Here is my file export\_scr.gml

#define AndroidCamera\_GetPicture

/// @description AndroidCamera\_GetPicture(async\_load);

/// @function AndroidCamera\_GetPicture

/// @param async\_load

var s = sprite\_add("gmcam\_capture.jpg", 1, 0, 0, 0, 0);

if (file\_exists("gmcam\_capture.jpg")) {

file\_delete("gmcam\_capture.jpg");

show\_debug\_message("FILE EXISTED");

} else {

show\_debug\_message("FILE DOESNT EXIST");

}

return s;

Now here is my extension AndroidCamera.ext:

package ${YYAndroidPackageName};

import android.util.Log;

import java.io.File;

import java.io.FileReader;

import java.io.BufferedReader;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.lang.String;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.nio.channels.FileChannel;

import ${YYAndroidPackageName}.R;

import com.yoyogames.runner.RunnerJNILib;

import android.content.Context;

import android.view.View;

import android.util.Log;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.view.KeyEvent;

import android.view.Menu;

import android.view.MenuItem;

import android.content.res.Configuration;

import android.app.Dialog;

import android.view.MotionEvent;

import android.net.Uri;

import android.provider.MediaStore;

import android.graphics.Bitmap.CompressFormat;

import java.lang.Thread;

import android.os.Build;

import android.content.pm.PackageManager;

import android.Manifest;

import android.Manifest.permission;

public class GMCamera

{

private static final int EVENT\_OTHER\_SOCIAL = 70;

private static final int CAMERA\_PERM\_REQUEST = 13;

private static final int ACTIVITY\_GOT\_PIC = 365;

private static final int ACTIVITY\_GOT\_PIC\_GALLERY = 366;

private Context c;

private Activity activity;

public double AddTwoNumbers(double arg0, double arg1)

{

double value = arg0 + arg1;

Log.i("yoyo", arg0 + "+" + arg1 + " = " + value);

return value;

}

public String BuildAString(String arg0, String arg1)

{

String myString = arg0 + " " + arg1;

Log.i("yoyo", myString);

return myString;

}

public void Init()

{

c = RunnerJNILib.ms\_context;

activity = RunnerActivity.CurrentActivity;

}

public void dispatchTakePictureIntent()

{

try

{

Log.i("camlisten", "cam started");

if (!hasCameraPermission())

{

getCameraPermission();

}

else

{

Intent takePictureIntent = new Intent(MediaStore.ACTION\_IMAGE\_CAPTURE);

if (takePictureIntent.resolveActivity(c.getPackageManager()) != null)

{

File f = new File(RunnerActivity.CurrentActivity.getApplicationContext().getFilesDir(), "gmcam\_capture.jpg");

f.delete();

//Create new file

FileOutputStream fos = c.openFileOutput("gmcam\_capture.jpg", Context.MODE\_WORLD\_WRITEABLE);

fos.close();

//Get reference to the file

File path = new File(RunnerActivity.CurrentActivity.getApplicationContext().getFilesDir(), "gmcam\_capture.jpg");

Uri uriSavedImage = Uri.fromFile(path);

takePictureIntent.putExtra(MediaStore.EXTRA\_OUTPUT, uriSavedImage);

takePictureIntent.putExtra("android.intent.extra.quickCapture", true);

activity.startActivityForResult(takePictureIntent, ACTIVITY\_GOT\_PIC);

}

}

}

catch (Throwable e)

{

Log.i("yoyo", e.getMessage());

}

}

public void dispatchGalleryIntent()

{

try {

Intent i = new Intent(Intent.ACTION\_GET\_CONTENT,

android.provider.MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI);

i.setType("image/\*");

i.putExtra("outputFormat", CompressFormat.JPEG.toString());

activity.startActivityForResult(i, ACTIVITY\_GOT\_PIC\_GALLERY);

} catch (Throwable e) {

Log.i("yoyo", e.toString());

}

}

public void Sleep(int i)

{

try

{

Thread.sleep(i);

}

catch(InterruptedException e)

{

//

}

catch (Throwable e)

{

Log.i("yoyo","sleep went wrong");

}

}

public boolean hasCameraPermission()

{

final Context c = RunnerJNILib.ms\_context;

final Activity activity = RunnerActivity.CurrentActivity;

return (activity.checkSelfPermission(Manifest.permission.CAMERA) == PackageManager.PERMISSION\_GRANTED);

}

private void getCameraPermission()

{

final Context c = RunnerJNILib.ms\_context;

final Activity activity = RunnerActivity.CurrentActivity;

if (activity.checkSelfPermission(Manifest.permission.CAMERA) != PackageManager.PERMISSION\_GRANTED)

{

activity.requestPermissions(new String[]{Manifest.permission.CAMERA}, CAMERA\_PERM\_REQUEST);

}

}

private void sendGMResult(int resultCode)

{

int dsMapIndex = RunnerJNILib.jCreateDsMap(null, null, null);

RunnerJNILib.dsMapAddString( dsMapIndex, "type", "camera");

if (Activity.RESULT\_CANCELED == resultCode) {

RunnerJNILib.DsMapAddDouble( dsMapIndex, "cancelled", 1);

} else {

RunnerJNILib.DsMapAddDouble( dsMapIndex, "cancelled", 0);

}

RunnerJNILib.CreateAsynEventWithDSMap( dsMapIndex, EVENT\_OTHER\_SOCIAL);

}

public void onActivityResult(int requestCode, int resultCode, Intent data)

{

if (requestCode == ACTIVITY\_GOT\_PIC)

{

sendGMResult(resultCode);

}

else if (requestCode == ACTIVITY\_GOT\_PIC\_GALLERY)

{

try {

File destFile = new File(RunnerActivity.CurrentActivity.getApplicationContext().getFilesDir(), "gmcam\_capture.jpg");

FileInputStream in = (FileInputStream) c.getContentResolver().openInputStream( data.getData());

FileOutputStream out = new FileOutputStream(destFile);

FileChannel inChannel = in.getChannel();

FileChannel outChannel = out.getChannel();

inChannel.transferTo(0, inChannel.size(), outChannel);

in.close();

out.close();

} catch (Exception t) {

}

sendGMResult(resultCode);

}

}

/\*

Faux methods

\*/

public void onStart()

{

}

public void onRestart()

{

}

public void onStop()

{

}

public void onDestroy()

{

}

public void onPause()

{

}

public void onResume()

{

}

public void onConfigurationChanged(Configuration newConfig)

{

}

public boolean onKeyLongPress(int keyCode, KeyEvent event)

{

return false;

}

public void onWindowFocusChanged(boolean hasFocus)

{

}

public void onRequestPermissionsResult(int requestCode,String permissions[], int[] grantResults)

{

if (requestCode == CAMERA\_PERM\_REQUEST)

{

//permission accepted. launch camera.

dispatchTakePictureIntent();

}

else

{

//launch permission denied async event?

int dsMapIndex = RunnerJNILib.jCreateDsMap(null, null, null);

RunnerJNILib.DsMapAddDouble( dsMapIndex, "permission", 0);

RunnerJNILib.dsMapAddString( dsMapIndex, "type", "camera");

RunnerJNILib.CreateAsynEventWithDSMap( dsMapIndex, EVENT\_OTHER\_SOCIAL);

}

}

public boolean onCreateOptionsMenu( Menu menu )

{

return false;

}

public boolean onOptionsItemSelected( MenuItem item )

{

return false;

}

public boolean onKeyDown( int keyCode, KeyEvent event )

{

return false;

}

public boolean onKeyUp( int keyCode, KeyEvent event )

{

return false;

}

public Dialog onCreateDialog(int id)

{

return null;

}

public boolean onTouchEvent(final MotionEvent event){ return false;};

public boolean onGenericMotionEvent(MotionEvent event){ return false;};

public boolean dispatchGenericMotionEvent(MotionEvent event) { return false;};

public boolean dispatchKeyEvent(KeyEvent event) { return false;};

}

} // End of class

I have made several functions based on this extension listed as name of extension, external name:

AndroidCamera\_Camera, dispatchTakePictureIntent

AndroidCamera\_HasPermission, hasCameraPermission

AndroidCamera\_Sleep, Sleep

AndroidCamera\_Gallery, dispatchGalleryIntent

These are my AndroidCameraScripts: \_AndroidCamera\_RotateImage, AndroidCameraFixOrientation, AndroidCamera\_Start, isset, and isset\_equality:

/// @description \_AndroidCamera\_RotateImage(sprite)

/// @function \_AndroidCamera\_RotateImage

/// @param sprite

function \_AndroidCamera\_RotateImage(argument0) {

//returns: Sprite

if (argument0 == -1) return -1;

var orig = argument0, ww = sprite\_get\_width(orig), hh = sprite\_get\_height(orig),

surf, newspr;

surf = surface\_create(hh,ww);

surface\_set\_target(surf);

draw\_sprite\_ext(orig, 0, hh, 0, 1, 1, -90, c\_white, 1);

surface\_reset\_target();

sprite\_delete(argument0);

newspr = sprite\_create\_from\_surface(surf, 0, 0, hh, ww, 0, 0, 0, 0);

surface\_free(surf);

return newspr;

}

/// @description AndroidCamera\_FixOrientation(image)

/// @function AndroidCamera\_FixOrientation

/// @param image

function AndroidCamera\_FixOrientation(argument0) {

switch (display\_get\_orientation())

{

case display\_portrait: case display\_portrait\_flipped:

return \_AndroidCamera\_RotateImage(argument0);

break;

default: return argument0;

}

return argument0;

}

/// @description AndroidCamera\_Start()

/// @function AndroidCamera\_Start

function AndroidCamera\_Start() {

// Polyfil for old function when there was no gallery option

AndroidCamera\_Camera();

}

function isset(argument0) {

return !is\_undefined(argument0);

}

/// @description isset\_equality(value,equality)

/// @function isset\_equality

/// @param value

/// @param equality

function isset\_equality(argument0, argument1) {

return (isset(argument0) && (argument0 == argument1))

}

This is the draw event for my object, obj\_camera\_test:

/// @description Draw Image and interface

// Draw GUI

var c\_text = $40343D, c\_bg = $F4CEFF;

draw\_clear(c\_bg);

draw\_set\_color(c\_text);

draw\_set\_halign(fa\_center);

draw\_set\_valign(fa\_middle);

draw\_text(room\_width/4, room\_height/10, string\_hash\_to\_newline("Image from Camera"))

draw\_text(3 \* room\_width/4, room\_height/10, string\_hash\_to\_newline("Image from Gallery"))

draw\_text(room\_width/2, room\_height / 20, string\_hash\_to\_newline("Double tap an image source to load from"));

draw\_line(room\_width/2, 0, room\_width/2, room\_height);

// Draw image (if available)

if (sprite\_exists(sprite\_index)) {

var ww, hh, asp = sprite\_get\_height(sprite\_index)/sprite\_get\_width(sprite\_index);

// Do some demo stuff to make it fit onscreen

ww = 2\*room\_width/3;

hh = ww \* asp;

if (hh > room\_height\*0.8) {

hh = room\_height \* 0.8;

ww = hh / asp;

}

draw\_sprite\_stretched(sprite\_index, 0, room\_width/2 - ww/2, room\_height/2 - hh/2, ww, hh);

}

else{

draw\_text(x+100,y+100,"sprite doesn't exist");

}

This is the global right pressed event

/// @description Double Tap the screen to launch

//LHS: Use camera

if (mouse\_x < room\_width/2)

AndroidCamera\_Camera();

else //RHS: Use gallery

AndroidCamera\_Gallery();

And this is an asynchronous event:

/// @description Get the picture

// Simple version

if (isset\_equality(async\_load[? "type"], "camera") && isset\_equality(async\_load[? "cancelled"],false)) {

if sprite\_exists(sprite\_index)

sprite\_delete(sprite\_index);

//retrieve photo

//with(GalleryObj){

//sprite\_index = AndroidCamera\_GetPicture(async\_load);

//}

//Having issues with orientation? Change the above line for this:

//with(GalleryObj){

sprite\_index = AndroidCamera\_FixOrientation(AndroidCamera\_GetPicture(async\_load));

//}

}

// Permission denial and cancellation detection

/\*

if (isset\_equality(async\_load[? "type"], "camera")) {

if (isset(async\_load[? "permission"])) {

//this is about permissions

if (!async\_load[? "permission"]) {

//permission was denied

//open camera to ask again

AndroidCamera\_Start();

}

} else if (isset\_equality(async\_load[? "cancelled"],false)) {

//photo was taken

if sprite\_exists(sprite\_index)

sprite\_delete(sprite\_index);

//retrieve photo

sprite\_index = AndroidCamera\_GetPicture(async\_load);

}

}

\*/

/\* \*/

/\* \*/

My main problem is that when I select an image from the gallery, it doesn’t appear on the following screen and instead it reads “sprite doesn’t exist” which it’s not supposed to at that point. Where is the error?