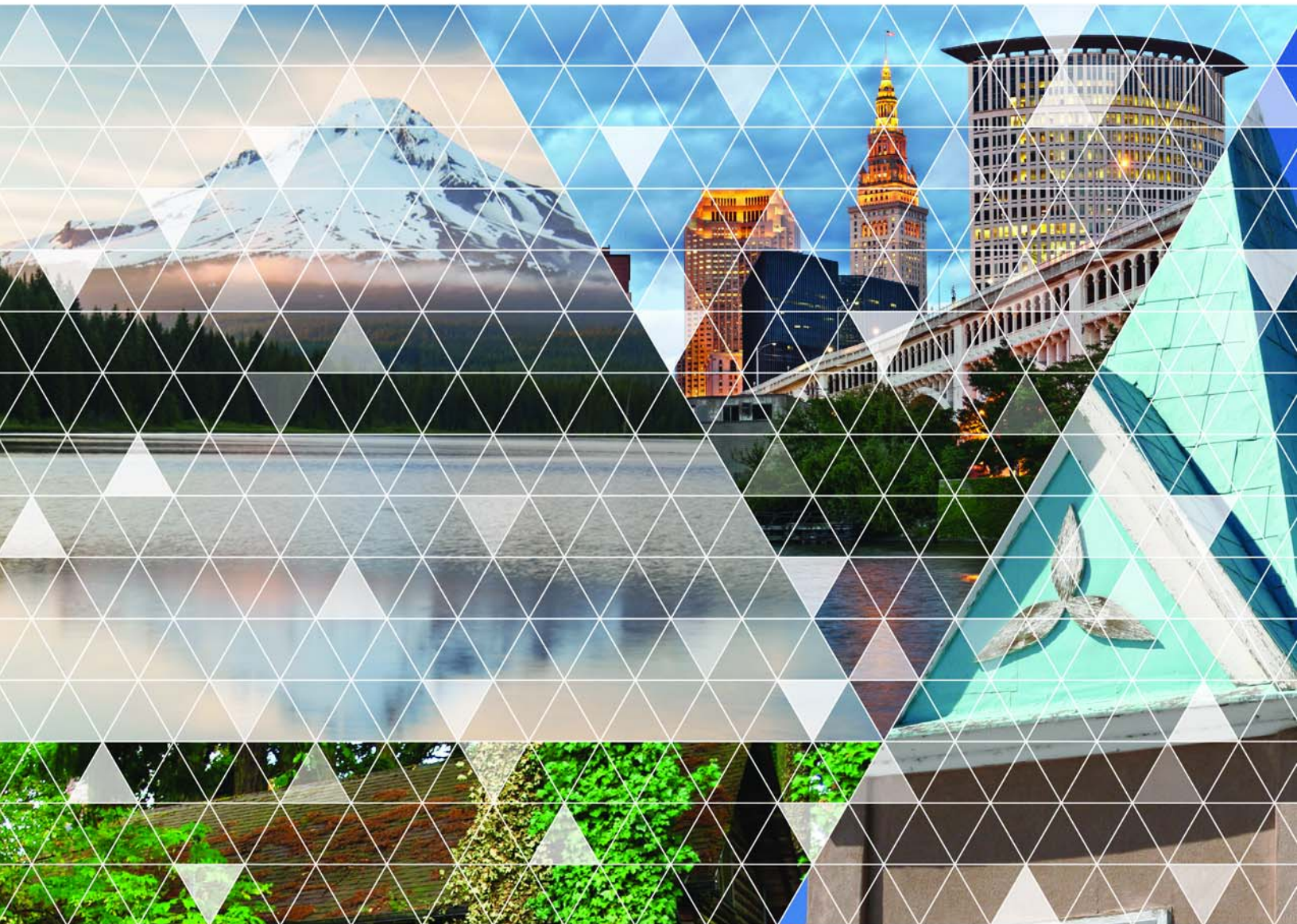


Accela Civic Platform®

# ePayments SDK Guide



## **Accela Civic Platform ePayments SDK Guide**

© 2016 Accela, Inc. All rights reserved.

Accela, the Accela logo, the Accela logo with “Government Software” notation, Accela Automation, Accela Asset Management, Accela Citizen Access, Accela Mobile Citizen Access, Accela ERS, Accela GIS, Accela IVR, Accela Land Management, Accela Licensing, Accela Mobile Office, Accela Public Health and Safety, Accela Service Request, Accela Wireless, Kiva DMS, Kiva Development Management System, 'PERMITS' Plus, SiteSynch, Tidemark Advantage, Civic Platform, Civic Cloud, Civic Hero, E-Boardroom, EnvisionConnect, Envista, GEOTMS, IQM2, Mediatraq, Minutetraq, PublicStuff, Trusted To Do More, VelocityHall, Vantage360, and other Accela logos, devices, product names, and service names are trademarks or service marks of Accela, Inc. Brava! Viewer is a trademark of Informative Graphics Corporation. Windows is a registered trademark of Microsoft Corporation. Acrobat is a trademark of Adobe Systems Incorporated. Portions copyright 2009 Ching-Lan 'digdog' Huang and digdog software. All other company names, product names, and designs mentioned herein are held by their respective owners.

**Version 8.0.3.0.0**

**August 2016**

### **Corporate Headquarters**

2633 Camino Ramon  
Suite 500  
Bishop Ranch 3  
San Ramon, CA 94583

Tel: (888) 722-2352

Fax: (925) 659-3201

[www.accela.com](http://www.accela.com)

# Contents

<b>Getting Started.....</b>	<b>4</b>
Checking the ePayment SDK Package.....	5
Deciding on Which ePayment Adapter to Use.....	6
Choosing Your ePayment Solution.....	7
Obtaining ePayment Gateway Information.....	9
Setting Up ePayment Adapter.....	10
<b>Developing Your Customized Adapter.....</b>	<b>12</b>
Gathering Required Info for Customized Adapters.....	13
Developing ePayments3 Adapter.....	14
Developing Your Payment Adapter with the .NET Sample Adapter.....	25
Developing Your Payment Adapter with the Java Sample Adapter.....	29
Implementing makePayment and voidPayment Calls.....	32
Developing Citizen Access Online Payment Adapter.....	38
Preparing Initial Parameters.....	39
Handling Payment Status.....	40
Handling Payment Notification.....	42
<b>Configuring ePayment Adapter.....</b>	<b>43</b>
General Configuration Steps Testing and Live.....	44
Setting Payment Types for Govolution and First Data Adapters.....	47
<b>Standard Choices Configuration.....</b>	<b>48</b>
Standard Choice EPaymentAdapter.....	49
Standard Choice PAYMENT_CHECK_ACCOUNT_TYPE.....	50
<b>XPOLICY Table Settings for ePayment Adapters.....</b>	<b>51</b>
XPOLICY for PayPal Payflow Pro4.3 Adapter.....	52
XPOLICY for Official Payments STP Adapter.....	53
XPOLICY for Virtual Merchant Adapter.....	55
XPOLICY for Official Payments CoBrand+ Adapter.....	56
XPOLICY for Govolution Adapter.....	58
XPOLICY for First Data Adapter.....	59
XPOLICY for ePayments3 Adapter.....	60
XPOLICY for MultipleAccountsEPayment Adapter.....	61
XPOLICY for ACA Online Payment Adapter.....	62
XPOLICY for CivicPay Adapter.....	63
<b>Terms and Definitions.....</b>	<b>64</b>

## Getting Started

---

The ePayment SDK package provides the components needed for building ePayment custom adapters.

### **Related Information**

[Checking the ePayment SDK Package](#)

[Deciding on Which ePayment Adapter to Use](#)

[Choosing Your ePayment Solution](#)

[Obtaining ePayment Gateway Information](#)

[Setting Up ePayment Adapter](#)

## Checking the ePayment SDK Package

---

The Civic Platform ePayment SDK package contains the scripts and documentation for building and configuring the ePayment adapter. The package contains three folders besides this document:

- The **Sample Code** folder provides the code stub and the test harness to assist you in developing your customized adapter.
- The **scripts** folder provides the script files that you need to run in the database server for configuring each type of adapter in Civic Platform.
- The **WSDL** folder provides the ePayments3.wsdl file for ePayment web services.

This document refers to the source files and the scripts in the package in various places.

## Deciding on Which ePayment Adapter to Use

---

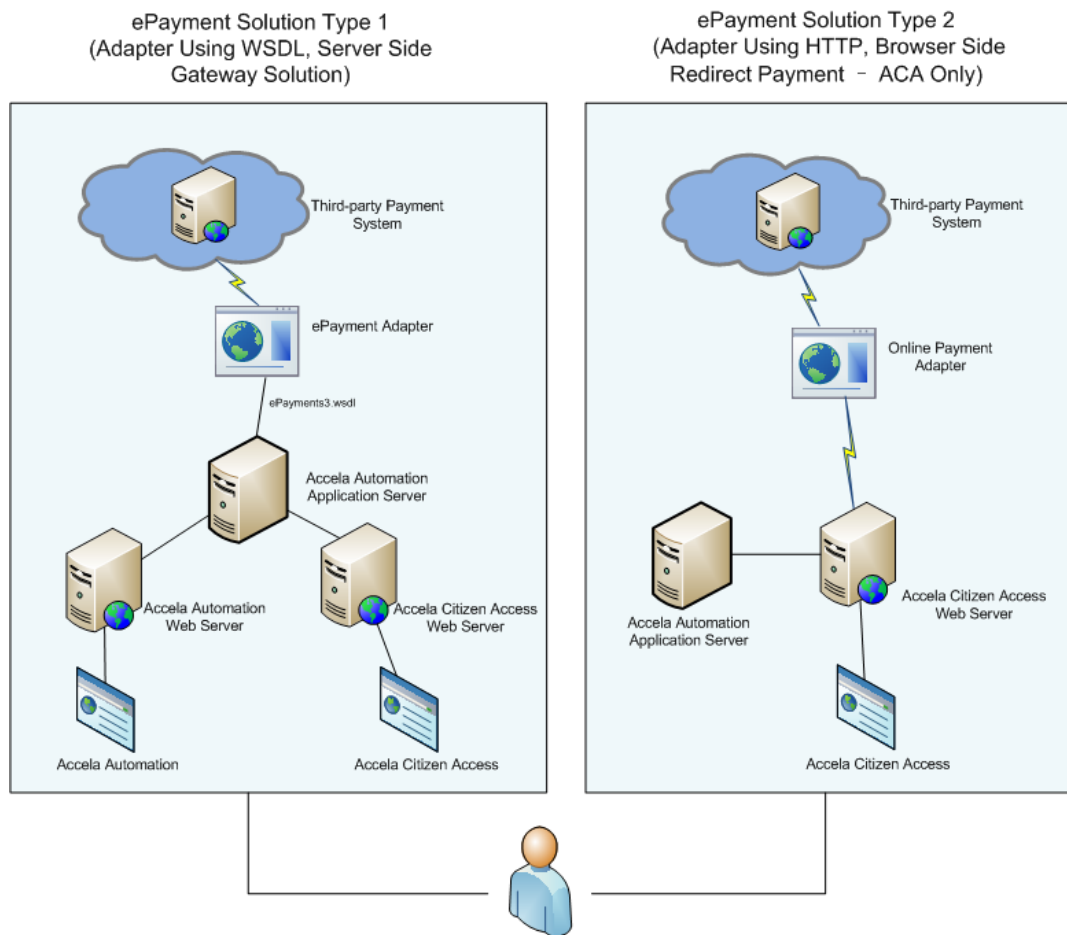
Civic Platform provides built-in ePayment adapters for the following ePayment gateways:

- CivicPay
- PayPal Payflow Pro 4.3
- Official Payments STP
- Virtual Merchant
- Official Payments CoBrand+ (for Citizen Access only)
- Govolution (for Citizen Access only)
- First Data (for Citizen Access only)

If your agency connects to an ePayment gateway which the built-in ePayment adapters do not support, you can create your own ePayment adapter, and then implement the adapter with the `epayments3.wsdl` file.

## Choosing Your ePayment Solution

Figure 1: [Accela ePayment Solution Diagram](#) shows how Civic Platform interacts with third party ePayment gateways through ePayment adapters.



**Figure 1: Accela ePayment Solution Diagram**

Consider which of the following two ePayment solution options suit your agency:

### 1. Server Side Gateway Solution

Both Civic Platform and Citizen Access supports this solution. With this solution, users can make payment directly in the Civic Platform website or Citizen Access website through the API provided by the online payment gateway.

This solution works with the following ePayment gateways:

- PayPal Payflow Pro 4.3 (with built-in adapter)
- Official Payments STP (with built-in adapter)
- Virtual Merchant (with built-in adapter)
- Or, your desired gateway (with customized ePayments3 adapter). For developing the ePayments3 adapter, see [Developing Your Customized Adapter](#).

### 2. Browser Side Redirect Payment Solution (for Citizen Access only)

This solution only works for Citizen Access 7.0.5 or later. Agencies create their own online payment adapter for Citizen Access. When users launch a payment request in Citizen Access, the payment adapter sends the payment request to the third-party payment gateway. Then the website of the third-party payment provider opens for users to pay their fees there.

This solution works with the following ePayment gateways:

- CivicPay
- Official Payments CoBrand+
- Govolution
- First Data
- Or, your desired gateway (with customized online payment adapter). For developing the online payment adapter, see [Developing Citizen Access Online Payment Adapter](#).



## Obtaining ePayment Gateway Information

---

Prior to configuring the ePayment solution, make sure you gather the following information about the ePayment Gateway:

1. Decide which ePayment gateway to use.

**Recommended Best Practice:** Use the PayPal Payflow Pro 4.3 payment gateway.

---



**Note:**

The test gateway for PayPal Payflow Pro was retired by PayPal on August 1, 2009, and the live gateway for PayPal Payflow Pro was retired in September 2009.

Please use the PayPal Payflow Pro 4.3 payment gateway to test PayPal online payments. Contact Accela CRC to assist you in migrating to the PayPal Payflow Pro 4.3 payment gateway.

---

2. Create a test account and a merchant account with the ePayment gateway provider.
3. Obtain the gateway host configuration information from your ePayment gateway provider, including:
  - a. The gateway URL.
  - b. The testing and/or live configuration information.
4. Be aware of the firewall and proxy requirements for the gateway provider.
5. Determine the ePayment adapter to access your ePayment gateway.

## Setting Up ePayment Adapter

### General Tasks for Setting Up ePayment Adapter

When you decide the ePayment gateway information, you must be already clear on the type of ePayment adapter to work with Civic Platform.

If you need to use a customized adapter, you must create the adapter first. See [Developing Your Customized Adapter](#) .

For either built-in or customized adapters, you must do the following two tasks:

1. Register the adapter and gateway information in the Civic Platform database server. See [Configuring ePayment Adapter](#).
2. Configure the related Standard Choices, for example, the required Standard Choice EPaymentAdapter. See [Standard Choices Configuration](#).

### (PayPal Payflow Pro 4.3 Adapter Only) Importing PayPal API Certificate

If you use the built-in adapter for PayPal Payflow Pro 4.3, follow the instructions below to import the PayPal API certificate into the Civic Platform application server.

To import the PayPal API certificate

1. Go to the PayPal site [https://cms.paypal.com/us/cgi-bin/?cmd=\\_render-content&content\\_ID=developer/apicertificates](https://cms.paypal.com/us/cgi-bin/?cmd=_render-content&content_ID=developer/apicertificates) to get a certificate.
  - For the test pilot-payflowpro.paypal.com, the certificate file is paypal43\_Test.crt
  - For the live payflowpro.paypal.com, the certificate file is paypal43\_Live.crt
2. Copy the certificate file `paypal43_Test.crt` or `paypal43_Live.crt` to the `<installdir>\av.biz\conf\certs` folder.
3. Run the DOS Command prompt.

4. Go to the

```
<installdir>\av.biz\conf\certs
```

folder with the following command:

```
cd D:\AA7.2.0\av.7.2.0\av.biz\conf\certs
```

5. Set the path of `keytool.exe` to system environment variable.

For example, run the following command:

```
set Path=D:\AA7.2.0\av.7.2.0\bin\jdk1.6.0\bin
```

6. Use the `keytool.exe` to import the certificate into

```
trusted_cacerts
```

in the server ( `<installdir>\av.biz\conf\certs` )

Run the following command:

```
keytool -import -file paypal43_Test.crt -keystore trusted_cacerts -alias paypal43_Test
```

Or:

```
keytool -import -file paypal43_Live.crt -keystore trusted_cacerts -alias  
paypal43_Live
```

## Developing Your Customized Adapter

---

This section provides instructions on developing your customized adapter. With Civic Platform, you can develop the customized adapter with the ePayments3 Web Service SDK that accompanies this document in the Scripts folder. With Citizen Access, you can either develop the ePayments3 adapter, or develop a custom online payment adapter.

If you plan to use a built-in adapter (see the list in [Deciding on Which ePayment Adapter to Use](#)), you can skip the instructions in this section.

The target audience of this section is software developers who know how to write, deploy, and test either Java using Java5 or higher and Ant, or .NET-based Web services using Visual Studio 2008 or higher and C#.

### Related Information

[Gathering Required Info for Customized Adapters](#)

[Developing ePayments3 Adapter](#)

[Developing Citizen Access Online Payment Adapter](#)

## Gathering Required Info for Customized Adapters

---

Make decisions on the following aspects before you start on creating the ePayments3 adapter:

1. Whether to implement the ePayment adapter with Java CXF, .NET, PHP, or Ruby. Currently we only provide test environment for Java CXF and .NET.
2. Where to install (or deploy) the adapter.
3. Whether to support credit card payment, check payment, or both.
4. Whether to support voiding payment.
5. What configuration data to be passed with each payment or void payment request, and what data to be configured internally:
  - a. What gateway host configuration data will be passed with every make payment or void payment request, and what data will be configured internally in the adapter.
  - b. What merchant configuration data will be passed with every make payment or void payment request, and what data will be configured internally in the adapter.
6. What payment processing data the adapter requires to process a payment request:
  - a. User Context
  - b. Request Context
  - c. General Payment Information
  - d. Credit Card Payment Information
  - e. Check Payment Information
  - f. Payment Billing Information
7. What Payment Processing Data the adapter requires to process a void payment request.
  - a. User Context
  - b. Transaction Context
8. How to format the payment result data, including:
  - a. Result Status
  - b. Transaction Information

## Developing ePayments3 Adapter

You can apply the provided ePayments Gateway 3 (ePayments3) code stub to develop the ePayments3 adapter. You can also use the code generator to create an empty stub and then develop your own adapter that comply with the namespace of Civic Platform.

**Best Practices:** Make a copy of the ePayments3 code stub for Java JaxWS 2.0 (CXF) or .NET, and then modify the code with the gateway information (collected in [Obtaining ePayment Gateway Information](#)) and adapter information (collected in [Gathering Required Info for Customized Adapters](#)).

The sample adapters are zipped along with this document:

- For Java JaxWS 2.0 (CXF), the sample adapter locates under the \Sample Code \SampleJavaEPG3Adapter folder.
- For .NET, the sample adapter locates under the \Sample Code\ SampleDotNetEPG3Adapter folder.

### Deploying Your Payment Adapter with the .NET Sample Adapter

All the required .NET sample adapter and code are available in the \Sample Code \DotNetEPG3TestHarness folder and \Sample Code\SampleDotNetEPG3Adapter folder.

### Implementing Your Payment Adapter in .NET

The web service implementation locates in the \SampleDotNetEPG3Adapter\App\_Code\Service.cs file. You can implement the makePayment and voidPayment methods based on your payment logic. Add your payment code between the following lines:

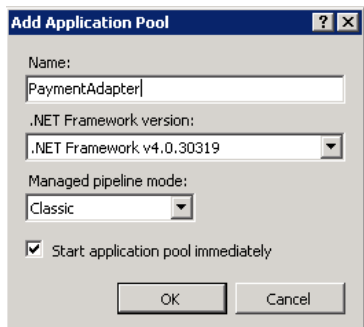
```
//Begin Implementation for EPayment
...
//End Implementation for EPayment
```

For more information, see [Implementing makePayment and voidPayment Calls](#).

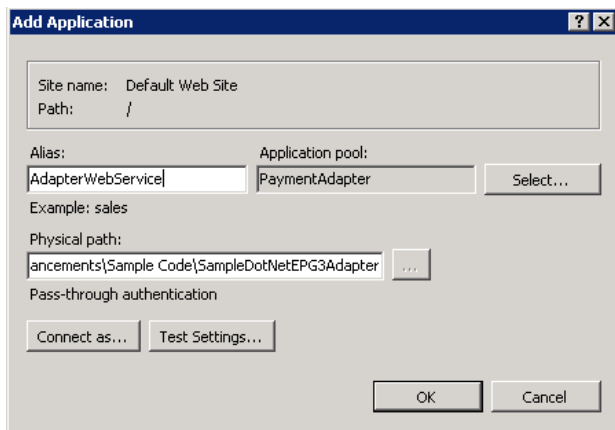
### Deploying Your Payment Adapter to IIS

#### To deploy your payment adapter to IIS

1. Open the IIS Manager.
2. Add an application pool for the application of your payment adapter.

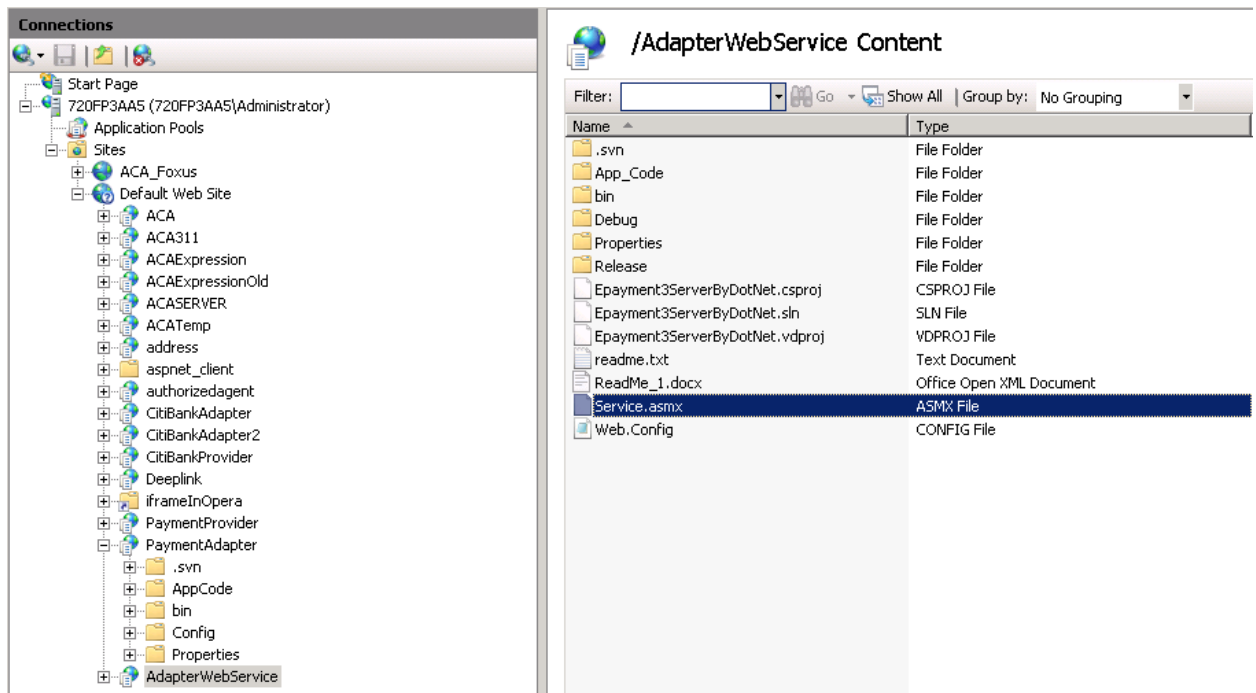


3. Add a web application for the payment adapter. Note that you must map the physical path of the web application to the \Sample Code\SampleDotNetEPG3Adapter folder.



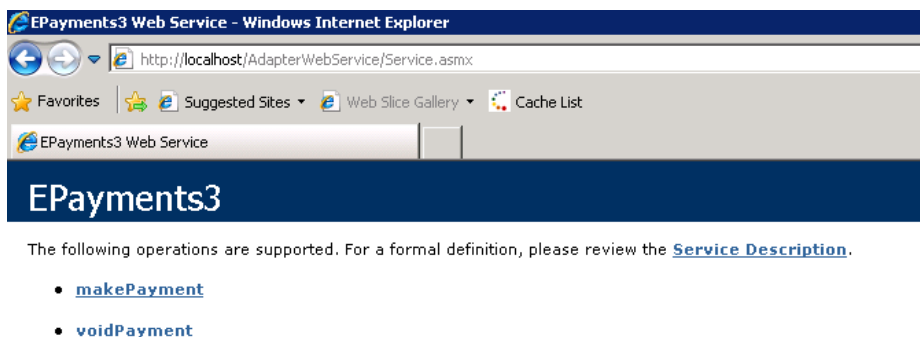
4. Test whether the deployment is successful.

- a. Right-click the payment adapter application, and click **Switch to Content View**.



- b. Right-click the Service.asmx file in the Content panel and click **Browse**.

If the deployment is successful, the following web page displays.



## Generating .NET Web Service Code Stub

You must generate the .NET web service code stub manually.

### To generate .NET web service code stub manually

1. Run wsdl.exe.

```
wsdl /si http://localhost:8080/web2/EPayments3?wsdl
```

2. Edit EPayments3Interfaces.cs. Change the following code:

From:

```
[System.Web.Services.WebServiceBindingAttribute (Name="EPayments3SoapBinding", Namespace="http://service.ws.accela.com")]
```

To:

```
[System.Web.Services.WebServiceBindingAttribute (Name = "EPayments3Port", Namespace = "http://service.ws.accela.com")]
```

3. Edit the web service implementation. Do the following:

```
namespace WebService2
{
    [WebService(Namespace = "http://service.ws.accela.com", Name = "EPayments3")]
    //add namespace and web service name

    public class Service1: IEPayments3SoapBinding
    {
        #region IEPayments3SoapBinding Members
        public EPaymentResult makePayment(EPaymentInfo paymentInfo)
        {
            mapItem[] paymentDataField = paymentInfo.configData;
            EPaymentResult result = new EPaymentResult();
            result.returnCode = "0";
            result.returnMap = paymentDataField;
            return result;
        }
        public EPaymentResult voidPayment(EPaymentInfo paymentInfo)
        {
            throw new NotImplementedException();
        }
    }
}
```



```

    }
    #endregion
}
}

```

## Configuring the .NET Test Harness

### To configure the .NET test harness

1. Open the following files located in \Sample Code\DotNetEPG3TestHarness, and update the configuration data for your test harness. Be sure to set up the proper test URL.

```

config/[ResourceName]/CreditCardConfigData.xml
config/[ResourceName]/CreditCardPaymentData.xml
config/[ResourceName]/CheckPaymentData.xml
config/[ResourceName]/CheckConfigData.xml

```

The [ResourceName] indicates the adapter type, be it PayPal43, OPSTP (official payment STP), or your customized adapter.

The value of the AdapterURL key is the test URL.

The configuration data are the XPOLICY table settings. For more information, see [XPOLICY Table Settings for ePayment Adapters](#).

2. Configure config/[ResourceName]/SSOConfig.xml.
  - Set “Enable” to false unless you are testing an internal ePayments3 adapter on av.biz.
  - Set “Enable” to true if you are testing an internal ePayments3 adapter on av.biz.
  - Set the user login parameters and the SSO URL.

## Calling the ePayments3 Web Service

### To call the ePayments3 Web service

1. Double-click the Epayment3ServerByDotNet.sln file in \Sample Code\SampleDotNetEPG3Adapter. Run the project in Visual Studio.
2. Open the ePayments3 test page.

The screenshot shows a web form with three dropdown menus: AdapterType (set to PayPay43), MethodType (set to CreditCard), and Method (set to Pay). There is a Button next to the Method dropdown. Below the form, a text box displays the result: "The web service return code is '0'; The return message is 'Success'; the return map is 11".

3. Specify the test parameters.

AdapterType	Select the web service adapter type. PayPay43 - If you want to use the PayPal43 adapter OPSTP - If you want to use the official payments STP adapter Customize - If you want to use your customized adapter
MethodType	Select the test payment method, credit card or check.
Method	Select the test method, pay or void.

4. Click **Button** to review the test result.

AdapterType	PayPay43
MethodType	CreditCard
Method	Pay
<input type="button" value="Button"/>	
The web service return code is "0";The return message is "Success";the return map is 11	

## Deploying Your Payment Adapter with the Java Sample Adapter

All the required Java sample adapter and code are available in the \Sample Code\JavaEPG3TestHarness folder and \Sample Code\SampleJavaEPG3Adapter folder.

### Preparing the Environment

Make sure you have installed the following software on the server where you want to deploy the payment adapter.

- Ant 1.7.0 or Ant 1.7.1. You can get it from <http://archive.apache.org/dist/ant/binaries> (apache-ant-1.7.1-bin.zip is recommended).
- Java JDK 5 or 6. You can get it from <http://www.oracle.com/technetwork/java/javase/downloads/jdk6-downloads-1637591.html>.
- Tomcat or JBoss. You can get tomcat and JBoss from the following paths.
  - <http://archive.apache.org/dist/tomcat/>
  - <http://www.jboss.org/jbossas/downloads/> (jboss-4.2.3.GA.zip is recommended.)

### Implementing Your Payment Adapter in Java

The web service implementation locates in the \SampleJavaEPG3Adapter\java\com\accele\ws\service\epayments3\EPayments3PayPallImpl.java file file. You can implement the makePayment and voidPayment methods based on your payment logic. Add your payment code between the following lines:

```
//Begin Implementation for EPayment
...
//End Implementation for EPayment
```

For more information, see [Implementing makePayment and voidPayment Calls](#).

### Deploying the Sample Payment Adapter

#### To deploy the sample payment adapter

1. Open SetEnv.cmd in the SampleJavaEPG3Adapter folder using a text editor.
2. Set ANT\_HOME to the actual Apache Ant install directory.  
For example:

```
set ANT_HOME=D:\apache-ant-1.7.1
```

3. Set JAVA\_HOME to the actual JDK install directory.  
For example:

```
set JAVA_HOME=D:\jdk1.6.0_43
```

4. Save your changes.
5. Double-click the Build.cmd file in the SampleJavaEPG3Adapter folder.  
The file creates two subfolders: build and dist. The dist folder contains the payment web service package named EPaymentServer3ByCXF.war.
6. Deploy the web service using Tomcat or JBoss. For example, if you use Apache Tomcat, do the following:
  - a. Copy and paste EPaymentServer3ByCXF.war into the \$TOMCAT\_HOME/webapps/ folder. Replace \$TOMCAT\_HOME with the actual Apache Tomcat install directory.
  - b. Double-click the startup.bat file under the \$TOMCAT\_HOME/bin/ folder.
7. Enter the following URL in your browser.

```
http://{serverIp:port}/EPaymentServer3ByCXF/CustomEpaymentAdapter?wsdl
```

Replace

```
serverlp
```

with the actual name of the server to which you deployed the sample adapter. Replace `port` with the actual port number that is used for Java.exe.

If the .wsdl file opens properly in your browser, the web service is deployed successfully.

## Configuring and Running the Java Test Harness

The test harness runs from the command prompt. The test harness reads the configuration and the test files stored in the resource directory and then test all of the ePayments3 calls, including credit card payment, void credit card payment, make check payment, and void check payment.

The Java test harness works with any ePayments3 Web Service adapter and can also test ePayments3 adapters built into the Civic Platform application server.

### To configure the Java test harness

1. Prepare the configuration resources and the test data for your ePayments3 test harness. Be sure to set up the proper test URL.

```
conf/[ResourceName]/CreditCardConfigData.xml
conf/[ResourceName]/CreditCardPaymentData.xml
conf/[ResourceName]/CheckPaymentData.xml
conf/[ResourceName]/CheckConfigData.xml
```

The [ResourceName] indicates the adapter type, be it PayPal43, OPSTP (official payment STP), or your customized adapter.

The value of the AdapterURL key is the test URL.

The configuration data are the XPOLICY table settings. For more information, see [XPOLICY Table Settings for ePayment Adapters](#).

2. Configure conf/[ResourceName]/SSOConfig.xml.
  - Set “Enable” to false unless you are testing an internal ePayments3 adapter on av.biz.
  - Set “Enable” to true if you are testing an internal ePayments3 adapter on av.biz, and the payment gateway need SSO authentication.
  - Set the user login parameters and the SSO URL.

### To run the Java test harness

1. Go to a command prompt.
2. Locate the test harness home directory. For example, go here:

```
d:\epg3\EPG3TestHarnessInJavaCXF
```

3. Run the program with the Resource you are testing. For example,

```
run: RunUT PayPal43
```

### To review the Java test results

1. Locate the reports directory. For example, go here:

```
d:\epg3\EPG3TestHarnessInJavaCXF\reports\junit-html
```

2. Open index.html in a Web browser.

### To remove temporary files for a new test

1. Run CleanProject.cmd.

## Implementing *makePayment* and *voidPayment* Calls

When a user pays by e-check or credit card in Civic Platform, Civic Platform can initiate the *makePayment* to fulfil the payment.

Civic Platform can only initiate the *voidPayment* when some errors occur. The following are two typical user cases.

### Use Cases for *makePayment* and *voidPayment* Calls

- Use case #1: Civic Platform calls the *makePayment* twice in a payment, the first one for handling the normal fee and the second one for the processing fee. If some error occurs in handling the second one (paying the processing fee), Civic Platform can call the *voidPayment* to void the first one (that pays the normal fee).
- Use case #2: When a user makes a payment with more than one payment method, Civic Platform calls *makePayment* for each method. If some error occurs in one call, Civic Platform calls *voidPayment* to roll back all the other successful calls, so to roll back the whole payment.

### Parameter Name-Value Pairs for *makePayment* Calls

The tables in the following section provide the parameter name-value pairs that can be sent with *makePayment* calls from Civic Platform.

### User Context

The *makePayment* calls always provide user context data. Adapter implementations can either ignore this data or record it as part of a log.

Table 1: User Context

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
AgencyID	A unique identifier for the agency, usually the name of the agency.	BridgeView	String	15

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
ModuleName	The module to which the current user belongs to and is making payment.	Building	String	15
UserID	The user ID of the user currently logged in and is making payment.	Mike	String	50
UserGroupID	The user group to which the current user belongs to and is making payment. For example, BuildingCashier.	Building Cashier	String	10
UserSessionID	The SSO auth ID for the Civic Platform session. For example, 12345678.	12345678	String	20

## Request Context

The *makePayment* calls always provide request context data. Adapter implementations can ignore this data, record it as part of a log, or use it to route payment processing requests.

Table 2: Request Context

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
RequestType	The type of records that the payment request comes from. Valid values are: RECORD, SET, POS, TRUSTACCOUNT, Transaction.	RECORD	String	-
RequestKey	The ID of the record that the payment request comes from. Valid Values are: Record ID, Set ID, POS Transaction ID, Trust Account ID, or Payment Processing Transaction ID.	09BLD-01234	String	-
RequestCapType	The record type if the payment request is from a record.	Building\Building \Construction \Residential\New	String	-

## General Payment Information

Civic Platform provides the general payment information to the adapter.

Table 3: General Payment Information

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
PaymentTransactionID	Unique transaction ID in Accela system, only available for makePayment.	4323	Number	20
PaymentAmount	The payment amount to be made. The value does not contain the courtesy fee.	100.00	Number	15,2
PaymentCourtesyFee	The amount of courtesy fee to be made.	0.00	Number	15,2
PaymentTotalAmount	The total amount of payment to be made, which is the sum of the PaymentAmount and PaymentCourtesyFee.	100.00	Number	15,2

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
PaymentCurrency	The currency used for the payment. The value must comply with the international standard of 3 characters.	USD	String	-
PaymentType	The type of the payment. Check = "EC" Credit Card = "CC"	CC	String	30
PaymentComment	Any comment from the user when making the payment. Same value as in the Comment field.	-	String	2000

## Credit Card Payment Information

Civic Platform provides the credit card payment information to the adapter if the payment type is set to "CC".

Table 4: Credit Card Payment Information

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
CreditCardType	Credit card type selected for the payment: Visa = "VI" MasterCard = "MC" American Express = "AX" Discover = "DC" Administrators can specify which credit card types are acceptable in Civic Platform.	VI	String	30
CreditCardNumber	For PCI Compliance, do not record this value or store it in a database.	4111111111111111	Number	28
CreditCardSecurityCode	Format: "mmyy". For PCI Compliance, do not record this value or store it in a database.	0412	Number	30
CreditCardExpiration	Format "mmyy". The value is from the expiration drop-down lists in UI.	0313	Date	-

## Check Payment Information

Civic Platform provides the check payment information to the adapter if the payment type is set to "EC".

Table 5: Check Payment Information

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
CheckType	Check type selected for the payment: "C" = Checking "S" = Savings "BC" = Business Checking "BS" = Business Savings You can configure Civic Platform to just specify Checking and Saving if you do not need to identify Business accounts.	C	String	-
CheckRoutingNumber	The check routing number users provide for payment. To keep data secure, do not record this value in the log or store it in a database.	123404231	Number	-

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
CheckAccountNumber	The check account number users provide for payment.To keep data secure, do not record the value in the log or store it in a database.	012323452	Number	-
CheckNumber	The check number users provide for payment.To keep data secure, do not record the value in the log or store it in a database.	1001	Number	7

## Payment Billing Information

Civic Platform can provide this information to the adapter for credit card and electronic check payment processing.

You must determine what billing information to provide to your ePayment gateway.

**Table 6: Payment Billing Information**

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
BillingBusinessName	The business name of the user who makes the payment.This parameter is only available for credit card payments.	Accela	String	65
BillingName	The full name of the user who makes the payment.	-	String	65
BillingFirstName	The first name of the user who makes the payment.	-	String	15
BillingMiddleName	The middle name of the user who makes the payment.	-	String	15
BillingLastName	The last name of the user who makes the payment.	-	String	25
BillingAddress	The address for billing.	-	String	100
BillingAddress2	The second line of address for billing.	-	String	40
BillingCity	The city of the user who makes the payment.	-	String	32
BillingState	The state or province of the user who makes the payment.	-	String	2
BillingZip	The Zip code or postal code of the user who makes the payment.	-	String	10
BillingPhone	The phone number of the user who makes the payment.	-	String	40
BillingPhoneCountry Code	The three character code indicating the phone country.This parameter is only available for credit card payments.	-	String	3
BillingEmail	The email address of the user who makes the payment.	-	String	70

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
BillingDriversLicense Nbr	The driver license number of the user who makes the payment. This parameter is only available for check payments.	-	String	100
BillingSocialSecurity Nbr	The social security number (SSN) of the user who makes the payment. This parameter is only available for check payments.	-	String	30

### Parameter Name-Value Pairs for *voidPayment* Calls

When Civic Platform calls *voidPayment* to void a successful payment, it gets the values of the parameters (AgencyID, UserID, PaymentType and TransCode) from the successful payment.

The *voidPayment* calls always provide user context data. Adapter implementations can either ignore this data or record it as part of a log.

Civic Platform also provides the transaction code and the payment type for processing the *voidpayment*.

Table 7: Payment Data for voidPayment Calls

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
AgencyID	A unique identifier for the agency, usually the name of the agency.	BridgeView	String	15
UserID	The user ID of the user currently logged-in and is making payment.	Mike	String	50
PaymentType	The type of the payment. Check = "EC" Credit Card = "CC"	CC	String	30
TransCode	The Transaction code that a <i>makePayment</i> call returns. If Civic Platform needs to void this transaction, send this value to the server.	V79B1BEEE310(PayPal)Or 00000000-0000-0000-0000-000000000000(VirtualMerchant)C	String	-

### Parameter Name-Value Pairs of Result Data

Table 8: Result Data Name Value Pair Definitions contains all the information that must be returned with every *makePayment* and *voidPayment*.

The EPaymentResult domain model must be the same for all ePayment Gateway 3 Adapters:

```
resultData ( [Name;Value] )
```

, returned to the paymentBusiness.


You need to specify valid data types and lengths.

You also need to create a map between these generic parameter names and the appropriate context data and payment data (credit card model, check model).

Table 8: Result Data Name Value Pair Definitions

Parameter Name	Description	Value
returnCode	Return 0 for success. Return any other value if the transaction failed. You must convert the result status from the EPayments Gateway the adapter communicates with into this return code format.	0
returnMap	Map of name value pairs.	



Parameter Name	Description	Value
AuthCode	Authorization code for a successful <i>makePayment</i> request. The payment gateway adapter defines the authorization code.	
TransCode	Unique transaction code, returned from all successful <i>makePayment</i> requests. This parameter identifies a transaction that needs to be voided through the <i>voidPayment</i> request. The payment gateway adapter defines the unique transaction code.	
returnMessage	<p>Transaction message extracted from GUI_TEXT based on the returnCode value. If you want to display a customized message to the user, create a new error code (making sure that it did not exist in GUI_TEXT), and then insert the customized message into GUI_TEXT with the error code. For example: 1) To specify a new error code PNP-002:</p> <pre>SELECT * FROM GUI_TEXT WHERE STRING_KEY = 'aca.creditCardPayment.resultcodePNP-002.label'</pre> <p>2) To insert your error message into GUI_TEXT.</p> <pre>VALUES ('aca.creditCardPayment.resultcodePNP-002.label','en', 'UserName is missed', 'ACA', GETDATE(), 'ADMIN','A', 'US');</pre> <p> <b>Note:</b> If you are using the PayPal Payflow Pro 4.3 gateway, you can create customized messages following the same way. If you use any other supported gateway, users can only view messages from the gateway adapter.</p>	

## Developing Your Payment Adapter with the .NET Sample Adapter

All the required .NET sample adapter and code are available in the \Sample Code \DotNetEPG3TestHarness folder and \Sample Code\SampleDotNetEPG3Adapter folder.

Topics:

- [Implementing Your Payment Adapter in .NET](#)
- [Deploying Your Payment Adapter to IIS](#)
- [Generating .NET Web Service Code Stub](#)
- [Configuring the .NET Test Harness](#)
- [Calling the ePayments3 Web Service](#)

### Implementing Your Payment Adapter in .NET

The web service implementation locates in the \SampleDotNetEPG3Adapter\App\_Code\Service.cs file. You can implement the *makePayment* and *voidPayment* methods based on your payment logic. Add your payment code between the following lines:

```
//Begin Implementation for EPayment
...
```

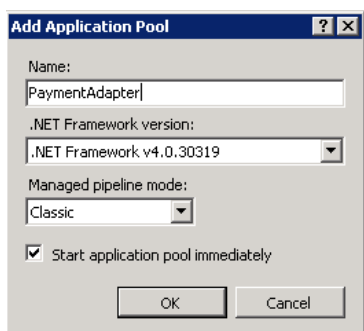
```
//End Implementation for EPayment
```

For more information, see [Implementing makePayment and voidPayment Calls](#).

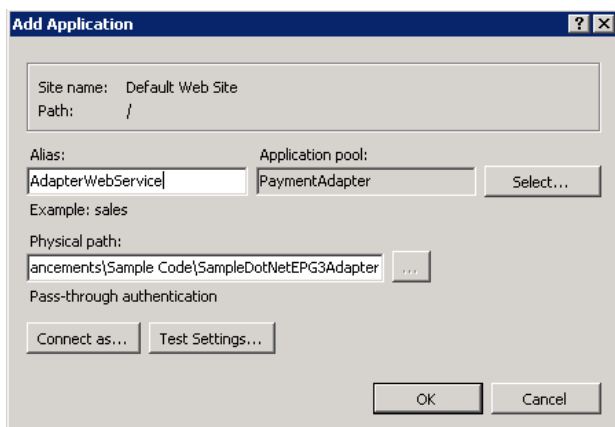
## Deploying Your Payment Adapter to IIS

### To deploy your payment adapter to IIS

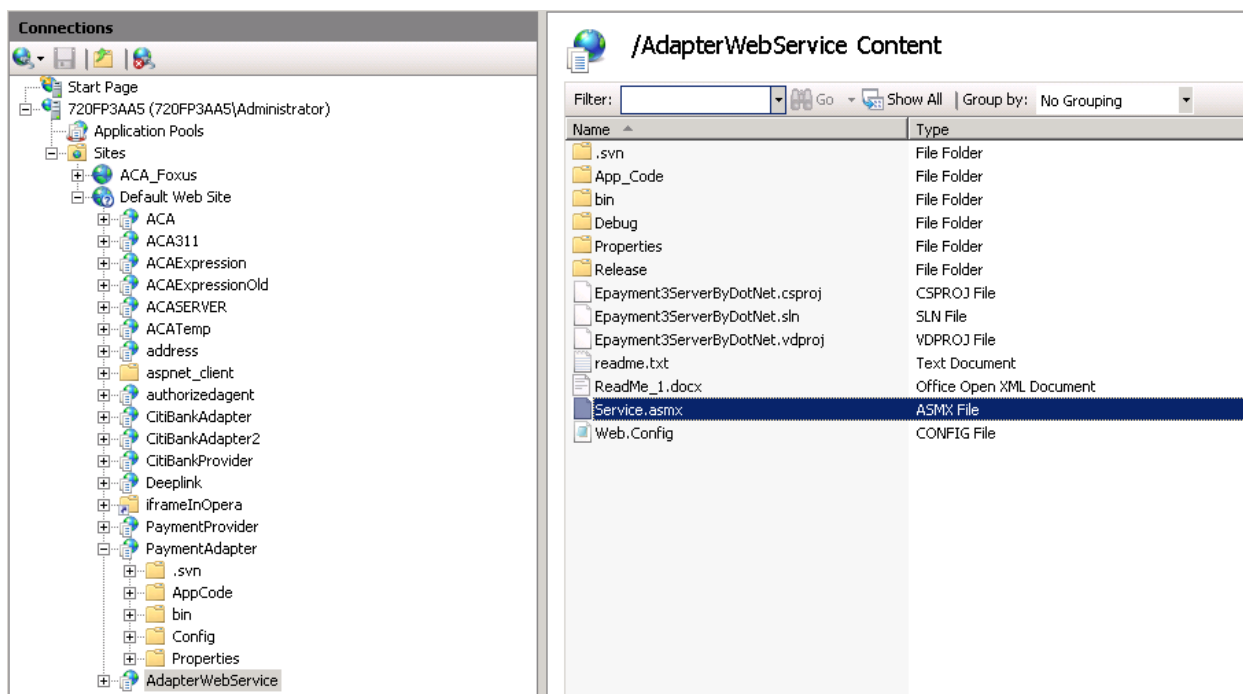
1. Open the IIS Manager.
2. Add an application pool for the application of your payment adapter.



3. Add a web application for the payment adapter. Note that you must map the physical path of the web application to the \Sample Code\SampleDotNetEPG3Adapter folder.

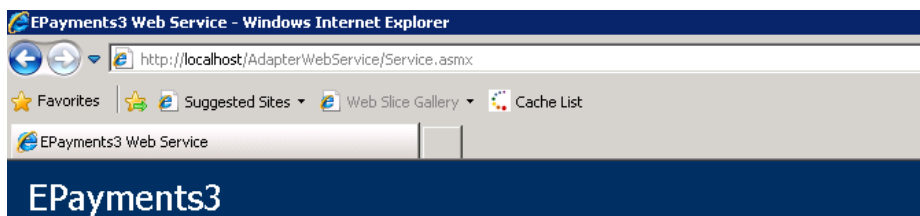


4. Test whether the deployment is successful.
  - a. Right-click the payment adapter application, and click **Switch to Content View**.



- b. Right-click the Service.asmx file in the Content panel and click **Browse**.

If the deployment is successful, the following web page displays.



The following operations are supported. For a formal definition, please review the [Service Description](#).

- [makePayment](#)
- [voidPayment](#)

## Generating .NET Web Service Code Stub

You must generate the .NET web service code stub manually.

### To generate .NET web service code stub manually

1. Run wsdl.exe.

```
wsdl /si http://localhost:8080/web2/EPayments3?wsdl
```

2. Edit EPayments3Interfaces.cs. Change the following code:

From:

```
[System.Web.Services.WebServiceBindingAttribute (Name="EPayments3SoapBinding", Namespace="http://service.ws.accela.com") ]
```

To:

```
[System.Web.Services.WebServiceBindingAttribute(Name =
"EPayments3Port", Namespace = "http://service.ws.accela.com")]
```

### 3. Edit the web service implementation. Do the following:

```
namespace WebService2
{
    [WebService(Namespace = "http://service.ws.accela.com", Name =
    "EPayments3")]
    //add namespace and web service name

    public class Service1: IEPayments3SoapBinding
    {
        #region IEPayments3SoapBinding Members
        public EPaymentResult makePayment(EPaymentInfo paymentInfo)
        {
            mapItem[] paymentDataField = paymentInfo.configData;
            EPaymentResult result = new EPaymentResult();
            result.returnCode = "0";
            result.returnMap = paymentDataField;
            return result;
        }
        public EPaymentResult voidPayment(EPaymentInfo paymentInfo)
        {
            throw new NotImplementedException();
        }
        #endregion
    }
}
}
```

## Configuring the .NET Test Harness

### To configure the .NET test harness

1. Open the following files located in \Sample Code\DotNetEPG3TestHarness, and update the configuration data for your test harness. Be sure to set up the proper test URL.

```
config/[ResourceName]/CreditCardConfigData.xml
config/[ResourceName]/CreditCardPaymentData.xml
config/[ResourceName]/CheckPaymentData.xml
config/[ResourceName]/CheckConfigData.xml
```

The [ResourceName] indicates the adapter type, be it PayPal43, OPSTP (official payment STP), or your customized adapter.

The value of the AdapterURL key is the test URL.

The configuration data are the XPOLICY table settings. For more information, see [XPOLICY Table Settings for ePayment Adapters](#).

2. Configure config/[ResourceName]/SSOConfig.xml.
  - Set "Enable" to false unless you are testing an internal ePayments3 adapter on av.biz.
  - Set "Enable" to true if you are testing an internal ePayments3 adapter on av.biz.
  - Set the user login parameters and the SSO URL.

## Calling the ePayments3 Web Service

### To call the ePayments3 Web service

1. Double-click the Epayment3ServerByDotNet.sln file in \Sample Code\SampleDotNetEPG3Adapter. Run the project in Visual Studio.
2. Open the ePayments3 test page.

The screenshot shows a web interface with three dropdown menus: AdapterType set to 'PayPay43', MethodType set to 'CreditCard', and Method set to 'Pay'. A 'Button' is visible next to the Method dropdown. Below the form, a blue highlighted text box displays the test result: "The web service return code is '0';The return message is 'Success';the return map is 11".

3. Specify the test parameters.

AdapterType	Select the web service adapter type. PayPay43 - If you want to use the PayPal43 adapter OPSTP - If you want to use the official payments STP adapter Customize - If you want to use your customized adapter
MethodType	Select the test payment method, credit card or check.
Method	Select the test method, pay or void.

4. Click **Button** to review the test result.

This screenshot is identical to the one above, showing the same configuration (AdapterType: PayPay43, MethodType: CreditCard, Method: Pay) and the test result: "The web service return code is '0';The return message is 'Success';the return map is 11".

## Developing Your Payment Adapter with the Java Sample Adapter

All the required Java sample adapter and code are available in the \Sample Code\JavaEPG3TestHarness folder and \Sample Code\SampleJavaEPG3Adapter folder.

### Topics:

- [Preparing the Environment](#)
- [Implementing Your Payment Adapter in Java](#)
- [Deploying the Sample Payment Adapter](#)
- [Configuring and Running the Java Test Harness](#)

### Preparing the Environment

Make sure you have installed the following software on the server where you want to deploy the payment adapter.

- Ant 1.7.0 or Ant 1.7.1. You can get it from <http://archive.apache.org/dist/ant/binaries> (apache-ant-1.7.1-bin.zip is recommended).

- Java JDK 5 or 6. You can get it from <http://www.oracle.com/technetwork/java/javase/downloads/jdk6-downloads-1637591.html>.
- Tomcat or JBoss. You can get tomcat and JBoss from the following paths.
  - <http://archive.apache.org/dist/tomcat/>
  - <http://www.jboss.org/jbossas/downloads/> (jboss-4.2.3.GA.zip is recommended.)

## Implementing Your Payment Adapter in Java

The web service implementation locates in the \SampleJavaEPG3Adapter\java\com\accele\ws\service\epayments3\EPayments3PayPallImpl.java file file. You can implement the makePayment and voidPayment methods based on your payment logic. Add your payment code between the following lines:

```
//Begin Implementation for EPayment
...
//End Implementation for EPayment
```

For more information, see [Implementing makePayment and voidPayment Calls](#).

## Deploying the Sample Payment Adapter

### To deploy the sample payment adapter

1. Open SetEnv.cmd in the SampleJavaEPG3Adapter folder using a text editor.
2. Set ANT\_HOME to the actual Apache Ant install directory.

For example:

```
set ANT_HOME=D:\apache-ant-1.7.1
```

3. Set JAVA\_HOME to the actual JDK install directory.

For example:

```
set JAVA_HOME=D:\jdk1.6.0_43
```

4. Save your changes.
5. Double-click the Build.cmd file in the SampleJavaEPG3Adapter folder.  
The file creates two subfolders: build and dist. The dist folder contains the payment web service package named EPaymentServer3ByCXF.war.
6. Deploy the web service using Tomcat or JBoss. For example, if you use Apache Tomcat, do the following:
  - a. Copy and paste EPaymentServer3ByCXF.war into the \$TOMCAT\_HOME/webapps/ folder. Replace \$TOMCAT\_HOME with the actual Apache Tomcat install directory.
  - b. Double-click the startup.bat file under the \$TOMCAT\_HOME/bin/ folder.
7. Enter the following URL in your browser.

```
http://{serverIp:port}/EPaymentServer3ByCXF/CustomEpaymentAdapter?wsdl
```

Replace

```
serverIp
```

with the actual name of the server to which you deployed the sample adapter. Replace `port` with the actual port number that is used for `Java.exe`.

If the `.wsdl` file opens properly in your browser, the web service is deployed successfully.

## Configuring and Running the Java Test Harness

The test harness runs from the command prompt. The test harness reads the configuration and the test files stored in the resource directory and then test all of the ePayments3 calls, including credit card payment, void credit card payment, make check payment, and void check payment.

The Java test harness works with any ePayments3 Web Service adapter and can also test ePayments3 adapters built into the Civic Platform application server.

### To configure the Java test harness

1. Prepare the configuration resources and the test data for your ePayments3 test harness. Be sure to set up the proper test URL.

```
conf/[ResourceName]/CreditCardConfigData.xml
conf/[ResourceName]/CreditCardPaymentData.xml
conf/[ResourceName]/CheckPaymentData.xml
conf/[ResourceName]/CheckConfigData.xml
```

The `[ResourceName]` indicates the adapter type, be it `PayPal43`, `OPSTP` (official payment STP), or your customized adapter.

The value of the `AdapterURL` key is the test URL.

The configuration data are the `XPOLICY` table settings. For more information, see [XPOLICY Table Settings for ePayment Adapters](#).

2. Configure `conf/[ResourceName]/SSOConfig.xml`.

- Set “Enable” to false unless you are testing an internal ePayments3 adapter on `av.biz`.
- Set “Enable” to true if you are testing an internal ePayments3 adapter on `av.biz`, and the payment gateway need SSO authentication.
- Set the user login parameters and the SSO URL.

### To run the Java test harness

1. Go to a command prompt.
2. Locate the test harness home directory. For example, go here:

```
d:\epg3\EPG3TestHarnessInJavaCXF
```

3. Run the program with the Resource you are testing. For example,

```
run: RunUT PayPal43
```

### To review the Java test results

1. Locate the reports directory. For example, go here:

```
d:\epg3\EPG3TestHarnessInJavaCXF\reports\junit-html
```

2. Open `index.html` in a Web browser.

### To remove temporary files for a new test

1. Run `CleanProject.cmd`.

## Implementing `makePayment` and `voidPayment` Calls

When a user pays by e-check or credit card in Civic Platform, Civic Platform can initiate the *makePayment* to fulfil the payment.

Civic Platform can only initiate the *voidPayment* when some errors occur. The following are two typical user cases.

### Topics:

- [Use Cases for `makePayment` and `voidPayment` Calls](#)
- [Parameter Name-Value Pairs for `makePayment` Calls](#)
- [Parameter Name-Value Pairs for `voidPayment` Calls](#)
- [Parameter Name-Value Pairs of Result Data](#)

### Use Cases for *makePayment* and *voidPayment* Calls

- Use case #1: Civic Platform calls the *makePayment* twice in a payment, the first one for handling the normal fee and the second one for the processing fee. If some error occurs in handling the second one (paying the processing fee), Civic Platform can call the *voidPayment* to void the first one (that pays the normal fee).
- Use case #2: When a user makes a payment with more than one payment method, Civic Platform calls *makePayment* for each method. If some error occurs in one call, Civic Platform calls *voidPayment* to roll back all the other successful calls, so to roll back the whole payment.

### Parameter Name-Value Pairs for *makePayment* Calls

The tables in the following section provide the parameter name-value pairs that can be sent with *makePayment* calls from Civic Platform.

#### Topics:

- [User Context](#)
- [Request Context](#)
- [General Payment Information](#)
- [Credit Card Payment Information](#)
- [Check Payment Information](#)
- [Payment Billing Information](#)

### User Context

The *makePayment* calls always provide user context data. Adapter implementations can either ignore this data or record it as part of a log.



Table 9: User Context

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
AgencyID	A unique identifier for the agency, usually the name of the agency.	BridgeView	String	15
ModuleName	The module to which the current user belongs to and is making payment.	Building	String	15
UserID	The user ID of the user currently logged in and is making payment.	Mike	String	50
UserGroupID	The user group to which the current user belongs to and is making payment. For example, BuildingCashier.	Building Cashier	String	10
UserSessionID	The SSO auth ID for the Civic Platform session. For example, 12345678.	12345678	String	20

## Request Context

The *makePayment* calls always provide request context data. Adapter implementations can ignore this data, record it as part of a log, or use it to route payment processing requests.

Table 10: Request Context

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
RequestType	The type of records that the payment request comes from. Valid values are: RECORD, SET, POS, TRUSTACCOUNT, Transaction.	RECORD	String	-
RequestKey	The ID of the record that the payment request comes from. Valid Values are: Record ID, Set ID, POS Transaction ID, Trust Account ID, or Payment Processing Transaction ID.	09BLD-01234	String	-
RequestCapType	The record type if the payment request is from a record.	Building\Building \Construction \Residential\New	String	-

## General Payment Information

Civic Platform provides the general payment information to the adapter.

Table 11: General Payment Information

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
PaymentTransactionID	Unique transaction ID in Accela system, only available for makePayment.	4323	Number	20
PaymentAmount	The payment amount to be made. The value does not contain the courtesy fee.	100.00	Number	15,2
PaymentCourtesyFee	The amount of courtesy fee to be made.	0.00	Number	15,2

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
PaymentTotalAmount	The total amount of payment to be made, which is the sum of the PaymentAmount and PaymentCourtesyFee.	100.00	Number	15,2
PaymentCurrency	The currency used for the payment. The value must comply with the international standard of 3 characters.	USD	String	-
PaymentType	The type of the payment. Check = "EC" Credit Card = "CC"	CC	String	30
PaymentComment	Any comment from the user when making the payment. Same value as in the Comment field.	-	String	2000

## Credit Card Payment Information

Civic Platform provides the credit card payment information to the adapter if the payment type is set to "CC".

Table 12: Credit Card Payment Information

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
CreditCardType	Credit card type selected for the payment: Visa = "VI" MasterCard = "MC" American Express = "AX" Discover = "DC" Administrators can specify which credit card types are acceptable in Civic Platform.	VI	String	30
CreditCardNumber	For PCI Compliance, do not record this value or store it in a database.	4111111111111111	Number	28
CreditCardSecurityCode	Format: "mmy". For PCI Compliance, do not record this value or store it in a database.	0412	Number	30
CreditCardExpiration	Format "mmy". The value is from the expiration drop-down lists in UI.	0313	Date	-

## Check Payment Information

Civic Platform provides the check payment information to the adapter if the payment type is set to "EC".

Table 13: Check Payment Information

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
CheckType	Check type selected for the payment: "C" = Checking "S" = Savings "BC" = Business Checking "BS" = Business Savings You can configure Civic Platform to just specify Checking and Saving if you do not need to identify Business accounts.	C	String	-

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
CheckRoutingNumber	The check routing number users provide for payment.To keep data secure, do not record this value in the log or store it in a database.	123404231	Number	-
CheckAccountNumber	The check account number users provide for payment.To keep data secure, do not record the value in the log or store it in a database.	012323452	Number	-
CheckNumber	The check number users provide for payment.To keep data secure, do not record the value in the log or store it in a database.	1001	Number	7

## Payment Billing Information

Civic Platform can provide this information to the adapter for credit card and electronic check payment processing.

You must determine what billing information to provide to your ePayment gateway.

**Table 14: Payment Billing Information**

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
BillingBusinessName	The business name of the user who makes the payment.This parameter is only available for credit card payments.	Accela	String	65
BillingName	The full name of the user who makes the payment.	-	String	65
BillingFirstName	The first name of the user who makes the payment.	-	String	15
BillingMiddleName	The middle name of the user who makes the payment.	-	String	15
BillingLastName	The last name of the user who makes the payment.	-	String	25
BillingAddress	The address for billing.	-	String	100
BillingAddress2	The second line of address for billing.	-	String	40
BillingCity	The city of the user who makes the payment.	-	String	32
BillingState	The state or province of the user who makes the payment.	-	String	2
BillingZip	The Zip code or postal code of the user who makes the payment.	-	String	10
BillingPhone	The phone number of the user who makes the payment.	-	String	40
BillingPhoneCountry Code	The three character code indicating the phone country.This parameter is only available for credit card payments.	-	String	3

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
BillingEmail	The email address of the user who makes the payment.	-	String	70
BillingDriversLicense Nbr	The driver license number of the user who makes the payment. This parameter is only available for check payments.	-	String	100
BillingSocialSecurity Nbr	The social security number (SSN) of the user who makes the payment. This parameter is only available for check payments.	-	String	30

### Parameter Name-Value Pairs for *voidPayment* Calls

When Civic Platform calls *voidPayment* to void a successful payment, it gets the values of the parameters (AgencyID, UserID, PaymentType and TransCode) from the successful payment.

The *voidPayment* calls always provide user context data. Adapter implementations can either ignore this data or record it as part of a log.

Civic Platform also provides the transaction code and the payment type for processing the *voidpayment*.

Table 15: Payment Data for *voidPayment* Calls

Parameter Name	Description	Example Value	Ref - Type	Ref - Length
AgencyID	A unique identifier for the agency, usually the name of the agency.	BridgeView	String	15
UserID	The user ID of the user currently logged-in and is making payment.	Mike	String	50
PaymentType	The type of the payment. Check = "EC" Credit Card = "CC"	CC	String	30
TransCode	The Transaction code that a <i>makePayment</i> call returns. If Civic Platform needs to void this transaction, send this value to the server.	V79B1BEEE310(PayPal)Or 00000000-0000-0000-0000-000000000000(VirtualMerchant)	-	-

### Parameter Name-Value Pairs of Result Data

The [Table 16: Result Data Name Value Pair Definitions](#) table contains all the information that must be returned with every *makePayment* and *voidPayment*.

The EPaymentResult domain model must be the same for all ePayment Gateway 3 Adapters:


```
resultData ([Name;Value])
```

, returned to the paymentBusiness.

You need to specify valid data types and lengths.

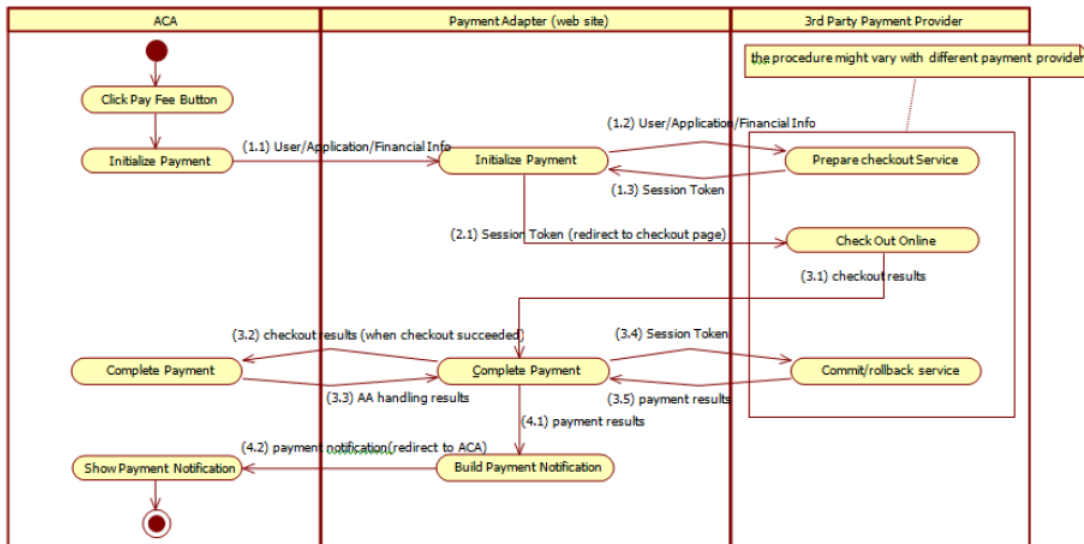
You also need to create a map between these generic parameter names and the appropriate context data and payment data (credit card model, check model).

Table 16: Result Data Name Value Pair Definitions

Parameter Name	Description	Value
returnCode	Return 0 for success. Return any other value if the transaction failed. You must convert the result status from the EPayments Gateway the adapter communicates with into this return code format.	0
returnMap	Map of name value pairs.	
AuthCode	Authorization code for a successful <i>makePayment</i> request. The payment gateway adapter defines the authorization code.	
TransCode	Unique transaction code, returned from all successful <i>makePayment</i> requests. This parameter identifies a transaction that needs to be voided through the <i>voidPayment</i> request. The payment gateway adapter defines the unique transaction code.	
returnMessage	<p>Transaction message extracted from GUI_TEXT based on the returnCode value. If you want to display a customized message to the user, create a new error code (making sure that it did not exist in GUI_TEXT), and then insert the customized message into GUI_TEXT with the error code. For example: 1) To specify a new error code PNP-002:</p> <pre>SELECT * FROM GUI_TEXT WHERE STRING_KEY = 'aca.creditCardPayment.resultcodePNP-002.label'</pre> <p>2) To insert your error message into GUI_TEXT.</p> <pre>VALUES('aca.creditCardPayment.resultcodePNP-002.label','en', 'UserName is missed', 'ACA', GETDATE(), 'ADMIN','A', 'US');</pre> <p> <b>Note:</b> If you are using the PayPal Payflow Pro 4.3 gateway, you can create customized messages following the same way. If you use any other supported gateway, users can only view messages from the gateway adapter.</p>	

## Developing Citizen Access Online Payment Adapter

The online payment adapter seamlessly connects Citizen Access and the third-party payment provider by exchanging HTTP messages between them.



**Figure 2: Interaction among ACA, Adapter and Third-Party Payment Provider**

**Table 17: Process Flow Narrative**

Stage	Step	Narrative
Initialize Payment	1.1	Citizen Access sends the payment initial parameters through the URL query string. For details about parameters and actions, see <a href="#">Preparing Initial Parameters</a> .
	1.2	Adapter interacts with the third-party payment provider to initiate the payment session. For details about the related parameters and actions, refer to the third-party payment provider API documents.
	1.3	The third-party payment provider returns a session token after initiating the payment session. For details about the related parameters and actions, refer to the third-party payment provider API documents.
	2.1	Adapter redirects users to the checkout page on the third-party payment provider to pay online. For details about the related parameters and actions, refer to the third-party payment provider API documents.
Complete Payment Status	3.1	After the payment succeeds, fails, or is cancelled, third-party payment provider sends the related information, such as payment status, selected payment method, inputted user information, payment type, etc., to the adapter. For details about the related parameters and actions, refer to the third-party payment provider API documents.
	3.2	If a user pays online successfully, adapter passes the payment status to Citizen Access to complete the payment. If a user cancels the payment or fail to pay online, adapter should skip this step, and go to the next step. If the third-party payment provider has API to check the authenticity of payment status, it should be done before this step. For details about the parameters and actions, see <a href="#">Handling Payment Status</a> .
	3.3	After Citizen Access completes payment process based on the payment status provided by the third-party payment provider, Citizen Access passes the handling results to the adapter. For details about the parameters and actions, see <a href="#">Handling Payment Status</a> .
	3.4	If the third-party payment provider has API to commit or rollback payment, adapter needs to commit or rollback payment based on the result of step 3.3. For details about the related parameters and actions, refer to the third-party payment provider API documents.

Stage	Step	Narrative
	3.5	The third-party payment provider returns the action results to adapter if Commit and Rollback commands are applicable. For details about the related parameters and actions, refer to the third-party payment provider API documents.
Send Payment Notification	4.1	Adapter gets payment results according to the results of step 3.3 or 3.5 (if applicable), and sends the results to the next step, 4.2. For details about the related parameters and actions, see <a href="#">Handling Payment Notification</a> .
	4.2	Adapter generates a notification that contains the payment status and the related message and passes the notification to Citizen Access. Citizen Access shows the payment results based on the returned information.

## Preparing Initial Parameters

Citizen Access automatically calls the HTTP method GET to pass the initial parameters to the online payment adapter (to the URL address defined in DATA2 of the XPOLICY table, see [XPOLICY for ACA Online Payment Adapter](#)).

**Table 18: Parameter Name-Value Pairs Initially Passed to Adapter from ACA** lists all the parameter names/values initially passed to adapter from Citizen Access. Note that the Ref-Type and Ref-Length requirements on the parameters are for your reference only, because you do not need to define the parameters.

**Table 18: Parameter Name-Value Pairs Initially Passed to Adapter from ACA**

Parameter Name	Description	Required/Optional	Ref - Type	Ref - Length
AgencyCode	Agency code. Adapter can use it to switch account settings for each agency.	Required	String	15
ApplicationID	Unique ID of the application that initiates the payment request, such as ACA. The default value is '1'.	Required	String	30
ConvFee	Convenience Fee. Value '0' indicates no convenience fee.	Required	Number	15,2
FullName	Full name of public user.	Optional	String	65
Lang	The selected language at user login.	Optional	String	20
PaymentAmount	Payment Amount, which does not contain convenience fee.	Required	Number	15,2
PostbackURL	Used to post back the payment results to Citizen Access in the completion payment stage.	Required	String	300
RedirectURL	Used to redirect user to Citizen Access when payment finishes (succeeds, being cancelled, or fails).	Required	String	300
TransactionID	Batch transaction ID. It is unique for each payment.	Required	String	30
UserID	The user ID of the user currently logged-in and is making payment.	Optional	String	50
UserEmail	The email of the public user who is making payment.	Optional	String	256

## Handling Payment Status

This section describes how to pass the payment status to Citizen Access from the third-party payment provider, and ACA response to the third-party payment provider.

Topics:

- [Passing Payment Status](#)
- [Passing Response](#)

### Passing Payment Status

The online payment adapter calls the HTTP method POST to pass the payment status to Citizen Access (to the PostbackURL address in the initiate-payment stage, see [Table 18: Parameter Name-Value Pairs Initially Passed to Adapter from ACA](#)).

[Table 19: Parameter Name-Value Pairs Indicating Payment Results from Adapter to ACA](#) lists all parameter name-value pairs that indicate payment results from the adapter to Citizen Access. Below is a .Net code snippet that demonstrates how to code the adapter to pass the payment status to Citizen Access.

```

/// <summary>
/// Post data to ACA and get the result, use this method to complete cap
  process
/// </summary>
/// <param name="responseUrl">the url to ACA</param>
/// <param name="postData">the data from third part web site</param>
/// <returns>the result from ACA, success or failed and the message</
returns>
public static string DoPostBack(string responseUrl, string postData)
{
  ASCIIEncoding encoding = new ASCIIEncoding();

  byte[] data = encoding.GetBytes(postData);

  // Prepare web request...
  HttpRequest myRequest = (HttpRequest)WebRequest.Create(responseUrl);

  myRequest.Method = "POST";
  myRequest.ContentType = "application/x-www-form-urlencoded";
  myRequest.ContentLength = data.Length;

  // Ignore Certificate Validation
  ServicePointManager.ServerCertificateValidationCallback = delegate(Object
  obj, X509Certificate certificate, X509Chain chain, SslPolicyErrors errors)
  { return (true); };

  using (Stream newStream = myRequest.GetRequestStream())
  {
    // Send the data.
    newStream.Write(data, 0, data.Length);
  }

  HttpResponse myResponse = null;
  string content = String.Empty;

  try
  {
    myResponse = (HttpResponse)myRequest.GetResponse();
  }
}

```



```

content = new StreamReader(myResponse.GetResponseStream(),
    Encoding.UTF8).ReadToEnd();
}
catch (Exception ex)
{
    _logger.Error("Error occurred when doing postback.", ex);
}
finally
{
    if (myResponse != null)
    {
        myResponse.Close();
    }
}

return content;
}

```

**Table 19: Parameter Name-Value Pairs Indicating Payment Results from Adapter to ACA**

Parameter Name	Description	Required/Optional	Ref - Type	Ref - Length
TransactionID	Batch transaction ID generated by Citizen Access. It is unique for each payment.	Required	String	50
PaymentAmount	Payment amount, which does not include convenience fee.	Required	String	15,2
ConvFee	Convenience Fee. If agency does not need to support convenience fee, pass '0' to this parameter.	Required	String	15,2
ProcTransactionID	The third-party payment transaction ID for the permit fee, license fee, or any other principle fee.	Required	String	50
ProcTransactionType	Check/Credit Card.	Optional	String	16
CCType	Credit card type, such as Visa, Discover, and MasterCard.	Optional	String	30
CCAuthCode	The credit card's Auth Code from the third-party payment provider.	Optional	String	32
CCNumber	Credit card number.	Optional	String	70
Payee	Payer full name.	Optional	String	150
PayeeAddress	Payer address.	Optional	String	240
PayeePhone	Payer phone.	Optional	String	240
PaymentComment	Payment comments.	Optional	String	2000

## Passing Response

Citizen Access forwards response to the third-party payment provider.

After Citizen Access completes the internal process based on the payment results passed from the third-party payment provider, Citizen Access responds to adapter with the handle result to decide whether to commit or rollback the third-party payment transaction.

HTTP methods: N/A (need to capture HTTP text output from Citizen Access.)

[Table 20: Parameter Name-Value Pairs Indicating Response Results from ACA to Adapter](#) below lists all parameter name-value pairs that indicate the response results from Citizen Access to adapter. Note that

the Ref-Type and Ref-Length requirements on the parameters are for your reference only, because you do not need to define the parameters.

Below is an example of the response message pattern with URL encoded value:  
 success=True&TransactionID=123456&UserMessage=message.

Table 20: Parameter Name-Value Pairs Indicating Response Results from ACA to Adapter

Parameter Name	Description	Required/Optional	Ref - Type	Ref - Length
Success	True: Succeed to complete the internal process.False: Fail to complete the internal process.Adapter needs to query, commit, or rollback (if applicable) the third-party payment transaction based on the result.	Required	String	-
TransactionID	Batch transaction ID. It is unique for each payment.	Required	String	-
UserMessage	Result message.	Required	String	-

## Handling Payment Notification

The online payment adapter calls the HTTP method GET to notify Citizen Access with the payment results (to the RedirectURL address in the initiate-payment stage, see [Table 18: Parameter Name-Value Pairs Initially Passed to Adapter from ACA](#)).

The following table lists all the parameter name-value pairs passed from adapter to Citizen Access through RedirectURL. Citizen Access passes the RedirectURL to the adapter in the Initiate-Payment stage.

Table 21: Parameter Name-Value Pairs Passed from Adapter to ACA via RedirectURL

Parameter Name	Description	Required/Optional	Ref - Type	Ref - Length
TransactionID	ACA transaction ID.	Required	String	20
ProcTransactionID	The third-party payment transaction ID.	Required	String	50
ReturnCode	Return code.1: success-1: fail 0: exit	Required	String	1
UserMessage	Payment result message.	Optional	String	2000

## Configuring ePayment Adapter

---

This section provides the configuration steps for built-in and customized ePayment adapters. For each ePayment adapter, The configuration steps apply to both testing and live ePayment adapters. You can first configure the testing adapter, and when you get ready to 'go live', switch to the live configuration.

### **Related Information**

[General Configuration Steps Testing and Live](#)

[Setting Payment Types for Govolution and First Data Adapters](#)

## General Configuration Steps Testing and Live

For every type of the ePayment adapters (either built-in or customized), Civic Platform provides the configuration script file for you to register the adapter in the database. All you need to do is to find the associated script file in the relevant \scripts folder that accompanies this document, update the configuration script file according to your environment, and then run the script in the database server.

### To configure an adapter

1. Locate the script file for the adapter in the \scripts folder.

All script files were zipped along with this document with folders named after the corresponding adapters. Each script file works for either the testing or live mode, with either Oracle or MS SQL database. For example, if you plan to configure the PayPal43 adapter for your testing environment that works with Oracle database, you can get the script file *EPaymentsConfig\_PayPal43\_Test\_oracle.sql* under the \scripts\PayPal43\oracle folder.

2. Make a copy of the script file and add your agency ID to the SQL file name.

For example, rename the script file *EPaymentsConfig\_PayPal43\_Test\_oracle.sql* to *Bridgeview\_EPaymentsConfig\_PayPal43\_Test\_oracle.sql*.

3. Modify the contents of the script to include the appropriate configuration parameters.

- a. Locate the @TODO lines in the script file that needs modification.

Below @TODO excerpt is from the *EPaymentsConfig\_PayPal43\_Test\_oracle.sql*.

```
---@TODO: replace me with SERV_PROV_CODE, e.g:'FLAGSTAFF'
AGENCY_ID      := 'FLAGSTAFF';
---@TODO: replace with proxy server configuration if agency is using a
proxy server for external internet access. Leave unchanged if not using
a proxy server.
PROXY_CONF
:= 'PROXYADDRESS=; PROXYPORT=; PROXYLOGON=; PROXYPASSWORD=';
---@TODO: replace with merchant account configuration
MERCHANT_CONF
:= 'PARTNER=PayPal; VENDOR=AccelaPayPal43TestCC; USER_NAME=AccelaPayPal43TestCC; PASS
---@TODO: replace with CountryCode=${CountryCode}
PAYMENT_CONF   := 'CountryCode=US';
```

- b. Modify the lines below the @TODO lines following the instructions in the @TODO lines. After the modification, remove the @TODO lines.

After the modification, the excerpt may look like this:

```
AGENCY_ID      := 'BridgeView';
PROXY_CONF     := 'PROXYADDRESS=; PROXYPORT=; PROXYLOGON=; PROXYPASSWORD=';
MERCHANT_CONF
:= 'PARTNER=PayPal; VENDOR=AccelaPayPal43TestCC; USER_NAME=Barbara; PASSWORD=TestPayF
PAYMENT_CONF   := 'CountryCode=US';
```

- c. Save the script file.



#### Note:

Each ePayment adapter requires different data that you need to modify in the script file. For details, see [Table 22: Data to Be Modified for Different Gateway Adapters](#).

4. Run the script.

**Note:**

The values you provide in the script file are updated to the XPOLICY table at the running of the script file. For the XPOLICY table settings of each ePayment adapter, see [XPOLICY Table Settings for ePayment Adapters](#).

Table 22: Data to Be Modified for Different Gateway Adapters

Gateway	ACA Only?	Data To be Modified under @TODO
CivicPay	Yes	Set the AGENCY_ID value to your Serv_Prov_Code. Set the MERCHANT_CONF value to include your SYSTEM_TOKEN, GATEWAY_NAME, APPLICATION_IDENTIFIER and AUTH_TOKEN (equivalent with the DATA4 value in the XPOLICY table). Set the GATEWAY_CONF value to your gateway host information (equivalent with the DATA2 value in the XPOLICY table).
PayPal Payflow Pro 4.3		Set the AGENCY_ID value to your Serv_Prov_Code.If you access the Internet through a proxy server then modify the PROXY_CONF with your proxy server configuration.Set the MERCHANT_CONF value to your gateway merchant login information (equivalent with the DATA3 value in the XPOLICY table).Set the PAYMENT_CONF value to include your country code (equivalent with the DATA4 value in the XPOLICY table).
OPSTP		Set the AGENCY_ID value to your Serv_Prov_Code.Set the MERCHANT_CONF value to your gateway merchant login information (equivalent with the DATA3 value in the XPOLICY table).Set the PAYMENT_CONF value to include your country code (equivalent with the DATA4 value in the XPOLICY table).
Virtual Merchant		Set the AGENCY_ID value to your Serv_Prov_Code.Set the MERCHANT_CONF value to your gateway merchant login information (equivalent with the DATA3 value in the XPOLICY table).Set the PAYMENT_CONF value to include your country code (equivalent with the DATA4 value in the XPOLICY table).
Official Payments CoBrand+	Yes	Set the AGENCY_ID value to your Serv_Prov_Code.Set the MERCHANT_CONF value to your gateway parameter information (equivalent with the DATA3 value in the XPOLICY table).Set the PAYMENT_CONF value to include your ProductID (equivalent with the DATA4 value in the XPOLICY table).
Govolution	Yes	Set the AGENCY_ID value to your Serv_Prov_Code.Set the MERCHANT_CONF value to include your ApplicationID and MessageVersion (equivalent with the DATA4 value in the XPOLICY table).
First Data	Yes	Set the AGENCY_ID value to your Serv_Prov_Code.Set the GATEWAY_CONF value to host gateway access configuration (equivalent with the DATA2 value in the XPOLICY table).Set the GATEWAY_PARAMETERS_CONF value to the gateway configuration information (equivalent with the DATA3 value in the XPOLICY table).
EPayments3		Set the AGENCY_ID value to your Serv_Prov_Code.Set the ADAPTER_NAME value to your custom adapter name testing or live (equivalent with the LEVEL_DATA value in the XPOLICY table).Set the ADAPTER_CONF value to Adapter=EPayments3;AdapterURL=\${URL of your Adapter}.Set the GATEWAY_CONF value to your gateway host access information (equivalent with the DATA2 value in the XPOLICY table).Set the MERCHANT_CONF value to your gateway merchant log in information (equivalent with the DATA3 value in the XPOLICY table).Set the PAYMENT_CONF value to your include your country code and, optionally, your amount and convenience fee codes (equivalent with the DATA4 value in the XPOLICY table).

Gateway	ACA Only?	Data To be Modified under @TODO
Custom Online Payment	Yes	Set the AGENCY_ID value to your Serv_Prov_Code. Set the ADAPTER_NAME value to your custom adapter name testing or live (equivalent to the LEVEL_DATA value in the XPOLICY table). Set the GATEWAY_CONF value to ACA Online Payment Adapter gateway host access information (equivalent to the DATA2 value in the XPOLICY table). Set the GATEWAY_URL_PARAMETERS value to the gateway's static URL parameters (equivalent to the DATA3 value in the XPOLICY table). Set the MERCHANT_CONF value to your application id (equivalent to the DATA 4 value in the XPOLICY table).

## Setting Payment Types for Govolution and First Data Adapters

---

Govolution and First Data adapters for Citizen Access only supports payments by credit card, but not electronic check. If you use the Govolution or First Data adapter, you must set the payment types in Citizen Access.

### To set the payment types in Citizen Access

1. Navigate to Citizen Access Setup in the setup portlet of Civic Platform Vantage360.
2. Choose a module from the navigation panel and go to the Pages area.
3. Select and customize the Payment Types in the Pay Fees page in the module-defined Apply folder.
  - Change the payment type Pay with Credit Card to Pay Online
  - Disable the two payment types: Pay with Trust Account, and the Pay with Bank Account
4. Save the changes and repeat the settings for each module.

## Standard Choices Configuration

---

This section provides the Standard Choices configuration required in the ePayment configuration. The Standard Choice names, values, and the value description are case-sensitive.

### Related Information

[Standard Choice EPaymentAdapter](#)

[Standard Choice PAYMENT\\_CHECK\\_ACCOUNT\\_TYPE](#)



## Standard Choice EPaymentAdapter

The Standard Choice EPaymentAdapter specifies which ePayment adapters Civic Platform and Citizen Access use respectively. See [Figure 3: Example of the Standard Choice EPaymentAdapter](#) .

### Standard Choices Item - Edit

Use this form to set up a Standard Choices Item.

Standard Choices Item Name: EPaymentAdapter

Description:  
(250 char max)

Status:  Enable  Disable

Type:  System Switch  Shared drop-down  EMSE  Business Configuration

Standard Choices Value	Value Desc	Active	
ACAAdapterType	MyAdapter_Test	<input checked="" type="checkbox"/>	Delete
AdapterType	MyAdapter_Test	<input checked="" type="checkbox"/>	Delete

**Figure 3: Example of the Standard Choice EPaymentAdapter**

In the Standard Choice, *ACAAdapterType* defines the ePayment adapter for Citizen Access, and *AdapterType* defines the ePayment adapter for Civic Platform. After you determine the ePayment adapter type, fill the **Value Desc** with the LEVEL\_DATA value from the XPOLICY table settings of the adapter. For a description of the XPOLICY table settings of the adapters, see [Configuring ePayment Adapter](#).

Table 23: Configuring the Standard Choice EPaymentAdapter

Accela Product	Value	Value Desc
Citizen Access	ACAAdapterType	The adapter name, which must be identical to the LEVEL_DATA value in the XPOLICY table of the adapter. Example: OPCoBrandPlus_Test
Civic Platform	AdapterType	The adapter name, which must be identical to the LEVEL_DATA value in the XPOLICY table of the adapter. Example: PayPal43_Test

## Standard Choice PAYMENT\_CHECK\_ACCOUNT\_TYPE

Administrators can configure the Standard Choice PAYMENT\_CHECK\_ACCOUNT\_TYPE to determine the account types and check types available when Citizen Access and Civic Platform access the customized Epayments3 adapter.

### Standard Choices Item - Edit

Use this form to set up a Standard Choices Item.

Standard Choices Item Name: PAYMENT\_CHECK\_ACCOUNT\_TYPE

Description:   
 (250 char max)

Status:  Enable  Disable

Type:  System Switch  Shared drop-down  EMSE  Business Configuration

Standard Choices Value	Value Desc	Active	
Business Checking	BC	<input checked="" type="checkbox"/>	Delete
Business Savings	BS	<input checked="" type="checkbox"/>	Delete
Checking	C	<input checked="" type="checkbox"/>	Delete
Savings	S	<input checked="" type="checkbox"/>	Delete

**Figure 4: Example of the Standard Choice PAYMENT\_CHECK\_ACCOUNT\_TYPE**

Table 24: Configuring the Standard Choice PAYMENT\_CHECK\_ACCOUNT\_TYPE

Value	Value Desc
Checking	C
Savings	S
Business Checking	BC
Business Savings	BS

## XPOLICY Table Settings for ePayment Adapters

---

The XPOLICY table in the Civic Platform database server stores the connection information and the configuration information that the ePayment adapter needs when it connects to the ePayment gateway.

This appendix lists the XPOLICY table settings after you successfully configure the ePayment Adapter by following the instructions in [General Configuration Steps Testing and Live](#).

In an XPOLICY table, all the values are case-sensitive. The LEVEL\_DATA and DATA2 reflect whether you are configuring the adapter for the testing or live sites. For example, PayPal43\_Test is for the testing site, and PayPal43\_Live is for the live site.

If you use ACA online payment adapter, you can pass static parameters to the adapter in the DATA3 column of the XPOLICY table. The static parameters can transfer additional information that the third party payment gateway requires, for example, the information about identifying the payer.

### Related Information

[XPOLICY for PayPal Payflow Pro4.3 Adapter](#)

[XPOLICY for Official Payments STP Adapter](#)

[XPOLICY for Virtual Merchant Adapter](#)

[XPOLICY for Official Payments CoBrand+ Adapter](#)

[XPOLICY for Govolution Adapter](#)

[XPOLICY for First Data Adapter](#)

[XPOLICY for ePayments3 Adapter](#)

[XPOLICY for MultipleAccountsEPayment Adapter](#)

[XPOLICY for ACA Online Payment Adapter](#)

[XPOLICY for CivicPay Adapter](#)

## XPOLICY for PayPal Payflow Pro4.3 Adapter

Table 25: XPOLICY Table Settings for PayPal Payflow Pro4.3 Testing and Live

XPOLICY Table	Data Values	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	PayPal43_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=EPayments3;AdapterURL=\${av.biz.url}/av-epayments3-adapters/PayPalPayflowPro43?wsdl	This is the configuration information about the adapter, including the adapter type and its URL. Enter this information exactly as specified.
DATA2	HOSTADDRESS=pilot-payflowpro.paypal.com;HOSTPORT=443;TIMEOUT=100;PROXYADDRESS=;PROXYPORT=;PROXYLOGON=;PROXYPASSWORD=	The data value is the configuration information about the gateway. Enter this information exactly as specified. Note that if you are using a proxy server, you must also provide the Proxy server configuration. For a test environment use HOSTADDRESS=pilot-payflowpro.paypal.com. For a live environment use HOSTADDRESS=payflowpro.paypal.com.
DATA3	PARTNER=\${partnername};VENDOR=\${vendorname};USER_NAME=\${myuser};PASSWORD=\${mypwd123};CHECKPARTNER=\${check-partnername};CHECKVENDOR=\${check-vendorname};CHECKUSER_NAME=\${check-username};CHECKPASSWORD=\${check-password}	The data value is the merchant information. You must enter the configuration that PayPal provides to access your account through the EPayments Gateway. PARTNER is the name of the company that you purchased PayPal access from. VENDOR is the name of your agency that you registered with the PARTNER. USERNAME and PASSWORD is your access information to log in to the gateway. You can also use CHECKPARTNER, CHECKVENDOR, CHECKUSER_NAME, and CHECKPASSWORD to configure an independent PayPal43 check payment merchant account.
DATA4	CountryCode=US	The data value is the currency your agency is using when processing online payments. With "CountryCode=US", the amounts are in US Dollars.

## XPOLICY for Official Payments STP Adapter

---

Table 26: XPOLICY Table Settings for Official Payments STP Testing and Live

XPOLICY Table	Data Value	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".

---

XPOLICY Table	Data Value	Description
LEVEL_DATA	OPSTP_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=EPayments3;AdapterURL=\${av.biz.url}/av-epayments3-adapters/OfficialPaymentsSTP?wsdl	This is the configuration information about the adapter, including the adapter type and its URL. Enter this information exactly as specified.
DATA2	HOSTADDRESS=https://staging.officialpayments.com/srv/stp	The data value is the configuration information about the gateway. Enter this information exactly as specified. For a test environment use HOSTADDRESS=https://staging.officialpayments.com/srv/stp. For a live environment use HOSTADDRESS=https://www.officialpayments.com/srv/stp.
DATA3	ClientID=\${ClientID};ProductID=\${ProductID};CustomFieldMap=cde-NotiNumb-0:\$\$RequestID\$\$,cde-UniqID-1:\$\$PaymentTransactionID\$\$,cde-Text-1:\$\$PaymentComment\$\$,cde-WateBill-0:\$\$UserID\$\$ Example: ClientID=123456789123456789; ProductID=123456789123456789; CustomFieldMap=cde-NotiNumb-0:\$\$RequestID\$\$,cde-UniqID-1:\$\$PaymentTransactionID\$\$,cde-Text-1:\$\$PaymentComment\$\$,cde-WateBill-0:\$\$UserID\$\$'	The data value is the merchant information. You must enter the configuration that Official Payments provides to access your account through the EPayments Gateway. ClientID is the Client ID. ProductID is the product ID of your merchant account.
DATA4	CountryCode=US; TotalAmountFormula=1,1.0295,0,1,80,120,1,1.03,0,1,999999,999999; ConvenienceFeeFormula=0.0295,1.0295,0,1,80,120,0.03,1.03,0,1,999999,99999999 The two formulas are defined as: A1, B1, C1, Min, Max, R1, A2, B2, C2, Min, Max, R2, ... Fee = Amount * Ax/Bx + Cx Example: Charged fee is 90; Convenience Fee is 3% of Charged Fee, that is 3.04; Total Payment Amount = Charged Fee + Convenience Fee = 90 + 3.04 = 93.04 Then ConvenienceFeeFormula can be configured as 0.035, 1.035, 0, 1, 80, 120, 0.035, 1.035, 0, 1, 9999999, 999999 (A1=0.035; B1=1.035; C1=0; Min=1; Max=80; R1=120; A2=0.035; B2=1.035; C2=0; Min=1; Max=9999999; R2=999999) Because charged fee < R1=120, so ConvenienceFee=90*0.035/1.035 + 0=90*3.4%=3.06 TotalAmountFormula can be configured as 1, 1, 0, 80, 120, 1, 1, 0, 1, 9999999, 999999; (A1=1; B1=1; C1=0; Min=1;	The data value is the currency that your agency is using when processing online payments. With "CountryCode=US", the amounts are in US Dollars. The data value also contains the formulas for calculating the Total Amount and Convenience fees for each transaction. The Example Formulas below are for a 3% convenience fee. The formulas are calculated like other fee formulas in Civic Platform.

## XPOLICY for Virtual Merchant Adapter

Table 27: XPOLICY Table Settings for Virtual Merchant Testing and Live

XPOLICY Table	Data Value	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	VirtualMerchant_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=EPayments3;AdapterURL=\${av.biz.url}/av-epayments3-adapters/VirtualMerchant?wsdl	The data value is the configuration information about the adapter, including the adapter type and its URL. Enter this information exactly as specified.
DATA2	VIRTUALMERCHANT_URL=https://www.myvirtualmerchant.com/VirtualMerchant/process.do;SSL_TEST_MODE=TRUE	The data value defines the host gateway access configuration. Enter this information exactly as specified. The Virtual Merchant adapter does not support proxy servers. For a test environment use SSL_TEST_MODE=TRUE. For a live environment use SSL_TEST_MODE=FALSE.
DATA3	ssl_merchant_id=000170; ssl_user_id=accelatest; ssl_pin=7M2XP3	The data value is the merchant configuration. You must enter the configuration that Virtual Merchant provides to access your account through the EPayments Gateway. ssl_merchant_id is your Merchant ID. ssl_user_id is the User ID, and ssl_pin is your User Password.
DATA4	CountryCode=US	The data value is the currency that your agency is using when processing online payments. With "CountryCode=US", the amounts are in US Dollars.

## XPOLICY for Official Payments CoBrand+ Adapter

---

Table 28: XPOLICY Table Settings for Official Payments CoBrand+ Testing and Live

XPOLICY Table	Data Values	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".

---



XPOLICY Table	Data Values	Description
LEVEL_DATA	OPCoBrandPlus_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=Redirect	The data value is the configuration information about the adapter. No URL is needed for redirect.
DATA2	HostURL=https:// staging.officialpayments.com/ pc_entry_cobrand.jsp ECheckHostURL=https:// staging.officialpayments.com/ echeck/pc_entry_cobrand.jsp	The data value defines the host gateway access configuration. Enter this information exactly as specified. For a test environment use the test environment gateway host access configuration, HostURL=https://staging.officialpayments.com/pc_entry_cobrand.jsp. For a live environment use the live environment host access configuration, HostURL=https://www.officialpayments.com/pc_entry_cobrand.jsp. For a test environment that includes the ECheck payment option, use the test environment host access configuration ECheckHostURL=https://staging.officialpayments.com/echeck/pc_entry_cobrand.jsp. For a live environment that includes the E-Check payment option, use the live environment host access configuration ECheckHostURL= https://www.officialpayments.com/echeck/pc_entry_cobrand.jsp.
DATA3	<b>RequestParameterMap</b> =productId: \$\$ProductId\$\$, address1:\$ \$Address1\$\$, cde-NotiNumb-0: \$\$CAPID\$\$, cde-UniqID-1:\$ \$UNIQUEID\$\$, cityName:\$\$City\$\$, email:\$ \$Email\$\$, firstName:\$\$FirstName \$\$, lastName:\$\$LastName\$ \$, middleName:\$\$MiddleName\$\$, paymentAmount:\$\$PaymentAmount \$\$, phoneNum:\$\$PhoneNum\$ \$, postalCd:\$\$Zip\$\$, provinceCd: \$\$State\$\$, cde-NumbOf-2:\$ \$NumberItems\$\$; <b>PostbackParameterMap</b> =uid: \$\$UNIQUEID\$\$, account_type: \$\$AccountType\$\$, address1: \$\$Address1\$\$, address2:\$ \$Address2\$\$, address3:\$\$Address3\$ \$, confirmation:\$\$Confirmation \$\$, email:\$\$Email\$\$, firstName: \$\$FirstName\$\$, lastName:\$ \$LastName\$\$, middleName:\$\$MiddleName\$ \$, suffix:\$\$Suffix\$\$, telephone:\$ \$PhoneNum\$\$	The data value is the gateway configuration. You must enter the mapping data RequestParameterMap and PostbackParameterMap. This is always customized to include the unique ID parameters.
DATA4	ProductID=\${ProductID} Example: ProductID= 0123456789012345678901234567890123456 Note: Official Payments CoBrand + does not require Client ID.	The data value is your agency's ProductID. Official Payments CoBrand+ does not use Client ID.

## XPOLICY for Govolution Adapter

Table 29: XPOLICY Table Settings for Govolution Testing and Live

XPOLICY Table	Data Value	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	Govolution_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=Redirect	The data value is the configuration information about the adapter. No URL is needed for Redirect. Enter this information exactly as specified.
DATA2	HostURL=https:// demo.velocitypayment.com/ vrelay/verify.do	The data value defines the host gateway access configuration. Enter this information exactly as specified in the table below. For a test environment use the test environment gateway host access configuration, HostURL= HostURL=https://demo.velocitypayment.com/vrelay/verify.do. For a live environment use the live environment host access configuration, HostURL=https://www.velocitypayment.com/vrelay/verify.do.
DATA3	<i>Not Used. Reserved for parametermapping.</i>	<i>Not Used. Reserved for parametermapping.</i>
DATA4	ApplicationID=1234; MessageVersion=2.2	The data value is your agency's ApplicationID and MessageVersion configuration for Govolution.

## XPOLICY for First Data Adapter

Table 30: XPOLICY Table Settings for First Data Testing and Live

XPOLICY Table	Data Value	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	FirstData_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=Redirect	The data value is the Adapter Type. No URL is needed for Redirect. Enter this information exactly as specified in the table below.
DATA2	HostURL= https://xx.com/epayconsumerweb/oh/col/building/default.aspx	The data value defines the host gateway access configuration. Enter this information exactly as specified in the table below. For a test environment use the test environment gateway host access configuration, HostURL=https://xx.com/epayconsumerweb/oh/col/building/default.aspx. Replace xx.com with the correct host name. For a live environment use the live environment host access configuration, HostURL=https://xx.com/epayconsumerweb/oh/col/building/default.aspx. Replace xx.com with the correct host name.
DATA3	RequestParameterMap=returnurl: \$\$ReturnURL\$\$, amount: \$\$PaymentAmount\$\$, id: \$TransactionID\$\$, ref: \$TransactionID\$\$	The data value is the gateway configuration. You must enter the mapping data for the parameters sent to First Data.
DATA4	<i>Not used. Reserved for Merchant Configuration.</i>	<i>Not used. Reserved for Merchant Configuration.</i>

## XPOLICY for ePayments3 Adapter

Table 31: XPOLICY Table Settings for EPayments3 Testing and Live

XPOLICY Table	Data Values	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	MyAdapter_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter=EPayments3;AdapterURL=\${MyAdapterURL}/MyEPaymentsAdapter?wsdl	The data value is the Adapter Type and its URL. Enter this information exactly as specified.
DATA2	\${Gateway Host Access -- custom to the adapter/gateway}	The data value defines the host gateway access configuration. Enter this information exactly as specified. For a test environment use the test environment gateway host access configuration. For a live environment use the live environment host access configuration.
DATA3	\${Gateway Merchant Account -- custom to the adapter/gateway}	The data value is the gateway configuration. You must enter the configuration that your Gateway Host provides to access your account through the EPayments Gateway. The data value is custom to your EPayments3 Adapter.
DATA4	CountryCode=US; TotalAmountFormula=1,1.0295,0,1,80, 120,1,1.03,0,1,999999,99999; ConvenienceFeeFormula=0.0295, 1.0295,0, 1,80,120,0.03,1.03,0,1, 999999,99999999	The data value specifies the currency that your agency is using when processing online payments. The data value also contains the formulas for calculating the Total Amount and Convenience fees for each transaction it processes. The Example Formulas below are for a 3% convenience fee. The formulas are calculated like other fee formulas in Civic Platform. Note: the TotalAmountFormula and ConvenienceFeeFormula are optional and should only be configured if your Custom EPayments Adapter uses convenience fees.

## XPOLICY for MultipleAccountsEPayment Adapter

Use the MultipleAccountsEPayment adapter type if an agency or department needs to use different merchant accounts for item fees and convenience fees. Specify the merchant account information and convenience fee formula on Civic Platform V360 Administration Tool > Finance > Merchant Account Settings.

Table 32: XPOLICY Table Settings for MultipleAccountsEPayment Testing and Live

XPOLICY Table	Data Values	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	{MyAdapter}_Test	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	<pre>Adapter=MultipleAccountsEPayment; AdapterURL=\${MyAdapterURL}/ {MyEPaymentsAdapter}?wsdl</pre>	<p>The data value is the Adapter Type and the ePayment adapter URL. Enter this information exactly as specified, replacing MyAdapterURL and MyEPaymentsAdapter with the appropriate names. For example:</p> <pre>Adapter=MultipleAccountsEPayment; AdapterURL=\${av.biz.url}/ av-epayments3-adapters/ Citibank?wsdl</pre>
DATA2	<pre>HOSTADDRESS=\${Gateway Host Access -- custom to the adapter/gateway}</pre>	<p>The data value defines the host gateway access configuration. Enter this information exactly as specified. For a test environment use the test environment gateway host access configuration. For a live environment use the live environment host access configuration. For example:</p> <pre>HOSTADDRESS=https:// paydirectapi.ca.link2gov.com</pre>
DATA3	NULL	This field is ignored. The merchant accounts for item fees and convenience fees are defined on Civic Platform V360 Admin Tool > Finance > Merchant Account Setting > Merchant Account Details. Note that "Principle Fee" on the Merchant Account Details page refers to the payment item fee.
DATA4	NULL	This field is ignored. The convenience fee formula is defined on Civic Platform V360 Admin Tool > Finance > Merchant Account Setting > Convenience Fee Formula.

## XPOLICY for ACA Online Payment Adapter

Table 33: XPOLICY Table Settings for ACA Online Payment Adapter

Field Name	Field Value	Description
SERV_PROV_CODE	\$Variable	Replace \$Variable with the actual Service provider code (the login user's agency code). For example, Montana.
POLICY_SEQ	\$Variable	Replace \$Variable with the actual table. Sequence number.
POLICY_NAME	PaymentAdapterSec	Keep the value as "PaymentAdapterSec".
LEVEL_TYPE	Adapter	Keep the value as "Adapter".
LEVEL_DATA	\$Variable	Replace \$Variable with the adapter name. The Standard Choice EPaymentAdapter uses this value to identify the adapter.
RIGHT_GRANTED	F	Keep the value as "F".
STATUS	A	Keep the value as "A".
DATA1	Adapter=Redirect	The data value is the configuration information about the adapter. No URL is needed for Redirect. Enter this information exactly as specified.
DATA2	HostURL=\$Variable	Replace \$Variable with the value of adapter initiate payment handler URL. For example, HostURL=http://aca-server.achievo.com/TPEPayment/BeginPayment.aspx.
DATA3	paraName1=paraValue1; paraName2=paraValue2	If necessary, you can specify static values in paraName1 and paraName2 for passing to the payment adapter. The static values can be the additional information that the third party payment gateway requires, for example, the information about identifying the payer.
DATA4	ApplicationID= \$ApplicantID; ConvenienceFeeFormula= \$ConvenienceFeeFormula	Application ID. The \$ApplicationID should be replaced with '1' by default. If there are multiple client systems (ACA) that need to share the adapter system, assign a unique application ID for each client system. If convenience fee needs to be calculated in Civic Platform, the convenience fee formula should be specified. The convenience fee is disabled by default. Replace \$ConvenienceFeeFormula with actual value.
DATA5	NULL	Not Used.
MENU_LEVEL	NULL	Not Used.
MENUITEM_CODE	NULL	Not Used.
REC_STATUS	A	Keep the value as "A".
REC_FUL_NAM	Admin	Keep the value as "Admin".
REC_DATE	Current Date	The data value is the current date when you configure the adapter by running the DB script.
RES_ID	0	Keep the value as "0".

## XPOLICY for CivicPay Adapter


Table 34: XPOLICY Table Settings for CivicPay Testing and Live

XPOLICY Table	Data Value	Description
LEVEL_TYPE	Adapter	Keep the value as "Adapter"
LEVEL_DATA	CivicPay_Test or CivicPay	The Standard Choice EPaymentAdapter uses this value to identify the adapter.
DATA1	Adapter = Redirect	The data value is the configuration information about the adapter. No URL is needed for Redirect. Enter this information exactly as specified.
DATA2	HOST_URL= \${Gateway Host Access}	The data value defines the host gateway access configuration. Enter this information exactly as specified. For a test environment use the test environment gateway host access configuration. For a live environment use the live environment host access configuration. For example: HOST_URL= https://civicpayintegrationpayments.azurewebsites.net/api
DATA3	Not Used. Reserved for parameter mapping.	Not Used. Reserved for parameter mapping.
DATA4	SYSTEM_TOKEN=917CE049-2D97-40AB-9159-9CAE82924B1D; PAYMENT_URL=/Payments/ IncludeNonPCIFlow; AGENCY_URL=/AgencyDetails; GATEWAY_NAME=Bluefin; APPLICATION_IDENTIFIER=VajzcuE4; AUTH_TOKEN=ACA-AuthToken	The data value is SYSTEM_TOKEN, GATEWAY_NAME, APPLICATION_IDENTIFIER and AUTH_TOKEN for ACA application.

## Terms and Definitions

Table 35: Terms and Definitions provides definitions of common terms used in this guide.

Table 35: Terms and Definitions

Term	Definition
Browser Side Redirect Payment Solution	A hosted payment gateway solution that processes ePayment by sending the customer to a secure external payment website to accept sensitive payment details such as credit card numbers.
Convenience fee	A service fee paid as a cost of using the third-party payment gateway. For example, Official Payments STP and Official Payments CoBrand+ both charge convenience fees to a user for each transaction.
Electronic Payments	Online processing where users make payments using credit cards or checks.
ePayment	An abbreviation of "Electronic Payment".
ePayment Adapter	An interface connecting Civic Platform and the third-party payment system to complete payment transactions. Civic Platform supports two types of adapters, Server Side Gateway Payment adapter and Browser Side Redirect Payment adapter.
EPayments3	The current standard Web service definition used to implement the ePayment solutions in Civic Platform. The WSDL file name is epayments3.wsdl.
Merchant Account	The bank account agencies use to receive money from electronic payments.
Server-Side Gateway	An API that serves as the server-side gateway to a payment processing provider where payment information is sent for payment confirmation.   <b>Note:</b> Many ePayment Gateway providers can support merchant accounts from several different banks.
Standard Choice EPaymentAdapter	A place where you specify and enable the EPayments Adapter for Civic Platform and Citizen Access. For example, Adapter=PayPal43_Test and ACAAdapter=PayPal43_Test.
XPOLICY	The database table where all EPayments configuration data is stored. The table applies to both Civic Platform and Citizen Access.