worldpay

Intercept eCommerce with Token Management Service Integration Guide

Intercept eCommerce with Token Management Service Integration Guide

Contents

Contents

Contents

1 Version history

Version	Description
20250206-1	Added PCI v4.0 A compliance 85
20240731-1	• Updated Integration and overview planning 10 • Updated Display content based on the selected payment type 59 • Changed *.paymetric.com to *.worldpay.com • Changed the Cert URL: https://qaapp02.xisecurenet.com to https://cert-xiecomm.worldpay.com/diecomm • Changed the Prod URL: https://prdapp02.xisecurenet.com to https://xiecomm.worldpay.com/diecomm
20201005-1	• Updated Onboarding 9 • Added eProtect to Field requirements per token provider 13
20200813-1	Modified the 3DS token 75 and XML Samples to include Token and TokenProviders.
20200519-1	Modified Get Access Token Packet XML descriptions for Form Post 18, Ajax 29, Iframe 43, and Hosted 63 integrations to clarify TokenProvider values are configured by the merchant in merchant Portal.
20200326-1	First GA version of the document.

2 Introduction

Intercept for eCommerce (Intercept eComm) utilizes XML packets to submit sensitive data directly to our SOC 1 compliant data center where it is encrypted and securely stored. A token is returned for use by the merchant's application; thereby, removing sensitive data from the merchant's environment.

Purpose

The purpose of the document is to provide an how to integrate to Intercept eComm for all supported token types through the Token Management Service.

Audience

This document is intended for use by Web Developers and assumes a familiarity with XML and XSD.

3 Compatibility

Browser compatibility

Intercept eComm is compatible with the following browsers:

- Internet Explorer 10.0 and above
- Firefox (latest version)
- Chrome (latest version)
- Safari (latest version)

Mobile device support

Intercept eComm is compatible with both IOS and Android devices. Use the same integration steps as documented in this guide for these devices.

4 Onboarding

After Paymetric enables the services on the **Services** tab and the token providers on the **TES** tab in the merchant's profile wizard, options display under the **Settings** menu for Intercept for eCommerce and Token Exchange Service to configure the components to use Token Management Service.

Onboarding for the following integration types requires creating and configuring TES providers and then enabling Intercept for eCommerce for TES:

- Form Post
- Ajax
- Iframe
- Hosted
- SecureLink

If you are implementing 3D Secure with Intercept eComm, you must obtain the following from Cardinal Commerce before you can submit your Intercept eComm environment requests.

- Cardinal Commerce Merchant ID
- Processor ID
- Password

For more information about the onboarding process for the Token Management Service, see the following topics in Paymetric Assist:

- Configure TES account
- Configure TES profile

4 Onboarding

- Submit TES onboarding request
- Configure Intercept eCommerce for TES

4.1 Integration overview and planning

Review this section to determine which type(s) are applicable for your implementation.

Form Post, Ajax, Iframe and Hosted are the Intercept eComm "Standard Integration Types" used to tokenize directly-entered information (i.e. not from some sort of wallet functionality), such as credit cards, account numbers for echeck payments, and other non-payment related sensitive data like Social Security Numbers.

These standard integration types all use the same Intercept eComm endpoint addresses to trigger each of the calls.

Form Post	• A POST request is made to Intercept eComm from the client's browser. • Intercept eComm redirects to the URL defined by the merchant; additional fields aside from what is being tokenized can be sent in the XML packet. • The page has a high-level of page responsiveness since it is completely hosted by the merchant. • Merchant is responsible for all card validations.
Ajax	• An Ajax call to Intercept eComm from the client's browser. • User never leaves page.

	• The page has a high-level of page responsiveness since it is completely hosted by the merchant. • Merchant is responsible for all card validations.
Iframe	• Paymetric hosts the content of the Iframe. • Merchant can style and control what fields display; there are multiple content templates or the merchant can completely customize. • The page has a potential delay while loading the Iframe; the merchant can hide all contents of page while Iframe loads.
Hosted	• Paymetric hosts the entire page. • When initiating payment, the merchant's customer is redirected to the Paymetric URL. • Intercept eComm redirects to the URL defined by the merchant after processing the request. • The page has a high-level of page responsiveness since it is completely hosted.

⚠ Note:

When implementing the Form Post or Ajax integration type, do NOT post or send raw card numbers in your environment.

4 Onboarding

Securelink integration

- Used to trigger an email with a secure link to Intercept eComm hosted page to obtain credit card information. Can alternatively choose to copy/paste the link into a chat window.
- Paymetric hosts the entire page.
- Hosted page is launched via a link sent in an email to the specified address or from a chat window.
- Intercept eComm redirects to the URL defined by the merchant after processing the request.

4.2 Authentication

The identify of the system making calls to the Intercept eComm platform is authenticated using the following security elements and URL endpoints:

Security element	Description
Merchant GUID	A unique ID assigned by Paymetric that is included in the Request packet. Each merchant has a single GUID.
Shared Key	A string generated and provided used to "sign" the Request Packet. Paymetric uses the same shared key to confirm the signature.

Security element	Description
Merchant GUID	A unique ID assigned by Paymetric that is included in the Request packet. Each merchant has a single GUID.
Access Token	A string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key. The returned AccessToken is then used in all subsequent calls within the given session. The AccessToken is valid for 30 minutes.

4.3 Field requirements per token provider

Each supported token provider has varying minimum field requirements for the tokenization call. You can implement one or more token providers. Ensure to include the fields that meet the minimum requirements for all providers that you are implementing if more than one.

Field names are case-sensitive.

XiSecure minimum field requirements

Credit card fields

Field	Description	Type	Size
CardNumber	Credit card number	string	50

eCheck fields

Field	Description	Type	Size
AccountNumber	Check account number	string	25

Vantiv OmniToken minimum field requirements

Credit card fields

Field	Description	Type	Size
CardNumber	Credit card number	string	50

eCheck fields

Field	Description	Type	Size
AccountType	String defining the type of account. Valid values are: “personal checking”, “personal savings”, “business checking” and “business savings”	string	50
RoutingNumber	Check routing number	string	25
AccountNumber	Check account number	string	25

Worldpay STS minimum field requirements

Credit card fields

Fieldname	Description	Type	Size
CardNumber	Credit card number	string	50
Name	Name associated with the address	string	50
ExpirationDate	MM/YY Formatted expiration date	string	5

eCheck fields

Not supported

eProtect minimum field requirements

Credit card fields

Fieldname	Description	Type	Size
CardNumber	Credit card number	string	50

eCheck fields

Not supported

5 TMS form post integration

> **Note:**
>
> Do not send raw card numbers in your environment.

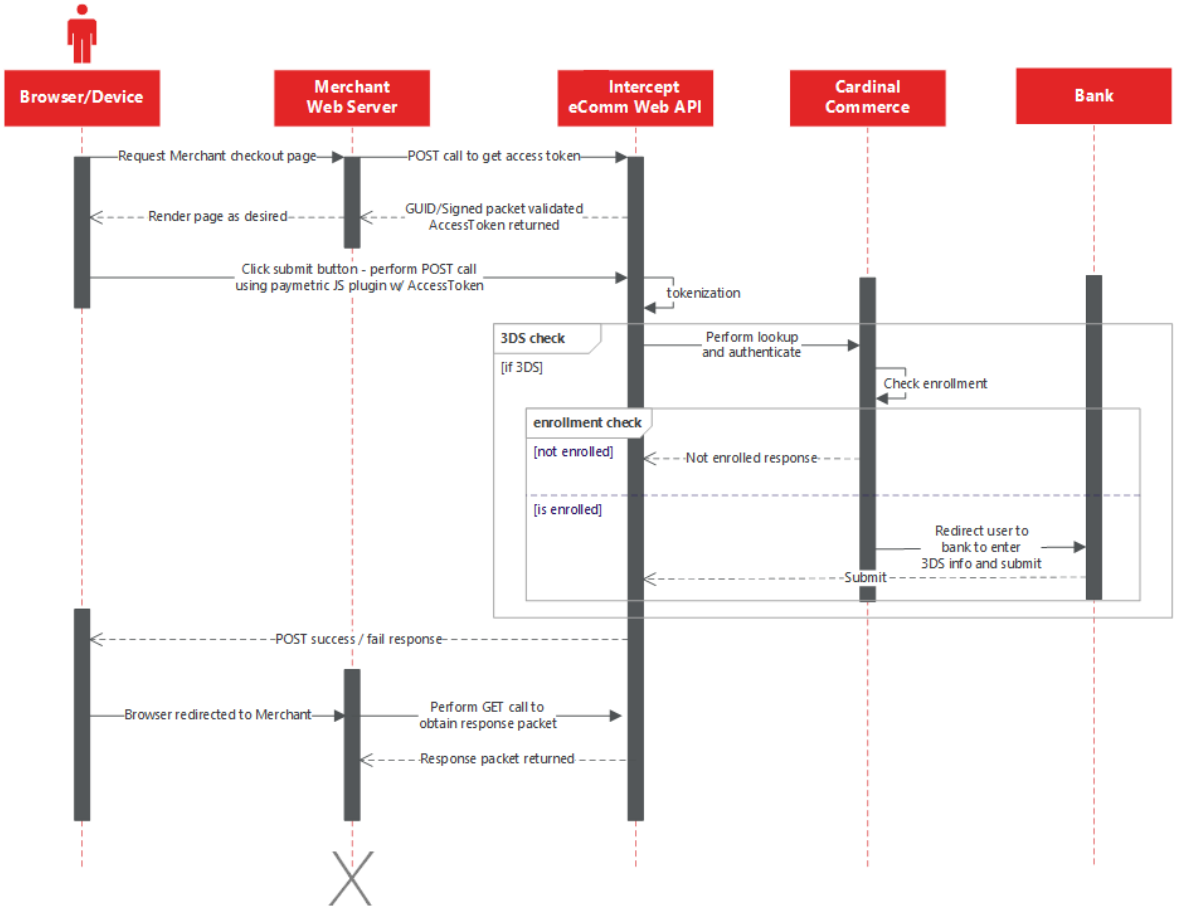
Form Post integration consists of the following high-level steps:

1. Initiate Intercept eComm session with a POST call to get the access token.
    - If implementing 3DS, there are extra fields that may be provided in the PacketXML. If 3DS2 there is also an attribute to indicate the 3DS version.
2. Render page as desired and upon user submission of credit card data, perform POST call using the JS plugin.
    - If implementing 3DS 2.0, Intercept eComm automatically performs the additional device data collection (DDC) used by Cardinal and the Banks.
3. Perform GET call to obtain the response packet containing the token.

The following sequence diagram illustrates the Form Post integration flow. Review the subsections below for more details.

> **Note:**
>
> If implementing 3DS2, the user is only redirected to the bank to enter 3DS credentials if the DDC or merchant-provided risk information is determined to be insufficient.

5 TMS form post integration



5.1 Form Post: Get AccessToken

AccessToken is required to initiate an Intercept eComm session.

AccessToken is a string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

AccessToken is valid for 30 minutes. If AccessToken expires before the process is complete, you must restart the process and obtain a new AccessToken.

Request method

A direct server-side POST is required to retrieve AccessToken. How this POST is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''.

Endpoints

Cert

```
https://cert-xiecomm.worldpay.com/DIeComm/AccessToken
```

Production

```
https://xiecomm.worldpay.com/DIeComm/AccessToken
```

POST Call parameters

```
MerchantGuid={Your-GUID}&SessionRequestType={Int-Type-Code}
&Signature={Signature&MerchantDevelopmentEnvironment={code language e.g.
asp.net}&Packet={PACKET_XML}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for TES to support the Token Management Service integration.
SessionRequestType	Indicates the integration type. Use 3 for Form Post.

Parameter	Description
Signature	Sign the Packet XML; encode it with HMAC-SHA-256 using the shared key then convert it to a base 64 string. The Signature must be URL encoded.
MerchantDevelopmentEnvironment	Indicate the language in which you are coding your integration to Intercept eComm, for example, asp.net or php.
Packet XML	The Packet XML must be URL encoded. Following are the elements within `PostPacketMode` container: • `RedirectUri` = the redirect URL for the Response Packet from Intercept eComm • `TokenProvider` container = specify in `Provider ProfileId` elements the token types to be returned; these are the values configured as the Name in Merchant Portal under Token Exchange Service Accounts. It is not case-sensitive. If implementing 3DS, include the following fields as well within `Cardinal3DSRequest` container: • `Amount` = amount of transaction • `CurrencyCode` = currency code for transaction • `CardNumberFieldName` = the merchant's field name that will contain the card number value entered by the User • `ExpirationMonthFieldName` = the merchant's field name that will contain the expiration month value entered by the User

Parameter	Description
	• `ExpirationYearFieldName` = the merchant's field name that will contain the expiration year value entered by the User • `OrderNumber` = the merchant's order number • `AdditionalFields` = optional container with additional fields for 3DS2 See samples below.

5.1.1 Form Post XML sample

Form Post XML sample

```
<?xml version="1.0"?>
<PostPacketModel>
   <RedirectUri>{http://Yourdomain.here}</RedirectUri>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
</PostPacketModel>
```

5.1.2 Form Post with 3DS 1 XML sample

Form Post with 3DS 1 XML sample

```
<?xml version="1.0"?>
<PostPacketModel>
   <RedirectUri>{http://Yourdomain.here}</RedirectUri>
   <Cardinal3DSRequest>
       <Amount>{AmountValue}</Amount>
       <CurrencyCode>{CurrencyCodeValue}</CurrencyCode>
       <CardNumberFieldName>{CCNumValue}</CardNumberFieldName>
       <ExpirationMonthFieldName>{ExpMonthValue}
</ExpirationMonthFieldName>
       <ExpirationYearFieldName>{ExpYearValue}
</ExpirationYearFieldName>
       <OrderNumber>{OrderNumValue}</OrderNumber>
   </Cardinal3DSRequest>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
</PostPacketModel>
```

5.1.3 Form Post with 3DS 2 XML sample

The 3DS fields that differ form the prior example and are specific to version 2 are highlighted blue in the example below. These include `threeDSVersion` attribute within the `<Cardinal3DSecure>` container and the `<AdditionalFields>` container and child elements.

The `threeDSVersion` attribute must be set to "2" or can be a specific version up to 3 digits. If there is a dot (.), then there must be a following number. E.g. "2" or "2.0" or "2.2", and so on.

Any Cardinal 3DS field can be included in `<AdditionalFields>` container. See the following URL for a complete list of the additional Cardinal fields:

https://cardinaldocs.atlassian.net/wiki/spaces/CCen/pages/905478187/Lookup+Request+Response

⚠ Note:

Cardinal `<AdditionalFields>` container cannot have duplicate field elements.

Form Post with 3DS 2 XML sample

```
<?xml version="1.0"?>
<PostPacketModel>
   <RedirectUri>{http://Yourdomain.here}</RedirectUri>
   <Cardinal3DSRequest threeDSVersion="2.2.0">
       <Amount>{AmountValue}</Amount>
       <CurrencyCode>{CurrencyCodeValue}</CurrencyCode>
       <CardNumberFieldName>{CCNumValue}</CardNumberFieldName>
       <ExpirationMonthFieldName>{ExpMonthValue}
</ExpirationMonthFieldName>
       <ExpirationYearFieldName>{ExpYearValue}
</ExpirationYearFieldName>
       <OrderNumber>{OrderNumValue}</OrderNumber>
       <AdditionalFields>
           <Field name="UserAccountAge" value="04" />
           <Field name="UserPasswwordChange" value="20190325" />
           <Field name="DeliveryTimeframe" value="02" />
           <Field name="GiftCardAmount" value="100" />
       </AdditionalFields>
       <TokenProviders>
           <Provider ProfileId="dm-vantiv" />
           <Provider ProfileId="dm-xisecure" />
           <Provider ProfileId="dm-wpg" />
       </TokenProviders>
   </Cardinal3DSRequest>
</PostPacketModel>
```

5.2 Form Post: On-Submit Javascript plugin

When the customer performs the submit action by clicking the merchant-created submit button, the data will be posted using JavaScript functions.

Following are examples of pulling the values from the text boxes and populating the data variable.

Populating data variable example default field names

```
var myData = $XIPlugin.createJSRequestPacket(merchantGuid, accessToken);
myData.addField($XIPlugin.createField('CardNumber', true, cc));
myData.addField($XIPlugin.createField('Name', false, name));
myData.addPayloadFields("card", {});
```

If you do not use default field names, then you must map the custom field names in `addPayloadFields`. See the following example for mapping the custom field CreditCardNumber to the default field CardNumber.

Populating data variable example custom field names

```
var myData = $XIPlugin.createJSRequestPacket(merchantGuid, accessToken);
 myData.addField($XIPlugin.createField('CreditCardNumber', true, cc));
 myData.addField($XIPlugin.createField('Name', false, name));
 myData.addPayloadFields("card", {'CreditCardNumber':'CardNumber'});
```

Either reference or download the following JavaScript plugin:

`https://xiecomm.worldpay.com/DIeComm/Scripts/XIPlugin/XIPlugin-1.2.0.js`

Include the Paymetric JavaScript.

Form Post: QA JS submit example

```
$XIPlugin.submit({

url: "https://cert-xiecomm.worldpay.com/DIeComm/Form",
data: {Formatted JS Data – See above example myData},
validation: function(a)
{
//Run custom validation
return true; //should return true or false.

// return true will go through with post.
// return false will not go through with post.
}
  });
```

Form Post: QA JS submit 3DS2 example

```
$XIPlugin.submit({

url: "https://cert-xiecomm.worldpay.com/DIeComm/Form",
data: {Formatted JS Data – See above example myData},
validation: function(a)
{
//Run custom validation
return true; //should return true or false.

// return true will go through with post.
// return false will not go through with post.
},
threeDSVersion: "2.2.0"

  });
```

5.3 Form Post: Get Response packet

Request method

A direct server-side GET is required to retrieve the Response Packet. How this GET is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''.

Endpoints

Cert

```
https://cert-xiecomm.worldpay.com/DIeComm/ResponsePacket
```

Production

```
https://xiecomm.worldpay.com/DIeComm/ResponsePacket
```

GET Call parameters

```
MerchantGUID={MerchGUID}&Signature={Signature}&AccessToken={AccessToken}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for TES to support the Token Management Service integration.
Signature	Sign the Access Token; encode it with HMAC-SHA-256 using the shared key then covert it to a base 64 string. The Signature must be URL encoded.

AccessToken	A string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

Response packet sample

```
<TokenizationResponse>
 <Fields>
  <Field xsi:type="TokenizedField">
   <Name>Card Number</Name>
   <Values>
    <Value tokenProvider="Vantiv Token Provider" profileId="dm-
vantiv">4111116905781111</Value>
    <Value tokenProvider="XiSecure Token Provider" profileId="dm-
xisecure">-E803-1111-N6QTDT86AZYJ52</Value>
    <Value tokenProvider="Worldpay Token Provider" profileId="dm-
wpg">9963105801395450000</Value>
   </Values>
   <TokenErrors />
  </Field>
  <Field xsi:type="RawField">
   <Name>Name</Name>
   <Value>John Doe</Value>
  </Field>
 </Fields>
</TokenizationResponse>
```

6 TMS Ajax integration

> **Note:**
>
> Do not send raw card numbers in your environment.

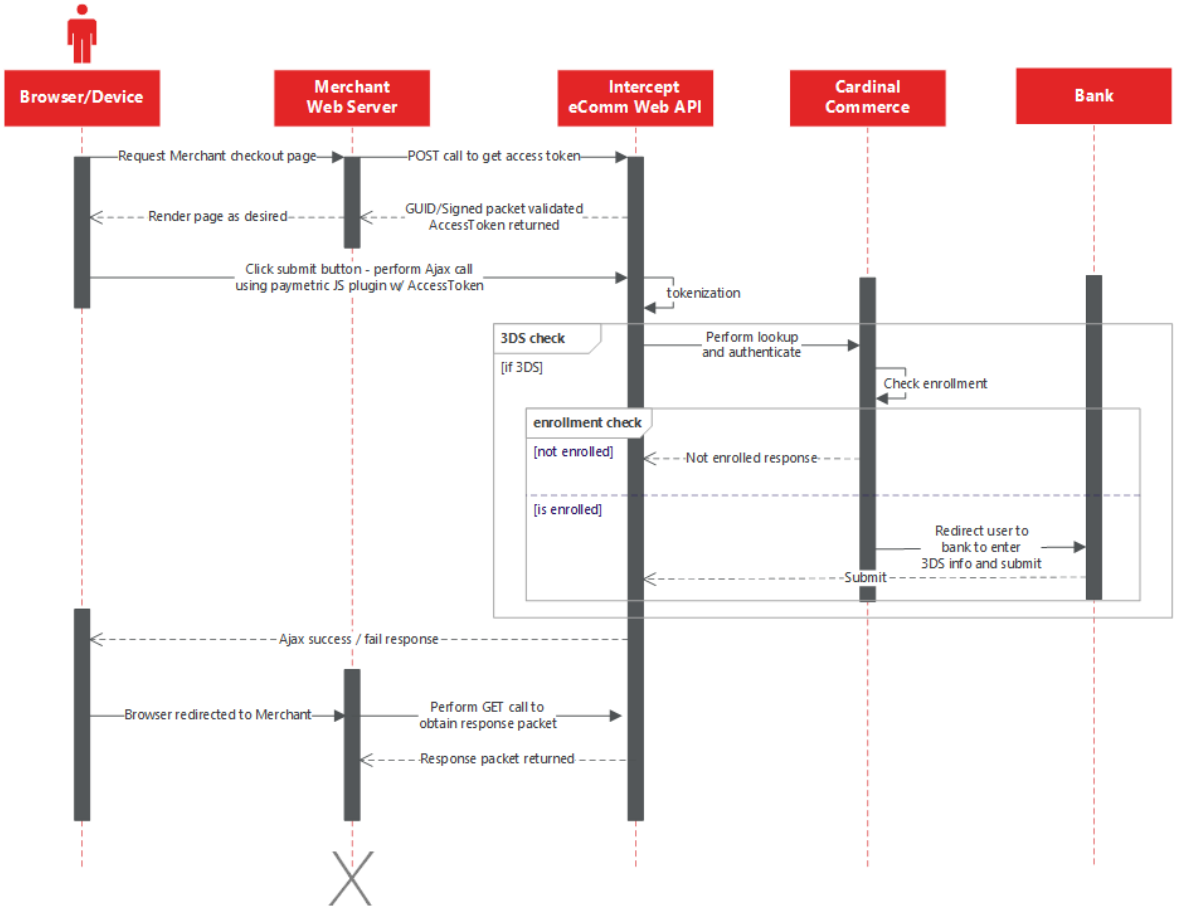
Ajax integration consists of the following high-level steps:

1. Initiate Intercept eComm session with a POST call to get the access token.
    - If implementing 3DS, there are extra fields that may be provided in the PacketXML. If 3DS2 there is also an attribute to indicate the 3DS version.
2. Render page as desired and upon user submission of Credit Card data, perform POST call using the JS plugin.
    - If implementing 3DS 2.0, Intercept eComm automatically performs the additional device data collection (DDC) used by Cardinal and the Banks.
3. Perform GET call to get the response packet containing the token. If implementing 3DS, all response fields from Cardinal Commerce will be included as well.

The following sequence diagram illustrates the Ajax integration flow. Review the subsections below for more details.

> **Note:**
>
> If implementing 3DS2, the user is only redirected to the bank to enter 3DS credentials if the DDC or merchant-provided risk information is determined to be insufficient.

6 TMS Ajax integration



6.1 Ajax: GetAccess token

AccessToken is required to initiate an Intercept eComm session.

AccessToken is a string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

AccessToken is valid for 30 minutes. If the AccessToken expires before the process is complete, you must restart the process and obtain a new AccessToken.

Request method

A direct server-side POST is required to retrieve the Access Token. How this POST is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''.

Endpoints

Cert

```
https://cert-xiecomm.worldpay.com/DIeComm/AccessToken
```

Production

```
https://xiecomm.worldpay.com/DIeComm/AccessToken
```

POST Call parameters

```
MerchantGuid={Your-GUID}&SessionRequestType={Int-Type-Code}
&Signature={Signature}&MerchantDevelopmentEnvironment={code language e.g.
asp.net}&Packet={PACKET_XML}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for TES to support the Token Management Service integration.
SessionRequestType	Indicates the integration type. Use 2 for Ajax.

Parameter	Description
Signature	Sign the Packet XML; encode it with HMAC-SHA-256 using the shared key then convert it to a base 64 string. The Signature must be URL encoded.
MerchantDevelopmentEnvironment	Indicate the language in which you are coding your integration to Intercept eComm, for example, asp.net or php.
Packet XML	The Packet XML must be URL encoded. Following are the elements within `AjaxPacketModel` container: • TokenProvider container = specify in Provider ProfileId elements the token types to be returned; these are the values configured as the Name in Merchant Portal under Token Exchange Service Accounts. It is not case-sensitive. If implementing 3DS, include the following fields as well within `Cardinal3DSRequest` container: • Amount – Amount of transaction • CurrencyCode – currency code for transaction • CardNumberFieldName – Merchant's field name that will contain the card number value entered by the User • ExpirationMonthFieldName – Merchant's field name that will contain the expiration month value entered by the User • ExpirationYearFieldName – Merchant's field name that will contain the expiration year value entered by the User

Parameter	Description
	• OrderNumber – Merchant's order number • AdditionalFields – Optional container with additional fields for 3DS2 **Note**: If you have a previous integration that sends HostUri, this field is not used. It will have no effect. See samples below.

6.1.1 Ajax XML sample

Ajax XML sample

```
<?xml version="1.0"?>
<AjaxPacketModel>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
   <AjaxPacketModel/>
```

6.1.2 Ajax with 3DS 1 XML sample

Ajax with 3DS 1 with existing XML sample

```
<?xml version="1.0"?>
<AjaxPacketModel>
   <Cardinal3DSRequest>
       <RedirectUri>{http://YourRedirectURLHere}</RedirectUri>
       <Amount>{AmountValue}</Amount>
       <CardNumberFieldName>{CCNumValue}</CardNumberFieldName>
       <CurrencyCode>{CurrencyCodeValue}</CurrencyCode>
       <ExpirationMonth>{ExpMonthValue}</ExpirationMonth>
       <ExpirationYear>{ExpYearValue}</ExpirationYear>
       <OrderNumber>{OrderNumValue}</OrderNumber>
   </Cardinal3DSRequest>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
</AjaxPacketModel>
```

6.1.3 Ajax with 3DS 2 XML sample

The 3DS fields that differ form the prior example and are specific to version 2 are highlighted blue in the example below. These include `threeDSVersion` attribute within the `<cardinal3DSecure>` container and the `<AdditionalFields>` container and child elements.

The `threeDSVersion` attribute must be set to "2" or can be a specific version up to 3 digits. If there is a dot (.), then there must be a following number. E.g. "2" or "2.0" or "2.2", and so on.

Any Cardinal 3DS field can be included in `<AdditionalFields>` container. See the following URL for a complete list of the additional Cardinal fields:

https://cardinaldocs.atlassian.net/wiki/spaces/CCen/pages/905478187/Lookup+Request+Response

Note:

Cardinal <AdditionalFields> container cannot have duplicate field elements.

Ajax with 3DS 2 with existing XML sample

```
<?xml version="1.0"?>
<AjaxPacketModel>
   <Cardinal3DSRequest threeDSVersion="2.2.0">
       <RedirectUri>{http://YourRedirectURLHere}</RedirectUri>
       <Amount>{AmountValue}</Amount>
       <CardNumberFieldName>{CCNumValue}</CardNumberFieldName>
       <CurrencyCode>{CurrencyCodeValue}</CurrencyCode>
       <ExpirationMonth>{ExpMonthValue}</ExpirationMonth>
       <ExpirationYear>{ExpYearValue}</ExpirationYear>
       <OrderNumber>{OrderNumValue}</OrderNumber>
       <AdditionalFields>
           <Field name="UserAccountAge" value="04" />
           <Field name="UserPasswwordChange" value="20190325" />
           <Field name="DeliveryTimeframe" value="02" />
           <Field name="GiftCardAmount" value="100" />
       </AdditionalFields>
   </Cardinal3DSRequest>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
</AjaxPacketModel>
```

6.2 Ajax: On-Submit Javascript plugin

When the Customer performs the submit action by clicking the merchant-created submit button, the data will be posted to Paymetric using JavaScript functions.

Following is an example of pulling the values from the text boxes and populating the data variable.

Note:

For 3DS2, you can add the Cardinal additional fields via the JavaScript plugin as well. The following "Populating Data Variables Example" includes one of the 3DS2 AdditionalFields.

If you send the same additional field that you sent when getting the access token, it will be overwritten.

If you accidentally add more than one field with the same name and different values via the plugin, it will take the value of the first entry. So, for example:

- Access token field "UserAccountAge" = 01
- JavaScript field "UserAccountAge" = 02
- JavaScript field "UserAccountAge" = 03

The value used will be "02".

Following are examples of pulling the values from the text boxes and populating the data variable.

Populating data variable example default field names

```
var myData = $XIPlugin.createJSRequestPacket(merchantGuid, accessToken);
myData.addField($XIPlugin.createField('CardNumber', true, cc));
myData.addField($XIPlugin.createField('Name', false, name));
```

Populating data variable example default field names

```
myData.addPayloadFields("card", {});
```

If you do not use default field names, then you must map the custom field names in `addPayloadFields`. See the following example for mapping the custom field CreditCardNumber to the default field CardNumber.

Populating data variable example custom field names

```
var myData = $XIPlugin.createJSRequestPacket(merchantGuid, accessToken);
 myData.addField($XIPlugin.createField('CreditCardNumber', true, cc));
 myData.addField($XIPlugin.createField('Name', false, name));
 myData.addPayloadFields("card", {'CreditCardNumber':'CardNumber'});
```

You can reference or download the following JavaScript plugin:

`https://xiecomm.worldpay.com/DIeComm/Scripts/XIPlugin/XIPlugin-1.2.0.js`

Include the JavaScript and reference the Submit Example below to handle the posting of the credit card details directly to Intercept eComm.

Ajax: QA JS Submit example

```
 $XIPlugin.ajax({

   url: '<% Intercept Url%>' + "/Ajax",
   type: "POST", //<GET/POST>
   data: {
       Formatted JS Data– See above example myData
   },
   success: function(a) {
       //function triggered on success.
   },
   error: function(a) {
       //function triggered on failure.
   },
   beforeSend: function(a) {
       //function triggered before ajax call.
```

Ajax: QA JS Submit example

```
    },
    complete: function(a) {
        //function triggered after ajax call.
        // After success or failure
    }
    validation: function(a) {
        //Run custom validation
        return true; //should return true or false.

        // return true will go through with ajax call.
        // return false will not go through with ajax call.
    }


});
```

Ajax: QA JS Submit 3DS2 example

```
$XIPlugin.ajax({

    url: '<% Intercept Url%>' + "/Ajax",
    type: "POST", //<GET/POST>
    data: {
        Formatted JS Data– See above example myData
    },
    success: function(a) {
        //function triggered on success.
    },
    error: function(a) {
        //function triggered on failure.
    },
    beforeSend: function(a) {
        //function triggered before ajax call.
    },
    complete: function(a) {
        //function triggered after ajax call.
        // After success or failure
    }
    validation: function(a) {
        //Run custom validation
        return true; //should return true or false.

        // return true will go through with ajax call.
        // return false will not go through with ajax call.
```

Ajax: QA JS Submit 3DS2 example

```
    },
    threeDSVersion: "2.2.0"

    intcp3DSecure: function(a) {
        //Optional function for custom 3DS challenge prompt; set to null
or omit for default prompt; default is full redirect to a hosted page
    }
});
```

6.3 Ajax: Get Response packet

Request method

A direct server-side GET is required to retrieve the Response Packet. How this GET is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''

Endpoints

Cert

`https://cert-xiecomm.worldpay.com/DIeComm/ResponsePacket`

Production

`https://xiecomm.worldpay.com/DIeComm/ResponsePacket`

GET Call parameters

```
MerchantGUID={MerchGUID}&Signature={Signature}
&AccessToken={AccessToken}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for integration with Token Management Service.
Signature	Sign the Access Token; encode it with HMAC-SHA-256 using the shared key then covert it to a base 64 string. The Signature must be URL encoded.
AccessToken	A string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

Response packet sample

```
<TokenizationResponse>
   <Fields>
       <Field xsi:type="TokenizedField">
           <Name>Card Number</Name>
           <Values>
               <Value tokenProvider="Vantiv Token Provider"
profileId="dm-vantiv">4111116905781111</Value>
               <Value tokenProvider="XiSecure Token Provider"
profileId="dm-xisecure">-E803-1111-N6QTDT86AZYJ52</Value>
               <Value tokenProvider="Worldpay Token Provider"
profileId="dm-wpg">9963105801395450000</Value>
           </Values>
           <TokenErrors />
       </Field>
       <Field xsi:type="RawField">
           <Name>Name</Name>
           <Value>John Doe</Value>
```

Response packet sample

```
        </Field>
    </Fields>
</TokenizationResponse>
```

7 TMS Iframe integration

Iframe integration consists of the following high-level steps:

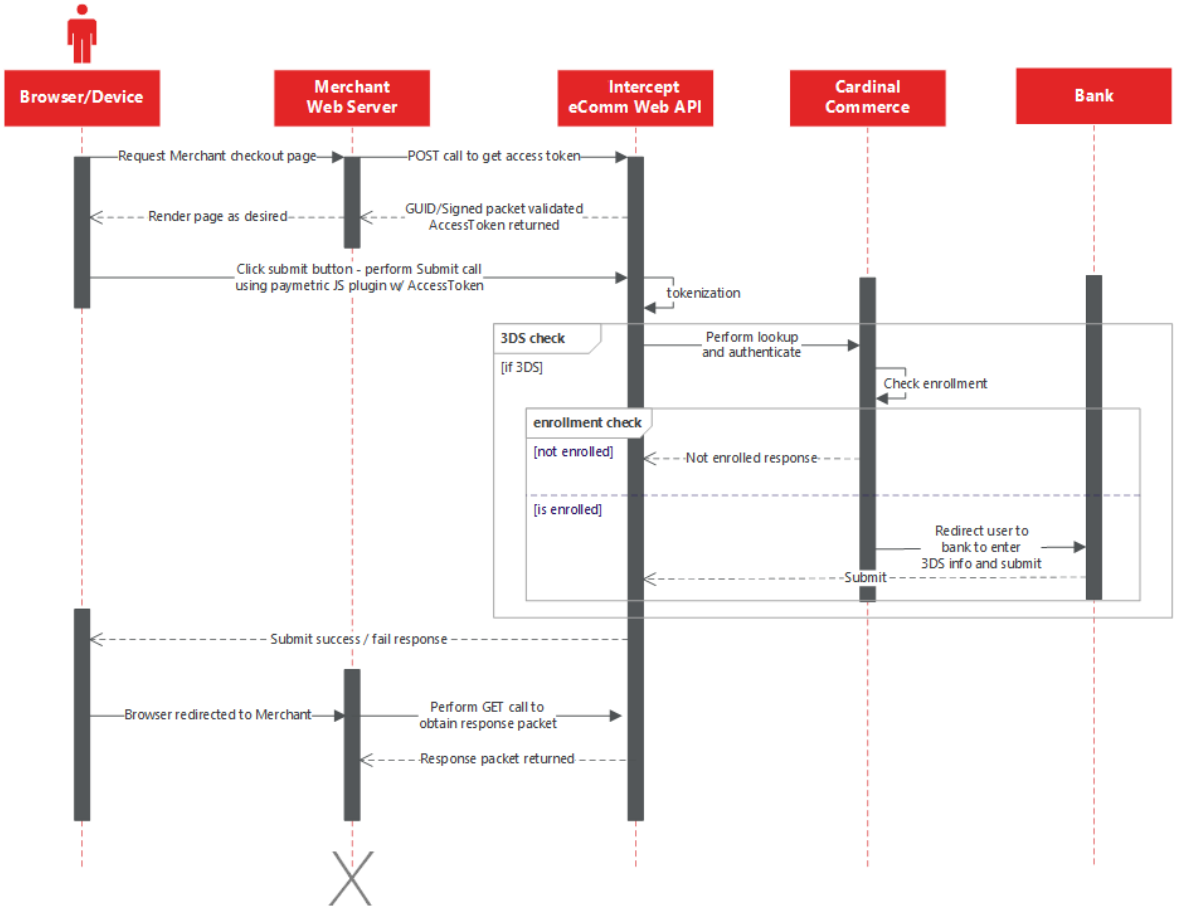
1. Initiate Intercept eComm session with a POST call to get the access token.
    - If implementing 3DS, there are extra fields that may be provided in the PacketXML including an attribute to indicate the 3DS version.
2. Embed Iframe in web page.
3. Render page as desired and upon user submission of Credit Card data, perform POST call using the JS plugin.
    - If implementing 3DS 2.0, Intercept eComm automatically performs the additional device data collection (DDC) used by Cardinal and the Banks.
4. Perform GET call to get the response packet containing the token. If implementing 3DS, all response fields from Cardinal Commerce will be included as well.

The following sequence diagram illustrates the Iframe integration flow. Review the subsections below for more details.

Note:

If implementing 3DS2, the user is only redirected to the bank to enter 3DS credentials if the DDC or merchant-provided risk information is determined to be insufficient.

7 TMS Iframe integration



Templates and customizations

When using the Iframe integration type, there are templates available for the following:

- Card
- Card, CVV
- Card, CVV and Address

Alternatively, you can customize the fields that display. Refer to Custom HTML reference 123 for complete details on the Custom Iframe option.

Whether using an Iframe template or the Custom Iframe option, you can use your own Cascading Style Sheet (CSS) to control the look and feel of your web page.

Localization

You have the ability to localize standard labels and messaging for Iframe integrations; see the Localization 98 section for more information.

7.1 Iframe: Get AccessToken

AccessToken is required to initiate an Intercept eComm session.

AccessToken is a string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

AccessToken is valid for 30 minutes. If the AccessToken expires before the process is complete, you must restart the process and obtain a new AccessToken.

Request method

A direct server-side POST is required to retrieve the Access Token. How this POST is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''

Endpoints

Cert

```
https://cert-xiecomm.worldpay.com/DIeComm/AccessToken
```

Production

```
https://xiecomm.worldpay.com/DIeComm/AccessToken
```

POST Call parameters

```
MerchantGuid={Your-GUID}&SessionRequestType={Int-Type-Code}
&Signature={Signature}&MerchantDevelopmentEnvironment={code language e.g.
asp.net}&Packet={PACKET_XML}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for TES to support the Token Management Service integration.
SessionRequestType	Indicates the integration type. Use 1 for Iframe.
Signature	Sign the Packet XML; encode it with HMAC-SHA-256 using the shared key then convert it to a base 64 string. The Signature must be URL encoded.
MerchantDevelopmentEnvironment	Indicate the language in which you are coding your integration to Intercept eComm, for example, asp.net or php.
Packet XML	The Packet XML must be URL encoded. Following are the elements within `merchantHtmlPacketModel` container: • Iframe Templates

Parameter	Description
	o `hostUri` – Domain of the Iframe parent page o `cssUri` – URL of the merchant's CSS file o `templateHtml` – Of three templates (creditcard, creditcardCVV, creditcardCVVAddress) o `paymentTypes` – Within templateHtml container, create separate payment type for each one supported; these display in the card type/payment type dropdown • TokenProvider container – Specify in Provider ProfileId elements the token types to be returned; these are the values configured as the Name in Merchant Portal under Token Exchange Service Accounts. It is not case-sensitive. • Iframe Custom: o `hostUri` – Domain of the Iframe parent page o `cssUri` – URL of the merchant's CSS file o `merchantHTML` – See Custom HTML Reference for details o `TokenProvider container` – Specify in Provider ProfileId elements the token types to be returned; these are the values configured as the Name in Merchant Portal under Token Exchange Service Accounts. It is not case-sensitive. • If implementing 3DS, include the following attributes for the <cardinal3DSecure> element for either the IWrame Template or Iframe Custom:

Parameter	Description
	o `redirectUri` – Redirect URL back to the merchant after 3DS processing o `amount` – Amount of transaction o `currencyCode` – Currency code for transaction o `orderNumber` – Merchant's order number o `additionalFields` – Optional container with additional fields for 3DS2 See samples below.

7.1.1 Iframe template XML sample

Iframe template XML sample

```
<?xml version="1.0" encoding="utf-8"?>
<merchantHtmlPacketModel>
   <iFramePacket>
       <hostUri>{http://Yourdomain.here}/</hostUri>
       <cssUri>{http://Yourcssurlhere.here}</cssUri>
   </iFramePacket>
   <templateHtml name="CreditCard">
       <paymentTypes>
           <paymentType type="american express" />
           <paymentType type="mastercard" />
           <paymentType type="maestro" />
           <paymentType type="visa" />
           <paymentType type="custom" value="HD" text="Home Depot" />
       </paymentTypes>
   </templateHtml>
   <TokenProviders>
```

Iframe template XML sample

```
        <Provider ProfileId="dm-vantiv" />
        <Provider ProfileId="dm-xisecure" />
        <Provider ProfileId="dm-wpg" />
    </TokenProviders>
</merchantHtmlPacketModel>
```

7.1.2 Iframe custom XML sample

If you do not use the default field names (as defined in the Field Requirements Per Token Provider 13 section), you need to add the `map-to` attribute, such as `PostalCode` or `Country` in the following example.

Iframe custom XML sample

```
<merchantHtmlPacketModel>
   <iFramePacket>
       <hostUri>{http://Yourdomain.here}</hostUri>
       <cssUri>{http://Yourcssurlhere.here}</cssUri>
   </iFramePacket>
   <htmlFieldValues>
       <field for="cardholderName" value="John Smith" />
       <additionalField for="foo" value="My custom value" />
   </htmlFieldValues>
   <merchantHtml>
       <htmlSection class="merchant_paycontent"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
           <xiProperties error-class="xiInputError">
               <errorTooltip class="errorTooltip" show-effect="fadeIn"
show-duration="5000" hide-effect="fadeOut" hide-duration="5000" />
           </xiProperties>
           <cardDropdownSection>
               <tag name="div" class="PaymentDetailHeader">Enter your
credit card information</tag>
               <tag name="div">
                   <label for="cardType" text="card type" />
                   <ddlCardType id="cd">
                       <items>
```

Iframe custom XML sample

```
                            <item for="visa" />
                            <item for="american express" />
                            <item for="mastercard" />
                        </items>
                    </ddlCardType>
                </tag>
                <tag name="div">
                    <label for="cardNumber" text="card number" />
                    <tboxCardNumber tokenize="true" />
                    <validationMsg for="cardNumber" class="valmsg" />
                </tag>
                <tag name="div">
                    <label for="cardholderName" text="name on card" />
                    <tboxCardHolderName tokenize="true" />
                    <validationMsg for="cardholderName"
class="valmsg" />
                </tag>
                <additionalHtmlSection>
                    <tag name="div">
                        <label text="Postal Code" />
                        <textBox name="PostalCode" map-
to="PostalCode" />
                    </tag>
                    <tag name="div">
                        <label text="Country" />
                        <textBox name="Place" map-to="Country" />
                    </tag>
                </additionalHtmlSection>
            </cardDropdownSection>
        </htmlSection>
   </merchantHtml>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
</merchantHtmlPacketModel>
```

7.1.3 Iframe custom with 3DS 1 XML sample

If you do not use the default field names (as defined in the Field Requirements Per Token Provider 13 section), you need to add the `map-to` attribute, such as `PostalCode` or `Country` in the following example.

Iframe Custom with 3DS 1.0 XML sample

```
<merchantHtmlPacketModel
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
   <iFramePacket>
       <hostUri>{http://Yourdomain.here}</hostUri>
       <cssUri>{http://Yourcssurihere.here}</cssUri>
       <cardinal3DSecure amount="13.42" currencyCode="840"
orderNumber="12345" redirectUri="{http://Yourredirecturi.here}" />
   </iFramePacket>
   <merchantHtml>
       <htmlSection class="merchant_paycontent"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
           <xiProperties error-class="xiInputError">
               <errorTooltip class="errorTooltip" show-effect="fadeIn"
show-duration="5000" hide-effect="fadeOut" hide-duration="5000" />
           </xiProperties>
           <cardDropdownSection>
               <tag name="div" class="PaymentDetailHeader">Enter your
credit card information</tag>
               <tag name="div">
                   <label for="cardType" text="card type" />
                   <ddlCardType id="cd">
                       <items>
                           <item for="visa" />
                           <item for="american express" />
                           <item for="mastercard" />
                       </items>
                   </ddlCardType>
               </tag>
               <tag name="div">
                   <label for="cardNumber" text="card number" />
                   <tboxCardNumber tokenize="true" />
                   <validationMsg for="cardNumber" class="valmsg" />
               </tag>
               <tag name="div">
                   <label for="cardholderName" text="name on card" />
                   <tboxCardHolderName tokenize="true" />
                   <validationMsg for="cardholderName"
class="valmsg" />
```

Iframe Custom with 3DS 1.0 XML sample

```
                </tag>
                <additionalHtmlSection>
                    <tag name="div">
                        <label text="Postal Code" />
                        <textBox name="PostalCode" map-
to="PostalCode" />
                    </tag>
                    <tag name="div">
                        <label text="Country" />
                        <textBox name="Place" map-to="Country" />
                    </tag>
                </additionalHtmlSection>
            </cardDropdownSection>
        </htmlSection>
    </merchantHtml>
    <TokenProviders>
        <Provider ProfileId="dm-vantiv" />
        <Provider ProfileId="dm-xisecure" />
        <Provider ProfileId="dm-wpg" />
    </TokenProviders>
</merchantHtmlPacketModel>
```

7.1.4 Iframe custom with 3DS 2 XML sample

The 3DS fields that differ form the prior example and are specific to version 2 are highlighted blue in the example below. These include `threeDSVersion` attribute within the `<cardinal3DSecure>` container and the `<additionalFields>` container and child elements.

The `threeDSVersion` attribute must be set to "2" or can be a specific version up to 3 digits. If there is a dot (.), then there must be a following number. E.g. "2" or "2.0" or "2.2", and so on.

Any Cardinal 3DS field can be included in `<additionalFields>` container. See the following URL for a complete list of the additional Cardinal fields:

https://cardinaldocs.atlassian.net/wiki/spaces/CCen/pages/905478187/Lookup+Request+Response

Note

- Cardinal `<additionalFields>` container cannot have duplicate field elements.
- If you do not use the default field names (as defined in the Field Requirements Per Token Provider section), you need to add the `map-to` attribute, such as `PostalCode` or `Country` in the following example.

Iframe custom with 3DS 2.0 XML sample

```
<merchantHtmlPacketModel
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
   <iFramePacket>
       <hostUri>{http://Yourdomain.here}</hostUri>
       <cssUri>{http://Yourcssurl.here}</cssUri>
       <cardinal3DSecure amount="13.42" currencyCode="840"
orderNumber="12345" redirectUri="{http://Yourredirecturi.here}"
threeDSVersion="2.2.0">
           <additionalFields>
               <field name="UserAccountAge" value="04" />
               <field name="UserPasswwordChange" value="20190325" />
               <field name="DeliveryTimeframe" value="02" />
               <field name="GiftCardAmount" value="100" />
           </additionalFields>
       </cardinal3DSecure>
   </iFramePacket>
   <merchantHtml>
       <htmlSection class="merchant_paycontent"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
           <xiProperties error-class="xiInputError">
               <errorTooltip class="errorTooltip" show-effect="fadeIn"
show-duration="5000" hide-effect="fadeOut" hide-duration="5000" />
           </xiProperties>
           <cardDropdownSection>
               <tag name="div" class="PaymentDetailHeader">Enter your
credit card information</tag>
               <tag name="div">
                   <label for="cardType" text="card type" />
                   <ddlCardType id="cd">
                       <items>
```

Iframe custom with 3DS 2.0 XML sample

```
                            <item for="visa" />
                            <item for="american express" />
                            <item for="mastercard" />
                        </items>
                    </ddlCardType>
                </tag>
                <tag name="div">
                    <label for="cardNumber" text="card number" />
                    <tboxCardNumber tokenize="true" />
                    <validationMsg for="cardNumber" class="valmsg" />
                </tag>
                <tag name="div">
                    <label for="cardholderName" text="name on card" />
                    <tboxCardHolderName tokenize="true" />
                    <validationMsg for="cardholderName"
class="valmsg" />
                </tag>
                <additionalHtmlSection>
                    <tag name="div">
                        <label text="Postal Code" />
                        <textBox name="PostalCode" map-
to="PostalCode" />
                    </tag>
                    <tag name="div">
                        <label text="Country" />
                        <textBox name="Place" map-to="Country" />
                    </tag>
                </additionalHtmlSection>
            </cardDropdownSection>
        </htmlSection>
   </merchantHtml>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" />
       <Provider ProfileId="dm-xisecure" />
       <Provider ProfileId="dm-wpg" />
   </TokenProviders>
</merchantHtmlPacketModel>
```

7.2 Iframe: Embed Iframe in Web page

To set the source URL of the Iframe:

```
Iframe source = {DieCommUrl}/view/iframe/{MerchantGuid}/
{AccessTokenPacket}/{BubbleUpErrorToParentWindow(boolean Type)}
```

For example:

```
https://cert-xiecomm.worldpay.com/diecomm/View/Iframe/8c9aea21-6063-
4f30-b09d-e44706fedb05/942fb4bf-x546-4b32-942d-76aadg3087e2/True
```

Iframe renders using fields defined in the Iframe: Get Access Token request.

Iframe sizing can be handled by HTML or CSS.

Resizing of the Iframe can be handled by CSS, but if there is a need to handle resizing depending upon content that is inside Iframe (e.g. different fields for different payment methods or validation error messages), then include the following function.

Resize Iframe function sample

```
$XIFrame.onload({
      iFrameId: {Iframe ID},
      targetUrl: {Iframe source}
      autosizewidth: true,
      onSuccess: function (e) {
        //Successfully Iframe Loaded
      },
      onError: function (e) {
        //Error on Iframe Loading
      }
    });
```

7.3 Iframe: On-Submit Javascript plugin

When the customer performs the submit action by clicking the merchant-created submit button, the data will be posted using JavaScript functions.

You can reference or download the following JavaScript plugin:

https://xiecomm.worldpay.com/DIeComm/Scripts/XIFrame/XIFrame-1.2.0.js

Include the JavaScript and reference the Submit Example below to handle the posting of the credit card details directly to Intercept eComm.

QA JS submit example

```
  $XIFrame.submit({
    iFrameId: {
        Iframe ID
    },
    targetUrl: {
        Iframe source
    }
    onSuccess: function(msg) {
        //function triggered on success.
    },
    onError: function(msg) {
        //function triggered on failure
    }
});
```

If validation passes and tokenization is successful, then the onSuccess function is called; otherwise, onError function is called.

QA JS submit 3DS 2 example

```
  $XIFrame.submit({
    iFrameId: {
        Iframe ID
    },
    targetUrl: {
```

QA JS submit 3DS 2 example

```
        Iframe source
    }
    onSuccess: function(msg) {
        //function triggered on success.
    },
    onError: function(msg) {
        //function triggered on failure
    },
    threeDSVersion: "2.2.0"

});
```

7.4 Iframe: Get Response packet

Request method

A direct server-side GET is required to retrieve the Response Packet. How this GET is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''

Endpoints

Cert

`https://cert-xiecomm.worldpay.com/DIeComm/ResponsePacket`

Production

`https://xiecomm.worldpay.com/DIeComm/ResponsePacket`

GET Call parameters

Following is the GET data:

```
MerchantGUID={MerchGUID}&Signature={Signature}
&AccessToken={AccessToken}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for integration with Token Management Service.
Signature	Sign the Access Token; encode it with HMAC-SHA-256 using the shared key then covert it to a base 64 string. The Signature must be URL encoded.
AccessToken	A string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

Response packet sample

```
<TokenizationResponse>
   <Fields>
       <Field xsi:type="TokenizedField">
           <Name>Card Number</Name>
           <Values>
               <Value tokenProvider="Vantiv Token Provider"
profileId="dm-vantiv">4111116905781111</Value>
               <Value tokenProvider="XiSecure Token Provider"
profileId="dm-xisecure">-E803-1111-N6QTDT86AZYJ52</Value>
               <Value tokenProvider="Worldpay Token Provider"
profileId="dm-wpg">9963105801395450000</Value>
           </Values>
           <TokenErrors />
       </Field>
       <Field xsi:type="RawField">
```

Response packet sample

```
            <Name>Name</Name>
            <Value>John Doe</Value>
         </Field>
   </Fields>
</TokenizationResponse>
<TokenProviders>
   <Provider ProfileId="dm-vantiv" />
   <Provider ProfileId="dm-xisecure" />
   <Provider ProfileId="dm-wpg" />
</TokenProviders>
</merchantHtmlPacketModel>
```

7.5 Iframe: Additional information

The section contains additional information related to Iframe integration:

- Multiple IFrames 58
- Display content based on selected payment type 59
- Embed 3DS in IFrame 59

7.5.1 Multiple IFrames

You can implement multiple IFrames on your web page, for example, if you want to allow multiple payment methods.

The workflow for multiple IFrames is the same as documented above in Iframe Workflow Step 3.b -- for the Java Iframe Submit function, you define the Iframe instance and then reference that, e.g. instead of `$XIFrame.submit` it could be `iframe1Plugin.submit` and then `iframe2Plugin.submit`.

You can find JavaScript examples under JavaScript Plugin API Reference > XIFrame - Iframe API Reference > Multiple IFrames 119 in this document.

7.5.2 Display content based on selected payment type

Intercept eComm has a feature that allows a change to the Payment Type field within the Iframe to fire an event at the parent level.

The `onCardTypeChange` element is used to trigger the on-change event.

```
///Create 2 IFrames.  hold on to the XIFrame instance
iframe1Plugin = $XIFrame({
    iFrameId: iFrameName1,
    targetUrl: $(iframe1).attr("src"),
    onCardTypeChange: IFrame1CardChange, //register event for card type change
    autosizeheight: true,
    autosizewidth: true,
    onError: function(msg) { alert(msg); },
    onValidate: Validation,
    storeCardAmount: true,
    intcp3DSecure: Custom1Handling
```

7.5.3 Embed 3DS in IFrame

In the standard workflow defined above, the 3DS call redirects to the Bank's website. You can choose to embed the 3DS fields within the Iframe. The following screenshot displays an example of 3DS fields embedded within the Iframe.

7 TMS Iframe integration

8 TMS hosted integration

Hosted integration consists of the following high-level steps:

1. Initiate Intercept eComm session with a POST call to get the access token upon page load then render page as desired.

    - If implementing 3DS, there are extra fields that should be provided in the PacketXML including an attribute to indicate the 3DS version.

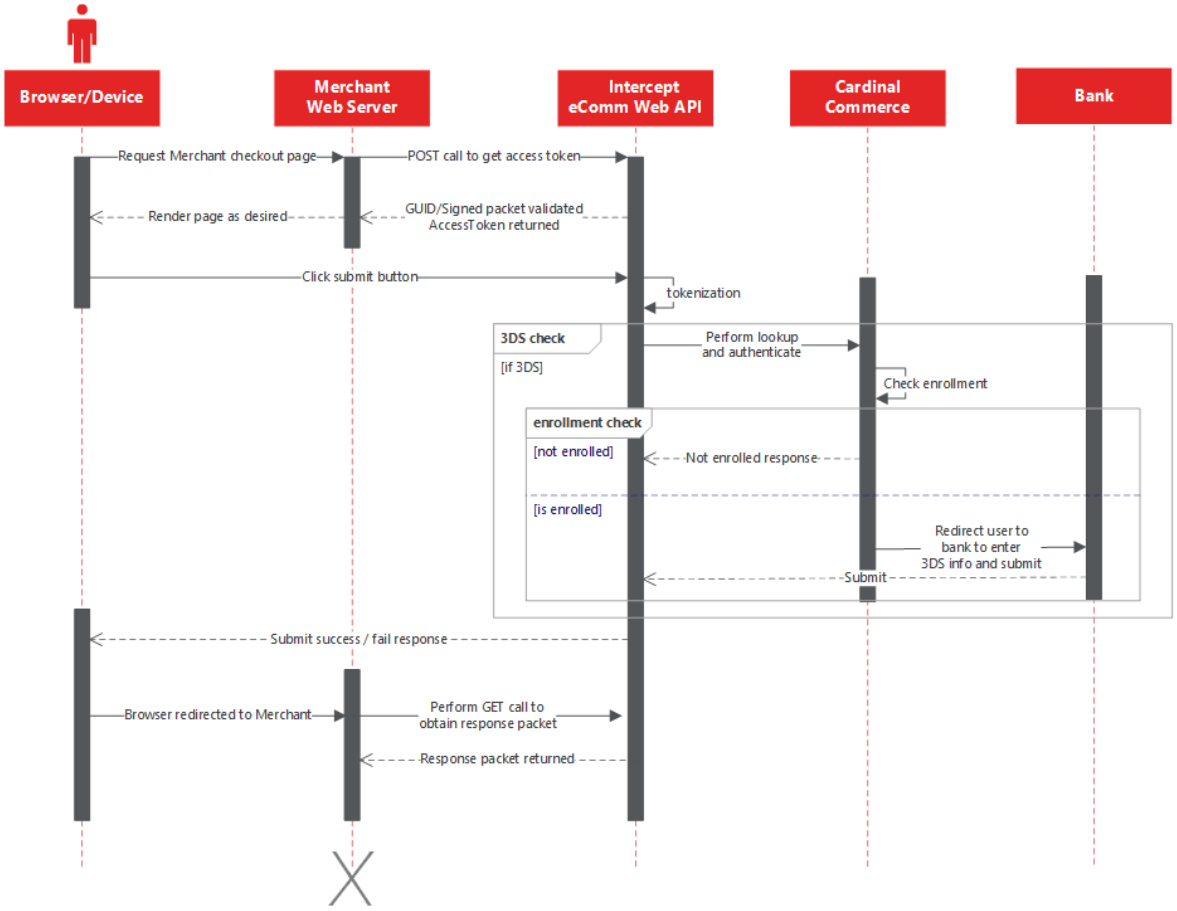
2. Redirect to Hosted page.

    - If implementing 3DS 2.0, Intercept eComm automatically performs the additional device data collection (DDC) used by Cardinal and the Banks.

3. Perform GET call to get the response packet containing the token. If implementing 3DS, all response fields from Cardinal Commerce will be included as well.

Note:

If implementing 3DS2, the user is only redirected to the bank to enter 3DS credentials if the DDC or merchant-provided risk information is determined to be insufficient.



Localization

You have the ability to localize standard labels and messaging for hosted integrations; see the Localization 98 section for more information.

8 TMS hosted integration

8.1 Hosted: Get AccessToken

AccessToken is required to initiate an Intercept eComm session.

AccessToken is a string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

AccessToken is valid for 30 minutes. If the AccessToken expires before the process is complete, you must restart the process and obtain a new AccessToken.

Request method

A direct server-side POST is required to retrieve the Access Token. How this POST is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''.

Endpoints

Cert

```
https://cert-xiecomm.worldpay.com/DIeComm/AccessToken
```

Production

```
https://xiecomm.worldpay.com/DIeComm/AccessToken
```

POST Call parameters

Following is the POST data:

```
MerchantGuid={Your-GUID}&SessionRequestType={Int-Type-Code}
&Signature={Signature}&MerchantDevelopmentEnvironment={code language e.g.
asp.net}&Packet={PACKET_XML}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned.
SessionRequestType	Indicates the integration type. Use 4 for Hosted.
Signature	Sign the Packet XML; encode it with HMAC-SHA-256 using the shared key then convert it to a base 64 string. The Signature must be URL encoded.
MerchantDevelopmentEnvironment	Indicate the language in which you are coding your integration to Intercept eComm, for example, asp.net or php.
Packet XML	The Packet XML must be URL encoded. Following are the elements within `merchantHtmlPacketModel` container: • `redirectUri` = the redirect URL for the Response Packet from Intercept eComm • `cssUri` = the URL of the Merchant's CSS file • `merchantHtml` = see Custom HTML Reference for details (and XML sample below) • `TokenProvider` container = specify in `Provider ProfileId` elements the token types to be returned; these are the values configured as the Name in Merchant Portal under Token Exchange Service Accounts. It is not case-sensitive.

Parameter	Description
	If implementing 3DS, include the following attributes for the <cardinal3DSecure> element for either the Iframe Template or Iframe Custom: • `redirectUri` = the redirect URL back to the merchant after 3DS processing • `amount` = amount of transaction • `currencyCode` = currency code for transaction • `orderNumber` = the Merchant's order number • `additionalFields` = optional container with additional fields for 3DS2 See samples below.
customerRedirectUri	Applicable only for the Hosted page when implementing XiSecurelink. Defines the merchant redirect URL after the User clicks submit.

8.1.1 Hosted custom XML sample

Hosted custom XML sample
`<?xml version="1.0" encoding="utf-8"?>`

Hosted custom XML sample

```
<merchantHtmlPacketModel
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
   <paymentPagePacket>
       <redirectUri>{http://Yourdomain.here}</redirectUri>
       <cssUri>http://dasvm121/DIeComm
Sample/Content/PaymentStyleSheet.css</cssUri>
   </paymentPagePacket>
   <merchantHtml>
       <htmlSection class="merchant_paycontent"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
           <xiProperties error-class="xiInputError">
               <errorTooltip class="errorTooltip" show-effect="fadeIn"
show-duration="5000" hide-effect="fadeOut" hide-duration="5000" />
           </xiProperties>
           <cardDropdownSection>
               <tag name="div" class="PaymentDetailHeader">Enter your
credit card information</tag>
               <tag name="div">
                   <label for="cardType" text="card type" />
                   <ddlCardType id="cd">
                       <items>
                           <item for="visa" />
                           <item for="american express" />
                           <item for="mastercard" />
                       </items>
                   </ddlCardType>
               </tag>
               <tag name="div">
                   <label for="cardNumber" text="card number" />
                   <tboxCardNumber tokenize="true" />
                   <validationMsg for="cardNumber" class="valmsg" />
               </tag>
               <tag name="div">
                   <label for="cardholderName" text="name on card" />
                   <tboxCardHolderName tokenize="true" />
                   <validationMsg for="cardholderName"
class="valmsg" />
               </tag>
               <additionalHtmlSection>
                   <tag name="div">
                       <label text="Postal Code" />
                       <textBox name="PostalCode" map-
to="PostalCode" />
                   </tag>
                   <tag name="div">
                       <label text="Country" />
```

Hosted custom XML sample

```
                        <textBox name="Place" map-to="Country" />
                    </tag>
                </additionalHtmlSection>
            </cardDropdownSection>
        </htmlSection>
    </merchantHtml>
```

8.1.2 Hosted custom with 3DS 1 XML sample

Hosted custom with 3DS XML sample

```
<?xml version="1.0" encoding="utf-8"?>
<merchantHtmlPacketModel
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
   <paymentPagePacket>
       <redirectUri>{http://Yourdomain.here}</redirectUri>
       <cssUri>http://dasvm121/DIeComm
Sample/Content/PaymentStyleSheet.css</cssUri>
       <cardinal3DSecure amount="13.42" currencyCode="840"
orderNumber="12345" />
   </paymentPagePacket>
   <merchantHtml>
       <htmlSection class="merchant_paycontent"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
           <xiProperties error-class="xiInputError">
               <errorTooltip class="errorTooltip" show-effect="fadeIn"
show-duration="5000" hide-effect="fadeOut" hide-duration="5000" />
           </xiProperties>
           <cardDropdownSection>
               <tag name="div" class="PaymentDetailHeader">Enter your
credit card information</tag>
               <tag name="div">
                   <label for="cardType" text="card type" />
                   <ddlCardType id="cd">
                       <items>
                           <item for="visa" />
                           <item for="american express" />
                           <item for="mastercard" />
                       </items>
```

Hosted custom with 3DS XML sample

```
                    </ddlCardType>
                </tag>
                <tag name="div">
                    <label for="cardNumber" text="card number" />
                    <tboxCardNumber tokenize="true" />
                    <validationMsg for="cardNumber" class="valmsg" />
                </tag>
                <tag name="div">
                    <label for="cardholderName" text="name on card" />
                    <tboxCardHolderName tokenize="true" />
                    <validationMsg for="cardholderName"
class="valmsg" />
                </tag>
                <additionalHtmlSection>
                    <tag name="div">
                        <label text="Postal Code" />
                        <textBox name="PostalCode" map-
to="PostalCode" />
                    </tag>
                    <tag name="div">
                        <label text="Country" />
                        <textBox name="Place" map-to="Country" />
                    </tag>
                </additionalHtmlSection>
            </cardDropdownSection>
        </htmlSection>
    </merchantHtml>
    <TokenProviders>
        <Provider ProfileId="dm-vantiv" />
        <Provider ProfileId="dm-xisecure" />
        <Provider ProfileId="dm-wpg" />
    </TokenProviders>
</merchantHtmlPacketModel>
```

8.1.3 Hosted custom with 3DS 2 XML sample

The 3DS fields that differ form the prior example and are specific to version 2 are highlighted blue in the example below. These include `threeDSVersion` attribute within the

<cardinal3DSecure> container and the <additionalFields> container and child elements.

The threeDSVersion attribute must be set to "2" or can be a specific version up to 3 digits. If there is a dot (.), then there must be a following number. E.g. "2" or "2.0" or "2.2", and so on.

Any Cardinal 3DS field can be included in <additionalFields> container. See the following URL for a complete list of the additional Cardinal fields:

https://cardinaldocs.atlassian.net/wiki/spaces/CCen/pages/905478187/Lookup+Request+Response

Note:

Cardinal <AdditionalFields> container cannot have duplicate field elements.

Hosted custom with 3DS XML sample

```
<?xml version="1.0" encoding="utf-8"?>
<merchantHtmlPacketModel
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
   <paymentPagePacket>
       <redirectUri>{http://Yourdomain.here}</redirectUri>
       <cssUri>http://dasvm121/DIeComm
Sample/Content/PaymentStyleSheet.css</cssUri>
       <cardinal3DSecure amount="13.42" currencyCode="840"
orderNumber="12345" threeDSVersion="2.2.0">
           <additionalFields>
               <field name="UserAccountAge" value="04" />
               <field name="UserPasswwordChange" value="20190325" />
               <field name="DeliveryTimeframe" value="02" />
               <field name="GiftCardAmount" value="100" />
           </additionalFields>
       </paymentPagePacket>
       <merchantHtml>
           <htmlSection class="merchant_paycontent"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
               <xiProperties error-class="xiInputError">
```

Hosted custom with 3DS XML sample

```
                    <errorTooltip class="errorTooltip" show-
effect="fadeIn" show-duration="5000" hide-effect="fadeOut" hide-
duration="5000" />
                </xiProperties>
                <cardDropdownSection>
                    <tag name="div" class="PaymentDetailHeader">Enter
your credit card information</tag>
                    <tag name="div">
                        <label for="cardType" text="card type" />
                        <ddlCardType id="cd">
                            <items>
                                <item for="visa" />
                                <item for="american express" />
                                <item for="mastercard" />
                            </items>
                        </ddlCardType>
                    </tag>
                    <tag name="div">
                        <label for="cardNumber" text="card number" />
                        <tboxCardNumber tokenize="true" />
                        <validationMsg for="cardNumber"
class="valmsg" />
                    </tag>
                    <tag name="div">
                        <label for="cardholderName" text="name on
card" />
                        <tboxCardHolderName tokenize="true" />
                        <validationMsg for="cardholderName"
class="valmsg" />
                    </tag>
                    <additionalHtmlSection>
                        <tag name="div">
                            <label text="Postal Code" />
                            <textBox name="PostalCode" map-
to="PostalCode" />
                        </tag>
                        <tag name="div">
                            <label text="Country" />
                            <textBox name="Place" map-to="Country" />
                        </tag>
                    </additionalHtmlSection>
                </cardDropdownSection>
            </htmlSection>
        </merchantHtml>
        <TokenProviders>
            <Provider ProfileId="dm-vantiv" />
```

Hosted custom with 3DS XML sample

```
            <Provider ProfileId="dm-xisecure" />
            <Provider ProfileId="dm-wpg" />
        </TokenProviders>
    </merchantHtmlPacketModel>
```

8.2 Hosted: Redirect to hosted page

After receiving the AccessToken response packet, redirect Customer to the Intercept eComm hosted page at the following URLs.

Cert

```
 https://cert-xiecomm.worldpay.com/DIeComm/paymentpage/index/{Merchant GUID}/{Access Token}/
```

Production

```
https://xiecomm.worldpay.com/DIeComm/paymentpage/index/{Merchant GUID}/{Access Token}/
```

User enters information and submits. If user cancels on the rendered page.

8.3 Hosted: Get Response packet

Request method

A direct server-side GET is required to retrieve the Response Packet. How this GET is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''.

Endpoints

Cert

```
https://cert-xiecomm.worldpay.com/DIeComm/ResponsePacket
```

Production

```
https://xiecomm.worldpay.com/DIeComm/ResponsePacket
```

GET Call parameters

Following is the GET data:

```
MerchantGUID=<MerchGUID>&Signature=<SIGNATURE&AccessToken=<AccessToken>
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned that is enabled for integration with Token Management Service.
Signature	Sign the Access Token; encode it with HMAC-SHA-256 using the shared key then covert it to a base 64 string. The Signature must be URL encoded.
AccessToken	A string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

Response packet sample

```
<TokenizationResponse>
   <Fields>
       <Field xsi:type="TokenizedField">
           <Name>Card Number</Name>
           <Values>
               <Value tokenProvider="Vantiv Token Provider"
profileId="dm-vantiv">4111116905781111</Value>
               <Value tokenProvider="XiSecure Token Provider"
profileId="dm-xisecure">-E803-1111-N6QTDT86AZYJ52</Value>
               <Value tokenProvider="Worldpay Token Provider"
profileId="dm-wpg">9963105801395450000</Value>
           </Values>
           <TokenErrors />
       </Field>
       <Field xsi:type="RawField">
           <Name>Name</Name>
           <Value>John Doe</Value>
       </Field>
   </Fields>
</TokenizationResponse>
```

9 3DS Call with token integration

Intercept eComm supports a workflow that allows you to submit a token for the 3DS call if, for example, you have an existing customer with a token on file.

The Ajax call in this workflow does not refer to the Ajax Intercept eComm integration type for raw card numbers. This is a completely separate workflow that is implemented independently of the raw card section.

3DS with token integration consists of the following high-level steps:

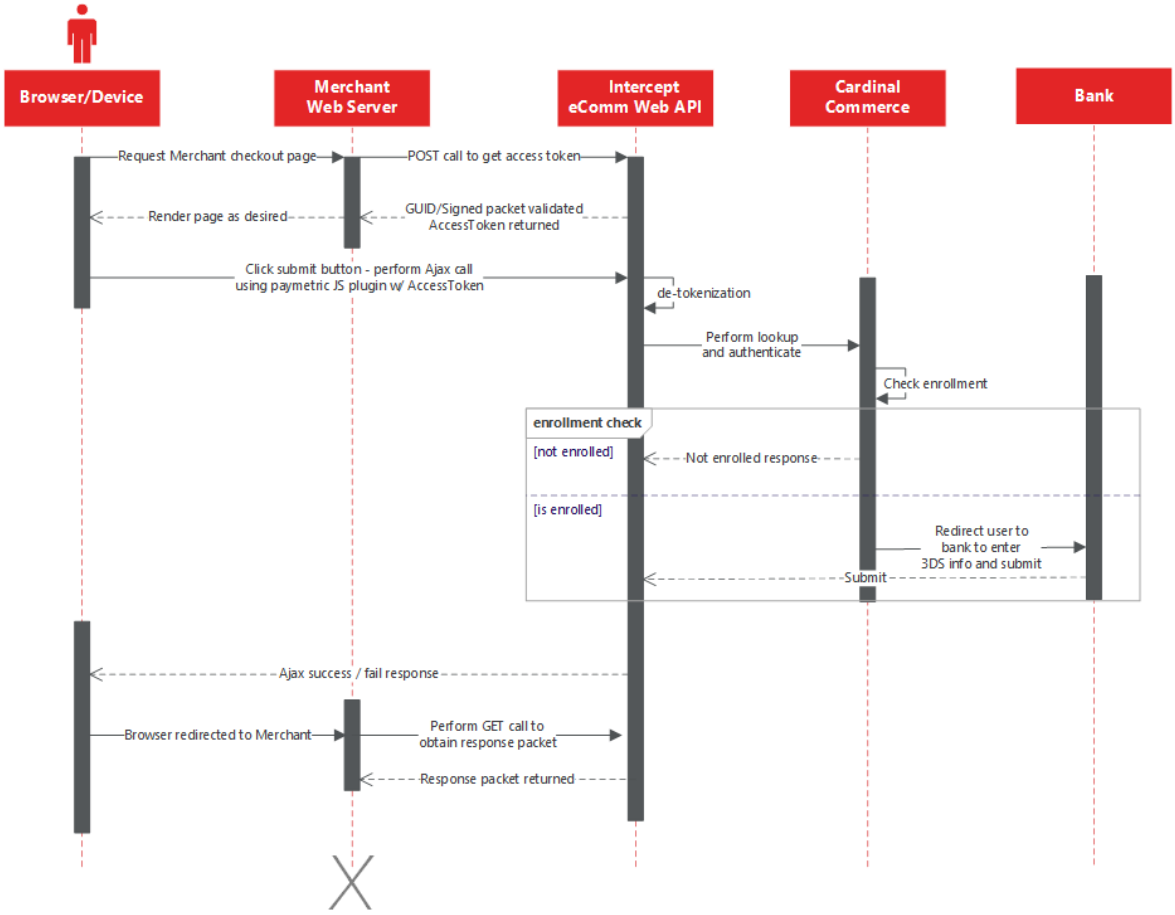
1. Initiate Intercept eComm session with a POST call to get the access token.
2. Merchant renders page as desired and upon user initiation of 3DS lookup call, perform POST call using the JS plugin.
3. Perform GET call to get the response packet with token. All response fields from Cardinal Commerce are included in the response.

The following sequence diagram illustrates the 3DS with Token integration flow. Review the subsections below for more details.

Note:

If implementing 3DS2, the user is only redirected to the bank to enter 3DS credentials if the DDC or merchant-provided risk information is determined to be insufficient.

9 3DS Call with token integration



9.1 3DS Token: Get AccessToken

AccessToken is required to initiate an Intercept eComm session.

AccessToken is a string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

AccessToken is valid for 30 minutes. If the AccessToken expires before the process is complete, you must restart the process and obtain a new AccessToken.

Request method

A direct server-side POST is required to retrieve the Access Token. How this POST is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''

Endpoints

Cert

```
<%QADOMAIN%>/DIeComm/AccessToken
```

Production

```
<%PRODDOMAIN%>/DIeComm/AccessToken
```

POST Call parameters

```
MerchantGuid={Your-GUID}&SessionRequestType={Int-Type-Code}
&Signature={Signature}&MerchantDevelopmentEnvironment={code language e.g.
asp.net}&Packet={PACKET_XML}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned.
SessionRequestType	Indicates the integration type. Use 2 for Ajax.

Parameter	Description
Signature	Sign the Packet XML; encode it with HMAC-SHA-256 using the shared key then convert it to a base 64 string. The Signature must be URL encoded.
MerchantDevelopment Environment	Indicate the language in which you are coding your integration to Intercept eComm, for example, asp.net or php.
Packet XML	The Packet XML must be URL encoded. Following are the elements: • RedirectUri = the redirect URL for the Response Packet from Intercept eComm • Within the <Cardinal3DSWithToken> container: o `Amount` = amount of transaction o `CurrencyCode` = currency code for transaction o `ExpirationMonth` = the expiration month value entered by the User o `ExpirationYear` = the expiration year value entered by the User o `OrderNumber` = the Merchant's order number o `Token` = the Merchant's token provider number o `AdditionalFields` = optional container with additional fields for 3DS2

Parameter	Description
	o `TokenProviders` = additional container with `ProfileID` and `InputToken` fields for the token providers o `InputToken` = the profile that can detokenize that token in the packet See samples below.

9.1.1 3DS 1 with token XML sample

3DS with token XML sample

```
<?xml version="1.0"?>
<Cardinal3DSWithTokenPacketModel>
   <RedirectUri>http://dasvm121/DIeComm Sample/Status.aspx</RedirectUri>
   <Cardinal3DSRequestWithToken>
       <Amount>1.42</Amount>
       <CurrencyCode>840</CurrencyCode>
       <ExpirationMonth>06</ExpirationMonth>
       <ExpirationYear>2018</ExpirationYear>
       <OrderNumber>12345</OrderNumber>
       <Token>1234567890123456</Token>
   </Cardinal3DSRequestWithToken>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" InputToken="true" />
       <Provider ProfileId="dm-xisecure" InputToken="false" />
       <Provider ProfileId="dm-wpg" InputToken="false" />
   </TokenProviders>
</Cardinal3DSWithTokenPacketModel>
```

9.1.2 3DS 2 with token XML sample

The 3DS fields that differ form the prior example and are specific to version 2 are highlighted blue in the sample below. These include `threeDSVersion` attribute within the `<Cardinal3DSRequestWithToken>` container and the `<AdditionalFields>` container and child elements.

The `threeDSVersion` attribute must be set to "2" or can be a specific version up to 3 digits. If there is a dot (.), then there must be a following number. E.g. "2" or "2.0" or "2.2", and so on.

Any Cardinal 3DS field can be included in `<AdditionalFields>` container. See the following URL for a complete list of the additional Cardinal fields:

https://cardinaldocs.atlassian.net/wiki/spaces/CCen/pages/905478187/Lookup+Request+Response

Note:

Cardinal <AdditionalFields> container cannot have duplicate field elements.

3DS 2 with token XML sample

```
<?xml version="1.0"?>
<Cardinal3DSWithTokenPacketModel>
   <RedirectUri>http://dasvm121/DIeComm Sample/Status.aspx</RedirectUri>
   <Cardinal3DSRequestWithToken threeDSVersion="2.2.0">
       <Amount>1.42</Amount>
       <CurrencyCode>840</CurrencyCode>
       <ExpirationMonth>06</ExpirationMonth>
       <ExpirationYear>2018</ExpirationYear>
       <OrderNumber>12345</OrderNumber>
       <Token>1234567890123456</Token>
       <AdditionalFields>
           <Field name="UserAccountAge" value="04" />
           <Field name="UserPasswwordChange" value="20190325" />
           <Field name="DeliveryTimeframe" value="02" />
           <Field name="GiftCardAmount" value="100" />
       </AdditionalFields>
```

3DS 2 with token XML sample

```
   </Cardinal3DSRequestWithToken>
   <TokenProviders>
       <Provider ProfileId="dm-vantiv" InputToken="true" />
       <Provider ProfileId="dm-xisecure" InputToken="false" />
       <Provider ProfileId="dm-wpg" InputToken="false" />
   </TokenProviders>
</Cardinal3DSWithTokenPacketModel>
```

9.2 3DS Token: On-submit Javascript Plugin

When the Customer performs the submit action by clicking the merchant-created submit button, the data will be posted using JavaScript functions.

Following is an example of pulling the values from the text boxes and populating the data variable.

Note:

For 3DS2, you can add the Cardinal additional fields via the JavaScript plugin as well. The following "Populating Data Variables Example" includes one of the 3DS2 AdditionalFields.

If you send the same additional field that you sent when getting the access token, it will be overwritten.

If you accidentally add more than one field with the same name and different values via the plugin, it will take the value of the first entry. So, for example:

- Access token field "UserAccountAge" = 01
- JavaScript field "UserAccountAge" = 02

- JavaScript field "UserAccountAge" = 03

The value used will be "02".

Populating data variables example

```
   var myData = $XIPlugin.createJSRequestPacket(merchantGuid,
accessToken);


    myData.addField($XIPlugin.createField('Amount', false,
$('#textbox').val()));
    myData.addAdditional3DSField("UserAccountAge", "04");
```

Note: Amount is optional from the browser.

You can reference or download the following JavaScript plugin:

```
<%PRODDOMAIN%>/DIeComm/Scripts/XIPlugin/XIPlugin-1.2.0.js
```

Include the JavaScript and reference the Submit Example below to handle the posting of the credit card details directly to Intercept eComm.

QA JS submit example

```
  $XIPlugin.ajax({

    url: '<% Intercept Url%>' + "/Cardinal3DSWithToken",
    type: "POST", //<GET/POST>
    data: {
        Formatted JS Data– See above example myData
    },
    success: function(a) {
        //function triggered on success.
    },
    error: function(a) {
        //function triggered on failure.
    },
    beforeSend: function(a) {
```

QA JS submit example

```
        //function triggered before ajax call.
    },
    complete: function(a) {
        //function triggered after ajax call.
        // After success or failure
    }

    validation: function(a) {
        //Run custom validation
        return true; //should return true or false.

        // return true will go through with ajax call.
        // return false will not go through with ajax call.
    }
});
```

QA JS submit 3DS 2 example

```
 $XIPlugin.ajax({

    url: '<% Intercept Url%>' + "/Cardinal3DSWithToken",
    type: "POST", //<GET/POST>
    data: {
        Formatted JS Data– See above example myData
    },
    success: function(a) {
        //function triggered on success.
    },
    error: function(a) {
        //function triggered on failure.
    },
    beforeSend: function(a) {
        //function triggered before ajax call.
    },
    complete: function(a) {
        //function triggered after ajax call.
        // After success or failure
    }

    validation: function(a) {
        //Run custom validation
        return true; //should return true or false.

        // return true will go through with ajax call.
```

QA JS submit 3DS 2 example

```
        // return false will not go through with ajax call.
    },
    threeDSVersion: "2.2.0"

});
```

9.3 3DS Token: Get Response packet

Request method

A direct server-side GET is required to retrieve the Response Packet. How this GET is performed depends on the language you are using.

Content type

The content type must be "application/x-www-form-urlencoded''

Endpoints

Cert

```
<%QADOMAIN%>/DIeComm/ResponsePacket
```

Production

```
<%PRODDOMAIN%>/DIeComm/ResponsePacket
```

GET Call parameters

```
MerchantGUID={MerchGUID}&Signature={Signature}
&AccessToken={AccessToken}
```

Parameter	Description
MerchantGuid	The Global Unique Identifier assigned.
Signature	Sign the Access Token; encode it with HMAC-SHA-256 using the shared key then covert it to a base 64 string. The Signature must be URL encoded.
AccessToken	A string that is dynamically generated by Intercept eComm during the first application server call. The call includes the Merchant GUID and the Intercept eComm Request Packet that is signed by the Shared Key.

10 PCI v4.0 A compliance

Intercept eComm supports PCI 4.0 SAQ-A workflows that still require the submission of the CVV during an authorization.

To do this, Intercept eComm has an endpoint that allows you to submit the authorization without a CVV populated. Intercept eComm then populates the CVV from the response packet into the authorization before submitting it to XiPay. This is supported for every integration type, including 3DS.

High level process

The PCI 4.0 SAQ-A workflow is detailed at a high-level below:

1. Initiate Intercept eComm session with a POST call to get the access token.
2. Merchant renders page as desired and submits according to the integration type.
3. Perform GET call to get the response packet with token.
    - The CVV will not be returned in this response if you have **PCI 4.0 SAQ-A** support enabled.
4. Create XML for authorization to XiPay.
    - This XML does not need to change, but the CVV will not be available to populate.
5. Submit authorization XML to the Intercept eComm PerformAuthorization endpoint using a POST call with merchant GUID and access token.
    - Intercept eComm will populate the CVV in the authorization XML if CVV was submitted in step 2.
6. PerformAuthorization endpoint responds with the XiPay authorization response.

- The CVV will be removed from the XiPay response if it was originally returned to comply with PCI 4.0 SAQ-A.

10.1 Form Post: Perform authorization

Request method

A direct server-side POST is required to submit the authorization. How this POST is performed depends on the language you are using.

Content type

The content type must be "application/xml" or "text/xml".

Endpoints

Cert

```
https://cert-
xiecomm.paymetric.com/DIeComm/XiPay/PerformAuthorization/<MerchantGui
d>/<AccessToken>
```

Production

```
https://xiecomm.paymetric.com/DIeComm/XiPay/PerformAuthorization/<Mer
chantGuid>/<AccessToken>
```

POST body

The body should contain the entire XiPay XML to be submitted for authorization.

Example URL

```
https://cert-xiecomm.paymetric.com/
DIeComm/XiPay/PerformAuthorization/81175969-3FFA-49C0-A246-
493A8854E3C9/188192f9-fb74-454b-aafd-7989c813584e
```

Example POST body

```
<?xml version="1.0" encoding="utf-8"?>

<SOAP-ENV:Envelope xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:wsse="http://docs
.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ns1="http:
//Paymetric/XiPaySoap30" xmlns:ns2="http://Paymetric/XiPaySoap30/messag
e/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">

    <SOAP-ENV:Body>
        <ns2:SoapOp>
            <ns2:pPacketsIn>
                <ns2:count>1</ns2:count>
                <ns2:xipayvbresult>false</ns2:xipayvbresult>
                <ns2:packets>
                    <ns2:ITransactionHeader>
                        <ns2:PacketOperation>1</ns2:PacketOperation>
                        <ns2:MerchantID>PMSalem</ns2:MerchantID>

<ns2:CardNumber>CardNumberValue</ns2:CardNumber>
                        <ns2:CardDataSource>E</ns2:CardDataSource>
                        <ns2:CardPresent>0</ns2:CardPresent>
                        <ns2:CardType>VI</ns2:CardType>
                        <ns2:CardCVV2>555</ns2:CardCVV2>
                        <ns2:Client>300</ns2:Client>

<ns2:CardExpirationDate>08/26</ns2:CardExpirationDate>

<ns2:CardHolderName>Worlpay Test</ns2:CardHolderName>
                        <ns2:Amount>123</ns2:Amount>
                        <ns2:CurrencyKey>USD</ns2:CurrencyKey>
                        <ns2:CardDataSource>E</ns2:CardDataSource>
                        <ns2:TaxLevel1>.01</ns2:TaxLevel1>
                        <ns2:TaxLevel2>.02</ns2:TaxLevel2>
                        <ns2:TaxLevel3>.03</ns2:TaxLevel3>
```

Example POST body

```
                        <ns2:TaxLevel4>.04</ns2:TaxLevel4>
                        <ns2:AdditionalInfo>X</ns2:AdditionalInfo>

<ns2:CardHolderAddress1>123 E Main St</ns2:CardHolderAddress1>

<ns2:CardHolderAddress2>Suite 101</ns2:CardHolderAddress2>
                        <ns2:CardHolderCity>Austin</ns2:CardHolderCity>

<ns2:CardHolderCountry>USA</ns2:CardHolderCountry>
                    </ns2:ITransactionHeader>
                </ns2:packets>
            </ns2:pPacketsIn>
        </ns2:SoapOp>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example response

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
    <s:Header>
        <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
            <u:Timestamp u:Id="_0">
                <u:Created>2025-01-31T16:16:35.038Z</u:Created>
                <u:Expires>2025-01-31T16:21:35.038Z</u:Expires>
            </u:Timestamp>
        </o:Security>
    </s:Header>
    <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <SoapOpResponse xmlns="http://Paymetric/XiPaySoap30/message/">
            <SoapOpResult>
                <count>1</count>
                <xipayvbresult>true</xipayvbresult>
                <packets>
                    <ITransactionHeader>
                        <AVSAddress />
                        <AVSCode>N2</AVSCode>
                        <AVSZipCode />
                        <AccountingDocNumber />
                        <ActionCode />
                        <AdditionalInfo>X</AdditionalInfo>
```

Example response

```
                        <Amount>123.0000</Amount>
                        <AuthorizationCode>tst555</AuthorizationCode>
                        <AuthorizationDate>2025-01-
31T10:16:34</AuthorizationDate>
                        <AuthorizationReferenceCode />
                        <AuthorizationTime>10:16:0034</AuthorizationTim
e>
                        <AuthorizedThroughCartridge>XiPayCartPTSalem.PT
Salem.1</AuthorizedThroughCartridge>
                        <BankBatchID />
                        <BankSubBatchID />
                        <BankTransactionID />
                        <BatchID />
                        <BillingDate>1899-12-30T00:00:00</BillingDate>
                        <BillingPlanItem />
                        <CaptureDate>1899-12-30T00:00:00</CaptureDate>
                        <CaptureReferenceCode />
                        <CardDataSource>E</CardDataSource>
                        <CardExpirationDate>08/26</CardExpirationDate>
                        <CardFollowOnNumber />
                        <CardHolderAddress1>123 E Main
St</CardHolderAddress1>
                        <CardHolderAddress2>Suite
101</CardHolderAddress2>
                        <CardHolderCity>Austin</CardHolderCity>
                        <CardHolderCountry>USA</CardHolderCountry>
                        <CardHolderDistrict />
                        <CardHolderName1 />
                        <CardHolderName2 />
                        <CardHolderName>Worlpay Test</CardHolderName>
                        <CardHolderState />
                        <CardHolderZip />
                        <CardNumber>CardNumberValue</CardNumber>
                        <CardPresent>0</CardPresent>
                        <CardType>VI</CardType>
                        <CardValidFrom />
                        <ChargeAmount>0.0000</ChargeAmount>
                        <Client>300</Client>
                        <CompanyCode />
                        <CreationDate>2025-01-
31T10:16:34</CreationDate>
                        <CurrencyKey>USD</CurrencyKey>
                        <CustTXN />
                        <CustomerNumber />
                        <FiscalYear />
                        <GLAccount />
```

Example response

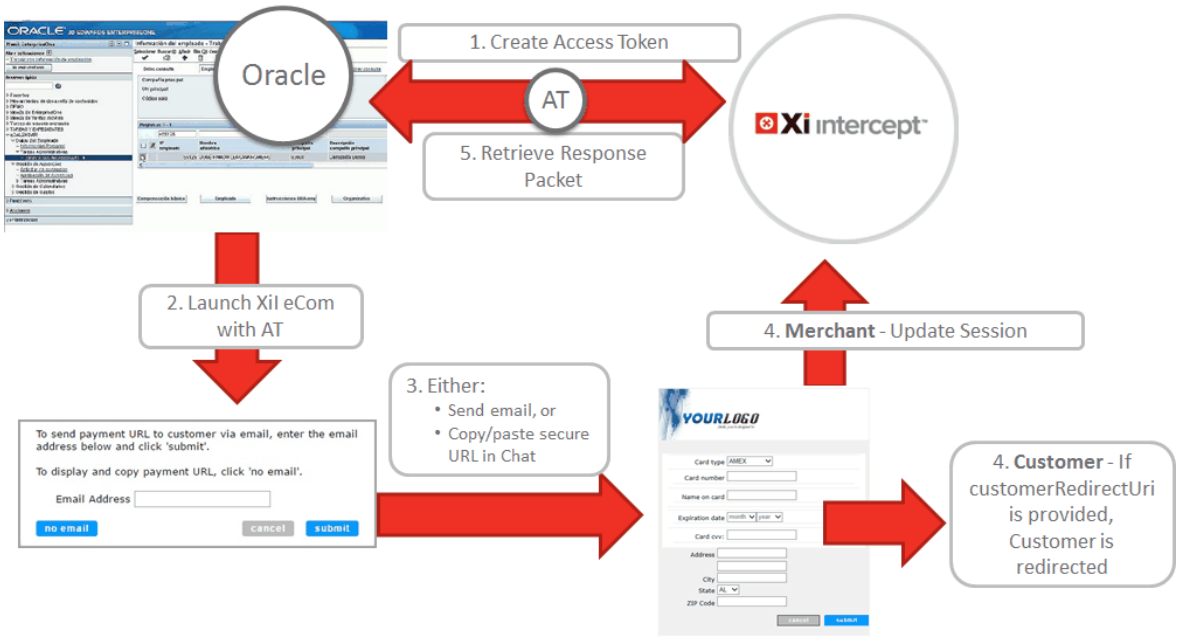
```
                        <LastModificationDate>2025-01-
31T10:16:35</LastModificationDate>
                        <LocationID />
                        <MerchantID>PMSalem</MerchantID>
                        <MerchantTXN />
                        <MerchantTransactionID />
                        <Message>100 Approved - Successfully
approved.</Message>
                        <ModifiedStatus>1</ModifiedStatus>
                        <OrderDate>1899-12-30T00:00:00</OrderDate>
                        <OrderID />
                        <Origin>XiPay</Origin>
                        <PONumber />
                        <PacketOperation>1</PacketOperation>
                        <Preauthorized />
                        <ReferenceCode />
                        <ReferenceLineItem />
                        <ResponseCode>100</ResponseCode>
                        <SalesDocNumber />
                        <SettlementAmount>123.0000</SettlementAmount>
                        <SettlementDate>1899-12-
30T00:00:00</SettlementDate>
                        <SettlementReferenceCode />
                        <ShippingCaptureDate>1899-12-
30T00:00:00</ShippingCaptureDate>
                        <ShippingLocationID />
                        <ShippingMethod />
                        <StatusCode>100</StatusCode>
                        <StatusTXN>Authorized</StatusTXN>
                        <TaxLevel1>0.0100</TaxLevel1>
                        <TaxLevel2>0.0200</TaxLevel2>
                        <TaxLevel3>0.0300</TaxLevel3>
                        <TaxLevel4>0.0400</TaxLevel4>
                        <TerminalID />
                        <TransactionID>22222222</TransactionID>
                        <TransactionType>Authorization</TransactionType
>
                        <VATNumberCustomer />
                        <VATNumberMerchant />
                        <XIID>1111111</XIID>
                    </ITransactionHeader>
                </packets>
            </SoapOpResult>
        </SoapOpResponse>
    </s:Body>
</s:Envelope>
```

11 XiSecurelink integration

Allows the Merchant Representative to send an email that contains a secure link to the Intercept interface hosted in B2B Payment's SaaS environment. The email recipient clicks the link to launch B2B Intercept and then enters the credit card information. The credit card details and a token are returned directly to the ERP system.

Optionally, the Merchant Representative can choose to obtain the secure Paymetric hosted URL and copy/paste it into a chat window or other secure form of communication with the Customer.

For Oracle, other web stores or non-SAP ERP systems, XiSecurelink is integrated by adding the `<emailPaymentOption>` element in the XML packet for the get access token call that is made to Intercept eComm.



1. A direct server-side POST is required to retrieve the Access Token. How this POST is performed depends on the language you are using. The content type must be "application/x-www-form-urlencoded''.

2. Intercept for eCommerce Hosted UI is launched in the default browser with the AccessToken and the merchant representative is presented with two options to obtain the payment information:

   A. A field to enter an email address for the customer.

   B. A button to generate the secure URL that can be copied.

3. Depending upon which option is chosen in Step 2,

   A. The email is sent with the link that launches Intercept hosted page. The customer enters payment details.

   B. Or, the merchant representative sends URL via chat or other method. The customer enters payment details.

4. The token is obtained.

5. Response packet is obtained via a server-side GET call back into Oracle.

Note that other non-SAP ERP or applications/systems have the same workflow as defined for Oracle.

11.1 emailPaymentOption element attributes

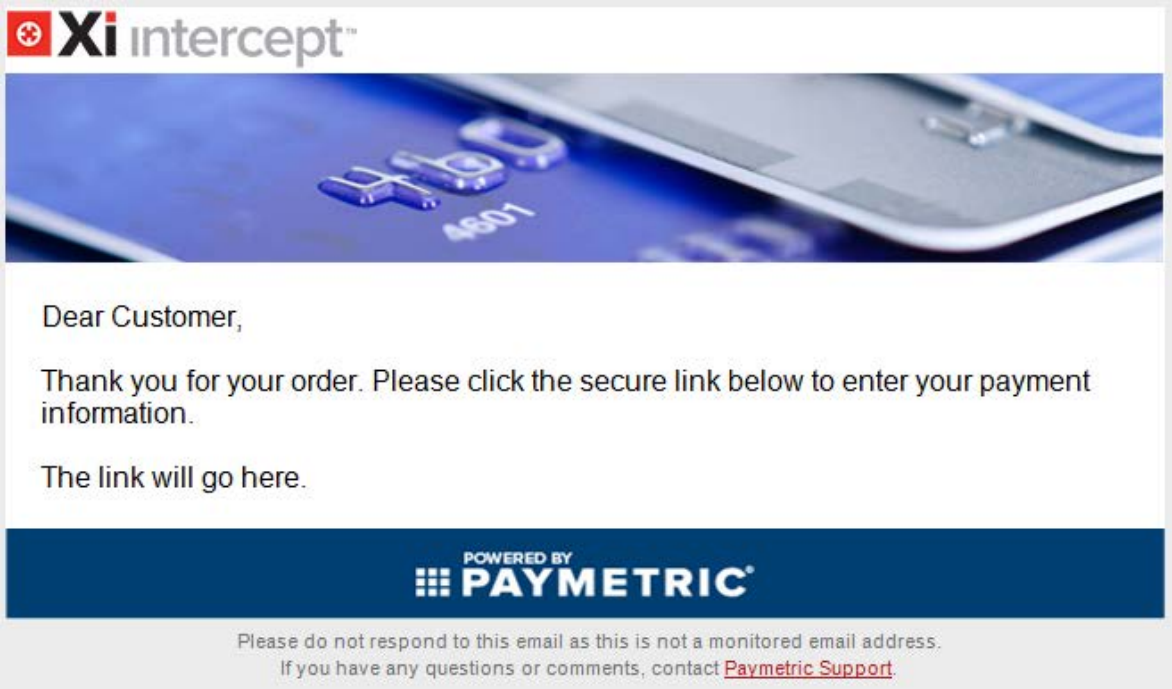
The following table defines the available `<emailPaymentOption>` attributes:

Element	Description
emailTemplate	The text for the email body. Use [BR] to insert a line break as needed.

Element	Description
	The following two variables are available to add the URL link in the body. Choose which to use depending upon how much control you want over the actual link. [PAYMENTURL] Is replaced with the appropriate QA or PROD URL based upon which system you are calling. [PAYMENTLINK] Is replaced with the following where PAYMENTURL is QA or PROD based on which system you are calling. `<a href='[PAYMENTURL]'>Enter Payment Information</a>`
merchantLogoURL	Enter the URL for your custom logo. If this attribute is not provided, Intercept logo displays by default.
fromEmailAddress	Defines the from address for the email. Defaults to noreply@worldpay.com.
customerRedirectUri	The URL to which the Customer is redirected once he/she clicks submit after the entering the payment information.

11.2 Default email text

If you do not define any of the `<emailPaymentOption>` attributes, following is an example of the default email that will be send if you choose the email option.



11.3 XML packet example with emailPaymentOption element

The Intercept URLs to invoke XiSecurelink call are as follow:

Cert

https://cert-xiecomm.worldpay.com/diecomm/diecomm/paymentpage/startemailpayment/<merchantGuid>/<accessToken>

Production

https://xiecomm.worldpay.com/diecomm/paymentpage/startemailpayment/<merchantGuid>/<accessToken>

Following is an example of the XML packet with the <emailPaymentOption> element highlighted.

```
<?xml version="1.0" encoding="utf-8"?>
<merchantHtmlPacketModel
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
   <paymentPagePacket>
       <redirectUri>http://dh-ecomm-
w2k12.pmdev.local/InterceptDemo/Status.aspx</redirectUri>
       <cssUri>http://dh-ecomm-
w2k12.pmdev.local/DIeComm/Content/PaymentStyleSheet.css</cssUri>
       <emailPaymentOption
emailTemplate="Thank you for your interest.[BR]Please click the
following link to continue.[BR][PAYMENTLINK][BR][BR]If you are unable to
click the link, copy/paste this URL into your browser: [PAYMENTURL][BR]
Regards,[BR]My Company, Inc.[BR] 1-800-555-5555"
merchantLogoUrl="http://img13.mysite.net/images/mylogo.png"
fromEmailAddress="noreply@company.com"
customerRedirectUri="https://mycompany.com" />
   </paymentPagePacket>
   <merchantHtml>
       <htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
           <tag name="div" class="merchant-logo"/>
           <tag name="div" class="billing-content">
               <tag name="div" class="billing-header">Billing
Information</tag>
               <tag name="div" class="billing-info">
                   <paymentOptionRadioSection>
                       <tag name="div">
                           <tag name="div">choose payment type</tag>
                       </tag>
                       <label text="echeck" for="echeck" />
```

```
                        <radPaymentOption for="echeck" />
                        <label text="credit card" for="card" />
                        <radPaymentOption for="card" />
                    </paymentOptionRadioSection>
                    <cardDropdownSection>
                        <tag name="div">Enter your credit card
information</tag>
                        <tag name="div">
                            <label for="cardType" text="select a card" />
                            <ddlCardType id="cd">
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <customItem for="MSC" text="My Super
Card" />
                                </items>
                            </ddlCardType>
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="card number" />
                            <tboxCardNumber tokenize="true" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="name on
card" />
                            <tboxCardHolderName />
                        </tag>
                        <tag name="div">
                            <label for="expMonth" text="exp date" />
                            <ddlExpMonth default-text="month"
class="merchant_combos" />
                            <ddlExpYear default-text="year"
class="merchant_combos" years-to-display="10" />
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="card cvv:" />
                            <tboxCvv />
                            <htmlCvvHelp name="div" raise-click="1"/>
                        </tag>
                    </cardDropdownSection>
                    <echeckRadioSection>
                        <tag name="div">
                            <label for="checking" />
                            <radAccountType for="checking" />
                            <label for="savings" />
                            <radAccountType for="savings" />
                        </tag>
                        <tag name="div">
```

```
                            <label for="accountNumber" text="account
number" />
                            <tboxAccountNumber tokenize="true" />
                        </tag>
                        <tag name="div">
                            <label for="routingNumber" text="routing
number" />
                            <tboxRoutingNumber />
                        </tag>
                        <tag name="div">
                            <label for="nameOnAccount" text="name on
account" />
                            <tboxNameOnAccount />
                        </tag>
                        <tag name="div">
                            <label for="bankName" text="bank name" />
                            <tboxBankName />
                        </tag>
                    </echeckRadioSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-button" />
                    <submitButton text="submit" class="submit-button" />
                </tag>
            </tag>
        </htmlSection>
    </merchantHtml>
</merchantHtmlPacketModel>
```

For Iframe and Hosted integration types, there are three main options associated with localization as follows:

- Do not specify any localization which then defaults to U.S. English
- Specify to use the Browser locale; if it is a language not supported by Intercept eComm, then it defaults to U.S. English
- Hard code the locale using the Intercept eComm supported languages/culture codes

The following table defines the languages currently supported by Intercept eComm along with the culture codes to use if you are hard coding the localization element. Reminder, when using browser localization, if the browser locale is not in the following list, then the Intercept eComm labels and messaging will default to U.S. English unless hard-coded in custom HTML text or message attributes.

Language	"cultureName" value
Czech	cs
Danish	da
German	de
Spanish (European)	es
Spanish (Latin)	es-es
Finnish	fi
French (Native)	fr

Language	"cultureName" value
French (Canadian)	fr-ca
Hungarian	hu
Italian	it
Japanese	ja
Korean	ko
Dutch	nl
Flemish	nl-be
Norwegian	no
Polish	pl
Portuguese (Portugal)	pt
Portuguese (Brazil)	pt-br
Russian	ru
Slovak	sk
Slovenian	sl
Swedish	sv

Language	"cultureName" value
Turkish	tr
Chinese (Simplified)	zh
Chinese (Traditional)	zh-tw

13 Supported card types and returned values

Defines the supported card types and the values returned from Intercept eComm:

Card type	Returned value
Visa	VI
MasterCard	MC
American Express	AX
Diner's	DC
Discover	DI
JCB	JC
Maestro	SW

All other card types are considered custom cards. The values are set in the XML.

14 Status codes

Error code	Description
-100	Indicates an unexpected error occurred processing the server request
-101	Indicates the web request could not be created
-102	Indicates the attempt to retrieve the response failed
-103	Indicates no response packet was returned from the server
-104	Indicates the response packet was invalid and could not be deserialized
100	Internal DI eComm error
101	Merchant GUID is invalid
102	Signature validation failed
103	Request packet is invalid
104	DI eComm session has expired
105	DI eComm session error
106	Error decrypting data
107	Get AccessToken error
108	Requested type hasn't been purchased

Error code	Description
120	XiSecure error
121	XiSecure no token
122	XiSecure certificate expired UID: {0}, Expiration Date {1}
140	Error accessing database
150	The payment form has been submitted multiple times with the same access token. (I.e. user clicks submit payment twice)
300	Cardinal Commerce response: {message provided by Cardinal}
301	No Cardinal Commerce configuration
302	The Cardinal Commerce configuration is missing information
401	An attempt was made to load the payment page in an Iframe
805	TokenProviderConfigError - Token provider configuration error
806	TokenFormSubmissionError - Tokenization form submission error
810	TesNoTokenProviderDefined - No token provider defined
820	TesIntegrationNotSupported - TES integration not supported

Error code	Description
830	TesIntegrationPackeInvalid - TES integration packet invalid: {0}
840	TesTokenizationError - TES tokenization error: {0}

15 JavaScript Plugin API reference

Section contents:

15.1 XIPlugin - Form Post and Ajax API reference

The XIPlugin JavaScript plugin is used to reach out to the Intercept eComm service by Ajax or Form POST. XIPlugin can either be referenced directly from the Intercept service or downloaded and then referenced from your server.

Reference or download the following:

`https://xiecomm.worldpay.com/DIeComm/Scripts/XIPlugin/XIPlugin-1.0.0.js`

15.1.1 Formatted JS data structure

To perform the Ajax or Form POST call to Intercept eComm, you must first create your data packet. The XIPlugin can assist in preparing this data packet.

Section contents:

- $XIPlugin.createField(fieldName, tokenize, value) 107
- XiRequestPacket - Ajax and Form Post 107

15.1.1.1 $XIPlugin.createJSRequestPacket(merchantGuid, accessToken)

Creates an empty data packet (XiRequestPacket).

Parameters

`merchantGuid`
The Merchant GUID generated when the Intercept for eCommerce Onboarding request is processed. Login to Merchant Portal and navigate to Settings | Intercept for eCommerce to obtain the GUID.

`accessToken`
The AccessToken obtained during the Get AccessToken call.

Returned from the function

An instance of the XiRequestPacket data packet is returned.

15.1.1.2 $XIPlugin.createField(fieldName, tokenize, value)

Parameters

`fieldName`
Name of the field associated to the value. This will identify the field in the return packet.

`tokenize`
The Access Token obtained during the Get Access Token call. Boolean value that specifies that this value needs to be tokenized

`value`
The value associated with the field name

Returned from the function

An instance of the XiField data field is returned to add to a data packet.

15.1.1.3 XiRequestPacket - Ajax and Form Post

The data packet sent to Ajax or Post methods of XiPlugin.

`XiRequestPacket.addField`

This adds an XiField object to the data packet.

15.1.2 Communication method details - Ajax and Form Post

Section contents:

15.1.2.1 $XIPlugin.ajax

```
$XIPlugin.ajax({

     url: '<% Intercept Url%>' + "/Ajax",
     type: "POST", //<GET/POST>
     data: '<% Formatted JS Data – see Packaging above %>',
     success: function(a)
                    {
                    },
     error: function(a)
                    {
                    },
     beforeSend: function(a)
                    {
                    },
     complete: function(a)
                    {
                    },
      validation: function(a)
                    {
                       return true; //should return true or false.
                    }
     intcp3DSecure: function(a)
                    {
                    },

      });
```

Parameters

```
url: '<% Intercept Url%>' + "/Ajax",
```

Define Intercept eComm url and append "/Ajax". Use the appropriate QA or Production URL.

QA: https://cert-xiecomm.worldpay.com/DIeComm

Production: https://xiecomm.worldpay.com/DIeComm

```
type: "POST",
```

Call Type should be POST.

```
data: '<% Formatted JS Data – see Packaging above %>',
```

Set the Data element as defined in the Packaging 105 section above.

```
success: function(a)
```

The function that is triggered if the call is successful.

```
error: function(a)
```

The function that is triggered if the call is NOT successful.

```
beforeSend: function(a)
```

The function that is triggered before the call is sent to Intercept eComm.

```
complete: function(a)
```

The function that is triggered after the Ajax call is complete whether is successful or not .

```
validation: function(a)
   {
        return true; //should return true or false.
     }
```

The function used to perform custom validations. It returns either true false.

If True, will go through with ajax call.

If False, will not go through with ajax call.

```
intcp3DSecure: function(a)
```

Function triggered when a card is enrolled in 3DSecure for customizing Cardinal display in an Iframe or Pop up instead of redirecting to Cardinal site. If this function in not intercepted then default behavior is invoked in which the site is redirected to Cardinal site.

Function Parameters:

- args has necessary argument to post the form to //3DSecure cardinal.
- UrlsAndPayloadsFor3Ds.CentinelAcsUrl → form **action**
- UrlsAndPayloadsFor3Ds.CentinelPayload → hidden //input name: **PaReq**
- UrlsAndPayloadsFor3Ds.CentinelTermUrl → hidden //input name: **TermUrl**
- e.UrlsAndPayloadsFor3Ds.PaymetricPayload → hidden //input name: **MD**

15.1.2.2 $XIPlugin.submit

```
$XIPlugin.submit({
```

```
url: '<% Intercept Url%>' + "/Form",
type: "POST", //<GET/POST>
data: '<% Formatted JS Data – See Packaging above %>',
validation: function(a)
                    {
                      return true; //should return true or false.

                    }


    });
```

Parameters

`url: '<% Intercept Url%>' + "/Form",`
Define Intercept eComm url and append "/Form". Use the appropriate QA or Production URL. QA: https://cert-xiecomm.worldpay.com/DIeComm Production: https://xiecomm.worldpay.com/DIeComm

`type: "POST",`
Call Type should be POST.

`data: '<% Formatted JS Data – see Packaging above %>',`
Set the Data element as defined in the Packaging 105 section above.

`validation: function(a)`
`{`
`return true; //should return true or false.`
`}`

The function used to perform custom validations. It returns either true false.

If True, will go through with post.

If False, will not go through with post.

15.1.3 Examples

```
var cc = document.getElementById('PaymentCreditCard').value;
var name = document.getElementById('Name').value;
```

Get value from TextBox with JavaScript code

```
var myData = $XIPlugin.createJSRequestPacket(merchantGuid, accessToken);
```

Create a JavaScript object to be used in Ajax or Form Post with parameters:

- MerchantGuid
- AccessToken

```
var myField = $XIPlugin.createField('creditcard', true, cc);
```

Create a JavaScript object to be added to myData (JSRequestPacket)

- First Param : Field Name (This is up to the Merchant. Any possible name)
- Second Param : true (To Tokenize Value)
- Third Param : Value from TextBox

```
myData.addField(myField);
```

Add myField to myData (JSRequestPacket) Object

Ajax call

```
$XIPlugin.ajax({

      url: '<% Intercept Url%>' + "/Ajax",
      type: "POST", //<GET/POST>
      data: '<% Formatted JS Data – see above %>',
      success: function(e)
            {
             //function triggered on success.
            },
      error: function(e)
            {
             //function triggered on failure.
            },
      beforeSend: function(e)
            {
             //function triggered before ajax call.
            },
      complete: function(e)
            {
             //function triggered after ajax call.
             // After success or failure
            },
      validation: function(e)
            {
             //Run custom validation
             return true; //should return true or false.

             // return true will go through with ajax call.
             // return false will not go through with ajax call.
            },
      intcp3DSecure: function(e)
            {
                        //Function triggered when a card is enrolled in
                        3DSecure for customizing Cardinal display in an
                        Iframe or Pop up instead of redirecting to
                        Cardinal site. If this function in not
                        intercepted then default behavior is invoked in
                        which site is redirected to Cardinal site.

                        //e – args has necessary argument to post the
                        form to 3DSecure cardinal.

                        //e.UrlsAndPayloadsFor3Ds.CentinelAcsUrlàform
                        action

                        //e.UrlsAndPayloadsFor3Ds.CentinelPayloadàhidden
                        input name: PaReq
```

```
                        //e.UrlsAndPayloadsFor3Ds.CentinelTermUrlàhidden
                        input name: TermUrl

                        //e.UrlsAndPayloadsFor3Ds.PaymetricPayloadàhidde
                        n input name: MD

            }

      });
```

Form Post call

```
$XIPlugin.submit({

         url: '<% Intercept Url%>' + "/Form",
         data: '<% Formatted JS Data – See above %>',
         validation: function(a)
               {
                        //Run custom validation
                        return true; //should return true or false.

                        // return true will go through with post.
                        // return false will not go through with post.
                     }


          });
```

15.2 XIFrame - IFrame API reference

This javascript plugin is used for reaching out Intercept service by embedded Iframe in your website. XIFrame can either be referenced directly from Intercept service or downloaded from the Intercept service to your server and then referenced from your server.

Reference or download the following:

https://xiecomm.worldpay.com/DIeComm/Scripts/XIFrame/XIFrame-1.1.0.js

Section contents:

15.2.1 $XIFrame.onload

Resizing of the Iframe can be handled by a stylesheet. If you need to resize (based upon content within the Iframe), include the following javascript function on Iframe onload function.

$XIFrame.onload must be called after the Iframe renders the content from Intercept eComm. It initializes the communication from the parent page to the Iframe.

```
function Iframe_OnLoad(){
$XIFrame.onload ({
      iFrameId: <% Iframe Name or ID %>,
   targetUrl: <%Intercept Url%>/View/Iframe/<%MerchantGuid%>/<%
AccessToken%>
   autosizeheight: true,
   autosizewidth: true,
   storeCardAmount: true,
   onCardTypeChange: function (e){},
onValidate: function (e){},
onLoadSuccess: function (){},
onError: function(msg) {},
onInvalidHandler: function (e){},

   amount: 30.99  //Can pass amount outside of iframe.

   });
}
```

$XIFrame.onload parameters

iFrameID

Name or ID of the Iframe.

`targetUrl`
Name or ID of the Iframe.

`autosizeheight`
Specifies whether to resize the height of the Iframe if the content in the Iframe increases. If this is set to false, the Iframe must be large enough to handle all content even with validation messages.

`autosizewidth`
Specifies whether to resize the width of the Iframe if the content in the Iframe increases. If this is set to false, the Iframe must be large enough to handle all content even with validation messages

`storeCardAmount`
Boolean specifying to return Amount from Iframe to parent window when $XIFrame.getCardAmount() is invoked.

`onCardTypeChange`
Callback function triggered when Card Type is changed amount.

onValidate
Function that should be explicitly called to perform any validation defined within the IFramePacket with (e) as the argument if iframe passes or fails validation.

onLoadSuccess
Callback function triggered when the Iframe is correctly rendered.

onError
Callback function triggered if there are any problems rendering the content of the Iframe.

onInvalidHandler
This function results in reports on controls failing any Intercept eComm validations with errors.

amount
Used to change default amount passed to 3D Secure for validation. It allows you to pass the amount outside of the Iframe.

Best practice related to page/Iframe Onload

To ensure the Iframe loads the correct URL with a valid AccessToken, see the following functions. Note that this prevents problems that can occur when a User clicks the back button after the Submit.

This function should reside on the page frontend outside of any other functions.

```
window.onload = function () {
      var frameElement = document.getElementsByName('<% Iframe Name or ID
%>')[0];
      frameElement.contentWindow.location.href = frameElement.src;
    };
    window.onunload = function () { }
```

This function should reside on the page backend.

```
protected override void OnInit(EventArgs e)
  {
    Response.Cache.SetCacheability(System.Web.HttpCacheability.NoCache);
    Response.Cache.SetNoStore();
    base.OnInit(e);
  }
```

15.2.2 $XIFrame.submit

```
$XIFrame.submit({
    iFrameId: <% Iframe Name or ID %>,
    targetUrl: <%Intercept Url%>/View/Iframe/<%MerchantGuid%>/<%AccessToken
%>
    onSuccess: function (e) {},
    onError: function (e) {},
intcp3DSecure: function(e) {},

 });
```

Parameters

iFrameID
Name or ID of the Iframe.

targetUrl

Name or ID of the Iframe.

`success: function(a)`
The function that is triggered if the call is successful.

`error: function(a)`
The function that is triggered if the call is NOT successful.

15.2.3 Multiple IFrames example

If you are using multiple IFrames, you must create the same number of $XIFrame instances. The following example demonstrates using two IFrames to split a payment:

```
var iFrameName1 = 'IFrame1';
var iFrameName2 = 'IFrame2';
var iframe1 = document.getElementsByName(iFrameName1);
var iframe2 = document.getElementsByName(iFrameName2);
```

Hold on to the XIFrame instance 1

```
var iframe1Plugin = $XIFrame({
        iFrameId: iFrameName1,
        targetUrl: $(iframe1).attr("src"),
        onCardTypeChange: IFrame1CardChange,
        autosizeheight: true,
        autosizewidth: true,
        onError: function(msg) { alert(msg); },
        onValidate: Validation,
        storeCardAmount: true,
        intcp3DSecure: Custom1Handling
    });
```

Hold on to the XIFrame instance 2

```
var iframe2Plugin = $XIFrame({
  iFrameId: iFrameName2,
  targetUrl: $(iframe2).attr("src"),
  autosizeheight: true,
  autosizewidth: true,
  onError: function (msg) { alert(msg); },
  onValidate: Validation,
  storeCardAmount: true,
  intcp3DSecure: Custom2Handling
});
```

Holding on to the instances assist in making the future call as follows:

1. `iframe1Plugin.onload(); or iframe1Plugin.submit({onSuccess: function (e) { //trigger on iframe1 submit success }});`
2. `iframe2Plugin.onload();or iframe2Plugin.submit({onSuccess: function (e) { //trigger on iframe1 submit success }});`

15.2.4 $XIFrame.validate

Use this function is if you are using multiple IFrames.

```
function submitform() {
      var iframe = document.getElementsByName('dieCommFrame');
      if (iframe) {
        $XIFrame.validate({
          iFrameId: 'dieCommFrame',
          targetUrl: iframe[0].getAttribute("src"),
          autosizeheight: true,
          autosizewidth: true,
          onValidate: function(e) {}

        });
   }
```

Parameters

`iFrameID`

Name or ID of the Iframe.

`targetUrl`
Name or ID of the Iframe.

`autosizeheight`
Specifies whether to resize the height of the Iframe if the content in the Iframe increases. If this is set to false, the Iframe must be large enough to handle all content even with validation messages.

`autosizewidth`
Specifies whether to resize the width of the Iframe if the content in the Iframe increases. If this is set to false, the Iframe must be large enough to handle all content even with validation messages

`onValidate`
Function that should be explicitly called to perform any validation defined within the IFramePacket with (e) as the argument if iframe passes or fails validation.

`onError`
Callback function triggered if there are any problems rendering the content of the Iframe.

Example using onValidate

```
function submitform() {
      var iframe = document.getElementsByName('dieCommFrame');
      if (iframe) {
        $XIFrame.validate({
          iFrameId: 'dieCommFrame',
          targetUrl: iframe[0].getAttribute("src"),
          autosizeheight: true,
          autosizewidth: true,
          onValidate: function(e) {
            if (e) {
              $XIFrame.submit({
                iFrameId: 'dieCommFrame',
                targetUrl: iframe[0].getAttribute("src"),
                onSuccess: function(msg) {
                  var message = JSON.parse(msg);
                  if (message && message.data.HasPassed) {
                    var accessToken = document.getElementById('<%=
AccessToken.ClientID %>');
                    var signedToken = document.getElementById('<%=
SignedToken.ClientID %>');
                    window.location = "Status.aspx?id=" +
accessToken.value + "&s=" + signedToken.value;
                  } else {
                    alert(message.data.Message);
                  }
                },
                onError: function(msg) {
                  alert("Error function : " + msg);
                }
              });
            }
 else {
              alert('Iframe Validation failed');
            }
          },
          onError: function(msg) {
            alert("Error function : " + msg);
          }

        });
   }
```

16 Custom HTML reference

Merchants can define custom html layouts for the Intercept for eCommerce (Intercept eComm) **Iframe** and **Hosted** solutions. This is done by creating XML that adheres to the Paymetric schema ***MerchantHtmlPacketModel.xsd***.

16.1 Element: <htmlFieldValues>

Defines default values for textboxes.

Child elements

- field: applicable to textboxes only such as cardholder name or echeck. This is limited to the textboxes:
  - cardHolderName (credit card)
  - cardAmount (credit card)
  - bankName (echeck)
  - nameOnAccount (echeck)
- additionalField: can be used for any merchant-defined custom textbox

16.2 Element: <merchantHtml>

The XML consists of a main outer container defined by the **<merchantHtml>** element. Depending on the merchant's needs, this element can contain one or more of the following types of child sections:

- **Payment Option Section** – Allows the web user to choose between eCheck and Credit Card payment types
- **Credit Card Section** – Provides inputs for credit card information
- **eCheck Section** – Provides inputs for eCheck information
- **Additional HTML Section** – Provides inputs for additional data the merchant wants collected. For example, address data can be collected in this type of section

Each section above defines elements for labels and inputs to capture data.

Default validations (such as required field, Luhn check for credit card numbers) are defined for the input elements. Additionally, merchants can override these defaults and/or provide custom validations for any input element.

Merchants can make use of common HTML elements such as DIVs to further control the layout of any section, label or input in the xml.

The following sections provide descriptions of the elements defined in the schema.

16.2.1 Common elements and attributes

Descriptions of the common elements and attributes used throughout the schema.

16.2.1.1 Common attributes

Common attributes used throughout the custom html:

- **name**: Unless otherwise specified, this is the name of the html element to generate. Supported values are "div", "span" and "section". The default value is "div"
- **class**: This is the CSS class for the element

> ⚠ **Note:**
>
> The class attribute should be used to style elements. Other element attributes (such as id, name, etc) are internally generated values that are subject to change in the future. The merchant's CSS should not reference those attributes.

- **for**: This attribute is used to indicate a relationship between elements. For example, a "label' has a "for" attribute indicating what input it is linked to. An input has a "for" attribute indicating what data the input is for
- **is-default**: For radio and dropdown inputs, if "1" then this item is the default value
- **text**: For labels, indicates the text to display. If no text is displayed, the default is used for that label. Note this is not currently supported for default text in an input text box

- **default-text**: For dropdown inputs, indicates the default text to display. For example, in a credit card type dropdown, if "display-text" is "select a card", then "select a card" is the text in the dropdown when the page loads
- **tokenize**: Indicates if the data collected in the input should be tokenized. For credit card number and eCheck account number, this must be "true"

16.2.1.2 Common attributes for validation

In addition to the attributes above, these are attributes for input elements that control validation:

- **error-class**: The CSS class to use on the input element if validation fails.

> **Note:**
>
> The class attribute should be used to style elements. Other element attributes (such as id, name, etc) are internally generated values that are subject to change in the future. The merchant's CSS should not reference those attributes.

- **error-msg-style**: Supported values are:
  - **text**: The error message displays as text
  - **imageTooltip**: The error message displays an image with a tooltip

  - **inputTooltip**: The error message displays as a tooltip when hovering over the input

- **digits-only**: If "true", only digits can be entered
- **digits-only-msg**: The error message displayed for digits-only validation
- **luhn-check**: If "true", a luhn check is performed. Applicable to card numbers only
- **luhn-check-msg:** The error message displayed for luhn check validation.
- **max**: If "true", the maximum numeric value allowed
- **max-msg**: The error message displayed for max validation
- **min**: The minimum numeric value allowed
- **min-msg**: The error message displayed for min validation
- **maxlength**: The maximum text length allowed
- **maxlength-msg**: The error message displayed for maxlength validation
- **minlength**: The minimum text length allowed
- **minlength-msg**: The error message displayed for minlength validation
- **number**: If "true", only a number can be entered
- **number-msg**: The error message for number validation
- **pattern**: A regular expression used to validate the input
- **pattern-msg**: The error message for pattern validation
- **range**: A numeric range for the input

- **range-msg**: The error message for range validation
- **rangelength**: A range indicating the length of the input
- **rangelength-msg**: The error message for range-length validation
- **required**: If "true", indicates input is required (cannot be blank)
- **required-msg**: The error message for required validation

16.2.1.3 Element: <tag>

Indicates whether an HTML tag should be written. The most common use is to use a <tag> anywhere an HTML "div" element is desired

Attributes

- name ("div", "span", or "section")
- class

Child elements allowed

- <tag>'s can contain any child element supported by its ***parent*** section
- <tag>'s can also contain inner text

16.2.1.4 Element: <label>

Defines a label associated with an input.

Attributes

- name: Name of the html element to generate. Supported values are “div”, “span” and “label”. The default is “label”
- class
- text: Indicates what text is applied to the generated html element. If no text is provided, the default is used
- for: Indicates which input is linked to the label

Child elements allowed

- None

16.2.1.5 Element: <voidTag>

Indicates whether an HTML void tag should be written. A void tag is a tag that cannot contain content.

Attributes

- name: The only allowed value is “br” (generates an html
 element)

Child elements allowed

- None

16.2.1.6 Elements: <submitButton> and <cancelButton>

These elements are used by the Hosted solution to either submit or cancel the input and redirect to the merchant.

These button elements ***cannot*** be used in an iFrame solution. The merchant is required to provide their own submit/cancel framework using the JavaScript plugin.

Attributes

- text: the button text displayed
- class

Child elements allowed

- None

16.2.1.7 Element: <validationMsg>

Provides control over the layout and display of validation error messages. If no <validationMsg> is used, then the default (JQuery) behavior is used to display the message immediately following the input control.

Use this element to create a separate html element for displaying validation messages or error images.

Attributes

- name ("div", "span", or "section")
- class
- for (the input the validation is for. The sections below define the valid values for this)
- error-class: the CSS class used on the validation message when a validation error exists

Child elements allowed

- None

> ⚠ **Note:**
>
> The class and error-class attributes should be used to style elements. Other element attributes (such as id, name, etc) are internally generated values that are subject to change in the future. The merchant's CSS should not reference those attributes.

16.2.2 Main container section

16.2.2.1 Element: <htmlSection>

The outermost container (div) for the custom defined HTML. All other sections and elements in the XML are defined within this section.

Attributes

- name
- class
- xmlns="Paymetric:Intercept:MerchantHtmlPacketModel"

Child elements allowed

- <xiProperties>
- <paymentOptionRadioSection>
- <paymentOptionExpandedRadioSection>
- <paymentOptionDropdownSection>
- <paymentOptionExpandedDropdownSection>
- <cardRadioSection>
- <cardDropdownSection>
- <cardNoTypeSection>
- <echeckRadioSection>
- <echeckDropdownSection>

- <cancelButton> (Hosted page only)
- <submitButton> (Hosted page only)
- <tag>
- <voidTag>
- <additionalHtmlSection>
- <textbox>
- <label>

16.2.3 XiProperties section

This section defines global properties that can be applied to all elements.

16.2.3.1 Element: <xiProperties>

Container element for global xiProperties.

Attributes

- error-class: Global CSS error class to use on validation messages

> **Note:**
>
> The class attribute should be used to style elements. Other element attributes (such as id, name, etc) are internally generated values that are subject to change in the future. The merchant's CSS should not reference those attributes.

- inline-validation: Provides the ability to perform inline validation as soon as Iframe loads

Child elements allowed

- <errorTooltip>

16.2.3.1.1 Child element: <errorTooltip>

Defines effects and attributes when displaying an error tooltip.

Attributes

- class
- hide-duration: String or number (in ms) determining how long the animation will run. Refer to http://api.jqueryui.com/hide/ for detailed information
- hide-effect: Refer to http://api.jqueryui.com/category/effects/ for supported jquery vales
- show-duration: String or number (in ms) determining how long the animation will run. Refer to http://api.jqueryui.com/hide/ for detailed information
- show-effect: Refer to http://api.jqueryui.com/category/effects/ for supported jquery values
- position-at: Defines which position on the target element to align the positioned element against. Refer to http://api.jqueryui.com/position for supported jquery values
- position-collision
- position-my: Defines which position on the element being positioned to align with the target element. Refer to http://api.jqueryui.com/position for supported jquery values

Child elements allowed

- None

16.2.4 Payment option section

The payment option section is used to allow the web user to select the type of payment method – eCheck or Credit Card. If the merchant only supports eCheck or Credit Card (not both) then this section should be omitted and only the eCheck or Credit Card section needs to be defined in the XML.

There are four types of payment option sections as follows:

- Element: <paymentOptionRadioSection> 134
- Element: <paymentOptionDropdownSection> 136
- Element: <paymentOptionExpandedRadioSection> 137
- Element: <paymentOptionExpandedDropdownSection> 139

16.2.4.1 Element: <paymentOptionRadioSection>

Displays the payment options of “echeck” and “credit card” using radio buttons.

Attributes

- name
- class

Child elements allowed

- <label>
- <radPaymentOption>
- <tag>
- <voidTag>

16.2.4.1.1 Child element: <radPaymentOption>

Defines a radio button within the <paymentOptionRadioSection>

Attributes

- for (***required***) : value can be “echeck” or “card”
- class
- is-default

Child elements allowed

- None

16.2.4.1.2 Child element: <label>

Defines a label within the <paymentOptionRadioSection>

Attributes

- for (***required***) : value can be “echeck” or “card”
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.4.2 Element: <paymentOptionDropdownSection>

Displays the payment options of "echeck" and "credit card" using a dropdown.

Attributes

- name
- class

Child elements allowed

- <label>
- <ddlPaymentOption>
- <tag>
- <voidTag>

16.2.4.2.1 Child element: <ddlPaymentOption>

Defines a dropdown within the <paymentOptionDropdownSection>

Attributes

- class
- default-text
- required
- required-msg
- error-class
  - **Note**: The class attribute should be used to style elements. Other element attributes (such as id, name, etc) are internally generated values that are subject to change in the future. The merchant's CSS should not reference those attributes.
- error-msg-style

Child elements allowed

- <items>: Contains a list of **<item>** elements

- <item>
  - Attributes:
    - for (***required)***: value can be "echeck" or "card"
    - is-default
    - text
  - Child elements: None

16.2.4.2.2 Child element: <label>

Defines a label within the <paymentOptionDropdownSection>

Attributes

- for (***required)*** : value can be "paymentOptionType"
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.4.3 Element: <paymentOptionExpandedRadioSection>

Displays the payment options of "echeck" and specific card types using radio buttons. For example, these payment options could be defined: "echeck", "mastercard" and "visa".

Attributes

- name
- class

Child elements allowed

- <label>
- <radPaymentOption>
- <tag>
- <voidTag>

16.2.4.3.1 Child element: <radPaymentOption>

Defines a radio button within the <paymentOptionExpandedRadioSection>

Attributes

- for (***required***) : value can be “echeck”, “american express”, “discover”, “jcb”, “maestro”, “mastercard” or “visa” or "diners"
- is-default
- required
- required-msg

Child elements allowed

- None

16.2.4.3.2 Child element: <label>

Defines a label within the <paymentOptionExpandedRadioSection>

Attributes

- for (***required***) : value can be “echeck”, “american express”, “discover”, “jcb”, “maestro”, “mastercard” or “visa” or
- text

- class
- name (div, span or label)

Child elements allowed

- None

16.2.4.4 Element: <paymentOptionExpandedDropdownSection>

Displays the payment options of "echeck" and specific card types using a dropdown. For example, these payment options could be defined: "echeck", "mastercard" and "visa".

Attributes

- name
- class

Child elements allowed

- <label>
- <ddlPaymentOption>
- <tag>
- <voidTag>

16.2.4.4.1 Child element: <ddlPaymentOption>

Defines a dropdown within the <paymentOptionExpandedDropdownSection>

Attributes

- class
- default-text
- required

- required-msg
- error-class

> **Note:**
>
> The class attribute should be used to style elements. Other element attributes (such as id, name, etc) are internally generated values that are subject to change in the future. The merchant's CSS should not reference those attributes.

- error-msg-style

Child elements allowed

- <items>: Contains a list of **<item>** elements
  - <item>
    - Attributes:
      - for: value can be "echeck", "american express", "discover", "jcb", "maestro", "mastercard" or "visa" or "diners"
      - is-default
      - text
    - Child elements: None

16.2.4.4.2 Child element: <label>

Defines a label within the <paymentOptionExpandedDropdownSection>

Attributes

- for (***required***) : value can be "echeck", "american express", "discover", "jcb", "maestro", "mastercard" or "visa" or "diners"
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.5 Credit card sections

The credit card section is used to allow the web user to input credit card information.

There are three types of credit card sections. Each of these sections contains common credit card elements defined in the next section.

16.2.5.1 Default card validations

By default, the following client-side validations will be performed against credit card data. Any or all of these can be overridden or disabled in the XML.

Card number checks

- Luhn check
- Amex: Card number matches regex "^(34|37)([0-9]{13})$"
- Discover: Card number matches regex "(^(64|65|66)([0-9]{12}|[0-9]{14})$)|(^(622|624|625|626|628)([0-9]{11}|[0-9]{13})$)|(^(6011)([0-9]{10}|[0-9]{12})$)$ "
- JCB: Card number matches regex "(^(353|355|356|357|358)([0-9]{13})$)|(^(3528|3529)([0-9]{12})$)"
- MasterCard: Card number matches regex "^(36|51|52|53|54|55)([0-9]{12}|[0-9]{14})$"
- Visa: Card number matches regex "^(4)([0-9]{12}|[0-9]{15})$"

Expiration and start dates

Validation to verify the card expiration and start dates are valid is enabled by default.

16.2.5.2 Element: <cardRadioSection>

Displays the credit card section with credit card types rendered using radio buttons.

Attributes

- name
- class

Child elements allowed

- <radCardType>
- <radCustomCardType>
- <label>
- <customLabel>
- <tag>
- <voidTag>
- <validationMsg> - "for" attribute can be any supported input ("cardType", "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear", "cvv"), or the merchant can provide a custom value.
- ***See Common Credit Card Elements for descriptions of the following:***
  - <tboxCardNumber>
  - <tboxCardHolderName>
  - <tboxStartMonth>
  - <tboxStartYear>
  - <ddlStartMonth>
  - <ddlStartYear>
  - <tboxExpMonth>
  - <tboxExpYear>
  - <ddlExpMonth>
  - <ddlExpYear>
  - <tboxCvv>

16.2.5.2.1 Child element: <radCardType>

Defines a radio button for a supported card type within the <cardRadioSection>

The <radCardType> must be declared as an empty element. <radCardType for="visa"/> is valid. <radCardType for="visa"></radCardType> is not at this time.

Attributes

- for (***required***) : value can be "american express", "discover", "jcb", "maestro", "mastercard" or "visa" or "diners"
- class
- is-default
- error-class
- error-msg-style
- required
- required-msg

Child elements allowed

- <validateFor> - ***See Common Credit Card Elements***

16.2.5.2.2 Child element: <radCustomCardType>

Defines a radio button for a custom (merchant-defined) card type within the <cardRadioSection>

Attributes

- for (***required***) : merchant defines the value
- class
- is-default
- error-class
- error-msg-style
- required

- required-msg
- show-exp-date: If “true” or not supplied, expiration date fields are displayed. If “false”, expiration date fields are hidden.
- show-start-date: If “true”, start date fields are displayed. If “false” or missing, start date fields are hidden.
- show-cvv: If “true” or not supplied, cvv fields are displayed. If “false”, cvv fields are hidden.
- show- cardholder-name: If “true” or not supplied, cardholder name fields are displayed. If “false”, cardholder name fields are hidden.

Child elements allowed

- <validateFor> - ***See Common Credit Card Elements***

16.2.5.2.3 Child element: <label>

Defines a label within the <cardRadioSection>

Attributes

- for (***required***) : value can be “american express”, “discover”, “jcb”, “maestro”, “mastercard”, “visa”, "diners", “cardType”, “cardNumber”, “cardholderName”, “startDate”, “startMonth”, “startYear”, “expDate”, “expMonth”, “expYear” or “cvv”
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.5.2.4 Child element: <customLabel>

Defines a label to be used with a custom card type within the <cardRadioSection>

Attributes

- for: Merchant defined value. This should match the "for" used in the <customItem> defined in <radCustomCardType>
- text (***required)***
- class
- name (div, span or label)

Child elements allowed

- None

16.2.5.3 Element: <cardDropdownSection>

Displays the credit card section with credit card types rendered using a dropdown.

Attributes

- name
- class

Child elements allowed

- <label>
- <ddlCardType>
- <tag>
- <voidTag>
- <validationMsg> - "for" attribute can be any supported input ("cardType", "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear", "cvv"), or the merchant can provide a custom value.
- ***See Common Credit Card Elements for descriptions of the following:***

- <tboxCardNumber>
- <tboxCardHolderName>
- <tboxStartMonth>
- <tboxStartYear>
- <ddlStartMonth>
- <ddlStartYear>
- <tboxExpMonth>
- <tboxExpYear>
- <ddlExpMonth>
- <ddlExpYear>
- <tboxCvv>

16.2.5.3.1 Child element: <ddlCardType>

Defines a dropdown for card type within the <cardDropdownSection>

Attributes

- class
- default-text
- required
- required-msg
- error-class
- error-msg-style

Child elements allowed

- <items>: Contains a list of **<item>** elements
  - <item>
    - Attributes:
      - for (***required)***: value can be "american express", "discover", "jcb", "maestro", "mastercard" or "visa" or "diners"
      - is-default
      - text
    - Child elements:

- <validateFor>: ***See Common Credit Card Elements***

o <customItem> - Merchant's can use this to define custom card types
- Attributes:
  - for (***required)***: Merchant supplied value (code) for the custom card
  - is-default
  - text
  - show-exp-date: If "true" or not supplied, expiration date fields are displayed. If "false", expiration date fields are hidden.
  - show-start-date: If "true", start date fields are displayed. If "false" or missing, start date fields are hidden.
  - show-cvv: If "true" or not supplied, cvv fields are displayed. If "false", cvv fields are hidden.
  - show- cardholder-name: If "true" or not supplied, cardholder name fields are displayed. If "false", cardholder name fields are hidden.
- Child elements:
  - <validateFor>: ***See Common Credit Card Elements***

16.2.5.3.2 Child element: <label>

Defines a label within the <cardRadioSection>

Attributes

- for (***required***) : value can be "cardType", "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear" or "cvv"
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.5.3.3 Child element: <customLabel>

Defines a label for custom card types within the <cardRadioSection>

Attributes

- for: Merchant defined value. This should match the "for" defined in the <customItem> in <ddlCardType>
- text **(required)**
- class
- name (div, span or label)

Child elements allowed

- None

16.2.5.4 Element: <cardNoTypeSection>

Displays the credit card section without a control for the card type. Merchants use this in conjunction with the <paymentOptionExpandedRadio> or <paymentOptionExpandedDropdown>, which defines the card type outside the Card Section.

Attributes

- name
- class

Child elements allowed

- <label>
- <tag>
- <voidTag>

- <validationMsg> - "for" attribute can be any supported input ("cardType", "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear", "cvv"), or the merchant can provide a custom value.
- ***See Common Credit Card Elements for descriptions of the following:***
  - <tboxCardNumber>
  - <tboxCardHolderName>
  - <tboxStartMonth>
  - <tboxStartYear>
  - <ddlStartMonth>
  - <ddlStartYear>
  - <tboxExpMonth>
  - <tboxExpYear>
  - <ddlExpMonth>
  - <ddlExpYear>
  - <tboxCvv>

16.2.5.4.1 Child element: <label>

Defines a label within the <cardNoTypeSection>

Attributes

- for (***required***) : value can be "cardType", "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear" or "cvv"
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.5.5 Element: <cardIndicatorSection>

Displays the credit card section with credit card types rendered using a dropdown.

Attributes

- name
- class

Child elements allowed

- <label>
- <cardTypeIndicator>
- <tag>
- <voidTag>
- <validationMsg> - "for" attribute can be any supported input ("cardType", "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear", "cvv"), or the merchant can provide a custom value.
- ***See Common Credit Card Elements for descriptions of the following:***
  - <tboxCardNumber>
  - <tboxCardHolderName>
  - <tboxStartMonth>
  - <tboxStartYear>
  - <ddlStartMonth>
  - <ddlStartYear>
  - <tboxExpMonth>
  - <tboxExpYear>
  - <ddlExpMonth>
  - <ddlExpYear>
  - <tboxCvv>

16.2.5.5.1 Child element: <cardTypeIndicator>

Defines a dropdown for card type within the <cardTypeIndicator>

Attribute

- class

Child elements allowed

- <items>: Contains a list of **<item>** elements
  - <item>
    - Attributes:
      - for (***required)***: value can be "american express", "discover", "jcb", "maestro", "mastercard" or "visa" or "diners"
      - class (***required***): represents the class in the CSS file containing the unselected/selected image; for the selected image, the CSS should contain a class with the same name + " .selected"; for unselected image, the CSS should contain a class with the same name + " .unselected"

        EXAMPLE in CSS file:

```
.mcIndicator {
  background-image: url("../Images/mc.png");
  width: 33px;
  height: 26px;
  vertical-align: middle;
  margin-left: 40px;
  opacity: .3;
}
.mcIndicator.selected {
  opacity: 1;
}
```

      - bin-range-expression: represents the card number BIN range regular expression that selects the appropriate card type image
  - <customItem> – Merchants can use this to define custom card types
    - Attributes:
      - for (***required)***: Merchant supplied value (code) for the custom card

- class (***required)***: Represents the class in the CSS file containing the unselected image; for the selected image, the CSS should contain a second class with the same name + " .selected"
- bin-range-expression (***required)***: Represents the card number BIN range regular expression that selects the appropriate card type image
- show-exp-date: If “true” or not supplied, expiration date fields are displayed. If “false”, expiration date fields are hidden.
- show-start-date: If “true”, start date fields are displayed. If “false” or missing, start date fields are hidden.
- show-cvv: If “true” or not supplied, cvv fields are displayed. If “false”, cvv fields are hidden.
- show-cardholder-name: If “true” or not supplied, cardholder name fields are displayed. If “false”, cardholder name fields are hidden.

16.2.5.5.2 Card type indicator validations

If bin-range-expression is not included, the following values are used by default to determine the card type:

- Amex – Starts with 34 or 37
- Visa – Starts with 4
- MC – Starts with 222, 2720,36, or 51 thru 55
- Discover – Starts with 64-66, 622, 624-626,628, 6011
- JCB – Starts with 353-358, 3528, 3529
- Diners – Starts with 30+[0-5]+[0-9], 3095+[0-9], 36+[0-9], or 3+[8-9]+[0-9]
- Maestro – We do not provide a default match. Merchants must send in the regex match pattern in the 'bin-range-expression' field

16.2.6 Common credit card elements

Below is a list of common inputs used by all three types of Credit Card section elements.

- <tboxCardNumber>
- <tboxCardHolderName>
- <tboxStartMonth>
- <tboxStartYear>
- <ddlStartMonth>
- <ddlStartYear>
- <tboxExpMonth>
- <tboxExpYear>
- <ddlExpMonth>
- <ddlExpYear>
- <tboxCvv>

16.2.6.1 Textbox elements: <tbox{XXX}>

<tboxCardNumber>, <tboxCardHolderName>, <tboxStartMonth>, <tboxStartYear>, <tboxExpMonth>, <tboxExpYear>, <tboxCvv>

Attributes (All of these elements support these common attributes)

- class
- error-class
- error-msg-style
- text
- tokenize (***required for <tboxCardNumber>***)
- is-numeric: prevents any character other than numeric; mobile device displays numeric keyboard
- html-required: required if Merchant wants to use floating labels
- placeholder: provides the default text for textbox
- any of the ***Validation Attributes***, with the exception that only <tboxCardNumber> supports "luhn-check" and "luhn-check-msg")

Child elements allowed

- None

<tboxCardNumber> only

In addition to the above attributes, tboxCardNumber has the following attribute as well:

- mask-number
  - Displays the credit card number that is entered by the user with spaces based on one of the following:
    - <cardTypeIndicator> 151 – Based off BIN range
    - <cardDropdownSection> 145 – Based off selected card type in dropdown
    - <cardRadioSection> 143 – Based off selected card type radio button
  - Spacing pattern for Amex is 4 6 5, for example:
    - 34XX XXXXXX XXXXX
  - Spacing pattern for all other card types are in groups of 4, for example:
    - XXXX XXXX XXXX
    - XXXX XXXX XXXX X
    - XXXX XXXX XXXX XXX
    - XXXX XXXX XXXX XXXX
  - Spacing pattern is controlled by Intercept for eCommerce; users are not allowed to enter spaces.

- This attribute is ignored if Intercept for eCommerce is configured to accept token entry. You can see this setting in Merchant Portal. Go to Settings > Services > Intercept for eCommerce.

16.2.6.2 Elements: <ddlStartMonth>, <ddlStartYear>

These elements render select dropdowns for card start month and year. These dropdowns only display if **Maestro** is selected.

Attributes

- class
- default-text
- display-format: valid values for <ddlStartMonth> are "M","MM","MMM" or "MMMM" (e.g. '9', '09', 'Sep' or 'September'); valid values for <ddlStartYear> are "yy" or "yyyy" (e.g. '14' or '2014') The defaults are "MMM" and "yyyy"
- error-class
- error-msg-style
- required
- required-msg
- start-date: if "true", then a single validation is performed against both the month and year. Use this to simplify validation by having a single message for the web user rather than separate messages for month and date. If this is used, set the ***required*** attributes on <ddlStartMonth> and <ddlStartYear> to ***"false"***
- start-date-msg: the message to display if the selected month and year are not a valid start date
- years-to-display (***<ddlStartYear> only)***: The number of years to display in the dropdown. The default is 10.

Child elements allowed: For <ddlStartMonth> only

- <months>: this is an optional element that contains child elements <jan>, <feb>, <mar>, <apr>, <may>, <jun>, <jul>, <aug>, <sep>, <oct>, <nov> and <dec>.

Each of these child elements can have a single attribute text to define the text to use for that month if the default text generated is not sufficient.

16.2.6.3 Elements: <ddlExpMonth>, <ddlExpYear>

These elements render select dropdowns for card expiration month and year.

Attributes

- class
- default-text
- display-format: valid values for <ddlExpMonth> are "M","MM","MMM" or "MMMM" (e.g. '9', '09', 'Sep' or 'September'); valid values for <ddlExpYear> are "yy" or "yyyy" (e.g. '14' or '2014') The defaults are "MMM" and "yyyy"
- error-class
- error-msg-style
- required
- required-msg
- exp-date: if "true", then a single validation is performed against both the month and year. Use this to simplify validation by having a single message for the web user rather than separate messages for month and date. If this is used, set the ***required*** attributes on <ddlExpMonth> and <ddlExpYear> to ***"false"***
- exp-date-msg: the message to display if the selected month and year are not a valid expiration date
- years-to-display (***<ddlExpYear> only)***: The number of years to display in the dropdown. The default is 10.

Child elements allowed: For <ddlExpMonth> only

- <months>: this is an ***optional*** element that contains child elements <jan>, <feb>, <mar>, <apr>, <may>, <jun>, <jul>, <aug>, <sep>, <oct>, <nov> and <dec>. Each of these child elements can have a single attribute **text** to define the text to use for that month if the default text generated is not sufficient.

16.2.6.4 Element: <validationFor>

The <validationFor> element is only used within a Credit Card section. This element is used to define a cross-control validation dependency between the selected card type and another credit card input field. For example, a "regex" validation dependency can be created between a specific card type and the card number textbox such that if that card type is selected, then card number must follow the regex pattern defined.

Use this element to create a separate html element for displaying validation messages or error images.

Attributes

- None

Child elements allowed

- <validateCardNumber>: This sets up a link between the card type and card number. Any of the ***Validation Attributes*** can be used on this element to add or remove validation.
- <validate>: This sets up a link between the card type and another input besides card number. The valid attributes for this element are:
  - for (***required)***: The input this validation applies to. Valid values are "cardNumber", "cardholderName", "startDate", "startMonth", "startYear", "expDate", "expMonth", "expYear" or "cvv"
  - Any of the ***Validation Attributes*** defined above, except for luhn-check and luhn-check-msg.

16.2.7 eCheck sections

The eCheck section is used to allow the web user to input electronic check information.

There are two types of eCheck sections. Each of these sections contains common eCheck elements defined in the next section.

16.2.7.1 Element: <echeckRadioSection>

Displays the eCheck section with eCheck options rendered using radio buttons.

Attributes

- name
- class

Child elements allowed

- <radAccountType>
- <label>
- <tag>
- <voidTag>
- <validationMsg> - "for" attribute can be any supported input ("accountType", "accountNumber", "routingNumber", "nameOnAccount", or "bankName")

See Common eCheck Elements for descriptions of the following:

- o <tboxAccountNumber>
- o <tboxBankName>
- o <tboxNameOnAccount>
- o <tboxRoutingNumber>

16.2.7.1.1 Child element: <radAccountType>

Defines a radio button for supported eCheck options within the <echeckRadioSection>

Attributes

- for (***required***) : value can be "checking" or "savings"
- class
- is-default
- error-class
- error-msg-style
- required
- required-msg

Child elements allowed

- None

16.2.7.1.2 Child element: <label>

Defines a label within the <echeckRadioSection>

Attributes

- for (***required***) : value can be "checking", "savings", "accountNumber", "routingNumber", "nameOnAccount", or "bankName"
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.7.2 Element: <echeckDropdownSection>

Displays the eCheck section with eCheck options rendered using a dropdown.

Attributes

- name
- class

Child elements allowed

- <ddlAccountType>
- <label>
- <tag>
- <voidTag>
- <validationMsg> - "for" attribute can be any supported input ("accountType", "accountNumber", "routingNumber", "nameOnAccount", or "bankName")
- ***See Common eCheck Elements for descriptions of the following:***
  - <tboxAccountNumber>
  - <tboxBankName>
  - <tboxNameOnAccount>
  - <tboxRoutingNumber>

16.2.7.2.1 Child element: <ddlAccountType>

Defines a dropdown for supported eCheck options within the <echeckDropdownSection>

Attributes

- class
- default-text
- error-class
- error-msg-style
- required
- required-msg

Child elements allowed

- <items>: Contains a list of **<item>** elements
  - <item>
    - Attributes:
      - for (***required)***: value can be "checking" or "savings"
      - is-default
      - text
    - Child elements:
      - None

16.2.7.2.2 Child element: <label>

Defines a label within the <echeckDropdownSection>

Attributes

- for (***required***) : value can be "accountType", "accountNumber", "routingNumber", "nameOnAccount", or "bankName"
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.8 Common eCheck elements

Below is a list of common inputs used by both eCheck section elements.

- <tboxAccountNumber>

- <tboxBankName>
- <tboxNameOnAccount>
- <tboxRoutingNumber>

16.2.8.1 Textbox elements: <tbox{XXX}>

<tboxAccountNumber>, <tboxBankName>, <tboxNameOnAccount>, <tboxRoutingNumber>

Attributes (All of these elements support these common attributes)

- class
- error-class
- error-msg-style
- text
- tokenize (***required for <tboxAccountNumber>***)
- is-numeric: prevents any character other than numeric; mobile device displays numeric keyboard
- html-required: required if Merchant wants to use floating labels
- placeholder: provides the default text for textbox
- any of the ***Validation Attributes***, except for “luhn-check” and "luhn-check-msg"

Child elements allowed

- None

16.2.9 Additional HTML section

This section is used to contain any additional data input that the merchant wants to collect. For example, to collect address information, add this section to the XML.

16.2.9.1 Element: <additionalHtmlSectionElements>

Attributes

- class
- name

Child elements allowed

- <tag>
- <label>
- <textbox>
- <ddlUsStateDropdown>
- <validationMsg>

16.2.9.1.1 Child element: <label>

Defines a label within the <additionalHtmlSectionElements>

Attributes

- for: Merchant defined value
- text
- class
- name (div, span or label)

Child elements allowed

- None

16.2.9.1.2 Child element: <textBox>

Defines an input textbox within the <additionalHtmlSectionElements>

Attributes

- class
- text
- name (***required***): Merchant defined value. This HTML input generated will have a "name" attribute with this value.
- is-numeric: prevents any character other than numeric; mobile device displays numeric keyboard
- html-required: **required** if Merchant wants to use floating labels
- placeholder: provides the default text for textbox
- any of the ***Validation Attributes***, except for "luhn-check" and "luhn-check-msg"

Child elements allowed

- None

16.2.9.1.3 Child element: <ddlUsStateDropdown>

Defines a select dropdown of US states within the <additionalHtmlSectionElements>

Attributes

- class
- default-text
- error-class
- error-msg-style
- required
- required-msg

Child elements allowed

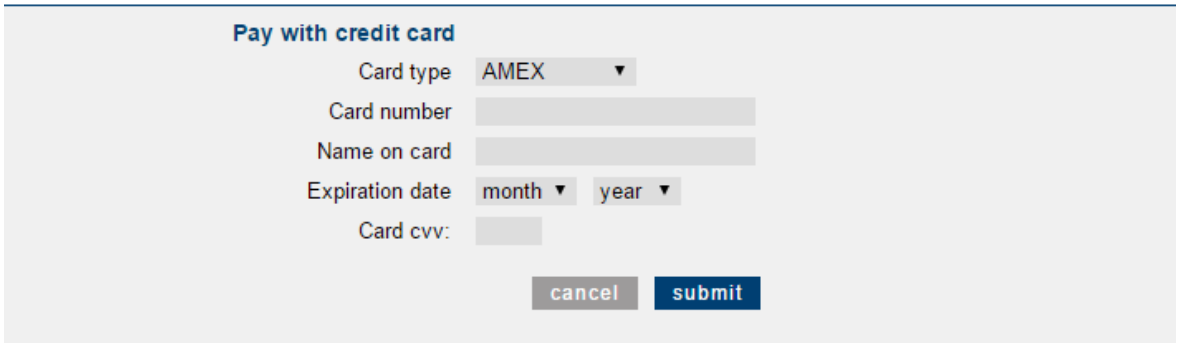
- None

16.2.10 Sample merchant XMLs

Below are samples that show various ways to configure the Merchant Custom XML and the HTML output generated for the web user.

> **Note:**
>
> These samples use logos and copyright information. Merchant logos and information can be supplied through the XML and CSS.

16.2.10.1 Sample 1: Credit card (dropdown)



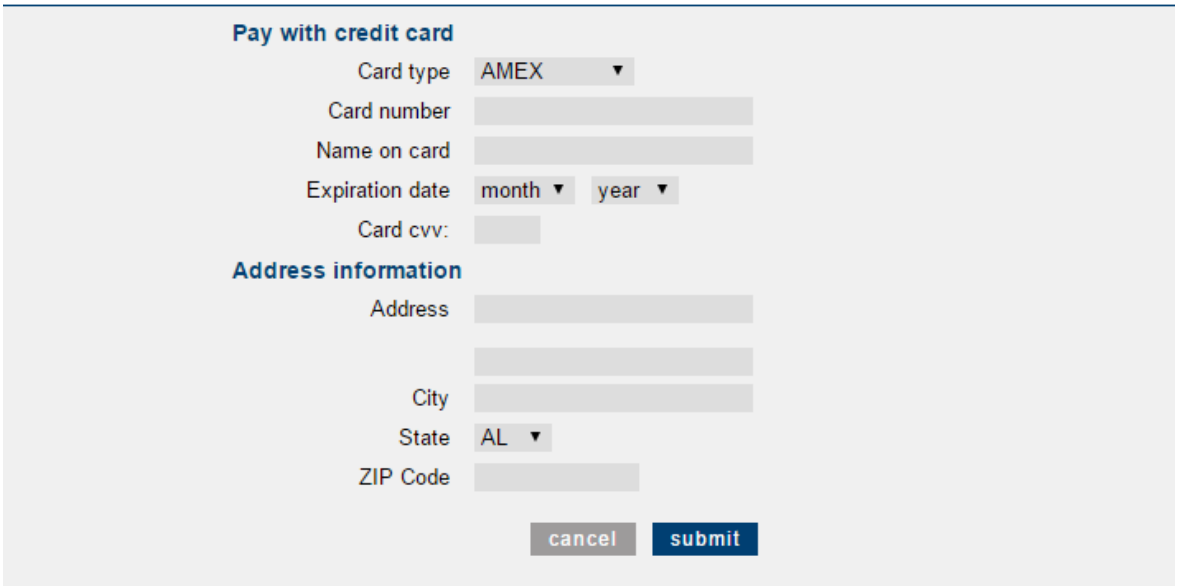
Sample 1: Credit card (dropdown)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType>
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
```

Sample 1: Credit card (dropdown)

```
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.2 Sample 2: Credit card (dropdown) with additional HTML section for capturing address



Sample 2: Credit card (dropdown) with additional HTML section for capturing address

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType>
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
```

Sample 2: Credit card (dropdown) with additional HTML section for capturing address

```
                            </ddlCardType>
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="address-header"></tag>
```

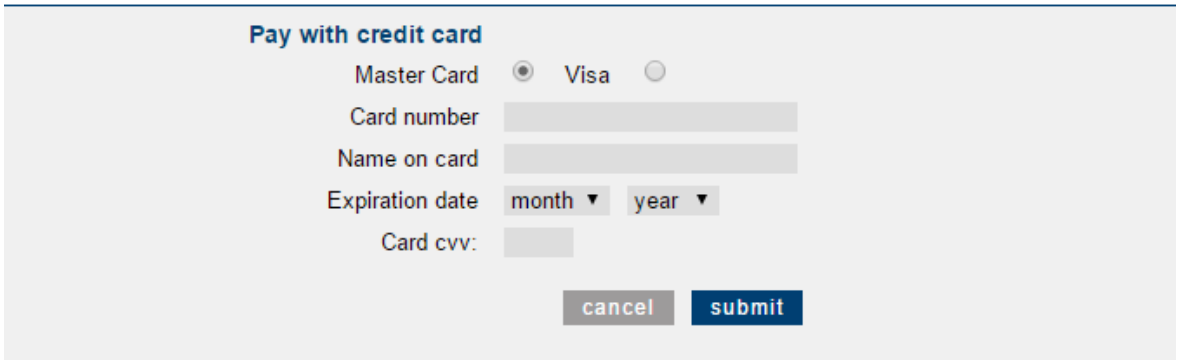
Sample 2: Credit card (dropdown) with additional HTML section for capturing address

```
                <tag name="div" class="address-info">
                    <additionalHtmlSection class="address-content">
                        <tag name="div">
                            <label for="address1" />
                            <textBox name="address1" class="xi-long-
text" required-msg="Please enter an address" />
                        </tag>
                        <tag name="div">
                            <label for="address2" />
                            <textBox name="address2" class="xi-long-
text" required="false" />
                            <validationMsg for="address1"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="city" />
                            <textBox name="city" class="xi-long-text"
required-msg="Please enter the city" />
                            <validationMsg for="city" class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="state" />
                            <ddlUsStateDropdown />
                            <validationMsg for="state"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="postalCode" />
                            <textBox name="postalCode" class="xi-
medium-text" pattern="^\d{5}(-\d{4})?$" required-msg="Please enter the
postal code" pattern-msg="Please enter a valid postal code" />
                            <validationMsg for="postalCode"
class="valmsg" />
                        </tag>
                    </additionalHtmlSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
```

Sample 2: Credit card (dropdown) with additional HTML section for capturing address

```
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.3 Sample 3: Credit card (radio)



Sample 3: Credit card (radio)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
```

Sample 3: Credit card (radio)

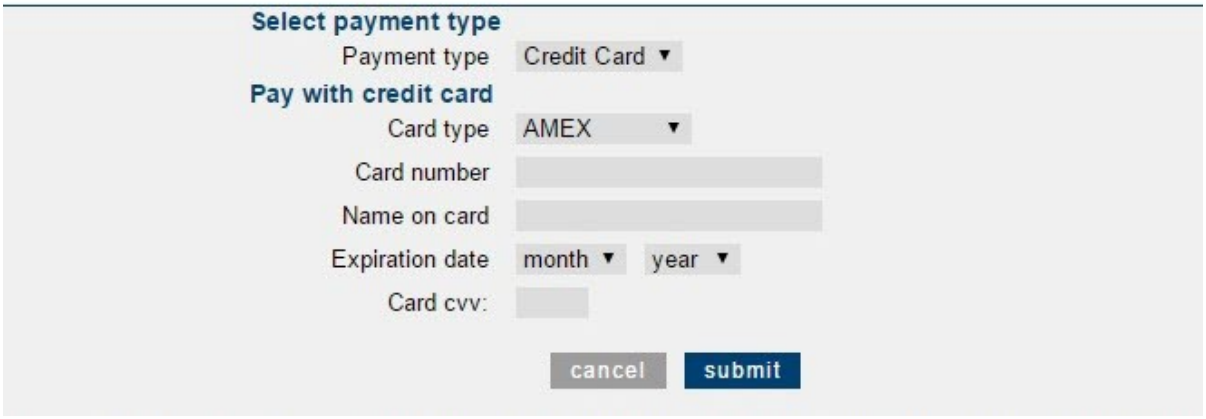
```
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardRadioSection>
                        <tag name="div">
                            <label for="mastercard" />
                            <radCardType for="mastercard" />
                            <label for="visa" text="Visa" />
                            <radCardType for="visa" />
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
```

Sample 3: Credit card (radio)

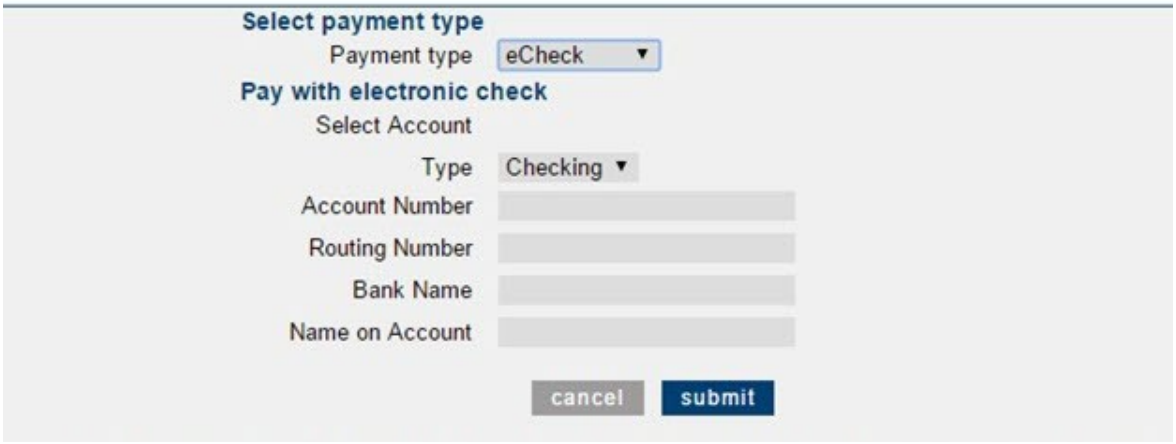
```
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardRadioSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.4 Sample 4: Credit card (dropdown) and eCheck (dropdown) with payment option (dropdown)

Credit card view



eCheck view



Sample 4: Credit card (dropdown) and eCheck (dropdown) with payment option (dropdown)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
```

Sample 4: Credit card (dropdown) and eCheck (dropdown) with payment option (dropdown)

```
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <paymentOptionDropdownSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="payoption-header"></tag>
                    <tag name="div" class="payoption-info">
                        <tag>
                            <label for="paymentOptionType"
text="Payment type" />
                            <ddlPaymentOption>
                                <items>
                                    <item for="card" />
                                    <item for="echeck" />
                                </items>
                            </ddlPaymentOption>
                        </tag>
                    </tag>
                </tag>
            </paymentOptionDropdownSection>
            <cardDropdownSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="billing-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType>
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
```

Sample 4: Credit card (dropdown) and eCheck (dropdown) with payment option (dropdown)

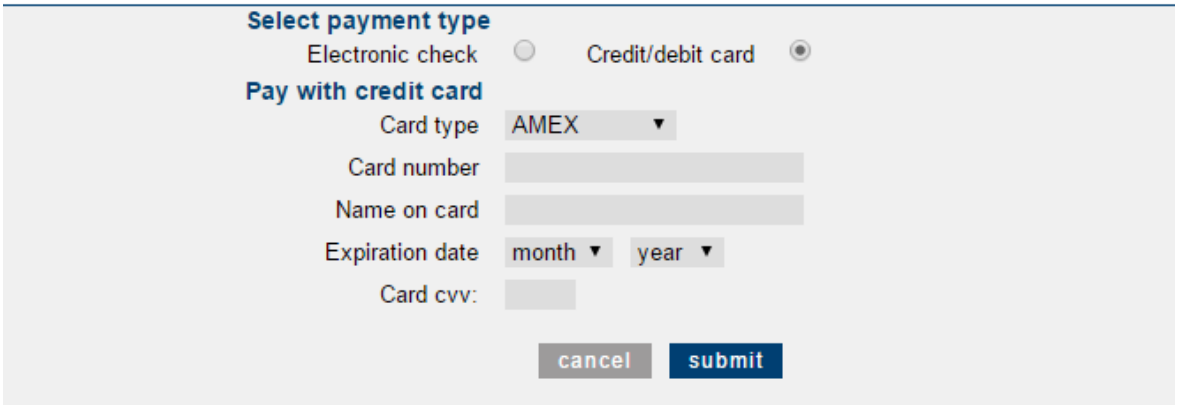
```
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </tag>
                </tag>
            </cardDropdownSection>
            <echeckDropdownSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="echeck-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="accountType" />
                            <ddlAccountType>
                                <items>
                                    <item for="checking" />
                                    <item for="savings" />
                                </items>
```

Sample 4: Credit card (dropdown) and eCheck (dropdown) with payment option (dropdown)

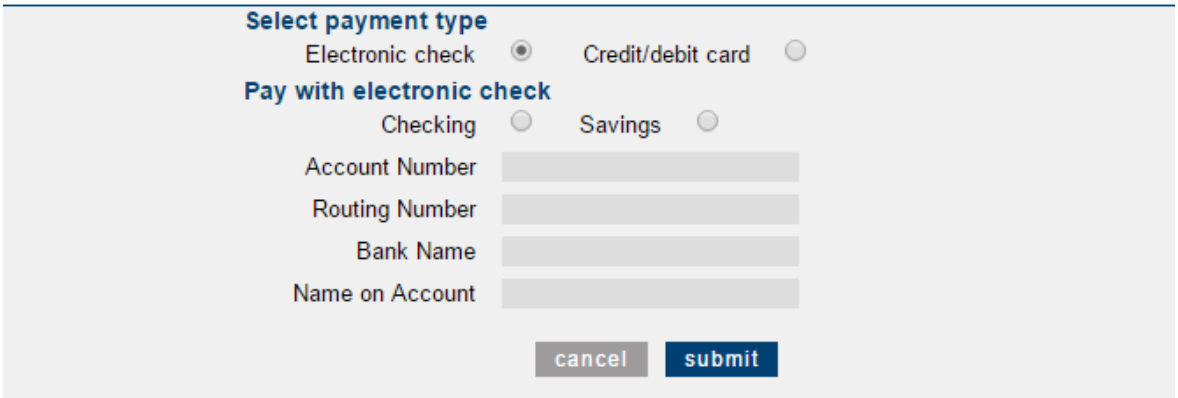
```
                            </ddlAccountType>
                        </tag>
                        <tag name="div">
                            <label for="accountNumber" />
                            <tboxAccountNumber tokenize="true" />
                        </tag>
                        <tag name="div">
                            <label for="routingNumber" />
                            <tboxRoutingNumber />
                        </tag>
                        <tag name="div">
                            <label for="bankName" />
                            <tboxBankName />
                        </tag>
                        <tag name="div">
                            <label for="nameOnAccount" />
                            <tboxNameOnAccount />
                        </tag>
                    </tag>
                </tag>
            </echeckDropdownSection>
            <tag name="div" class="payment-button">
                <cancelButton text="cancel" class="cancel-button" />
                <submitButton text="submit" class="submit-button" />
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.5 Sample 5: Credit card (dropdown) and eCheck (radio) with payment option (radio)

Credit card view



eCheck view



Sample 5: Credit card (dropdown) and eCheck (radio) with payment option (radio)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
```

Sample 5: Credit card (dropdown) and eCheck (radio) with payment option (radio)

```
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <paymentOptionRadioSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="payoption-header"></tag>
                    <tag name="div" class="payoption-info">
                        <tag>
                            <label for="echeck" text="Electronic check"
/>
                            <radPaymentOption for="echeck" />
                            <label for="card" text="Credit/debit
card" />
                            <radPaymentOption for="card" />
                        </tag>
                    </tag>
                </tag>
            </paymentOptionRadioSection>
            <cardDropdownSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="billing-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType>
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
```

Sample 5: Credit card (dropdown) and eCheck (radio) with payment option (radio)

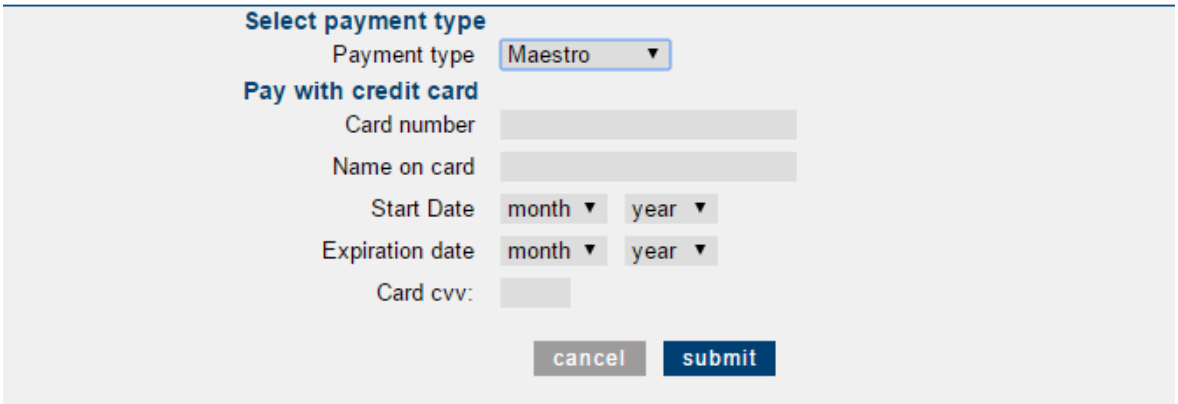
```
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </tag>
                </tag>
            </cardDropdownSection>
            <echeckRadioSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="echeck-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="checking" />
                            <radAccountType for="checking" />
                            <label for="savings" class="ec-label" />
                            <radAccountType for="savings" />
                        </tag>
                        <tag name="div">
                            <label for="accountNumber" />
                            <tboxAccountNumber tokenize="true" />
                        </tag>
```

Sample 5: Credit card (dropdown) and eCheck (radio) with payment option (radio)

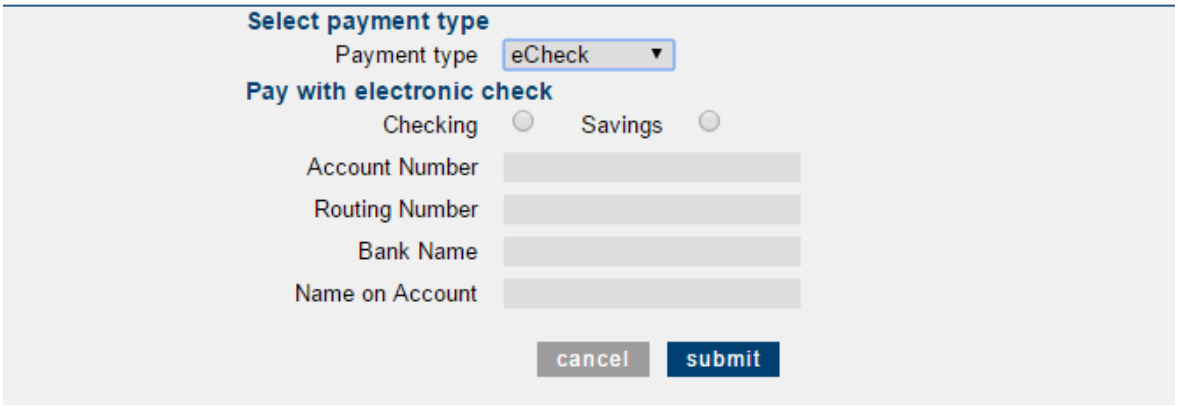
```
                        <tag name="div">
                            <label for="routingNumber" />
                            <tboxRoutingNumber />
                        </tag>
                        <tag name="div">
                            <label for="bankName" />
                            <tboxBankName />
                        </tag>
                        <tag name="div">
                            <label for="nameOnAccount" />
                            <tboxNameOnAccount />
                        </tag>
                    </tag>
                </tag>
            </echeckRadioSection>
            <tag name="div" class="payment-button">
                <cancelButton text="cancel" class="cancel-button" />
                <submitButton text="submit" class="submit-button" />
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.6 Sample 6: Credit card (NoType) and eCheck (radio) with payment option (expanded dropdown)

Credit Card view



eCheck view



Sample 6: Credit card (NoType) and eCheck (radio) with payment option (expanded dropdown)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
```

Sample 6: Credit card (NoType) and eCheck (radio) with payment option (expanded dropdown)

```
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <paymentOptionExpandedDropdownSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="payoption-header"></tag>
                    <tag name="div" class="payoption-info">
                        <tag>
                            <label for="paymentOptionType"
text="Payment type" />
                            <ddlPaymentOption>
                                <items>
                                    <item for="echeck" />
                                    <item for="american express" />
                                    <item for="discover" />
                                    <item for="jcb" />
                                    <item for="maestro" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                </items>
                            </ddlPaymentOption>
                        </tag>
                    </tag>
                </tag>
            </paymentOptionExpandedDropdownSection>
            <cardNoTypeSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="billing-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
```

Sample 6: Credit card (NoType) and eCheck (radio) with payment option (expanded dropdown)

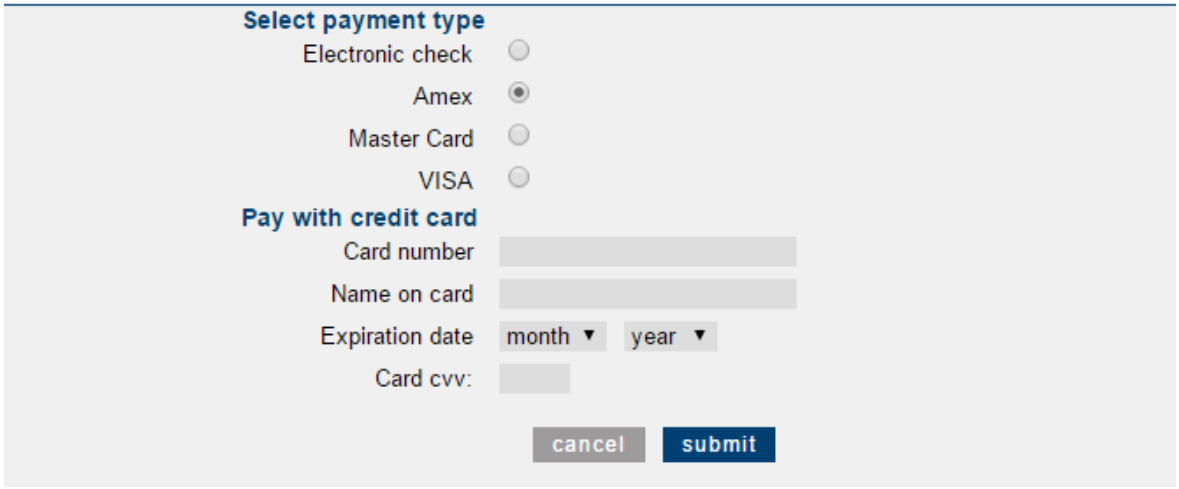
```
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" default-text="month" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" default-text="year" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </tag>
                </tag>
            </cardNoTypeSection>
            <echeckRadioSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="echeck-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="checking" />
                            <radAccountType for="checking" />
                            <label for="savings" class="ec-label" />
                            <radAccountType for="savings" />
                        </tag>
                        <tag name="div">
                            <label for="accountNumber" />
                            <tboxAccountNumber tokenize="true" />
                        </tag>
                        <tag name="div">
                            <label for="routingNumber" />
                            <tboxRoutingNumber />
                        </tag>
                        <tag name="div">
```

Sample 6: Credit card (NoType) and eCheck (radio) with payment option (expanded dropdown)

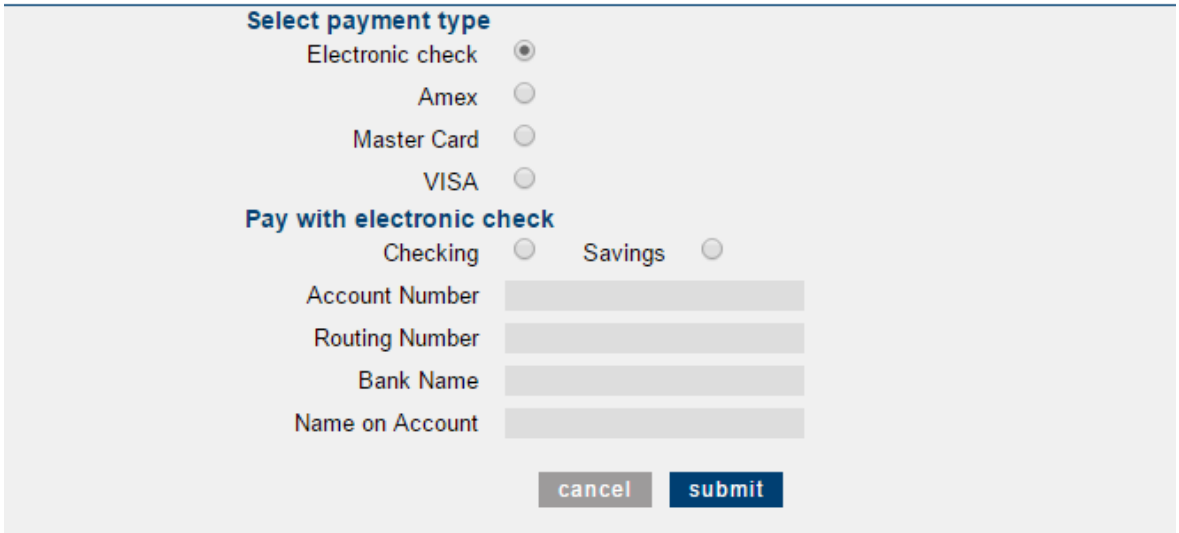
```
                            <label for="bankName" />
                            <tboxBankName />
                        </tag>
                        <tag name="div">
                            <label for="nameOnAccount" />
                            <tboxNameOnAccount />
                        </tag>
                    </tag>
                </tag>
            </echeckRadioSection>
            <tag name="div" class="payment-button">
                <cancelButton text="cancel" class="cancel-button" />
                <submitButton text="submit" class="submit-button" />
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.7 Sample 7: Credit card (NoType) and eCheck (radio) with payment option (expanded radio)

Credit card view



eCheck view



Sample 7: Credit card (NoType) and eCheck (radio) with payment option (expanded radio)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <paymentOptionExpandedRadioSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="payoption-header"></tag>
                    <tag name="div" class="payoption-info">
                        <tag>
                            <label for="echeck" text="Electronic check"
/>
                            <radPaymentOption for="echeck" />
                        </tag>
                        <tag>
                            <label for="american express"
text="Amex" />
                            <radPaymentOption for="american express" />
                        </tag>
                        <tag>
                            <label for="mastercard" />
                            <radPaymentOption for="mastercard" />
                        </tag>
                        <tag>
                            <label for="visa" />
                            <radPaymentOption for="visa" />
                        </tag>
                    </tag>
                </tag>
            </paymentOptionExpandedRadioSection>
            <cardNoTypeSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="billing-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
```

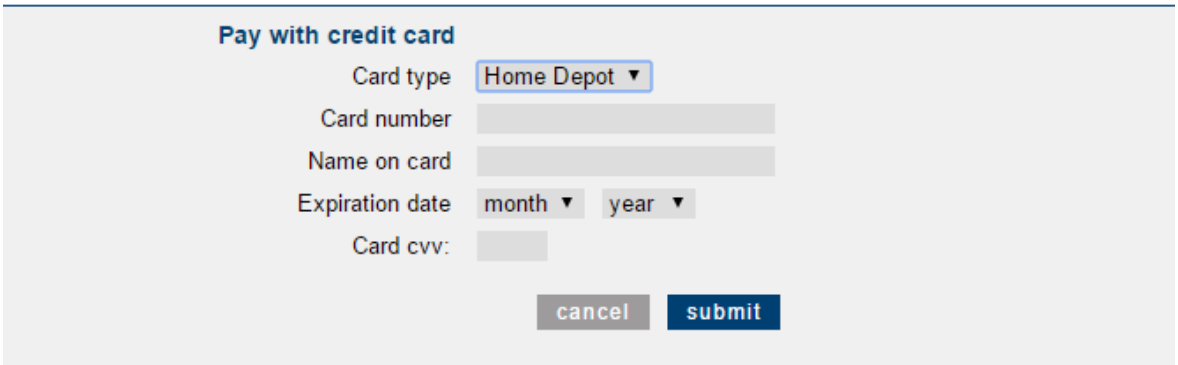
Sample 7: Credit card (NoType) and eCheck (radio) with payment option (expanded radio)

```
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </tag>
                </tag>
            </cardNoTypeSection>
            <echeckRadioSection>
                <tag name="div" class="billing-content">
                    <tag name="div" class="echeck-header"></tag>
                    <tag name="div" class="billing-info">
                        <tag name="div">
                            <label for="checking" />
                            <radAccountType for="checking" />
                            <label for="savings" class="ec-label" />
                            <radAccountType for="savings" />
                        </tag>
                        <tag name="div">
```

Sample 7: Credit card (NoType) and eCheck (radio) with payment option (expanded radio)

```
                            <label for="accountNumber" />
                            <tboxAccountNumber tokenize="true" />
                        </tag>
                        <tag name="div">
                            <label for="routingNumber" />
                            <tboxRoutingNumber />
                        </tag>
                        <tag name="div">
                            <label for="bankName" />
                            <tboxBankName />
                        </tag>
                        <tag name="div">
                            <label for="nameOnAccount" />
                            <tboxNameOnAccount />
                        </tag>
                    </tag>
                </tag>
            </echeckRadioSection>
            <tag name="div" class="payment-button">
                <cancelButton text="cancel" class="cancel-button" />
                <submitButton text="submit" class="submit-button" />
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.8 Sample 8: Credit card (dropdown) with merchant custom card type (company A)



Sample 8: Credit card (dropdown) with merchant custom card type (company A)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType>
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                    <customItem for="HD" text="Company
A" />
                                </items>
                            </ddlCardType>
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
```

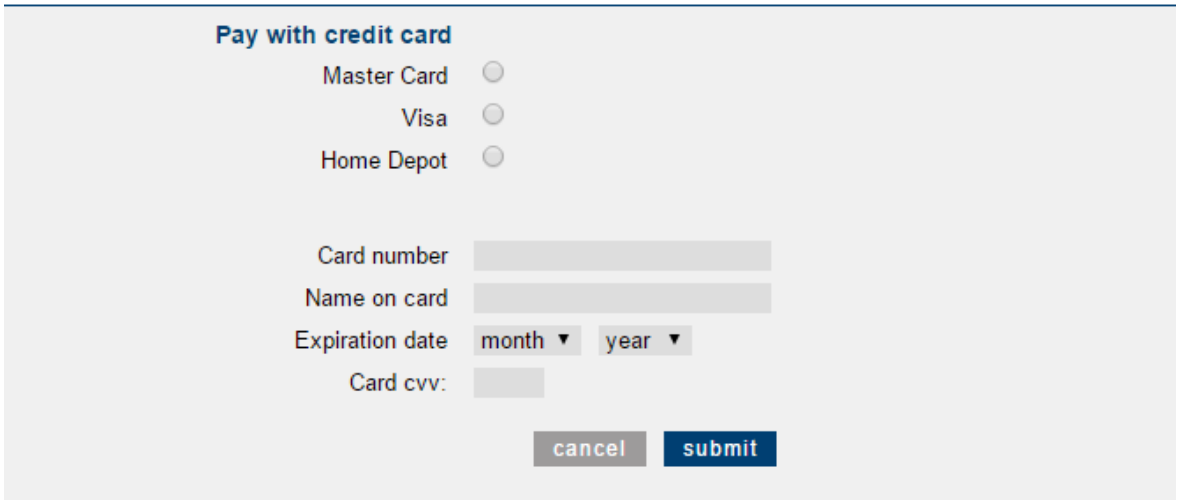
Sample 8: Credit card (dropdown) with merchant custom card type (company A)

```
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
```

Sample 8: Credit card (dropdown) with merchant custom card type (company A)

```
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.9 Sample 9: Credit card (radio) with merchant custom card type (company A)



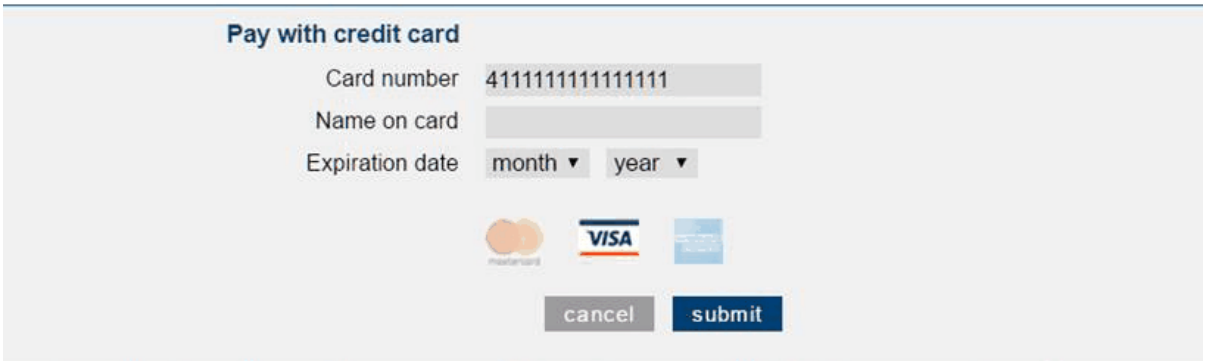
Sample 9: Credit card (radio) with merchant custom card type (company A)

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardRadioSection>
                        <tag name="div">
                            <label for="mastercard" />
                            <radCardType for="mastercard" />
                        </tag>
                        <tag name="div">
                            <label for="visa" text="Visa" />
                            <radCardType for="visa" />
                        </tag>
                        <tag name="div">
                            <customLabel for="HD" text="Company A" />
                            <radCustomCardType for="HD" />
                        </tag>
                        <voidTag name="br" />
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
```

Sample 9: Credit card (radio) with merchant custom card type (company A)

```
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardRadioSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.10 Sample 10: Credit card with card indicator



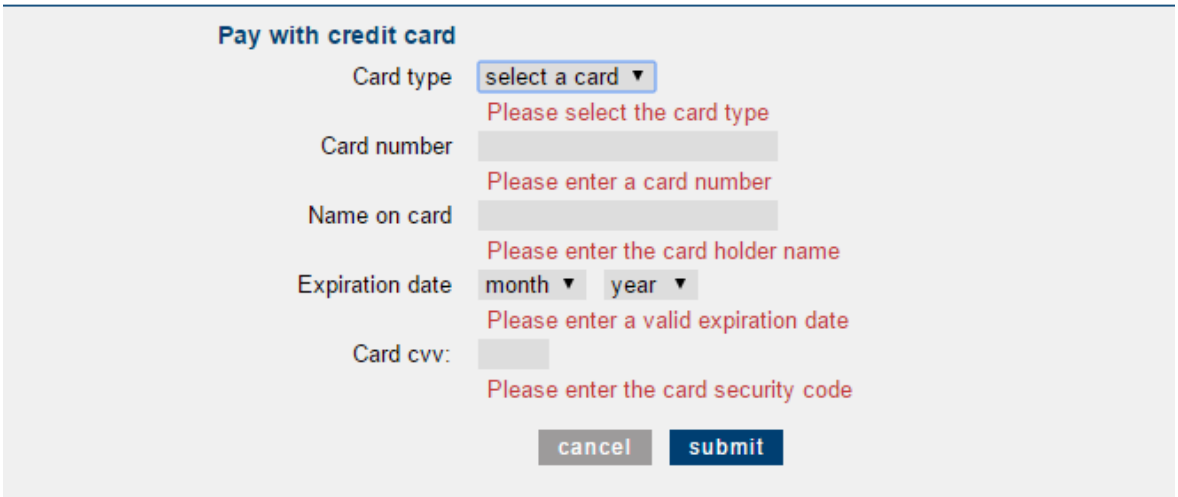
Sample 10: Credit card with card indicator

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header" />
                <tag name="div" class="billing-info">
                    <cardIndicatorSection>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
```

Sample 10: Credit card with card indicator

```
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <cardTypeIndicator class="cardIndicator">
                                <items>
                                    <item for="mastercard"
class="mcIndicator" bin-range-expression="^5"></item>
                                    <item for="visa"
class="viIndicator"></item>
                                    <item for="american express"
class="axIndicator"></item>
                                </items>
                            </cardTypeIndicator>
                        </tag>
                    </cardIndicatorSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing" />
            <tag name="div" class="top-bottom-frame" />
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-
content">Copyright © 2025. Worldpay.com. All Rights Reserved. Various
trademarks held by their respective owners. Legal Notice.</tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.11 Sample 11: Input validations: text



Sample 11: Input validations: text

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType default-text="select a card">
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                            <validationMsg for="cardType"
class="valmsg" />
                        </tag>
                        <tag name="div">
```

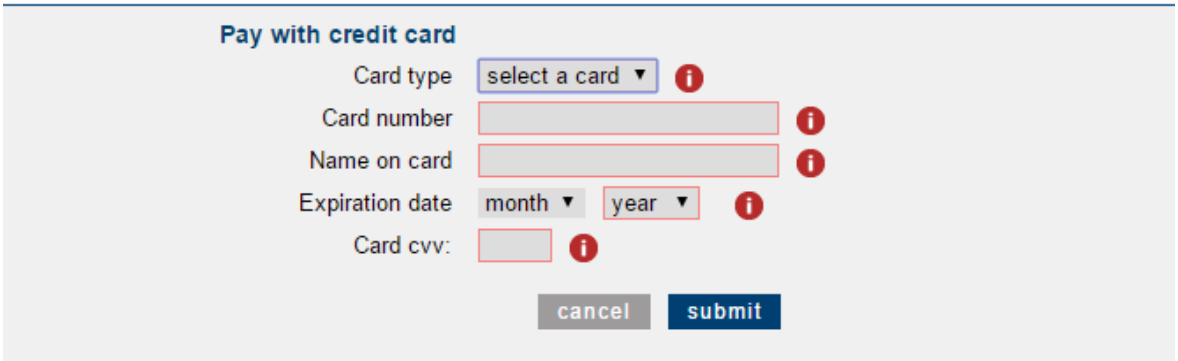
Sample 11: Input validations: text

```
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
```

Sample 11: Input validations: text

```
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.12 Sample 12: Input validations: image with Tooltip



Sample 12: Input validations: image with Tooltip

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <xiProperties error-class="input-error" inline-validation="true">
```

Sample 12: Input validations: image with Tooltip

```
        <errorTooltip class="errorTooltip" show-effect="fadeIn" show-
duration="5000" hide-effect="fadeOut" hide-duration="5000" />
    </xiProperties>
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType default-text="select a card"
error-msg-style="imageTooltip">
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                            <validationMsg for="cardType" error-
class="valmsg-img" />
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" error-msg-style="imageTooltip" />
                            <validationMsg for="cardNumber" error-
class="valmsg-img" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text"
error-msg-style="imageTooltip" />
                            <validationMsg for="cardholderName" error-
class="valmsg-img" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
```

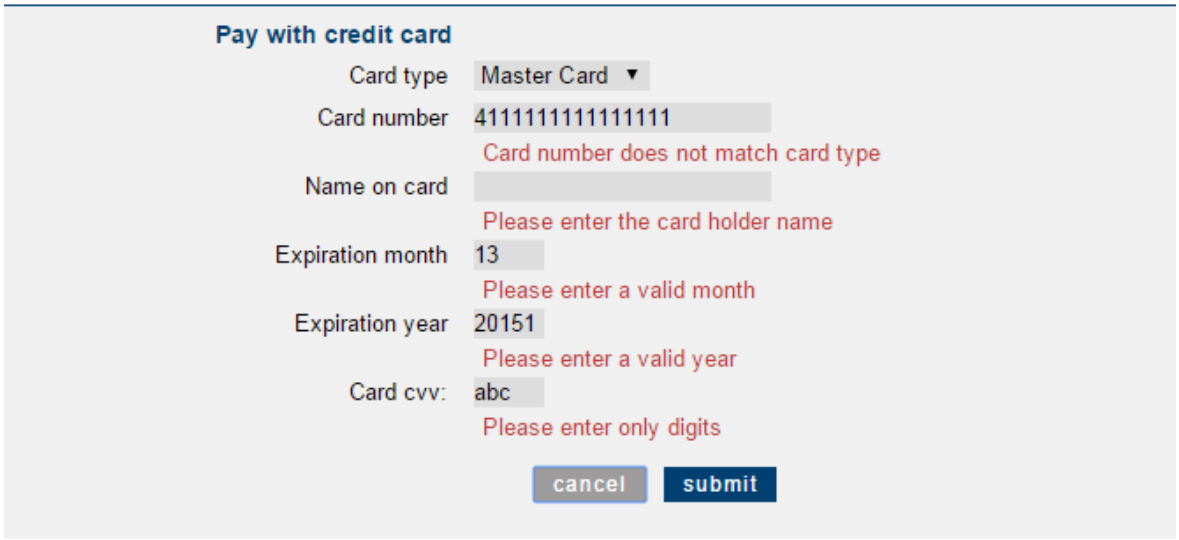
Sample 12: Input validations: image with Tooltip

```
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" error-msg-
style="imageTooltip" />
                            <validationMsg for="startYear" error-
class="valmsg-img" />
                        </tag>
                        <tag name="div">
                            <label for="expDate" text="Expiration date"
/>
                            <ddlExpMonth default-text="month"
class="date_combos" required="false" />
                            <ddlExpYear default-text="year"
class="date_combos" years-to-display="10" required="false" exp-
date="true" error-msg-style="imageTooltip" />
                            <validationMsg for="expYear" error-
class="valmsg-img" />
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text" error-msg-
style="imageTooltip" />
                            <validationMsg for="cvv" error-
class="valmsg-img" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
```

Sample 12: Input validations: image with Tooltip

```
</htmlSection>
```

16.2.10.13 Sample 13: Custom validations



Sample 13: Custom validations

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
```

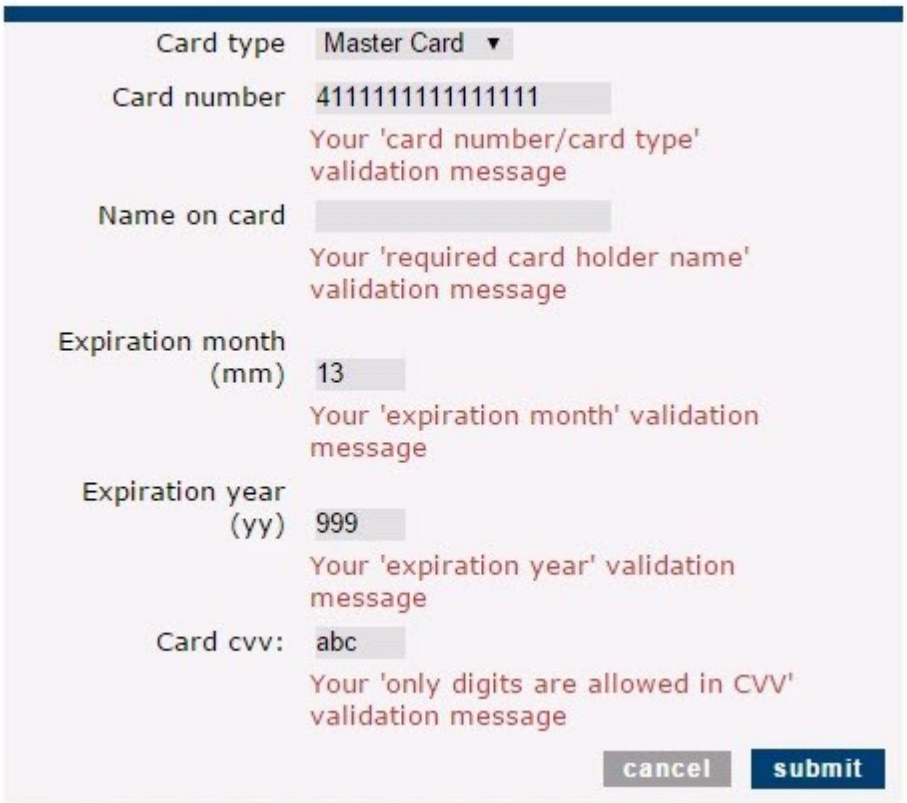
Sample 13: Custom validations

```
                            <ddlCardType default-text="select a card">
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                            <validationMsg for="cardType"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" luhn-check="false" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expMonth" text="Expiration
month" />
                            <tboxExpMonth class="xi-short-text" />
                            <validationMsg for="expMonth"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expYear" text="Expiration year"
/>
```

Sample 13: Custom validations

```
                            <tboxExpYear pattern="[0-9]{4}" pattern-
msg="Please enter a valid year" class="xi-short-text" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text"
minlength="3" maxlength="3" minlength-msg="CVV must be 3 digits"
maxlength-msg="CVV must be 3 digits" digits-only="true" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```

16.2.10.14 Sample 14: Custom messages



Sample 14: Custom messages

```
<htmlSection class="payment-content"
xmlns="Paymetric:Intercept:MerchantHtmlPacketModel">
    <tag>
        <tag name="div" class="header-top" />
        <tag name="div" class="merchant-logo" />
        <tag name="div" class="main-content">
            <tag name="div" class="billing-content">
                <tag name="div" class="billing-header"></tag>
                <tag name="div" class="billing-info">
                    <cardDropdownSection>
                        <tag name="div">
                            <label for="cardType" text="Card type" />
                            <ddlCardType>
                                <items>
                                    <item for="american express" />
                                    <item for="mastercard" />
```

Sample 14: Custom messages

```
                                    <item for="visa" />
                                    <item for="maestro" />
                                </items>
                            </ddlCardType>
                            <validationMsg for="cardType"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardNumber" text="Card
number" />
                            <tboxCardNumber tokenize="true" class="xi-
long-text" luhn-check-msg="Your 'luhn-check' validation message"
pattern-msg="Your 'card number/card type' validation message" />
                            <validationMsg for="cardNumber"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="cardholderName" text="Name on
card" />
                            <tboxCardHolderName class="xi-long-text"
required-msg="Your 'required card holder name' validation message" />
                            <validationMsg for="cardholderName"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="startDate" />
                            <ddlStartMonth display-format="MMM"
class="date_combos" required="false" />
                            <ddlStartYear class="date_combos" years-to-
display="10" required="false" start-date="true" />
                            <validationMsg for="startYear"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expMonth" text="Expiration
month (mm)" />
                            <tboxExpMonth class="xi-short-text" range-
msg="Your 'expiration month' validation message" />
                            <validationMsg for="expMonth"
class="valmsg" />
                        </tag>
                        <tag name="div">
                            <label for="expYear" text="Expiration year
(yy)" />
```

Sample 14: Custom messages

```
                            <tboxExpYear range="0,99" range-msg="Your
'expiration year' validation message" class="xi-short-text" exp-
date="false" />
                            <validationMsg for="expYear" class="valmsg"
/>
                        </tag>
                        <tag name="div">
                            <label for="cvv" text="Card cvv:" />
                            <tboxCvv class="xi-short-text"
minlength="3" maxlength="3" minlength-msg="Your 'CVV must be 3 digits'
validation message" maxlength-msg="Your 'CVV must be 3 digits'
validation message" digits-only="true" digits-only-msg="Your 'only
digits are allowed in CVV' validation message" />
                            <validationMsg for="cvv" class="valmsg" />
                        </tag>
                    </cardDropdownSection>
                </tag>
                <tag name="div" class="payment-button">
                    <cancelButton text="cancel" class="cancel-
button" />
                    <submitButton text="submit" class="submit-
button" />
                </tag>
            </tag>
        </tag>
        <tag name="div" class="copyright-footer">
            <tag name="div" class="footer-spacing"></tag>
            <tag name="div" class="top-bottom-frame"></tag>
            <tag name="div" class="middle-bottom-frame">
                <tag name="div" class="middle-bottom-frame-content">
                    Copyright © 2025. Worldpay.com. All Rights
Reserved. Various trademarks held by their respective owners. Legal
Notice.
                </tag>
            </tag>
        </tag>
    </tag>
</htmlSection>
```