**Barret J. Nobel**

**March 18th , 2016**

**Hullabalu Function**

**Hex to Float Converter**

**Problem Statement:**

Given a color hex-code, return an array of floats representing each of the three primary color groups – Red, Green and Blue.

**Analysis:**

The function that takes an integer color hex-code, i.e. 0x123456, as its sole input and returns an array of floats of each color component - red, green, and blue. The three color values, red, green, and blue will be normalized between 0.0 and 1.0.

**2.1 Table of Data Description**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Data Name** | **Identifier** | **Data Type** | **Value** | **Value Obtained** | **Definition** |
| **H** | **hexInput** | **NSString** |  | **UITextField** | **A six-digit string representing a color code in hex** |
| **A** | **normalizedArray** | **NSMutableArray** |  | **Algorithm I** | **An array of normalized floating point numbers representing the converted hex input** |
| **B** | **convertButton** | **UIButton** |  |  | **A button the user will press to convert the hex value in the UITextField and display the output to the UILabel** |
| **Lg** | **greenLabel** | **UILabel** |  | **UIButton touch event** | **A label to display the green converted hex value** |
| **Lr** | **redLabel** | **UILabel** |  | **UIButton touch event** | **A label to display the red converted hex value** |
| **Lb** | **blueLabel** | **UILabel** |  | **UIButton touch event** | **A label to display the blue converted hex value** |
| **T** | **hexConvertTextField** | **UITextField** |  | **User input** | **A UITextField for the user to input the hex value they wish to convert** |

**2.2 Input**

Before populating the array with normalized floating point values, we must first obtain the input to convert from the user. This value is obtained from a hexConvertTextField UITextField, and triggered when the user presses the convertButton UIButton.

**2.3 Output**

Screen output will be to all three UILabel’s located on the screen. It will also output to the standard console window, displaying the normalizedArray values.

**2.4 Methods**

All UILabel’s text values are set when the user clicks the convertButton. The converButton UIButton’s touch event implements Algorithm I to populate the normalizedArray.

**2.5 Test Data 1.**

**Hex input**

T = FFFFFF

H = FFFFFF

**Normalized Array**

A = [ 1.0, 1.0, 1.0 ]

**Labels**

Lr = 1.00 **|** Lg = 1.00 **|** Lb = 1.00

**2.5 Test Data 2.**

**Hex input**

T = 000000

H = 000000

**Normalized Array**

A = [ 0.0, 0.0, 0.0 ]

**Labels**

Lr = 0.00 **|** Lg = 0.00 **|** Lb = 0.00

**2.5 Test Data 3.**

**Hex input**

T = 33BAE7

H = 33BAE7

**Normalized Array**

A = [ 0.2, 0.73, 0.91 ]

**Labels**

Lr = 0.20 **|** Lg = 0.73 **|** Lb = 0.91

**Design:**

The program will have a single ViewController with the necessary .h and .m files. Allowing a user to input a hex-code in, and receive the normalized RGB conversion as output.

It includes the following private methods:

Class - ConvertViewController:

Method - hexConvert – converts a two-value string to a decimal number

Method – normalizeThisNumber – converts a decimal number to a normalized floating point number

Method – isValidHex – checks to see if the string contains valid hex value

**Algorithm I:** Loops through the hexInput string recursively and populate normalizedArray

Get hexInput from hexConvertTextField

if hexInput length is valid and all characters are valid hex numbers

pass hexInput to hexConvert

If hexInput is empty

break out

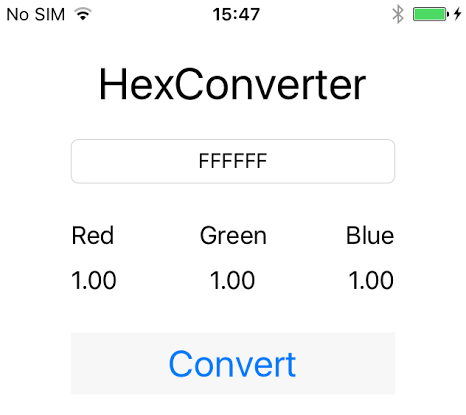
convert first two characters to decimal

pass converted decimal number to normalizeThisNumber

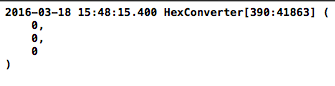
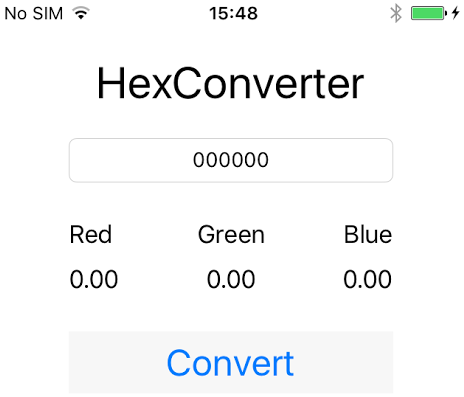
populate normalizedArray with value returned from normalizeThisNumber

recursively pass substring of hexInput to hexConvert

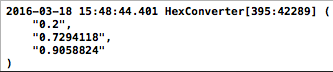
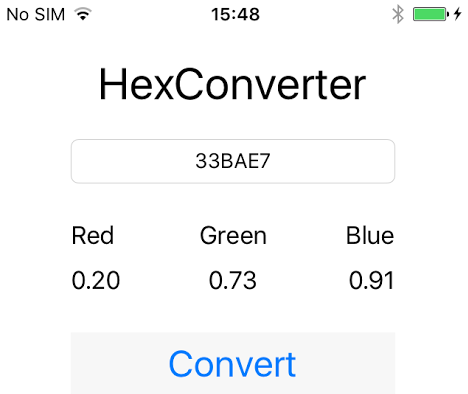
**Tests:  
Test Data 1:**



**Test Data 2:**



**Test Data 3:**



**Test the program – Results**

Results are accurate and expected

**Instructions**

The program HexConverter was created with deployment target of iOS 9.0 written in Objective-C using Xcode 7.2.1 by Barret J. Nobel.

To run the program press: command + R

OS Requirements: OSX 10+