https://verifone.cloud/docs/online-payments/strong-customer-authentication-sca/server-server-payments-3d-secure-setup-2
Updated: 14-May-2024

Inline Display Method for 3D Secure

Overview

The inline view framework allows merchants to receive the UI elements from Songbird and inject the elements directly into their page to appear as if their website has rendered the content. This view framework provides you with the most control over the UI , but also requires you to do the most amount of work.

Configuration for Inline

To use the inline option, the framework must be set to “inline” when running the configure() method.

Example

```
Cardinal.configure({
  payment: {
    framework: 'inline'
  }
});
```

Setup Events

With the inline view framework, we need to subscribe to a passive event, ui.inline.setup, for injecting the actual content into the merchant's website. This event will receive four arguments:

- **Html Template** - This is the element that needs to be injected.
- **Inline Data Object** - An object that describes the current UI experience and how the merchant should render it.

- **Resolve function** - A function to call when you have successfully injected the html template.

- **Reject function** - A function to call when you have encountered an error and are unable to inject the html template.

The merchant is required to inform Songbird when they have successfully injected the content into the page, or have failed to do so. Songbird will not move forward with the transaction until either the resolve or reject functions have been called. Be sure to have good error handling in this event handler or you could cause the consumer to be stuck if you don't properly call either resolve or reject.

Failure to call the resolve or reject functions may cause the transaction to get stuck. Songbird is waiting for you, the merchant, to tell it when you have finished with this asynchronous task.

The inline data object will contain information related to the current transaction and allow you to intelligently render for specific payment types, and different variations of each payment type. Additionally, this object will include the height and width of the html template to allow you to adapt the element receiving the html template accordingly.

Its important to know that html templates injected into the page will be cross-domain iframe. This means that if the injected html template is moved after Songbird has started the cross domain rendering, the content within the iframe will be whipped away by the browser. There is nothing Songbird or CardinalCommerce can do about this behavior, it is built into the browsers themselves.

Note: Once the html template has been injected, moving the elements around the DOM will cause the content to be deleted by the browser. For more on this, refer to this SO article.

Implementation Example

```
Cardinal.on('ui.inline.setup', function (
htmlTemplate, details, resolve, reject) {

  try {

    var container; // The element we will inject the HTML template into

    if (htmlTemplate !== undefined && details !== undefined) {


// Depending on your integration you may need to account for other items when pr


      switch (details.paymentType) {

        case 'CCA':

          // Process CCA authentication screen

          switch (details.data.mode) {
```

```
            case 'static':
              // Inject Iframe into DOM in visible spot
              container = document.getElementById('my-visible-wrapper');
              break;

            case 'suppress':
              // Inject Iframe into DOM out of view
              container = document.getElementById('my-hidden-wrapper');
              break;
            default:
              throw new Error("Unsupported inline mode found ["
 + details.data.mode + "]");
          }

          break;
        default:
          throw new Error("Unsupported inline payment type found ["
 + details.paymentType + "]");
      }

// Because the template we get from Songbird is a string template, we need to in

      container.innerHTML = htmlTemplate;

// Inform Songbird that we have successfully injected the iframe into the DOM an

      resolve();
    } else {
      throw new Error("Unable to process request due to invalid arguments"
);
    }

  } catch (error) {
```

```
// An error occurred, we need to inform Songbird that an error occurred so Songbi

    reject(error);
  }
});
```

Using the above implementation, you can specify the location of the pop-up by setting the relevant HTML element’s ID to “my-visbile-wrapper".

Example

```
<div id="my-visible-wrapper"> </div>
```

When Continue.continue() is called, the pop-up will display within the above div element.