Mobile Application Programming: iOS

CS4962 Spring 2015
Project 1 - Color Chooser
Due: 11:59PM Monday Feb 10

Abstract

Create a fully-functional user interface element in Swift that allows the selection of a color. The color chooser can use any method to do color selection that satisfies the components section of this document. It must include at least one custom control that draws itself using the CoreGraphics API. It should should also try to be visually pleasing. Put some thought into its design. If you're not feeling particularly creative, building the example shown on the following page will give 90%. Improve its visual appeal in some way (gradients, backgrounds, better borders, shadows, glass sheen, etc.) to get the last 10%.

Components

- Main composite view inherits from UIView or UIControl and includes a property for:
  - Color
  - Previous Color
- Includes an element that allows side-by-side comparison between the currently selected color and the color passed into the Previous Color setter.
- Allows selection of different "hues" (different colors like red, green, yellow, etc.)
- Allows selection of different "saturations" (for red, from pure red up to saturated white)
- Allows selection of different "brightnesses" (for red, near black to full bright red)
- Allows selection of a range of "alpha values" (completely transparent to opaque)
- These don't need to be selected by 4 different controls. Just that different combinations of control values allow a wide range of resultant colors in these 4 axes. Not all possible colors need to be reached, but many locations spread within this 4D space should be available.

Considerations

- Your color chooser will be much easier to build if you break it into several sub-views. Those sub-views should be subclasses of UIView or UIControl that use the target-action mechanism or delegation to post "value changed" notifications to registered listener objects. When the main color chooser view creates its sub-views, it sets itself up to receive the messages from them, then forward appropriate messages to the registered receiver for the main view.
- UI elements used in this assignment must be subclasses of UIView, UIImageView, UIControl, or UIButton. Do not use any other UIKit elements (UISlider, etc.). The idea of this restriction is to give you the need to use the CoreGraphics API to make your own controls. If you have an idea that needs to use other UIKit elements, request permission from the TA.
- Your color chooser must support portrait orientations on an iPhone 4, iPhone 5, and iPad 2 device. Ideally, your chooser should support all sizings from 200x200 to 1000x1000 points.
- 5% extra credit will be given to anyone using a knob-like selection mechanism. The knob must include color "labels" around the outside. These labels can be words, colored dots, or preferably, a circle or arc containing a continuous gradient of colors. If the selection is

continuous, the knob should move continuously. If the selection is discrete (e.g. using labels “red”, “green”, “yellow”, etc.) the knob must snap-to the nearest available selection location.
- 5% more will be given to anyone with a physical-analog themed color chooser. Think of a 70’s amp knob-based color selector, steampunk-style brass-and-gear color selector, Star Trek color selector, cave-man rock-and-twig color selector, or contemporary brushed-steel and glass color selector. Look around Lowe’s or Home Depot for contemporary examples.

Handin

You should hand in a zip file containing your project folder, including any image resources (please check that they are in the folder). Hand this zip into:

handin cs4962 project1 your_zip_file.zip