LECTURE 5

OPTIMIZATION, DATA MANIPULATION, LIFE CYCLE

DANIEL RYS
JAN VÁCLAVÍK

OVERVIEW

- Optimization
- Callbacks
- Promises
- Angular resource ($resource)
- Life cycle
- Generate splash screen and icons
- Google Analytics

OPTIMIZATION

OPTIMIZATION

Who noticed that the homework was slow?

OPTIMIZATION

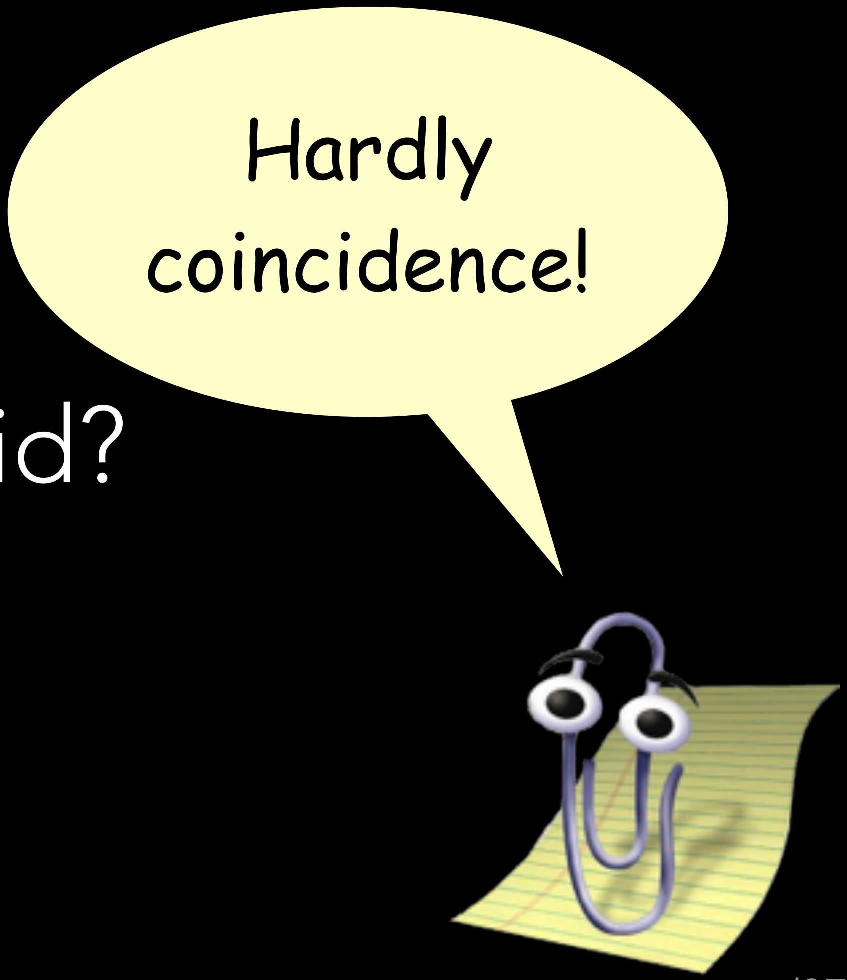
Who noticed that the homework was slow?

Who of you has Android?

OPTIMIZATION

Who noticed that the homework was slow?

Who of you has Android?



WHAT WE CAN OPTIMIZE

- Native scroll
- Collection repeat
- Track by
- One time data binding
- Crosswalk

WHAT WE CAN OPTIMIZE

- Native scroll ← Android only
- Collection repeat
- Track by
- One time data binding
- Crosswalk

WHAT WE CAN OPTIMIZE

- Native scroll
- Collection repeat
- Track by
- One time data binding
- Crosswalk

Not works with native scroll and one time data binding

WHAT WE CAN OPTIMIZE

- Native scroll
- Collection repeat
- Track by

Works everywhere

- One time data binding
- Crosswalk

WHAT WE CAN OPTIMIZE

- Native scroll
- Collection repeat
- Track by
- One time data binding ← Works everywhere
- Crosswalk

WHAT WE CAN OPTIMIZE

- Native scroll
- Collection repeat
- Track by
- One time data binding
- Crosswalk ← Android only

NATIVE SCROLL

- Greatly **increasing scrolling smoothness**
- Works only for Android platform
- For enable native scroll, disable Javascript scrolling

SET NATIVE SCROLL GLOBALLY

www/app/routes.js

```
app.config(function($ionicConfigProvider,...){
    ...
    $ionicConfigProvider.scrolling.jsScrolling(false);
    ...
}
```

SET NATIVE SCROLL FOR VIEW

```
<ion-content overflow-scroll="true">
    ...
</ion-content>
```

COLLECTION REPEAT

- Collection-repeat allows an app **to show huge lists of items** much more performantly than ng-repeat
- Renders into the DOM only as many items as are currently visible
- **Doesn't have native scroll**
- Only for arrays
- Fixed height
- Inspired by iOS's UITableView

COLLECTION REPEAT

```
<ion-list>
    <ion-item collection-repeat="movie in data.movies">
        {{movie.title}}
    </ion-item>
</ion-list>
```

TRACK BY

Use for avoid useless DOM manipulation

```
<ion-list>
    <ion-item ng-repeat="movie in movies track by movie.id">
        {{movie.title}}
    </ion-item>

    <ion-item ng-repeat="movie in movies track by $index">
        {{movie.title}}
    </ion-item>
</ion-list>
```

TRACK BY

Use for avoid useless DOM manipulation

If items have unique ID

```
<ion-list>
    <ion-item ng-repeat="movie in movies track by movie.id">
        {{movie.title}}
    </ion-item>

    <ion-item ng-repeat="movie in movies track by $index">
        {{movie.title}}
    </ion-item>
</ion-list>
```

TRACK BY

Use for avoid useless DOM manipulation

If items have unique ID

```
<ion-list>
    <ion-item ng-repeat="movie in movies track by movie.id">
        {{movie.title}}
    </ion-item>

    <ion-item ng-repeat="movie in movies track by $index">
        {{movie.title}}
    </ion-item>
</ion-list>
```

If item's dont have unique ID

ONE TIME DATA BINDING

- Data does not get synchronized from view to controller
- Don't use angular one-time binding with collection-repeat

```
<ion-list>
    <ion-item ng-repeat="movie in movies">
        {{::movie.title}}
    </ion-item>
</ion-list>
```

"::" is where the magic happens

CROSSWALK

CROSSWALK

- **Replacing native browser with Chromium–based**
- Useful for older Android devices with Android browser (Android < 4.4)
- **Significantly better performance** and CSS support

https://crosswalk-project.org

ADD/REMOVE CROSSWALK

```
$ ionic browser add crosswalk
$ ionic browser rm crosswalk
```

CALLBACKS

CALLBACKS

- Function
- Passed as argument to another function
- Called after some event or action

CALLBACKS

```
function getMoviesFromApi(callback){
  // Do something
  callback();
}

getMoviesFromApi(function(){
  console.log("Hello!")
})
```

CALLBACKS

```
function getMoviesFromApi(callback){
  // Do something
  callback();
}

getMoviesFromApi(function(){
  console.log("Hello!")
})
```

Anonymous callback

```
function(){
  console.log("Hello!")
}
```

CALLBACKS

```
function getMoviesFromApi(callback){
  // Do something
  callback();
}

getMoviesFromApi(function(){
  console.log("Hello!")
})
```

Call callback function

Anonymous callback

```
function(){
  console.log("Hello!")
}
```

CALLBACKS

```
function getMoviesFromApi(callback){
  // Do something
  callback();
}

function sayHello(){
  console.log("Hello!")
}

getMoviesFromApi(sayHello)
```

CALLBACKS

```
function getMoviesFromApi(callback){
  // Do something
  callback();
}

function sayHello(){
  console.log("Hello!")
}

getMoviesFromApi(sayHello)
```

Named function callback

```
sayHello
```

CALLBACKS

```
function getMoviesFromApi(callback){
  // Do something
  var name = "Katka";
  callback(name);
}

function sayHelloTo(somebody){
  console.log("Hello!" + somebody)
}

getMoviesFromApi(sayHello)
```

CALLBACKS

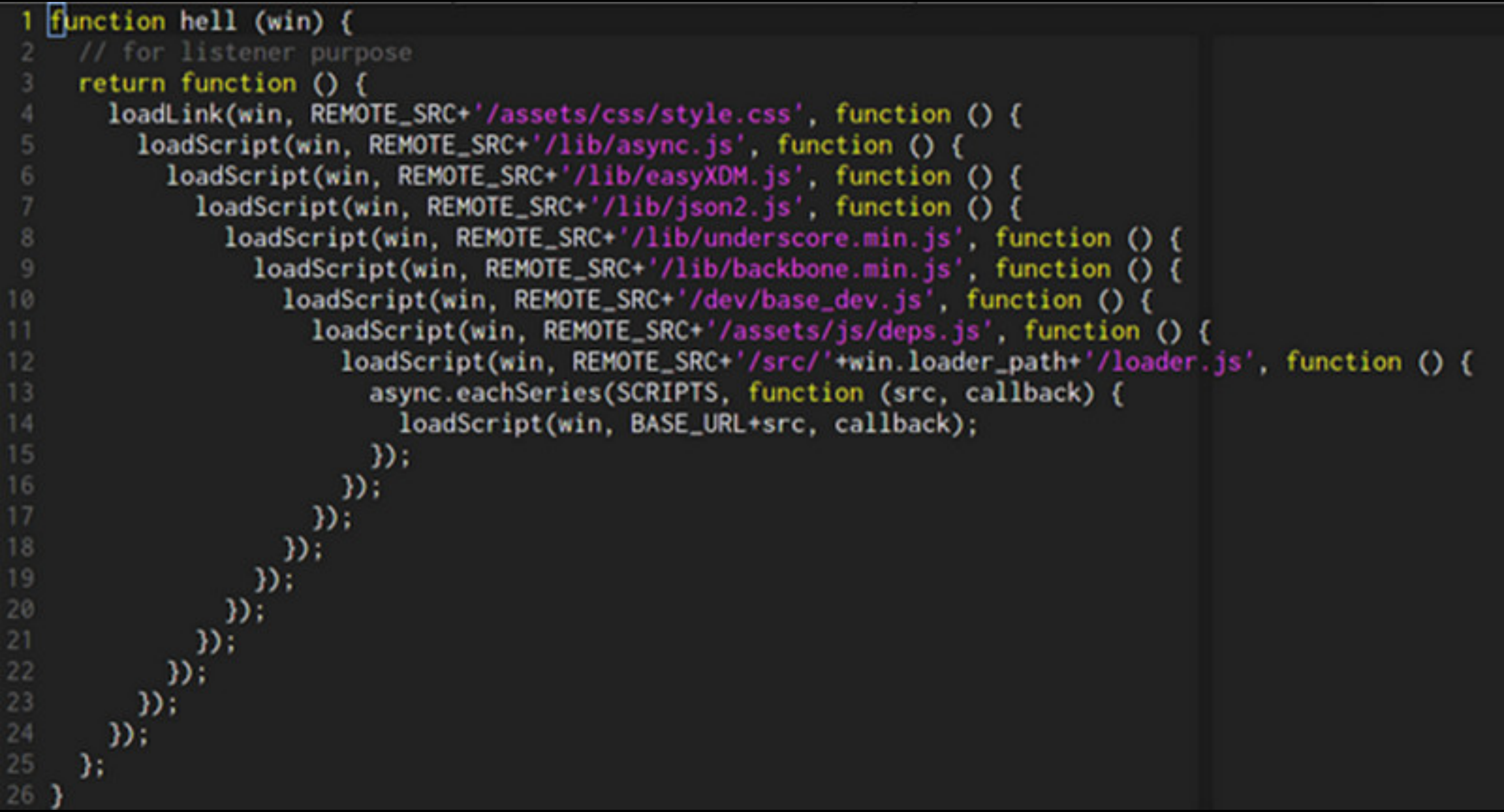
```
function getMoviesFromApi(callback){
  // Do something
  var name = "Katka";
  callback(name);
}

function sayHelloTo(somebody){
  console.log("Hello!" + somebody)
}

getMoviesFromApi(sayHello)
```

Callback
with parameter

CALLBACK HELL



```javascript
function hell (win) {
  // for listener purpose
  return function () {
    loadLink(win, REMOTE_SRC+'/assets/css/style.css', function () {
      loadScript(win, REMOTE_SRC+'/lib/async.js', function () {
        loadScript(win, REMOTE_SRC+'/lib/easyXDM.js', function () {
          loadScript(win, REMOTE_SRC+'/lib/json2.js', function () {
            loadScript(win, REMOTE_SRC+'/lib/underscore.min.js', function () {
              loadScript(win, REMOTE_SRC+'/lib/backbone.min.js', function () {
                loadScript(win, REMOTE_SRC+'/dev/base_dev.js', function () {
                  loadScript(win, REMOTE_SRC+'/assets/js/deps.js', function () {
                    loadScript(win, REMOTE_SRC+'/src/'+win.loader_path+'/loader.js', function () {
                      async.eachSeries(SCRIPTS, function (src, callback) {
                        loadScript(win, BASE_URL+src, callback);
                      });
                    });
                  });
                });
              });
            });
          });
        });
      });
    });
  };
}
```

PROMISES

PROMISES

- Object for execution of **asynchronous code**
- Represents an operation which is not completed yet (in EcmaScript 5 standard), but it’s expected
- **3 states** – Pending, Fulfilled (resolved), Rejected

STATES IN PROMISE

- **Pending**
  - In progress
  - Initial state
- **Fulfilled**
  - Finished with success
- **Rejected**
  - Finished without success

PROMISE EXAMPLE (ANGULAR)

```
var promise = $http(request);

var success = function (response) {
  service.data.movies = response.results
};

var error = function (response) {
  console.error("error");
}

promise.then(success, error);
```

PROMISE EXAMPLE (ANGULAR)

```
var promise = $http(request);

var success = function (response) {
  service.data.movies = response.results
};

var error = function (response) {
  console.error("error");
}

promise.then(success, error);
```

"Then" calls callback when request is complete. Returns promise.

PROMISE EXAMPLE (ANGULAR)

- **Then()**
  - Called when request is complete with success
- **Catch()**
  - Called when request is complete without success
- **Finally()**
  - Called when request is complete

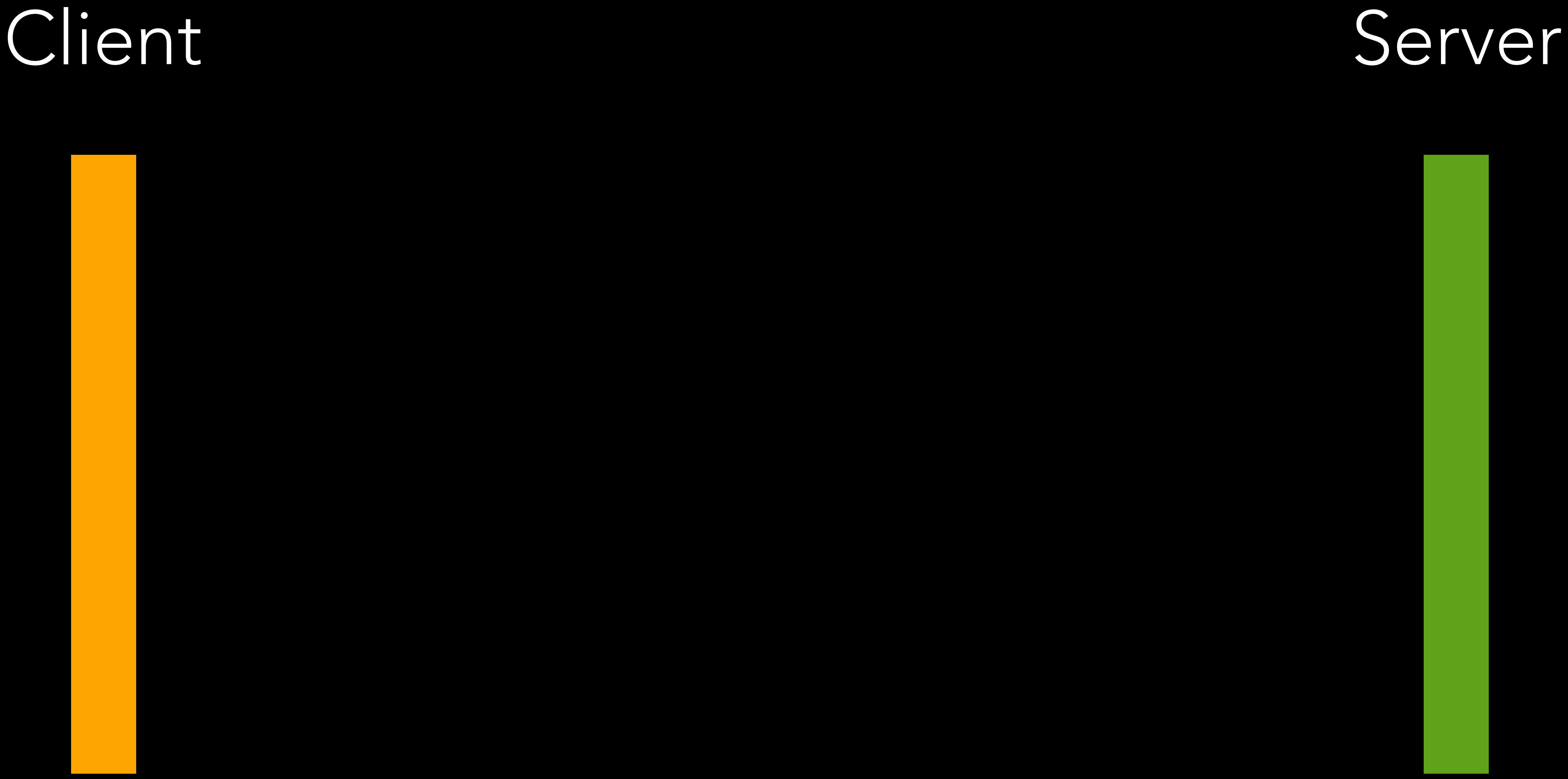
PROMISES VS. CALLBACKS

PROMISES VS. CALLBACKS

- **Promise**
  - Object
  - Mainly for asynchronous operations
  - Chaining
- **Callback**
  - Function
  - Both synchronous/asynchronous operations
  - Nesting

HTTP REQUEST/RESPONSE

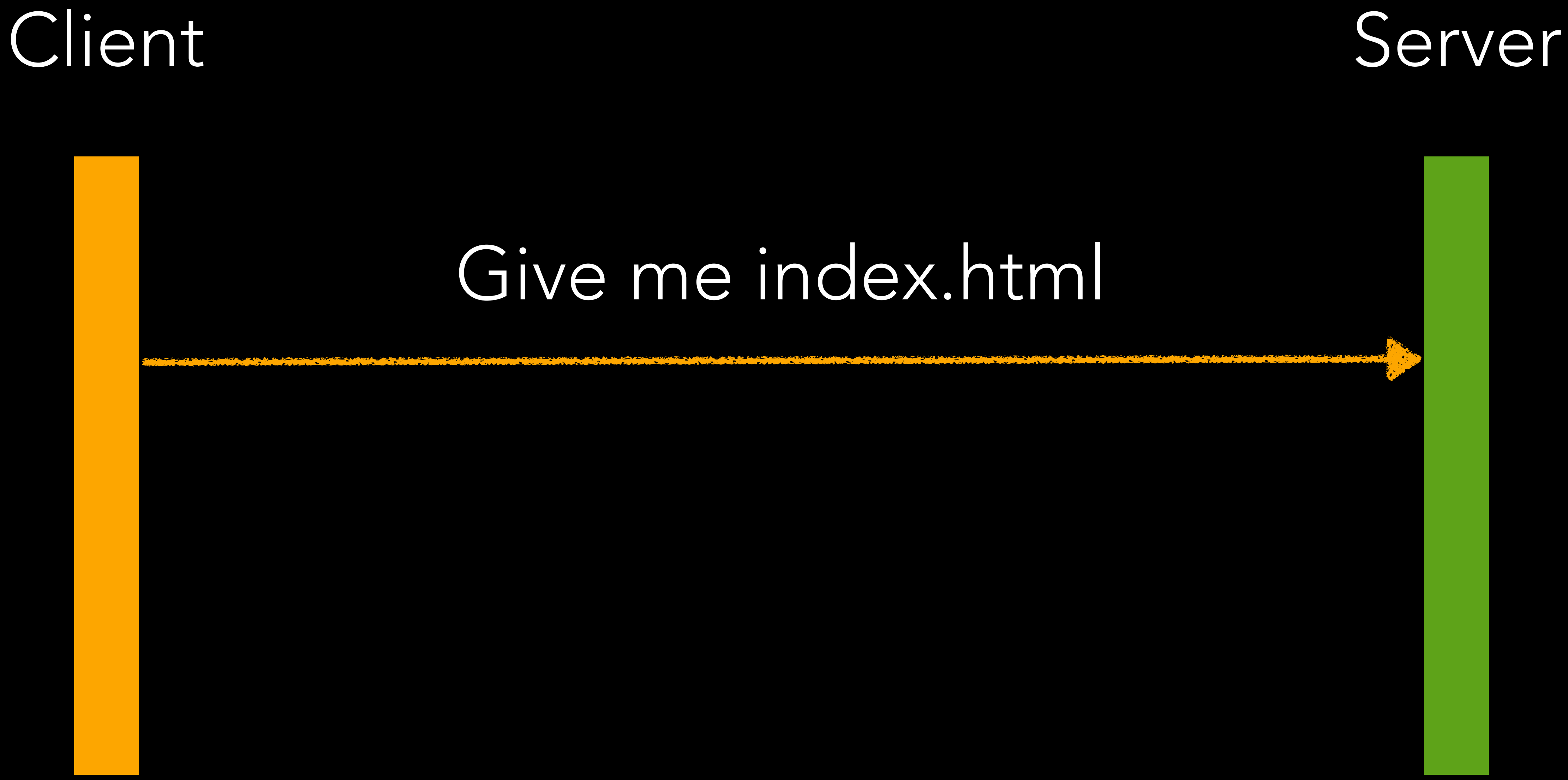
HTTP REQUEST – RESPONSE



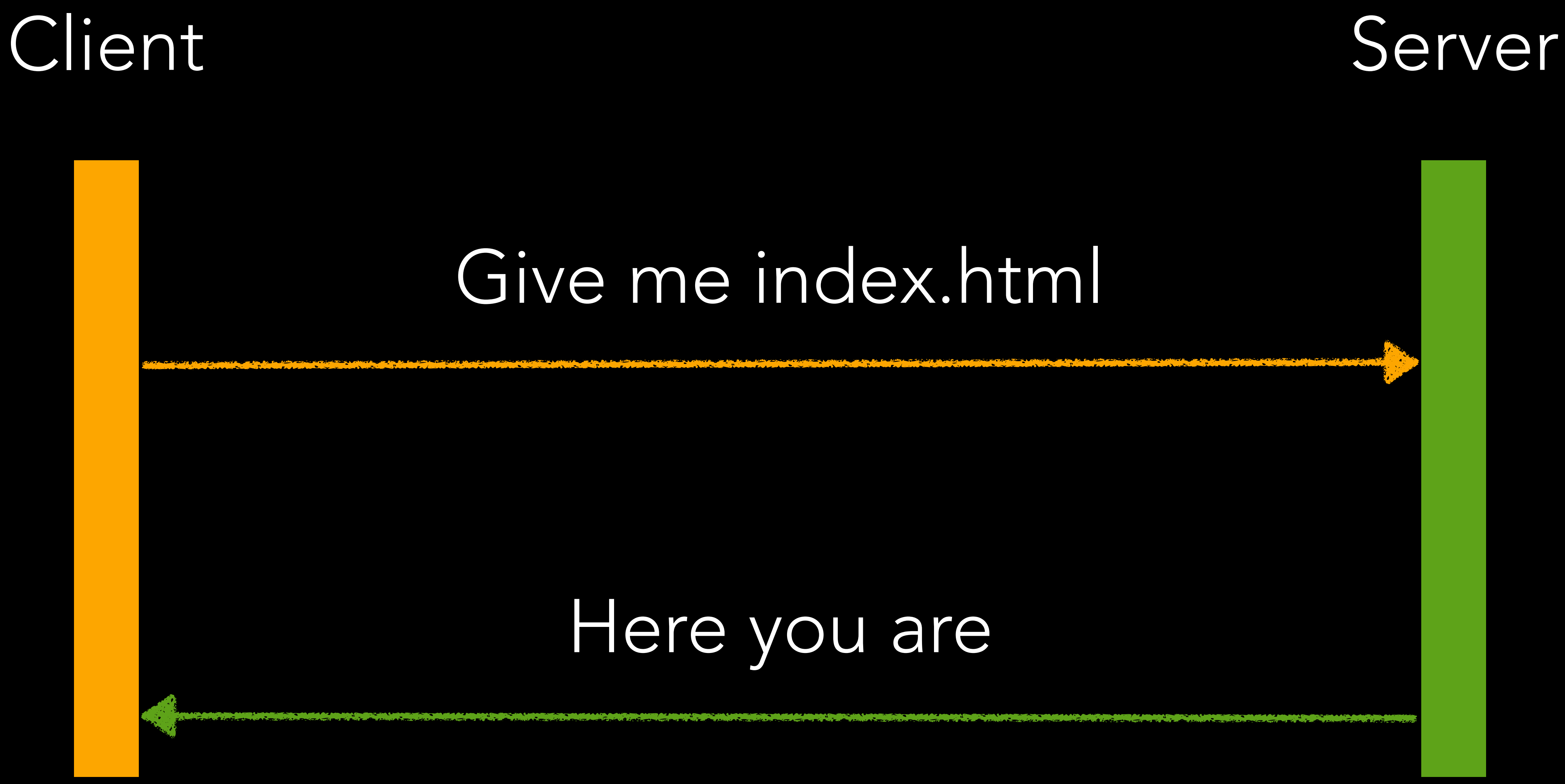
Client

Server

HTTP REQUEST – RESPONSE



HTTP REQUEST – RESPONSE



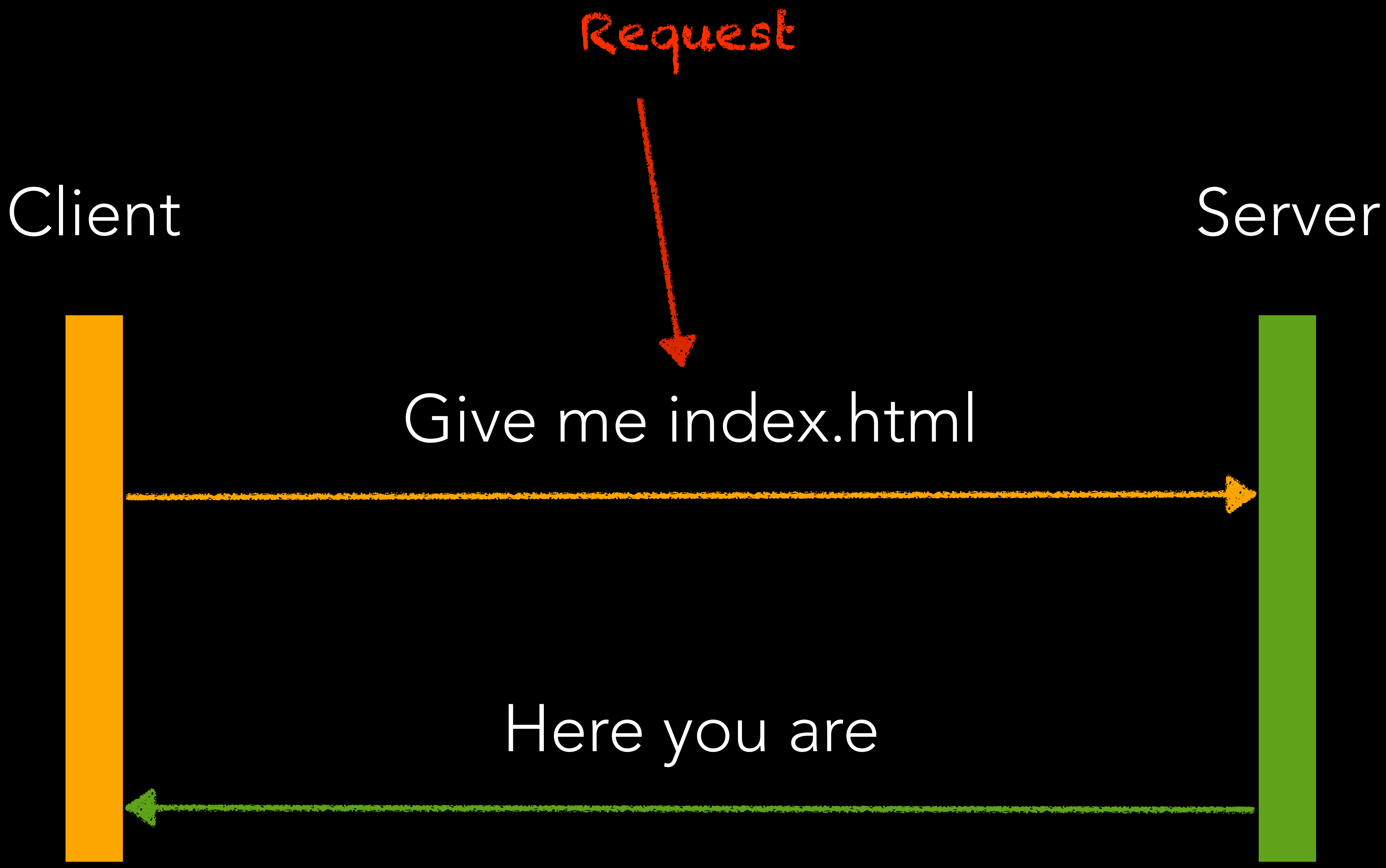
Client

Server

Give me index.html

Here you are

HTTP REQUEST – RESPONSE



Request

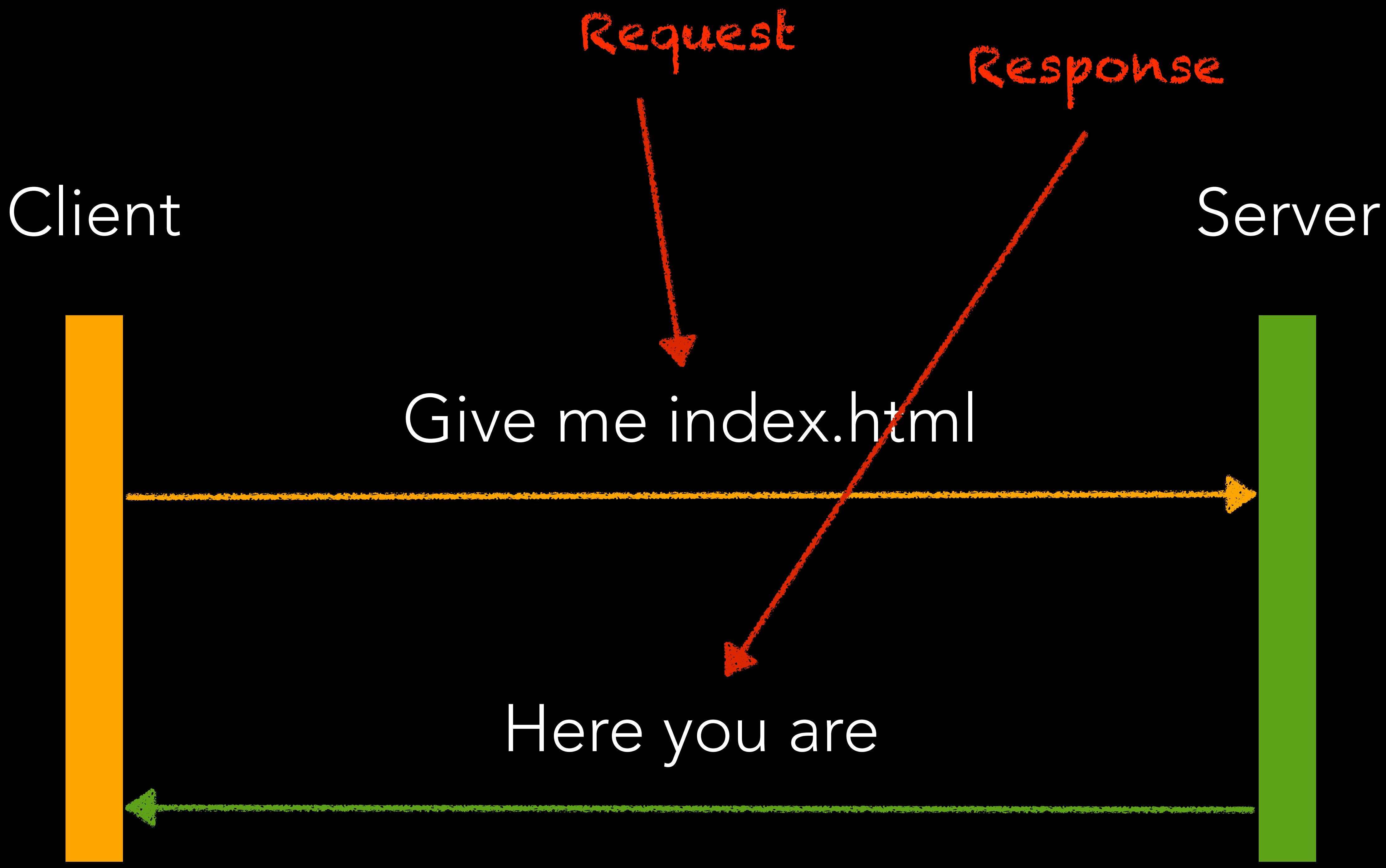
Client

Server

Give me index.html

Here you are

HTTP REQUEST – RESPONSE



HTTP REQUEST

- Client requests for some data
- Headers
  - Many formats: Content type (JSON, XML, form data, …)
- Body
- Many types: GET, POST, PUT, DELETE,…

HTTP RESPONSE

- Server gives us data in specified format
- **Headers**
  - Meta information
- **Body**
  - Data which we requested

REST PRINCIPLE

- **URL** – some entity (data resource) we want to interact with
- **Methods** – what we want to do with entity
- **Example 1**: I want to get movies
  - **URL**: http://movies.com/api/movies
  - **Method**: GET

HTTP METHODS IN REST

- Methods used for specific operations
  - **GET** – get some data (request is without body)
  - **POST** – create new data
  - **PUT** – update some data
  - **DELETE** – delete some data

REST EXAMPLES

- **Example 2**: I want to get a single movie with ID 6
  - **URL**: http://movies.com/api/movies/6
  - **Method**: GET

REST EXAMPLES

- **Example 3**: I want to add rating for movie with ID 6
  - **URL**: http://movies.com/api/movies/6/rating
  - **Method**: POST

REST EXAMPLES

- **Example 4**: I want update movie info with ID 6
  - **URL**: http://movies.com/api/movies/6
  - **Method**: PUT

REST EXAMPLES

- **Example 5**: I want delete movie info with ID 6
  - **URL**: http://movies.com/api/movies/6
  - **Method**: DELETE

HTTP STATUS CODES

- **Every response returns status code**
- Next operations can be based on returned status code
- Numbers between 100 - 505

HTTP STATUS CODES

- **1..** – Informational
- **2..** – Success
- **3..** – Redirect
- **4..** – Client error
- **5..** – Server error

ANGULAR RESOURCE ($RESOURCE)

$RESOURCE

- Designed to use REST API
- Angular object for manipulation with data
- Has predefined methods mapped to REST operations
- More abstract than $http (produce cleaner code)

$RESOURCE METHODS

- get (GET one)
- save (POST)
- query (GET all)
- delete (DELETE)

INCLUDE $RESOURCE

www/app/app.js

```
var app = angular.module('App', ['ionic', 'ui.router', 'ngResource']);
```

$RESOURCE EXAMPLE

- **Example 1**: I want to get movies
  - **URL**: http://movies.com/api/movies
  - **Method**: GET

```
$resource("http://movies.com/api/movies").query();
```

$RESOURCE EXAMPLE

- **Example 2**: I want to get a single movie with ID 6
  - **URL**: http://movies.com/api/movies/6
  - **Method**: GET

```
$resource(
  "http://movies.com/api/movies/:movieId",
  {movieId: "@movieId"}
).get({movieId: 6});
```

$RESOURCE EXAMPLE

- **Example 3**: I want to add rating for movie with ID 6
  - **URL**: http://movies.com/api/movies/6/rating
  - **Method**: POST

```
$resource(
  "http://movies.com/api/movies/:movieId/rating",
  {movieId: "@movieId"}
).save(
  {movieId: 6},
  body
);
```

$RESOURCE EXAMPLE

- **Example 4**: I want update movie info with ID 6
  - **URL**: http://movies.com/api/movies/6
  - **Method**: PUT

```
$resource(
  "http://movies.com/api/movies/:movieId",
  {movieId: "@movieId"
}, {
  update: {method: "PUT"}
}).update(
  {movieId: 6},
  body
);
```

$RESOURCE EXAMPLE

- **Example 5**: I want delete movie info with ID 6
  - **URL**: http://movies.com/api/movies/6
  - **Method**: DELETE

```
$resource(
  "http://movies.com/api/movies/:movieId",
  {movieId: "@movieId"}
).delete({movieId: 6});
```

LIFE CYCLE

LIFE CYCLE

- Controller is loaded only once by default
- We can keep track of view state using predefined events

BROADCAST

```
$scope.$on("someEvent", function (args) {

});

$scope.$broadcast("someEvent", args);
```

BROADCAST

```
$scope.$on("someEvent", function (args) {

});


$scope.$broadcast("someEvent", args);
```

Call event by broadcast

BROADCAST

Called every time someEvent is broadcasted

```
$scope.$on("someEvent", function (args) {

});


$scope.$broadcast("someEvent", args);
```

Call event by broadcast

LIFE CYCLE

- Broadcast automatically by Ionic

```
$scope.$on("$ionicView.enter", function () {

});
```

LIFE CYCLE

- **loaded** – only once, when DOM of View is created
- **enter** – every time when user enter View
- **leave** – every time when user leave View
- **beforeEnter** – every time before showing transition
- **beforeLeave** – before leave transition
- **afterEnter** – after transition has finished and View is visible and active
- **afterLeave** – after transition has finished and View is hidden and inactive
- **unloaded** – View has been destroyed and its element has been removed from DOM

GENERATE SPLASH SCREEN AND ICONS

GENERATE SPLASH SCREEN AND ICONS

- Prepare two PNG / PSD images
  - **Icon** (at least 192 x 192px) without rounded corners
  - **Splash screen** (minimum dimensions should be 2208 x 2208px)

GENERATE SPLASH SCREEN AND ICONS

- Copy them to following paths
  - **www/resources/icon.png**
  - **www/resources/splash.png**

Generate resources

```
$ ionic resources
```

GOOGLE ANALYTICS

GOOGLE ANALYTICS

Register as website on

https://analytics.google.com

ADD GOOGLE ANALYTICS TO PROJECT

```
$ bower install ga-localstorage -S
```

https://github.com/ggendre/GALocalStorage

ADD TO GULPFILE.ES6.JS

gulpfile.es6.js

```
…
var bowerLibs = {
    js: [
        …
        'ga-localstorage/GALocalStorage.js'
    ],
…
```

START GULP TASK FOR MINIMIZE

```
$ gulp bowerlibs
```

SET GOOGLE ANALYTICS ACCOUNT

www/app/app.js

```
$ionicPlatform.ready(function (...) {
    ga_storage._setAccount('UA-12345678-1');
});
```

TRACK VIEW IN CONTROLLER

```
$scope.$on('$ionicView.enter', function () {
    ga_storage._trackPageview("/tutorial", "Tutorial");
});
```

HOMEWORK

$resource, Google Maps

http://bit.ly/25GoJv5

QUESTIONS?

U+_

JAN VÁCLAVÍK

@janvaclavik

DANIEL RYS

@danielrys

WWW.USERTECHNOLOGIES.COM