How to Use Hex Color Codes in Objective C with UIColor

TechRounder PDF Edition

Live article:
https://www.techrounder.com/ios/add-color-to-label-button-background-using-hexa-code-in-ios-application/

By Vipin PG | Published November 14, 2013 | Updated March 14, 2026 | Format: Guide | 3 min read

Quick answer

When a designer hands you a color like #FFFFFF or #000000, manually typing out RGB values in Objective-C gets tedious fast. UIKit's UIColor API expects red, green, blue, and alpha components - so the practical workaround is a small helper method that converts a hex string directly into a UIColor object.

When a designer hands you a color like #FFFFFF or #000000, manually typing out RGB values in Objective-C gets tedious fast. UIKit's UIColor API expects red, green, blue, and alpha components - so the practical workaround is a small helper method that converts a hex string directly into a 'UIColor' object. Web-style color codes in your iOS app, no repeated RGB math required.

How to Use Hex Color Codes in Objective-C

The cleanest approach is to drop a reusable method into your ViewController.m file. Once it's there, you can call it from anywhere in that class - backgrounds, labels, text fields, buttons, you name it.

Add This Function in ViewController.m

```
-(UIColor*)colorWithHexString:(NSString*)hex
{
NSString *cString = [[hex stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceAndNewlineCharacterSet]] uppercaseString];
// String should be 6 or 8 characters
if ([cString length] < 6) return [UIColor grayColor];
// strip 0X if it appears
if ([cString hasPrefix:@"0X"]) cString = [cString substringFromIndex:2];
if ([cString length] != 6) return [UIColor grayColor];
// Separate into r, g, b substrings
NSRange range;
range.location = 0;
range.length = 2;
NSString *rString = [cString substringWithRange:range];
range.location = 2;
NSString *gString = [cString substringWithRange:range];
range.location = 4;
NSString *bString = [cString substringWithRange:range];
// Scan values
unsigned int r, g, b;
```

```
[[NSScanner scannerWithString:rString] scanHexInt:&r];
[[NSScanner scannerWithString:gString] scanHexInt:&g];
[[NSScanner scannerWithString:bString] scanHexInt:&b];
return [UIColor colorWithRed:((float) r / 255.0f)
green:((float) g / 255.0f)
blue:((float) b / 255.0f)
alpha:1.0f];
}
```

What this does is read a six-digit hex string, split it into its red, green, and blue components, then hand back a properly formatted ‘UIColor‘. That lines up exactly with how Apple expects colors to be constructed in UIKit.

Call the Function with a Hex Code

With the helper in place, calling it is as simple as this:

```
[self colorWithHexString:@"FFFFFF"];
```

That produces a white color from the hex value. Swap out ‘FFFFFF‘ for any other six-digit code you need.

Set the Background Color of the View

To paint the background of your current view, drop this line inside ‘viewDidLoad‘:

```
self.view.backgroundColor = [self colorWithHexString:@"000000"];
```

That gives you a black background. UIKit exposes a ‘backgroundColor‘ property on view objects specifically for this kind of styling - so this is the right place to handle screen-level color changes.

Change the Background Color of a Button

Styling a button works the same way:

```
[your_button setBackgroundColor:[self colorWithHexString:@"FFFFFF"]];
```

The button background is now white.

Change the Text Color of a Button

To update the title color on a button, use this:

```
[your_button setTitleColor:[self colorWithHexString:@"FFFFFF"] forState:UIControlStateNormal];
```

That sets the button text to white for the normal control state. Apple’s UIKit button model uses state-based title colors, which is why ‘setTitleColor:forState:‘ is what you want here - not trying to reach into the label directly.

Apply Hex Colors to Labels and Text Fields

And here’s where things really start to pay off. Once that helper method exists, you don’t need separate color logic scattered across your code. You just call the same function everywhere.

For a label text color:

```
yourLabel.textColor = [self colorWithHexString:@"333333"];
```

For a text field text color:

```
yourTextField.textColor = [self colorWithHexString:@"222222"];
```

For a text field background color:

```
yourTextField.backgroundColor = [self colorWithHexString:@"F5F5F5"];
```

For a label background color:

```
yourLabel.backgroundColor = [self colorWithHexString:@"FFF8E1"];
```

A Practical Tip Before You Paste Colors Everywhere

Be consistent with your hex values. If your app uses the same brand blue, dark text, muted gray, and button accent across multiple screens, define those colors once and reuse them. In my experience, that saves far more cleanup time than most people expect. Scattering random inline colors throughout a project makes things messy in a hurry.

One more thing worth knowing: this helper expects a clean six-character hex string like ‘FFFFFF‘ or ‘000000‘. If you’re copying colors with a leading ‘#‘, you’ll need to strip that character first - or extend the method to handle it automatically. That small detail catches a lot of people off guard the first time.

Final Code Pattern

If your goal is simple, the whole workflow comes down to three steps:

- Add the ‘colorWithHexString:‘ method to ViewController.m
- Call it anywhere you need a ‘UIColor‘
- Assign the returned color to ‘backgroundColor‘ , ‘textColor‘ , or button title color

That’s really all there is to it. Once this helper is sitting in your project, working with hex colors in Objective-C stops feeling like a workaround and just becomes part of the routine.