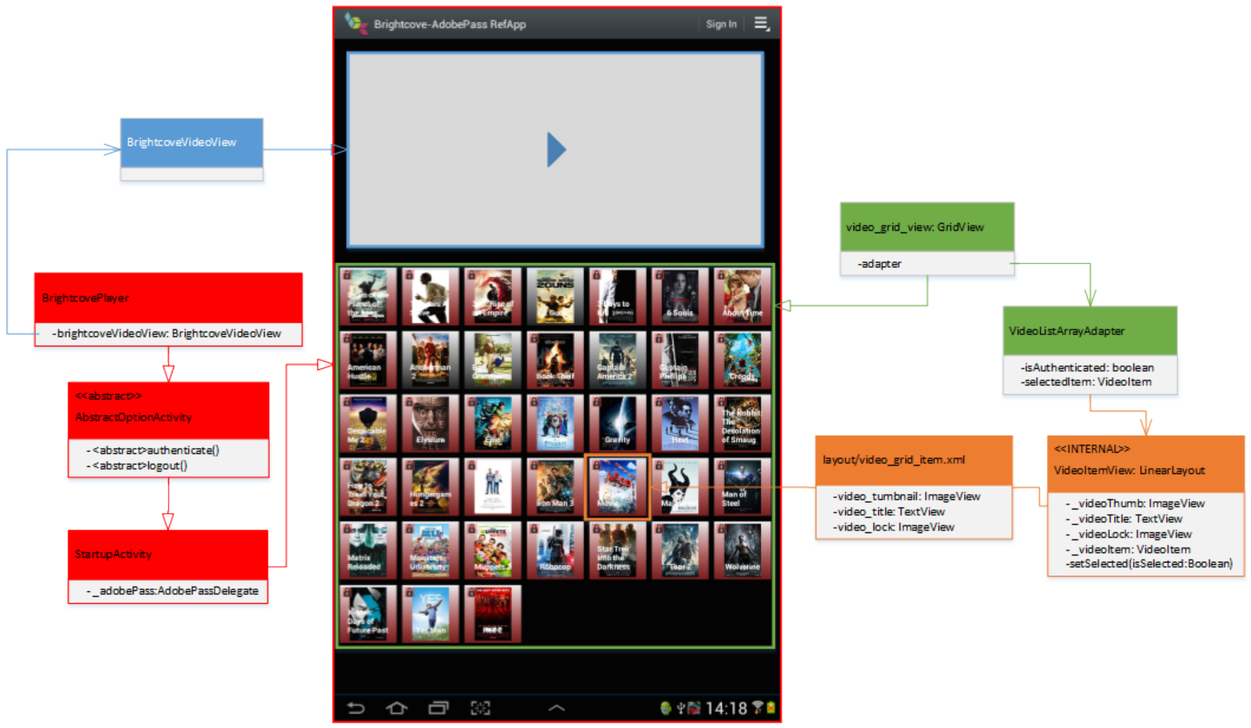
TVE Reference App [Android]



Overview

Android reference implementation for integrating Brightcove Player SDK with Adobe Pass, with end-to-end authentication, authorization, and video playback flows.

The Adobe Pass interaction is abstracted behind the com.brightcove.auth and com.brightcove.auth.ap packages with the AdobePassDelegate class as the entry point. This abstraction streamlines the initialization and also manages the different steps of the authentication flow.

Suggested Enhancements

- Create and integrate with a Configuration service to manage the Adobe Pass config data
- Integrate with Brightcove catalog service for retrieving the video metadata
- Create and integrate with a video retrieval service for validation and secure video stream endpoint
- Add integration with Akamai Identity Service (AIS)

Complete Happy Auth Flow

1. **[StartupActivity.java] Initiates the app**
   Inherits: BrightcovePlayer.java -> AbstractOptionActivity.java
   - 1.1. [StartupActivity:onCreate(Bundle savedInstanceState)]
     - 1.1.1. Initiates the BrightcovePlayer
     - 1.1.2. Register event listeners with the EventEmitter
     - 1.1.3. Initiates AdobePass
       - 1.1.3.1. Calls the AdobePass API method setRequestor
       - 1.1.3.2. Traps the setRequestor complete
       - 1.1.3.3. Calls the AdobePass API method checkAuthentication
       - 1.1.3.4. Traps the authentication result
       - 1.1.3.5. Calls the AdobePass API method getSelectedProvider
       - 1.1.3.6. Traps the get selected provider result
       - 1.1.3.7. Emits the AUTH_INITIATED event
     - 1.1.4. Initiates the VideoGrid
     - 1.1.5. Initiates the Video Delegate
   - 1.2. [AbstractOptionActivity:onCreateOptionsMenu(Menu menu)] Initiates the Option Menu
   - 1.3. [AbstractOptionActivity:onPrepareOptionsMenu (Menu menu)] Initiates the Option Menu
     - 1.3.1. Keeps the Authentication menu item instance as a local variable
   - 1.4. [AbstractOptionActivity:onOptionsItemSelected(MenuItem item)] Handles Option menu click
     - 1.4.1. If Close menu item -> Close App
     - 1.4.2. If AuthN menu item and not currently authenticated -> Invoke abstract method logout()
     - 1.4.3. If AuthN menu item and currently authenticated -> Invoke abstract method authenticate()
   - 1.5. [StartupActivity:initListener:processEvent(Event event)] Traps the emitted AUTH_INITIATED event
     - 1.5.1. Updates the VideoGrid authentication status
     - 1.5.2. Enables the Authentication menu item and sets the appropriate label text

2. **[menu/app_main.xml:menuItemAuthN] Authentication button clicked**
   - 2.1. [AbstractOptionActivity:authenticate() -> StartupActivity:authenticate()] Starts the Authentication flow
     - 2.1.1. Calls the AdobePass API method getAuthentication
     - 2.1.2. If authenticated, then emits AUTHENTICATED event
     - 2.1.3. If not authenticated, then emits DISPLAY_PROVIDER_SELECTOR event
   - 2.2. [StartupActivity:displayProviderSelectorListener:processEvent(Event event)]
     - 2.2.1. Create and initiate new Intent for the MvpdSelectorActivity class and then startup the activity
   - 2.3. [MvpdSelectorActivity]
     - 2.3.1. Create and initiate the MvpdListArrayAdapter
     - 2.3.2. Capture the item selected Provider

        - 2.3.3. Return the selected Provider back to the StartupActivity
    - 2.4. [StartupActivity:onActivityResult] MvpdListArrayAdapter complete and Provider selected
        - 2.4.1. Calls the AdobePass API method setSelectedProvider
    - 2.5. [StartupActivity:openLoginUrlListener:processEvent(Event event)]
        - 2.5.1. Create and initiate new Intent for the MvpdLoginActivity class and then startup the activity
    - 2.6. [MvpdLoginActivity]
        - 2.6.1. Create and initiate the MvpdLoginActivity
        - 2.6.2. Create and initiate the AdobePassLoginView
        - 2.6.3. Capture the Close event and return back to the StartupActivity
    - 2.7. [AdobePassLoginView]
        - 2.7.1. Initiating WebView and setting the WebViewClient
        - 2.7.2. [WebViewClient:shouldOverrideUrlLoading] Finalizes the Authentication Flow
        - 2.7.3. [AdobePassDelegate] Calls the AdobePass API method getAuthenticationToken
    - 2.8. [StartupActivity:authenticatedListener:processEvent(Event event)]
        - 2.8.1. Updates the authentication menu item label
        - 2.8.2. Updates the VideoGrid authentication status

3. **[StartupActivity:onVideoItemClickListener:onItemClick] Video Item selected**
    - 3.1. Gets the selected VideoItem
    - 3.2. Calls AdobePass API method authorize
    - 3.3. [StartupActivity:authorizedListener:processEvent(Event event)]
        - 3.3.1. [VideoDelegate:getVideo(IVideoItem videoItem, String shortMediaToken)]
            - 3.3.1.1. (Validate shortMediaToken at server side)
            - 3.3.1.2. Get Video rendition(s)
            - 3.3.1.3. Create a Brightcove Video instance
            - 3.3.1.4. Emit GOT_VIDEO event
    - 3.4. [StartupActivity:gotVideoListener:processEvent (Event event)]
        - 3.4.1. Start Video Playback

Components

com.brightcove.auth

Package for all authentication (AuthN) and authorization (AuthZ) classes. Both generic abstraction classes as well as specific Adobe Pass API abstraction.

The idea behind the abstraction is to make sure that the upper layers of the application could be made independent of the current Auth service in use, and that the current Adobe Pass library could be switch out for some other Auth service such as Akamai Identity Service without affecting any of the upper layers.

com.brightcove.auth.IAuthConfig.java

Interface for the configuration model

com.brightcove.auth.IAuthDelegate.java

Interface for the generic AuthN/AuthZ abstraction layer.

com.brightcove.auth.ap

Package for Adobe Pass specific abstraction and implementation

com.brightcove.auth.ap.delegates.AccessEnablerCallbackDelegate.java

Adobe Pass library (AccessEnabler) callback delegate, which will consume all the callbacks and emit the appropriate events on the EventEmitter.

com.brightcove.auth.ap.delegates.AdobePassDelegate.java

The main Adobe Pass abstraction layer, which extends the AccessEnablerCallbackDelegate.

It provides a simplified interaction with the underlying Adobe Pass library and will ensure the sequencing of calls is correct, and handle the initialization steps of the Adobe Pass library.

com.brightcove.auth.ap.model.AdobePassConfig.java

Encapsulating the required config data for the AdobePassDelegate

com.brightcove.auth.ap.model.ProviderFactory.java

Manages the conversion of the Adobe Pass Mvpd objects into the generic Provider objects.

com.brightcove.auth.ap.view.AdobePassLoginView.java

Extends the WebView and handles the loading and completion of the MVPD login page and authentication process.

com.brightcove.auth.model

Generic model interfaces and classes

com.brightcove.auth.model.IProvider.java

Provider model interface

com.brightcove.auth.model.IVideoItem.java

Video item model interface

com.brightcove.auth.model.Provider.java

Implementation of the IProvider interface to encapsulate the MVPD data

com.brightcove.examples

The actual reference application implementation

com.brightcove.examples.adapters.MvpdListArrayAdapter.java

ArrayAdapter for the MVPD selector ListView to handle the rendering of each MVPD list row

com.brightcove.examples.adapters.VideoListAdapter.java

ArrayAdapter for the Video selector GridView to handle the rendering of each Video item grid cell

com.brightcove.examples.delegates.VideoDelegate.java

Handles the validation and video retrieval for the selected and authorized video. Emits the GOT_VIDEO event on the EventEmitter

com.brightcove.examples.model.VideoItem.java

Implementation of the IVideoItem model interface to encapsulate the video metadata

com.brightcove.examples.model.VideoPlaylistFactory.java

Factory class to serialize the video JSON feeds into VideoItem objects

com.brightcove.examples.view.AbstractOptionActivity.java

Abstract Activity to render and handle the Option menu by adding a AuthN and Close menu item

com.brightcove.examples.view.MvpdLoginActivity.java

Activity to wrap the WebView AdobePassLoginView to manage the MVPD login process

com.brightcove.examples.view.MvpdSelectorActivity.java

Activity to wrap the ListView rendering the available providers using the MvpdListArrayAdapter and handle the selected provider event

com.brightcove.examples.view.StartupActivity.java

The main application Activity

com.brightcove.utils

Utility classes

com.brightcove.utils.DownloadImageTask.java

Utility class to handle the asyncrounous retrieval of an image, which is then rendered into the specified view component