From `rails new` to App Store

Building iOS and Android Apps with Rails 5 and Turbolinks

Who We Are

Geoff Buesing @gbuesing

Trevor Turk @trevorturk

What we built

- a Rails chat app

- ...with a native iOS app

- ...and a native Android app

github.com/gbuesing/turbochat

What we used

- Rails 5

- ActionCable

- Turbolinks

- Turbolinks-iOS adapter

- Turbolinks-Android adapter

What we didn't use

- PhoneGap

- Cordova

- React Native

- RubyMotion

- etc

Why build native apps with Turbolinks?

- Simply use Rails to build bulk of app functionality

- Progressively enhance with native functionality as needed

- Deliver bug fixes and new features with a simple web app deploy

Overview of Turbolinks

Turbolinks makes your web application faster.

"Get the performance benefits of a **single-page application** without the added complexity of a client-side JavaScript framework."

github.com/turbolinks/turbolinks

Turbolinks history lesson (part one):

- **pjax**
- Turbolinks "Classic" (Rails 4.0)
- Turbolinks 5 (Rails 5.0)

pjax

Jquery plugin to make GitHub's inline code browser faster.

- Replace page elements without a full page reload
- Use jQuery and pushState() to maintain URLs

github.com/defunkt/jquery-pjax

📕 gbuesing / **turbochat**   👁 **Unwatch** ▾ **2**   ★ **Star** **0**   ⑂ **Fork** **0**

‹› Code   ⚠ Issues **0**   ⑂ Pull requests **0**   ▥ Projects **0**   📖 Wiki   ∿ Pulse   �░ Graphs

Branch: **master** ▾   **turbochat** / **app** /   Create new file   Upload files   Find file   History

🖼 **gbuesing** Auto link message body   Latest commit **912dead** 23 days ago

..

📁 assets	Scroll to end of messages on page load and after every new message. A...	24 days ago
📁 channels	Add identified_by current_user for channel (#7)	2 months ago
📁 controllers	Set title tag to room name when in room so that app can show name in ...	24 days ago
📁 helpers	rails new	2 months ago
📁 jobs	rails new	2 months ago
📁 mailers	rails new	2 months ago
📁 models	users/rooms/messages (#1)	2 months ago
📁 views	Auto link message body	23 days ago

```
<!DOCTYPE html>
<html>
<head>
  <script src="pjax.js"></script>
</head>
<body>
  <div id="file-navigation">
    <!-- links to folders, files, etc -->
  </div>
  <script>
    // enable faster browsing within the file-navigation
    $(document).pjax('a', '#file-navigation')
  </script>
</body>
</html>
```

Turbolinks history lesson (part two):

- pjax

- **Turbolinks "Classic" (Rails 4.0)**

- Turbolinks 5 (Rails 5.0)

Turbolinks "Classic"

While pjax only replaces part of the page, Turbolinks operates on the entire page. When you follow a link, Turbolinks automatically fetches the page, swaps in its `<body>`, and merges its `<head>`, **all without incurring the cost of a full page load**."

github.com/turbolinks/turbolinks-classic

```html
<!DOCTYPE html>
<html>
<head>
  <title>My application</title>
  <script src="turbolinks-classic.js"></script>
</head>
<body>
  <a href="/about">About us</a>
</body>
</html>
```

Turbolinks history lesson (part three):

- pjax

- Turbolinks "Classic" (Rails 4.0)

- **Turbolinks 5 (Rails 5.0)**

Turbolinks 5

- Ground-up rewrite with the same core idea as Turbolinks Classic

- Drops jQuery dependency

- Supports mobile apps via native adapters

github.com/turbolinks/turbolinks

Turbolinks adapters for iOS and Android

The native adapters automatically manage a single web view instance across multiple view controllers, giving you native navigation UI with all the client-side performance benefits of Turbolinks.

You can use native code and navigation UI progressively where needed. You're not locked into a custom app development framework.
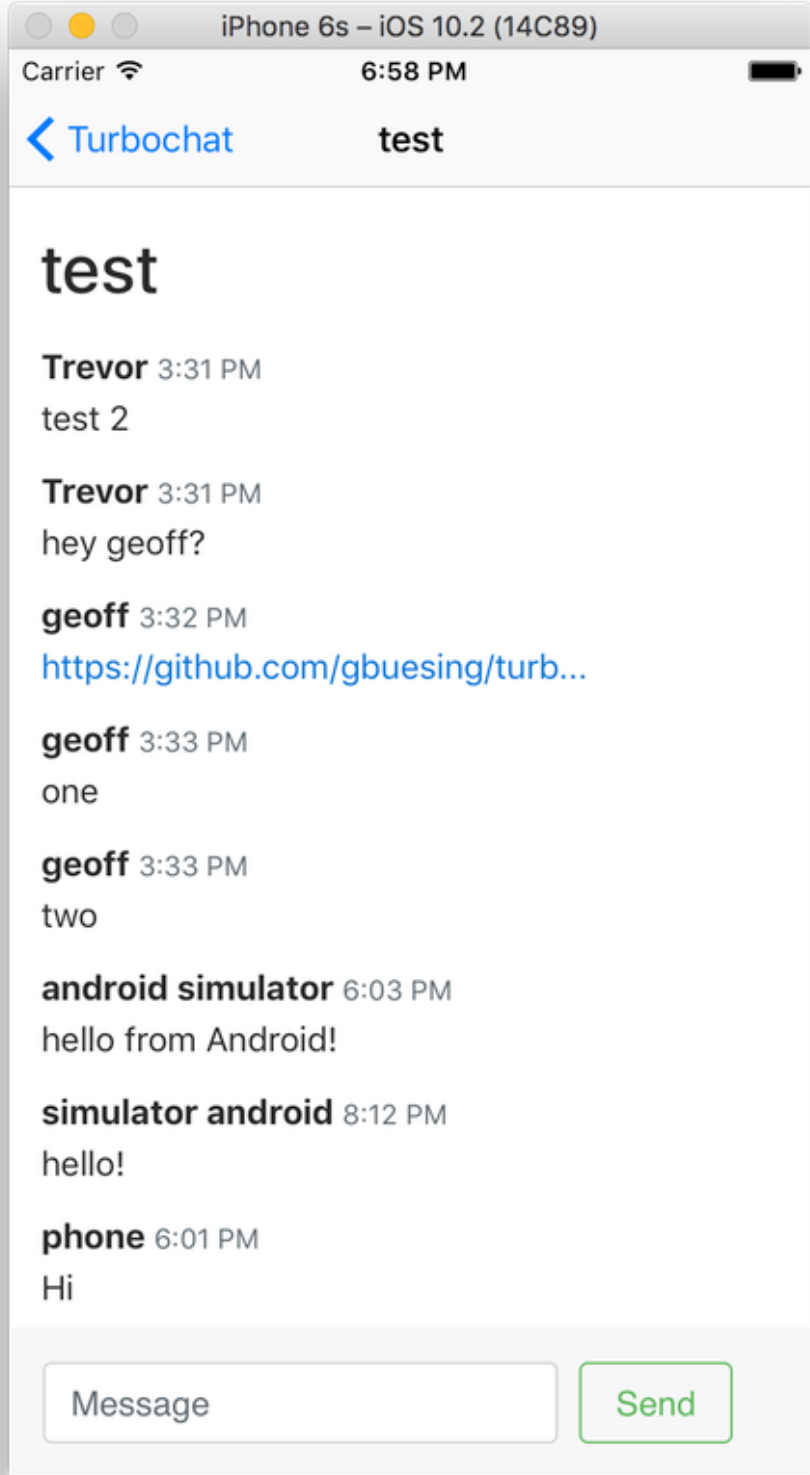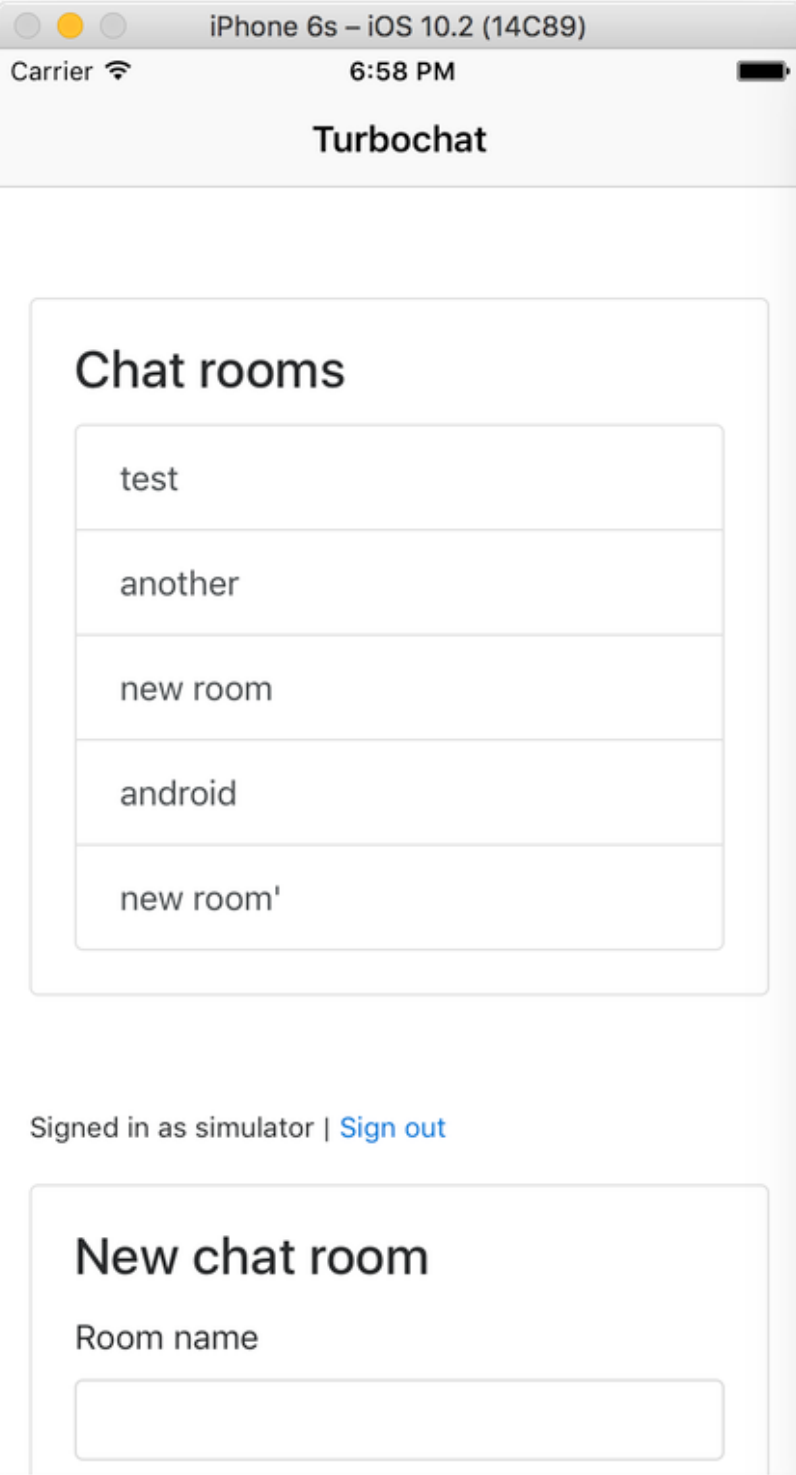
github.com/turbolinks/turbolinks-ios
github.com/turbolinks/turbolinks-android

Turbolinks adapter benefits:

- Leverage your existing web views on native platforms

- Use HTML/CSS for layouts, instead of complex storyboards etc

- Use the Safari developer mode and Chrome web tools you love

- Release native app bug fixes with a simple web deploy

The app: Turbochat

Breaking out of the web view box, i.e. hybrid apps

Communicating with your iOS and Android apps

- Web app can send messages to the native app via Javascript

- Native app can call Javascript functions in the web app

Example use cases

- Update web view cookie when device geolocation changes

- Display a native alert

- Add button to native toolbar

Web app sends message to iOS

```js
// js
window.webkit.messageHandlers.log.postMessage("hi");
```

...iOS app exposes "log" handler:

```swift
// swift
webViewConfiguration.userContentController
  .addScriptMessageHandler(self, name: "log")

func userContentController(userContentController:
      WKUserContentController, didReceiveScriptMessage
   message: WKScriptMessage) {
   print("Logging: \(message.body)")
}
```

Web app sends message to Android

```js
// js
window.AndroidNative.log("hi")
```

...Android app exposes "log" handler:

```java
// java
@JavascriptInterface
public void log(String value) {
        Log.d("Logging:", value)
}

webView.addJavascriptInterface(this, "AndroidNative");
```

iOS calls web app function

```swift
// swift
let js = "(function() { return new Date() })();";

webView.evaluateJavaScript(js) { (result, error) in
  print("Date is \(result)")
}
```

# Android calls web app function

```
String js = "(function() { return new Date() })();";

ValueCallback callback = new ValueCallback<String>() {
    @Override
    public void onReceiveValue(String s) {
        Log.d("Date is:", s);
    }
};


webView.evaluateJavascript(js, callback);
```

Use native code and navigation UI progressively where needed

- Intercept links and render select views as native

- Use completely native views for pages that need native speed
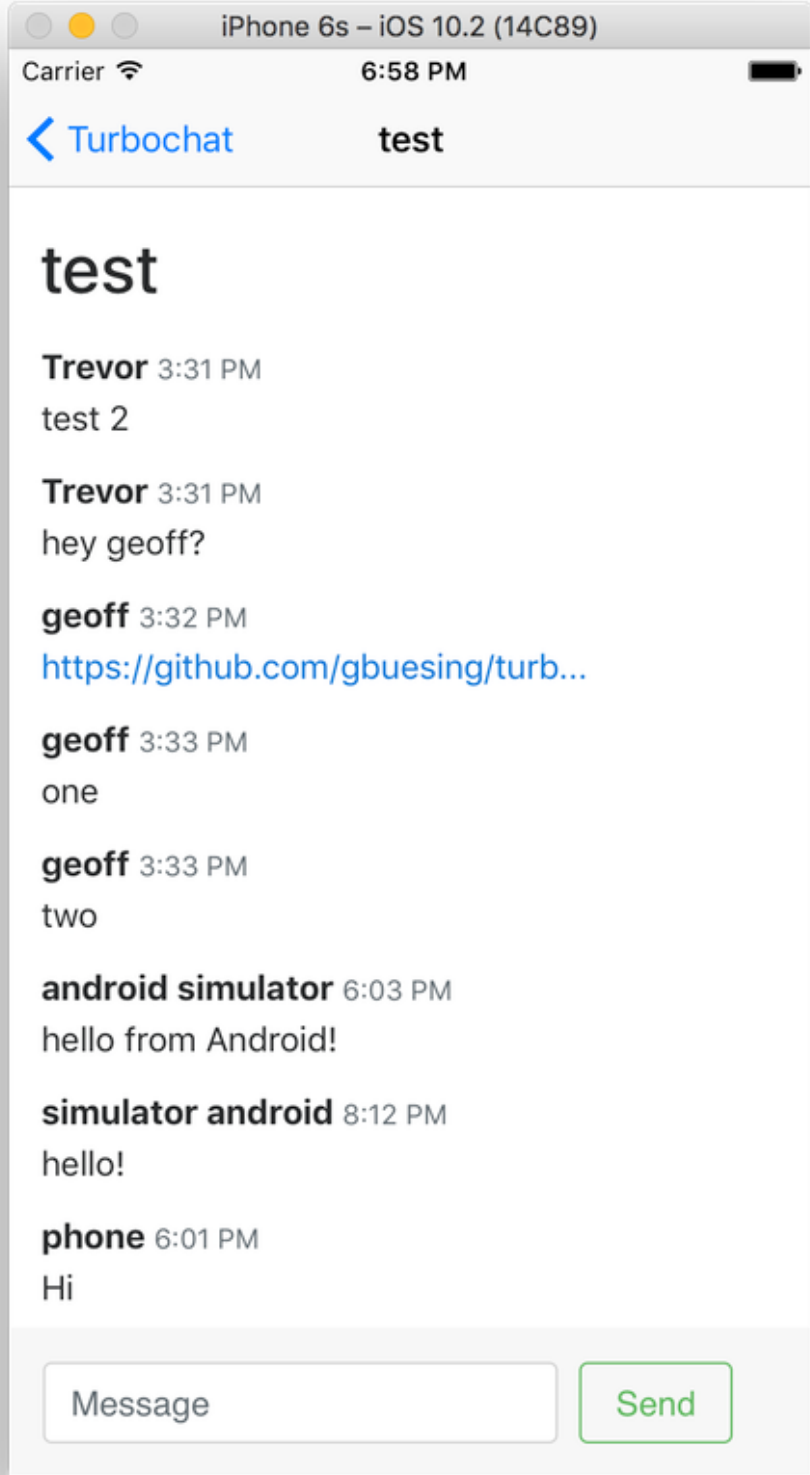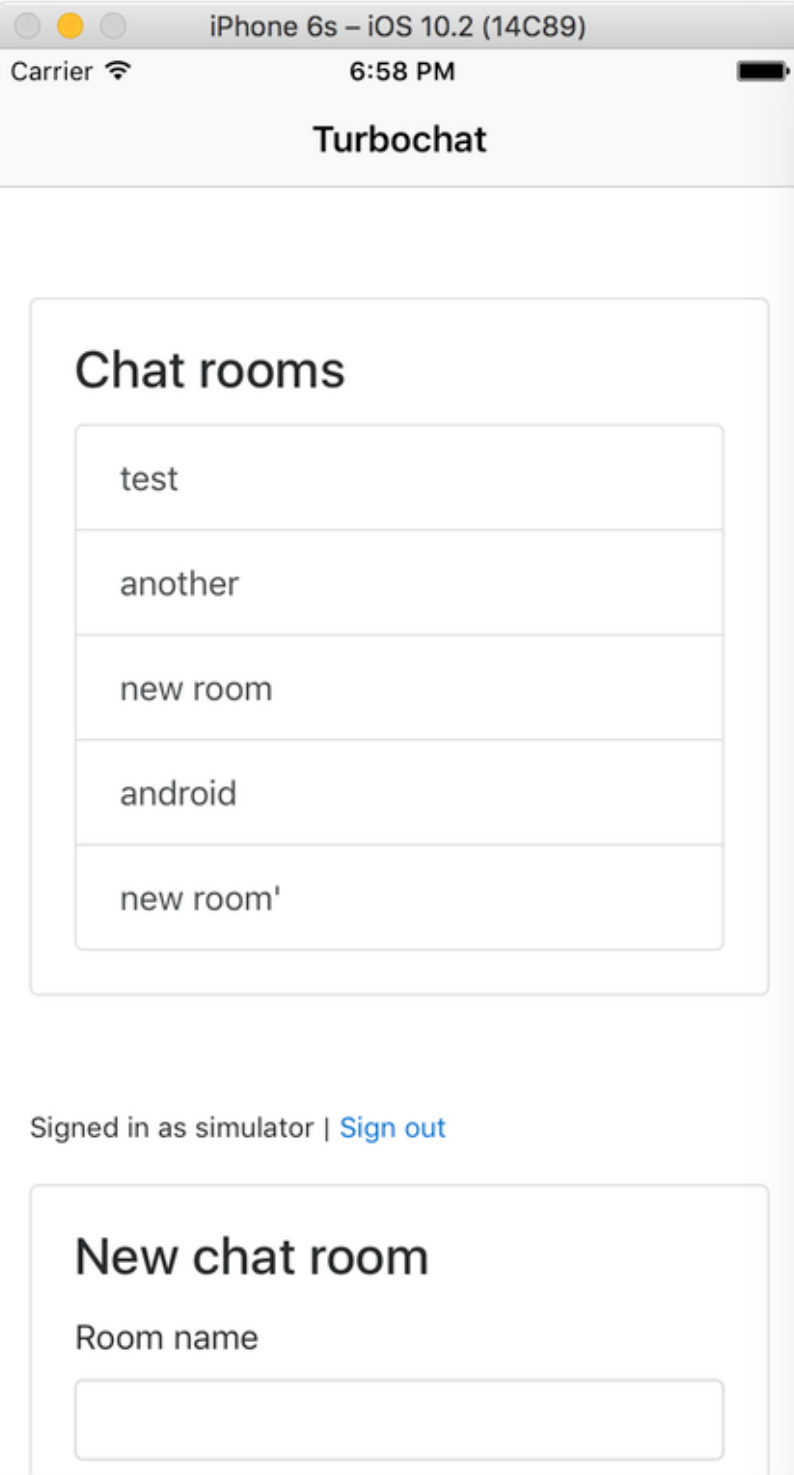
Intercept a link and render native (iOS)

```html
<a href="/rooms">Show me rooms</a>
```

```swift
// swift
func session(session: Session, didProposeVisitToURL
    URL: NSURL, withAction action: Action) {
    if URL.path == "/rooms" {
        // render native view
    } else {
        presentVisitableForSession(session, URL: URL,
            action: action)
    }
}
```

Intercept a link and render native (Android)

```html
<a href="/rooms">Show me rooms</a>
```

```java
// java
@Override
public void visitProposedToLocationWithAction(
        String location, String action) {
    if (location == 'https://myapp.com/rooms') {
        // render native view
    } else {
        Intent intent = new Intent(this,
                MainActivity.class);
        intent.putExtra(INTENT_URL, location);
        this.startActivity(intent);
    }
}
```

Detecting native app server-side

- Customize HTML, JS and CSS that your Rails app serves to native apps

Detecting native app server-side

Set custom User-Agent header in native apps:

```
// iOS
let config = WKWebViewConfiguration()
config.applicationNameForUserAgent = "Turbochat iOS"
```

```
// Android
TurbolinksSession.getDefault(this).getWebView()
    .getSettings()
    .setUserAgentString("Turbochat Android");
```

Detecting native app server-side

Helper method:

```ruby
class ApplicationController < ActionController::Base
private
  def native_app?
    request.user_agent =~ /Turbochat/
  end
  helper_method :native_app?
end
```

Detecting native app server-side

In views:

```erb
<% if native_app? %>
  <%= javascript_include_tag 'native-only-js' %>
<% end %>

<% unless native_app? %>
  <nav>Navbar</nav>
<% end %>
```

Conclusion

Pros

- Use mostly web tech you know (much easier to build adaptive views in HTML vs. native constraint layouts)

- Publish apps in the App Store and Play Store

- Deploy most bug fixes and new features for your web app without going through the store

Cons

- There's no magic, you'll need to learn Xcode and Android Studio

- ...and Swift and Java

Questions?

Geoff Buesing @gbuesing

Trevor Turk @trevorturk