Creation of a Pong Game Through Visual Basic.NET

Antony Gardner

May 27, 2014

Abstract

This is the trials of creating a simple Pong game in six days time. there were severak errors along the way, and the game itself was never actually completed, but it was a fantastic learning experience that will prepare me for future programming courses.

1 Introduction

I started this project thinking that it would be a rather easy way of getting the project dome while still having fun with it. But I soon found out that programming is much more complex that I had originaly remembered it to be.

2 The Tribulation of Finding a Programing Language

2.1 JavaScript

My first attempts at creating this game were done with JavaScript programming. I slowly incorporated OOP, VMC, and UML programming systems in my work while going through a Java lesson at codecadamy.com. I found this very helpful, but when I tried to apply these concepts on my own, I ran into difficulties.

During all this time, I was attempting to use an Apple Ipad do complete the project, unaware that it does not actually support JavaScript debugging. Try as I might, there was no application or pair of Apps that would allow me to easlily write the code AND debug it. It was clear that I needed to find an easier way to complete the assignment.

2.2 Reunion with VB.NET

During my Freshman year, I took a very basic web design course with Mr. Reblin. I was learning very basic HTML, CSS, and Java, but I loved it so much that I immediatly signed up for the remaining Computer Science classes that were avaliable. This introduced me to Visual Basic.Net. I spent the entirty of my Sophomore year programing in this language.

Needless to say, when my Java programming was turning out to be more stress than programming, I readily turned to the familiar for a solution. I got onto my hom computer that night and started coding.

3 The Code

3.1 First Attempt

My first attempt at creating the walls and paddle for this pong game was ... less than stellar. I origionally thought that it was simple enough to draw each shape, with certain variables set for the paddle in place of numbers, and then have a KeyDown Sub linked to to an update function that would show the movement of the paddle when the user pressed either the left or right arrow keys. This ended up being a problem, because in order to do that, you would have to call the variables used in the paint sub again in the update funtion. Since these variables are services and aspects of the system itself, the program would crash when the variable was called a second time in the same class.

3.2 The Learning Curve

The solution, as later discovered, was multiple classes. It was much easier to create the walls in one class, the ball in another class, and the paddle in a third class. Each could then have it's own paint fucntion, while still relying on a universal paint sub. I also learned that, through this method, many other aspects of the game can be controled. For example, when a brick is destroyed, the brick can run a checkForPowerUp fuction, and if there is a powerup there, it can transfer the data to the createPowerUp class to decide what kind of power up it is, and where it would fall from and what speed it is falling. Furthermore, if the powerup hits the paddle, it can determine the affects of the power up and if it, say, makes the paddle bigger, the paddle class can use the data to enlarge the paddle.

Lastly, I leared much more detailed and decorative ways to create my games. Each object could have properties that changed the visual data only during the game. A brick that takes more than one hit to destroy may have a special craking affect when numberOfHits is less that the breakingPoint of the brick. Parts of the wall may light up depending on where it is hit and the ferocity that it is hit with. The paddle may begin to shake when the ball hits it off-ballance. There are countless other examples that could be stated in place of these, all to be considered if there would be a final product.

4 Final Thoughts

This project, if anything, has taught me that it takes consitency to be not just good, but efficent in your programming. You are always finding better ways to do things. Would it be better to use and if ... then statment or a select ... case form? What about for loops? Is there a specific exception to be noted? All steps taken in the condesation and easy execution of the code. Not only is it less haphazard, it is easier to find when and where you made a mistake. Furthermore, it has given me enlightment as to what the next four years will be like at Champlain. I fell better prepared to tackle the more complex aspects of programming in the future.