Easel$_{PY}$

1. Introduction

Easel is an engine for creating real time games.This version, Easel$_{PY}$, allows the programmer to create games by writing Python code, and runs on Windows or Mac OS.

2. Data model

Using Easel$_{PY}$ requires understanding the following data types which are used by the engine.

- A *point* is a pair (*x,y*) where *x* and *y* are integers. The point (x,y) is thought of a point in a coordinate plane with (0,0) in the center of the game window, the x-axis point right and y axis pointing up.
- A *color* is written (*R,G,B*) where R, G, and B are integers and 0 ≤ *R,G,B* ≤ 255
- An *image* is a *segment*, *circle*, *filled triangle, text image, disc* or *file image* loaded from a file. Image should be written as a polymorphic class, with a different draw function for each kind of image. The image class is imported with EaselLib.py
    - *A segment* is written seg(*p*,*q*,*C*) where *p* and *q* are points, interpreted as the endpoints of the segment, and *C* is a color.
    - A *circle* is written circ(*p*, *r*, *C*) where *p* is a point, *r* is a positive integer, and *C* is a color. We interpret *p* and *r* as the center and radius of the circle, respectively.
    - A *filled triangle* is written ftri(*p, q, r, C*) where *p*, *q*, and *r* are points and *C* is a color. The filled triangle ftri(*p, q, r, C*), where *p, q,* and *r* are noncollinear points and *C* is a color, represents the filled triangle with vertices *p, q,* and *r,* of color *C*.
    - *A text image* is written txt(*S, p, n, C*) where S is a string, *p* a point, *n* integer in [4,100], and *C* is a color. The text image `txt`(*S, p, n, C*) represents the text string *S*, displayed with height *n* centered at *p* with color C.
    - A *Disc* is written `disc`(*p*, *r*, *C*) where *p* is a point, *r* is a positive integer, and *C* is a color. We interpret *p* and *r* as the center and radius of the circle, respectively.
    - A *file image* is written fileImg(I,p) where I is an image loaded from a file (see below) and p is a point. We interpret p as the point where the top-left corner of I is located
- A *sprite* is a list of images
- A *click* is either a point or `None`
- A *sound* is either one of the following global constants (found in EaselLib.py): DING, BANG, BOING, CLAP, CLICK, or a sound loaded from a file (see below)
- A *key* is an integer. Keys are named by global variables which are imported with EaselLib.py, given in the first column of the table in Appendix A.

3. PlayGame algorithm

a) Global variables and user-defined functions

In order to create an Easel$_{PY}$ application, EaselLib must be imported at the beginning of the game file. The following global variable names are imported from EaselLib.py.

- `mouseX` and `mouseY` are the horizontal and vertical position of the mouse in the window
- `mouseDown` is true iff the left mouse button is down
- `oldMouseDown` is False in the first frame, and in each subsequent frame is the previous value of `mouseDown`from the previous frame. It is set to False initially.
- `keysDown` stores the set of keys which are currently held down
- `oldKeysDown` is the previous value of keysDown

In order to create an Easel$_{PY}$ application, zero or more of the following must be defined.

1. init()-- a procedure that initializes the program state and loads any sound and/or image files. init() must declare as global all the variables it sets
2. update() -- a procedure that updates the game state variables. update() must declare as global the variables it changes.
3. display() -- takes no parameters and returns a sprite consisting of the images to be displayed in the current frame. Display may read the global variables from init and update.
4. frameRate() -- returns the frame rate on frames per second. If this function is not defined it defaults to 20 frames per second.
5. windowDimensions() -- returns the window dimensions as a pair (width, height), which is the size of the game window. If this is not defined it defaults to (800,600).
6. insert playSound(s) instructions where desired, where s is a sound.
7. sounds() -- takes no parameters and returns a list of sounds to be played in the current frame.

b) PlayGame pseudocode

Given definitions above, the game engine runs as follows until interrupted (typically, by the user closing the game window or hitting the “esc” key on the keyboard)

State variables:

The global variables used in the algorithm are as follows :

```
mouseX: int, mouseY: int, mouseDown: Bool, oldMouseDown: Bool,
oldKeysDown: list<int>, keysDown: list<int>
```

Procedure:

- `If frameRate() is not defined, set the frame rate to 20; otherwise set it to the return value of frameRate()`

- If windowDimensions() is not defined, set the window dimensions to (800,600). otherwise set it to the return value of windowDimensions().
- if init() is defined, call it
- HALT := False
- mouseDown := False; keysDown := [ ]
- while not HALT
  - mouseX := horizontal position of the mouse in the window
  - mouseY := vertical position of the mouse in the window
  - oldMouseDown := mouseDown
  - oldKeysDown := keysDown
  - mouseDown := true iff the left mouse button is down
  - keysDown := the set of keys which are currently held down
  - if display() is defined,display all the images in the list returned by display()
  - if sounds() is defined, play the sounds in the list returned by sounds()
  - if update() is defined, call it to update the state of the game
  - sleep until next frame
  - set HALT to true if needed

c) Loading sound and image files

If an image is loaded from a file, then the init() procedure should contain an instruction of the form

    <variable name> = loadImageFile(<file name>)

and the variable name should be declared within init() as global. For example, if you want to load an image from the file dogBark.wav, and store it in the global variable DOGBARK, your init() procedure would declare DOGBARK as global and contain the instruction

```
DOGBARK = loadImageFile("dogBark.wav")
```

4. Using Easel_PY

A) Installation requirements

Before using EaselPY, The following must be installed:

1. python 3.x. Please refer to following link to download and install python https://www.python.org/downloads/

2. pygame 1.9.x. For Windows, please refer to the following link to download and install the corresponding pygame version of your installed python http://pygame.org/download.shtml. You can download either http://pygame.org/ftp/pygame-1.9.1.win32-py3.1.msi or http://pygame.org/ftp/pygame-1.9.2a0.win32-py3.2.msi to install pygame for python 3.x on your 32 bit or 64 bit windows machine. For Mac users, please refer to the following link to install pygame for python 3.x: http://dudeslife.com/blog/2014/programming/installing-python-3-3-3-pygame-on-os-x-mavericks/
3. Easel$_{PY}$. The code for the game engine can be downloaded at https://github.com/qianji/Easel-Game-Engine/archive/master.zip . There is no "install" to do; just download and extract the files. This zip includes the following files
    a. Easel.py (code for the game engine)
    b. EaselLib.py (code for the game engine)
    c. boxClick.py (a super-simple sample game)
    d. tutorial.py (a sample game)

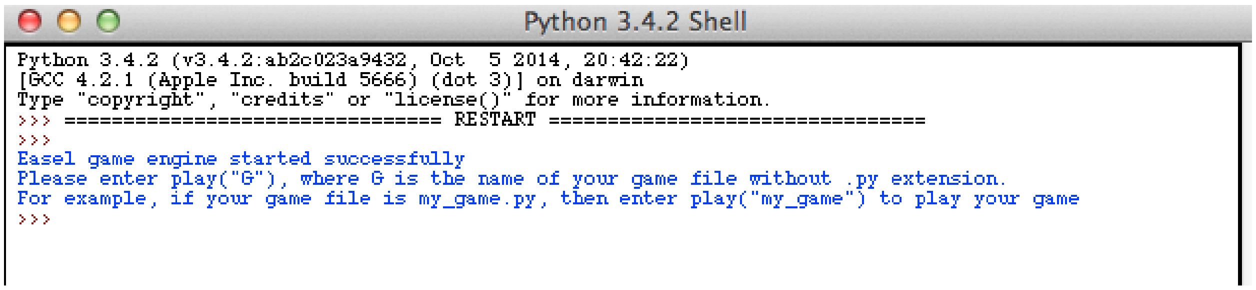
b) Running the game

Before running the game make sure that the following requirements are meet.

1. Python 3.x is installed.
2. pygame 1.9.x for your installed python is installed.
3. Easel$_{PY}$ is downloaded.
4. EaselLib is imported in your game program.
5. Your game program is placed in the same folder as EaselLib.py

In order to run the game using Easel$_{PY}$, following instructions are recommended

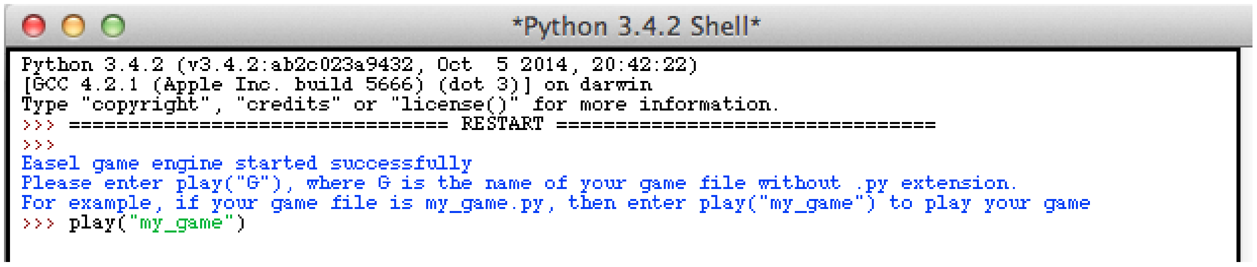
1. Open Easel.py in IDLE.
2. Click Run->Run Module in the menu of IDLE
3. After the game engine Easel.py module is run, you will see the following message in your python Shell



```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  5 2014, 20:42:22)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Easel game engine started successfully
Please enter play("G"), where G is the name of your game file without .py extension.
For example, if your game file is my_game.py, then enter play("my_game") to play your game
>>>
```

4. Enter play("G") in the Shell, where G is the name of your game file without .py extension. For example, if your game file is my_game.py, then enter play("my_game") to play your

game.



```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  5 2014, 20:42:22)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Easel game engine started successfully
Please enter play("G"), where G is the name of your game file without .py extension.
For example, if your game file is my_game.py, then enter play("my_game") to play your game
>>> play("my_game")
```

Appendix A: Key Tables

KeyASCII	ASCII	Common Name
K_BACKSPACE	\b	backspace
K_TAB	\t	tab
K_CLEAR		clear
K_RETURN	\r	return
K_PAUSE		pause
K_ESCAPE	^[	escape
K_SPACE		space
K_EXCLAIM	!	exclaim
K_QUOTEDBL	"	quotedbl
K_HASH	#	hash
K_DOLLAR	$	dollar
K_AMPERSAND	&	ampersand
K_QUOTE		quote
K_LEFTPAREN	(	left parenthesis
K_RIGHTPAREN	)	right parenthesis
K_ASTERISK	*	asterisk
K_PLUS	+	plus sign
K_COMMA	,	comma
K_MINUS	-	minus sign
K_PERIOD	.	period
K_SLASH	/	forward slash
K_0	0	0
K_1	1	1
K_2	2	2
K_3	3	3
K_4	4	4
K_5	5	5
K_6	6	6
K_7	7	7
K_8	8	8
K_9	9	9
K_COLON	:	colon
K_SEMICOLON	;	semicolon
K_LESS	<	less-than sign

K_EQUALS	=	equals sign
K_GREATER	>	greater-than sign
K_QUESTION	?	question mark
K_AT	@	at
K_LEFTBRACKET	[	left bracket
K_BACKSLASH	\	backslash
K_RIGHTBRACKET	]	right bracket
K_CARET	^	caret
K_UNDERSCORE	_	underscore
K_BACKQUOTE	`	grave
K_a	a	a
K_b	b	b
K_c	c	c
K_d	d	d
K_e	e	e
K_f	f	f
K_g	g	g
K_h	h	h
K_i	i	i
K_j	j	j
K_k	k	k
K_l	l	l
K_m	m	m
K_n	n	n
K_o	o	o
K_p	p	p
K_q	q	q
K_r	r	r
K_s	s	s
K_t	t	t
K_u	u	u
K_v	v	v
K_w	w	w
K_x	x	x
K_y	y	y
K_z	z	z
K_DELETE		delete
K_KP0		keypad 0
K_KP1		keypad 1
K_KP2		keypad 2
K_KP3		keypad 3
K_KP4		keypad 4
K_KP5		keypad 5
K_KP6		keypad 6
K_KP7		keypad 7
K_KP8		keypad 8
K_KP9		keypad 9
K_KP_PERIOD	.	keypad period

K_KP_DIVIDE	/	keypad divide
K_KP_MULTIPLY	*	keypad multiply
K_KP_MINUS	-	keypad minus
K_KP_PLUS	+	keypad plus
K_KP_ENTER	\r	keypad enter
K_KP_EQUALS	=	keypad equals
K_UP		up arrow
K_DOWN		down arrow
K_RIGHT		right arrow
K_LEFT		left arrow
K_INSERT		insert
K_HOME		home
K_END		end
K_PAGEUP		page up
K_PAGEDOWN		page down
K_F1		F1
K_F2		F2
K_F3		F3
K_F4		F4
K_F5		F5
K_F6		F6
K_F7		F7
K_F8		F8
K_F9		F9
K_F10		F10
K_F11		F11
K_F12		F12
K_F13		F13
K_F14		F14
K_F15		F15
K_NUMLOCK		numlock
K_CAPSLOCK		capslock
K_SCROLLOCK		scrollock
K_RSHIFT		right shift
K_LSHIFT		left shift
K_RCTRL		right ctrl
K_LCTRL		left ctrl
K_RALT		right alt
K_LALT		left alt
K_RMETA		right meta
K_LMETA		left meta
K_LSUPER		left windows key
K_RSUPER		right windows key
K_MODE		mode shift
K_HELP		help
K_PRINT		print screen
K_SYSREQ		sysrq
K_BREAK		break

K_MENU	menu
K_POWER	power
K_EURO	euro