University of Puerto Rico
Mayagüez Campus

Final Report
Picture Animation Language
PAL

Jonathan Santiago González
Adahid Galan Rivera
Jesiely Martínez Rodríguez
Gilson Rivera González
ICOM4036/CIIC4030
Prof. Wilson Rivera
May 29, 2018

Introduction

During the technology era, we have seen a new way to see the media. From drawings digitized, movies, series, clips, live streaming, animations, etc. We as a group thought that a good way to continue those trends came in one idea: Programing Animation Language (PAL). This programming language is intended to create animations and then export them as a GIF file. Our goal is to create an easy to use tool for users to bring ideas to life and explore their imagination. It is a new way to express your creativity with something simple and fun. One of the main purposes of our project is to improve the world of short animations, making it easier and accessible to create, so that even children can use. This is done by simplifying the process, not requiring advanced skills in programming and graphic design.

Language tutorial

Language reference manual

Below are the functions and commands used in Picture Animation Language:

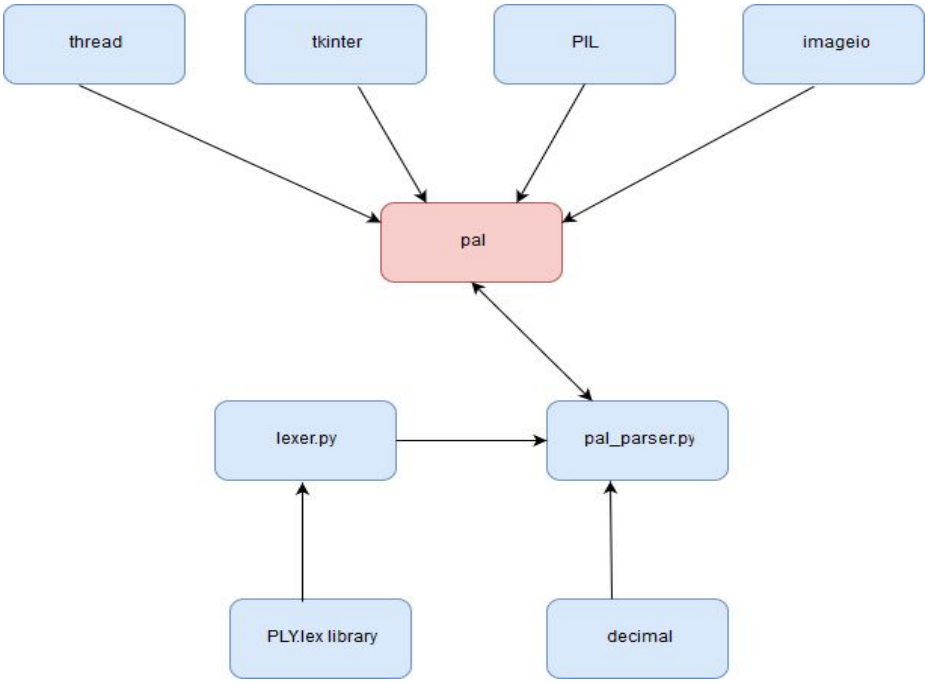
- **init(width, height) -** It initialize the animation with the given width and height passed as parameters.
- **show -** It displays a preview of the current frame with all its content.
- **createFrame -** Creates a frame that looks exactly the same as the one already created, just to facilitate the flow and development of the animation.
- **changeFrame(index) -** Change the frame to the one specified by the index given as a parameter.
- **moveBackground(deltaX) -** Moves the background according to the number passed as a parameter.
- **setBackground(fileName) -** Sets the background of the frame to the image passed as a parameter.
- **createSprite(spriteName, fileName, width, height) -** Creates a sprite with the given name and dimensions for the sprite states. The name will be used to call all the other sprite functions to modify this sprite.
- **spriteName moveSprite(Mode, moveX, moveY) -** Moves the sprite. Use “R” in the Mode parameter to move it to its relative position, otherwise, use “A” to move it absolute.
- **spriteName resizeSprite multiplier(factor)-** Resizes the sprite by using a factor in which all the states of the sprite resize as well.
- **spriteName rotateSprite relative(angle)-** Rotate the sprite by adding or substracting the angle with the sprite’s angle.

- **spriteName rotateSprite absolute(angle)-** Rotate the sprite by giving the specific angle to be rotated.
- **createAsset(assetName, assetImage.png) -** Creates an image to add to the frame with the given name. The name will be used to call all the other asset functions to modify this asset.
- **assetName moveAsset(Mode, moveX, moveY) -** Moves the asset. Use “R” in the Mode parameter to move it to its relative position, otherwise, use “A” to move it absolute.
- **assetName resizeAsset absolute(width, height) -** Resizes the given asset with the specified dimensions.
- **assetName resizeAsset multiplier(factor)-** Resizes the asset with a factor to be multiplied by the current asset’s dimensions.
- **assetName rotateAsset relative(angle)-** Rotate the asset by adding or substracting the angle with the asset’s angle.
- **assetName rotateAsset absolute(angle)-** Rotate the asset by giving the specific angle to be rotated.
- **createAnimation(displayTime) -** Creates an animation with each of the frames in the order they are created. Each frame will be showed by the time passed as a parameter which are in seconds.

List of reserved words:

- init
- createframe
- show
- test
- changeframe
- setbackground
- movebackground
- createasset
- moveasset
- reziseasset
- rotateasset
- removeasset
- createsprite
- changespritestate
- movesprite
- resizesprite
- rotatesprite
- removesprite
- createanimation
- multiplier
- relative
- absolu

Translator Architecture:



PAL uses the modules lexer.py and PAL_parser.py. PAL_parser is the main module that runs the application. It contains the main methods of the application. User input is parsed by yacc using the PLY library. These functions define grammar characteristics of the language. The yacc parser must match the functions found in PAL_parser module with the command entered by the user, considering the tokens defined in the lexer file for PAL. The tokens form the regular expression that will be matched and execute the command.

PAL uses the following libraries:

Imageio:

Imageio is a Python library that provides an easy interface to read and write a wide range of image data, including animated images, video, volumetric data, and scientific formats. It is cross-platform,

runs on Python 2.7 and 3.4+, and is easy to install. We will be using this library for exporting the finished animation.

PLY:

Also referred as Python Lex- Yacc, is a parsing tool written in Python and implements the Lex parsing tool. This tool s divided in both stages, the lexical analyzer(lex) is the one responsible that all the syntax what the user writes follows the rules specified by the Developers. The second one is yac, practically yac is the one that parses de code.

Tkinter:

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package, used in pal module to provide the graphics needed.

PIL:

The Python Imaging Library by Fredrik Lundh and Contributors.

_Thread:

The thread module provides a basic synchronization data structure called a *lock object* The synchronization primitives go hand in hand with thread management.

The software development environment used to develop PAL are detailed next:

Python:

An interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development. We prefer to use this programming language because it’s easy to implement, it’s one of the most used language in the world and the documentation and community is growing fast. Also there are many libraries that can help developers take a starting point and continue creating new projects instead of re-creating the wheel.

PyCharm:

Integrated Development Environment (IDE) used to develop applications and programming projects, using Python as the main programming language. We as a team choose this IDE and Python because is one of the most popular programming and have a great feedback in the developer community.

Test Methodology:

For the testing process of the language, we use the method show(p). Any change we made to the frame, we test with the show method. If the change was successful, the method executed successfully, failed otherwise, sending an error. Pycharm was the main engine used to test the program.

The testing phase lasted about two weeks, doing several test rounds to make sure the language executes as expected.

Conclusions

The main goal of Picture Animation Language is to reduce the complexity and the time it takes to create a cool animation. Since the beginning, we really wanted to create something simple that does not require previous knowledge of programming so that even children can begin to learn to program while having fun and express their creativity and imagination. After finishing the project, we are extremely proud of having accomplished all of our goals. We hope this project can inspire others to continue improving the World with software and encourage children's creativity to express themselves while acquiring new knowledge in the world of computers.