PyGraphics

Facilitating Multimedia Graphics Programming in the Introductory Computer Science Curriculum.

Analysis & Estimation

Supervisor/Customer: Professor Paul Gries
Author: Leo Kaliazine
Version: 1.0
Date: June 12, 2007
Home: http://code.google.com/p/pygraphics/

Abstract

The aim of the Summer 2007 PyGraphics project is to prepare the project for use in the Fall 2007 first year Computer Science courses. The major facets of the project that still need to be implemented are a multi-platform installer, picture explorer tools and removing PyGraphic's dependency on Pygame and Tkinter python libraries.

Definitions

Jython	An implementation of Python written in Java, running atop the JVM
CPython	The standard Python implementation written in C
PyGame[8]	A python game library which supports images, sounds, and much more
PIL[9]	A python imaging library which provides a variety of image operations on a wide range of image file formats
Media.py (Jython)	The python multimedia library which makes use of multimedia capabilities in Java through Jython.
Unit Tests (Jython)	Unit tests for the multimedia libraries (currently dependent on Jython)
Media.py[4] **(CPython)**	A version of the multimedia library to CPython, with dependencies on PyGame, PIL and Numeric. It is currently incomplete.
DocString[10]	A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. Such a docstring becomes the `__doc__` special attribute of that object.[10]
DistUtils[11]	Distribution Utilities ('DistUtils') exists to make the distribution and installation of python modules, extensions and applications easier and standardized.

Existing and Non-Existing Functionality

PyGraphics provides a CPython, PyGame[5], PIL[6], and Numeric based multimedia library that allows manipulation of images, sounds and movies. This library is based on the one developed using Jython under the supervision of Mark Guzdial[2]. The features still not implemented are Media Explorer tools, multi-platform installer and the removal of the project's dependency on Pygame and Tkinter libraries.

Objectives

Task 1

Create a Google code[9] project website for the PyGraphics project, including the transferring of all data, wikipages and tickets currently located on the DrProject website.

This task has already been completed as part of the warm up.

Task 2

Implement the Media Explorer Tools, most importantly the Picture Explorer tool. This also covers tickets #14 and #15.

The Picture tool has been created as part of the warm up. It still needs to be integrated into media.py[4]. Also, support for forking in order to add the ability launch more than one instance of the tool simultaneously must still be added.

Tickets:

Ticket #14
Implement the image zooming capability. This ticket is part of Task 2.

Ticket #15
Implement the ability to view pixel information with the cursor. This ticket is part of Task 2.

Ticket #22
Create a platform-agnostic distutils[8] installer

All previous installers for media.py[4] were created as platform specific installers. Using python's distutils[8] will make installing and distributing the PyGraphics modules much easier and streamlined across Windows, Linux and OSX platforms.

Ticket #25
Switch to 0-based indexing. Some of the code currently implemented in media.py uses 1-based indexing.

Ticket #26
Add support for forking using multithread and exec. This is useful for running tools implemented with Tkinter such as openPictureTool.

Ticket #27
Add DocStrings[7] to media.py[4] functions.

There are currently almost no comments for the functions implemented in media.py[4] Since help() is a very useful tool for the first year students who will be using this library it is important to add at least general DocString[7] comments.

Ticket #28
Remove code implementing sound, turtle and movie from media.py. This is in accordance to moving away from depending on Pygame and Tkinter libraries. In preparation for the style and PIL switch, leave in only image-related code.

Ticket #29
Switch all method and variable names in media.py to pot-hole case.

Ticket #30
Remove Pygame dependencies. Figure out which top-level functions can be done only with PIL (and Tkinter dialogs), and make it happen.

Ticket #31
Redo existing unit tests of PyGraphics' image manipulation functions. Existing tests are to be modified to comply with dependency modifications. (Removal of Pygame, Tkinter)

Schedule

High	1, #29	#25,#28	#22,#30
Medium	#27	#31	----
Low	----	2, #14, #15,#26	----
Priority/Effort	**Low**	**Medium**	**High**

Task	Ticket #	Priority	Time estimate
Create DistUtils PyGraphics Installer	22	Very High	3 weeks
Remove Pygame dependency, including the removal of Sound, Movie and Turtle implementations.	28, 30	Very High	3 weeks
At same time, get rid of 1-based indexing.	25	High	3 days
Add DocStrings to media.py functions	27	High	2 days
Create Google Code PyGraphics webpage	Task 1	High	2 days
implement pot-hole case	29	High	1 day
Create and integrate Picture Tool	14, 15 (Task 2)	Medium	2 weeks
Testing reimplementation for image manipulation code.	31	Medium	1 week
Add support for forking	26	Low	1 week

References

1 Media Comp at Georgia Institute of Technology
http://coweb.cc.gatech.edu/mediaComp-plan

2 Mark Guzdial
http://www.cc.gatech.edu/gvu/people/Faculty/Mark.Guzdial.html,
guzdial@cc.gatech.edu;

3 Mark Guzdial, Introduction to Computing and Programming with Python. A Multimedia Approach
Prentice Hall; ISBN 978-0131176553 (December 27, 2004)

4 media.py (CPython) development page
http://coweb.cc.gatech.edu/mediaComp-plan/117

5 PyGame
http://www.pygame.org/

6 PIL
http://www.pythonware.com/products/pil/

7 DocString
http://www.python.org/dev/peps/pep-0257/#what-is-a-docstring

8 DistUtils
http://www.python.org/community/sigs/current/distutils-sig/

9 Google code
http://code.google.com/