CSSApply

Cascading Style Sheets for UIView/NSView/CALayer

(it's amazing)

The Team

- Zac Bowling - @zbowling
- Sam Stewart - @programpro
- Jonathan Dalrymple - @veritech
- Julie Silverman - @jssqrd

What is CSS?

- CSS separates the design and presentation semantics from the content by moving it into a separate file.
- Popular standard on the web since 1996.

```
body {
    margin: 4px;
    border: 3px dotted #
    font-family: sans-serif;
    color: #000000;
    background-color: #FFFFFF;
}

h1 {
    padding: 5px;
    margin: 10px;
    border: 1px solid #C0C0C0;
    color:#FF0000;
    background-color:#0000FF;
}
```

CSS

Why use CSS for UIKit/AppKit/ Core Animation?

- CSS fits the UIView/NSView/CALayer hierarchy really well.
- CSS is not really tied down to HTML. Other toolkits (such as QT and XUL) already support using CSS to style their UI frameworks.
- Designers and developers are both familiar with it and understand it.

Easy to use!

(Overly simplistic 2 line example!)

```
- (void)viewDidLoad
{
  [super viewDidLoad];
  NSURL *cssURL = [[[NSBundle mainBundle] bundleURL]
                   URLByAppendingPathComponent:@"mainview.css"]];
  CSSStyleSheet *styleSheet = [CSSStyleSheet styleSheetFromURL:cssURL];
  [self.view applyCSS:styleSheet];
}
```

Apply CSS IDs and Class to any UIView/NSView/CALayer (via a category)

```
UIButton *coolButton = ...;

coolButton.cssID = @"CoolButton";
coolButton.cssClassNames = [NSSet setWithObjects:@"highlight", @"awesome", nil];

[self.view addSubview:coolButton];
[self.view applyCSS:styleSheet];
```

CSS:

```
UIButton { /* NSObject class names */
    backgroundColor: #f8fbff;
    font: UIFont("Helvetica", 10pt);
}

#CoolButton { /* CSS hash/id names */
    backgroundColor: #282828;
    frame: CGRect(1,1,300,300);
}

.highlight { /* CSS classes */
    backgroundColor: #ff00ff;
}
```

DESCENDENT MATCHING:

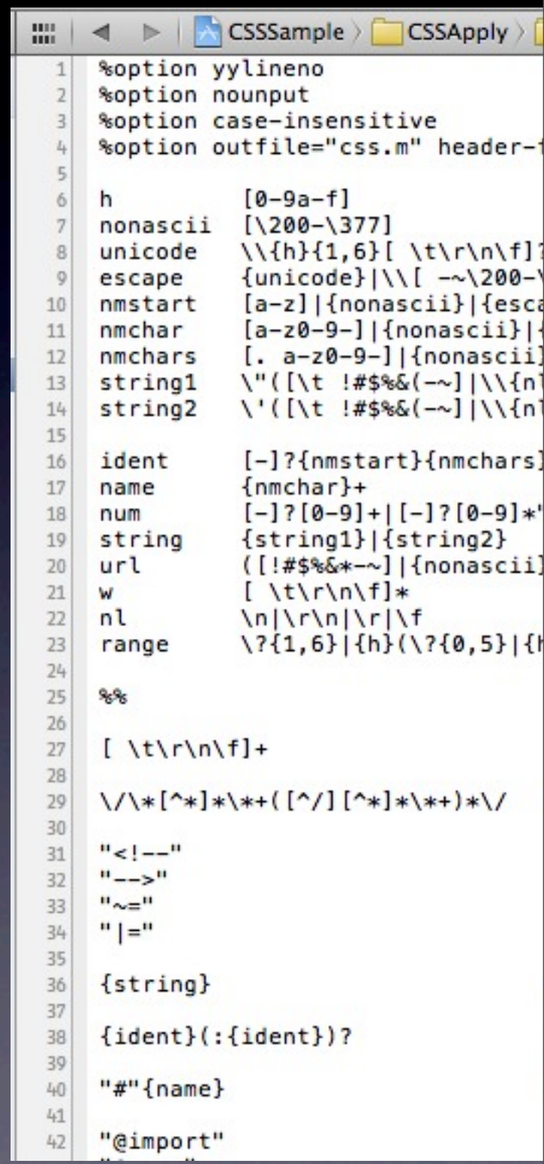
```
#CoolView UIButton .highlight {
    backgroundColor: #ff00ff;
}
```

Works great with UIView animations!

```
[UIView animateWithDuration:1.0f
                      delay:0
                    options:UIViewAnimationCurveEaseIn
                 animations:^{
                     coolButton.cssClassNames = [NSSet setWithObject:@"flashing"];
                     ...
                     //[coolButton applyCSS:styleSheet];
                     [self.view applyCSS:styleSheet];
                 }
                 completion:NULL];
```

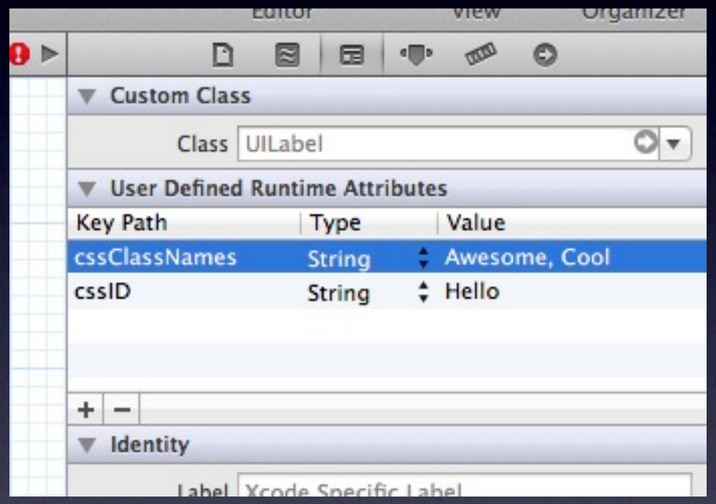
How does it work?

- A CSS tokenizer and parser was built using the CSS syntax specification on the W3C website using with Flex.
- Basic CSS properties set on their respective objects via KVC. (mappings can be registered for specially named properties)
- CSS functions can be registered to evaluate special properties (UIFont, CGRect, CGSize, url, UIImage, etc)
- Categories on UIView, NSView, and CALayer know how to traverse their hierarchies to apply the CSS rules given their precedence.



Works inside interface builder!

*on iOS 5



iOS 5 features!

- Even more powerful on iOS 5 with the new [redacted per NDA].
- It works with setting the [redacted per NDA] in your application delegate to setup [redacted per NDA]
- (Ask me later...)

Current limitations

- Compatible with a subset of CSS 2.1.
- Decedent matching works. Computing styles based on precedence works. Many other advanced features do not (yet!!)
- The "!important" flag is not allowed (it's an awful hack in CSS anyways)
- The project is currently parsing CSS and updating UIView hierarchies for simple properties (backgroundColor, text, etc)

Cute Ideas

(none of these ideas are really recommended or even fully thought out)

- Download the CSS files from the web and re-skin your app after shipping.
- Full visual layout in CSS
- CSS based animations
- @import
- user configurable CSS files (heck.. why not?)

Side note:

- Cascading style sheets are **way** more complicated then they look.
- There are complicated rules about the order how styles should be applied based.
- I heard you like inherited styles, so I styled your parent so you can style your view while you style your view.
- Computing the final style for a view has to recurse up the view tree and down the style search tree. (Don't ask... our brains hurt.)

one more thing...

JQuery selector like view searching

- UIView *view = [self.view find:@"#CoolButton"];
- NSArray *views = [self.view findAll:@".highlighted"];

```
#import <Foundation/Foundation.h>
#import "CSSStyleSheet.h"
#import "CSSSelectorTree.h"

@interface UIView (CSS)

- (void)applyAll:(CSSStyleSheet*)sheet;
- (void)apply:(CSSSelectorTree*)sheet;

- (NSArray*)find:(CSSSelector*)selector;
- (NSArray*)findAll:(CSSSelector*)selector;

-(id) CSSParent;

@end
```

Thanks!

- All the code was developed here over the weekend and we plan to continue to use this. (We need it for our stuff.)
- Releasing it open source under the X11 (BSD 2 clause) license.
- Final version will be at github: http://github.com/zbowling/CSSApply