

PLT Project Proposal

Spring 2016

Roles

Alice Hwang (ah2813) - Manager

Julian Edwards (jse2122) - Language Guru

Bryan Yu (by2181)- Systems Architect

Laura Hu (lh2718) - Tester

The Language

Our team has decided to implement GridLok, an object-oriented programming language for creating basic grid-based (2D) games for those new to coding.

Essentially, the language will be a good introduction to the coding world and provide easily accessible functions for creating simple games such as tic-tac-toe, battleship, or even chess. GridLok will have a simple, straightforward syntax, similar to a mix of Python and Java, in order for new coders to have a smoother transition from writing in a non-computing language to a computing language. By adopting a structure that is similar to that of English, we hope that beginning programmers will be able to focus more on the semantics rather than being limited by the syntax. In addition, coders will not have to deal with maintaining memory, pointers, or garbage collection as the language's compiler will take care of that for them. Our language will be compiled into Python.

Parts of the Language

Control flow:

- If/Else statements
- While loops
- For Loops

Comments

- Multi Line

User Defined Functions

- `func functionname(parameters):`
 <function code here>

Operators

- Parenthesis
 - () dictate order of operation
- Mathematical operations will be left associative, and mathematical operations will also follow the usual order of operations. For example, $a + b + c$ will be interpreted as $(a + b) + c$, and $a + b * c$ will be interpreted as $a + (b * c)$

Mathematical Operator	Meaning
=	Assignment
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulo

● Comparison

Comparison Operator	Meaning
==	Equivalence
!=	Does not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

● Logical Operators:

Logical Operator	Example
NOT	if NOT isFull;
AND	if isFull AND isSleepy;
OR	if isFull OR isSleepy;

Classes specific to our language/in the standard library:

● Grid

- Holds a double matrix of type Actor representing the board for the game
- Contains functions to check if a board space contains an actor, what actors are left on the board, etc.

● Actor

- A class that contains String owner, Grid board, int row, int col, and boolean moveable
- Contains various functions to place or move an Actor on a certain Grid, change owners, swap boards, etc.

Code Snippet:

```
/* Hardcoded data ensures wolf wins this game */
/* game is similar to main(), necessary for an executable */
game:
    String p1 = "player1";
    String p2 = "player2";

    /* Creates a 3 by 3 matrix and the two "game pieces" */
    create Grid field(3, 3);
    create Actor sheep(p1, True);
    create Actor wolf(p2, True);

    /*Places the game pieces on the board called "field" */
    sheep.place(field, 1, 1);
    wolf.place(field, 1, 2);

    /*Wolf moves to eat sheep*/
    if field.hasActor(1, 1) == False:
        sheep.remove();
        wolf.move(1, 1);
        print "Wolf ate sheep. " + p2 + " wins.";

    /* exits game function */
    gameover;
```