**The Classic Mini Game Engine**

Components

What is attached to each bean object. When writing your own components, make sure it extends from the components class in “com.bean.components”

Bean

These are the objects that the game runs on. Each bean holds components which each do separate things. A Scene hold beans.

Bean.getBean(String name) – returns a bean in the current scene if it finds one. Null is returned if no bean is present with that name.

Bean.addBean(Bean added) – adds the bean to the current scene.

Scene

A collection of beans. See the scene documentation to see how to write a scene.classicmini file. An application starts on start.classicmini, found in res/raw. A scene cannot have 2 beans with the same name.

Scene.loadScene(int resourceId) – loads in all beans and attriubutes from a .classicmini file.

Scene.mainloop() – runs the mainloop of each component on each bean. Run every frame.

Surface View

The center point of the program – holds everything.

surfaceView.startTime = the time the program was opened

surfaceView.deltaTime = the time taken to run the last frame

surfaceView.currentTime = the time currently the program has been open for

Button

A UI element for the classicmini engine.

Button.type = how the button should act when clicked

Button.BUTTON\_CLICK\_DOWN = peform an action when pressed down initially

Button.BUTTON\_HOLD = peform an action whilst the button is being clicked

Button.BUTTON\_CLICK\_UP = peform an action when the button press is released

Button.onClickClass = A component object on a bean where the button will run the onClick() method. Override this in a file that extends components.

Camera

A component which renders the scene from a point.

Camera.nearPlane = the near render distance of the camera

Camera.farPlane = the far render distance of the camera

Camera.backgroundColour = what colour the screen should be where nothing is rendered

Camera.setMain() – set this as the current camera which the scene uses

Collider

The component which stops rendered items going into each other.

Collider.solid = how the collider will work.

Solid colliders gets pushed by non solid colliders.

Solid colliders push other solid colliders.

Non solid goes through non solid collider.

Collider.minInfoForNoRenderer = the minimum info used to collide objects if no renderer is attached

Collider.maxInfoForNoRenderer = the minimum info used to collide objects if no renderer is attached

Dropdown

A list of buttons that dropdown to select a value. A dropdown cannot hold two of the same values.

Dropdown.dropdownItemsBegin – items must be assigned to this in a .classicmini scene file (string array)

Dropdown.dropdownItemInterval – the gap between each value of the dropdown

Dropdown.backgroundColour – the colour of each dropdown box behind text

Dropdown.textColourBegin – the colour of the text to be set in a .classicmini file

Dropdown.addItem(String item) – add a word to the end of a dropdown list

Dropdown.removeItem(String item) – remove a value from a dropdown list by its name

Dropdown.removeItem(int index) – remove a value from a dropdown by its entry index

Dropdown.getValue() – return the string of what is selected currently

Dropdown.setTextColour(Vec4 used) – set the text colour of the entries during runtime

Image

A UI element of the classicmini engine.

Image.colour – the colour to multiply the texture by

Image.roundEdgeRadius – how much to round the edges of an image by. This cannot be more than a value of the scale

Image.backgroundColour – the colour to draw if nothing is drawn in that position

Image.material – the classicminimaterial that gets the texture to draw

Light

A light that projects onto meshes, sprites and simplemeshses. The ClassicMini engine allows up to 10 lights per scene.

Light.lightColour – the colour of the light

Light.ambient – the lowest level the light can reach

Light.diffuse - the directional impact of a light on an object

Light.specular – creates the bright spot of light on an object

Light.constant – constant reduction of light effect

Light.linear – reduction due to distance

Light.quadratic – reduction due to distance also

Mesh

A 3D model which supports light and textures.

Mesh.vertices – the points to be set in the .classicmini file. (3 floats for position, 3 normals, 2 uv)

Mesh.material – the classicminimaterial that gets the texture to draw

Mesh.useLight – use light or not

Mesh.colour – the colour to multiply the texture colour by

Mesh.outputOBJVertices(int resourceId) – output the mesh vertices into the logcat console. Must be formatted with textures and normal for meshes. For debugging.

OBJ files must be formatted with normals, textures, triangulated faces, no edges and obj groups.

ParticleSystem

A point which creates particles infinitely and sends them in a direction.

ParticleSystem.resolution = the type of mesh to draw for each particle

ParticleSystem.LOW\_RESOLUTION = triangle particles

ParticleSystem.MED\_RESOLUTION = pyramid particles

ParticleSystem.MAX\_RESOLUTION = cube particles

ParticleSystem.particleGenerateTime = the time between each particle generation

ParticleSystem.particleSpeed = speed of a particle in its direction

ParticleSystem.particleDeleteDistance = the distance of the particle from the particlesystem to delete it

ParticleSystem.particleXRotationBounds = the limits for a random x rotation of a particle

ParticleSystem.particleYRotationBounds = the limits for a random y rotation of a particle

ParticleSystem.particleZRotationBounds = the limits for a random z rotation of a particle

SimpleMesh

A mesh without lights or texture – uses much less memory.

SimpleMesh.vertices = the vertices for the mesh to draw – to be set in the .classicmini scene file. (3x float for position)

SimpleMesh.vcolour = the colour of the mesh

SimpleMesh.outputOBJVertices(int id) – output the simplemesh vertices from an obj file.

OBJ files must be formatted with normals, textures, triangulated faces, no edges and obj groups.

Slider

A side to side slider that returns a value from touch.

Slider.colour = the colour of the slider and background

Slider.minimumValue = the value to return when the slider is far left

Slider.maximumValue = the value to return when the slider is far right

Slider.getValue() return the current value

Sprite

A 2D square that renders a texture.

Sprite.useLight = use light or not

Sprite.material – the classicminimaterial that generates a texture

Sprite.colour = the colour that is multiplied by the texture

Text

A text rendered to the screen.

Text.material – the classicminimaterial that generates the text texture

Text.colour = the colour of the text

Text.backgroundColour = the colour of the box around the text

Text.realTextScale = if true, make the text not squashed

Text.setText(String newText) – set the displayed text during runtime – don’t run this every frame

Text.setTextSize(int newSize) – set the text size during runtime – don’t run this every frame

Text.setTextCentered(Boolean as) – set the text centered during runtime – don’t run this every frame

Text.setTextFont(int newFont) – newFont is resourcePath, set font in runtime. Don’t run this every frame

Toggle

A switch.

Toggle.colour = the colour of the toggle and the background.

Toggle.isToggled() – returns the current value

Toggle.setRoundEdgeRadius(float radius) – set the round edges of the background and the switch

Transform

The component used in classicmini to hold position, rotation, scale and parents.

Transform.position = the position of the Bean

Transform.rotation = the rotation of the Bean

Transform.scale = the scale of the Bean

Transform.velocity = speed in a direction (1, 0, 0) for forward

Transform.upVector() = the upwards direction of the Bean

Transform.forwardVector() = the forward direction of the Bean

Transform.rightVector() = the right direction of the bean

Transform.getHigestParent() – return the highest parent of parent of parent of…

Transform.toMatrix4() – the matrix used to multiply to vectors to achieve a result.

Transform.getRelativePosition() – the position of the transform including parent’s information

Transform.getRelativeScale() – the scale of the transform including parent’s information

Transform.setRelativePosition(Vec3 position) – set the position relative to parent information

Transform.setRelativeScale(Vec3 scale) – set the scale relative to parent information

Transform.gameGravitySpeed = the speed of gravity the entire game uses.

Transform.useGravity – whether or not this object should use gravity.

ClassicMiniAdverts

ClassicMini’s implementation of google ads. Make sure to follow the ClassicMini building guide if using these. Rewarded ads are videos that reward the user when watched fully. Interstitial ads are fullscreen ads. Banners are adverts at the bottom of the screen.

ClassicMiniAdverts.loadRewardedAd() – load an advert. Make sure an ad id is set

ClassicMiniAdverts.showRewardedAd() – show the rewarded ad if one has finished loading

ClassicMiniAdverts.setRewardedAdListener() – set callbacks for the classicmini rewarded ad. Refer to <https://developers.google.com/android/reference/com/google/android/gms/ads/reward/RewardedVideoAdListener>

For help setting a listener up

ClassicMiniAdverts.loadInterstitialAd() – load an advert. Make sure an ad id is set

ClassicMiniAdverts.showInterstitialAd() – show interstitial ad if one is loaded

ClassicMiniAdverts.setInterstitialAdListener() – set callbacks for the classicmini interstitial ad. Refer to

<https://developers.google.com/custom-search-ads/android/afsnative/reference/com/google/android/gms/ads/afsn/AdListener>

for help setting one up.

ClassicMiniAdverts.loadBannerAd() – load a banner ad . make sure an id is set. Banner ads show automatically

ClassicMiniAdverts.setBannerAdListener() – set callbacks for the classicmini banner ad. Refer to

<https://developers.google.com/custom-search-ads/android/afsnative/reference/com/google/android/gms/ads/afsn/AdListener>

for help setting one up

ClassicMiniAdverts.getBannerAd() – used for setting the banner ad in front of the game (dealt with already)

ClassicMiniAdverts.getAdParams() – used for setting the banner ad in front of the game (dealt with already)

ClassicMiniAdverts.getMainRelativeLayout) – used for setting the banner ad in front of the game (dealt with already)

ClassicMiniAdverts.begin() – initialize google ads. Dealt with already, do not run this.

ClassicMiniAudio

The audio engine for classicmini. The engine only allows 100 duplicates of the same audio track. For each function, a second optional parameter is used to ask which version of the track of audio you want to interact with.

ClassicMiniAudio.ClassicMiniAudioMainloop() – update audio samples, run every frame. Dealt with already.

ClassicMiniAudio.loadAudio(id, \*duplicate\*) = load audio

ClassicMiniAudio.playAudio(id, \*duplicate\*) = play audio, make sure loaded first.

ClassicMiniAudio.stopAudio(id, \*duplicate\*) = stop audio. Make sure loaded first

ClassicMiniAudio.pauseAudio(id, \*duplicate\*) = pause audio. Make sure loaded first.

ClassicMiniAudio.resumeAudio(id, \*duplicate\*) = resume audio. Make sure loaded first.

ClassicMiniGIFMaterialInfo

A subclass used in ClassicMiniMaterial.

ClassicMiniGIFMaterialInfo.gifPath = the resource path to the gif file. This must be stored in raw

ClassicMiniTextMaterialInfo

A subclass used in ClassicMiniMaterial.

ClassicMiniTextMaterialInfo.displayedText – the text on a texture. Must be set in scene file.

ClassicMiniTextMaterialInfo.textSize = the text size on a texture (defines the resolution of the image quality). Must be set in scene file.

ClassicMiniTexMaterialInfo.textCenterd = whether or not the text should be centered. Must be set in scene file.

ClassicMiniTextMaterialInfo.fontPath – the resource path of the selected font. Must be set in scene file.

ClassicMiniTextMaterialInfo.textColour – the colour of the text. Must be set in scene file.

ClassicMiniMaterial

The material class used to generate textures for the classicmini engine.

ClassicMiniMaterial.rgbToHex(Vec3i used) – convert rgb values to hex values.

ClassicMiniMaterial.rgbToHex(Vec3 openglValues) – convert opengl values (0.0 – 1.0) values to hex values.

ClassicMiniMaterial.type = the type of texture to load (either colour, image, text or gif)

ClassicMiniMaterial.begin() = load the texture

ClassicMiniMaterial.imagePath – the resource path of a selected image

ClassicMiniMaterial.loadImage(id) – load image and assign information

ClassicMiniMaterial.textMaterialInfo – the text information of the material

ClassicMiniMaterial.getXTextMultiplier() – multiply x scale by this to get unsquashed text

ClassicMiniMaterial.loadText(int size, String text, Boolean centered, Vec4 colour) – load text texture and assign information

ClassicMiniMaterial.colourHex = the colour of the texture

ClassicMiniMaterial.loadColour(string colourHex) – load colour texture and assign information

ClassicMiniMaterial.gifMaterialInfo – the gif information of the material

ClassicMiniMaterial.loadGif(int id) – load gif from resource path and assign information

ClassicMiniOutput

Output information. If the engine does not support your variable as a parameter, use String.valueOf(variable) to set parameter.

ClassicMiniMath

Math functions for classicmini.

ClassicMiniMath.getOrtho() – get the orthographic matrix for the program

ClassicMiniMath.classicMiniCos(degrees) – the cos function using degrees instead of radians

ClassicMiniMath.classicMiniSin(degrees) – the sin function using degrees instead of radians

ClassicMiniMath.highestCommonFactor(numOne, numTwo) – get highest factor of two numbers

ClassicMiniMath.roundDecimal(decimal, places) – round a decimal to a certain number of places

ClassicMiniMath.vectorTwoEquals(one, two) – check if two vector twos are equal

ClassicMiniMath.vectorThreeEquals(one, two) – check if two vector threes are equal

ClassicMiniMath.vectorFourEquals(one, two) – check if two vector fours are equal

ClassicMiniMath.vectorThreeDistance(one, two) – get distance between two vector threes

ClassicMiniMath.vectorThreeLessThan(one, two) – check if each point of one is less than two

ClassicMiniMath.copyVectorFour(used) – create a new instance of vector four with the same values

ClassicMiniMath.copyVectorThree(used) – create a new instance of a vector three with the same values

ClassicMiniMath.copyVectorTwo(used) – create a new instance of a vector two with the same values

ClassicMiniMath.bearing(one, two) – get the angle bearing from one to two

ClassicMiniMath.randomInteger(lowInclusive, maxInclusive) – get a random integer between two numbers, including them

ClassicMiniMath.randomFloat(lowInclusive, maxInclusive, places) – get a random float between two floats with a certain amount of places.

ClassicMiniMath.touchToUICoords(x, y) – convert touch point to ui co ordinates

ClassicMiniShaders

The open gl shader functions for classicmini.

ClassicMiniShaders.createShader(int codePath, shaderType) – create an opengl shader with a resource path and opengl shadertype.

ClassicMiniShaders.createProgram(int[] shaders) – create an opengl shader program with different shader types

The rest of the functions pass uniform variables to a shader in the format : (variable, uniform name, shader number)

ClassicMiniSavefiles

The save files function for the classicmini engine.

ClassicMiniSavefiles.readLines(id) – read lines of a resource file into a list of strings

ClassicMiniSavefiles.readLinesString(id) – read lines of a resource file into a continuous string

ClassicMiniSavefiles.readAsset(path) – read lines of an asset on the device storage

ClassicMiniSavefiles.writeAsset(path, lines, append) – write lines to an asset on device storage choosing wether to add to existing or not

ClassicMiniSavefiles.createAsset(path) – create a file on device storage

ClassicMiniSavefiles.deleteAsset(path) – delete file on device storage