConfiguration

UserServicePasswordConfiguration (Password Configuration) is a child container under the UserService in an AX-3.7 or later station. It contains global properties to define the periodic password expiration for station users as well as the unique password history setting. Also note each station User has a related Password Configuration child container as well.

Note

This component is not available in a new station until it is started and saved.

Due to our policy of continuous product innovation, some specifications may change without notification.

baja-Vector

Vector is a dynamic Struct which allows properties to be added at runtime.

baja-VirtualComponent

A VirtualComponent is the Baja base class for implementations of transient (non-persisted) components that reside in the station’s “virtual component space,” as defined by a VirtualGateway.

Initial applications of virtual components are expected in various drivers for Niagara.

baja-VirtualGateway

A VirtualGateway is the Baja base class for a component that resides under the station’s component space (Config), and acts as a gateway to the station’s “virtual component space.”

Other object spaces are Files and History. Initial applications of virtual gateways are expected in Niagara drivers.

baja-WsTextBlock

WsTextBlock (Text Block) is a component you can drop onto a wire sheet (WireSheet) and position to add text notes. Properties include “Text” (to display), Foreground and Background colors, Font, Border, and whether the Text Block is directly “Selectable” in the wire sheet (by default, Selectable is true). If Selectable is false, you must select this component via its node in the Nav tree.

Text Block is available in the baja palette.

baja-ZipFile

ZipFile represents a zip file in the file system of a session.

Components in chart module

The following components are in chart module:

- BarChart
- ChartCanvas
- ChartHeader
- ChartPane
- DefaultChartLegend
- LineChart

- chart-BarChart

This is one of two chart types available. The other is LineChart.

- chart-ChartCanvas

ChartCanvas is the canvas widget under a ChartPane.

- chart-ChartHeader

ChartHeader is the header widget under a ChartPane.

- chart-ChartPane

ChartPane is the container widget created when adding a Px chart.

- chart-DefaultChartLegend

DefaultChartLegend is the legend widget under a ChartPane.

- chart-LineChart

This is one of the two chart types available. The other is BarChart.

Components in control module

The following components are in control module:

- BooleanPoint
- BooleanWritable
- DiscreteTotalizerExt
- EnumPoint
- EnumWritable
- NullProxyExt
- NumericPoint
- NumericTotalizerExt
- NumericWritable
- StringPoint
- StringWritable
- TimeTrigger

control-BooleanPoint

BooleanPoint is a basic read-only control point, with default slots: Facets, Out, and ProxyExt. See “About control points” for details. The BooleanPoint is available in the Points folder of the control palette.

control-BooleanWritable

BooleanWritable extends BooleanPoint to include 16 levels of command priority control. The highest priority active command is reflected in the out property. Commands at the emergency and manual levels (1 and 8) are stored persistently. See “About control points”for more details.

The BooleanWritable is available in the Points folder of the control palette.

control-DiscreteTotalizerExt

DiscreteTotalizerExt is a control point extension for accumulating runtime and change of state counts on binary or enum values. Two actions are available to clear (zero) accumulated totals, ResetChangeOfStateCount and ResetElapsedActiveTime.

The DiscreteTotalizerExt is available in the Extensions folder of the control palette.

control-EnumPoint

EnumPoint is a basic read-only control point, with default slots: Facets, Out, and ProxyExt. See “About control points” for details. The EnumPoint is available in the Points folder of the control palette.

control-EnumWritable

WritableEnumPoint extends EnumPoint to include 16 levels of command priority control. The highest priority active command is reflected in the out property. Commands at the emergency and manual levels (1 and 8) are stored persistently. See “About control points” for more details.

The EnumWritable is available in the Points folder of the control palette.

control-NullProxyExt

NullProxyExt is the default implement AbstractProxyExt used to indicate that a point is not a proxy point. The NullProxyExt is in the control palette's Extensions folder, but is already present by default on compatible point types. Presence indicates that you can add other types of extensions to the parent component, for example alarm or history extensions.

control-NumericPoint

NumericPoint is a basic read-only control point, with default slots: Facets, Out, and ProxyExt. See "About control points" for details. The NumericPoint is available in the Points folder of the control palette.

control-NumericTotalizerExt

NumericTotalizerExt is a control point extension used for integrating a numeric point value over time. For example, a totalizer with a minutely totalization interval can convert an instantaneous flow reading in cubic feet per minute (cfm) into a value representing total cubic feet consumed. An available resetTotal action clears (zeroes) any accumulated total.

The NumericTotalizerExt is available in the Extensions folder of the control palette.

control-NumericWritable

NumericWritable extends NumericPoint to include 16 levels of command priority control. The highest priority active command is reflected in the Out property. Commands at the Manual and manual levels (1 and 8) are stored persistently. See the sections, "About control points" and "About writable points" for more details.

The NumericWritable is available in the Points folder of the control palette.

control-StringPoint

StringPoint is a basic read-only control point, with default slots: Facets, Out, and ProxyExt. See "About control points" for details. The StringPoint is available in the Points folder of the control palette.

control-StringWritable

StringWritable extends StringPoint to include 16 levels of command priority control. The highest priority active command is reflected in the Out property. Commands at the Manual and manual levels (1 and 8) are stored persistently. See "About control points" and "About writable points" for more details.

The StringWritable is available in the Points folder of the control palette.

control-TimeTrigger

TimeTrigger is a component that fires a topic at configured times. It is available configured as "Interval" or "Daily" in the Trigger folder of the control palette.

Components in converters module

The following component is in converters module:

- EnumToSimpleMap

converters-EnumToSimpleMap

EnumToSimpleMap maps ordinals to Simple instances of the same type.

Components in crypto module

The following components are in crypto module:

- CryptoService
- SslProvider

crypto-CryptoService

CryptoService is a station service that can provide SSL features for earlier model QNX-based JACEs (controllers running the IBM J9 JVM) at any Niagara release level. It also applies to any Niagara platform running a pre-AX-3.7 release that require SSL capablility.

Note:

Any AX-3.7 or later platform that runs the Oracle Hotspot JVM (most newer QNX-based JACEs) and all Windows-based hosts should be configured differently for SSL, and should not use this station service or module. .

crypto-SslProvider

This provides SSL configuration for the CryptoService in a NiagaraAX host that cannot run the newer, platform based SSL introduced in AX-3.7 (such as controllers running the IBM J9 JVM).

Components in file module

The following components are in file module:

- ApplicationFile
- AudioFile
- BajadocFile
- BogFile
- BogScheme
- BogSpace
- CFile
- CssFile
- ExcelFile
- GifFile
- HtmlFile
- ImageFile
- JavaFile
- JpegFile
- NavFile
- PaletteFile
- PdfFile
- PngFile
- PowerPointFile
- PrintFile
- PxFile
- TextFile
- VideoFile
- VisioFile
- WordFile
- XmlFile

file-ApplicationFile

ApplicationFile stores an application file.

file-AudioFile

AudioFile stores an audio file.

file-BajadocFile

BajadocFile represents Bajadoc documentation. Bajadoc is a special file that can describe components in a database.

file-BogFile

BogFile represents a BogFile in the file system of a session. A Bog File is a special file that can describe components in a database.

file-BogScheme

BogScheme represents a BogScheme in the file system of a session. A Bog File is a special file that can describe components in a database.

file-BogSpace

BogSpace represents a BogSpace in the file system of a session. A Bog File is a special file that can describe components in a database.

file-CFile

CFile stores a c source file.

file-CssFile

CssFile stores a CSS cascading style sheet.

file-ExcelFile

ExcelFile stores a Microsoft Excel file.

file-GifFile

GifFile stores a GIF image.

file-HtmlFile

HtmlFile stores HTML markup.

file-ImageFile

ImageFile stores an image.

file-JavaFile

JavaFile stores a java source file.

file-JpegFile

JpegFile stores a JPEG image.

file-NavFile

NavFile stores XML nav markup.

file-PaletteFile

This file is a Bog file with a different extension and icon. Many modules include a palette.

file-PdfFile

PdfFile stores a Adobe PDF file.

file-PngFile

PngFile stores a PNG image.

file-PowerPointFile

PowerPointFile stores a Microsoft PowerPoint file.

file-PrintFile

PrintFile stores XML print markup.

file-PxFile

PxFile is just a Bog File file with a different extension and icon.

file-TextFile

TextFile stores plain text.

file-VideoFile

VideoFile stores a video file.

file-VisioFile

VisioFile stores a Microsoft Visio file.

file-WordFile

WordFile stores a Microsoft Word file.

file-XmlFile

XmlFile stores an xml file.

Components in help module

The following component is in help module:

- BajadocOptions

help-BajadocOptions

The BajadocOptions stores the options used by the BajadocViewer. The Bajadoc options allow you to change the following:

- Show Baja Only
- Flatten Inheritance

These are stored under /user/{user}/bajadoc.options.

Components in net module

The net module contains the components listed below.

- InternetAddress
- HttpProxyServer

net-InternetAddress

InternetAddress models an Internet address which is a composite of a hostname (or raw IP address) and a port number.

net-HttpProxyServer

HttpProxyService is used to support connections to the Internet through a non-transparent proxy.

All services that make use of the bajax.baja.net.HttpConnection class will automatically roll over to using the proxy server once the HttpProxyService has been configured and enabled. The Weather Service is one of the services that is affected by this feature.

To use this component you must:

- add it to your station (under the Services node, for example)
- configure the following properties in the Property Sheet view:

  – Status

  This display-only property displays the status of the Http Proxy Service.

  – Fault Cause

  This display-only property provides an error message that indicates the reason for a fault.

  – Enabled

  This property has a true (enabled) and false (disabled) option.

  – Server

  This property provides a text field for entering the address of the proxy server you are connecting to.

  – Port

  This property is for specifying the port number for communicating with the proxy server.

  – Exclusions

  This text field allows you to designate addresses that are allowed to be contacted directly, therefore “excluding” them from having to be contacted through the proxy server. The default addresses in the field are typical addresses followed by a slash(/) and the subnet mask designation.

  – Authentication Scheme

  This property provides the following two options: None (no authentication is required at the proxy server when this option is set) and Basic (basic authentication is required at the proxy server when this option is set).

  – User

  This is the login name to be used when authentication is set to Basic.

  – Password

  This is the password text that is to be used when authentication is set to Basic.

Components in program module

The following components are in program module:

- Program
- ProgramModule
- ProgramService

program-Program

Program uses an instance based classfile to implement user defined component logic. The Program can be viewed and edited using the ProgramEditor. Program is available in the program palette.

Actions

Actions include Execute.

program-ProgramModule

ProgramModule provides a Program Module Builder view used to build standard Niagara modules from Program components. This provides a mechanism to version and provision Program modules just like any other Niagara modules.

Starting in AX-3.5, the ProgramModule is available in the program palette.

program-ProgramService

The ProgramService provides a (default) Batch Editor view to launch a batch operation on multiple selected component slots.

The ProgramService also provides a secondary RobotEditor view to create “Robots”, which are similar to Program components (written using Java code), but are not persisted in the station database.

The ProgramService is available in the program palette, but typically exists in most station’s Services container, as it is included among default services when using the New Station tool (wizard) in Workbench.

Actions

Actions include Run Robot.

Components in timesync module

The following components are in time sync module:

- TimeSyncClient
- TimeSyncService

timesync-TimeSyncClient

Slot of a TimeSyncService used to poll a time protocol server. The TimeSyncClient is available in the timesync palette.

timesync-TimeSyncService

This component uses the RFC 868 protocol to synchronize servers to the same time. This service is available in the timesync palette.

You synchronize the current server to other servers by adding TimeSyncClients to the current server's TimeSyncService component. As a convenience, the TimeSyncService provides a disabled default client, which needs configuring. The module palette contains clients that have been preconfigured for well known time protocol servers.

Using multiple clients, the service polls each configured server, and if the service detects a drift greater than the tolerance property, it resets the system time.

The primary view of this component is Time Sync Manager.

From the property sheet, you can set Enabled and Server Enabled to false to disable the time sync client. Server Enabled enables the time sync server. To sync between stations, enable master TimeSyncService to be a server, and then add a TimeSyncClient in the station, configured for the server stations.

Properties

Property	Value	Description
Enabled [general]	true or false	Activates and deactivates use of the function.
Server Enabled	true or false	Activates and deactivates use of the component.

Actions

To invoke the action, right-click on the component.

Action	Description
ForceSync	Causes immediate time synchronization.
Sync	Time synchronization

Components in web module

The following component is in web module:

- WebService

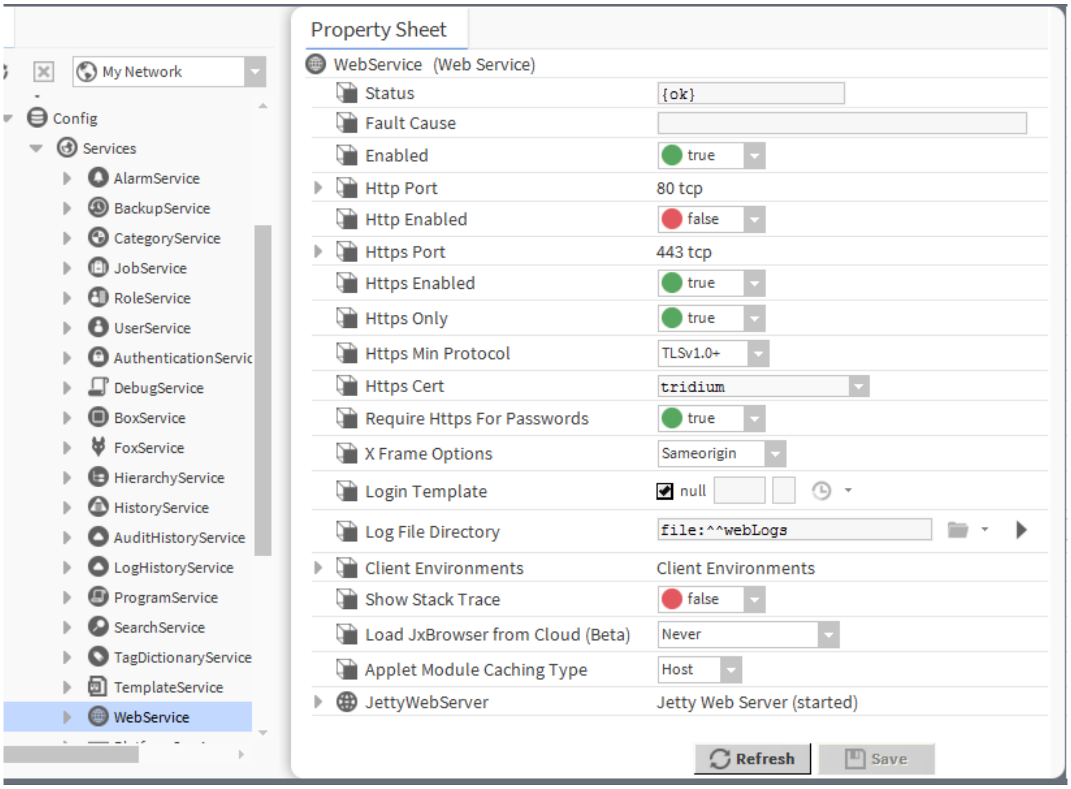
web-MobileClientEnvironment

MobileClientEnvironment (mobile) is a child of the ClientEnvironments container of a station’s WebService, and present only if the host is licensed with the mobile feature. It is used in the automatic selection of the appropriate webProfile type for a user, based on the detection of the incoming browser client type (e.g. desktop or mobile).

web-WebService

This service encapsulates access to the HTTP server as well as the servlet infrastructure used to expose custom applications over HTTP. The WebService is available in the web palette. It is also one of the default services in a station created by using the New Station tool. Only one WebService is supported in a station.

Figure 143: Example of WebService properties



WebService (Web Service)

Status	{ok}
Fault Cause	
Enabled	● true
Http Port	80 tcp
Http Enabled	● false
Https Port	443 tcp
Https Enabled	● true
Https Only	● true
Https Min Protocol	TLSv1.0+
Https Cert	tridium
Require Https For Passwords	● true
X Frame Options	Sameorigin
Login Template	☑ null
Log File Directory	file:^^webLogs
Client Environments	Client Environments
Show Stack Trace	● false
Load JxBrowser from Cloud (Beta)	Never
Applet Module Caching Type	Host
JettyWebServer	Jetty Web Server (started)

Name	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • {ok} indicates that the component is polling successfully. • {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. • {disabled} indicates that the Enable property is set to false. • fault indicates another problem.
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Enabled [general]	true or false	Activates and deactivates use of the function.
Http Port	80 (default)	Specifies the TCP port the service listens on for HTTP client connections, where port 80 is the default.
Http Enabled	true (default) or false	Determines if HTTP client requests are processed. When set to true, turns on a standard Http connection (no communication security) using port 80. When enabled, Fox Enabled in the FoxService must also be set to true (for wbapplet use). When set to false, turns off the standard Http connection causing the system to ignore any attempts to connect using Http port 80. If Https Only is enabled, this setting (false for Http Enabled) is irrelevant.
Https Port	443 (default)	Specifies the TCP port the service listens on for HTTPS (secure) client connections, where port 443 is the default.
Https Enabled	true (default) or false	Determines if HTTPS client requests are processed. When set to true, turns on secure Http communication using port 443. When enabled, Foxs Enabled in the FoxService must also be set to true (for webapplet use). When set to false, turns off the secure Https connection causing the system to ignore any attempts to connect using Https port 443.
Https only	true (default) or false	When set to true, redirects any attempt to connect using a connection that is not secure (Http alone) to Https. When set to false, does not redirect attempts to connect using Http alone.

Name	Value	Description
Https Min Protocol	drop-down list TLSv1.0+ (default) TLSv1.1+ TLSv1.2	The minimum level of the TLS (Transport Layer Security) protocol to which the server will accept negotiation. The default includes versions 1.0, 1.1 and 1.2. It works with most clients, providing greater flexibility than an individual version. During the handshake, the server and client agree on which protocol to use. Change Protocol from the default if your network requires a specific version, or if a future vulnerability is found in one of the versions.
Https Cert	drop-down list of server certificates; defaults to tridium	Specifies the alias of the host platform's server certificate, which the client uses to validate server authenticity. The default identifies a self-signed certificate that is automatically created when you initially log in to the server. If other certificates are in the host platform's key store, you can select them from the drop-down list.
Require Https For Passwords	true (default) or false	When set to true, the HTTPs Enabled property also must be is set to true, or the system disables the New button (for creating a new user in the UserService). This prevents the creation of a password for a new user across a connection that is not secure. When set to false, the New button (for creating a new user in the UserService) remains enabled even if HTTPs Enabled is false. This combination of settings creates security vulnerability when creating passwords for new users and is not recommended.
X Frame Options	Sameorigin (default) Deny or Any (least secure)	The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a <frame> or <iframe>. You can use this to avoid clickjacking attacks, by ensuring that content is not embedded into other sites. If you specify Sameorigin, the page will load in a frame as long as the site including it in a frame is the same as the one serving the page (same server). If a page specifies Sameorigin, browsers will forbid framing only if the top-level origin FQDN (fully-qualified-domain-name) does not exactly match FQDN of the subframe page that demanded the Sameorigin restriction. This is considered a safe practice. If you specify the Deny option, attempts to load the page in a frame will always fail. ***Note:*** • The Deny option inhibits display of some typical Shell Hx Profile views. • If you specify the Any option, then Cross-Frame Scripting (SFS) and Cross-Site Scriptin (XSS) are allowed.

Name	Value	Description
Login Template	Null	When Any is selected, no custom login template is used. When Any is not selected, the option list shows available custom login templates that you can select for a station login page.
Log File directory		Default is file:^^webLogs. The folder in the station’s file space in which log files are stored. Log file names use an YYMMDD.log (date) convention, such as 230501.log for a file created May 1, 2023.
Client Environments		This property is a container for Mobile Client Environment (mobile) entries, available if the station’s host is licensed with the mobile feature. It is used in detection of a user’s browser type (for desktop or mobile) and the selection of the appropriate webProfile for that user.
Show Stack Trace	true or false (default)	
Load JxBrowser from Cloud	Never	
Applet Module Caching Type	Host (default) or User	The default option, Host, results in a folder and the downloading of installation modules to the module folder (n4applet for N4, and wbapplet for AX). This results in the creation of multiple folders of downloaded modules, which negatively affects platform memory usage. The User option results in the creation of a .sharedModuleCache folder. The system then downloads to a sub-folder at this location (n4applet for N4, and wbapplet for AX). This option minimizes the memory required when running in JACE.
JettyWebServer	read-only	Jetty Web Server Started

Client Environments—Mobile

The Client Environments container slot allows the station to automatically detect the user agent of an incoming client and use the appropriate Web Profile for the user:

- Default Web Profile if using a Java-enabled device, such as a PC
- Mobile Web Profile if using a mobile device, such as a cell phone or tablet

Name	Value	Description
Enabled [general]	true or false	Activates and deactivates use of the function.
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • {ok} indicates that the component is polling successfully. • {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. • {disabled} indicates that the Enable property is set to false. • fault indicates another problem.
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
User Agent Pattern	text separated by the pipe symbol (|)	A list of one or more user agents separated by the pipe symbol that identify the target display types.

JettyWebServer

Name	Value
Server State	
Min Threads	3 (default)
Max Threads	30 (default)
N C S A Log	NCSA Request Log

N C S A Log

This is a common format of a standardized text file that web servers use to keep track of processed requests.

Name	Value	Description
Enabled [general]	true or false	Activates and deactivates use of the function.
Retain Days	7 (default)	Limits the size of the log by defining how many days to save log information.
Extended Format	true (default) or false	
Log Cookies	true or false (default)	
Log Time Zone	list of time zones	Identifies the time zone to use for time stamps.

Components in workbench module

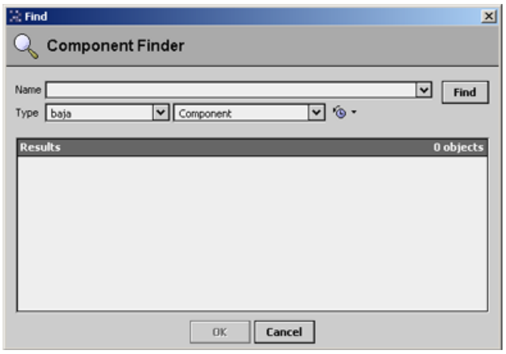
The following components are in workbench module:

- ComponentFinder
- KioskService
- WbFieldEditorBinding
- WbViewBinding
- WsOptions

workbench-ComponentFinder

The ComponentFinder dialog provides a means of finding Components.

Figure 144: Sample Window



Find

Component Finder

Name:

Type: baja, Component

Results: 0 objects

workbench-KioskService

workbench-KioskService is used to enable the Kiosk mode. Kiosk mode provides a way to run a Workbench station so that it fulfills many of the needs of a stand-alone operator workstation as well as many needs of a touchscreen interface. In order to automatically start Kiosk mode, you must have a locally connected display.

Note:

The Kiosk profile is not invoked or displayed by remote browser clients. You cannot use a remote computer with a touchscreen and expect to get the same Kiosk interface.

This component is located in the Workbench palette. To set a station to Kiosk mode, add the service to the station's "Services" folder (Config > Services) and set the enabled property to "true". While Kiosk mode is not limited to touchscreen applications, it does provide advantages that make it well suited for use in a touchscreen application.

workbench-WbFieldEditorBinding

WbFieldEditorBinding is used to bind WbFieldEditors to an object. It allows any existing WbFieldEditor (BooleanFE, EnumFE, AbsTimeFE, etc) to be used in a PX presentation.

workbench-WbViewBinding

WbViewBinding is used to bind WbViews to an object. It allows any existing WbView (PropertySheet, WireSheet, manager view, etc.) to be used in a PX presentation.

workbench-WsOptions

The WireSheet options allow you to view and change the following:

- Show Thumbnail
- Show Grid
- Show Status Colors

These are stored under /user/{user}/wiresheet.options.

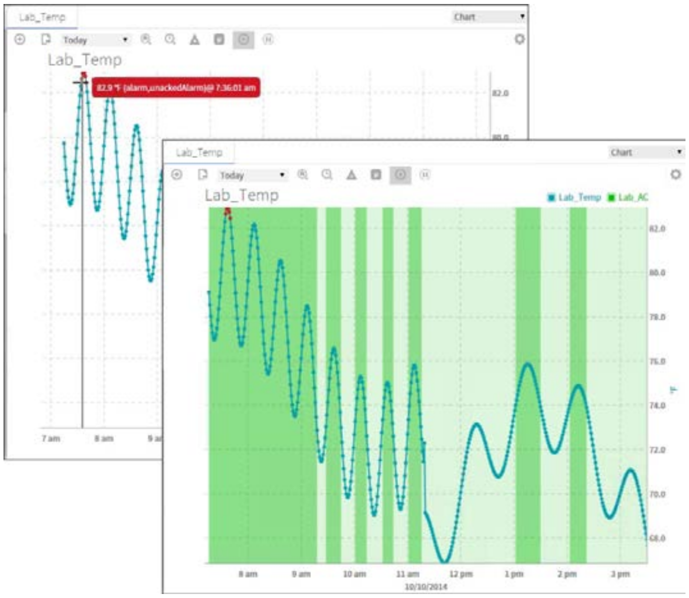
workbench-WebWidget

This is a bajaux, HTML5-based application that incorporates a view with interactive functionality which allows you to edit properties and invoke commands from the view. You can easily add data to a WebWidget, such as the WebChart or Dashboard, simply by dragging one or more components onto the widget. The widget renders in both Workbench and HTML5 Hx interfaces. The widget also integrates into the environment. For example, commands defined for a WebWidget render as added tool bar icons in Workbench, as well as in the HTML5 Hx profile in a web browser.

Examples of the bajaux WebWidget include the following:

- The WebChart displays the Chart view which can display historical data and update with live data. Also, in the view you can easily add data and invoke numerous commands and settings to modify data presentation.

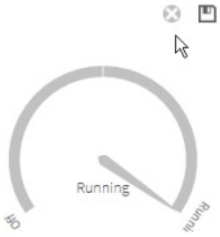
Figure 145: Chart WebWidget



82.9 °F (alarm,unackedAlarm)@ 7:36:01 am

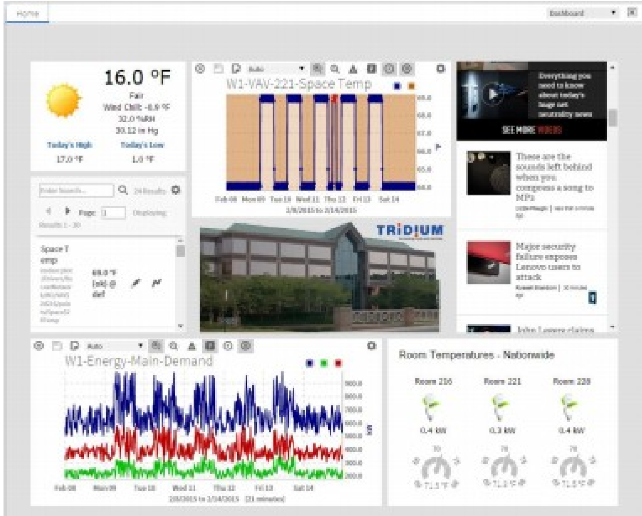
- The CircularGauge displays the graphical gauge view which updates with live data and provides contextual information for the current value. At any time you can dynamically switch the display to another component simply by dragging and dropping a different component onto this widget.

Figure 146: CircularGauge WebWidget



- A Dashboard may be added to any PxPage and displayed in the PxViewer. Additional WebWidgets may be added to the Dashboard pane to customize the presentation of data. The dashboard is used to write dashboard-specific data to and from a station for a specific user.

Figure 147: Dashboard WebWidget



CHAPTER 6 PLUGIN GUIDES

The following topics are covered in this chapter

- Types of plugin modules
- Plugins in backup module
- Plugins in chart module
- Plugins in help module
- Plugins in html module
- Plugins in program module
- Plugins in raster module
- Plugins in timesync module
- Plugins in wiresheet module
- Plugins in wbutil module
- Plugins in workbench module

There are many ways to view plugins (views). One way is directly in the tree. In addition, you can right-click on an item and select one of its views. Plugins provide views of components.

In Workbench, access the following summary descriptions on any plugin by selecting Help→ On View (F1) from the menu, or pressing F1 while the view is open.

Types of plugin modules

Following is a list of the types of plugin modules:

- Plugins in alarm module
- Plugins in backup module
- Plugins in chart module
- Plugins in email module
- Plugins in help module
- Plugins in history module
- Plugins in html module
- Plugins in program module
- Plugins in raster module
- Plugins in schedule module
- Plugins in timesync module
- Plugins in wiresheet module
- Plugins in wbutil module
- Plugins in Workbench module

Plugins in backup module

- backup-BackupManager

backup-BackupManager

The BackupManager is the default view for a station's BackupService.

From this view you can issue a backup command, to back up the station's configuration to your local PC, in dist file format. When you issue a Backup command, a File Chooser window opens to select the destination directory on your PC and the file name for the backup .dist file.

The default backup destination depends on your station connection, as either one of the following:

- Workbench (Fox) — !backups

This is a subdirectory "backups" under your Niagara release directory. If you have not previously made station backups, this directory is automatically created.

- Browser access (Wb Web Profile) — !backups

This is a subdirectory "backups" under the Niagara subfolder of your installed Java 2 runtime environment (Java plugin).

For example: C:\Program Files\Java\j2rel1.4.2_05\niagara\backups. The first time you make a station backup, the system automatically creates this directory.

The default name for a backup file uses a format of: backup_stationName_YYMMDD_HHMM.dist

For example, backup_J403IP98c_200618_1429.dist for a backup made of station "J403IP98c" on June 18, 2020 at 2:29 pm.

The BackupManager provides a progress bar and Job Log (>> control) for an initiated backup. The BackupManager displays a table of the 10 most recent backups, with the following data columns:

- Timestamp: Station date and time when the backup was initiated.
- Host: IP address of the requesting (remote) PC for the backup.
- Path: File path used on the requesting (remote) PC for saving the backup. Typically, this is relative to the default Niagara directory (!), however, it may be an absolute file path.
- User: The station user that initiated the backup.

Plugins in chart module

chart-ResourceManager

The Resource Manager is an available view on any running station. It provides a line chart of both CPU and memory usage of the host platform, and updates in real time. In addition, individual resource statistics are provided in a table, which you can refresh by clicking the Update button.

Plugins in help module

There is one plugin in the help module.

help-BajadocViewer

The BajadocViewer Plugin provides the ability to navigate and browse Baja reference documentation. Baja reference documentation includes both Java API details as well as Baja slot documentation.

The viewer shows documentation for the following:

- Subclasses
- Properties
- Actions
- Topics
- Constructors
- Methods
- Fields

To access Bajadoc, either right-click on a Bajadoc file and select Views→Bajadoc Viewer or select Bajadoc On Target from the main Help menu or popup menu.

BajadocViewer Menus

The BajadocViewer menu functions include:

- Show Baja Only

This option allows you to view only the documentation for slots (Properties, Actions, and Topics). When it is set to false, documentation on the Java constructors, methods and fields is also displayed.

- Flatten Inheritance

This option allows you to flatten the inheritance hierarchy into a single set of documentation. When it is false only the Java members and Baja slots declared in the specified class are displayed. When it is set to true all the Java members and Baja slots inherited from super classes are also shown.

Subclasses

Subclasses provide a subclass tree of all that are subclassed from this item.

Properties

Properties represent a storage location of another Niagara object. Flags are boolean values which are stored as a bitmask on each slot in a Baja Object. Some flags apply to all slot types, while some only have meaning for certain slot types.

Flags

The following table lists the Configuration flags.

Flag	Char	Applies	Description
readonly	r	P	The readonly flag is used to indicate slots which are not accessible to users.
transient	t	P	Transient properties do not get persisted when saving a object graph to the file system. Transient properties are usually also readonly, unless they are designed to be a linkable input slot.
hidden	h	P,A,T	Hidden slots are designed to be invisible to the user, and exist only for Java developers. User interfaces should rarely display hidden slots.
summary	s	P	Summary properties are the focal points of any given BComponent. This flag is used by user interface tools to indicate primary properties for display. This might be as columns in a table or as a glyph in a graphical programming tool.
async	a	A	By default Action are invoked synchronously on the callers thread. By using the async flag on an Action, invocations are coalesced and executed asynchronously at some point in the near future on the engine's thread.
noExecute	x	P	No execute properties prevents start/stop from recursing on properties with this flag set.
defaultOnClone	d	P	Specifies that when an object is cloned via the newCopy () method these properties retain their default value, not the clone source's value.
confirmRequired	c	A	When the Action is invoked by a user, a confirmation dialog must be acknowledged before proceeding.
operator	c	P,A,T	This makes a slot as operator security level. By default when this flag is clear, the slot is an admin security level.
userDefined1	1	P,A,T	User defined.
userDefined2	2	P,A,T	User defined.
userDefined3	3	P,A,T	User defined.
userDefined4	4	P,A,T	User defined.

Actions

An Action is a slot that specifies behavior which may be invoked either through a user command or by an event. Actions provide the capability to provide direction to Components. They may be issued manually by the operator or automatically through links. Actions can be issued by Right-clicking on the Component. The Component bajadoc provides a complete list of Actions available for each Component.

Typical actions include the following:

- Manual actions are available based on Component type. The following are commonly available:

  – Auto (BooleanWritable, NumericWritable and EnumWritable)

  – Active (BooleanWritable)

  – Inactive (BooleanWritable)

  – Override (NumericWritable and EnumWritable)

Each of the above actions is issued to the priorityArray of the Component at level 8 (Manual Operator).

- Many other Actions are available on other Components. Each Action available for a Component is listed in the Actions sect2 of the Component bajadoc.

Topics

Topics represent the subject of an event. Topics contain neither a storage location, nor a behavior. Rather a topic serves as a place holder for an event source.

Plugins in html module

This module has two plugins.

html-WbHtmlView

This component allows you to view the contents of HTML files.

WbHtmlView Menus

The Workbench main menu functions are available.

WbHtmlView Toolbar

In the WbHtmlView, the toolbar contains navigation and editing buttons. In addition, Find, FindNext and FindPrev toolbar buttons are available.

HTML Support: HTML Tags

The WbHtmlView attempts to ignore mismatched or missing Tags. It can parse any HTML, no matter how badly messed up the Tags are, although it might do a pretty bad job of rendering the results. It does simplistically attempt to repair missing <p>, but other problems are solved by ignoring Tags. Warnings will appear on the command line showing its best guess as to what the problem was.

Valid tags include:

- a - anchor Ex:<a href="#fragment">Title<a> and <a name="fragment">Title<a> (Attributes href and name)
- applet - not supported
- b - bold text style Ex: <b>bold text<b>
- body - document body
- br - forced line break
- code - computer code fragment
- col - not supported
- colgroup - not supported
- dd - definition description
- div - not supported
- dl - definition list
- dt - definition term
- em - emphasis Ex: <em>emphasis text<em>
- font - local change to font (Attributes Color, name and size)
- h1 - heading Ex:<h1 class='title'>Title<h1>
- h2 - heading
- h3 - heading
- h4 - heading
- h5 - heading
- h6 - not supported
- head - document head
- hr - horizontal rule
- html - document root element
- i - italic text style
- img - embedded image Ex: <img src="../doc/icons/x16/home.png"> An <img> without an align attribute is treated as an 'inline' image. An align of left or right causes text to flow around the image.
- li - list item
- link - a media-independent link Ex:<link rel='StyleSheet' href='module://bajaui/doc/style.css' type='text/css'/>
- meta - not supported
- object - not supported
- ol - ordered list
- p - paragraph Ex:<p class='note'>Note<p>
- pre - preformatted text
- span - not supported
- style - not supported

- table - table (Attributes align, border, bordercolor, cellpadding, cellspacing and width)
- tbody - not supported
- td - table data cell supports ALIGN (left, right, and center), BGCOLOR, COLSPAN, ROWSPAN and WIDTH
- tfoot - not supported
- th - table header cell supports ALIGN (left, right, and center), BGCOLOR, COLSPAN, ROWSPAN and WIDTH
- thead - not supported
- title - document title
- tr - table row supports ALIGN (left, right, and center) and BGCOLOR
- tt - teletype or monospaced text style
- ul - unordered list

HTML Attributes

HTML Elements have associated properties, called attributes, which may have values. Attribute/value pairs appear before the final ">" of an element's start tag.

Valid attributes include:

- align - vertical or horizontal alignment Deprecated; elements: img, object? values: bottom, left, middle, right or top.
- align - table position relative to window Deprecated; elements: table; values: center, left or right
- align - align, text alignment Deprecated; elements: div?, h1?, h2?, h3?, h4?, h5?, h6?, p; values: center, justify, left or right
- align - alignment; elements: col?, colgroup?, tbody?, td, tfoot?, th, thead?, tr; values: center, char, justify, left or right
- bgcolor - background Color Deprecated; elements: h1, h2, h3, h4, h5, h6?, p, td, th, tr
- border - controls frame width around table; elements: table
- cellpadding - spacing within cells; elements: table
- cellspacing - spacing between cells; elements: table
- class - class name or set of class names for stylesheets; elements: most elements
- color - text Color Deprecated; elements: font, h1, h2, h3, h4, h5, h6?, p, pre, td?, th?, tr?
- colspan - number of cols spanned by cell; elements: td, th
- content - associated information; elements: meta?
- href - URI for linked resource; elements: a, link
- id - name to an element; elements: most elements
- lang - Language Code; elements: most elements not supported
- name - named link end; elements: a
- name - meta information name; elements: meta?
- rel - forward link types; elements: link; values: stylesheet
- rowspan - number of rows spanned by cell; elements: td, th
- size - size of font; elements: font; value: fixed 1-7 or relative -7 to +7
- summary - purpose/structure for speech output; elements: table
- type - advisory content type; elements: link; values: text/css
- width - table width; elements: table

Character Entity References

Character Entity References are supported including the following:

- & - ampersand &
- &apos; - apostrophe '
- &copy; - copyright ©
- &gt; - greater than >
- &ldquo; - double quotation, left “
- &lsquo; - single quotation, left ‘
- &lt; - less than <
- &mdash; - em dash —
- &nbsp; - non breaking space " "
- &ndash; - en dash –
- &rdquo; - double quotation, right ”
- &rsquo; - single quotation, right ’
- &quot; - quotation mark "
- &reg; - registered trademark ®
- &trade; - trademark ™

Stylesheet support

Stylesheets are supported using a link in the header as follows:

```
<head>

<title>Sample<title>

<link rel='StyleSheet'  href='module://bajaui/doc/style.css'  type='text/css'/>

<head>
```

See the stylesheet: module://bajaui/doc/style.css or the CSS stylesheet used for this document at docbook.css.

Pseudo-classes and Pseudo-elements

Anchor pseudo-classes include:

- A:link unvisited links unsupported
- A:visited visited links unsupported
- A:active active links unsupported

CSS1 Properties

Stylesheet elements supported include:

- background - The 'background' property is a shorthand property for setting the individual background properties (i.e., 'background-color', 'background-image', 'background-repeat', 'background-attachment' and 'background-position') at the same place in the style sheet.
- background-attachment unsupported
- background-color - This property sets the background Color of an element.
- background-image unsupported
- background-position unsupported
- background-repeat unsupported
- border unsupported
- border-bottom unsupported

- border-bottom-width unsupported
- border-color unsupported
- border-left unsupported
- border-left-width unsupported
- border-right unsupported
- border-right-width unsupported
- border-style unsupported
- border-top unsupported
- border-top-width unsupported
- border-width unsupported
- clear unsupported
- color - This property describes the text Color of an element (often referred to as the foreground Color).
- display unsupported
- float unsupported
- font unsupported
- font-family unsupported
- font-size unsupported
- font-style unsupported
- font-variant unsupported
- font-weight unsupported
- height unsupported
- letter-spacing unsupported
- line-height unsupported
- list-style-image unsupported
- list-style-position unsupported
- list-style-type unsupported
- margin unsupported
- margin-bottom unsupported
- margin-left unsupported
- margin-right unsupported
- margin-top unsupported
- padding unsupported
- padding-bottom unsupported
- padding-left unsupported
- padding-right unsupported
- padding-top unsupported
- text-align
- text-decoration unsupported
- text-indent unsupported
- text-transform unsupported
- white-space unsupported
- width unsupported
- word-spacing unsupported

html-SpyViewer

SpyViewer allows you to view diagnostic information about the system. It contains the following:

- sysinfo: sysinfo provides system information.
- stdout: stdout provides access to standard output.
- systemProperties: systemProperties provides access to system properties.
- logSetup: logSetup allows you to config your log severities dynamically. There is also an option to flush the current settings to log.properties.
- sysManagers: sysManagers provides information on managers. These include:

  – registryManager

  – schemaManager

  – componentNavEventManager

  – moduleManager

  – engineManager

  – leaseManager

  – serviceManager

  – licenseManager

  – stationManager
- navSpace: provides information on the navSpace.
- userinterface: provides information on the user interface framework.
- fox: fox provides information on fox client and server sessions.

Plugins in program module

The following plugins are in the program module:

- BatchEditor
- ProgramEditor
- ProgramModuleBuilder
- RobotEditor

program-BatchEditor

The Batch Editor is the default view on the ProgramService. It allows you to perform a variety of operations on slots of multiple components by issuing a single “batch” command.

You can add (specify) components using dragfrom the Nav tree, or copy and paste into the view, or by using the Find objects (Bql Query Builder) function—or any combination of the three methods.

As needed, use the ⊖ Clear Selected Items control to remove any items before running the operation. The Batch Editor can be a real time saver when the same configuration change needs to be replicated among multiple component slots.

program-ProgramEditor

The ProgramEditor view provides the ability to view and edit Program Components. To view, right-click a Program and select ProgramEditor. It shows Edit, Slots, Imports and Source tabs.

ProgramEditor Edit

Edit allows you to edit the onExecute, onStart, onStop and freeForm methods. An example from the demo Database follows:

```
BStatusNumeric inA  = getInA(); BStatusNumeric inB = getInB(); BStatusNumeric out  =  getOut();

out.setValue(  inA.getValue() +  inB.getValue() );
```

ProgramEditor Slots

Slots allows you to view and change the slots of the program component. It includes Slot, #, Name, Display Name, Definition, Flags and Type for each slot.

ProgramEditor Imports

This allows you to view the modules that have been imported. It also allows the following:

- ProgramEditor ImportType: ImportType allows you to import a new type.
- ProgramEditor ImportPackage: ImportPackage allows you to import a new type.
- ProgramEditor Remove: ProgramEditor Remove allows you to import a new type.

ProgramEditor Source

Source allows you to view and edit the source of the program component. The editor supports special Color coding for Java Files. An example from the demo Database follows:

```
/*  Auto-generated  ProgramImpl Code   */

import java.util.*;   /* java Predefined*/ import javax.baja.sys.*;      /* baja Predefined*/ import javax.baja.status.*;  /* baja
Predefined*/ import javax.baja.util.*;   /* baja Predefined*/ import com.tridium.program.*;          /*  program Predefined*/

public class  ProgramImpl

extends  com.tridium.program.ProgramBase

{

////////////////////////////////////////////////////////////////

// Getters

////////////////////////////////////////////////////////////////

public  BStatusNumeric getOut() {  return  (BStatusNumeric)get("out");  } public  BStatusNumeric getInA()  {  return
(BStatusNumeric)get("inA"); } public  BStatusNumeric getInB() {  return  (BStatusNumeric)get("inB"); }

////////////////////////////////////////////////////////////////

// Setters

////////////////////////////////////////////////////////////////

public void setOut(javax.baja.status.BStatusNumeric  v) {  set("out", v); } public void setInA(javax.baja.status.BStatusNumeric  v) {
set("inA",  v);  } public void setInB(javax.baja.status.BStatusNumeric  v) {  set("inB",  v);  }

////////////////////////////////////////////////////////////////

// onExecute

////////////////////////////////////////////////////////////////

public  void onExecute()

throws  Throwable

{

BStatusNumeric inA   =  getInA();

BStatusNumeric  inB =  getInB(); BStatusNumeric out  =  getOut();

out.setValue(  inA.getValue() +  inB.getValue() );

}

}
```

ProgramEditor Menus

The Workbench main menu functions are available. When the ProgramEditor is visible, the following ProgramEditor menu function is also available:

- ProgramEditor ImportType
- ProgramEditor ImportPackage
- ProgramEditor Remove
- Add Slot Ctrl + A
- Delete
- Rename Slot Ctrl + R
- Reorder
- ProgramEditor Compile & Save F9
- ProgramEditor Compile Ctrl + F9

ProgramEditor Toolbar

The Workbench toolbar contains navigation and editing buttons When the ProgramEditor is visible, additional toolbar buttons include the following:

- Find F5
- Replace F6
- ProgramEditor Compile & Save F9
- ProgramEditor Compile Ctrl + F9
- Console Prev
- Console Next

ProgramEditor Compile & Save

CompileSave allows you to compile and save the source of the program component. The shortcut is F9.

ProgramEditor Compile

Compile allows you to compile the source of the program component. The shortcut is Ctrl + F9.

program-ProgramModuleBuilder

The Program Module Builder is the default view on the ProgramModule. It lets you create a module from one or more Program components, such that they may be versioned and provisioned just like other modules.

program-RobotEditor

The RobotEditor is used to write Robots that can be run via the ProgramService.

RobotEditor Menus

The Workbench main menu functions are available. When the RobotEditor is visible, the following RobotEditor menu functions are also available:

- RobotEditor Compile Ctrl + F9
- RobotEditor Compile & Run F9

RobotEditor Toolbar

The Workbench toolbar contains navigation and editing buttons as described in “About the toolbar”. When the RobotEditor is visible, additional toolbar buttons include:

- Find F5
- Replace F6
- RobotEditor Compile Ctrl + F9
- RobotEditor Compile & Run F9
- Console Prev (?need to find SearchConsolePrev.html source?)
- Console Next (?need to find SearchConsoleNext.html source?)

RobotEditor Compile

Compile allows you to compile the source of the Robot component. The shortcut is Ctrl + F9.

RobotEditor Compile & Run

Compile& Run allows you to compile and run the source of the Robot component. The shortcut is F9.

Plugins in raster module

This module has one plugin.

raster-RasterViewer

The RasterViewer displays bitmapped image files: GIF, JPEG, PNG in the main window with Format and image Size at the bottom.

Plugins in timesync module

timesync-TimeSyncManager

The TimeSyncManager is used to display the status of TimeSyncClients, as well as add new or edit existing TimeSyncClients. By default, it displays Server Name, Status, Poll Delta, Poll Local Time, and Poll Server Time for each TimeSyncClient.

Plugins in wiresheet module

wiresheet-WireSheet

The WireSheet view shows the contents of this component. It can be used on a component of a running station or a component in a Bog File. If in a running station, it is active and real-time updates are provided. In order to command or select a different view of the item, you may right-click to get the popup menu.

WireSheet Menus

The WireSheet includes the following menus:

- WireSheet Main Menu
- WireSheet Component Menu
- WireSheet Background Menu
- WireSheet Link Menu

WireSheet Main Menu

The WireSheet main menu functions are available. When the *wiresheet; is visible, the following WireSheet menu functions are also available:

- Delete Links
- Arrange
- Select All
- Show Thumbnail
- Show Grid
- Show Status Colors

WireSheet Component Menu

If you right-click any Component in the WireSheet, you can choose from the following:

- Views: Go to any of the views of the Component.
- Actions: Perform any of the Actions on the Component.
- Cut: Ctrl + X
- Copy: Ctrl + C
- Paste: Ctrl + V
- Duplicate: Ctrl + D
- Delete
- CompositeEditor
- Rename
- Reorder
- Pin Slots

WireSheet Background Menu

If you right-click the Background of the wire sheet, you can choose from the following:

- Cut: Ctrl + X
- Copy: Ctrl + C
- Paste: Ctrl + V
- Duplicate: Ctrl + D
- Delete
- Delete Links
- Arrange
- Select All
- workbench-CompositeEditor

WireSheet Link Menu

If you right-click any link in the wire sheet, you can choose from the following:

- Cut: Ctrl + X
- Copy: Ctrl + C
- Paste: Ctrl + V
- Duplicate: Ctrl + D
- Delete
- Delete Links
- Arrange
- Select All
- CompositeEditor

WireSheet Toolbar

The Workbench toolbar contains navigation and editing buttons as described in “About the toolbar”. When the WireSheet is visible, additional toolbar buttons include the following:

- Delete Links: Delete Links may be accessed from the Menu under Edit or from the toolbar. It allows you to eliminate the selected link(s). You can Delete Links only in the WireSheet.
- Pin Slots: Pin Slots provide the capability to make Properties of a Component visible in the WireSheet view of the Component. Right-click on the Component and select Pin Slots. This will bring up the Pin Slots window.

The selected Properties are now visible in the WireSheet.

- Arrange: Arrange allows you to arrange the items in the WireSheet. You can Arrange All or Arrange Selection.
- Select All: Select All allows you to select all items in the WireSheet.
- Show Thumbnail: Show the thumbnail view in the upper right corner of the wire sheet. This allows you to toggle whether this function is enabled. This function may be accessed from the menu under WireSheet. It allows you to display a thumbnail of the WireSheet view in the corner of the wire sheet. You can use the thumbnail to move around the wire sheet by dragging the shaded area in the thumbnail. You can also hold down the Ctrl key and move the thumbnail around the wire sheet to keep it out of your way.
- Show Grid: When enabled, the grid is displayed in the background of the wire sheet. This helps you to align Components when you move them. You may control whether the grid layer is enabled or visible from the WireSheet menu or the Tools→Options menu. This allows you to enable and disable this function.

Show Status Colors

Show the Status Colors. This allows you to toggle whether this function is enabled. This function may be accessed from the Menu under WireSheet. It allows you to display StatusColors in the WireSheet.

workbench-PrintDialog

Print provides the capability to print the WireSheet. The WireSheet will remain fixed at logical size of 100 x 100 "blocks". Printing will scale the WireSheet drawing to fit the page size (minus header and footer). Therefore, if you build your WireSheet logic in the top left corner, you will get a large scale for your print out. If you use the entire WireSheet down to the bottom right corner, you will get a much smaller scale for your print out.

Margins are 1" for left and 1/2" for top, bottom, and right. A standard header is included which displays container's reference and current date. A future version will create a footer listing external links and knobs.

workbench-CompositeEditor

Composite Editor provides the capability to expose child slots of a Component as slots on the parent. Right-click on the Component and select Composite Editor. This will bring up the Composite Editor Dialog.

The selected Properties are now visible in the WireSheet.

workbench-ExportDialog

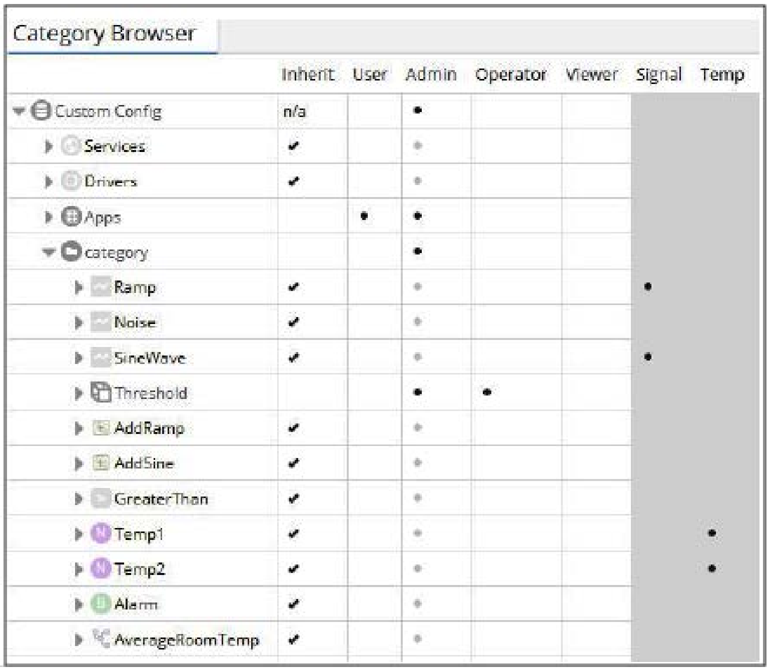
Export provides the capability to Export views. You can export any table found in the Workbench (anything built with bajaui:Table). The Table options menu (that little button to the right of the header), includes a Print and Export commands, which allow you to export the table to PDF, HTML, or CSV (ExcelFile). Export uses the current sort order and visible columns currently displayed.

Plugins in wbutil module

wbutil-CategoryBrowser

This view is the default view of the station’s CategoryService, and typically where you spend most of your time assigning categories to components after initially creating the categories.

Figure 148: Category Browser



Category Browser

	Inherit	User	Admin	Operator	Viewer	Signal	Temp
Custom Config	n/a		●				
Services	✔		•				
Drivers	✔		•				
Apps		●	●				
category			●				
Ramp	✔		•			●	
Noise	✔		•				
SineWave	✔		•			●	
Threshold			●	●			
AddRamp	✔		•				
AddSine	✔		•				
GreaterThan	✔		•				
Temp1	✔		•				●
Temp2	✔		•				●
Alarm	✔		•				
AverageRoomTemp	✔		•				

There is no need to define component-to-category associations for Tagged Categories, so each Tagged Category column is grayed out and cannot be edited in the Category Browser. Any component with a tag that satisfies the NEQL query is visible in the Category Browser.

The following table describes the columns in the Category Browser.

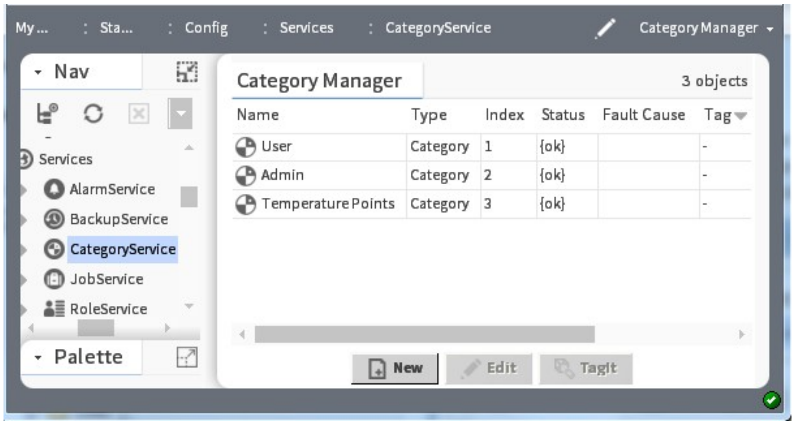
Column	Value	Description
Inherit	check mark or blank	A check mark indicates that the object inherits the category from its parent in the table.
User	Category 1	All system objects except for those listed as assigned to Admin are assigned to this category.
Admin	Category 2	These objects default to the Admin category: • The configuration services: UserService, CategoryService, and ProgramService • All files (the entire file space)
Categories 3–10	bold bullet, grayed out bullet, or blank	Objects with a bold bullet have tags that satisfy a tagged category. A bold bullet indicates that the object is assigned to the category. A grayed out bullet indicates inheritance. Blank indicates that the category has not been assigned.

wbutil-CategoryManager

This view of the CategoryService allows you to create, enable and delete the groups that the security model uses to control access to the objects in a station. Once you create categories, you use the Category Browser view to centrally assign system objects to categories. Or, at the individual component level, you use a component’s Category Sheet view to assign the component to one or more categories.

You can assign an object to many categories at the same time. Each object stores its own categories.

Figure 149: Category Manager with tagged category, Temperature Points as Category 3



Category Manager 3 objects

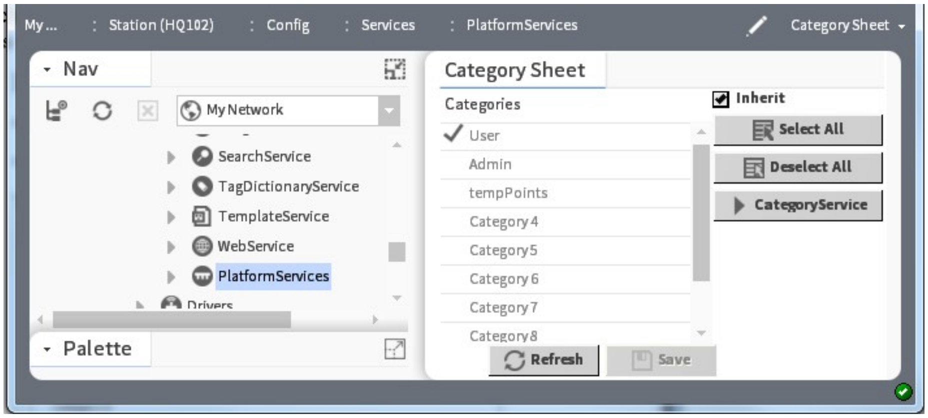
Name	Type	Index	Status	Fault Cause	Tag
User	Category	1	{ok}		-
Admin	Category	2	{ok}		-
Temperature Points	Category	3	{ok}		-

Column	Value	Description
Name	text string	Descriptive text that reflects the purpose of the entity or logical grouping.
Type	Category or Tagged Category	A basic Category is a name that is associated with individual objects. A Tagged Category includes a NEQL query that returns objects with tags that satisfy the query.
Index	number	A unique number for the category, as it is known to the station.
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • {ok} indicates that the component is polling successfully. • {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. • {disabled} indicates that the Enable property is set to false. • fault indicates another problem.
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Tag Query Name	text	A descriptive name to represent the results of the search.
Tag Query	NEQL	A NEQL query. This property is required when Type is Tagged Category.

wbutil-CategorySheet

This view assigns a component to one or more categories (or configures it to inherit categories from its parent. Every component has a Category Sheet view.

Figure 150: Category Sheet

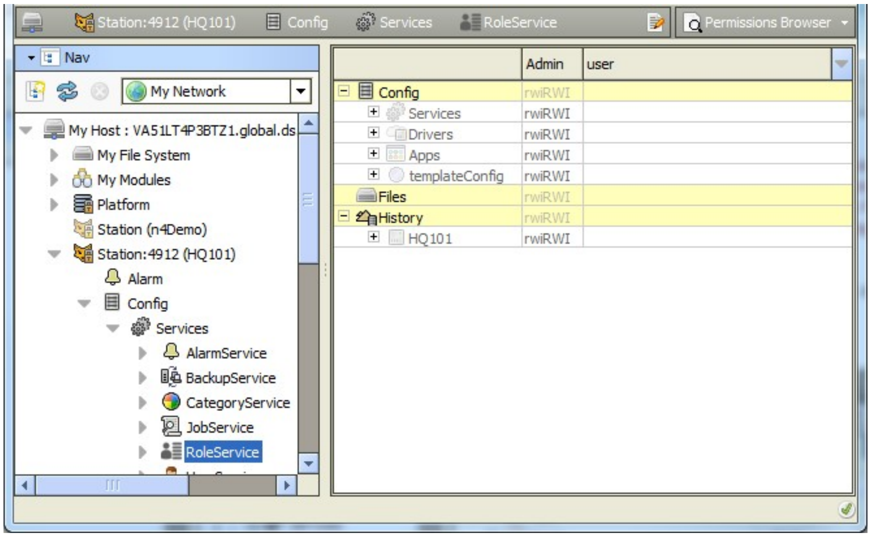


Option/button	Value	Description
Categories	text	Provides one table row for each category name.
Inherit	toggle	A check mark indicates that the component belongs to the same categories as its parent component. No check mark allows you to make explicit category assignments for this component.
Select All	button	Effective if Inherit is cleared, clicking this button assigns this component to all categories in this station.
Deselect All	button	Effective if Inherit is cleared, clicking this button removes this component from all categories.
CategoryService	button	Opens the Category Browser.
Refresh	button	Redisplays the Category Sheet.
Save	button	Records the changes made.

wbutil-PermissionsBrowser

This view allows you to quickly review the objects that someone, who has been assigned a given role, may access. You access this view by right-clicking RoleService in the Nav tree and clicking Views→Permissions Browser.

Figure 151: Permissions Browser view



	Admin	user
Config	rwiRWI	
Services	rwiRWI	
Drivers	rwiRWI	
Apps	rwiRWI	
templateConfig	rwiRWI	
Files	rwiRWI	
History	rwiRWI	
HQ101	rwiRWI	

Columns represent roles, and rows identify the objects in the station, with each table cell showing user permissions.

- Yellow rows are objects explicitly assigned with permissions.
- Dimmed rows represent objects that inherit their permissions from their parent object.

Double-click a cell to bring up the permissions dialog for that role. This allows you to globally change a user’s permission levels for any category in the station.

Column	Value	Description
First column	Nav tree for station Config, Files and History	Each Nav tree node occupies a row in the table. This expandable tree lets you navigate to objects of interest to review current permissions.
Admin	permissionsR = readW = writeI = invoke actionadmin level permissions appear in upper case.	Reports the rights assigned to the admin role. As this is a super user, admin has rights to read, write and invoke an action for all objects.
user	permissionsr = readw = writei = invoke actionoperator permissions appear in lower case.	Reports the rights assigned to the user role. The default is no rights assigned.

wbutil-ResourceEstimator

The Resource Estimator tool allows you to estimate station resources based a number of variables, which you enter in various fields. It is one of several tools in Workbench’s Tools menu.

wbutil-TodoList

The Todo List tool allows you to enter, summarize, group, and prioritize pending Workbench tasks. It is one of several tools in Workbench’s Tools menu.

wbutil-UserManager

The User Manager is the primary view of the UserService. You use it to add, edit, and delete users for accessing the station.

Plugins in workbench module

There are a number of views in the workbench module.

workbench-CollectionTable

The CollectionTable allows you to view tables. One way to create a table is through a BQL collection like the following: local:|fox:|station:|slot:/ControlObjects|bql:select displayName,type, out, facets from co

The Table options menu, allows you to Reset Column Widths, Print and Export.

workbench-DirectoryList

The DirectoryList view provides a listing of the subdirectories and files found in a given Directory. Double-clicking an item opens its default view. Files are displayed with an icon based on file type.

DirectoryList Menus

The Workbench main menu functions are available. When the DirectoryList is visible, the following popup menu functions are also available:

- Views: View menus for the Component.
- Actions: Perform any of the Actions on the Component.
- New: Create a new file of standard types (if a Directory).
- Cut Ctrl + X
- Copy Ctrl + C
- Paste Ctrl + V
- Duplicate Ctrl + D
- Delete Delete
- Rename
- Reorder

DirectoryList Toolbar

The Workbench toolbar contains navigation and editing buttons as described in “About the toolbar”.

Refresh: Refresh allows you to synchronize the cached components with the actual file system.

New: This allows you to make a new Folder or File in the selected Folder of these types:

- New Folder allows you to make a new Folder in the selected Folder.
- BogFile.bog
- HtmlFile.html
- JavaFile.java
- NavFile.nav
- PrintFile.print
- TextFile.txt

workbench-HexFileEditor

The HexFileEditor allows you to view hexadecimal files. It provides a binary view of a file's contents.

workbench-JobServiceManager

The Job Service Manager is the default view on a station’s JobService. It provides a table listing of up to the last 10 Jobs executed by the station since the last station start. Order is oldest job at top, most recent job at bottom.

To see details on any job, click the button next to its status descriptor. A popup Job Log window displays all the interim steps for the job, including timestamps and relevant messages.

To dispose of any job, click the close (X) button to remove it from the station.

Note:

Only the last ten jobs are saved. The system clears all jobs when it restarts.

workbench-LinkSheet

The LinkSheet allows you to view and delete links. It provides a view of the links on a Component.

LinkSheet Main Menu

The Workbench main menu functions are available. When the LinkSheet is visible, the followingLinkSheet menu functions are also available:

- Delete Links
- ⇨ GoTo

LinkSheet Toolbar

The Workbench toolbar contains navigation and editing buttons. When the LinkSheet is visible, additional toolbar buttons include:

- Delete Links
- ⇨ Go To goes to the selected item

workbench-ModuleSpaceView

The Module Space View allows you to view Modules.

OptionsButton

The Options button enables and disables columns (turns them on and off).

workbench-NavContainerView

Nav Container View is a default listing of nNav children.

workbench-NavFileEditor

The NavFileEditor allows you to view and edit Nav Files. It provides a view of the pages in the station and a view of the Nav tree. You can drag pages to the Nav tree to add them to the Nav File. The name and Ord are shown at the bottom of the window.

To use the Nav File, place its filename in the Default Nav File property of the WebService.

workbench-PropertySheet

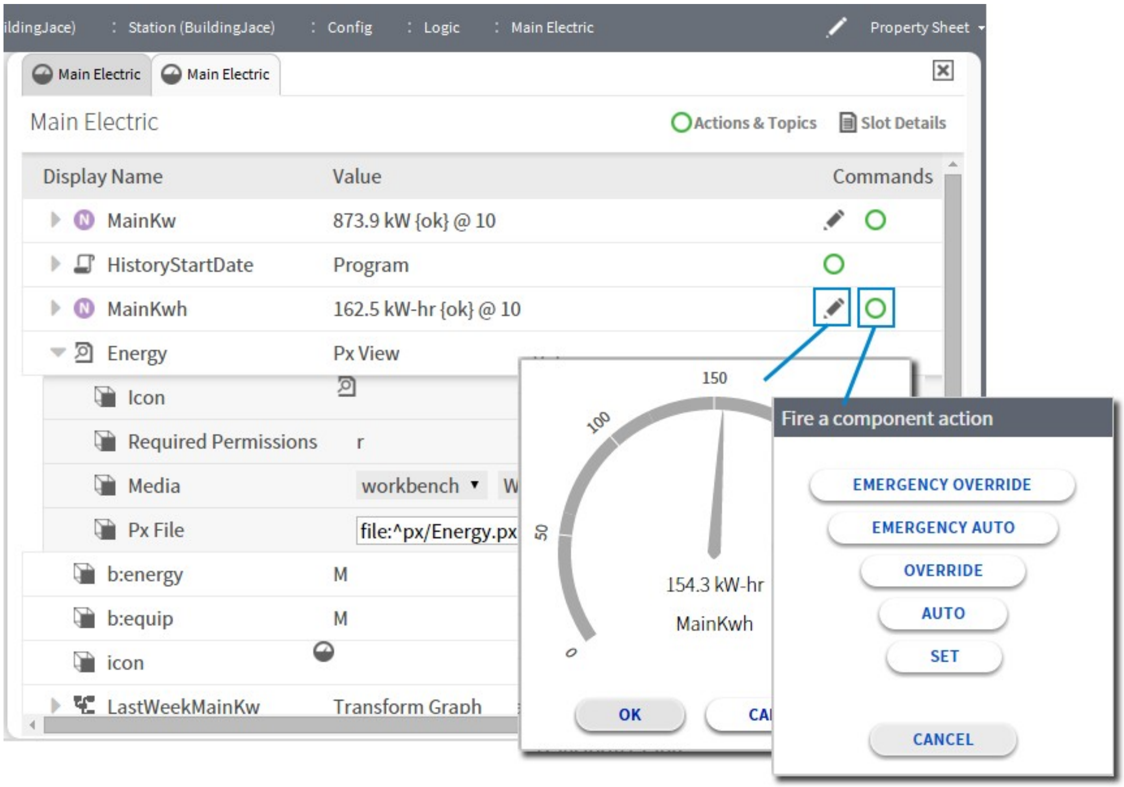
The property sheet views display all of the user visible properties of the selected component. You can change any properties that you have permissions to change. The property sheet views apply to a component of a running station or a component in a bog file. To see properties of components in a PropertySheet, expand each component.

There are two types of Property Sheet views:

- Property Sheet view

An HTML5 Property Sheet view, shown here, which provides functionality such as interactive field editors and action commands, graphical web gauge display for points, and slot sheet details. Property changes that you make in this view are saved automatically.

Figure 152: HTML5 Property Sheet view



Display Name	Value	Commands
MainKw	873.9 kW {ok} @ 10	
HistoryStartDate	Program	
MainKwh	162.5 kW-hr {ok} @ 10	
Energy	Px View	
Icon		
Required Permissions	r	
Media	workbench ▾ W	
Px File	file:^px/Energy.px	
b:energy	M	
b:equip	M	
icon		
LastWeekMainKw	Transform Graph	

- AX Property Sheet view

The default property sheet view for the NiagaraAX and Niagara 4 releases.

In the AX Property Sheet view, if you want to enter a URL, copy the value and paste it into the property sheet. When you change any property, its symbol will become red until you press Save.

AX Property Sheet Menus

The Workbench main menu functions are available. If you right-click any component in the AX Property Sheet, you can choose from the following:

Item	Description
Views	Goes to any of the views of the selected component.
Actions	Perform any available actions on the component.
New	Create a new item of standard types.
Edit Tags	Opens the Edit Tags window, permitting you to add or remove tags on a component.
Make Template	Creates a template of the selected root component, collects all associated Px and graphic files, and invokes the Template view, allowing you to configure the template for deployment.
Cut	Copies to clipboard and after pasting the copy, deletes the cut item.
Copy	Copies to clipboard, copied item remains
Paste	Pastes the copied item from the clipboard
Duplicate	Makes a copy in the same location as the original item
Delete	Removes the item
Find	This menu item displays the Component Finder window.
Link Mark	Sets a selected component to your popup menu, making it temporarily available for “Linking From” or “Linking To” other points.
Link From	Allows you to link to a selected component from another component that has been marked, using Link Mark from the popup menu.
Link To	Allows you to link from a selected component to another component that has been marked, using Link Mark from the popup menu.
Relation Mark	Sets a selected component to your popup menu, making it temporarily available for “Relating From” or “Relating To” other points.
Relate From	Allows you to add a relation from selected component to another that has been marked, using Link Mark from the popup menu.

Item	Description
Relate To	Allows you to add a relation to a selected component from another that has been marked, using Link Mark from the popup menu.
Rename	Allows you to rename the selected component’s actual slot name (as it appears in the Ord). This menu item displays the Rename window. You can only rename one component at a time.
Set Display Name	Allows you to set a display name for the selected component (as it appears in a Nav tree, and in the Wire Sheet, and Property Sheet views).
Reorder	Opens the Reorder Points window, which provides the following commands for reordering points within the selected parent component. • Move Up • Move Down • Sort by Name • Sort by Type • Reset
Composite	This menu item opens the Composite Editor window.
Export	Exports a selected component to the oBix .xml format.
Config Flags	Allows you to assign or remove permissions flags on a component. This is available on properties in the AX Property Sheet view. Right-click a property to see the Config Flags window.

PropertySheet Toolbar

The Workbench toolbar contains navigation and editing buttons.

Config Flags: Configure Flags on a component. This is available on Properties. Right-click a property to open the Config Flags window.

Links: Links provide the mechanism to dynamically attach Components. They allow you to attach an output Property of one Component to an input Property of another Component. The links are visible in the Property Sheet of the Component. Links show whether they are direct or indirect and their source Property.

To view the properties of the link, left-click the plus sign (+) on the Component. Each link shows the following:

- Link Type and source
- Source Ref: This is the linked Component.
- Source Slot Name: This is the linked Component's property.
- Target Slot Name: This is the current Component's property.
- Enabled: This shows whether the link is currently enabled.

OrdChooser

To select an Ord instead of typing it, you can use one of the Ord editors. They include the following:

- Bql Builder
- Directory Ord Chooser
- File Ord Chooser
- History Ord Chooser
- Ref Chooser

Once you have selected an editor, you can use the » button to the right of the selection box. It will present the selected editor.

workbench:FileOrdChooser

In order to select a file Ord instead of typing it, you can use the FileOrdChooser.

You can select FileSpaces, Bookmarks and directories and files. In addition toolbar functions are provided including the following:

- ⇦ Back: The back icon moves returns to the previous view.
- Uplevel: The uplevel icon moves up one level in tree.
- Home: The home icon moves up one level in tree.
- New Folder: The new folder creates a new directory.
- ListView: ListView opens the list view.
- DetailsView: DetailsView opens the details view.
- Bookmarks: The bookmarks icon () and Bookmarks button add the selection to Bookmarks.
- Show/Hide Preview: Show/Hide Preview opens and hids the preview pane.

workbench:DirectoryOrdChooser

In order to select a Directory Ord instead of typing it, you can use the DirectoryOrdChooser.

You can select FileSpaces, Bookmarks and directories. In addition toolbar functions are provided including the following:

- New Folder: New Folder creates a new Directory.
- Bookmarks: You can press the Bookmarks button to add the selection to Bookmarks.

workbench:HistoryOrdChooser

To select a reference Ord instead of typing it, use the HistoryOrdChooser. You can select items from the tree by type slot or Handle.

workbench:RefChooser

To select a reference Ord instead of typing it, use the RefChooser. You can select items from the tree by type slot or Handle.

workbench:BqlQueryBuilder

To select a BQL query instead of typing it, use the BqlQueryBuilder.

BQL is one Scheme used to Query in the Niagara Framework. An Ord is made up of one or more Queries. A Query includes a Scheme and a body. The bql Scheme has a body with one of the following formats:

- BQL expression
- Select projection FROM extent Where predicate

You can create the Ord Qualifier, Select, FROM and Where portions of a Query.

Ord Qualifier

In the left window, you can select an Ord to use as the qualifier. It will immediately be placed in the BQL statement at the top when you select it.

If you select the history Scheme, your options will vary from those shown here.

workbench:ProjectionBuilder

To build the projection for a BQL request instead of typing it, use the ProjectionBuilder in Bql Builder. You can select an item from the center window to use with the select statement. Press the ⇨right arrow to add each one to the projection.

Bql Expressions"

An expression can be one of the following:

- field like displayName
- x.y.z
- any method that returns non-void and takes zero parameters

After you build the BQL statement, you can remove the "Select" to provide a BQL expression instead of a Select statement.

workbench:ExtentBuilder

An extent can be one or more of the following:

- "*" all available from the target
- all property slots
- all methods that return non-void and take zero parameters

In order to build the Extent for a BQL request instead of typing it, you can use the ExtentBuilder in Bql Builder. You can select the Restrict Type and choose the module and item to use with the FROM clause. It will immediately appear in the BQL statement at the top.

This also changes the items that are available in the projection.

If you choose a history Scheme, the ExtendBuilder will provide different selections. The extent for a History typically can be one of the following:

- "from /demo/Float" where demo is the station name
- "from !Float" where "!" signifies the current station

workbench:QualifierBuilder

To build the Qualifier for a BQL request, type the request in the QualifierBuilder of the Bql Builder. The text you type immediately appears in the BQL statement at the top.

Edit Facets

Edit Facets allows for the viewing and editing of facets. To change facets, use the button to the right of the facets. It opens the Edit Facets window.

From here you can Add, Remove or select the Enum Range window.

Add: This adds a Key and Type.

Remove: This Remove deletes facets.

Enum Range window: To view or set the range of values for a Enum, use the »button. It opens the Enum Range window.

You can enter one of the standard Enumerations in the field under the Use Enum Type in Range (module:name).

Press Use Enum Type in Range (module:name) to use the Enumeration that you entered. To enter or change an Enumeration, enter the Ordinal in the field above the Add button. Next, enter the new value for the Tag and press Modify. Press OK to complete the procedure.

workbench-ServiceManager

Service Manager allows you to view services. It is available on the ServiceContainer.

workbench-SlotSheet

The Slot Sheet shows all the user visible slots of the Component. This includes Properties, Actions, and Topics.

The Slot Sheet is a table that shows the following for each slot:

- Slot
- #
- Name
- Display Name
- Definition
- flags
- Type

The Slot Sheet also optionally shows any Name Maps. The Slot Sheet includes the following menus:

- SlotSheet Main Menu
- SlotSheet Component Menu

SlotSheet Main Menu

The Workbench main menu functions are available. When the Slot Sheet is visible, the following menu functions are also available:

- Add Slot: Ctrl + A
- Rename Slot: Ctrl + R
- Config Flags
- Config Facets
- Reorder
- Add Name Map

SlotSheet Component Menu

If you right-click any Component in the Slot Sheet or the background, you can choose from the following:

- Add Slot: Ctrl + A
- Copy: Ctrl + C
- Delete
- Rename Slot: Ctrl + R
- Config Flags
- Config Facets
- Add Name Map

SlotSheet Toolbar

The Workbench toolbar contains navigation and editing buttons. When the Slot Sheet is visible, additional toolbar buttons include the following:

- Add Slot: Add Slot allows you to add a slot to the Component.
- Rename Slot: Rename Slot allows you to rename a slot in the Component
- Config Flags
- Reorder

Add Name Map

Add Name Map allows you to add a Name Map to the Component. You right-click on the property displayNames to delete or rename Name Map, and displayNames_xx to delete or rename Name Map (xx) where xx is the Language Code.

Config Facets

Configure Facets on the Component is available on Properties. Right-click on a property to view the Config Facets window.

workbench-StationSummary

StationSummary is the default view on a station. It holds primary components (for example, Config, Files, History) and shows specific configuration information about the station’s host platform, including the following:

- Station Name
- Host
- Host Model
- Host Id
- Niagara Version
- Java Version
- OS Version
- Locale
- Current Time

workbench-Synthetic Module File View

(AX-3.7 and later) The Synthetic Module File View is the default view on a synthetic module.

workbench-TextFileEditor

The TextFileEditor Plugin provides a powerful Color coded text editor. It supports Color coding of C, java and xml file types.

TextFileEditor Menus

The Workbench main menu functions are available.

TextFileEditor Toolbar

The Workbench toolbar contains navigation and editing buttons as described in “About the toolbar”.

File Types

The TextFileEditor Plugin provides a powerful Color coded text editor. It supports Color coding of C, java, properties, Python and xml file types. See the section, “Text Editor options” to change editor options including Color coding.

C Files

The TextFileEditor supports special Color coding for C files including:

- Preprocessor - #include
- Line Comment - / comment /
- Multiline Comment - /* comment */
- String literal - "string" and 'string'
- Number literal - '0' and 'F'
- Keyword - blue - if

CSS Files

The TextFileEditor supports special Color coding for CSS files including:

- Identifier - HTML element or CSS identifier
- Line Comment - / comment /
- Multiline Comment - /* comment */
- String literal - "string" and 'string'
- Number literal - '0' and 'F'
- Keyword - blue - if

HTML Files

The TextFileEditor supports special Color coding for HTML files including:

- Identifier - HTML element
- Multiline Comment - <!– Comments here –>
- String literal - "string" and 'string'
- Number literal - '0' and 'F'
- Keyword - blue - if

Java Files

The TextFileEditor supports special Color coding for Java files including:

- Bracket - ({ [
- Keyword - if
- Line Comment - / comment /
- Multiline Comment - /* comment */
- String literal - "string" and 'string'
- Number literal - '0' and 'F'

JavaScript Files

The TextFileEditor supports special Color coding for JavaScript files including:

- Bracket - ({ [
- Keyword - if
- Line Comment - / comment /
- Multiline Comment - /* comment */
- String literal - "string" and 'string'
- Number literal - '0' and 'F'

Properties Files

The TextFileEditor supports special Color coding for properties files including:

- Line Comment - #
- Bracket - =

Python Files

The TextFileEditor supports special Color coding for Python files including:

- Bracket - {}()[]
- Keyword - if
- Line Comment - #
- String literal - "string" and 'string'
- Number literal - '0' and 'F'

Xml Files

The TextFileEditor supports special Color coding for Xml files including:

- Multiline Comment - <!– comment –>>
- Bracket - < > < >
- String literal - "string" and 'string'

workbench-WbPxView

PxView is a dynamic view which may be added to Components as a property. PxViews store the view contents in a PxFile which is an XML file with a Px extension. The view itself is defined as a tree of bajaui: Widgets.

WbPxView Menus

The Workbench main menu functions are available. When the Px Viewer is visible.

PxViewer View Source Xml

You can view the source XML of a Px Page by selecting View Source Xml from the main menu.

workbench-WbServiceManagerView

Workbench Service Manager allows you to view services. It is available from the Tools menu.

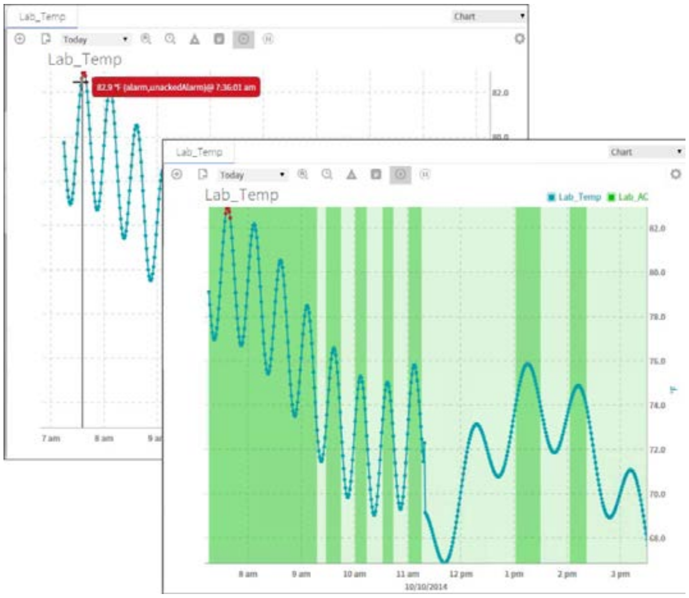
workbench-WebWidget

This is a bajaux, HTML5-based application that incorporates a view with interactive functionality which allows you to edit properties and invoke commands from the view. You can easily add data to a WebWidget, such as the WebChart or Dashboard, simply by dragging one or more components onto the widget. The widget renders in both Workbench and HTML5 Hx interfaces. The widget also integrates into the environment. For example, commands defined for a WebWidget render as added tool bar icons in Workbench, as well as in the HTML5 Hx profile in a web browser.

Examples of the bajaux WebWidget include the following:

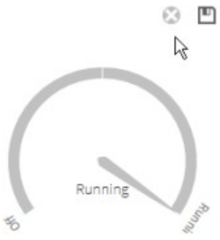
- The WebChart displays the Chart view which can display historical data and update with live data. Also, in the view you can easily add data and invoke numerous commands and settings to modify data presentation.

Figure 153: Chart WebWidget



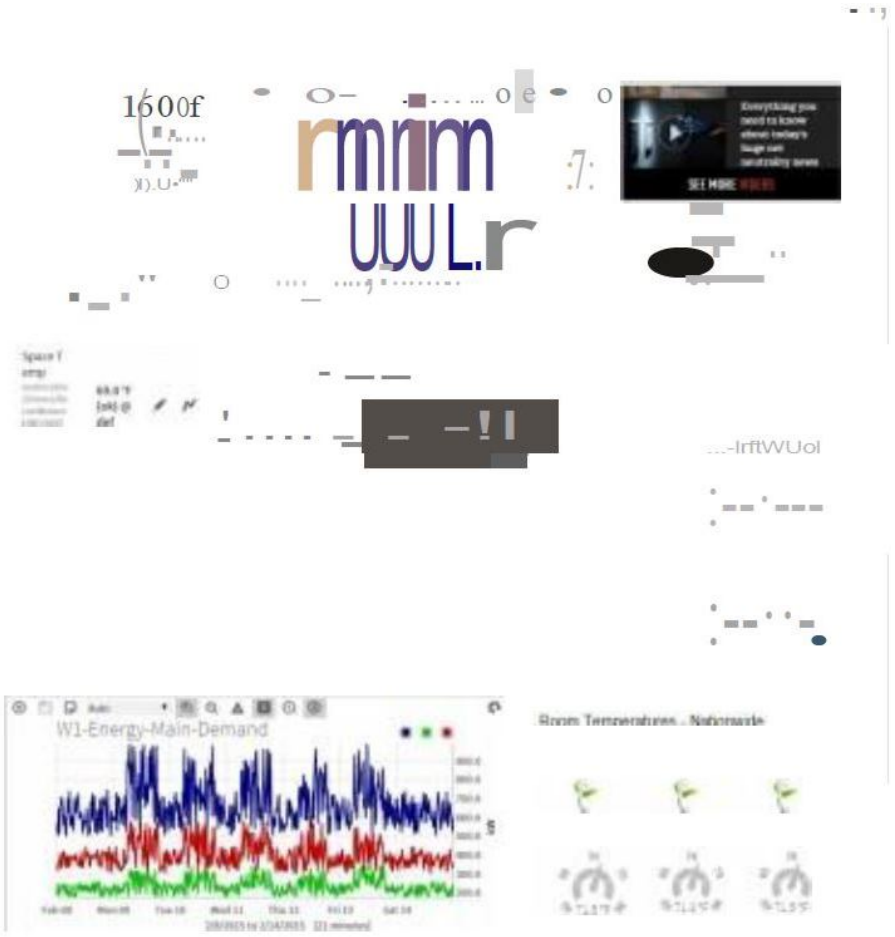
- The CircularGauge displays the graphical gauge view which updates with live data and provides contextual information for the current value. At any time you can dynamically switch the display to another component simply by dragging and dropping a different component onto this widget.

  Figure 154: CircularGauge WebWidget

- A Dashboard may be added to any PxPage and displayed in the PxViewer. Additional WebWidgets may be added to the Dashboard pane to customize the presentation of data. The dashboard is used to write dashboard-specific data to and from a station for a specific user.

  Figure 155: Dashboard WebWidget

APPENDIX

The following topics are covered in this appendix:

- About keyboard shortcuts
- Types of menu bar items
- Types of popup menu items
- Types of side bars
- Types of edit commands
- Types of toolbar icons
- Types of console commands

This appendix contains reference topics about keyboard shortcuts, menu bar items, popup menus, and toolbar icons.

- About keyboard shortcuts: Explains how to use the keyboard to enter menu bar and popup menu commands, without using the mouse.
- Types of menu bar items: Describes the different menu items available from the menu bar.
- Types of popup menu items: Describes the popup menus that appear in different views and contexts.
- Types of toolbar icons: Describes the types of icons that are in the toolbar.

About keyboard shortcuts

Workbench provides a number of keyboard shortcuts for common actions. These are performed by holding down the combination of keys listed. They include the following:

- F1 - Help On View
- F3 - Console
- F4 - Hide Console
- F5 - Find
- F6 - searchReplace;
- F7 - Goto File
- F8 - searchConsoleNext
- F9 - Save amp; Compile
- Alt + Left - Back
- Alt + Right - Forward
- Alt + Up - Up Level
- Alt + Home - Home
- Alt + Space - Recent Ords
- Ctrl + A - Add Slot
- Ctrl + C – Copy
- Ctrl + D – Duplicate
- Ctrl + F - Find Next
- Ctrl + G - Goto Line
- Ctrl + L - Open Ord

- Ctrl + N - New Window
- Ctrl + O - Open File...
- Ctrl + P - Print
- Ctrl + R - Rename Slot
- Ctrl + S - Save
- Ctrl + T - New Tab
- Ctrl + V - Paste
- Ctrl + W - Word Wrap
- Ctrl + X - Cut;
- Ctrl + Z - Undo
- Ctrl + F1 - Find Bajadoc
- Ctrl + F4 - Active Plugin
- Ctrl + F5 - Find Files
- Ctrl + F6 - Replace Files
- Ctrl + F9 - Compile
- Ctrl + PageUp - Next Tab
- Ctrl + PageDown - Previous Tab
- Shift + F8 - searchConsolePrev
- Ctrl + Shift + F - Find Prev
- Ctrl + Alt + Z - Redo

Types of menu bar items

This section describes the menu items that appear in the WorkbenchWorkbench menu bar:

- File menu
- Edit menu
- Search menu
- Bookmarks menu
- Tools menu
- Window menu
- Px Editor menu
- History Ext Manager menu
- Help menu

About the File menu

The File menu in the menu bar has the following options:

- Open Ord

You may open an ORD by selecting File > Open Open Ord from the main menu or from the toolbar open button.

The Ord Chooser also allows you to directly type the Ord of a location to go there. If no view is specified, the default view for the item will be presented. You may also paste an Ord instead of typing it, by pressing the paste shortcut Ctrl + V after you have copied a node or Ord into the clipboard.

- Open File

You may open a file by selecting > File→Open→Open File... from the main menu.

- Open Directory

You may open a Directory by selecting > File→Open→ Open Directory... from the main menu.

- Open Station (Fox)

You may open a Station by selecting > File→Open→ Open Station (Fox)… from the main menu. A popup window appears. Enter information in the following fields.

- Address: This is the address of the computer running the Station that you wish to access.
- User name: This is the user name given to you by your system administrator.
- Password: This is the password given to you by your system administrator.
- Remember these credentials: This will save this connection in your connection list.

Once you successfully connect to the Station, it will appear in the tree and your home page will be displayed.

If you will be away from the system, you should close the Station to prevent unauthorized access.

- Open Platform

You may open a Platform by selecting File→Open→ Open Platform... from the main menu. A popup window appears. Enter information in the following fields.

- Host: This is the address of the computer running the Platform that you wish to access.
- Port: This is the port used.
- Password: This is the password given to you by your system administrator.
- Remember these credentials: This will save this connection in your connection list.

Once you successfully connect to the Platform, it will appear in the tree and the available tools will be displayed.

- Find Stations

You may find Stations by selecting File→Open→ Find Stations.from the main menu. This command finds all the stations running on the network. It will search for 5 seconds, then display a Table of all the stations found. You may display additional columns about each station using the options button (on the right of the header). Double-click a Station to open a Fox connection to it.

- ⇦ Back

Back allows you to go to the previous view. The shortcut is Alt + Left.

- ⇨ Forward

Forward allows you to go to the next view. You must have used the Back command previously. The shortcut is Alt + Right.

- Up Level

Up Level allows you to go to the next level up. The shortcut is Alt + Up.

- Save

This saves the value of the Component.

- Save Bog

Save the Component changes to the Bog File file.

- Save All

Save all open views. This saves all tabs when browsing with multiple tabs.

- Recent Ords

Recent Ords allows you to see recent Ords. The shortcut is Alt + Space.

- Refresh

Refresh allows you to refresh the current view.

- Home

Home allows you to go to the home view. The shortcut is Alt + Home.

- Printing

Printing provides the capability to print the current view. When it appears dimmed, printing is not available.

- Export

Export provides the capability to Export the current view. When it appears dimmed, Export is not available.

- Logging off the System

You may exit the system by a number of means. You may Close the session, Close All sessions, Exit, or Close the current window.

- Close Session: You may logoff the Station at any time by selecting File→Close from the main menu. Once you successfully close a session, it will be removed from the tree. This could leave a user interface running without an open session.
- Disconnect: You may logoff the system at any time by right-clicking the connected system in the tree and select Disconnect. This allows you to close a session with a Station without removing it from the tree.
- Dismiss: You may logoff a connected system at any time by Right-clicking the connected system in the tree and select Dismiss. This allows you to close a session with a Station and remove it from the tree.
- Exiting the System: You may exit the system at any time by selecting File→Exit from the main menu. This differs from logoff in that it also causes the user interface to stop.
- Close the Current Window: You may close the current window at any time, if it is not the primary window, by any of the following methods:

  – selecting File→Close from the main menu.

  – pressing the box in the top left corner, and selecting Close from the menu.

  – pressing the close icon in the top right corner of the window

- New Window

New Window requests that a new window be created identical to the current one. This allows you to view multiple views concurrently.

- New Tab

New Tab requests that a new tab be created identical to the current one. This allows you to view multiple views in the same window. You can hold down the Ctrl key while double-clicking to hyperlink into a new tab. You can also Right-click to get a popup menu on tabs to close the tab, or close all other tabs.

- Close Tab

Close Tab requests that a tab be closed.

- Close Other Tabs

Close Other Tabs requests that all other tabs be closed.

- Next Tab

Next Tab selects the next tab. The shortcut is Ctrl + PageUp.

- Previous Tab

Previous Tab selects the previous tab. The shortcut is Ctrl + PageDown.

About the Edit menu

- Cut

This copies the current Selection to the clipboard and changes its Color to gray until a Paste is performed to delete it. If the Selection is a string, Cut makes a Copy of the current text selection and places it in the clipboard and removes it from the current string. It is accessed by selecting an item and doing the following:

– Right-clicking the item(s) and choosing Cut.

– Right-click in the tree and choosing Cut.

– In the Wire Sheet pressing the Cut toolbar icon.

– In the Wire Sheet selecting Edit→Cut from the main menu.

– Pressing the shortcut key Ctrl + X (hold down Ctrl and press X) when in a window other than the main browser window.

You can use Drag to cut and paste in one operation.

- Copy

This copies the current contents of the clipboard to the destination component as a set of new dynamic properties. If the selection is a string, Copy makes a copy of the current text selection and places it in the clipboard. To copy a component, do any of the following:

– Right-click a Component in a view or tree and choose Copy.

– Right-click the background in the Wire Sheet choosing Copy.

– In the Wire Sheet press the Copy toolbar button.

– In the Wire Sheet select Edit→Copy from the main menu

– Pressing the shortcut key Ctrl + C (hold down Ctrl and press C) when in a window other than the main browser window.

You can copy a selected group of Components in the Wire Sheet. You can use Drag to Copy also.

To copy a Component, you must copy it from the palette or an existing database. You can follow one the following steps:

– In the Palette Sidebar, expand a module (like control) and sub-directory (like Points), right-click on a component (like BooleanWritable) and select Copy.

– In the namespaces sidebar (tree), expand System, Modules, a module (like control), Files, module.palette, and sub-directory (like Points), right-click on a component (like BooleanWritable) and select Copy.

– Or, right-click on a Component that you want to Copy from a Wire Sheet, Property Sheet or the tree and select Copy.

- Paste

Paste copies the contents of the clipboard to the destination Component as a set of new dynamic properties. If the target is a string, Paste places a reference to the source that was cut or copied into clipboard.

It may be accessed by cutting or copying item(s) and doing the following:

– Right-clicking the item and choosing Paste.

– Right-click in the tree and choosing Paste.

– In the Wire Sheet press the Paste toolbar icon.

– In the Wire Sheet select Edit→Paste from the main menu

– Pressing the shortcut key Ctrl + V (hold down Ctrl and press V) when in a window other than the main browser window.

You can use Drag to Cut and Paste in one operation.

You can add acComponent to a running Station or an offline Bog File file. You can do this using any of the following:

– In the Wire Sheet, right-click on the background and select Paste to add the new Component. The new Component is created and is selected.

– In the tree, Right-click on a Container and select Paste to add the new Component inside the Container.

If you select Help→Guide On Target with a Component selected, you will get the Guide for that Component. If you select Help→Bajadoc On Target with a Component selected, you will get the bajadoc for that Component. The BooleanWritable bajadoc is at module://control/doc/javax/baja/control/BBooleanWritable.bajadoc.

- Paste Special

Paste Special copies the contents of the clipboard to the destination Component when it is a Special Transferable.

- Duplicate

Duplicate copies the current Selection and places a duplicate in the same Container. This function may be accessed from the menu under Edit (shortcut Ctrl + D) or from the toolbar.

- Delete

Delete removes the current Selection from its parent Container. It may be accessed by selecting an item and doing the following:

– Right-clicking the item(s) and choosing Delete.

– Right-clicking in the tree and choosing Delete.

– In the Wire Sheet pressing the Delete toolbar button.

– In the Wire Sheet selecting Edit→Delete from the main menu.

- Undo

This reverses the last Action as if it had not been performed. It is only available for certain actions such as:

- Paste Component
- Cut Component
- link
- Delete Links

- Redo

This restores an Action after Undo has removed it. It is only available after a successful Undo.

About the Search menu

The Search menu in the menu bar has the following options:

- Find

Find allows you to search in the file for the selected string. You can Match Case or Match Word. The shortcut is F5.

- Find Next

Find Next allows you to find the next occurrence of the selected string. The shortcut is Ctrl + F (hold down Ctrl and press F).

- Find Prev

Find Prev allows you to find the previous occurrence of the selected string. The shortcut is Ctrl + Shift + F (hold down Ctrl and Shift and press F).

- Replace

Replace allows you to replace the next occurrence in the file. The shortcut is Ctrl + R (hold down Ctrl and press R).

- Goto Line

Goto Line allows you to go to a line number in the file. The shortcut is Ctrl + G (hold down Ctrl and press G).

- Goto File

Goto File allows you to go to a file. The shortcut is Ctrl + F3 (hold down Ctrl and press F3).

- Find Files

Find Files allows you to find the all occurrences of a string in files. You can Match Case or Match Word. You can choose Files to Find. You can select a Folder. You can choose whether to Search subfolders.

- Replace Files

Replace Files allows you to replace the all occurrences in the files.

- Console Prev

Console Prev allows you to go to the previous console error. The shortcut is F7.

- Console Next

Console Next allows you to go to the next console error. The shortcut is F8.

About the Bookmarks menu

The Bookmarks menu in the menu bar has the following options:

- AddTo Bookmarks

You may add a Bookmark by selecting Bookmarks→Add to Bookmarks from the main menu.

- Manage Bookmarks

You may manage Bookmarks by selecting Bookmarks→ Manage Bookmarks from the main menu.

- Bookmarks Go Into

You may select Bookmarks by selecting Bookmarks→ Go Into from the main menu.

- Bookmarks File

You may select Bookmarks by selecting Bookmarks→ File from the main menu.

About the Tools menu

The Tools menu in the menu bar has the following options:

- Viewing and Changing the Options

The Options allow you to customize the framework for the way you use it. It can be selected from the main Menu by selecting Tools→ Options. It includes the following:

– General

– Lexicon

– Text Editor

– Wire sheet

- ColorChooser

The ColorChooser allows you to choose Colors.

- New Module Wizard

The New Module Wizard allows you to build a new Module. It can be selected from the main Menu by selecting Tools→ New Module.

- New Station Wizard

The New Station Wizard allows you to build a new station Database. It can be selected from the main menu by selecting Tools→ New Station.

You must enter a Station Name. You can also choose whether this is a JACE station or a Supervisor station.

Press ⇨ Next to proceed. Next you should enter an Admin Password.

You can also change the Fox Port and HTTP Port. Press ✓ Finish to proceed. You are then presented a view of your newly created Station at:

local:|file:!stations/stationname/config.bog|bog:|slot:/.

- Manage Credentials

Manage Credentials is available in the main Tools menu by selecting Manage Credentials. You can reset or remove selected Credentials or Remove All Credentials. You can also open selected Credentials.

- Request License

Request License is available in the main Tools menu by selecting Request License. You can request a license by submitting the form.

About the Window menu

The Window menu in the menu bar has the following options:

- Side Bars

– ✓ Show Side Bar: You can choose whether to have Side Bars by selecting Window→ Side Bars→ Show Side Bar from the main menu.

– Bookmarks: You can choose to show the Bookmarks Side Bar by selecting Window→ Side Bars→Bookmarks from the main menu.

– ? Help Sidebar: The typical configuration provides a Help Sidebar frame in the left side of the main window. If it is not open, you can choose to display the Help Side Bar by selecting Window→ Side Bars→ Help from the main menu. Initially, the Help Sidebar is empty until you press Load Help.

– ? Load Help will reappear any time any Module's timestamp is changed or a new Module is added or removed. Load Help searches through all the available Modules to create the help Directory. The Help Sidebar provides a view of the available Help.

– Jobs Sidebar: The Jobs SideBar shows all the current jobs in all the stations with which you have a connection. Controls are provided to allow you to view a job log or dismiss a completed job.

– Nav: You can choose to show the Nav side bar by selecting Window→ Side Bars→ Nav from the main menu.

– Palette: You can choose to show the Palette Side Bar by selecting Window→ Side Bars→ Palette from the main menu.

- PathBar Uses NavFile

You can toggle this option ON or OFF by selecting Window→ Side Bars→ PathBar Uses NavFile. When set to ON, the PathBar (located at the top of theWorkbench main window) displays your current path, as defined by the NavFile (logical path). When set to OFF, (not selected) the PathBar displays your absolute path regardless of whether it is mapped to the NavFile or not. You must refresh the view after changing this setting to see the PathBar change.

- Active Plugin

The Active Plugin function gives focus to the current view. From the main menu you can select Window→ Active Plugin or use the shortcut Ctrl + F4. It is very useful to use F3/Ctrl + F4 to toggle between the Console and the Text File Editor.

- Hide Console

You can hide the console by selecting Window→ Hide Console from the main menu or using the shortcut Ctrl + F2.

- Console

The Console provides the capability to issue console commands directly. From the main menu you can select Window and Console (F3) or Hide Console (F4) to determine if the console is visible.

About the Px Editor menu

The Px Editor menu appears in the menu bar when Px Editor is the active view. The Px Editor has the following context-sensitive options:

Item	Description
Toggle View/Edit Mode	This command toggles the active view between Px Editor (for editing) and Px Viewer (view only). If there are unsaved changes in your Px file, you are prompted to save before switching from Px Editor to Px Viewer.
View Source Xml	Selecting this command displays the Px source file in a separate read-only window.
Go to Source Xml	Selecting this opens the Px source (xml) file directly in the text file editor. Files can be edited and saved using the editor.
Grid	This command toggles the grid display on and off.
Snap	This command toggles the snap–to–grid feature on and off.
Show Hatch	This command toggles the hatching pattern visibility on and off. When hatching is on, dim angular lines (hatching pattern) displays on objects to make them more visibly distinct.
Zoom In	The Px Editor display zooms-in on the canvas pane displaying less of the page at an enlarged size.
Zoom Out	The Px Editor display zooms-out on the canvas pane displaying more of the page at a reduced size.
Reset Zoom	Resets the canvas pane magnification to x1.0 (100%), displaying the Px page in actual size.
Set Target Media	This command is available when you are editing a Px file directly in the Px Editor, not when you are editing the Px file as a view of a component. When selected, this command displays the Set Target Media dialog box to allow you to choose your expected media viewer: • Workbench:WbPx Media • hx: HxPxMedia • report: ReportPxMedia • mobile: MobilePx Media

About the History Ext Manager menu

The History Ext Manager menu appears in the menu bar when History Extension Manager is the active view. The History Ext Manager menu has the following options:

- ▸ Enable Collection

Select this menu item to enable (start the collection process) for the selected entries.

- ■ Disable Collection

Select this menu item to disable (stop the collection process) for the selected entries.

- Rename History

Select this menu item to rename the selected history. This menu item displays the Set History Name dialog box.

- Edit System Tags

Select this menu item to open the Set System Tags For Selected History Extensions dialog box. Use this dialog box to edit system tags associated with a single history extension or perform batch edits when you have more than one history extension selected.

About the Help menu

The Help menu in the menu bar has the following options:

- ? Help Contents Index

The help contents provide a common point of access to all system documentation. It is accessed by selecting Help→ Contents Index from the menu or pressing the ? Help button on the toolbar to see the Help Index.

- Help On View

This provides help for the current Plugin. It is accessed by selecting Help→ On View from the menu with the Plugin in use.

- ○ Help Guide On Target

This provides context sensitive help for Components. It is accessed by selecting Help→ Guide On Tar-get from the menu when the current view is a view of a Component. It is also available by right-clicking a Component and choosing Views→ Guide Help.

- ○ Help Bajadoc On Target

This provides context sensitive Bajadoc help for Components. It is accessed by selecting Help→Bajadoc.

On Target from the menu when the current view is a view of a Component. It is also available by Right clicking a Component and choosing Views→ Bajadoc Help.

- Help Find Bajadoc

This is accessed by selecting Help→ Find Bajadoc from the menu. It searches for the requested bajadoc.

- Help About

This is accessed by selecting Help→ About from the menu. It provides the software release and license information.

Types of popup menu items

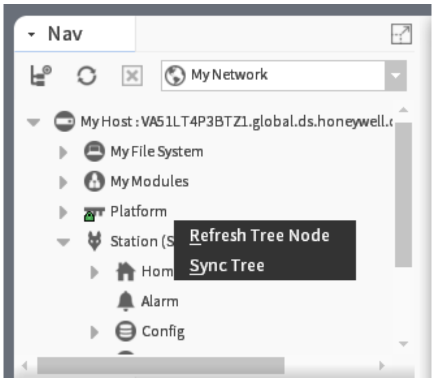
Workbench provides view–specific or context–specific commands for editing components in many of the views. Following is a list of the standard Workbench popup (right–click) menus.

- Nav side bar
- Wire sheet
- Property sheet
- Px Editor
- History extension manager
- Todo list
- Point Manager

About the Nav side bar popup menu items

The Nav sidebar provides a tree-type hierarchical view of the system. The Nav side bar menu is the popup menu that displays command options when you right-click on a component item in the Nav tree.

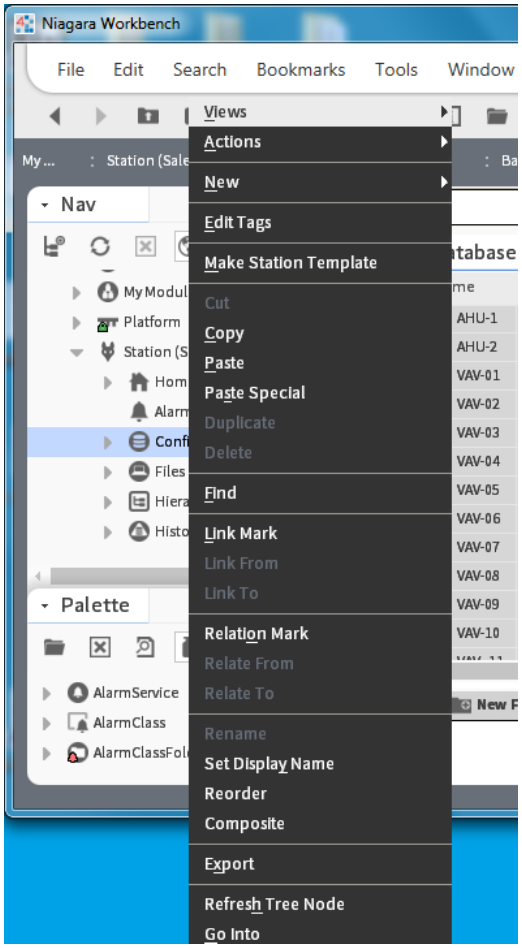
Figure 156: Nav side bar popup menu with nothing selected



This popup menu has these options:

Item	Description
Refresh Tree Node	Refreshes the Nav tree.
Sync Tree	

Figure 157: Nav side bar popup menu with component selected



The popup menu has the following options:

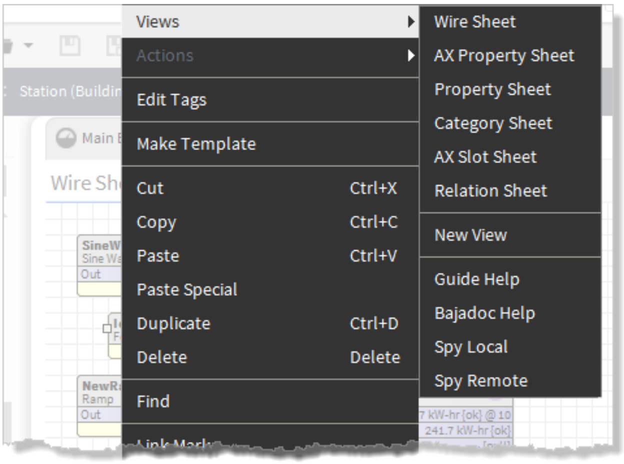
Item	Description
Views	Goes to any of the views of the selected component.
Actions	Perform any available actions on the component.
New	Provides additional options for creating new objects.
Edit Tags	Opens the Edit Tags window, permitting you to add or remove tags on a component.
Make Template	Creates a template of the selected root component, collects all associated Px and graphic files, and invokes the Template view, allowing you to configure the template for deployment.
Cut	Copies to clipboard and after pasting the copy, deletes the cut item.
Copy	Copies to clipboard, copied item remains.
Paste	Pastes the copied item from the clipboard.
Paste Special	
Duplicate	Makes a copy in the same location as the original item.
Delete	Removes the item.
Find	
Link Mark	Sets a selected component to your popup menu, making it temporarily available for “Linking From” or “Linking To” other points.
Link From	Allows you to link to a selected component from another component that has been marked, using Link Mark from the popup menu.
Link To	Allows you to link from a selected component to another component that has been marked, using Link Mark from the popup menu.
Relation Mark	Sets a selected component to your popup menu, making it temporarily available for “Relating From” or “Relating To” other points.
Relate From	Allows you to add a relation from selected component to another that has been marked, using Link Mark from the popup menu.
Relate To	Allows you to add a relation to a selected component from another that has been marked, using Link Mark from the popup menu.
Rename	Allows you to rename the selected component’s actual slot name (as it appears in the Ord). This menu item displays the Rename window. You can only rename one component at a time.
Set Display Name	Allows you to set a display name for the selected component (as it appears in a Nav tree, and in the Wire Sheet, and Property Sheet views).

Item	Description
Reorder	Opens the Reorder Points window, which provides the following commands for reordering points within the selected parent component. • Move Up • Move Down • Sort by Name • Sort by Type • Reset
Composite	This menu item opens the Composite Editor window.
Export	Exports a selected component to the oBix .xml format.
Refresh	This menu item updates the display of the currently active view.
Go Into	Go Into allows you to re-root the nav tree at any arbitrary node. Right-click the node and select the Go Into command. This will make that node the new root of the tree. All the nodes you have "gone into" are persistently saved as a special type of Bookmark. Use the pulldown to switch between them. This feature is quite handy when working with multiple stations or deep file systems and databases.
Pin Slots	Opens the Pin Slots window. Clicking to “Pin” a slot makes that slot visible in the Wire Sheet view. Clicking to “Unpin” a pinned slot has the opposite effect.
More...	Indicates the presence of additional menu items. Click More to display those items.

About the Wire Sheet popup menu items

Most of the Wire Sheet menu commands are described in “About the nav side bar popup menu items”.

Figure 158: Wire sheet popup menu



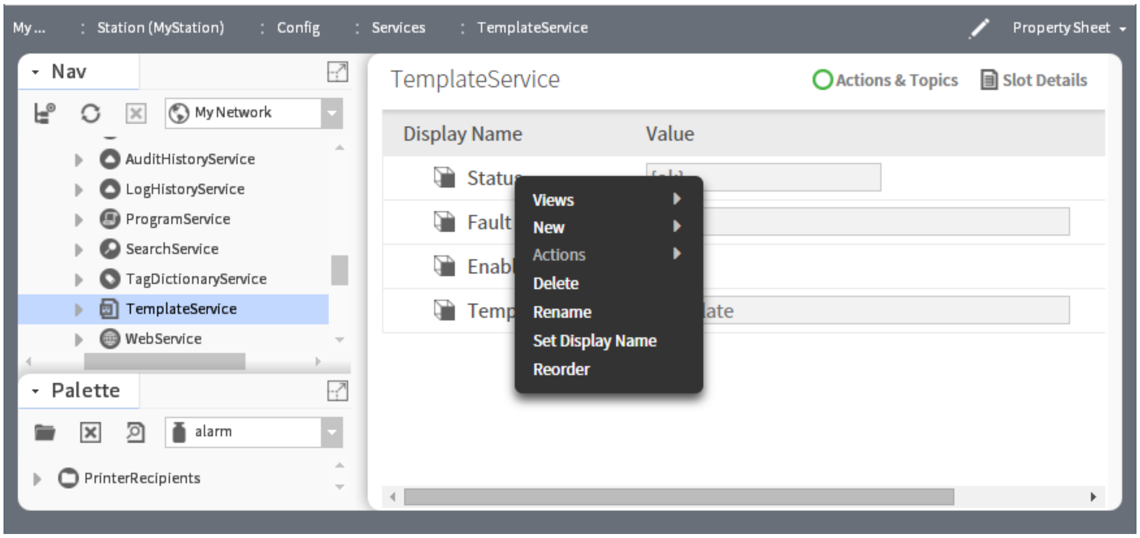
Following are wire sheet specific popup menu options:

Item	Description
Arrange	Provides options for aligning components on the wire sheet to make them easier to view. • Arrange All: Redistributes the layout of all components on the wire sheet. • Arrange Selection: Redistributes the layout of all selected components on the wire sheet.
Select all	Selects all components on the active wire sheet.

About the property sheet popup menu items

Most of the property sheet menu commands are described in “About the nav side bar popup menu items”.

Figure 159: Property sheet popup menu

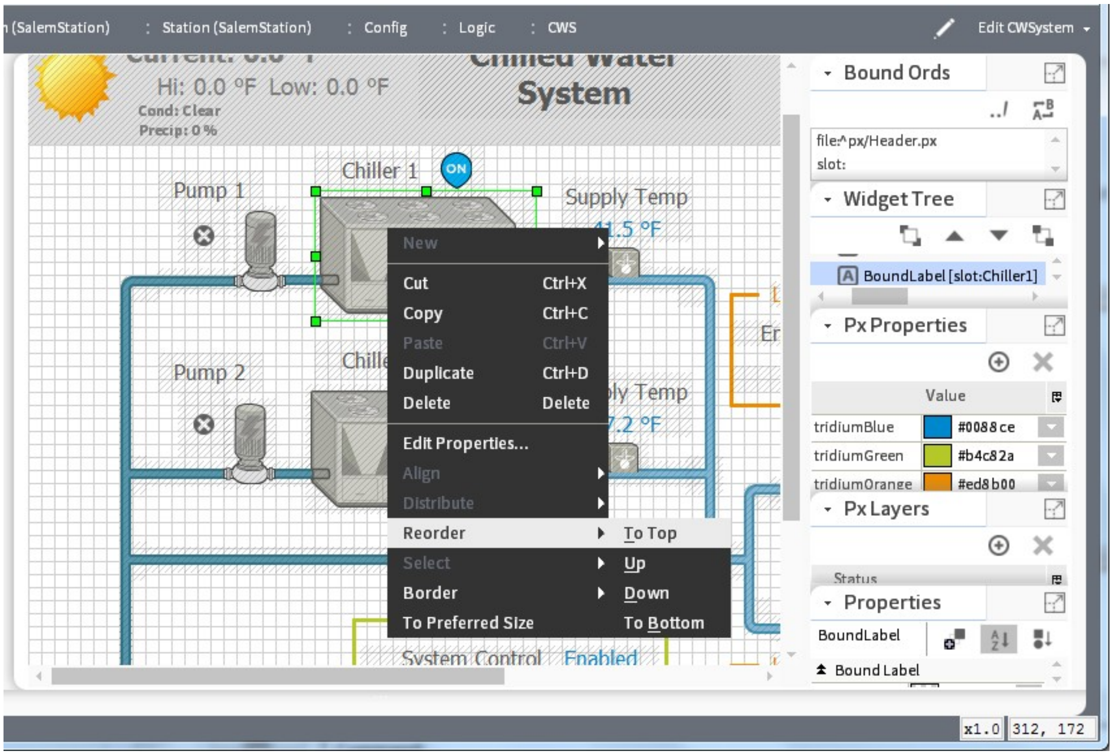


- Config flags: Opens the Config dialog box which you can use to add configuration flags on individual slots.

About the Px Editor pop-up menu items

This popup menu appears when you right-click on an object in the Px Editor view. The menu commands are context-sensitive and are dimmed or available, depending on the type of object that you select.

Figure 160: Px Editor popup menu



The following menu commands are on the Px Editor pop-up menu:

Item	Description
New	Create a new item of standard types.
Cut	Copies to clipboard and after pasting the copy, deletes the cut item.
Copy	Copies to clipboard, copied item remains
Paste	Pastes the copied item from the clipboard
Duplicate	Makes a copy in the same location as the original item
Delete	Removes the item
Edit Properties...	Refer to “About the Properties side bar”.

Item	Description
Align	• Left: This command is available when you select two or more objects in the Px Editor. Choose the Left command to align the left edges of two or more objects along a vertical line. • Right: This command is available when you select two or more objects in the Px Editor. Choose the Right command to align the right edges of two or more objects along a vertical line. • Top: This command is available when you select two or more objects in the Px Editor. Choose the Top command to align the top edges of two or more objects along a horizontal line. • Bottom: This command is available when you select two or more objects in the Px Editor. Choose the Bottom command to align the bottom edges of two or more objects along a horizontal line.
Reorder	• To Top: This command is available when you select one or more objects in the Px Editor. Choose the To Top command to move the selected object to the top position (inside its parent object) in the Widget Tree. This command will place the object in front (with respect to the view pane, or z-axis) of all other objects in the parent object, but will not move the object out of its parent object. • Up: This command is available when you select one or more objects in the Px Editor. Choose the Up command to move the selected object one position higher in the Widget Tree and forward one position in the view pane. This command will not move the object out of its parent object. • Down: This command is available when you select one or more objects in the Px Editor. Choose the Down command to move the selected object one position lower in the Widget Tree and back one position in the view pane. This command will not move the object out of its parent object. • To Bottom: This command is available when you select one or more objects in the Px Editor. Choose the To Bottom command to move the selected object to the bottom position (inside its parent object) in the Widget Tree. This command will place the object behind (with respect to the view pane, or z-axis) all other objects in the parent object, but will not move the object out of its parent object.
Border	• Add Border: This command is available when you select one or more objects in the Px Editor. Choose the Add Border command to wrap selected object(s) with a border pane. Each selection is wrapped in a separate border pane. • Remove Border: This command is available when you select one or more border panes in the Px Editor. Choose the Remove Border command to delete selected border pane(s).

About the history extension manager popup menu items

The history extension manager popup menu has the following items:

- Views

This menu item provides a submenu that lists all the available views of the history extension manager.

- Actions > Update History Id

This menu item provides a way to refresh the History Id after a rename. It applies the formatting property (as designated by the enclosing % signs) of the Name Format field to the Id of the History Config Id. For example, if the History Name property under a history extension is set to %parent.name%, then the History Config Id is initially named based on this parent display name. However, if you rename the parent component, the History Config Id property does not automatically or immediately change. The Update History Id action invokes a renaming of the History Config Id based on the formatting property, so if the parent component (in this example case) is changed, the the Update History Id action changes the Id property and, if different from the history name, it results in a change in the history name as well.

- Go To Point

This menu item displays the property sheet view of the point associated with the selected entry.

- Go To History

This menu item displays the default view of the history associated with the selected entry.

- Enable Collection

Select this menu item to enable (start the collection process) for the selected entries.

- Disable Collection

Select this menu item to disable (stop the collection process) for the selected entries.

- Rename History

Select this menu item to rename the selected history. This menu item displays the Set History Name dialog box. You can only rename one history at a time.

- Edit System Tags

Select this menu item to open the Set System Tags For Selected History Extensions dialog box. Use this dialog box to edit system tags associated with a single history extension or perform batch edits when you have more than one history extension selected.

About the Todo list popup menu items

The Todo list popup menu has the following items:

- Add

This menu item opens the Add dialog box, when clicked. Use this menu item to add a new item to your Todo checklist and assign a Summary and Group to the item. This menu item is available even if no items are selected.

- Mark Complete

This menu item, when clicked, dims and lines-through the selected item(s) in the Todo list so that the item(s) appears to be “crossed off” the list. If an item is already marked complete and this menu item is selected, the item will be restored to its “unmarked” state. With no items selected, this menu item is dimmed (unavailable).

- Edit

Displays the Edit dialog box, when clicked, allowing you to change the Summary and the Group fields associated with the item.

- Move to Top

This moves the selected item to the top of the list.

- Move Up

This moves the selected item up in the list, one increment per click.

- Move Down

This moves the selected item down in the list, one increment per click.

- Move to Bottom

This moves the selected item to the bottom of the list.

- Remove

This displays a “Remove selected items?” prompt, when clicked; deletes selected items when the prompt is affirmed.

About the point manager popup menu items

The point manager popup menu has the following items:

- Views

This menu item provides a submenu that lists all the available views of the point manager. These same views are available from the view selector menu when the point extension manager is the active view.

- Actions

This menu item allows you to perform any available actions on the component. For example, write to a “writable” point.

- New

This menu item allows you to add a new component to the points manager. Valid options are presented in the submenu.

- Cut
- Copy
- Paste
- Duplicate
- Delete
- Find

This menu item displays the Component Finder dialog box.

- Link Mark

This menu item sets a selected point to your popup menu, making it temporarily available for “Linking From” or “Linking To” other points.

- Link From

This menu item allows you to link a selected point to another point that has been marked, using Link Mark from the popup menu.

- Link To

This menu item allows you to link a selected point to another point that has been marked, using Link Mark from the popup menu.

- Rename

Select this menu item to rename the selected point. This menu item displays the Set Point Name dialog box. You can only rename one point at a time.

- Reorder

This menu item displays the Reorder Points dialog box, which provides the following commands for reordering points within the selected parent component.

- Move Up
- Move Down
- Sort by Name
- Sort by Type
- Reset

- Composite

This menu item displays the Composite Editor dialog box.

- New Folder

This menu item allows you to add and name a new points folder.

- New

This menu item allows you to add and name a new point.

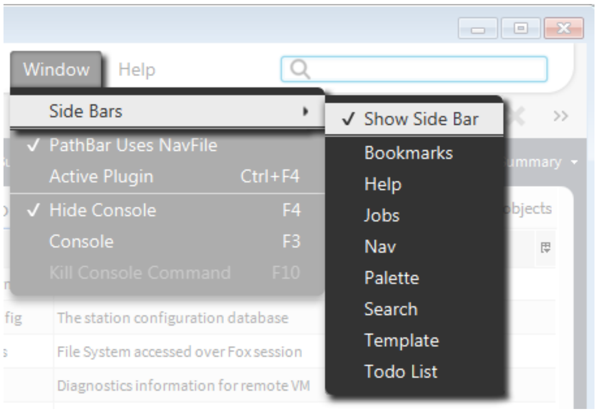
- Edit

This menu item displays the Point Editor dialog box.

Types of side bars

The Workbench interface may be customized by adding unique side bars that are designed to fit particular applications.

Figure 161: Default Side Bars menu



The following side bars may be displayed in the side bar pane by default:

- Bookmarks side bar

This displays a list of your bookmarks.

- Help side bar

This provides a tree view of available help documentation.

- Jobs side bar

The Jobs side bar lists all current jobs in all of the stations with which you have a connection. The current status of each job is shown.

- Nav side bar

This provides a tree view of the system.

- Palette side bar

This provides a tree view of components that are available in specific palettes.

- Search side bar

Allows you to enter search queries and access cached results from your previous queries. Also, you can edit Search Settings (requires super user permissions).

- Template side bar

Provides access to all template files located in your Workbench User Home ~templates folder, as well as to any templates stored in modules located in the SysHome !modules folder.

- Todo List side bar

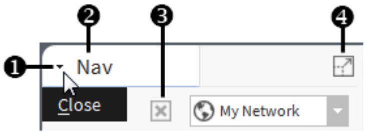
Provides a customizable list of tasks or notes.

About the side bar title bar

All side bars have a title bar across the top. You can click and drag on the title bar to vertically resize the side bar or you can click on a minimized side bar to restore it to the previous size.

In addition, all side bars have the following controls:

Figure 162: Side bar title bar



❶ Close dropdown — drop down control used to close the side bar

❷ Pane title— text that identifies the side bar type

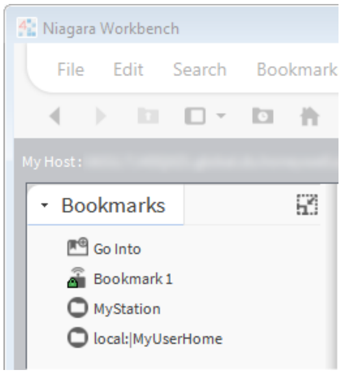
❸ Close — alternative to the Close dropdown, closes the side bar

❹ Expand/Restore — clicking on the expand/restore icon causes the side bar to expand, filling the side bar pane and collapsing other open side bars. Click again to restore the normal side bar display.

About the Bookmarks side bar

When you open the Bookmarks side bar, it appears in the side bar pane, as shown in the following figure.

Figure 163: Bookmarks side bar



From the bookmark side bar, you can double click on bookmark nodes or use popup menus to perform all operations that are available from the side bar (for example, go directly to a bookmarked location, manage bookmarks, edit bookmarks, and more). The quick access provided here is very helpful for changing screens without having to go through multiple selections using other menus or submenus.

About the Help side bar

When you open the Help side bar, it appears in the side bar pane.

The help side bar has three tabs that you may select by clicking on the tab. The three help tabs are listed and briefly described, as follows:

- Table of Contents tab

This contains a tree view of help topics, listed in alphabetical order by topic.

- API tab

This contains a tree view of help topics, listed in alphabetical order by module.

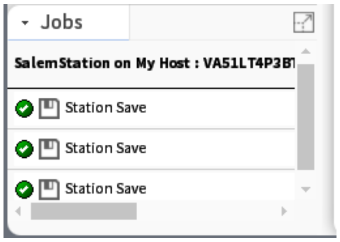
- Search tab

This contains a Find: text entry field and Search button.

About Jobs side bar

When you open the Jobs side bar, it appears in the side bar pane. The Jobs side bar contains a list of jobs that have been performed or that are currently being performed.

Figure 164: Jobs side bar



Jobs

SalemStation on My Host : VA51LT4P3B1

✓ Station Save

✓ Station Save

✓ Station Save

The following icons on the Jobs side bar indicate job status:

- Running

Indicates that the job is currently running.

- Success

Indicates that the job has completed without error.

- Failed

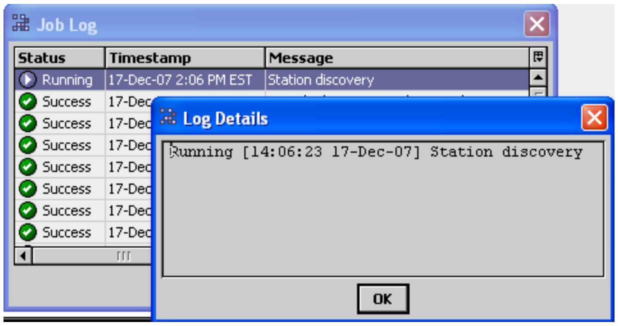
Indicates that the job did not complete.

- Unknown

Indicates that the job status is not available.

From the Jobs side bar, you can click on the arrow icon >> to open the Job Log dialog box. The Job Log dialog box displays a listing of the actions performed as part of the job. Each entry in this log contains a detailed description that you can view by double-clicking on the entry to open the Log Details dialog box.

Figure 165: Log Details dialog box



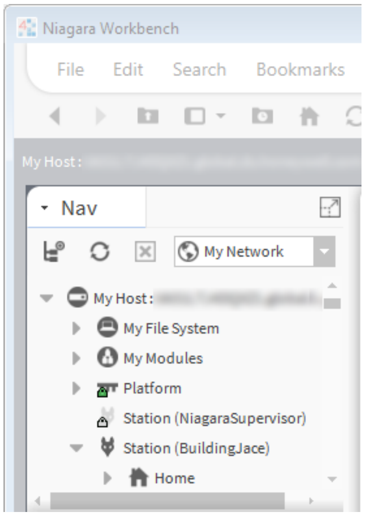
Status	Timestamp	Message
Running	17-Dec-07 2:06 PM EST	Station discovery
Success	17-Dec	
Success	17-Dec	
Success	17-Dec	
Success	17-Dec	
Success	17-Dec	
Success	17-Dec	
Success	17-Dec	

```
Running [14:06:23 17-Dec-07] Station discovery
```

About the Nav side bar

The Nav side bar contains the tree view that provides a hierarchical view of the whole system. When you open the Nav side bar, it appears in the side bar pane.

Figure 166: Nav side bar



Items are displayed in the tree with a symbol based on type. If the item is a file, the symbol reflects the file type. Refer to “Types of nodes in the Nav tree side bar”for more details about file types. Refer to “About the Nav side bar popup menu items” for more details about the Nav side bar popup menu.

At the highest level, the Nav side bar tree may include the following (when working from a localhost, as shown):

- My Host (local system)
- My File System
- My Modules
- Platform
- Stations (connected or disconnected)

From the Nav side bar, you can double click on nodes in the Nav tree or use popup menus to perform all operations that are available from the Nav side bar (for example, connect or disconnect to a station, refresh a tree node, and more). The expandable tree provided here is very useful for performing actions on nodes and for navigating through various screens and views inWorkbench. Items are displayed in the tree with an icon that represents the associated function or file type.

Types of nodes in the Nav tree side bar

The Nav tree side bar may include several different types of nodes and child nodes.

- My Host node

Represents a physical computer (hardware) that the rest of the nodes (subnodes) reside on.

- My File System node

Represents the top level of a tree view of the host file system. File system subnodes represent drives and locations on the host system. It is important to understand that the file system provides access to files that are outside of the station database.

- My Modules node

When expanded, displays a tree view of available modules, listed in alphabetical order by module.

- Platform node

When expanded, displays a hierarchical view of the Niagara host platform. You can double-click on the platform node and sub-nodes, or use a right-click shortcut menu to perform all operations that are available on or under this node (connecting, disconnecting, refreshing, and more).

- Station node

Represents a station (connected or disconnected). When expanded, the station node displays the station contents in a hierarchical tree. You can double-click on the station node and sub-nodes, or use a right click shortcut menu to perform all operations that are available on or under this node (connecting, disconnecting, selecting views, and more).

- Home
- Alarm
- Config node

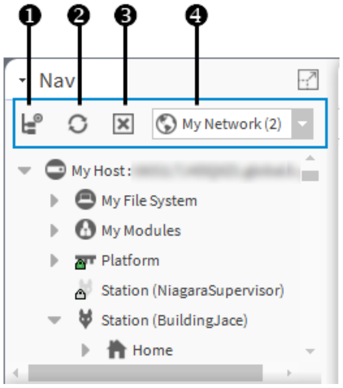
When expanded, displays a tree view of the station contents or “configuration”. The config node usually contains one or more of the following types of nodes:

- Services: Component for storing services, such as alarm service, history service, tagdictionary service, and more.
- Drivers: Provides a place to store driver modules (such as the NiagaraNetwork, BACnet drivers, Modbus, and more). When expanded, displays a tree view of loaded driver modules.
- Apps
- Schedules
- Control or Logic: Control points may be displayed directly in the root of the Config node.

About the Nav tree side bar toolbar

In addition to the standard side bar title bar the Nav tree side bar has a toolbar, located just below the title bar.

Figure 167: Nav side bar tool bar



❶ New tree button: Creates a new tree in the nav side bar. When you have more than one tree node, you can select one to activate from the dropdown tree selector.

❷ Sync tree button: Synchronizes the tree node display with the currently selected view.

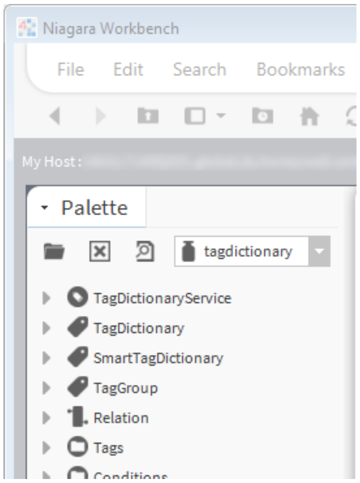
❸ Close tree button: Closes the currently displayed side bar.

❹ Drop-down tree selector: When multiple Nav side bars are open, this selector allows you to choose which one to display.

About the Palette side bar

When you open the Palette side bar, it appears on the left side of the Workbench in the side bar pane. The Palette side bar provides a place to open and view sets of modules or custom palettes that you build for yourself.

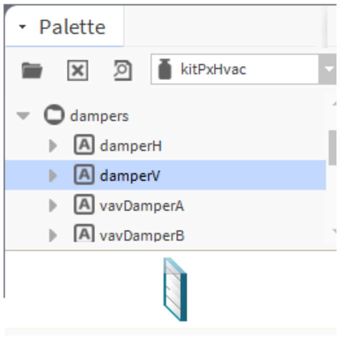
Figure 168: Palette side bar with preview pane



From the Palette side bar, you can open multiple palettes, close palettes and view modules within palettes. You may also double-click or use popup menus to perform all operations that are available from the Palette side bar (for example, copy modules, select a module view, refresh the tree node, and more). The expandable tree provided in the palette allows you to perform actions on nodes within the palette and to navigate through the palette sub-directories. Items are displayed in the tree with an icon that represents an associated function or file type.

The palette side bar also has a component preview pane (shown below) that displays an image (when available) of the selected component.

Figure 169: Palette preview pane

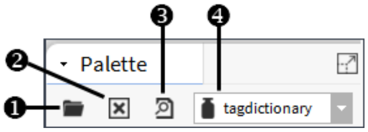


Palette previews display in the palette when components have images configured either as the default image property or as the image assigned to the comPreviewWidget property. If no preview is associated with a component, you can add a compPreviewWidget property to a widget to display an image in the preview pane of the palette side bar.

About the Palette side bar toolbar

In addition to the standard side bar title bar the Palette side bar has a toolbar, located just below the title bar.

Figure 170: Palette side bar tool bar



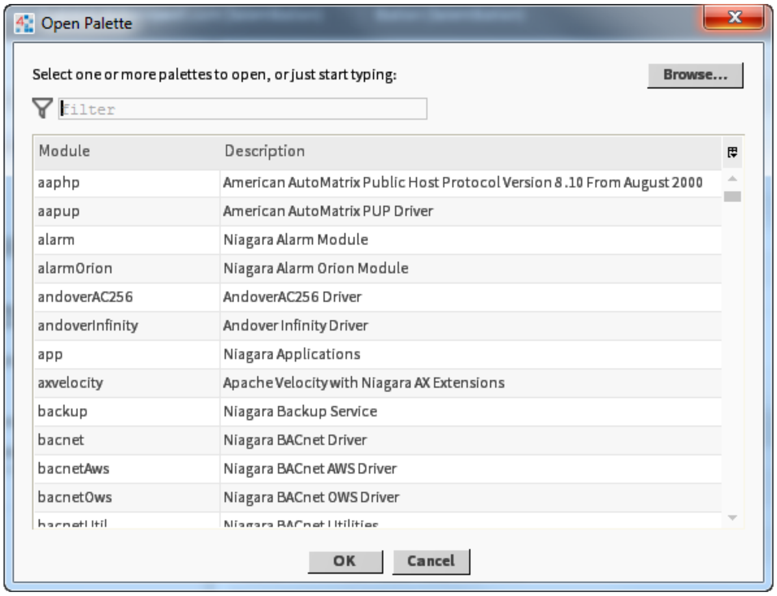
The palette toolbar includes the following:

❶	Open palette button — opens the Open Palette window.
❷	Close palette button — closes the currently displayed palette.
❸	Preview button — toggles the preview pane on and off. Previews are available on some components.
❹	Dropdown palette selector — when palettes are open in the palette side bar, this selector allows you to choose which palette to display.

About the Open Palette window

The Open Palette window displays a tabular list of available palettes. If there are palettes located in locations other than the My Modules directory, you can use the browse button to find them.

Figure 171: Open Palette window



Select one or more palettes to open, or just start typing:

Module	Description
aaphp	American AutoMatrix Public Host Protocol Version 8.10 From August 2000
aapup	American AutoMatrix PUP Driver
alarm	Niagara Alarm Module
alarmOrion	Niagara Alarm Orion Module
andoverAC256	AndoverAC256 Driver
andoverInfinity	Andover Infinity Driver
app	Niagara Applications
axvelocity	Apache Velocity with Niagara AX Extensions
backup	Niagara Backup Service
bacnet	Niagara BACnet Driver
bacnetAws	Niagara BACnet AWS Driver
bacnetOws	Niagara BACnet OWS Driver
bacnetUtil	Niagara BACnet Utilities

The Open Palette window is shown above and has the following features:

- Filter field

This is a text field that allows you to type the beginning letters of the desired palette name to filter out palettes from the view. For example, typing in the letters “mo” removes palettes that do not begin with the letters “mo”. You may use the * (asterisk) character as a “wild card” entry in this field. All palettes are listed in the table if no text is entered in this field.

- Browse button

This button opens the File Chooser window to allow you to select palettes that are located in alternate locations.

- Table of palettes

The table of palettes has the following columns

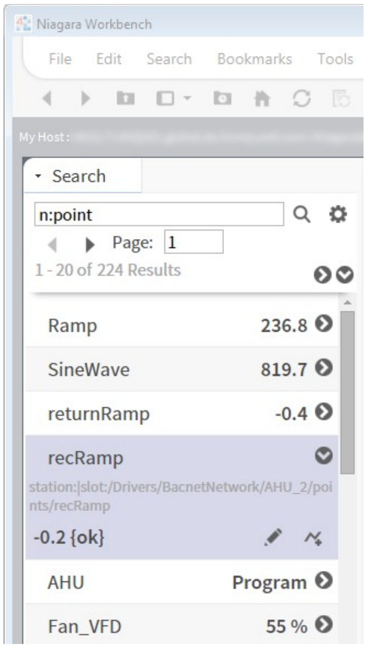
– Module: This is the name of the palette’s parent module

– Description: This is a short title or name of the palette’s parent module

About the Search side bar

The SearchService uses Niagara Entity Query Language (NEQL) syntax to query the system for component tags.

Figure 172: Search side bar



My Host : [redacted]

n:point

Page: 1

1 - 20 of 224 Results

Ramp 236.8

SineWave 819.7

returnRamp -0.4

recRamp

station:|slot:/Drivers/BacnetNetwork/AHU_2/points/recRamp

-0.2 {ok}

AHU Program

Fan_VFD 55 %

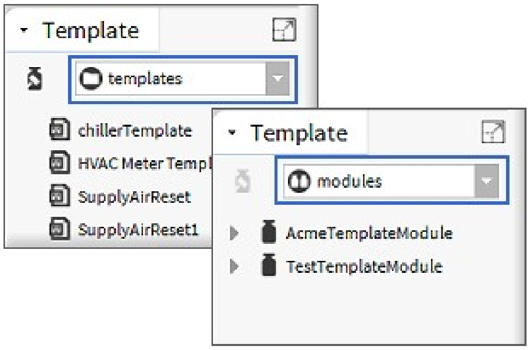
The Search side bar provides a query field for entering your search criteria. For example, you might enter “n:point” to query for all points in the station that have the Niagara tagdictionary point tag.

Click the gear icon to configure the number of search results to display per page. The “>” or “<” page through the results. The search results area provides access to additional information too. Click the “>” icon to the right of a result to expand it, showing the Ord and current status. In an expanded result, click the icons (at right) to view the live data either in a gauge or chart.

Template side bar

The Template side bar provides access to template files located in the Workbench User Home /templates folder as well as to templates stored in modules located in the SysHome /modules folder.

Figure 173: Template side bar



chillerTemplate

HVAC Meter Templ

SupplyAirReset

SupplyAirReset1

AcmeTemplateModule

TestTemplateModule

The pull-down list in the side bar switches the view between the \templates and \modules folders. When the \modules folder is selected, expand any module to see the template files contained within.

Double-click on a template file to open it in the Template view. When you open a file in the templates folder you can proceed to make changes and save the file. Optionally, you can create a new variation of an existing template by clicking Save As in the view to save it with a new name.

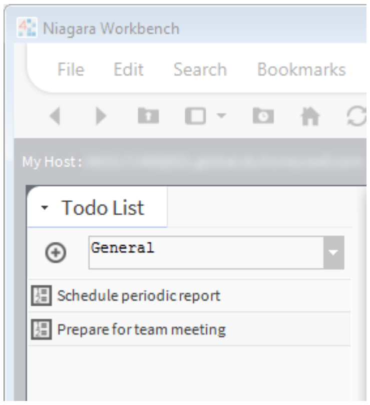
Note:

Any template stored in a module is a read-only file which you cannot edit. When you open a template in a module, you will see “ReadOnly” in the top left corner of the Template view. To make changes you must first click Save As and save the template with a different filename in the Workbench User Home /templates folder.

About the Todo list sidebar

The Todo List sidebar is a convenient way to create and access Todo List items from a palette.

Figure 174: Todo bar



Schedule periodic report

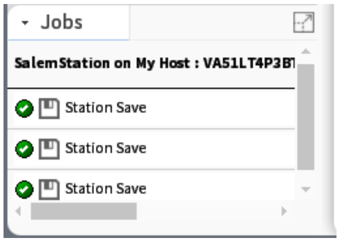
Prepare for team meeting

Click the + button on the toolbar to open the Add window. This window provides a text fields for adding and categorizing Todo list items.

About the Jobs side bar

The Jobs side bar shows all the current jobs in all the stations with which you have a connection.

Figure 175: Jobs side bar



SalemStation on My Host : VA51LT4P3B1

Station Save

Station Save

Station Save

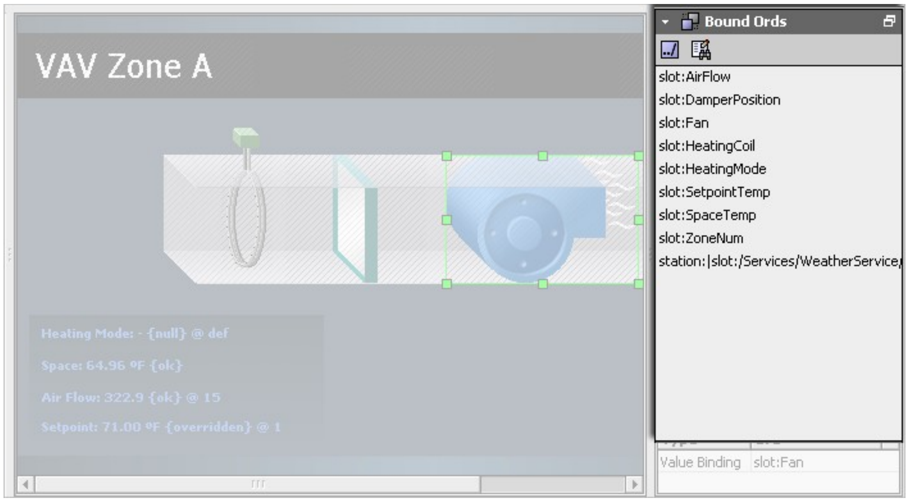
The current status of each job is shown as: running, canceling, canceled, success, or failed. If the job is running, a progress bar displays estimated progress.

You may cancel a running job by pressing the Cancel icon. Normally, once a job has completed you are notified via the async notification feature. You may then dismiss the job by pressing the Close icon. The details of the job may be accessed using the “>>” icon to display the Job Log dialog box.

About the bound ords side bar

The bound ords side bar is available when the Px Editor view is active. It displays a listing of all the bound ords in the current Px view.

Figure 176: Bound ords side bar

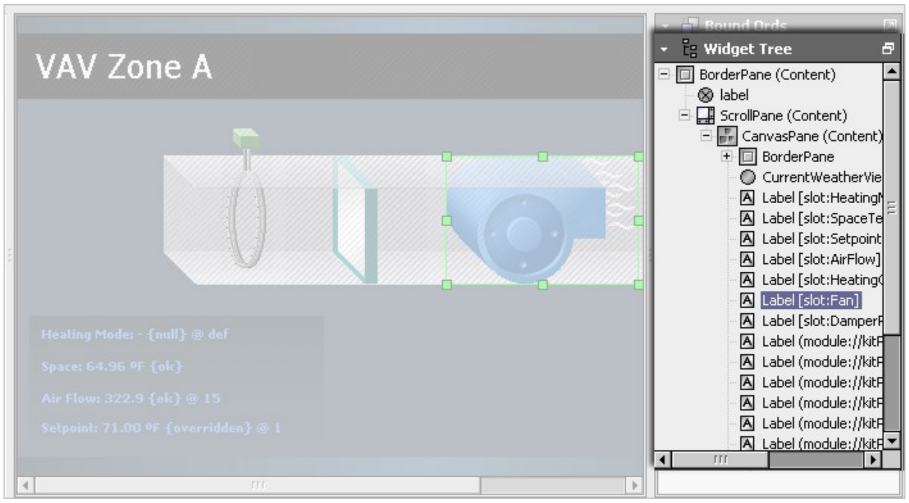


Double click on any ORD in the list to display the ORD in the ORD editor window.

About the widget tree side bar

The widget tree displays a tree hierarchy of the widgets (panes, labels, graphic elements, and so on) that are in the current Px view.

Figure 177: Widget tree side bar

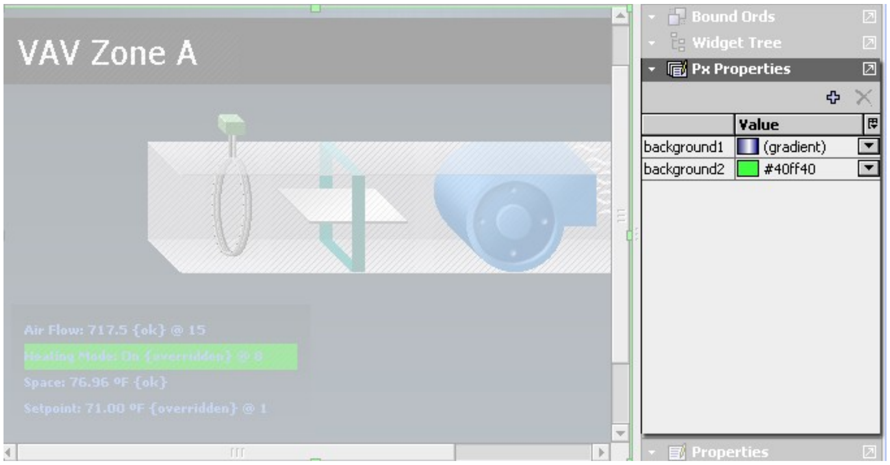


It is often easier to use the Widget Tree to select objects when you have a lot of objects on a view, especially when there are several layers of objects. When you select an object in the tree view it is selected in the Px view as well and displays the selection borders and handles.

About the Px properties side bar

This side bar is available when the Px Editor view is active. It displays a listing of all the Px properties that are defined in the currently active Px file.

Figure 178: Px properties side bar


VAV Zone A
Air Flow: 717.5 {ok} @ 15
Heating Mode: On {overridden} @ 8
Space: 76.96 °F {ok}
Setpoint: 71.00 °F {overridden} @ 1
Bound Ords
Widget Tree
Px Properties
Properties

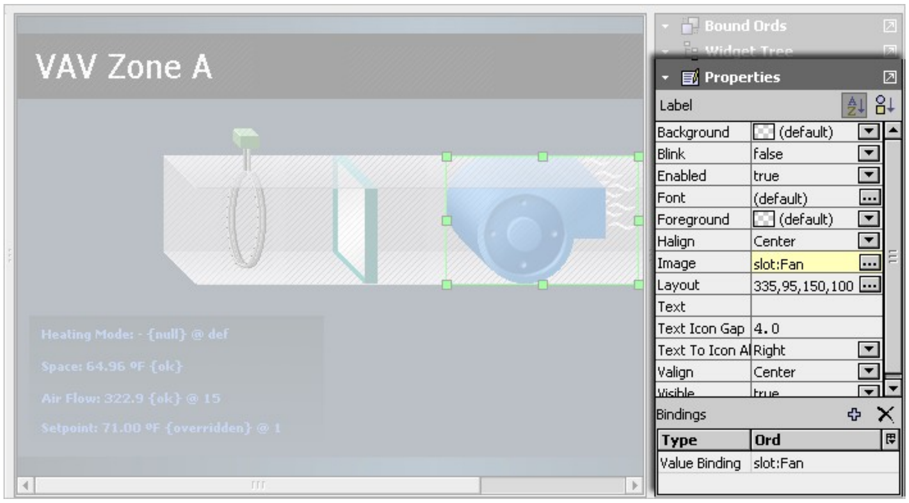
	Value
background1	(gradient)
background2	#40ff40

Use the menu bar icons to add, define, assign, and delete Px properties.

About the Properties side bar

This side bar is available when the Px Editor view is active. It displays a listing of all the properties that are in the currently selected object in the Px view.

Figure 179: Properties side bar


VAV Zone A
Heating Mode: - {null} @ def
Space: 64.96 °F {ok}
Air Flow: 322.9 {ok} @ 15
Setpoint: 71.00 °F {overridden} @ 1
Bound Ords
Widget Tree
Properties

Label

Background	(default)
Blink	false
Enabled	true
Font	(default)
Foreground	(default)
Halign	Center
Image	slot:Fan
Layout	335,95,150,100
Text	
Text Icon Gap	4.0
Text To Icon Al	Right
Valign	Center
Visible	true

Bindings

Type	Ord
Value Binding	slot:Fan

Double click on any object in the widget tree or in the in the Px viewer to display the properties window (same information as the properties side bar).

Types of edit commands

In addition to view–specific editing tools, Workbench provides commands for editing components in many of the views. Following, is a list of descriptions of the standard commands that are available in Workbench views for editing components.

- Drag

Dragging files or components only works within a single Workbench application. However, it is possible to drag files from Windows Explorer into the Workbench to see the default view. Dragging a component or file using the left mouse button performs a Copy operation. Dragging a component or file using the right mouse button always prompts you for a Copy, Move, or Cancel command selection.

- Cut

Use the Cut command to delete the selected object and send it to the clipboard.

- Copy

Use the Copy to send the selected object to the clipboard without deleting it.

- Paste

Use the Paste command to copy the current contents of the clipboard to the destination as a set of new dynamic properties.

- Duplicate

Use Duplicate to create a copy of the current selection in the same container as the selection.

- Delete

Use the Delete command to remove a selected item from its parent container.

- Undo

Use the Undo command to reverse the previous command. Undo is only available for certain commands, such as, Paste, Cut, Delete, and the Link action.

- Redo

Use the Redo command to restore a command–action after the Undo command has removed it.

- Rename

Use Rename to change the name of a Component.

Types of toolbar icons

Some icons are always visible as you navigate through the system. Additional icons may be added and removed from view when different views are active. When icons appear dimmed, their functions are unavailable.

Icons that appear on the toolbar are grouped in the following categories:

- Standard toolbar icons

These icons may be dimmed (or unavailable in certain views) but they are always present on the toolbar.

- Slot sheet toolbar icons

These icons appear only when the Slot Sheet is active.

- Px Editor toolbar icons

These icons appear only when the Px Editor or Px Viewer is active.

- History extension manager toolbar icons

These icons appear when the History Editor view is active.

- History Editor toolbar icons

These icons appear when the History Editor view is active.

- Todo list toolbar icons

These icons appear when the Todo list view is active.

Standard toolbar icons

A number of toolbar icons are available in all views.

- Back.
- Forward.
- Up Level.
- Recent Ords.
- Home.
- Refresh.
- Left Side Bar.
- Open.
- Save Ctrl + S.
- Save Bog.
- Printing Ctrl + P.
- Cut Ctrl + X.
- Copy Ctrl + C.
- Paste Ctrl + V.
- Duplicate Ctrl + D.
- Delete.
- Undo Ctrl + Z.
- Redo Ctrl + Alt + Z.

Slot Sheet toolbar icons

When the Slot Sheet view is active, a number of additional icons are available.

- Add Slot

This creates a new slot (appearing as a row) on the slot sheet.

- Rename Slot

This displays the Rename Slot dialog box, when clicked. This icon is dimmed when no slot or more than one slot is selected in the slot sheet editor view.

- Config Flags

This displays the Config Flags dialog box, when clicked. This icon is dimmed unless one or more slots are selected in the slot sheet editor view.

- Reorder

This displays the Reorder dialog box, when clicked.

Px Editor toolbar icons

When the Px Editor view is the active a number of additional icons are available.

- Toggle View/Edit Mode.

Displays, alternately, the Px Editor or the Px Viewer in the view pane. This icon appears inset when the Px Editor view is active and it appears normal when the Px Viewer is active.

- Right (Px Editor) side bar menu

Displays a dropdown menu of side bar options for the Px Editor. The following options are available:

– Bound Ords: Shows or hides the Bound Ords side bar.

– Widget Tree: Shows or hides the Widget Tree side bar.

– Properties: Shows or hides the Properties side bar.

- Left align

Aligns left edges of selected objects along a vertical line.

- Right align.

Aligns right edges of selected objects along a vertical line.

- Top align

Aligns top edges of selected objects along a horizontal line.

- Bottom align

Aligns bottom edges of selected objects along a horizontal line.

- To Top

Moves selected objects to the highest position (with regard to z-order) in the parent object.

- To Bottom

Moves selected objects to the lowest position (with regard to z-order) in the parent object.

- Select

Activates the pointer tool used to select objects in the Px Editor view using the mouse.

- Add Polygon

Activates the polygon tool for drawing polygons.

- Add Path

Activates the path tool that allows you to draw bezier curves in the Px Editor view.

- Add Point

Activates the Add Point tool that allows you to add a point to a path or a polygon in the Px Editor view.

- Delete Point

Activates the Delete Point tool that allows you to remove a point from a path or a polygon in the Px Editor view.

About the history extension manager toolbar icons

The Workbench toolbar contains navigation and editing buttons as described in “About the toolbar”. Following are the icons that appear when the history extension manager view is active:

- HistoryExtManager Enable Collection

Click this icon to enable (start the collection process) for the selected entries.

- HistoryExtManager Disable Collection

Click this icon to disable (stop the collection process) for the selected entries.

- HistoryExtManager Rename History

Click this icon to rename the selected history. When clicked, the Set History Name dialog box appears.

- HistoryExtManager Edit System Tags

Click this icon to open the Set System Tags For Selected History Extensions dialog box. Use this dialog box to edit system tags associated with a single history extension or perform batch edits when you have more than one history extension selected.

About the history editor toolbar icons

Following are the icons that appear when the history editor view is active.

- Hide

With one or more records selected, this icon is available to set the trend flag of all selected records to “hidden”. With no records selected, the icon is dimmed (unavailable).

- Unhide

With one or more records selected, this icon is available to set the trend flag of all selected records to “hidden”. With no records selected, the icon is dimmed (unavailable).

- Filter

Opens the Configure Flags window, when clicked. This icon is available even if no records are selected.

- Configure Outliers

Opens the Configure Outliers window, when clicked.

About the Todo list toolbar icons

Following are the icons that appear when the Todo list view is active.

- Add

This icon opens the Add dialog box, when clicked. Use this icon to add a new item to your Todo checklist and assign a Summary and Group to the item. This icon is available even if no items are selected.

- Mark Complete

This icon, when clicked, dims and lines-through the selected item(s) in the Todo list so that the item(s) appears to be “crossed off” the list. If the item is already marked as completed and this icon is selected, the item will be restored to its “unmarked” state. With no items selected, this toolbar icon is dimmed (unavailable).

- Edit

Displays the Edit dialog box. When clicked allows you to change the Summary and the Group fields associated with the item.

- Move to Top

Moves the selected item to the top of the list.

- Move Up

Moves the selected item up in the list, one increment per click.

- Move Down

Moves the selected item down in the list, one increment per click.

- Move to Bottom

Moves the selected item to the bottom of the list.

- Remove

Displays a “Remove selected items?” prompt, when clicked; deletes selected items when the prompt is affirmed.

Types of console commands

The Workbench console commands provide additional functionality.

The following types of commands may be typed in from the Workbench console.

- Niagara shell commands

These commands are used at the command line and may be typed in directly:

- nre commands

These commands are used at the command line and may be typed in using the “nre” prefix.

- wb commands

These commands are used at the command line and may be typed in using the “wb” prefix.

- plat commands

These commands are used at the command line and may be typed in using the “plat” prefix.

Shell commands

- cd

This command displays and changes the current directory. Type cd <directory name> to change to a specific directory. Type cd to display the current directory.

- debug

This command turns debug tracing on and off. Type debug on to turn on debug. Type debug off to turn off debug.

- print

This command prints a message to the output stream. Type print <message> to print the literal message to the console.

- reset

This command resets the environment to its default state. Type reset to set the environment to its default state.

- set

This command displays and modifies environment variables.

  - Type to set to display all the environment variables.
  - Type set <prefix> to display all the environment variables that start with the specified prefix.
  - Type to set <name>= to remove the “named” variable.
  - Type to set <name>=<value> to set the “named” variable to the “value” specified.

- which

This command resolves a filename in a path. Type to which <filename> to find the first occurrence of the specified “filename”.

nre (station) commands

Use the following syntax with the nre command:

nre [options] <class> [args]*

The following parameters may be used with the nre command.

- class

This is a class name or a module:classname to execute.

- args

This is the name of one or more arguments to pass through to main. The following options may be used with the nre command.

- -version

This option displays the nre version.

- -modules:<x>

This option displays the modules that match the pattern defined by “x”.

- -hosted

This option displays the id for the system host.

- -licenses

This option displays a summary of the license information.

- -props

This option displays a list of the system properties.

- -locale:<x>

This option allows you to set the default locale. For example, to set the default locale to US English, type: -locale:en_US

- -@option

This option allows you to pass the specified option to the Java VM.

- -testheap

This option tests and displays the max heap size.

- -buildreg

This option causes a rebuild of the registry.

wb (Workbench) commands

The wb command starts up an instance of Workbench. Use the following syntax with the wb command:

wb [options] <ord>

The following parameter may be used with the wb command.

- ORD

This option specifies the ORD of the initial view that you want display when Workbench starts up. The following options may be used with the wb command.

- -profile

This option specifies the Workbench profile to assign when Workbench starts up.

- -file:ord

This option specifies the initial file to display when Workbench starts up.

- -locale<x>

This option sets the locale on startup.

- -@<option>

This option allows you to pass the specified option to the Java VM.

plat (platform) commands

Use the following syntax with the plat command:

plat <command> <flags> <command-flags>

The following commands may be used with plat.

- details

This command displays a configuration summary for a remote host.

- fget

This command gets one or more files from a remote host.

- flist

This command provides file details for a single file, or for all files in a directory.

- ipconfig

This command displays the TCP/IP configuration for a remote host.

- jacejar

This command creates Niagara module files that can be run on embedded hosts.

- liststations

This command lists stations that are managed by the Niagara platform daemon.

- moduleinstall

This command installs Niagara modules to a remote host.

- reboothost

This command requests that a remote Niagara platform daemon reboot its host.

- script

This command runs one or more platform commands in a script.

- startstation

This command requests that a Niagara platform daemon start a station.

- stopstation

This command requests that a Niagara platform daemon stop a station.

- tellstation

This command sends text to the console of a running Niagara station.

- watchstation

This command monitors the output of a Niagara station.

- installdaemon

This command installs the Niagara platform service (Win32 only).

- uninstalldaemon

This command removes the Niagara platform service (Win32 only).

- installdialup

This command installs the Niagara dialup service (Win32 only).

- unistalldialup

This command removes the Niagara dialup service (Win32 only). The following options may be used with the plat command.

- -usage

plat -usage prints the help listing in the console

plat <command> -usage prints the command specific usage in the console

- -?
  - plat -? prints the help listing in the console
  - plat <command> -? prints the command specific usage in the console
- -help

  – plat -help prints the help listing in the console

  – plat <command> -help prints the command specific usage in the console

- -locale:<x>

This option sets the default locale (en_US).

- -@<option>

This option passes the option to the Java VM.

- -buildreg

This option forces a rebuild of the registry.

GLOSSARY

alarm	A notification that a defined event has occurred or an indication that some value is not within an appropriate or expected range. For example, a security breach, temperature limit, or equipment malfunction can initiate an alarm notification. Text and icons on the Alarm Console identify alarm severity.
alarm console	A table view that lists all current alarms for an individual station. This view is available on the alarm recipient component (in the Alarm Service).
alarm portal	A table view that lists alarms collected from multiple stations. To access this view from the main menu, click Tools→AlarmPortal.
category	A logical grouping of system objects (components, files and histories), either by directly assigning objects to the category, or by creating a NEQL query that returns a group of objects. Two types of categories are supported: • Basic categories, for example, may be used to group objects by geography (floor 1, floor 2, etc.) or type of object (lighting, HVAC, etc.). Each category may be subject to separate user roles and permissions. Each new station has two default categories: user (category 1) and Admin (category 2). • Tagged categories group objects based on a NEQL query. Assuming that you tag each new component with an identifying tag, your tagged category includes all components that satisfy the query. Adding a component only requires you to set up the component and tag it appropriately. You do not have to assign the component to a category. You manage categories using the CategoryService.
certificate	A PKI (Public Key Certificate) or digital certificate is an electronic document used to prove ownership of a public key. The certificate includes information about the key, the identity of its owner, and the digital signature of an entity that verified the validity of the certificate's contents. If the signature is valid, and the client can trust the signer, the client can be confident that it can use the public key contained in the certificate to communicate with the server.
component	A piece of self-describing framework software that can be assembled like building blocks to create new applications. Components represent individual points, such as outside temperature, office temperature, occupancy, and schedules that generate analog data for analysis within the system. Components differ from modules in that components comprise an implementation of the framework, whereas modules comprise the framework software itself.
config flag	A configuration flag is a boolean value that is stored as part of a bitmask on each slot of a Baja object. Some flags apply to all slot types, while others only have meaning for certain slot types. You access a slot's config flags by right-clicking the slot and clicking ConfigFlags.
container	For example: WebWidget

control point	In the narrowest terms, control points refer to the eight point types found in the baja control palette under the Points folder. In broader terms, control points include components from the control and kitControl palettes. Most of these components are based on the eight basic point types. They inherit from BooleanPoint, EnumPoint, NumericPoint, and StringPoint. ControlPoint is the base class for all point types in the Baja control architecture. A ControlPoint maps to one value that a driver reads or writes. All Control Points have a StatusValue property called Out. If the predefined proxyExt is not a NullProxyEct, the system considers the point a proxy point. This means that it is a local representation of a point that actually exists in an external device. The framework uses the driver to maintain synchronization.
framework	Software that provides generic functionality. The framework can be customized by adding user-written code.
history	An ordered collection of timestamped records. Each history is identified by a unique id. Histories can be periodically archived to a remote history database (archive). A history database is a set of histories. History is also used as a scheme in ORDs to refer to collective histories.
niagarad	The Niagara daemon (niagarad) is a server process used to communicate between Workbench (as a client) and the platform to which Workbench is connected.
node	1. A connection point between the system and a real or virtual device. Devices become a node when they register with the system, providing a name and connection information. An ORD provides access to the device. 2. A position in a Nav tree hierarchy. 3. The primary organizational unit of a Niagara Analytics Framework data model tree. Like a folder, a node is a container that holds points or other containers. The nodes of the Niagara Analytics Framework data model provide the structural framework for the model.
object	An object is the base class required for all system entities that conform to the baja model. Objects group information used to construct a model that includes building devices, virtual devices, individual points, users, system features and services. Objects appear in the Nav tree as files, modules, installers, administrators, copiers, drivers and apps. Metadata associated with objects, including categories, roles (permissions), and hierarchies, provide access control and configuration options to manage automated buildings efficiently.
ORD	An ORD is an “Object Resolution Descriptor”. The ORD is the Niagara universal identification system and is used throughout the framework. The ORD unifies and standardizes access to all information. It is designed to combine different naming systems into a single string and has the advantage of being parsable by a host of public APIs.
palette	The palette provides a hierarchical view of available components. You copy or drag a component from a palette and paste it or drop it where you need it, on a wire sheet, property sheet, Px View, or in the palette Nav side bar pane.
point extension	A component that extends control of point behavior in a consistent manner. Each property of a ControlPoint that exists as a subclass of a PointExtension is considered an extension on the point. Extensions allow plug-in functionality, such as alarming and historical data collection via special hooks that a ControlPoint provides to the PointExtension.

PX Editor	A tool for creating graphical representations of ducting, piping, and the equipment used in a building. The editor allows the creation of PC and mobile views.
template	A deployable package of Niagara objects used to streamline repetitive configuration steps when making multiple installations with similar functionality. For example, when setting up a new device by deploying a device template, only unique device properties require configuration. Templates are indexed and searchable.

LG Customer Information Center, Commercial Products

1-888-865-3026 USA

Follow the prompts for commercial A/C products and parts.

LG Electronics, U.S.A., Inc.
Commercial Air Conditioning Division
4300 North Point Parkway
Alpharetta, Georgia 30022
www.lghvac.com