Web Technologies in iOS Apps

Using UIWebView the Sane & Effective Way

By Anne K. Halsall, Product Designer @ Quora

Just Not One Of The Cool Kids

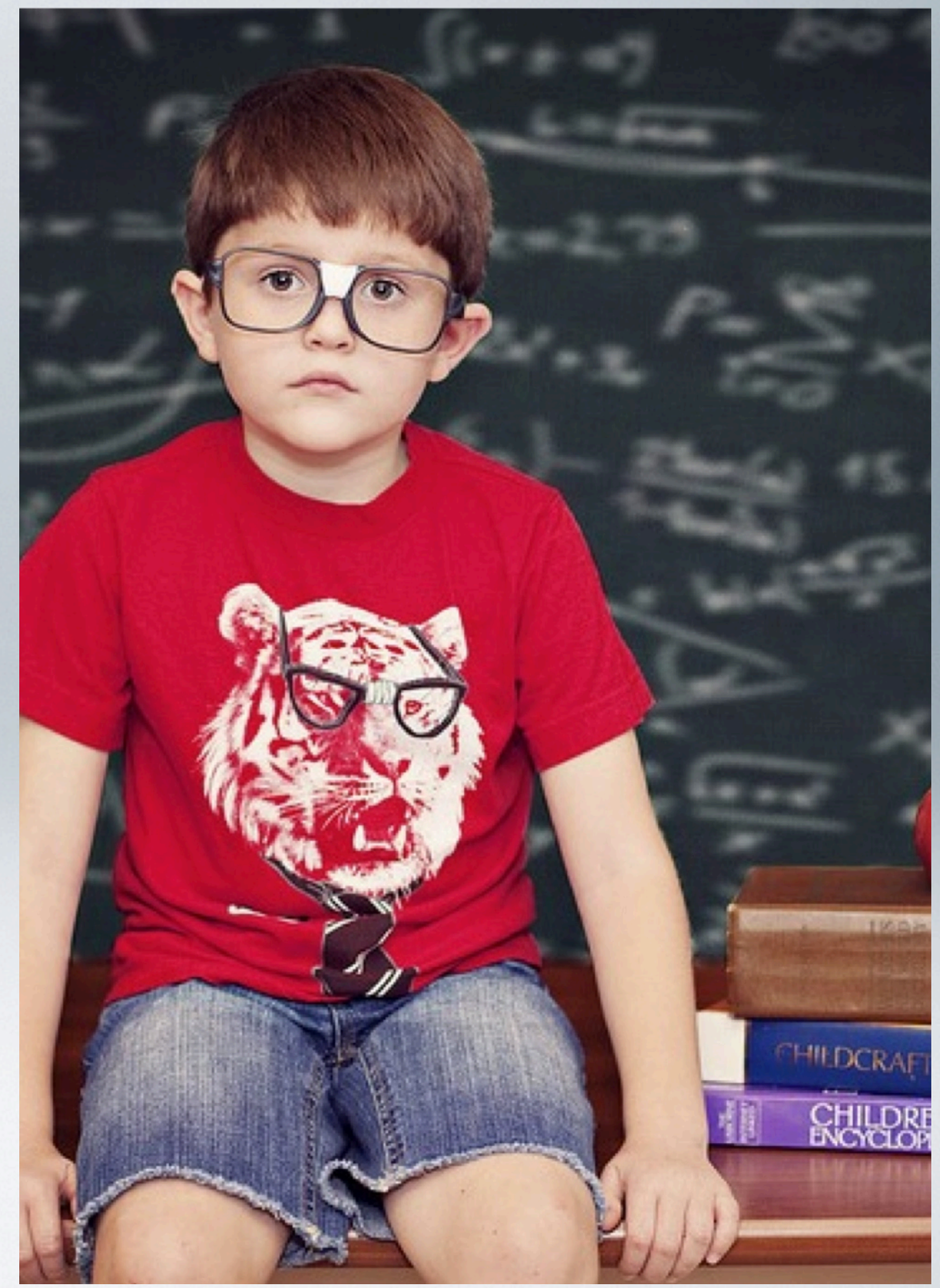
"It doesn't feel native."

"It's slow."

"It's a black box."

"The API sucks."

"It's impossible to debug."



Actually Pretty Cool When You Get To Know It

Fastest & simplest option for rich text display

Dynamic layouts make landscape and iPad views a snap

Can feel like any other UIView and be used in the same way

Easy for designers to work with

Don't Shave Yaks!

"But I really *want* to write my own rich text & layout engine!"

Treat It Like A UIView

Think Document, Not Browser

WebView as Browser

UIWebView

UIToolBar

WebView as Document

UINavigationBar

UIWebView

UITabBar

Demo: Makeover Party

- Remove shadows & gray background
- Set scroll deceleration rate to normal
- Use `autoresizingMask` and `-webkit-text-size-adjust:none` to reflow content for landscape
- Use `-webkit-touch-callout:none` to suppress link menu
- Use `-webkit-tap-highlight-color` to customize tap highlight

Demo: Component Web View

Design For Touch

Think Beyond The Web Page

Understanding Clicks vs. Touches

Click Events

- 300ms delay
- Only on anchor tags
- Padded out automatically
- Figures out gestures for you

Touch Events

- Instant response
- Any DOM node
- You manage touch area yourself
- Up to you to determine taps vs. gestures

Demo: Touch Interactions

Use Native Web Controls

It's All UIKit Under The Hood

Understanding Clicks vs. Touches

HTML Control

- `<script>alert(“hi!”);</script>`
- `<input type="range" />`
- `<input type=”datetime” />`
- `<select>...</select>`
- `<textarea>...</textarea>`

UIKit Control

- `UIAlertView`
- `UISlider`
- `UIPickerView (Date)`
- `UIPickerView (List)`
- `UITextView`

Demo: Slider

Call Out To Cocoa For Native UI

Use the right interface for the right task

Custom URL Schemes For Complex Actions

- Attach a custom URL to a link, or call document.location in JavaScript to get the web view to start a load request
- The web view's delegate will get `shouldStartLoadWithRequest:`, where you can parse the URL and do whatever you want
- Generally best to handle your own networking and block the web view from loading content

Custom URL Schemes For Complex Actions

```
- (BOOL)webView:(UIWebView *)webView shouldStartLoadWithRequest:
(NSURLRequest *)request navigationType:
(UIWebViewNavigationType)navigationType
{
    // do basically anything but let the web view handle this request
}
```

```
<a href="#" routing="myApp://viewEntry/213">View Entry #213</a>
```

```
MAURLRouter *router = [MAURLRouter sharedInstance];

[router mapURLPatternString:@"myApp://viewEntry/(initWithId:)"
toViewControllerClass:@"MAViewEntryController"];
```

Demo: Hybrid Behavior

General Tips & Tricks

- Think Retina: use images at 2x resolution and scale down in CSS
- Animate in CSS, not JS
- Debug with iOS 6 + Safari 6 web inspector
- When a load request is finished, UIWebView will reset its scroll offset; use KVO if you need to block this
- Use as little JS as possible and avoid heavy frameworks

Thank you!

@annekate // anne@quora.com

In-progress GitHub project at
http://github.com/annekate/AKWebView