

React Native

oliver.ochs@holisticicon.de

 [@holisticicon](https://twitter.com/holisticicon)



Use Case

- New Media Startup

Use Case

- New Media Startup
- Mediatheken-App übergreifend über kostenlose Mediatheken

Use Case

- New Media Startup
- Mediatheken-App übergreifend über kostenlose Mediatheken
- Welche (neuen) Filme gibt es wo

Use Case

- New Media Startup
- Mediatheken-App übergreifend über kostenlose Mediatheken
- Welche (neuen) Filme gibt es wo
- Nutzung im Webbrowser

Use Case

- New Media Startup
- Mediatheken-App übergreifend über kostenlose Mediatheken
- Welche (neuen) Filme gibt es wo
- Nutzung im Webbrowser
- Disclaimer: Eine Film-Liste ist das "Hello World" von React Native ;-)

React

- Open Source View Rendering Library

React

- Open Source View Rendering Library
- von Facebook für eigene Anwendungen

Reife

■ ist nicht neu (2013)

Reife

- ist nicht neu (2013)
- Instagram

Reife

- ist nicht neu (2013)
- Instagram
- Wall Street Journal

Reife

- ist nicht neu (2013)
- Instagram
- Wall Street Journal
- Netflix

Reife

- ist nicht neu (2013)
- Instagram
- Wall Street Journal
- Netflix
- Airbnb

Reife

- ist nicht neu (2013)
- Instagram
- Wall Street Journal
- Netflix
- Airbnb
- könnte also auch für unsere Mediatheken-App verwendet werden

Motivation

- UI-Komponenten besser (vorhersehbarer) entwickeln

Motivation

- UI-Komponenten besser (vorhersehbarer) entwickeln
- Hierarchie von modularisierten Komponenten

Motivation

- UI-Komponenten besser (vorhersehbarer) entwickeln
- Hierarchie von modularisierten Komponenten
- Komponenten beschreiben Struktur sowie Daten und Methoden

Komponenten sind Klassen mit einer Render-Methode

```
var Movie = React.createClass({  
  render() {  
    return (  
      <div>Hello EnterJS</div>  
    );  
  }  
});
```

Komponenten haben Properties

```
var Movie = React.createClass({
  propTypes : {
    item : React.PropTypes.object
  },
  render() {
    return (
      <div>
        <span><img src={this.props.item.posters.thumbnail} width="80px"
        <span>{this.props.item.title}</span>
        <span >{this.props.item.year}</span>
      </div>
    );
  }
});
```

Properties werden von außen in die Komponente gereicht

```
<Movie item={{  
  title: "Hello EnterJS",  
  year: "2016",  
  posters: {  
    thumbnail: "https://www.enterjs.de/images/logo.svg"  
  }  
}}></Movie>
```



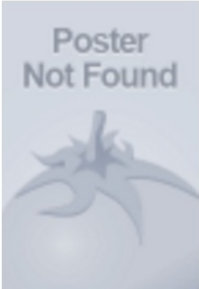

Komponenten haben einen internen Zustand

```
var MovieList = React.createClass({
  getInitialState() {
    return {
      movies:[]
    }
  },
  fetchData() {
    fetch('MoviesExample.json')
      .then((response) => response.json())
      .then((data) => {
        this.setState({
          movies : data.movies
        })
      });
  },
  componentDidMount : function(){
    this.fetchData();
  },
});
```

Komponenten lassen sich in einer Hierarchie rendern

```
var MovieList = React.createClass({
  render() {
    return (
      <div>
        {this.state.movies.map((result) => {
          return (<Movie key={result.id} item={result}></Movie>)
        })}
      </div>
    )
  },
});
```

Hands On Code

Poster	Title	Year
	Chain Reaction	1996
	React! A Woman's Guide to Safety and Basic Self-Defense	1996
	Reaction	
	Reactor	2013

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)
- Virtual DOM / Virtual DOM Diffing

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)
- Virtual DOM / Virtual DOM Diffing
- Wiederverwendbare Komponenten

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)
- Virtual DOM / Virtual DOM Diffing
- Wiederverwendbare Komponenten
- Methoden und Markup gehören zusammen

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)
- Virtual DOM / Virtual DOM Diffing
- Wiederverwendbare Komponenten
- Methoden und Markup gehören zusammen
- Bleeding Edge JavaScript

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)
- Virtual DOM / Virtual DOM Diffing
- Wiederverwendbare Komponenten
- Methoden und Markup gehören zusammen
- Bleeding Edge JavaScript
- Purity

Was ist daran das besondere?

- Deklarative Programmierung (wie HTML)
- Virtual DOM / Virtual DOM Diffing
- Wiederverwendbare Komponenten
- Methoden und Markup gehören zusammen
- Bleeding Edge JavaScript
- Purity
- Großes Ökosystem (Redux, Relay, GraphQL, ...)

Reacting to change over following a plan

- New Media Startup

Reacting to change over following a plan

- New Media Startup
- Nutzer verwenden Smartphones und Tablets zum Konsumieren von Filmen

Reacting to change over following a plan

- New Media Startup
- Nutzer verwenden Smartphones und Tablets zum Konsumieren von Filmen
- Unsere (responsive) Website ist nicht im Store

Reacting to change over following a plan

- New Media Startup
- Nutzer verwenden Smartphones und Tablets zum Konsumieren von Filmen
- Unsere (responsive) Website ist nicht im Store
- Wir brauchen eine mobile App!

Reacting to change over following a plan

- New Media Startup
- Nutzer verwenden Smartphones und Tablets zum Konsumieren von Filmen
- Unsere (responsive) Website ist nicht im Store
- Wir brauchen eine mobile App!
- Zielgruppe: iOS und Android, später evtl. Universal Windows Plattform App (UWP)...

Do The Simplest Thing That Could Possibly Work

- Wir haben bereits eine Website

Do The Simplest Thing That Could Possibly Work

- Wir haben bereits eine Website
- Wir können die Webseite in einem WebView rendern

Do The Simplest Thing That Could Possibly Work

- Wir haben bereits eine Website
- Wir können die Webseite in einem WebView rendern
- Mögliche Technologie: Apache Cordova bzw. Adobe PhoneGap

User Experience

- Jede Plattform hat ihre Besonderheiten

User Experience

- Jede Plattform hat ihre Besonderheiten
- Spezifische UI-Controls (z.B. Progress Bar, Radio, ...)

User Experience

- Jede Plattform hat ihre Besonderheiten
- Spezifische UI-Controls (z.B. Progress Bar, Radio, ...)
- Unterschiedliche Betriebssystemversionen (iOS 6 vs. iOS 7)

User Experience

- Jede Plattform hat ihre Besonderheiten
- Spezifische UI-Controls (z.B. Progress Bar, Radio, ...)
- Unterschiedliche Betriebssystemversionen (iOS 6 vs. iOS 7)
- Eingeschränkte Gestensteuerung

User Experience

- Jede Plattform hat ihre Besonderheiten
- Spezifische UI-Controls (z.B. Progress Bar, Radio, ...)
- Unterschiedliche Betriebssystemversionen (iOS 6 vs. iOS 7)
- Eingeschränkte Gestensteuerung
- Fühlt sich nie echt an

User Experience

- Jede Plattform hat ihre Besonderheiten
- Spezifische UI-Controls (z.B. Progress Bar, Radio, ...)
- Unterschiedliche Betriebssystemversionen (iOS 6 vs. iOS 7)
- Eingeschränkte Gestensteuerung
- Fühlt sich nie echt an
- Ein Smartphone ist kein Desktop-Rechner

User Experience

- Jede Plattform hat ihre Besonderheiten
- Spezifische UI-Controls (z.B. Progress Bar, Radio, ...)
- Unterschiedliche Betriebssystemversionen (iOS 6 vs. iOS 7)
- Eingeschränkte Gestensteuerung
- Fühlt sich nie echt an
- Ein Smartphone ist kein Desktop-Rechner
- Mark Zuckerbergs "biggest mistake"



Mark Zuckerberg: Our Biggest Mistake Was Betting Too Much On HTML5

Posted Sep 11, 2012 by [Drew Olanoff \(@drew\)](#)

71
SHARES



Next Story



Today, Mark Zuckerberg revealed that Facebook's mobile strategy relied too much on HTML5, rather than native applications.

Not only was this a big mistake with mobile, but Zuckerberg says that its biggest mistake period was the focus on HTML5. This is the first time that the Facebook CEO has openly admitted this, but things are looking good for the new iOS native app. According to Zuckerberg, people are consuming twice as

many feed stories since the update to the new iOS app, which is great.

The first half year has been a little bit slow on product, but for the next six months I expect a lot of really cool stuff.

This "really cool stuff" will probably have monetization in mind, as it's very clear that **mobile is the path to ad revenue** for the company.

Quelle: techcrunch.com

Native App

- Optimale UX auf allen Plattformen

Native App

- Optimale UX auf allen Plattformen
- Entwicklung für alle Plattformen aufwändig
(Swift/Objective C vs. Java), Web-Entwickler können
HTML, CSS und JS

Native App

- Optimale UX auf allen Plattformen
- Entwicklung für alle Plattformen aufwändig
(Swift/Objective C vs. Java), Web-Entwickler können
HTML, CSS und JS
- Developer Experience mäßig durch langsamen
Roundtrip

Native App

- Optimale UX auf allen Plattformen
- Entwicklung für alle Plattformen aufwändig
(Swift/Objective C vs. Java), Web-Entwickler können
HTML, CSS und JS
- Developer Experience mäßig durch langsamen
Roundtrip
- Release Zyklus durch Reviews langsam

Native App

- Optimale UX auf allen Plattformen
- Entwicklung für alle Plattformen aufwändig
(Swift/Objective C vs. Java), Web-Entwickler können HTML, CSS und JS
- Developer Experience mäßig durch langsamen Roundtrip
- Release Zyklus durch Reviews langsam
- Optimierung (A/B-Tests) schwierig

Cross Compiler

- **Microsoft Xamarin** kompiliert C# zu ObjectiveC bzw. Java

Cross Compiler

- **Microsoft Xamarin** kompiliert C# zu ObjectiveC bzw. Java
- Konsolidierte Codebasis für iOS und Android

Cross Compiler

- **Microsoft Xamarin** kompiliert C# zu ObjectiveC bzw. Java
- Konsolidierte Codebasis für iOS und Android
- Unterschiedliche UI-Entwicklung für iOS und Android

Cross Compiler

- **Microsoft Xamarin** kompiliert C# zu ObjectiveC bzw. Java
- Konsolidierte Codebasis für iOS und Android
- Unterschiedliche UI-Entwicklung für iOS und Android
- Kein "write once - run everywhere"-Ansatz

Cross Compiler

```
using Android.App;
using Android.OS;
using Android.Widget;

namespace MoviesList
{
    [Activity(Label = "Movies List", MainLauncher = true, Icon = "@drawable/icon")]
    public class MainActivity : Activity
    {
        protected override void OnCreate(Bundle bundle)
        {
            base.OnCreate(bundle);
            SetContentView(Resource.Layout.Main);
            var moviesListView = FindViewById<ListView>(Resource.Id.moviesListView);
            var moviesAdapter = new ArrayAdapter<Movie>(this, Android.Resource.Layout
            moviesListView.Adapter = moviesAdapter;
        }
    }
}

// Quelle: http://blog.falafel.com/
```


Cross Compiler

Cross Compiler

- Web-Entwickler können HTML, CSS und JS

Cross Compiler

- Web-Entwickler können HTML, CSS und JS
- ~~Hohe Lizenzkosten (\$1000 pro Plattform pro Entwickler)~~

Cross Compiler

- Web-Entwickler können HTML, CSS und JS
- ~~Hohe Lizenzkosten (\$1000 pro Plattform pro Entwickler)~~
- Kleine Community wegen des kommerziellen Ansatzes

Native Mobile Apps in JavaScript

■ Appcelerator Titanium

Native Mobile Apps in JavaScript

- **Appcelerator Titanium**
- Gibt es seit 2008 / 2009

Native Mobile Apps in JavaScript

- **Appcelerator Titanium**
- Gibt es seit 2008 / 2009
- Läuft auf den gewünschten Plattformen

Native Mobile Apps in JavaScript

- **Appcelerator Titanium**
- Gibt es seit 2008 / 2009
- Läuft auf den gewünschten Plattformen
- Alloy als MVC-Framework

Native Mobile Apps in JavaScript

- **Appcelerator Titanium**
- Gibt es seit 2008 / 2009
- Läuft auf den gewünschten Plattformen
- Alloy als MVC-Framework
- Wesentliche Teile sind

Native Mobile Apps in JavaScript

- **Appcelerator Titanium**
- Gibt es seit 2008 / 2009
- Läuft auf den gewünschten Plattformen
- Alloy als MVC-Framework
- Wesentliche Teile sind [OpenSource](#)

Native Mobile Apps in JavaScript

```
<Alloy>
  <Require id="index" src="directory" platform="android, windows" />
  <!-- iOS Window -->
  <NavigationWindow id="nav" platform="ios" class="container">
    <Require src="directory" />
  </NavigationWindow>
  <!-- MobileWeb -->
  <Window platform="mobileweb">
    <NavigationGroup id="nav" class="container">
      <Require src="directory" />
    </NavigationGroup>
  </Window>
</Alloy>
```

Native Mobile Apps in JavaScript

- Ansatz klingt vielversprechend

Native Mobile Apps in JavaScript

- Ansatz klingt vielversprechend
- Deklarative UI (wie HTML)

Native Mobile Apps in JavaScript

- Ansatz klingt vielversprechend
- Deklarative UI (wie HTML)
- Logik in JS (wie im Web)

Native Mobile Apps in JavaScript

- Ansatz klingt vielversprechend
- Deklarative UI (wie HTML)
- Logik in JS (wie im Web)
- Developer Experience

Native Mobile Apps in JavaScript

- Ansatz klingt vielversprechend
- Deklarative UI (wie HTML)
- Logik in JS (wie im Web)
- Developer Experience
- Alles UI-Controls sind nativ: kein HTML, kein Browser, keine WebViews

Native Mobile Apps in JavaScript

- Ansatz klingt vielversprechend
- Deklarative UI (wie HTML)
- Logik in JS (wie im Web)
- Developer Experience
- Alles UI-Controls sind nativ: kein HTML, kein Browser, keine WebViews
- Titanium könnte Vorbild für React Native gewesen sein

React Native - A framework for building native apps using react

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil
- Developer Experience wie im Web

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil
- Developer Experience wie im Web
- **Learn once, write everywhere**

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil
- Developer Experience wie im Web
- **Learn once, write everywhere**
- kein HTML, kein Browser, keine WebViews

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil
- Developer Experience wie im Web
- **Learn once, write everywhere**
- kein HTML, kein Browser, keine WebViews
- Native Components

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil
- Developer Experience wie im Web
- **Learn once, write everywhere**
- kein HTML, kein Browser, keine WebViews
- Native Components
- Touch Handling

React Native - A framework for building native apps using react

- Angekündigt 2015, inzwischen relativ stabil
- Developer Experience wie im Web
- **Learn once, write everywhere**
- kein HTML, kein Browser, keine WebViews
- Native Components
- Touch Handling
- Style & Layout

Projekt aufsetzen

```
npm install -g react-native-cli  
react-native init MyFirstProject
```

Ergebnis

```
drwxr-xr-x  5 ios
drwxr-xr-x  5 android
drwxr-xr-x  5 node_modules
-rw-r--r--  1 index.android.js
-rw-r--r--  1 index.ios.js
-rw-r--r--  8 package.json
```

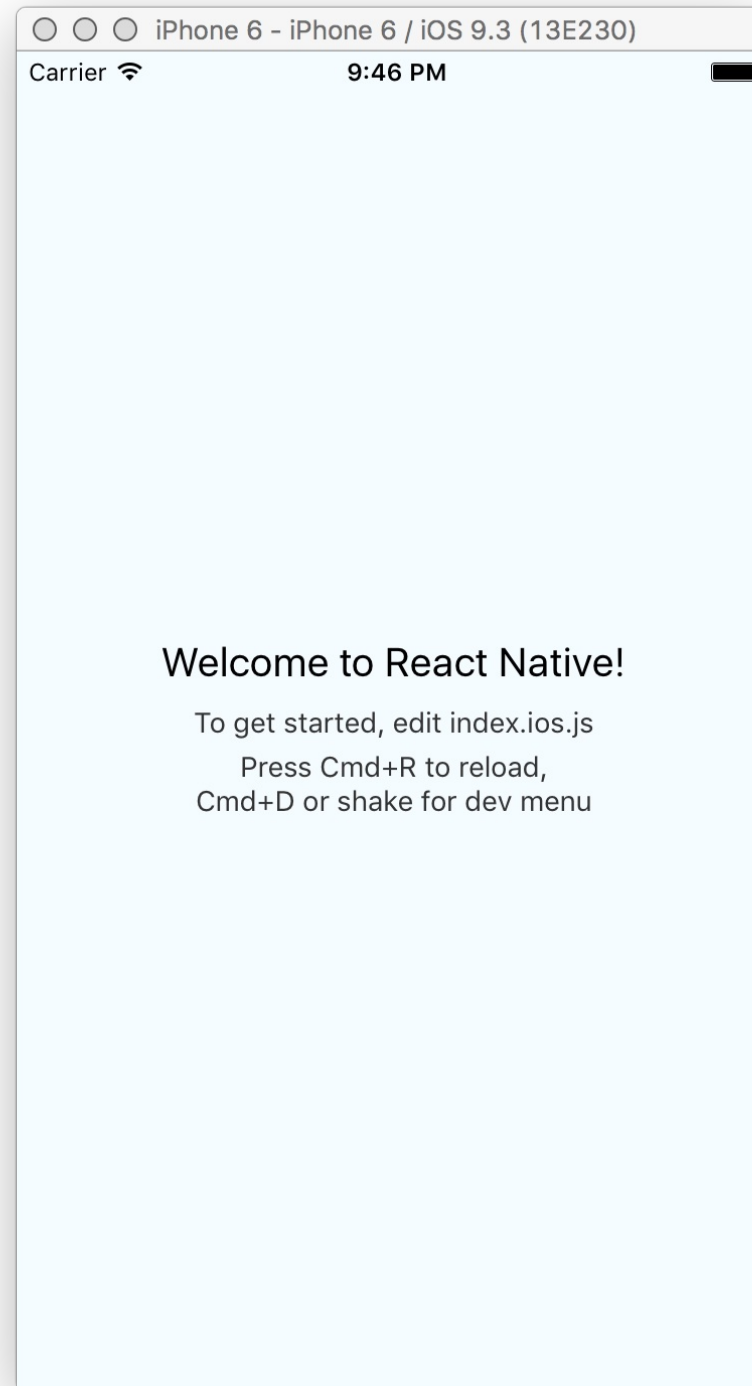
Was ist drinnen?

```
class MyFirstProject extends Component {
  render() {
    return (
      <View style={styles.container}>
        <Text style={styles.welcome}>
          Welcome to React Native!
        </Text>
        <Text style={styles.instructions}>
          To get started, edit index.ios.js
        </Text>
        <Text style={styles.instructions}>
          Press Cmd+R to reload,{'\n'}
          Cmd+D or shake for dev menu
        </Text>
      </View>
    );
  }
}
//...
AppRegistry.registerComponent('MyFirstProject', () => MyFirstProject);
```

Projekt ausführen

```
react-native run-android  
react-native run-ios
```

Hands On



Movie Component in React

```
var Movie = React.createClass({
  propTypes : {
    item : React.PropTypes.object
  },
  render() {
    return (
      <div>
        <span>
          <img src={this.props.item.posters.thumbnail} width="80px" />
        </span>
        <span>{this.props.item.title}</span>
        <span >{this.props.item.year}</span>
      </div>
    );
  }
});
```

Movie Component in React Native

```
var Movie = React.createClass({
  render() {
    return(
      <View style={styles.container}>
        <View style={styles.imageContainer}>
          <Image
            style={styles.poster}
            source={{uri: this.props.item.posters.thumbnail}}
          />
        </View>
        <View style={styles.infoContainer}>
          <Text style={styles.title}>{this.props.item.title}</Text>
          <Text style={styles.info}>{this.props.item.year}</Text>
        </View>
      </View>
    );
  }
});
```


Verwendung der Movie Component

```
<Movie item={{
  title: "Hello EnterJS",
  year: "2016",
  posters: {
    thumbnail: "https://www.enterjs.de/images/location/darmstadtium
  }}></Movie>
```

Styling

```
const styles = StyleSheet.create({
  title: {
    fontSize: 20,
    textAlign: 'left',
    paddingLeft: 8
  },
  year: {
    textAlign: 'left',
    padding: 8
  },
  info: {
    textAlign: 'left',
    padding: 8
  },
  poster: {
    width: 150,
    height: 243,
    borderRadius: 12,
  },
});
```

Flexbox

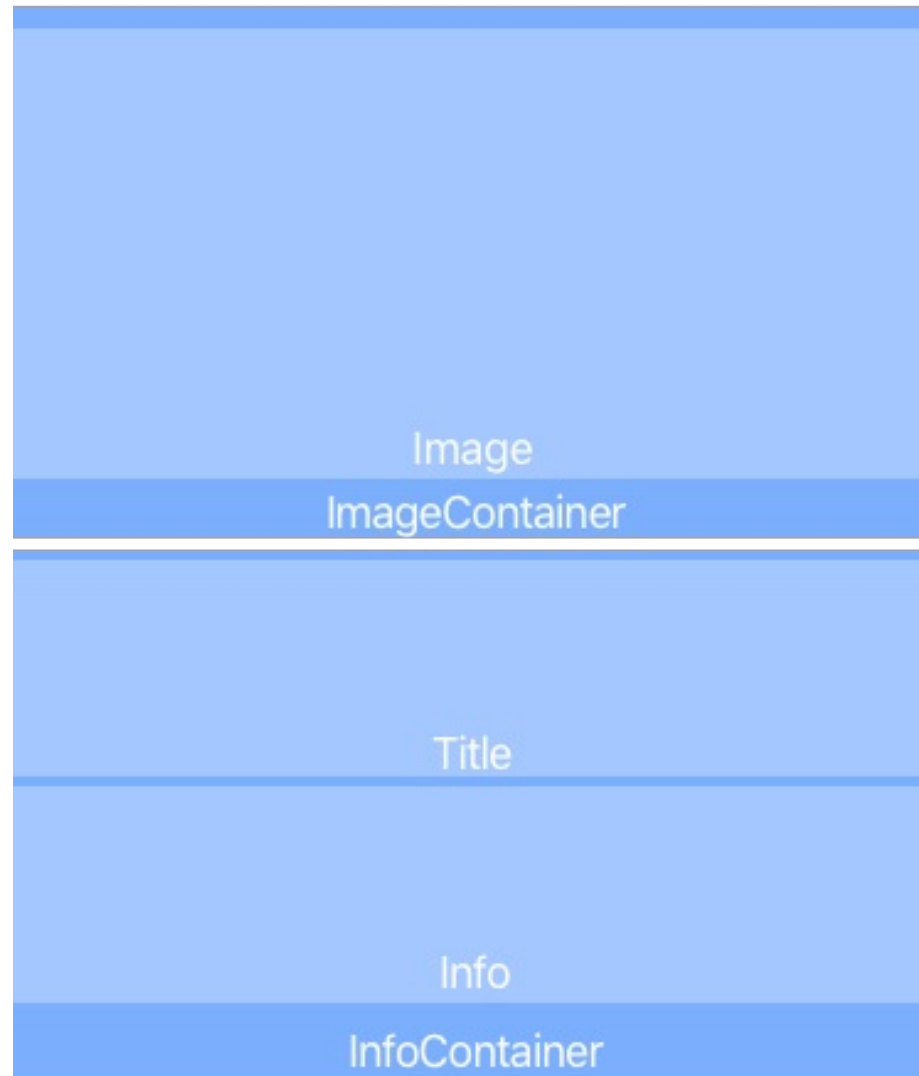
Flexbox

- Flexbox ist ein CSS-Modul, das Elemente in einem Container anordnet

Flexbox

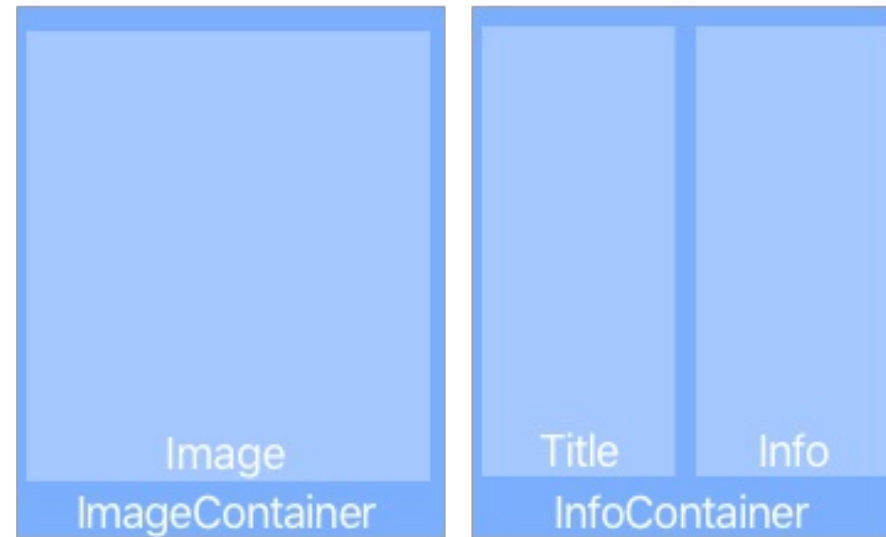
- Flexbox ist ein CSS-Modul, das Elemente in einem Container anordnet
- Flexbox kommt aus dem Response Web Design, um unterschiedliche Bildschirmgrößen effektiv zu nutzen

Flexbox



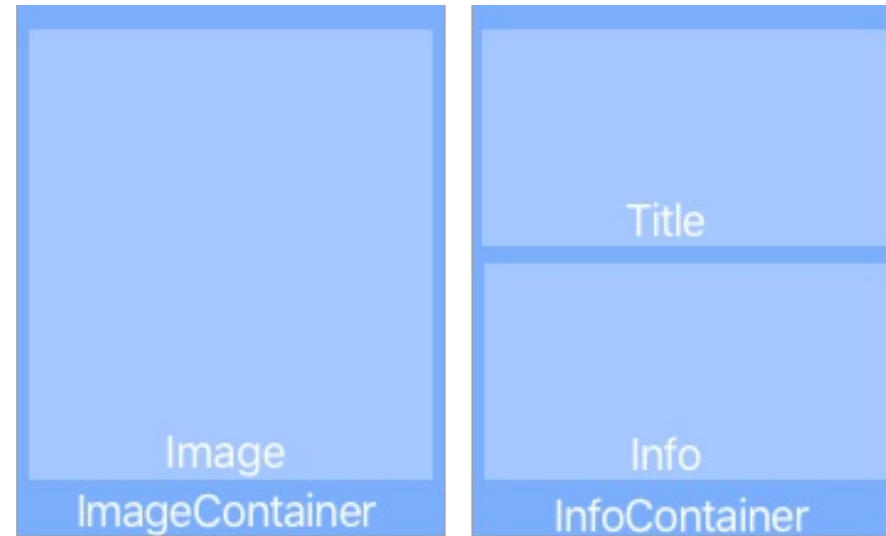
```
container: {  
  flex: 1,  
  flexDirection: 'column',  
  justifyContent: 'center',  
  alignItems: 'center',  
  backgroundColor: '#F5FCFF'  
},
```

Flexbox



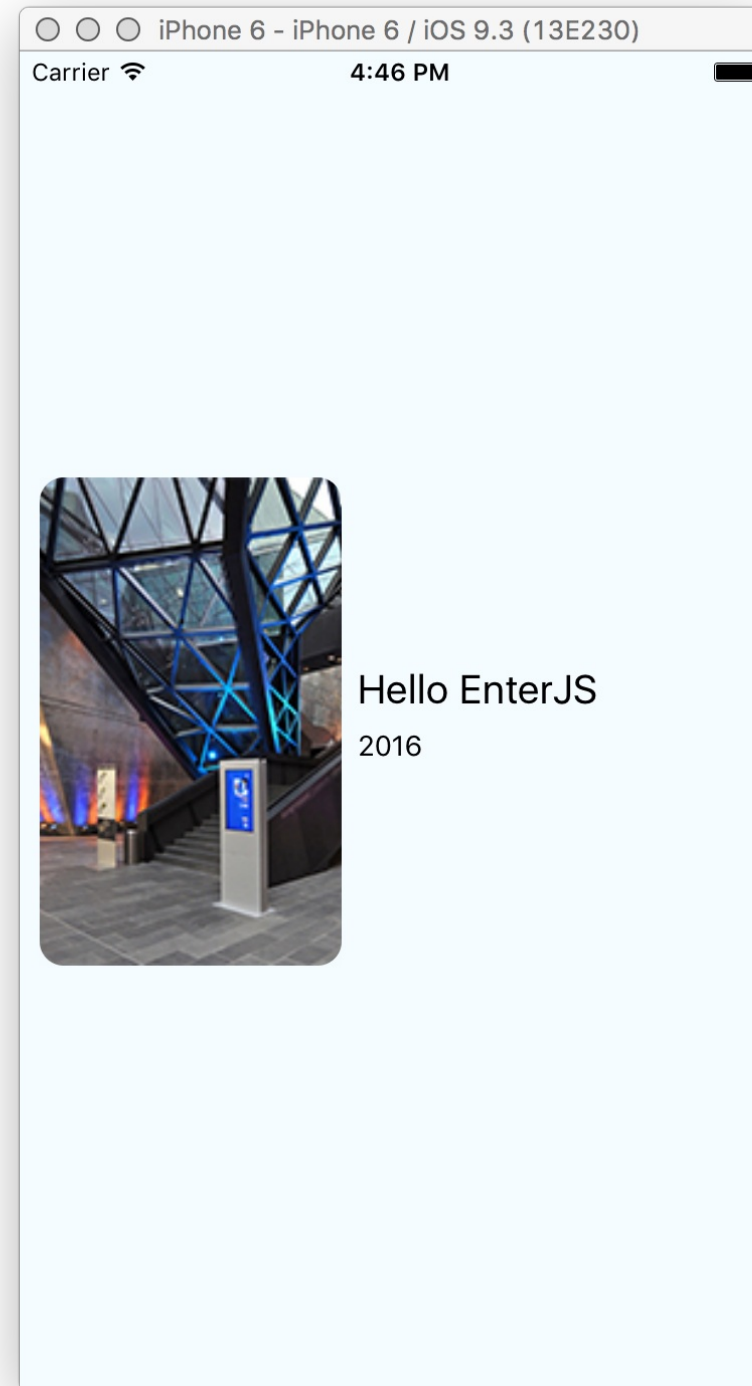
```
container: {  
  flex: 1,  
  flexDirection: 'row',  
  justifyContent: 'center',  
  alignItems: 'center',  
  backgroundColor: '#F5FCFF'  
},  
infoContainer: {  
  flex: 1,  
  flexDirection: 'row'  
},
```

Flexbox



```
container: {  
  flex: 1,  
  flexDirection: 'row',  
  justifyContent: 'center',  
  alignItems: 'center',  
  backgroundColor: '#F5FCFF'  
},  
infoContainer: {  
  flex: 1,  
  flexDirection: 'column'  
},
```


Flexbox Hands On



Performance

Performance

- JavaScript ist single threaded

Performance

- JavaScript ist single threaded
- Rendering vom eigentlichen Applikations-Thread getrennt

Performance

- JavaScript ist single threaded
- Rendering vom eigentlichen Applikations-Thread getrennt
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik

Performance

- JavaScript ist single threaded
- Rendering vom eigentlichen Applikations-Thread getrennt
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Nativer UI-Thread für das Rendern der Komponenten

Performance

- JavaScript ist single threaded
- Rendering vom eigentlichen Applikations-Thread getrennt
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Nativer UI-Thread für das Rendern der Komponenten
- Geschwindigkeit vergleichbar mit "naiv" geschriebenem nativen Code

Performance

- JavaScript ist single threaded
- Rendering vom eigentlichen Applikations-Thread getrennt
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Nativer UI-Thread für das Rendern der Komponenten
- Geschwindigkeit vergleichbar mit "naiv" geschriebenem nativen Code
- Weitere native Threads möglich für langlaufende Berechnungen o.ä.

ListView

ListView

- Kernkomponente

ListView

- Kernkomponente
- Vertikaler Scroll-View

ListView

- Kernkomponente
- Vertikaler Scroll-View
- Veränderliche Daten

ListView

- Kernkomponente
- Vertikaler Scroll-View
- Veränderliche Daten
- Array von Einträgen

ListView

- Kernkomponente
- Vertikaler Scroll-View
- Veränderliche Daten
- Array von Einträgen `ListView.DataSource`

ListView

- Kernkomponente
- Vertikaler Scroll-View
- Veränderliche Daten
- Array von Einträgen `ListView.DataSource`
- `renderRow` ist ein Renderer für einen Eintrag

ListView

- Kernkomponente
- Vertikaler Scroll-View
- Veränderliche Daten
- Array von Einträgen `ListView.DataSource`
- `renderRow` ist ein Renderer für einen Eintrag

```
<ListView  
  dataSource={this.state.dataSource}  
  renderRow={this.renderMovie}  
  style={styles.listView}  
>
```


ListView

```
this.state.dataSource = new ListView.DataSource({
  rowHasChanged: (row1, row2) => row1 !== row2,
})
this.state.dataSource.cloneWithRows([movie1, movie2, movie3, ....])
```

```
renderMovie(movie) => {
  return (
    <View style={styles.container}>
      <Image
        source={{uri: movie.posters.thumbnail}}
        style={styles.thumbnail}
      />
      <View style={styles.rightContainer}>
        <Text style={styles.title}>{movie.title}</Text>
        <Text style={styles.year}>{movie.year}</Text>
      </View>
    </View>
  );
}
```

Navigation

Navigation

- Übergang zwischen Scenes

Navigation

- Übergang zwischen Scenes
- Identifikation der Scene über ein Route-Objekt

Navigation

- Übergang zwischen Scenes
- Identifikation der Scene über ein Route-Objekt
- Navigation mit einem Navigator-Objekt

Navigation

- Übergang zwischen Scenes
- Identifikation der Scene über ein Route-Objekt
- Navigation mit einem Navigator-Objekt

```
<Navigator  
  initialRoute={{name: 'Start', index: 0}}  
  renderScene={this.renderScene}  
>
```

Navigation

- Navigator ist vergleichbar mit einem Stack

Navigation

- Navigator ist vergleichbar mit einem Stack

```
navigator.push({  
  name: 'MovieDetails',  
  index: nextIndex,  
});
```

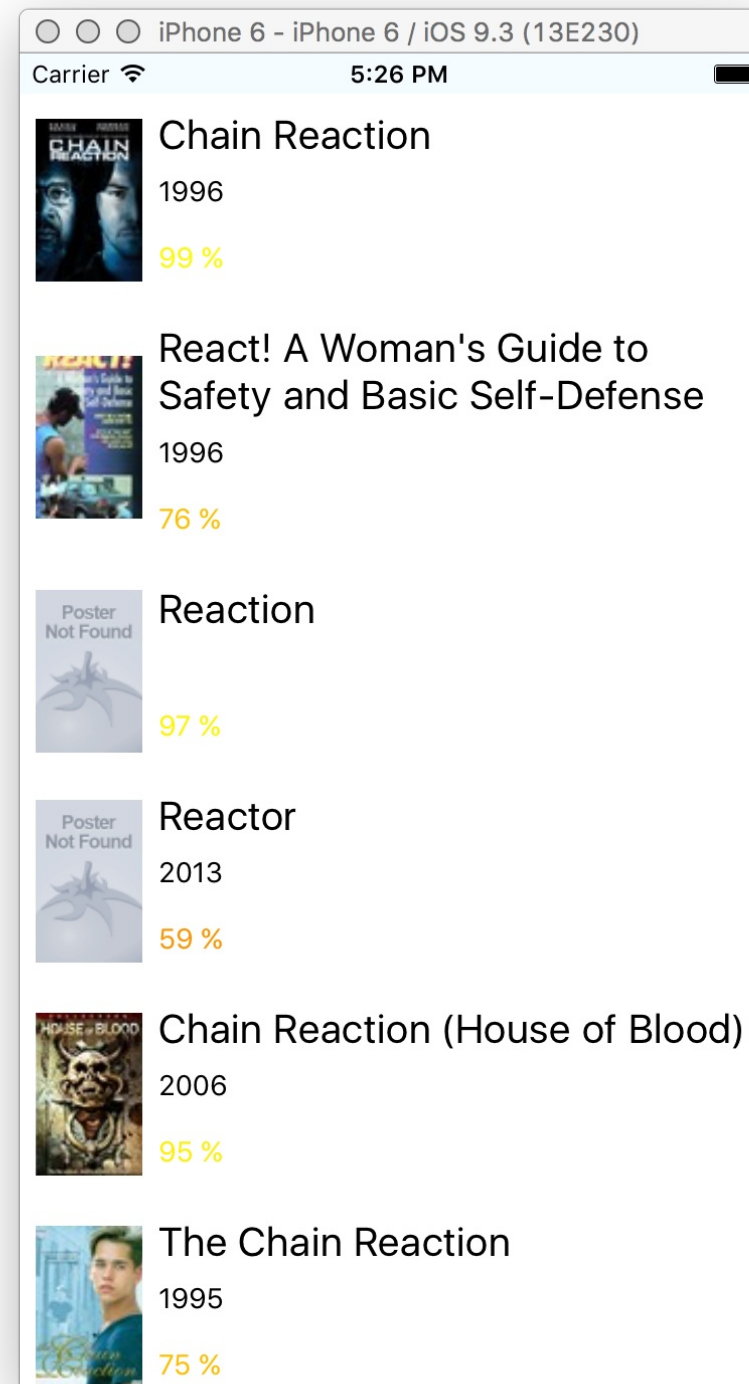

Navigation

- Navigator ist vergleichbar mit einem Stack

```
navigator.push({  
  name: 'MovieDetails',  
  index: nextIndex,  
});
```

```
<NavigationBar  
  title={{ title: 'Detail', tintColor: 'black', }}  
  leftButton={{ title: 'Back', handler: () => navigator.pop(), }}  
  style={{ backgroundColor: "white", }}  
  statusBar={{ tintColor: "white", }}  
</>
```

Code



Plattformspezifische Komponenten

Plattformspezifische Komponenten

```
-rw-r--r--  1  MediaPlayer.ios.js  
-rw-r--r--  1  MediaPlayer.android.js
```

Plattformspezifische Komponenten

```
-rw-r--r--  1  MediaPlayer.ios.js  
-rw-r--r--  1  MediaPlayer.android.js
```

```
import MediaPlayer from './components/MoviePlayer';
```

Plattformspezifischer Code

Plattformspezifischer Code

```
if (React.Platform.OS === 'ios') {  
  // iOS  
} else {  
  // Android  
}
```

Plattformspezifischer Code

```
if (React.Platform.OS === 'ios') {  
  // iOS  
} else {  
  // Android  
}
```

```
React.BackAndroid.addEventListener('hardwareBackPress', () => {  
  if (navigator && navigator.getCurrentRoutes().length > 1) {  
    navigator.pop();  
  }  
});
```


Debugging und Profiling

- Error Reporting

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Lässt sich verlagern in den Chrome-Browser für Chrome DevTools

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Lässt sich verlagern in den Chrome-Browser für Chrome DevTools
- Android:

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Lässt sich verlagern in den Chrome-Browser für Chrome DevTools
- Android: `adb logcat`

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Lässt sich verlagern in den Chrome-Browser für Chrome DevTools
- Android: `adb logcat`
- Logging:

Debugging und Profiling

- Error Reporting
- FPS (Frames per Second) Monitor, Inspector etc.
- JavaScript-Thread (JavaScriptCore) für die Applikations-Logik
- Lässt sich verlagern in den Chrome-Browser für Chrome DevTools
- Android: `adb logcat`
- Logging: `console.warn('Yellow box');`

Weitere Themen

Weitere Themen

- Testing mit Jest

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow
- Managing State mit Redux

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow
- Managing State mit Redux
- Routing mit React Native Router

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow
- Managing State mit Redux
- Routing mit React Native Router
- Push Notifications mit React Native Push Notifications

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow
- Managing State mit Redux
- Routing mit React Native Router
- Push Notifications mit React Native Push Notifications
- Software-Updates via [OTA](#) oder [Firebase](#)

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow
- Managing State mit Redux
- Routing mit React Native Router
- Push Notifications mit React Native Push Notifications
- Software-Updates via Microsoft Code Push oder

Weitere Themen

- Testing mit Jest
- Typisierung mit Flow
- Managing State mit Redux
- Routing mit React Native Router
- Push Notifications mit React Native Push Notifications
- Software-Updates via Microsoft Code Push oder Siphon

Geeignet für Entwicklung?

Geeignet für Entwicklung?

- "Easy things should be easy, and hard things should be possible" - Lary Wall

Geeignet für Entwicklung?

- "Easy things should be easy, and hard things should be possible" - Lary Wall
- Leichte Dinge sind möglich (siehe Beispiel)

Geeignet für Entwicklung?

- "Easy things should be easy, and hard things should be possible" - Larry Wall
- Leichte Dinge sind möglich (siehe Beispiel)
- Für schwierige Dinge können eigene Components (auch nativ) entwickelt werden

Probleme

- Entwicklung stabil nur unter

Probleme

- Entwicklung stabil nur unter OS X

Probleme

- Entwicklung stabil nur unter OS X macOS

Probleme

- Entwicklung stabil nur unter OS X macOS
- Bridging Code selbst schreiben oder suchen

Probleme

- Entwicklung stabil nur unter ~~OS X~~ macOS
- Bridging Code selbst schreiben oder suchen
- 3rd Party Module brechen manchmal

Probleme

- Entwicklung stabil nur unter ~~OS X~~ macOS
- Bridging Code selbst schreiben oder suchen
- 3rd Party Module brechen manchmal
- Handling nativer Libraries manchmal kompliziert

Probleme

- Entwicklung stabil nur unter ~~OS X~~ macOS
- Bridging Code selbst schreiben oder suchen
- 3rd Party Module brechen manchmal
- Handling nativer Libraries manchmal kompliziert
([RNMP](#) to the rescue!)

Probleme

- Entwicklung stabil nur unter ~~OS X~~ macOS
- Bridging Code selbst schreiben oder suchen
- 3rd Party Module brechen manchmal
- Handling nativer Libraries manchmal kompliziert
([RNMP](#) to the rescue!)
- Platform Parity

Probleme

- Entwicklung stabil nur unter ~~OS X~~ macOS
- Bridging Code selbst schreiben oder suchen
- 3rd Party Module brechen manchmal
- Handling nativer Libraries manchmal kompliziert
([RNMP](#) to the rescue!)
- Platform Parity
- Leaky Abstraction Layer

Alternativen

Alternativen

■ Stop!

Alternativen

- Stop! Wir entwickeln nicht mit React, sondern mit Angular?!

Alternativen

- Stop! Wir entwickeln nicht mit React, sondern mit Angular?!
- Telerik Native Script

Alternativen

- Stop! Wir entwickeln nicht mit React, sondern mit Angular?!
- Telerik Native Script

```
@Component({
  selector: "my-app",
  template: `
    <TextField hint="Email Address" keyboardType="email"
      autocorrect="false" autocapitalizationType="none"></TextField>
    <TextField hint="Password" secure="true"></TextField>
    <Button text="Sign in"></Button>
    <Button text="Sign up for Groceries"></Button>
  `
})
```

Quellen

Slides: holisticon.github.io/presentations/enterjs-react-native

Beispiele: github.com/simonox/enterjs-react-native

oliver.ochs@holisticon.de

www.holisticon.de

 [@oochs](https://twitter.com/oochs)