



The typical sequence of the actions:

1. On startup, in bootstrap.js, a play preview extension has to register the target mime type(s) with the nsPluginHost [A];
2. and provide the stream converter from the application/x-moz-playpreview mime type to the HTML;
3. When the plugin is about to be instantiated on a web page, the nsObjectLoadingContent [D] requests if the mime type has been registered as a play preview one. If the IsPluginPlayPreviewForType returns true, the instantiation of the native plugin will be delayed (similar to the click to play functionality). The new plugin state was introduced and XBL overlay is extended to provide the additional HTML markup for the preview content;
4. XBL overlay code creates the IFRAME element, which points to the '<data:application/x-moz-playpreview;,<target-mime-type>>' URL;
5. The play preview stream converter [E] shall ignore all contents except one with the '<data:application/x-moz-playpreview;,<target-mime-type>>' URL;
6. When the request started, the original channel might be canceled or suspended and the new channel is used instead, e.g. 'resource://playpreviewextension/viewer.html'. The extension relies on the IFRAME security boundaries to limit the access to the privileged content;
7. If the play preview extension will decide to fallback to the original plugin, the extension shall dispatch a trusted 'MozPlayPlugin' event. The detail field of the event can be populated with the boolean value, which tells if click-to-play logic has to be skipped;
8. the XBL overlay code will invoke one of the methods of the plugin's nsObjectLoadingContent instance: a) the playPlugin() method – if click-to-play logic has to be skipped, or b) the cancelPlayPreview() – if play preview is not supported and click-to-play logic can be used;
9. On shutdown, in bootstrap.js, a play preview extension have to unregister the target mime type(s);
10. And unregister the stream converter from the application/x-moz-playpreview mime type.