



# **IMAGI- Child Friendly Programming Language**

---

*Project Requirements and Main Features*

Prepared by:

Héctor García (hector.garcia@upr.edu)

Edgardo Rivera (edgardo.rivera@upr.edu)

Jariel Laureano (jariel.laureano@upr.edu)

# Introduction

---

## Project Motivation

With the technology industry developing at a rapid pace, the need for more programmers increases everyday. Little options exist to introduce programming to kids successfully, this is why our team decided to attack this need. By creating a new programming language focused and designed for children we will be able to encourage and motivate students at a young age to pursue STEM related careers but specifically, computer and software engineering majors. After experiencing IMAGI, kids will be able to understand the basics of programming, learn to have fun with it and be prepared to move to a more standard programming language like python or java in the future.

## Features

---

### Project Concept

IMAGI is a child friendly programming language developed with the goal of introducing programming to kids in a new and interesting way from a very young age. Lines of code alone will surely have a hard time arousing any children at such a young age, this is the reason why IMAGI's programming experience will be aided by a simple graphical user interface which will allow the programmer to interact with different characters and scenes, and create simple storylines by writing code. One of the challenges of kids as our scope is that this language must try to develop the programming instincts in the simplest way possible, avoiding complicated syntax, overcrowding of commands and ambiguity. Using commonly known words and making the programming language straightforward, as easy as talking, will be the focus. By doing this we will make sure the children understand the commands without putting too much thought into it, keeping them comfortable and confident while programming.

## Example Program

In general, the code will follow the convention showed bellow, taking into account indentation.

```
1 Character command
2   attribute-1
3   attribute-2
4   attribute-N
```

This convention will be used to interact with both characters and scenes. Apart from the simple commands for interaction, there is also a repeat command which will act as a simple loop and only two data types word(string) and number(integer). All this will be demonstrated below.

```
1 fish say
2   "Hello"
3 fish move
4   left
5 fish say
6   "I'm going to sing."
7 fish sing
8 word name
9 fish ask
10  "What's your name?"
11  name
12 fish say
13  "hello" + name
14 repeat 5
15   fish jump
16   fish say
17     "I'm happy"
18 number a_number=5
19 fish domath
20   +
21   5,6,a_number
22
```

Character Name

Move Command

Move Command Attribute

Word type variable and variable name

Repeat command with repeat times attribute

Commands to repeat

When running the program all these commands will be reflected on your graphical scene which should include a simple character and a background. The character will react based on the commands issued, it will jump, move, prompt messages and much more.

<b>Commands/Tokens</b>	<b>Target</b>	<b>Number of Attributes</b>	<b>Attributes Description</b>
<b>Jump</b>	Character	N/A	No attributes needed.
<b>Walk</b>	Character	1	Direction: left, right, forward or backward
<b>Say</b>	Character	1	Text to say: raw string or word type
<b>Domath</b>	Character	2	Operator: add, subtract, multiply or divide Number/s to add: can be a raw number or a number type described below.
<b>Turn</b>	Character	1	Direction: right or left
<b>Domultiple</b>	Character	1	Character name
<b>Dance</b>	Character	N/A	No attributes needed.
<b>Sing</b>	Character	N/A	No attributes needed.
<b>Grow</b>	Character	N/A	No attributes needed.
<b>Shrink</b>	Character	N/A	No attributes needed.
<b>Ask</b>	Character	2	Text to prompt: raw string or word type Variable to store input: word or number type
<b>Repeat</b>	Character	1	Times to repeat
<b>Flip</b>	Character	N/A	No attributes needed.
<b>Run</b>	Character	1	Direction: right or left
<b>Time</b>	Scene	1	Scene time: day or night
<b>Word</b>	N/A	N/A	Word type that will store strings.
<b>Number</b>	N/A	N/A	Number type that will store integers.

## Implementation

---

### Requirements and Tools

The tools and requirements needed to be able to use our programming language are kept to the minimum possible so that it is easy to use in any of the most popular operating systems. It is required that the user has Python version 2.7.11 and PyQt4 installed to make sure the language can be run from the command prompt. The use of an IDE or command prompt directly will be avoided by providing an executable file.

## Project Plan

Task	Description	Start Date	End Date
Programming Language Concept and Description	The team will get together to develop a proposal for a programming language to develop.	<b>Feb. 1</b>	<b>Feb. 10</b>
GUI Design and Development	The team will decide on a design for the GUI to aid our programming language and work on the code development.	<b>Feb. 10</b>	<b>Feb. 15</b>
Graphic Design and Animations	The team will design the different scenes and characters for the programming language and will work on the code development of the animations.	<b>Feb. 15</b>	<b>Feb. 22</b>
Scanner Development	The team will decide on a syntax for the programming language and use one of the provided online tools to generate a scanner.	<b>Feb. 22</b>	<b>Feb. 26</b>
Parser Development	The team will work on the development and testing of the parses for our programming language.	<b>Feb. 26</b>	<b>Mar. 7</b>
GUI, Scanner and Parser Integration	The team will integrate all the previous phases on a complete application that will make the GUI work along with the scanner, parser and graphic animations.	<b>Mar. 7</b>	<b>Mar. 24</b>
Final Report and Video Tutorial	The team will work on a video tutorial on how the programming language works and write the final report and documentation needed.	<b>Mar. 24</b>	<b>Mar. 31</b>
Web Page Development	The team will work on the development of a simple webpage to showcase the programming language, give support and make the source code available.	<b>Mar. 31</b>	<b>Abr. 7</b>

# Timeline

