LECTURE 7

INTRODUCTION TO IONIC 2

DANIEL RYS
JAN VÁCLAVÍK

OVERVIEW

- Create new Ionic2 application
- Why Ionic2
- Project structure
- Features

INSTALL IONIC 2 & CREATE NEW APPLICATION

```
$ npm install -g ionic@beta
$ ionic start my-first-ionic2-app [tabs|sidemenu|blank] --v2 [--ts]
$ ionic serve
```

INSTALL IONIC 2 & CREATE NEW APPLICATION

Ionic version 2

```
$ npm install -g ionic@beta
$ ionic start my-first-ionic2-app [tabs|sidemenu|blank] --v2 [--ts]
$ ionic serve
```

INSTALL IONIC 2 & CREATE NEW APPLICATION

Create TypeScript project

Ionic version 2

```
$ npm install -g ionic@beta
$ ionic start my-first-ionic2-app [tabs|sidemenu|blank] --v2 [--ts]
$ ionic serve
```

WHY IONIC2

- Component-based model
- Angular2
- ES6 / TypeScript (transpiling)
- Windows 10 support, Android Material Design
- Generator
- Better theming & performance
- More scalable structure


I can do
all this through
who gives me strength
Philippians 4:13

ECMASCRIPT 6 / TYPESCRIPT

- Classes (OOP)
- Fat arrow (=>)
- Promises
- Components (modules)
- Block scoping

CLASSES

- Mapping real world object into the abstract

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
}
```

CLASSES

- Mapping real world object into the abstract

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
}
```

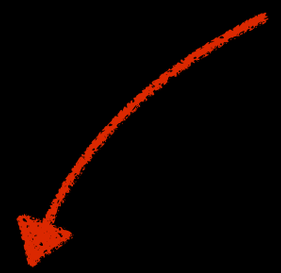
Constructor is called when creating instance of class

FAT ARROW FUNCTIONS

```
someFunction(function(response){
    console.log(response);
});
```

FAT ARROW FUNCTIONS

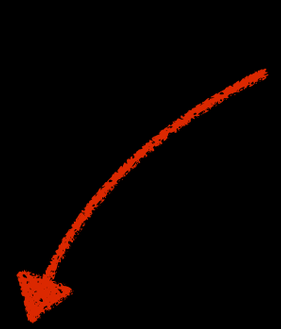
Has local context



```
someFunction(function(response){
    console.log(response);
});
```

FAT ARROW FUNCTIONS

Has local context



```
someFunction(function(response){
    console.log(response);
});

someFunction((response) => {
    console.log(response);
});
```

FAT ARROW FUNCTIONS

Has local context

```
someFunction(function(response){
    console.log(response);
});

someFunction((response) => {
    console.log(response);
});
```

Has parent context

PROMISES

```
doSomething().then((response) => {
    console.log(response);
});
```

For more details see Lecture 5

IMPORT & EXPORT COMPONENTS

```
//  lib/math.js
export function sum (x, y) {return x + y}
export var pi = 3.141593


//  someApp.js
import * as math from "lib/math"
console.log("2PI = " + math.sum(math.pi, math.pi))


//  otherApp.js
import {sum, pi} from "lib/math"
console.log("2PI = " + sum(pi, pi))
```

IMPORT & EXPORT COMPONENTS

```
//  lib/math.js
export function sum (x, y) {return x + y}
export var pi = 3.141593
```

Specify visible functions, vars

```
//  someApp.js
import * as math from "lib/math"
console.log("2PI = " + math.sum(math.pi, math.pi))


//  otherApp.js
import {sum, pi} from "lib/math"
console.log("2PI = " + sum(pi, pi))
```

IMPORT & EXPORT COMPONENTS

```
//  lib/math.js
export function sum (x, y) {return x + y}
export var pi = 3.141593
```

Specify visible functions, vars

```
//  someApp.js
import * as math from "lib/math"
console.log("2PI = " + math.sum(math.pi, math.pi))
```

```
//  otherApp.js
import {sum, pi} from "lib/math"
console.log("2PI = " + sum(pi, pi))
```

Can call functions from components

IMPORT & EXPORT COMPONENTS

```
//  lib/math.js
export function sum (x, y) {return x + y}
export var pi = 3.141593
```

Specify visible functions, vars

```
//  someApp.js
import * as math from "lib/math"
console.log("2PI = " + math.sum(math.pi, math.pi))
```

```
//  otherApp.js
import {sum, pi} from "lib/math"
console.log("2PI = " + sum(pi, pi))
```

Can call functions from components

IMPORT & EXPORT COMPONENTS IN IONIC2

```
import {Page} from 'ionic-angular';
import {MoviesPage} from '../movies/movies';
```

IMPORT & EXPORT COMPONENTS IN IONIC2


```
import {Page} from 'ionic-angular';
import {MoviesPage} from '../movies/movies';
```

BLOCK SCOPING

```
for (let i = 0; i < movies.length; i++) {
    let x = movies[i];
}
console.log(x);
```

BLOCK SCOPING

```
for (let i = 0; i < movies.length; i++) {
    let x = movies[i];
}
console.log(x); // Undefined
```

TYPESCRIPT EXAMPLE

```
function add(x : number, y : number) : number {
    return x + y;
}
add('a', 'b');
```

TYPESCRIPT EXAMPLE

```
function add(x : number, y : number) : number {
    return x + y;
}
add('a', 'b'); // Compiler error
```

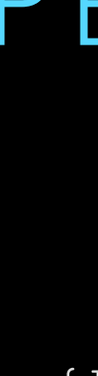
DEPENDENCY INJECTION

DEPENDENCY INJECTION

```
import {Page, NavController, Platform} from 'ionic-angular';
@Page({
    templateUrl: 'build/pages/movies/movies.html'
})
export class MoviesPage {
    static get parameters() {
        return [[NavController], [Platform]];
    }
    constructor(nav, platform) {
        this.nav = nav;
        this.platform = platform;
    }
}
```

DEPENDENCY INJECTION

```
import {Page, NavController, Platform} from 'ionic-angular';
@Page({
    templateUrl: 'build/pages/movies/movies.html'
})
export class MoviesPage {
    static get parameters() {
        return [[NavController], [Platform]];
    }
    constructor(nav, platform) {
        this.nav = nav;
        this.platform = platform;
    }
}
```



Constructor is function
called when
new instance is created

DEPENDENCY INJECTION

```
import {Page, NavController, Platform} from 'ionic-angular';
@Page({
    templateUrl: 'build/pages/movies/movies.html'
})
export class MoviesPage {
    static get parameters() {
        return [[NavController], [Platform]];
    }
    constructor(nav, platform) {
        this.nav = nav;
        this.platform = platform;
    }
}
```

Inject components to constructor

Constructor is function called when new instance is created

BINDING

BINDING

```
<input [value]="name">
```

BINDING

One-way data binding []

```
<input [value]="name">
```

BINDING

One-way data binding []

```
<input [value]="name">
```

```
<button (click)="someFunction($event)"></button>
```

BINDING

One-way data binding []

```
<input [value]="name">
```

```
<button (click)="someFunction($event)"></button>
```

Call function on event ()

BINDING

One-way data binding []

```
<input [value]="name">
```

```
<button (click)="someFunction($event)"></button>
```

Call function on event ()

```
<input [value]="name" (input)="name = $event.target.value">
```

BINDING

One-way data binding []

```
<input [value]="name">
```

```
<button (click)="someFunction($event)"></button>
```

Call function on event ()

```
<input [value]="name" (input)="name = $event.target.value">
<!-- or -->
<input [(ngModel)]="name">
```

BINDING

One-way data binding []

```
<input [value]="name">
```

```
<button (click)="someFunction($event)"></button>
```

Call function on event ()

```
<input [value]="name" (input)="name = $event.target.value">
<!-- or -->
<input [(ngModel)]="name">
```

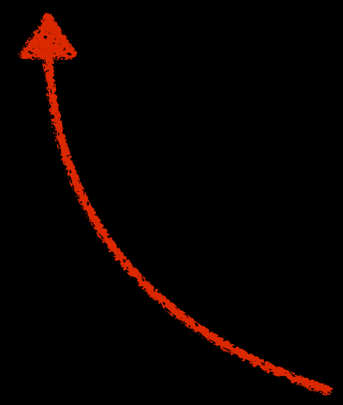
Two-way data binding [()]

RENDERING

```
<p>Hello my name is {{name}} and I like {{thing}}.</p>
```

RENDERING

```
<p>Hello my name is {{name}} and I like {{thing}}.</p>
```



The same as in Angular1

CREATE LOCAL VARIABLE

```
<p #myParagraph></p>
<button (click)="myParagraph.innerHTML = 'Fill element with
something...'">
```

CREATE LOCAL VARIABLE


```
<p #myParagraph></p>
<button (click)="myParagraph.innerHTML = 'Fill element with
something...'">
```

DECORATORS

- **Metadata and info about classes**
- **Directly above class definition**
- Starts with @

DECORATOR TYPES

- **@App** – Declare class for root component in Angular2
- **@Component** – Separately usable element (template, style, logic)
- **@Page** – Ionic-specific component
- **@Directive** – Modify behavior of existing component
- **@Injectable** – Inject to constructor, for creating services
- **@Pipe** – similar with filter in Angular1
- …

DECORATOR EXAMPLE

```
import {Page, NavController, Platform} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movies/movies.html'
})
export class MoviesPage {
    ...
}
```

DECORATOR EXAMPLE

```
import {Page, NavController, Platform} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movies/movies.html'
})
export class MoviesPage {
    ...
}
```

Decorator for Ionic2 page

DECORATOR EXAMPLE

```
import {Page, NavController, Platform} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movies/movies.html'
})
export class MoviesPage {
    ...
}
```

Object with metadata, contains template URL for specific page

Decorator for Ionic2 page

DIRECTIVES

Modify behavior of existing component

```
<section *ngIf="showSection"></section>
<li *ngFor="#item of items"></li>
```

DIRECTIVES

Modify behavior of existing component

```
<section *ngIf="showSection"></section>
<li *ngFor="#item of items"></li>
```

Array/Object iterator, the same as ng-repeat from Angular1

DIRECTIVES

Modify behavior of existing component

```
<section *ngIf="showSection"></section>
<li *ngFor="#item of items"></li>
```

Remember #?

Array/Object iterator, the same as ng-repeat from Angular1

DIRECTIVES

Modify behavior of existing component

* Syntactic sugar for <template>

```
<section *ngIf="showSection"></section>
<li *ngFor="#item of items"></li>
```

Remember #?

Array/Object iterator, the same as ng-repeat from Angular1

TEMPLATE ELEMENT, *

- <template> rendered when script is running
- * is **embedded template**
  - You can work with HTML elements like with <template>

LOOPING

```
<ion-list>
    <ion-item *ngFor="#movie of movies" (click)="viewMovie(movie)">
        <h2>{{movie.title}}</h2>
        <p>{{movie.rating}} / 10</p>
    </ion-item>
</ion-list>
```

IONIC2 PROJECT STRUCTURE

```
├── app
|   ├── app.js
|   ├── pages
|   |   ├── page1
|   |   |   ├── page1.html
|   |   |   ├── page1.js
|   |   |   └── page1.scss
|   |   ├── page2
|   |   ├── page3
|   |   └── tabs
|   └── theme
|       ├── app.core.scss
|       ├── app.ios.scss
|       ├── app.md.scss
|       ├── app.variables.scss
|       └── app.wp.scss
├── config.xml
├── gulpfile.js
├── hooks
├── ionic.config.js
├── ionic.config.json
├── node_modules
├── package.json
├── platforms
├── plugins
├── resources
└── www
    └── index.html
```

IONIC2 PROJECT STRUCTURE

- **Do not edit code in WWW directory except index.html!**

- Except resources and index.html, everything in **www directory** is generated automatically from **app** directory!

IONIC2 GENERATOR

IONIC2 GENERATOR

- We don’t need to create and include files manually!
- Use **Ionic Generator**, you will love it

```
$ ionic g [page|component|directive|pipe|provider|tabs] some-name
```

CREATE COMPONENT

```
$ ionic g component my-component
```

app/components/my-component/my-component.js

```
import {Component} from 'angular2/core';
import {IONIC_DIRECTIVES} from 'ionic-angular';
@Component({
    selector: 'my-component',
    templateUrl: 'build/components/my-component/my-component.html',
    directives: [IONIC_DIRECTIVES]
})
export class MyComponent {
    constructor() {
        this.text = 'Hello World';
    }
}
```

CREATE COMPONENT

```
$ ionic g component my-component
```

app/components/my-component/my-component.js

```
import {Component} from 'angular2/core';
import {IONIC_DIRECTIVES} from 'ionic-angular';
@Component({
    selector: 'my-component',
    templateUrl: 'build/components/my-component/my-component.html',
    directives: [IONIC_DIRECTIVES]
})
export class MyComponent {
    constructor() {
        this.text = 'Hello World';
    }
}
```

Decorator for component

CREATE PAGE

```
$ ionic g page my-page
```

app/pages/my-page/my-page.js

```
import {Page, NavController} from 'ionic-angular';
@Page({
  templateUrl: 'build/pages/my-page/my-page.html',
})
export class MyPagePage {
  static get parameters() {
    return [[NavController]];
  }
  constructor(nav) {
    this.nav = nav;
  }
}
```

CREATE PAGE

```
$ ionic g page my-page
```

app/pages/my-page/my-page.js

```
import {Page, NavController} from 'ionic-angular';
@Page({
  templateUrl: 'build/pages/my-page/my-page.html',
})
export class MyPagePage {
  static get parameters() {
    return [[NavController]];
  }
  constructor(nav) {
    this.nav = nav;
  }
}
```

Create bundle with JS + HTML + SASS

CREATE DIRECTIVE

```
$ ionic g directive my-super-directive
```

app/components/my-super-directive/my-super-directive.js

```
import {Directive} from 'angular2/core';
@Directive({
  selector: '[my-super-directive]' // Attribute selector
})
export class MySuperDirective {
  constructor() {
    console.log('Hello World');
  }
}
```

CREATE PIPE

```
$ ionic g pipe my-old-pipes
```

app/pipes/my-old-pipes.js

```
import {Injectable, Pipe} from 'angular2/core';
@Pipe({
  name: 'my-old-pipes'
})
@Injectable()
export class MyOldPipes {
  transform(value, args) {
    value = value + ''; // make sure it's a string
    return value.toLowerCase();
  }
}
```

CREATE PROVIDER

```
$ ionic g provider my-provider
```

app/providers/my-provider/my-provider.js

```
import {Injectable} from 'angular2/core';
import {Http} from 'angular2/http';
import 'rxjs/add/operator/map';
@Injectable()
export class MyProvider {
  static get parameters(){
    return [[Http]]
  }
  constructor(http) {
    this.http = http;
    this.data = null;
  }
  load() {
    if (this.data) return Promise.resolve(this.data);
    return new Promise(resolve => {
      this.http.get('path/to/data.json')
        .map(res => res.json()).subscribe(data => {
          this.data = data;
          resolve(this.data);
        });
    });
  }
}
```

CONDITIONALS

CONDITIONALS

```
<div *ngIf="someBoolean">Hello 1</div>
```

CONDITIONALS

```
<div *ngIf="someBoolean">Hello 1</div>


<div [ngSwitch]="someValues">
  <p *ngSwitchWhen="1">Paragraph 1</p>
  <p *ngSwitchWhen="2">Paragraph 2</p>
  <p *ngSwitchWhen="3">Paragraph 3</p>
  <p *ngSwitchDefault>Paragraph</p>
</div>
```

CONDITIONALS

```
<div *ngIf="someBoolean">Hello 1</div>


<div [ngSwitch]="someValues">
  <p *ngSwitchWhen="1">Paragraph 1</p>
  <p *ngSwitchWhen="2">Paragraph 2</p>
  <p *ngSwitchWhen="3">Paragraph 3</p>
  <p *ngSwitchDefault>Paragraph</p>
</div>


<div [hidden]="someBoolean">Hello 2</div>
```

CONDITIONALS

```
<div *ngIf="someBoolean">Hello 1</div>


<div [ngSwitch]="someValues">
  <p *ngSwitchWhen="1">Paragraph 1</p>
  <p *ngSwitchWhen="2">Paragraph 2</p>
  <p *ngSwitchWhen="3">Paragraph 3</p>
  <p *ngSwitchDefault>Paragraph</p>
</div>


<div [hidden]="someBoolean">Hello 2</div>


<div [class.my-custom-class]="someBoolean">Hello 3</div>
```

NAVIGATION

NAVIGATION

- Pop and push to **history stack**
- **push(SomePage)** – add to the top of stack
- **pop()** – delete last page from the stack

NAVIGATION EXAMPLE

NAVIGATION EXAMPLE

app/pages/movies/movies.html

```
<ion-navbar *navbar>
    <ion-title>movies</ion-title>
</ion-navbar>

<ion-content padding class="movies">
    <button class="positive" (click)="showDetail()">
        Go to detail
    </button>
</ion-content>
```

NAVIGATION EXAMPLE

app/pages/movies/movies.html

```
<ion-navbar *navbar>
    <ion-title>movies</ion-title>
</ion-navbar>

<ion-content padding class="movies">
    <button class="positive" (click)="showDetail()">
        Go to detail
    </button>
</ion-content>
```

Call method for showing another page

NAVIGATION EXAMPLE

app/pages/movie-detail/movie-detail.html

```
<ion-navbar *navbar>
    <ion-title>movie-detail</ion-title>
</ion-navbar>

<ion-content padding class="movie-detail">
    <button class="positive" (click)="goBack()">
        Go back
    </button>
</ion-content>
```

NAVIGATION EXAMPLE

app/pages/movie-detail/movie-detail.html

```
<ion-navbar *navbar>
    <ion-title>movie-detail</ion-title>
</ion-navbar>

<ion-content padding class="movie-detail">
    <button class="positive" (click)="goBack()">
        Go back
    </button>
</ion-content>
```

Call method for going back

NAVIGATION – GO TO PAGE

app/pages/movies/movies.js

```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage)
    }
}
```

NAVIGATION – GO TO PAGE

app/pages/movies/movies.js


```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage)
    }
}
```

NAVIGATION – GO TO PAGE

app/pages/movies/movies.js

Import

```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage)
    }
}
```

Dependency injection

NAVIGATION – GO TO PAGE

app/pages/movies/movies.js

```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage)
    }
}
```

Import

Dependency injection

Save as class variable

NAVIGATION – GO TO PAGE

app/pages/movies/movies.js

```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage)
    }
}
```

Import

Dependency injection

Save as class variable

Push MovieDetailPage to the stack

NAVIGATION – GO TO PAGE

app/pages/movies/movies.js

```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage)
    }
}
```

Import

Dependency injection

Save as class variable

Push MovieDetailPage to the stack

NAVIGATION – GO BACK

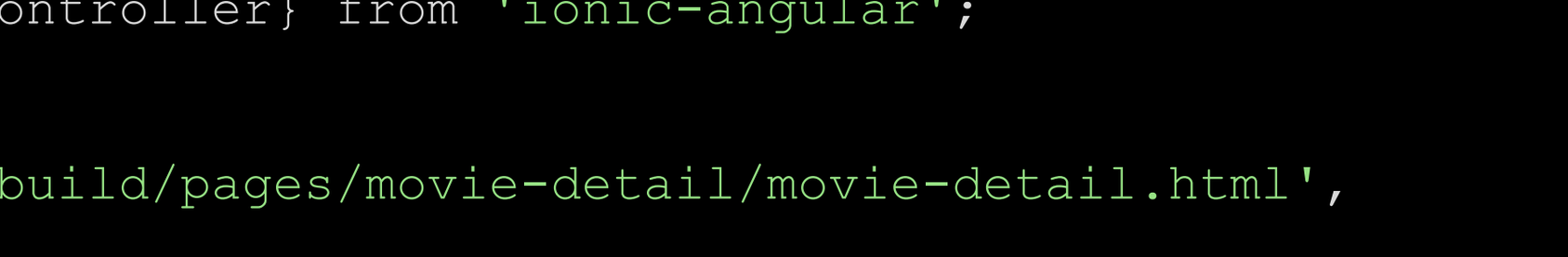
app/pages/movie-detail/movie-detail.js

```
import {Page, NavController} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    goBack() {
        this.nav.pop()
    }
}
```

NAVIGATION – GO BACK

app/pages/movie-detail/movie-detail.js

```
import {Page, NavController} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    goBack() {
        this.nav.pop()
    }
}
```



NAVIGATION – GO TO PAGE WITH PASSING PARAMS

```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage, {
            id: 4
        })
    }
}
```

NAVIGATION – GO TO PAGE WITH PASSING PARAMS

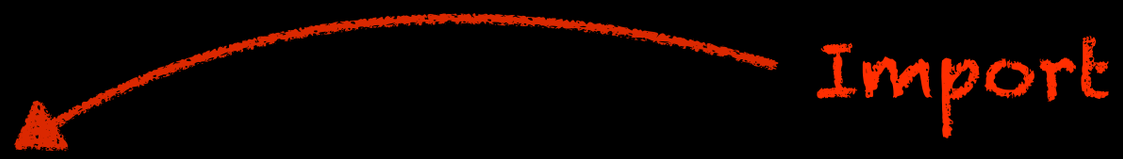
```
import {Page, NavController} from 'ionic-angular';
import {MovieDetailPage} from '../movie-detail/movie-detail'

@Page({
    templateUrl: 'build/pages/movies/movies.html',
})
export class MoviesPage {
    static get parameters() {
        return [[NavController]];
    }

    constructor(nav) {
        this.nav = nav;
    }

    showDetail() {
        this.nav.push(MovieDetailPage, {
            id: 4
        })
    }
}
```

Second parameter is an object with params

NAVIGATION – GET PARAMS

```
import {Page, NavController, NavParams} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController], [NavParams]];
    }

    constructor(nav, params) {
        this.nav = nav;
        this.params = params;
        console.log("ID: " + this.params.get("id"));
    }

    goBack() {
        this.nav.pop();
    }
}
```

NAVIGATION – GET PARAMS




```
import {Page, NavController, NavParams} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController], [NavParams]];
    }

    constructor(nav, params) {
        this.nav = nav;
        this.params = params;
        console.log("ID: " + this.params.get("id"));
    }

    goBack() {
        this.nav.pop();
    }
}
```

NAVIGATION – GET PARAMS

Import

```
import {Page, NavController, NavParams} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController], [NavParams]];
    }

    constructor(nav, params) {
        this.nav = nav;
        this.params = params;
        console.log("ID: " + this.params.get("id"));
    }

    goBack() {
        this.nav.pop();
    }
}
```

Dependency injection

NAVIGATION – GET PARAMS

Import

```
import {Page, NavController, NavParams} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController], [NavParams]];
    }

    constructor(nav, params) {
        this.nav = nav;
        this.params = params;
        console.log("ID: " + this.params.get("id"));
    }

    goBack() {
        this.nav.pop();
    }
}
```

Dependency injection

Save as class variable

NAVIGATION – GET PARAMS

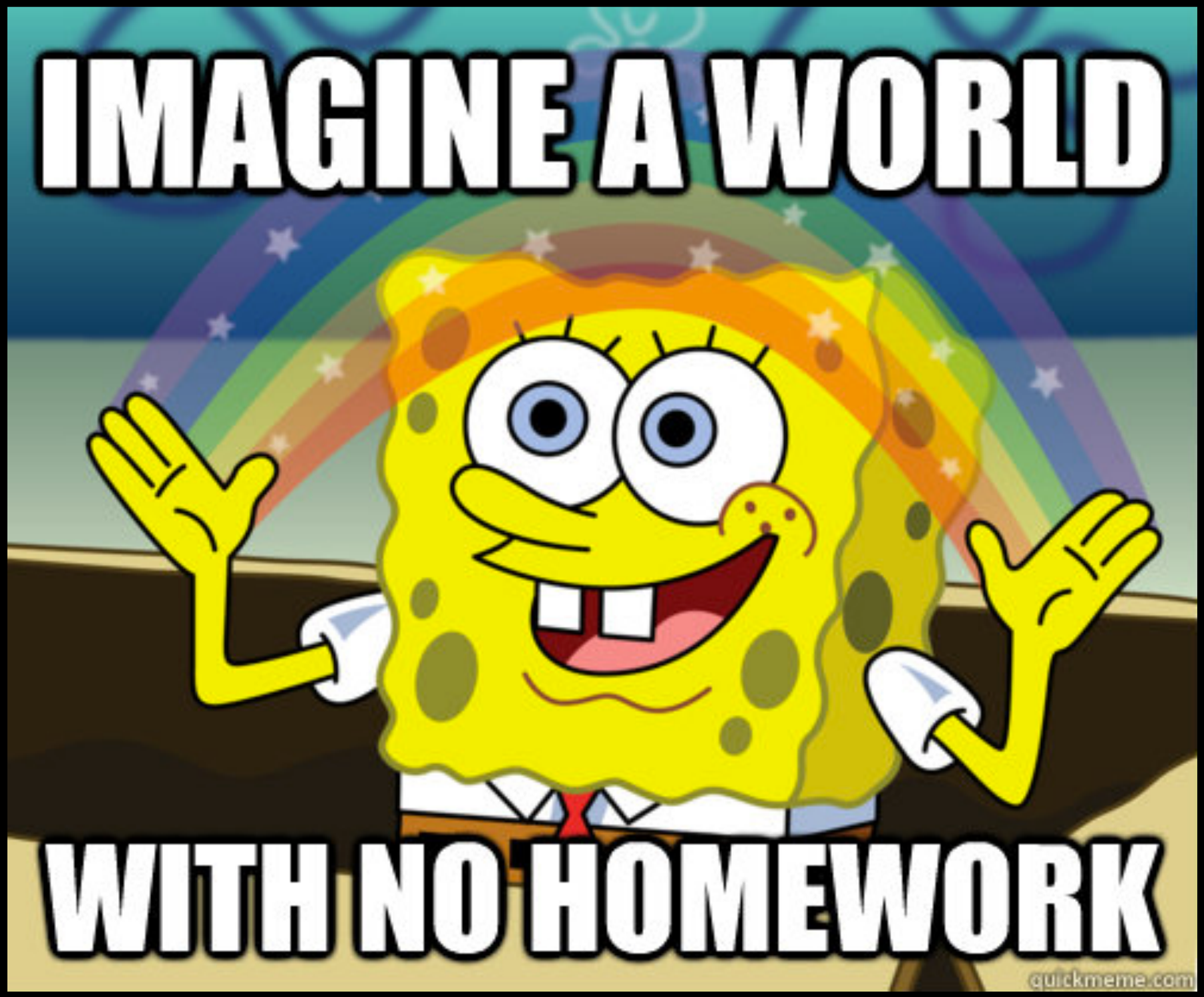
Import

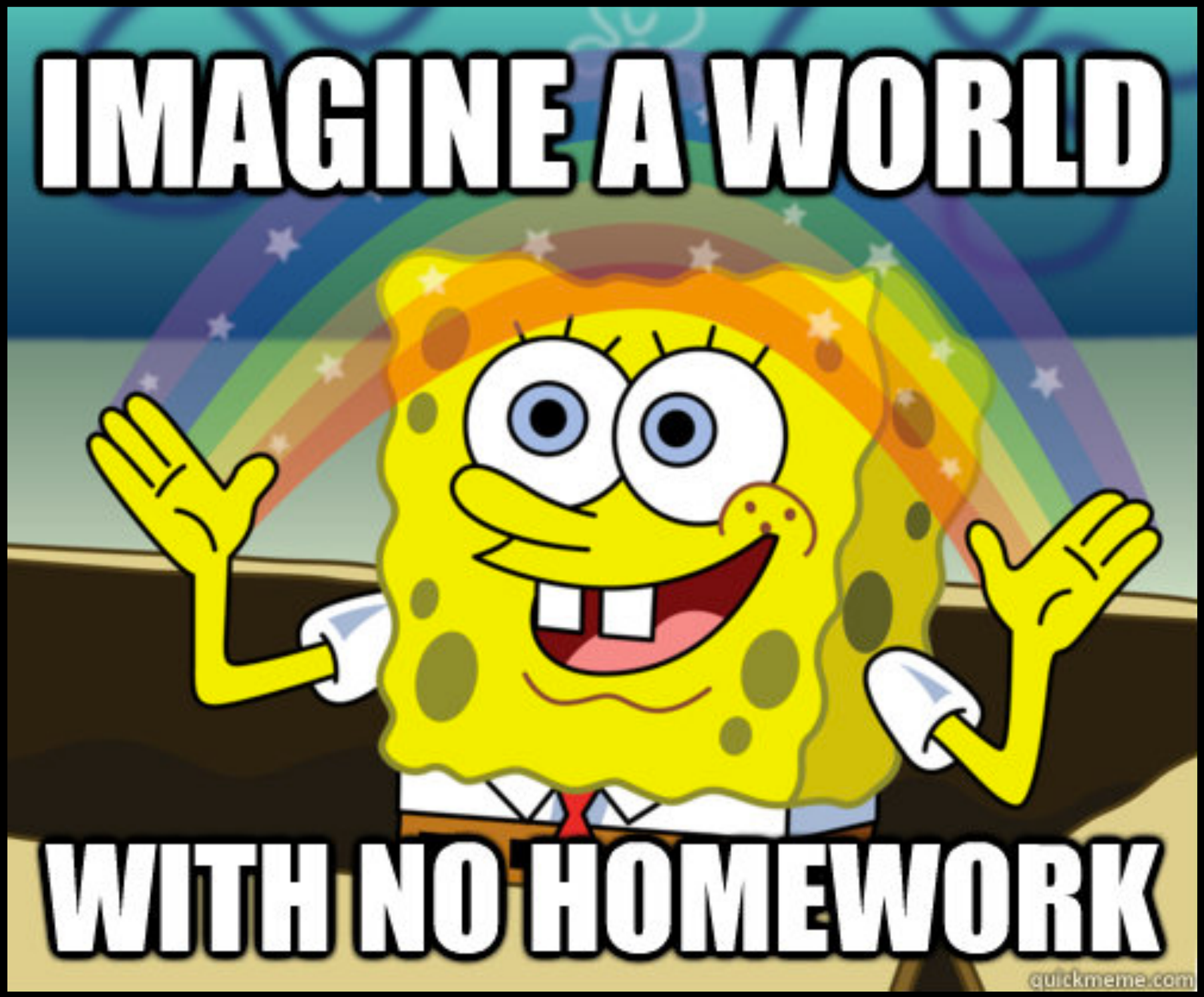
```
import {Page, NavController, NavParams} from 'ionic-angular';

@Page({
    templateUrl: 'build/pages/movie-detail/movie-detail.html',
})
export class MovieDetailPage {
    static get parameters() {
        return [[NavController], [NavParams]];
    }

    constructor(nav, params) {
        this.nav = nav;
        this.params = params;
        console.log("ID: " + this.params.get("id"));
    }

    goBack() {
        this.nav.pop();
    }
}
```

Dependency injection

Get ID from params

Save as class variable


IMAGINE A WORLD
WITH NO HOMEWORK
quickmeme.com



WORK ON PROJECTS!

QUESTIONS?

JAN VÁCLAVÍK

@janvaclavik

DANIEL RYS

@danielrys

WWW.USERTECHNOLOGIES.COM