

Create mobile WebApps and
NativeApps as a Ruby On Rails
developer

Native / Web

- iPhone/iPad, xCode and Objective-C
- Android, Eclipse and Java/XML
- JQuery-Mobile
- Sencha Touch
- HTML / CSS / JS

-
- **Titanium**
 - **PhoneGap**

Mobile WebApp: jQuery Mobile

- Based on jQuery (easy to learn)
- Touch-optimized layouts & UI widgets
- Themable designs
- Seriously cross-platform & cross-device

Integration in Rails 3.1

- `/assets/javascripts/jquery.mobile.js`
- `/assets/stylesheets/jquery.mobile.css`
- `/assets/images/images/` (Workaround! Lieber `asset_path` helper nutzen und Pfade in der css ersetzen.)

Demo: Page structure

- Data Roles (HTML 5 Data attributes)
- Page, Header, Content and Footer

AJAX

- Ajax by default
- New pages loaded in the DOM
- Smooth page transitions
- Rich, native-like experience
- Preloading of pages

No AJAX

- Disable AJAX for specific links
- `rel="external", data-ajax="false"`
- More Rails like behaviour
- Completely disable AJAX:
`$.mobile.ajaxEnabled=false`

Tips

- Viewport Meta Tag

```
<meta name="viewport" content="width=device-width"/>
```

- Home Screen Icon

```
<link rel="apple-touch-icon" href="icon.png"/>
```

Tips

- Full Screen Flag (iOS)

```
<meta name="apple-mobile-web-app-capable"  
content="yes"/>
```

- Splash Screen (iOS)

```
<link rel="apple-touch-startup-image"  
href="startup.png"/>
```

Tips

- Status Bar Style

```
<meta name="apple-mobile-web-app-status-bar-style" content="black"/>
```

- IOS 5 – New mobile Safari stuff
goo.gl/YVuQY

Native iPhone App without Objective-C: **Titanium**

- Supports all of the iPhone, iPad and Android UI, including table views, scroll views, native buttons, switches, tabs, popovers and more.

Titanium

- PURE JAVASCRIPT
- Compiled to Objective-C Code

What you need to get started

- Development tools for iOS
XCODE / IOS SDK
- Titanium Studio / Textmate

=> Play with it

Structure your JS Code

- MVC Pattern for JS
- Dont pollute your global namespace
- Use Closures

Test your code with Jasmine!

Use CoffeeScript!

Public API

- Back to Rails!
- Easy API with RABL gem
- Writing a public API using RABL and JSON Templates in the View.

RABL

- The RESTful Way
- We already likely have html representations to interact with.
- When building APIs, the simplest way to think about them is to think of the JSON API as “just another view representation” that lives alongside your HTML templates.

RABL

- `gem 'rabl'`
- `respond_to :json, :xml`

```
collection @posts
```

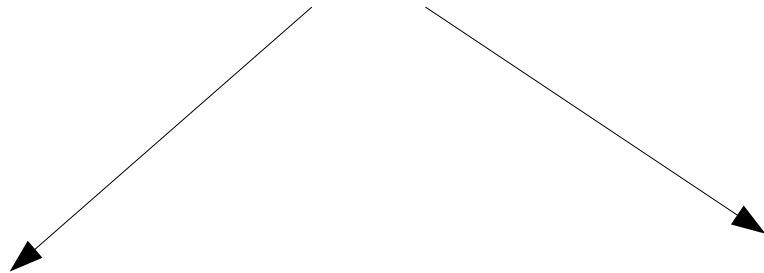
```
attributes :id, :title, :subject
```

```
child(:user) { attributes :full_name }
```

```
node(:read) { |post| post.read_by?(@user) }
```

Summary

Ruby On Rails



Mobile WebApp
Html representations/views

Public API
JSON/XML representations/views



Mobile NativeApp
(iPhone/Appstore)
Consumes Public API

TODO

- Namespace routes for API
- Separate frontend from Backend (Admin namespace)
- Always write tests!
Rspec, cucumber, selenium, jasmine
- Prepare App for AppStore submission!