Archived:How to develop brick-breaker game in PySymbian - Part 1

Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

The article is believed to be still valid for the original topic scope.

Brick-Breaker is a simple arcade game where the player moves a paddle on the screen and bounces a ball towards bricks at the top of the screen. When the ball hits a brick it breaks and the ball bounces back - the objective of the game is to destroy all the bricks. This is the first article explaining the PySymbian implementation of Brick-Breaker - it covers the different slider functions in detail.

Development tools

02 Aug 2009

1. Text editor - notepad is sufficient.
2. PySymbian - preferred version: PySymbian 1.4.5 stable release
3. PySymbian script shell - preferred version: PySymbian 1.4.5 stable release

The PySymbian tools can be downloaded from Sourceforge resources.

The following modules are used when developing this application:

- appuifw module
- sysinfo module
- graphics module
- e32 module
- key_codes module
- random module
- math module
- time module

Functions

The following functions are used in Brick Breaker game.

1. **`draw_screen`**: draw black screen in canvas.
2. **`draw_slider`**: draw slider.
3. **`leftkey`**: control the slider by left key.
4. **`rightkey`**: control the slider by right key.
5. **`draw_bricks`**: draw number of bricks.
6. **`erase_bricks`**: erase bricks.

Initializing slider parameter

Before discussing, any functions in detail - let us define some of the slider parameters.

```
#SLIDER PARAMETERS#####

SliderX1=120
SliderY1=305
SliderX2=160
SliderY2=315
SliderSpeed=22

SliderColor=RED
```


(C) Copyright Nokia 2014. All rights reserved.

```
import sysinfo
TotalScreenWidth=sysinfo.display_pixels()[1]
```

draw_screen()

By using this function, We create a black screen in canvas. The definition of the draw_screen() is as follows:

```
#defining draw function
def draw_screen():
       #Set the screen to full
       appuifw.app.screen = 'full'

       #Global variable for application UI
       global canvas
       canvas = appuifw.Canvas(redraw_callback=handle_redraw)
       appuifw.app.body = canvas
       appuifw.app.exit_key_handler = quit

       #Global variable for application UI
       global img
       img = graphics.Image.new(canvas.size)

       #Clear the image
       img.clear(BLACK)
       handle_redraw(None)
```

draw_slider()

By using this function, we create a slider in canvas using keyboard shortcuts. The definition of the draw_screen() is as follows:

```
#defining slider function
def draw_slider():
 global canvas

 #Keyboard shortcuts
 canvas.bind(key_codes.EKeyLeftArrow,lambda:  leftkey())
 canvas.bind(key_codes.EKeyRightArrow,lambda: rightkey())
 appuifw.app.body = canvas
 appuifw.app.exit_key_handler = quit

       #draw the rectangle
 img.rectangle((SliderX1,SliderY1,SliderX2,SliderY2),fill=SliderColor)
 handle_redraw(None)
```

leftkey()

The leftkey() is called from the draw_slider() function. By using this function, we can move the slider to left side by pressing left key. Thus, the code of leftkey() function is as follows:

```
#defining leftkey function
def leftkey():
 if (SliderX1>0):
```

 (C) Copyright Nokia 2014. All rights reserved.

```
                #draw the rectangle
   img.rectangle((SliderX1, SliderY1, SliderX2, SliderY2),fill=BLACK)

                #globally define variables
   global img, canvas,SliderX1, SliderY1, SliderX2, SliderY2

                #defining the slider coordinate X1 and X2
   SliderX1=SliderX1-SliderSpeed
   SliderX2=SliderX2-SliderSpeed
   img.rectangle((SliderX1, SliderY1, SliderX2, SliderY2),fill=SliderColor)

         #appuifw.note(u"Left arrow was pressed")
   handle_redraw(None)
```

rightkey()

The rightkey() is called from the draw_slider() function. By using this function, we can move the slider to right side by pressing right key. Thus the code of rightkey() function is as follows:

```
#defining right key function
def rightkey():
 if (SliderX2<TotalScreenWidth):
 #if (SliderX2<sysinfo.display_pixels()[1]):
                #draw rectangle
   img.rectangle((SliderX1, SliderY1, SliderX2, SliderY2),fill=BLACK)

                #globally define variables
   global img, canvas,SliderX1, SliderY1, SliderX2, SliderY2

               #defining the slider coordinate X1 and X2
  SliderX1=SliderX1+SliderSpeed
  SliderX2=SliderX2+SliderSpeed
  img.rectangle((SliderX1, SliderY1, SliderX2, SliderY2),fill=SliderColor)

  #appuifw.note(u"Left arrow was pressed")
   handle_redraw(None)
```

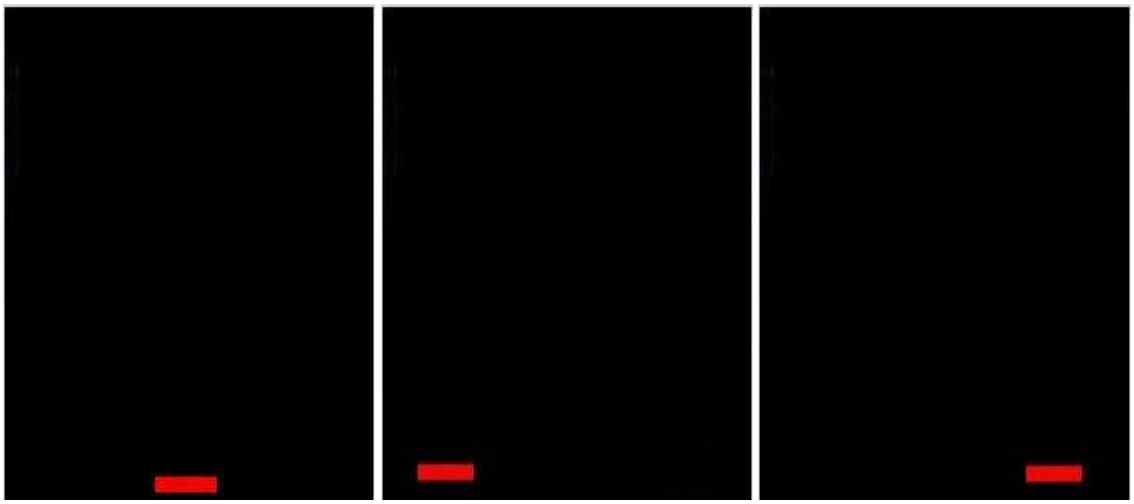
That finishes the slider functions (Part 1) for the Brick-Breaker Game.

The next article in the series : Part 2 for brick-Breaker, would cover Brick functions for this game.

Screenshots

These screenshots demonstrate the Slider and its movable position on canvas. The screenshots are relevant only to Part 1.

(C) Copyright Nokia 2014. All rights reserved.



Related Links

- How to develop brick-breaker game in PySymbian - Part 2
- How to develop brick-breaker game in PySymbian - Part 3
- Introduction to PySymbian

(C) Copyright Nokia 2014. All rights reserved.