University of Puerto Rico - Mayagüez Campus
Department of Electrical and Computer Engineering

Card ++
Final Report

Eliemer Vélez
Jonathan Irizarry
Ariel Torres

Introduction

A card game is any game that uses cards as the primary device with which the game is played. Card games have always been a big part of human history. As such, we want to give people a tool to be able to continue this tradition via computer programs.

As part of our Structure and Properties of Programming Languages course we have been tasked with the creation of a new programming language which, in our case, would result in being able to create card games in a short amount of time, with little experience in programming required.

Language Details

We want Card++ to be as high level as possible, readability is a high priority as we want anyone to be able to pick up our language and start creating. A key feature of our language is that of definition blocks. These blocks contain clear workspaces in which to work different aspects of the game. For example, at the start of your code, you declare a Deck{...} block in which you list all available cards in the entirety of your game. Poker for example, contains fifty-two casino suite cards, thus each item in the Deck{...} block would contain a number (from Ace to King) and a suite (hearts, clubs, spades, diamonds). These blocks contain different aspects of your game code, and are segregated for better readability.

Another key aspect of our language is that every action revolves around three things: player resources (money, energy, etc.), turn control (skip turns, reverse turn order), or fields. The latter is a critical part of our vision. A field is essentially a container for cards. These fields may have special properties to them, such as the Deck field being invisible to every player, that is to say, no one can view the values of the cards within. Other common fields include a player’s hand, the playing field, and card graveyards. Common actions that manipulate fields include: drawing cards (move 1 From Deck to Player) and playing cards (move 1 From Player to Game). Actions that manipulate player resources may include betting, or actions be conditioned to a value, say a mana bar. Thus, from this framework of actions we can implement anything we’d like the player to do.

Simple Keywords

- Card: A tuple representing an instance of a game card
- Draw: Adds an instance of a game card to a Field
- Move: Adds a particular instance of a game card to a Field
- Flip: Changes a card's visibility to certain Players
- Shuffle: Mixes all cards' locations in a given Field
- Rules: Boolean statements describing a particular scenario or condition in the game

Tools Used

PLY - PLY (Python Lex-Yacc) is an implementation of lex and yacc parsing tools for Python, developed by David Beazley. This tool is used to establish and interpret our language's grammar.

Language Tutorial

Installation

1. Download Card++ from our GitHub repository at https://github.com/Eliemer/Card-plus-plus .
2. Make sure that you have Python 3 (3.6) installed in your machine with the PLY (3.10) library.
3. Run the *Card++.py* file on your IDE or terminal.
4. Start using Card++.

Using Card++

Card++ allows you to create your card game using the fifty-two casino suit cards. A card declaration in Card++ is simple. The following is the syntax used to declare a card:

Card *cardname* = ("*value*","*suit*")

For example, if we want to declare the three of diamonds, we would type the following in Card++:

Card Diamonds3 = ("3", "Diamonds");

A valid card declaration receives a valid value and suit as parameters, though we can declare the name however we want.

After you declare the desired number of cards, a field can be created to represent the different types of fields in a card game. The following is the syntax used to declare a field:

Field *fieldname* [*Card 0, Card 1, Card 2, ... , Card N-1*];

We could declare a Field composed of only the three of diamonds:

Field Deck [Diamonds3];

It is also valid to declare an empty field, for example an empty hand would be declared as follows:

Field Hand [];

After the hands and fields are declared, it is valid to operate on them using Card++'s functions.

Some functions are:

Shuffle(Deck); #Shuffles the deck

flip (Diamonds3); #Flips the card

Draw(Deck); #Draws a card from the deck

Conclusion

The development of Card++ has given us the opportunity to truly see the complexity behind a programming language. The completed language implementation gives users the opportunity to develop their own card games in a simple, high-level environment. Card++'s simple keywords allow the readability of the code to be superior to any implementation using another programming language. The goal of this project is to allow card game creators to innovate by programming their own card games in a simple way. Card++ achieves this goal by providing the users with a simple interface that does not require much knowledge outside of basic programming.