WRP Android Video Player

for Unity Android Projects

by weRplay

Please read this read me thoroughly as it contains vital imformation for setting up the project.

1. Import the package to a Unity Android project.

2. Modify the Manifest file:

    If you do not already have an AndroidManifest file in the Assets>Plugins>Android folder, copy the AndrdoiManifest file from Assets>Plugins>WRP_AndroidVideoPlugin and paste into the Assets>Plugins>Android folder. Your manifest file is now ready.

    If you already have an AndroidManifest file in the Assets>Plugins>Android folder, then make the following changes:

    a. Add the following theme in your AndroidManifest file, inside <application> tag.

        android:theme="@android:style/Theme.Black"

    After this uour <application> tag should look similar to this:

        <application ....
        ....
        android:theme="@android:style/Theme.Black" >

    b. Remove this code if it is already written in your AndroidManifest.xml file:

    <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>

    c. Add the following between the <application .... > and </application> tags in the manifest file:

        <meta-data android:name="com.werplay.androidvideoplayerbasic.CallActivity" android:value="Activity_Helper" />
        <activity
        android:name="com.werplay.activityhelper.UnityPlayerNativeActivity"
        android:label="@string/app_name"
        android:configChanges="fontScale|keyboard|keyboardHidden|locale|mnc|mcc|navigation|orientation|screenLayout|screenSize
        |smallestScreenSize|uiMode|touchscreen">

            <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

        </activity>

    d. After the additions in the previous step, make sure your manifest file looks similar to the following format:

    ....
    ....
    ....
    <manifest ....
    .... >
    ....
    ....

    <uses-sdk ....
    .... >
    ....
    ....

    <application ....
    ....

```
android:theme="@android:style/Theme.Black" >
....
....

    <meta-data android:name="com.werplay.androidvideoplayerbasic.CallActivity" android:value="Activity_Helper" />
    <activity

        android:name="com.werplay.activityhelper.UnityPlayerNativeActivity"
        android:label="@string/app_name"
        android:configChanges="fontScale|keyboard|keyboardHidden|locale|mnc|mcc|navigation|orientation|screenLayout|screenSize|smallestScreenSize|uiMode|touchscreen">

        <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>

    </activity>

....
....

</application>
</uses-sdk>
</manifest>
```

Your manifest file is now ready.

3. Place the video file in the Assets>Plugins>Android>res>raw folder.
Also keep in mind the following rules:

a. Make sure that the video file name consisits only of lower case alphabets (a - z), numbers (0 - 9) and / or the underscore (_). Uppercase case characters and spaces are not allowed.

b. Only MP4 and 3GP (H.263, H.264 AVC, MPEG-4 SP) file formats are supported.

4. For the skip button, use any PNG file. Rename it to "theskipbutton2x.png" and place it in the Assets>Plugins>Android>res>drawable.

You can make other floders besides the "drawable" folder, named drawable-hdpi, drawable-ldpi, drawable-mdpi, drawable-xhdpi and drawable-xxhdpi. You can place different resolution images in these folders to be used for different resolution devices, depending on display resolutions.

For example:

res/drawable-mdpi/theskipbutton2x.png // for medium density
res/drawable-hdpi/theskipbutton2x.png // for high density
res/drawable-xhdpi/theskipbutton2x.png // for extra high density

5. Run the demo scene (Plugins>WRP_AndroidVideoPlugin>TestScene>TestScene) and take a look at the GuiManager.cs script (Plugins>WRP_AndroidVideoPlugin>TestScene>Script>GuiManager) to check out the working of the plugin.

Usage:

1. Drag and drop the AndroidVideoPlayer prefab from the Project pane (Assets>Plugins>WRP_AndroidVideoPlugin>Prefab>AndroidVideoPlayer) to the Hierarchy pane in Unity.

2. Call the functions on this object like following:

```
AndroidVideoPlayerBinding.instance.PlayVideo( videoName );
```

Please note that Unity is paused during video playback and resumes when video playback is finished.

List of Functions:

There are only 3 functions of note in the AndroidVideoPlayerBinding.cs script:

public void PlayVideo(string name)
- Plays a video file of given name
- Leave out the file extension from the video name when calling this function
- The video file must be placed in the Assets>Plugins>Android>res>raw folder

public void PlayVideo(string name, float xPosition, float yPosition)

- Plays a video file of given name and also shows a skip button
- Leave out the file extension from the video name when calling this function
- The skip button position is determined by xPosition and yPosition

public void VideoDonePlaying(string msg)
- This is a listener function
- It is called when the video playback is complete
- In case of normal playback duration completion, "Done" is passed as the argument
- In case of video end due to skip button, "Skip" is passed as the argument
- In case of video end due to minimizing application, "Minimize" is passed as the argument

For more Unity plugins and tools, please visit our website.
For further queries or information, please email us at unitysupport@werplay.com.