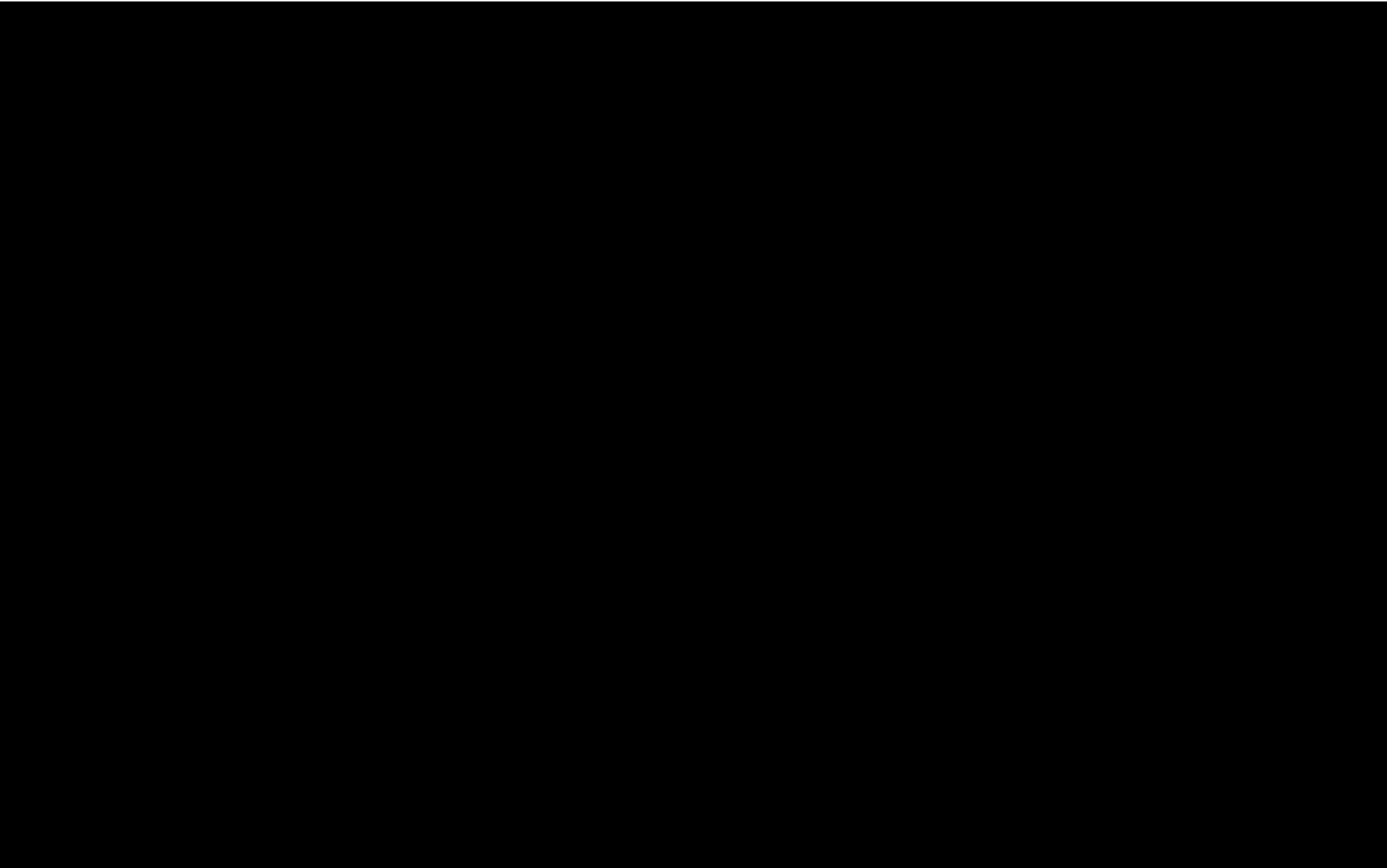# Rich-Text Editing

Using the new features of WebKit in iOS 5

Session 511

Justin Garcia
Software Engineer

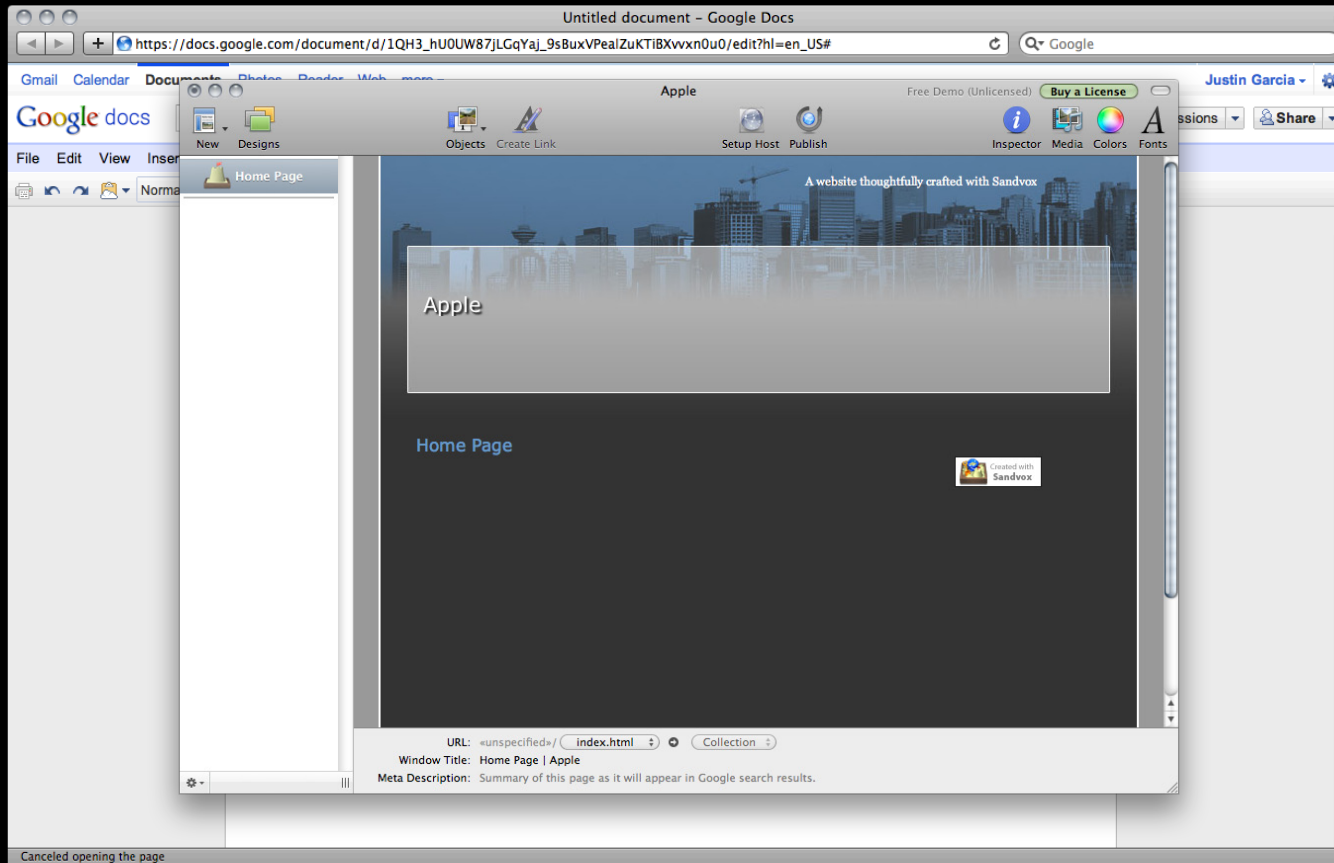These are confidential sessions—please refrain from streaming, blogging, or taking pictures

</markup>

WYRIWYG

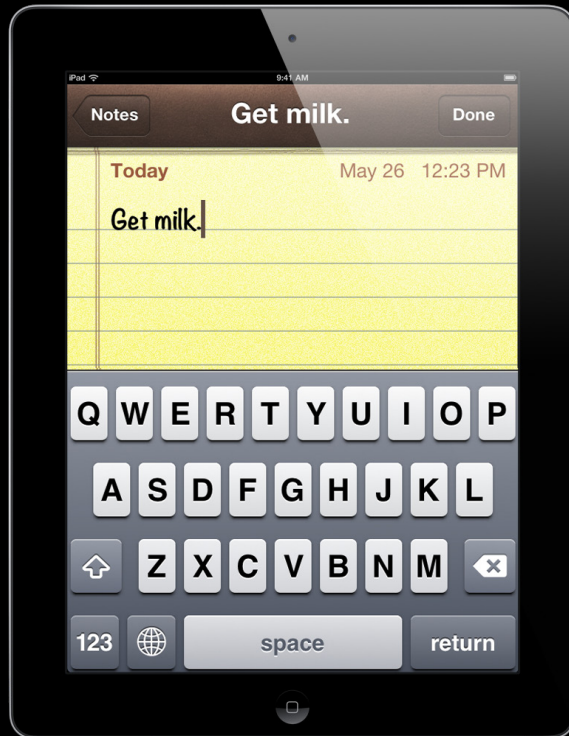# B/U

Desktop Caliber

From the Desktop

Features

- Beautiful typography
- Multistyled text
- Multicolumn layouts
- Custom backgrounds
- Transparency
- Images
- HTML export

What We're Going to Build

How?

Option One: UITextView

- Only a single style
- Only basic page layouts
- No HTML export

Option Two: Custom Text Layout Engine

You have to implement everything. Have fun!

Core Text Programming Guide
http://developer.apple.com

Option Three: WebKit

- Build a web-based editor or a native editor using UIWebView

Web App Demo

Justin Garcia
Software Engineer

Web Application

5

- contentEditable
- Positioning the cursor
- execCommand
- Validate editing UI
- Print-caliber typography
- Validate and format input
- Saving user edits

Web Application: contentEditable

Make any part of web document editable with
contentEditable

```
<div class="title" contentEditable="true">The City's Best Graffiti</div>
```

Web Application: Selection
Position the cursor programmatically

```
myDiv = document.getElementById("myDiv");

selection = window.getSelection();

selection.setPosition(myDiv, 0);
```

Web Application: Performing Commands

Perform editing commands with execCommand

5

bold   create link...   highlight

Web Application: Performing Commands
Perform editing commands with execCommand

```javascript
function bold() {
  document.execCommand("Bold");
}
function createLink(label, url) {
  var html = "<a href=" + url + ">" + label + "</a>";
  document.execCommand("InsertHTML", false, html);
}
// Each execCommand is Undo-able.
// Other execCommands: Italic, Indent/Outdent, Transpose
```

Web Application: Validating UI

Show "Bold" button in the correct state

Web Application: Validating UI
Show "Bold" button in the correct state

```javascript
function boldEnabled() {
    // Whether or not the selection is in bold text.
    return document.queryCommandEnabled("Bold");
}
```

Web Application: Typography
Before

- Ligatures and improved kerning

Web Application: Typography

After

- Ligatures and improved kerning

Graffiti

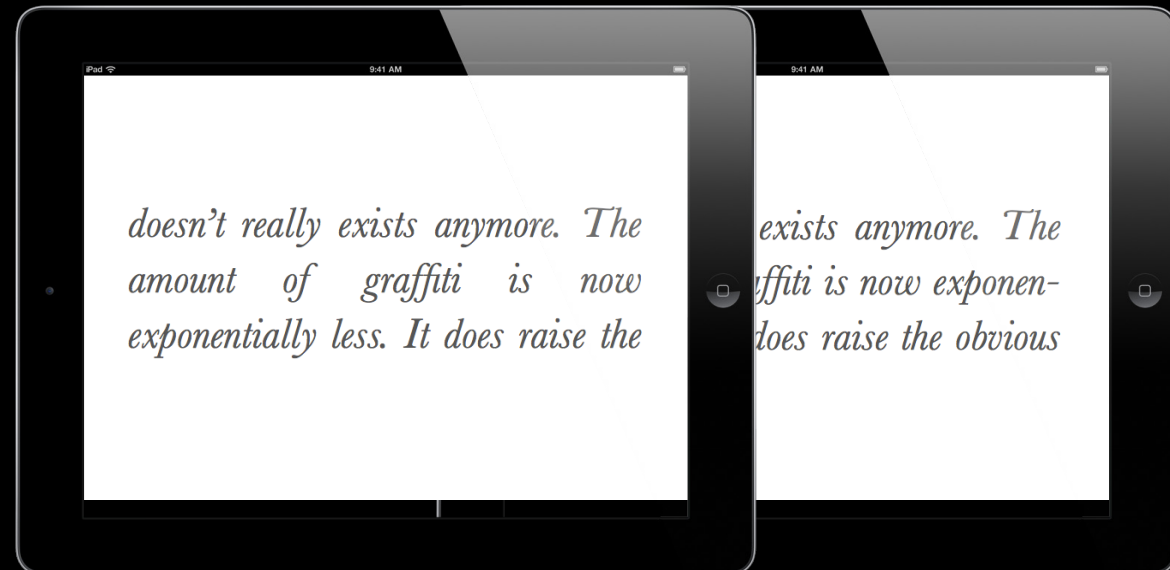# Web Application: Typography
Ligatures and improved kerning

```
body {
    text-rendering: optimizeLegibility;
}
```

Web Application: Typography
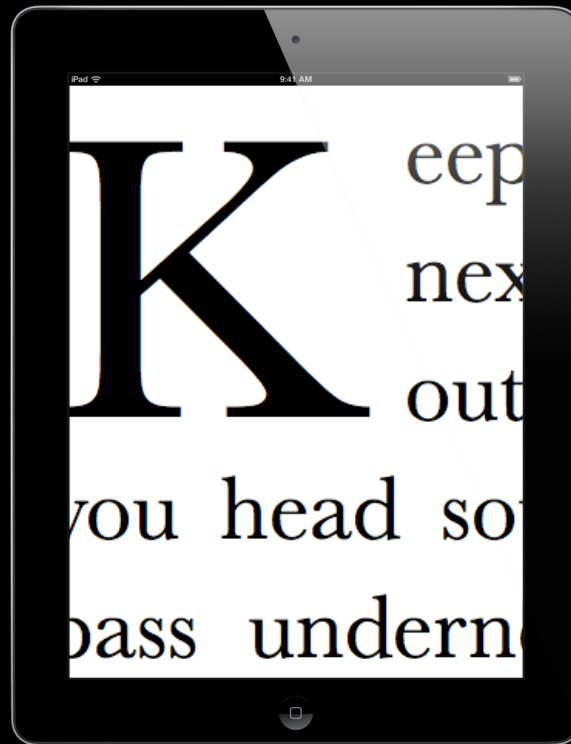## Auto-hyphenation for English

Web Application: Typography
Auto-hyphenation for English

```
body {
    -webkit-hyphens: auto;
}
```

Web Application: Typography
An aside: drop caps

Web Application: Typography
An aside: drop caps

```
:first-letter {
    font-size: 90px;
    float: left;
    margin-top: 28px;
    margin-bottom: 1px;
    margin-right: 8px;
}
```

Web Application: Validate Input

Validate input or enforce formatting with JavaScript event handlers

```
function isSingleLine(target) {
    var targetClass = target.getAttribute("class");
    return targetClass == "title" || targetClass == "date";
}

function onkeydownHandler(event) {
    return !(isSingleLine(event.target) && event.keyCode == 13);
}
```

```
<div class="title" contenteditable="true" onkeydown="return
onkeydownHandler(event)">The City's Best Graffiti</div>
```

iPad 🤶   9:41 AM

The City's Best Graffiti
Thursday, April 20th, 2011

Web Application: Saving

One option: save user edits with an XML HTTP request

```
function onkeyupHandler() {
  var xmlhttp = new XMLHttpRequest();
  xmlhttp.open("POST", "save.php", true);
  xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
  xmlhttp.onreadystatechange = handleServerResponse;
  xmlhttp.send("html=" + document.documentElement.innerHTML);
}
```

Web Application: Saving
Or save user edits using offline storage

Using HTML5 Offline Storage	Marina Thursday 3:15PM

Web App Demo

Justin Garcia

Software Engineer

Web Application: JavaScript Editing Libraries

- JavaScript editing libraries with cross-browser support:
  - TinyMCE

    http://tinymce.moxiecode.com
  - CKEditor

    http://ckeditor.com

What's Next?

Native App

Native App Demo

Justin Garcia

Software Engineer

UIWebView

- Leverage the entire web application
- Extend the system-editing menu
- Use native UI widgets
- Import photos from the user's device
- Draggable images

UIWebView
Embed the web application into a UIWebView

```
NSBundle *bundle = [NSBundle mainBundle];

NSURL *indexFileURL = [bundle URLForResource:@"index" withExtension:@"html"];

[self loadRequest:[NSURLRequest requestWithURL:indexFileURL]];
```

UIWebView
Extend the system-editing menu

UIWebView
Extend the system-editing menu

```objc
NSMutableArray *extraItems = [[NSMutableArray alloc] init];
UIMenuItem *boldItem = [[UIMenuItem alloc] initWithTitle:@"Bold" action:
@selector(bold:)];
[extraItems addObject:boldItem];
[[UIMenuController sharedMenuController].menuItems = extraItems;
```

UIWebView
Extend the system-editing menu

```objc
// For your UIWebView subclass:
- (void)bold:(id)sender {
    [self stringByEvaluatingJavaScript:@"document.execCommand('Bold')];
}


- (BOOL)canPerformAction:(SEL)action withSender:(id)sender {
    if (action == @selector(bold:))
     return YES;
    return [super canPerformAction:action withSender:sender];
}
```

UIWebView
Use native UI widgets

```objc
- (IBAction)sliderChanged:(id)sender {
    UISlider *slider = (UISlider *)sender;

    float progress = slider.value;

    int fontSize = (MAX_FONT_SIZE - MIN_FONT_SIZE) * progress +
    MIN_FONT_SIZE;

    NSString *js = [NSString stringWithFormat:@"document.body.style.fontSize
    = %dpx", fontSize];

    [self stringByEvaluatingJavaScript:js];
}
```

UIWebView

Import photos from user's device

```objc
- (void)insertImage:(NSString *)imagePath {
    NSString *html = [NSString stringWithFormat:"<div class=\"img-container
    \"><img src=\"%@\"></div>", imagePath];

    NSString *execCommand = [NSString stringWithFormat:" document.execCommand
    (\"InsertHTML\", false, \"%@\"", html];

    [self stringByEvaluatingJavaScript:execCommand];
}
```

UIWebView

Import photos from user's device

```objc
- (void)imagePickerController:(UIImagePickerController *)picker
didFinishPickingMediaWithInfo:(NSDictionary *)info {
    if (CFStringCompare((CFStringRef)mediaType, kUTTypeImage, 0)
        == kCFCompareEqualTo) {
        UIImage *image = (UIImage *)[info
    objectForKey:UIImagePickerControllerOriginalImage];
        NSString *path = [self writeImageToFile:image];
        [self insertImage:path];
    }
}
```

UIWebView
Draggable images

```
// Perform touch tracking at the App level.

// Send the touch coordinate to the web application with -
stringByEvaluatingJavaScript:.

// Find the DOM location as a Range to re-insert the image with
document.caretRangeForPoint(x, y).

// Use DOMRange to create a selection.

// Select and delete the image from its old location by programatically
selecting it and execCommand("Delete") (so that the drag will be Undo-able).

// Re-insert the image with execCommand("InsertHTML", ...).
```

Native App Demo

Justin Garcia
Software Engineer

More Information

Vicki Murley
Internet and Web Technologies Evangelist
vicki@apple.com

execCommand Documentation
Rich-Text Editing in Mozilla
https://developer.mozilla.org/en/rich-text_editing_in_mozilla

Apple Developer Forums
http://devforums.apple.com

Related Sessions

Advanced Text Processing	Mission Thursday 3:15PM
Getting Your Apps Ready for China and Other Hot New Markets	Pacific Heights Friday 10:15PM
What's New in Cocoa Touch	Pacific Heights Thursday 4:30PM

Labs

Text Editing on iOS Lab	Application Frameworks Lab A Thursday 11:30AM
Internationalization Lab	Application Frameworks Lab A Friday 11:30AM
HTML5 Offline Storage	Internet and Web Lab B Friday 9:00AM

Summary

- Why you should use WebKit-based rich-text editing
- How to get started
- How to take the next steps

Q&A