Magic xpa
Mobile Web Samples Guide

This document contains a collection of guidelines for developing an HTML-based application for your mobile device using Magic xpa as the backend server.

Prerequisites

To use the application, you need to have your Magic xpa configured for running a Merge or a Browser Client program.

Running the Application

Before you run this application, you should set the Deployment mode of the engine (under Options\Settings) to Background.

To launch the application, simply switch the engine to runtime mode (Ctrl+F7) and run the Logon program using a FireFox browser on your desktop or a Mobile device. You can do this via the following URL:

http://server/MagicScripts/MGrqispi.dll?appname=MobileWeb&prgname=Logon

File Location

This sample resides in two main folders:

1. The current folder, which contains the source code and files that are not accessible from the outside world.
2. A subfolder under the **%EngineDir%Scripts** folder. This folder contains the java scripts and files that need to be accessed via the browser.

How It Works

Mobile Device Detection

Magic xpa automatically detects the mobile device according to the browser's **USER_AGENT** var.

This var is sent from the mobile device to Magic xpa based on the definition of **HttpVars = HTTP_USER_AGENT** in the **%EngineDir%Scripts\mgreq.ini** file.

The **USER_AGENT** is then parsed in the Main program in order to detect the Mobile Device type.

Note that each request and its **USER_AGENT** string is added to the **WorkingDir%requests.log** file, so if your device was not detected, you can view this file to see the reason.

Logon

The first access to the application is done to the Logon program. This program simply loads the Logon.html page and uses the Magic xpa Merge technology to send it to the client.

Using Magic xpa programs instead of directly referring to the HTML files has some advantages:

- The HTML files are not required to be exposed to the outside world (in fact, they can be located on another server).
- By using the Merge technology, the HTML content can vary according to logic defined in the program. The Logon program, for example, also accepts a message, which will be shown in case of an error in another program.

The disadvantage in using Magic xpa programs for returning the HTML page is the additional request to the Magic xpa engine.

At the logon page, clicking the Logon button will call the VerifyLogon program in order to verify the user name and password.

Note that in this sample, there is no logic, so the logon always succeeds. You can change this program to fit your security requirements.

After the logon, a context is opened and the application menu is shown.

We will elaborate more on the context later on.

Menu

Similar to the Logon program, the Menu program is used to show the menu.

Products, Customers and Orders Lists

Similar to the Logon program, these programs are used to show the corresponding HTML file.

Each of the HTML files has a link to a JavaScript file that handles the logic of the page.

In this sample, Ajax is used to send an asynchronous request to the server in order to get the data.

This was done in order to benefit from:

- No need to refresh the entire page upon a change in the range.
- Passing data in chunks of records to the client and not all the records at once.
- The data transfer is not disconnected from the web page design, so you can change the page design or JavaScript logic without changing the Magic xpa programs.

The disadvantage here is the additional request to the server to get the data. If you have static pages or pages with small data, you may consider sending this data together with the page load request.

Order Details

The order details screen allows you to update the order details. The updated data is then sent to the server using a Synchronous Ajax call.

Logout

The Logout program closes the current context and calls again to the Logon program.

Context Management

The application creates a context per each user and handles all the requests from the user in the same context. This allows us to keep track of the user data and handle the user authentication only on the first page.

This context is created by the following definitions:

1. The Menu program (which is the first program in the context) is defined with **Keep created context**, so the context is kept open on the server.
2. The context ID is stored on the client using a cookie. This is done in the Logon Validation program.
3. Each request passes the context ID using the **CTX** HTTP argument. This is done in the JavaScript that generates the URL for the requests.

To reduce threats of fake IDs, you can define **HttpSigVars = REMOTE_ADDR** in the requester ini (**%EngineDir%Scripts\mgreq.ini**) file. This setting will block users from trying to use the same context ID from a different machine.

Note: Using a context is not mandatory. If you do not use a context, you should identify the user per each request.