Capture And Save (Pro)

CaptureAndSave (Pro) plugin is very useful if you want to capture current screen and save that to camera roll/gallery. There are many APIs which help you to capture screen as a texture, save texture and video to gallery. You can also put any water mark on the screenshot.

Please note, this plugin will not record screen in video format, it will capture screen as still image only. But if you have your existing video then it will transfer that video to gallery.

Integration Guide:

Follow these steps to integrate CaptureAndSave (Pro) into your existing project

1). Import CaptureAndSave plugin into your project.

2). Check these files should be there
- /Assets/CaptureAndSave/Documentation/
- /Assets/CaptureAndSave/Example/
- /Assets/CaptureAndSave/Plugins/CaptureAndSave.dll
- /Assets/CaptureAndSave/Plugins/CallNative.dll
- /Assets/CaptureAndSave/Plugins/iOS/CallNative.dll
- /Assets/CaptureAndSave/Plugins/Android/AndroidGalleryUtil.jar
- /Assets/CaptureAndSave/Plugins/Android/RefreshGalleryWrapper.cs
- /Assets/CaptureAndSave/Plugins/Android/AndroidPermissionsManager.cs
- /Assets/CaptureAndSave/Plugins/iOS/libCaptureAndSave.a
- /Assets/CaptureAndSave/Prefab/CaptureAndSave.prefab

3). Mark CallNative.dll library for target platform

i). Select **/Assets/CaptureAndSave/Plugins/iOS/CallNative.dll** and uncheck “Any Platform” then mark **iOS** only (uncheck all others) under “Select platform for plugin” in inspector

ii). Select **/Assets/CaptureAndSave/Plugins/CallNative.dll** and uncheck “Any Platform” then mark **Editor, Standalone, Android** only (uncheck all others) under “Select platform for plugin” in inspector

4). Configure AndroidManifest.xml file

Add this meta tag in main activity
<meta-data android:name="unityplayer.SkipPermissionsDialog" android:value="true" />

```
<activity android:name="com.unity3d.player.UnityPlayerActivity"
          android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <meta-data android:name="unityplayer.UnityActivity" android:value="true" />
    <meta-data android:name="unityplayer.SkipPermissionsDialog" android:value="true" />
</activity>
```

(Photo : Example of AndroidManifest.xml)

5). Drag **CaptureAndSave** prefab into your hierarchy and set values in inspector.

FILENAME_PREFIX : This is name prefix of screenshot, final name will be followed by date and time.
ALBUM_NAME : Album name where all image will be saved.

6). Default directory where screenshot will save-
- Window (My Pictures) : C:\Users\<USERNAME>\Pictures
- MAC (Pictures): /Users/<USERNAME>/Pictures
- iOS : Camera Roll
- Android (with SDCard) : Pictures folder on SDCard, can be found in gallery
- Android (without SDCard) : /Data/bundle-identifier/files/(installation directory/files), not be there in gallery

Notes :

- ALBUM_NAME will be appended after default path, if there is no ALBUM_PATH then default directory will be final directory where all screenshot will be saved.

iOS Specific :

- **ALBUM_PATH** will not work on iOS.

- If you are using old version (below v1.5) then you need to add NSPhotoLibraryUsageDescription (also NSPhotoLibraryAddUsageDescription if building for iOS 11 or above) key in info.plist in xCode if you are using xCode 8.x.x, see this link to how to add keys in info.plist
- http://unitydevelopers.blogspot.in/2017/05/add-keys-into-infoplist.html

  If you are using CaptureAndSave Pro 1.5 or above then you don't need to add these keys, plugin will do automatically for you.

If you have any issues or suggestions then please add a comment here
http://unitydevelopers.blogspot.com/2014/06/capture-and-save-to-camera-roll-and.html

Other points to remember :

- *See the Example scene for more details of function calling.*
- *Deploy your project on iOS to see your captured image into camera roll, on editor it will not work.*
- *For android, write permission should be given in Player Settings.*

How to use:

1). Get reference of CaptureAndSave script

CaptureAndSave snapShot = GameObject.FindObjectOfType<CaptureAndSave>();

2). Set album path where all screenshot will save (optional & not for iOS)

snapShot.SetAlbumPath(albumPath);
Ex: albumPath = "D:/MyData/Pictures"; and similarly for each platform.

See : point no.(5) in integration section for default paths. If this is not set then default path will be consider.

3). Save full screenshots

i). *snapShot.CaptureAndSaveToAlbum(ImageType imgType);*

ii). *snapShot.CaptureAndSaveToAlbum (ImageType imgType, Texture2D watermark, WatermarkAlignment align)*

iii). *snapShot.CaptureAndSaveAtPath(string path,ImageType imgType);*

iv). *snapShot.CaptureAndSaveAtPath (string path, ImageType imgType, Texture2D watermark, WatermarkAlignment align);*

// save on a particular absolute path, will not work on IOS
Note : ImageType is a enum which indicates that type of image, values will be JPG or PNG.

4). Save particular area of the screen

i). *snapShot.CaptureAndSaveToAlbum (int x, int y, int width, int height, ImageType imgType);*

ii). *snapShot.CaptureAndSaveToAlbum (int x, int y, int width, int height, ImageType imgType, Texture2D watermark, WatermarkAlignment align);*

iii). *snapShot.CaptureAndSaveAtPath (int x, int y, int width, int height, string path, ImageType imgType);*

iv). *snapShot.CaptureAndSaveAtPath (int x, int y, int width, int height, string path, ImageType imgType, Texture2D watermark, WatermarkAlignment align);*
// save on a particular path, will not work on IOS

5). Save texture at path

i). *snapShot.SaveTextureAtPath (Texture2D tex2D, string path,ImageType imgType);*

ii). *snapShot.SaveTextureAtPath (Texture2D tex2D, string path, ImageType imgType, Texture2D watermark, WatermarkAlignment align);*

Note : For IOS path should be Application.persistentDataPath\<fileName> or Application.persistentData-Path\<Folder>\<filename>

: For Android it can be /storage/sdcard0/<folder>/<filename> or any path you want.

: For PC and MAC any path you want like /users/admin/Pictures etc.

6). Save screenshot in your customised resolution

i). *snapShot.CaptureAndSaveToAlbum (Screen.width * 2 , Screen.height * 2 , Camera.main, ImageType imgType);*

ii). *snapShot.CaptureAndSaveToAlbum (Screen.width * 2 , Screen.height * 2 , Camera.main, ImageType imgType,* Texture2D watermark, WatermarkAlignment align);

This will save screenshot with double resolution than screen's resolution.

***Note** : this will save screenshot which is rendered by camera passed as parameter. UI element created in OnGUI will not rendered by Camera.main therefore UI will not be saved in screenshot taken by this method.*

7). Save screenshot in your customised resolution at your path

i). *snapShot.CaptureAndSaveAtPath(Screen.width * 2, Screen.height * 2, Camera.main, string path,ImageType imgType);*

ii). *snapShot.CaptureAndSaveAtPath(Screen.width * 2, Screen.height * 2, Camera.main, string path,ImageType imgType,* Texture2D watermark, WatermarkAlignment align);

This will save screenshot with double resolution than screen's resolution at specific, in iOS it will save in camera roll instead of path.

***Note** : this will save screenshot which is rendered by camera passed as parameter. UI element created in OnGUI will not rendered by Camera.main therefore UI will not be saved in screenshot taken by this method.*

8). Save your texture in gallery

i). *snapShot.SaveTextureToGallery(Texture2D tex2D, ImageType imgType);*

9). Transfer your pre saved image from any path to CameraRoll, simply call this function

Copy : *snapShot.CopyImageToCameraRoll(string path)*
Move : *snapShot.MoveImageToCameraRoll(string path)*

where path is the absolute url of the image saved in document directory(in ios) or any where else (in Android or other platforms).

10). Transfer your pre saved video from any path to CameraRoll, simply call this function

Copy : *snapShot.CopyIVideoToCameraRoll(string path)*
Move : *snapShot.MoveVideoToCameraRoll(string path)*

where path is the absolute url of the image saved in document directory(in ios) or any where else (in Android or other platforms).

11). Get full screenshot

i). *snapShot.GetFullScreenShot(ImageType imgType)*
ii). *snapShot.GetFullScreenShot (ImageType imgType,* Texture2D watermark, WatermarkAlignment align);

12). Get specific screenshot

// particular screen
i). *snapShot.GetScreenShot (int x, int y, int width, int height, ImageType imgType);*

// particular screen with watermark
ii). *snapShot.GetScreenShot (int x, int y, int width, int height, ImageType imgType,* Texture2D watermark, WatermarkAlignment align);

***Note** : (Point 11 & 12)*
snapShot.GetFullScreenShot() and snapShot.GetScreenShot() will fire OnScreenShot event when screenshot ready.

13). Add watermark on your image

// Add watermark on background with given alignment.

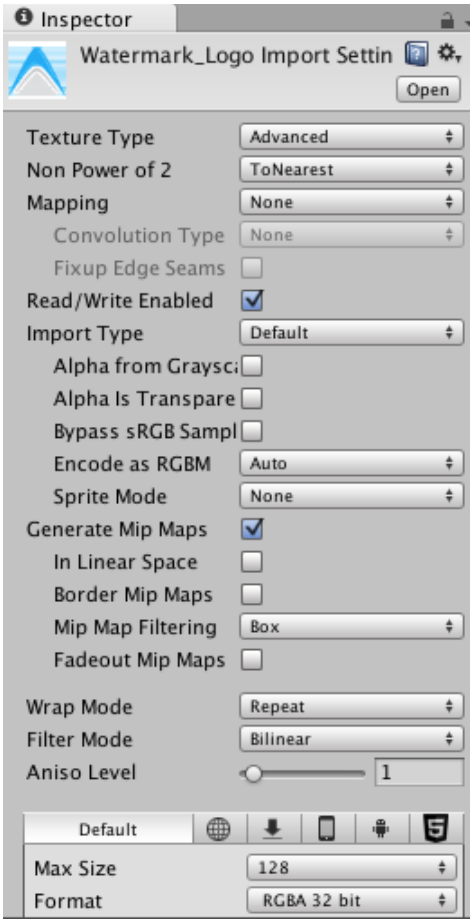
i). *snapShot.AddWatermark (* Texture2D background, Texture2D watermark, WatermarkAlignment align);

```
public enum WatermarkAlignment
{
        TOP_LEFT,
        TOP_RIGHT,
        BOTTOM_LEFT,
        BOTTOM_RIGHT,
        CENTER
}
```

// Add watermark on background at specific location.

ii). *snapShot.*AddWatermark(Texture2D background, Texture2D watermark, int startX, int startY);

***Note** : background & watermark texture both should be readable texture, you can mark it in texture setting in inspector as shown in following image.*



Events :

```
CaptureAndSaveEventListener.onError += OnError; // add event
CaptureAndSaveEventListener.onError -= OnError; // remove event
CaptureAndSaveEventListener.onSuccess += OnSuccess;// add event
CaptureAndSaveEventListener.onSuccess -= OnSuccess; // remove event
CaptureAndSaveEventListener.onScreenShotInvoker += OnScreenShot; // add
CaptureAndSaveEventListener.onScreenShotInvoker -= OnScreenShot; // Remove
```

```
void OnError(string error)
{
        Debug.Log ("Error : "+error);
}

void OnSuccess(string msg)
{
        Debug.Log ("Success : "+msg);
}

void OnScreenShot(Texture2D tex2D)
{
        Texture2D tex = tex2D;
        // use this texture anywhere you want.
}
```