Documentation - CameraFocus

CameraFocus is a 2D-Camera class designed for use with Starling Framework.
Check out the source code from GitHub: https://github.com/joeonmars/CameraFocus/
The master branch is created and maintained by joeonmars: http://www.joeonmars.com/

Getting Started

CameraFocus offers a set of interface associated with a CameraFocus instance. It allows Flash developers to easily simulate 2D camera effect without sophisticated math. CameraFocus is good for any 2D games/applications stressing upon an engaging experience.

From a technical perspective, the routine of CameraFocus is constantly altering the geometric properties of a root level container, which wraps up all display objects that should interact with the camera. This concept ensures that the camera would never interfere with the movements of your display objects (except for offsetting the parallax layers).

So, first of all, please construct your display list as illustrated below:

Starling Stage > Display Object - stage container > DisplayObject – layer1
> DisplayObject – layer2
> DisplayObject – layer3
> DisplayObject – ...

Or simply refer to this code snippet:

```
var stageContainer:Sprite = new Sprite();
stageContainer.addChild( background );
stageContainer.addChild( foreground );
stageContainer.addChild( character );
Starling.current.stage.addChild( stageContainer );
```

Another tip worth your attention is that CameraFocus does calculations based on the top left (0, 0) corner of each layer that you provided. That means if you want to make parallax scrolling, the handy pivotX, pivotY from Starling framework may not work properly in joint with this class. Though you would become very careful when using the pivot properties for individual layers, the left corner consistency would save your efforts in tweaking the registration point. Because CameraFocus has already integrated the logic of aligning and scaling multiple layers, thus you can fully rely on CameraFocus to handle those implementations.

Constructor

StarlingCameraFocus(stage:Stage, stageContainer:DisplayObject, focusTarget:*, layersInfo:Array, autoStart:Boolean = false)

Parameters
stage:Stage - The Starling Stage.(not the native stage)

stageContainer:DisplayObject - A wrapper that contains all display objects to move with the camera. Normally a Sprite.

focusTarget:* - The initial focused object, could be of any type but must own x and y properties.

layersInfo:Array - A group of objects that holds each layer's properties. For example:

```
var layersInfo:Array = [
        {name:'background', instance:this.background, ratio:.2},
        {name:'midground', instance:this.midground, ratio:.1},
        {name:'foreground', instance:this.foreground, ratio:0},
        {name:'charaters', instance:this.characters, ratio:0}, ];
```

The above code provides necessary data for CameraFocus to simulate parallax scrolling effect. Consider 'ratio' as z-depth in 3D space. Values between 0 to 1 are commonly used. A value of 0 indicates the closest distance from the camera. Thus the greater the ratio is, the farther the layer is.

autoStart:Boolean - Determines whether the camera should start working as long as is created.

```
var stageContainer:Sprite = new Sprite();
Starling.current.stage.addChild( stageContainer );

var character:Sprite = new Sprite();
var background:Sprite = new Sprite();
background.addChild( character );

stageContainer.addChild( background );

camera = new StarlingCameraFocus( Starling.current.stage, stageContainer,
character, [{name:'background',instance:background,ratio:0}], true );
```

The above code creates a CameraFocus instance that follows a character right to the spot.

Public Properties

focusTarget:* - The current focused object. If you want to transfer the focus to another object, instead of setting this property, try using jumpToFocus() or swapFocus() method.

trackStep:int - Defines the number of steps the camera would take to reach its focus. The minimum step is 1.

swapStep:int - Defines the number of steps the camera would travel between two focuses. The minimum step is 1.

zoomStep:int - Defines the number of steps the camera would take to zoom. The minimum step is 1.

ignoreLeftBound:Boolean - If false then the camera would stop tracking as the boundary layer reaches the left of the stage.

ignoreRightBound:Boolean - If false then the camera would stop tracking as the boundary layer reaches the right of the stage.

ignoreTopBound:Boolean - If false then the camera would stop tracking as the boundary layer reaches the top of the stage.

ignoreBottomBound:Boolean - If false then the camera would stop tracking as the boundary layer reaches the bottom of the stage.

enableCallBack:Boolean - Turn on/off dispatching CameraFocus events.

zoomFactor:Number - [read-only]Get the camera's end zooming factor which was passed into zoomFocus()

isFocused:Boolean - [read-only]Whether the camera has reached its focus.

isSwaping:Boolean - [read-only]Whether the camera is during the swaping transition.

isZooming:Boolean - [read-only]Whether the camera is during the zooming transition.

isShaking:Boolean - [read-only]Whether the camera is shaking.

Public Methods

getLayerByName(name:String):DisplayObject - Returns the layer with the name registered via layersInfo.

start():void - Start/Resume running CameraFocus.

pause():void - Pause CameraFocus.

destroy():void - Mark CameraFocus for garbage collection.

setFocusPosition(x:Number, y:Number):void - Set a focus position relative to Stage. It is the stage x and y that your focused object will be gradually aligned to.

For example:
```
camera.setFocusPosition( stage.stageWidth/2, stage.stageHeight/2 );
```
It sets the mid-point of the stage as the destination position that the focused object will appear on the stage.

setBoundary(layer:DisplayObject):void - Passed in a layer for boundary detection.

jumpToFocus(focusTarget:*=null):void - Locate to the focused object instantly. Optionally, you can take the chance of re-assigning the focus via the focusTarget argument.

swapFocus(focusTarget:*, swapStep:int=10, zoom:Boolean=false, zoomFactor:Number=1, zoomStep:int=10):void - Transit the focus to another object. Zooming is also applicable during transition.

zoomFocus(zoomFactor:Number, zoomStep:uint=10):void - Zoom a focus.

shake(intensity:Number, shakeTimer:int):void - Shake the camera.

Parameters
intensity:Number - The intensity of shaking.
shakeTimer:int - The shaking duration in frames.

update():void - Run the CameraFocus internal logic. Ideally be called at the end of an enterframe event. Or right after setting the x/y properties of objects.

Events

CameraFocusEvent(type:String, bubbles:Boolean=**false**)

type:String
CameraFocusEvent.HIT_BOUNDARY - Dispatched when the top/bottom/left/right coordinates of boundary layer collides with the stage's boundaries.

CameraFocusEvent.SWAP_STARTED - Dispatched when swaping starts.

CameraFocusEvent.SWAP_FINISHED - Dispatched when swaping completes.

CameraFocusEvent.ZOOM_STARTED - Dispatched when zooming starts.

CameraFocusEvent.ZOOM_FINISHED - Dispatched when zooming completes.

CameraFocusEvent.SHAKE_STARTED - Dispatched when shaking starts.

CameraFocusEvent.SHAKE_FINISHED - Dispatched when shaking completes.

boundary:String - Use with CameraFocusEvent.HIT_BOUNDARY event type. The returned values are 'left', 'right', 'top' or 'bottom'. For example:

```
Starling.current.stage.addEventListener( CameraFocusEvent.HIT_BOUNDARY, onHitBoundary );

function onHitBoundary( e:CameraFocusEvent ):void
{
	switch( e.boundary )
	{
		case 'left':
		case 'right':
		case 'top':
		case 'bottom':
		trace("hit boundary!");
		break;
	}
}
```

Note: The CameraFocusEvent extends Starling Event rather than Flash Event.