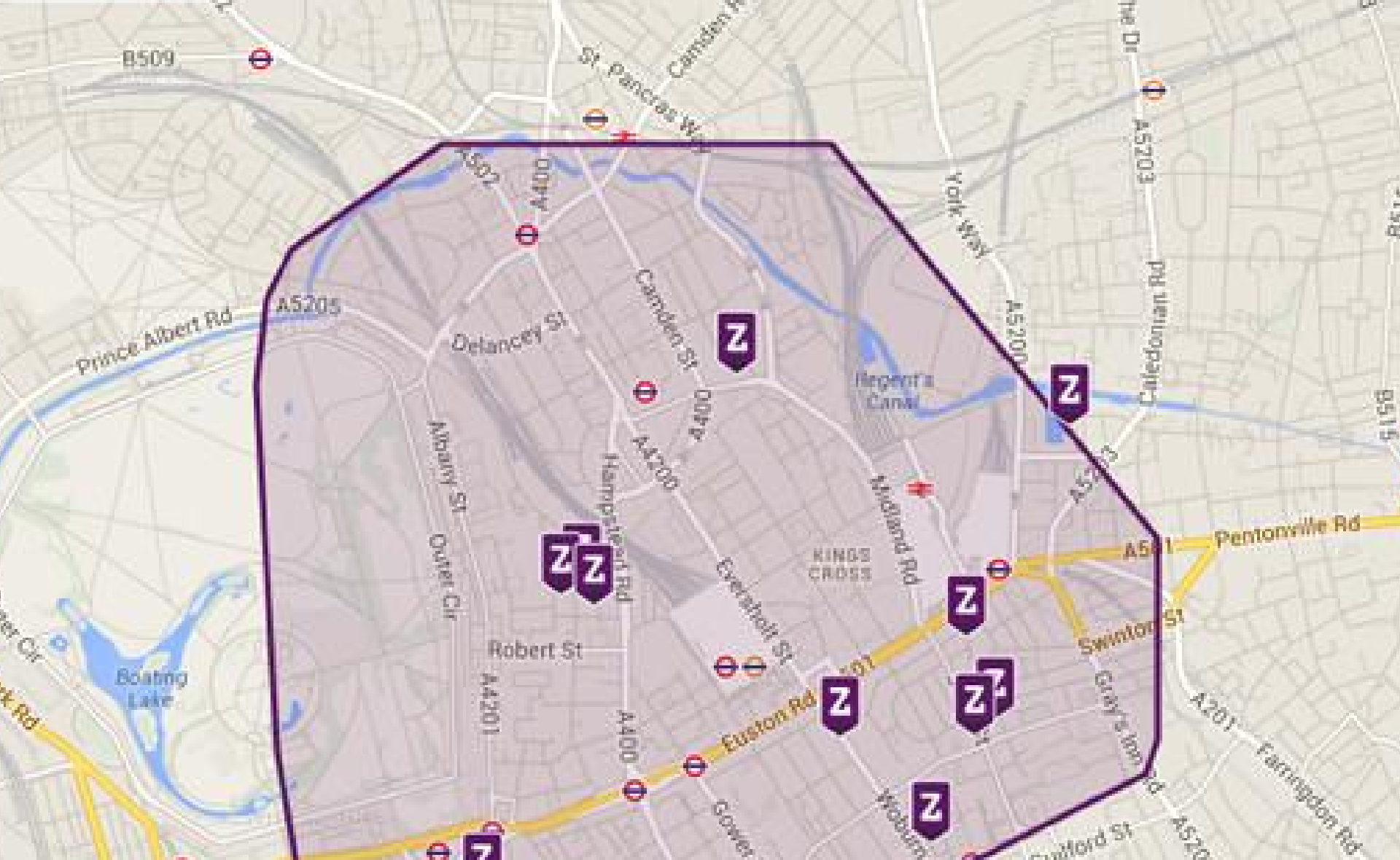


# Geo Catching Project Kick-off

11/10/2016

Cécile Camillieri/Clément Duffau





Reminder

Time to start!  
(finally)

# GeoCatching first version

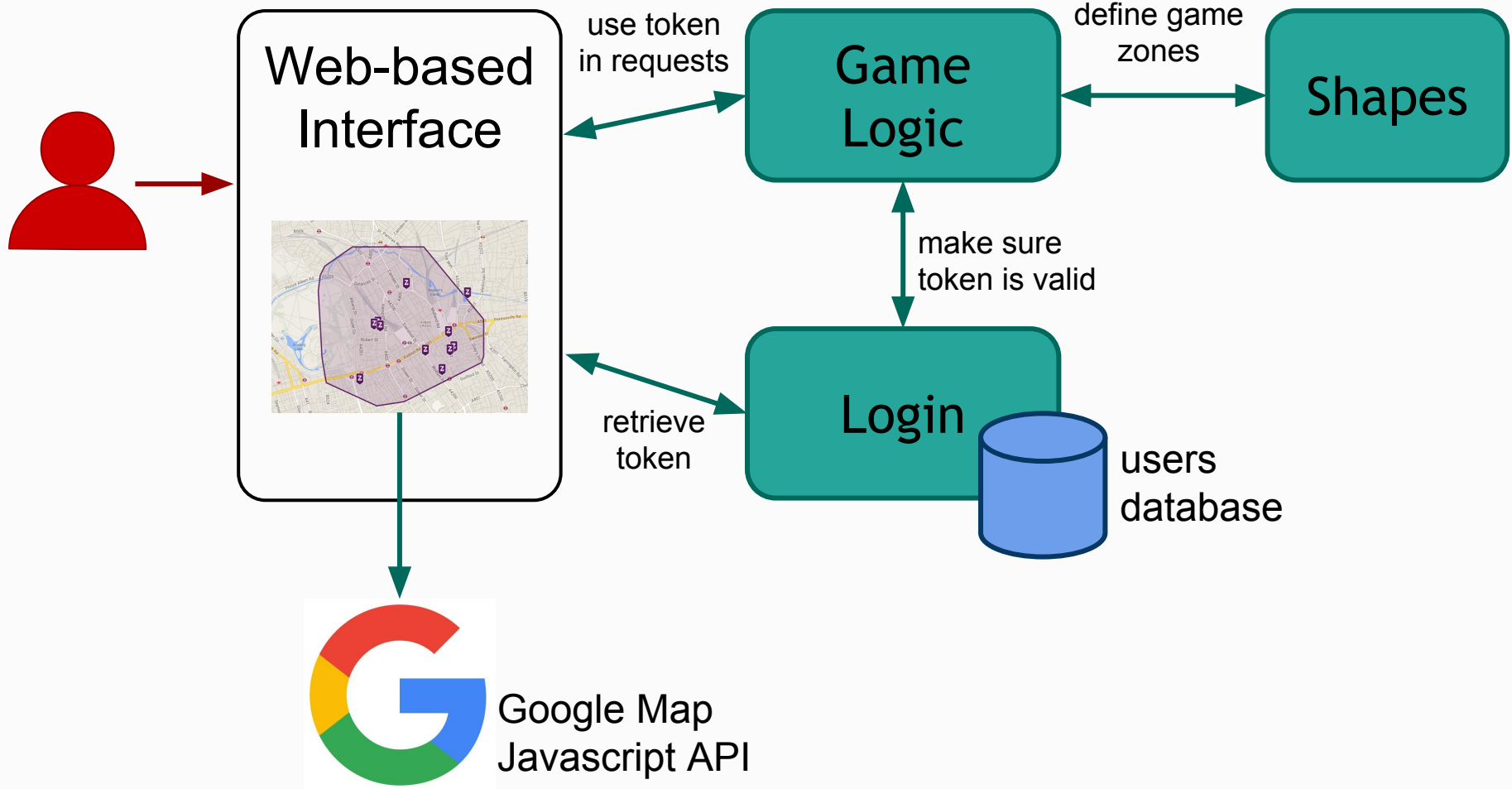
Drawing of zones on a map

User login and joining of a game

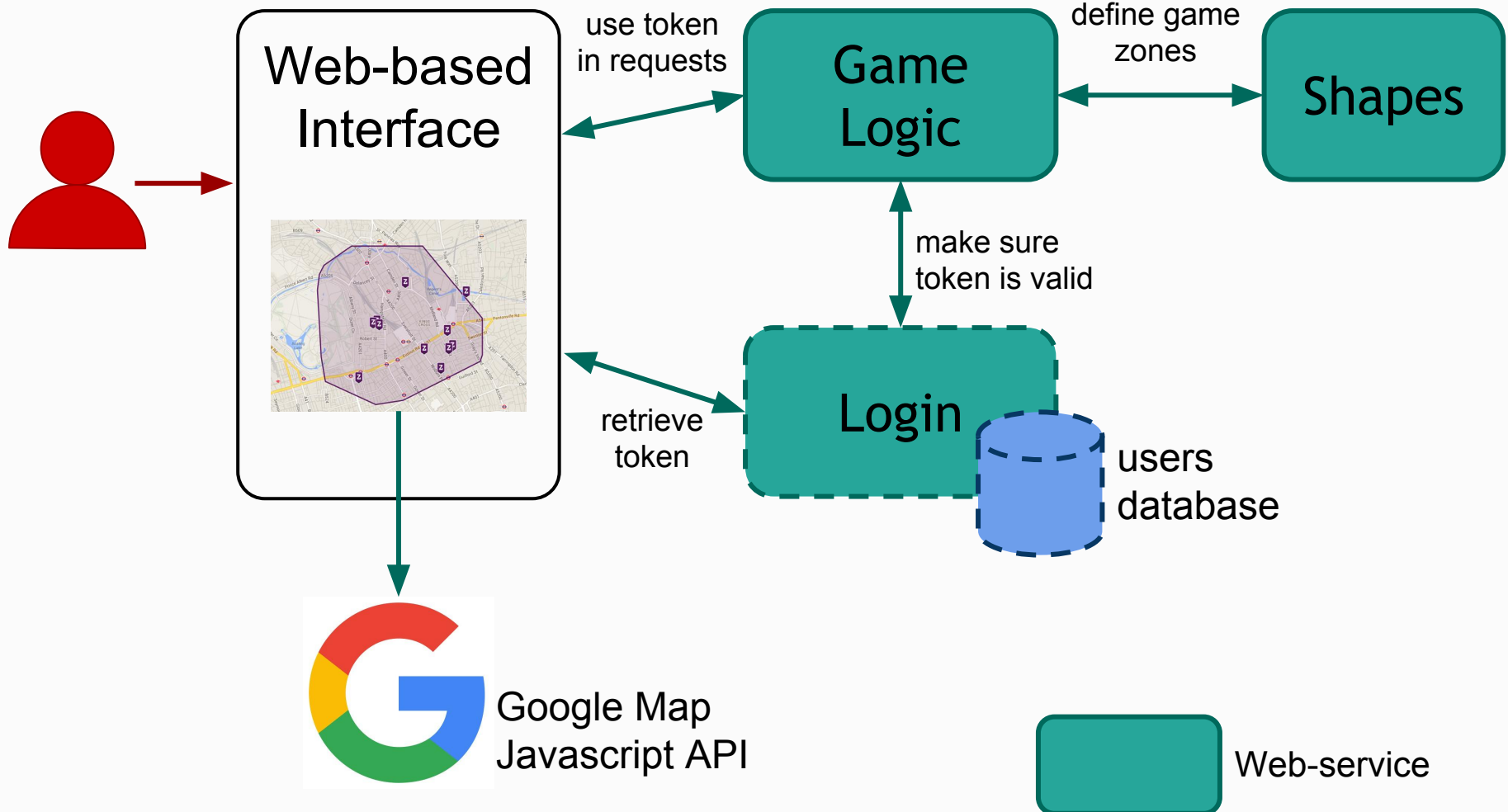
Browser-based geolocation of the player

Color differently the zone the player is in

# Proposed Architecture



# Proposed Architecture



# Implementation

A Java web-service to handle the **logic of the game**

- Knows who the players are
- Knows the zones for the game
- Tracks location of players
- Checks if a player is in a zone

Connect to the **login web-service** (provided next week)

A user interface using Google Map's API and the web-services above.

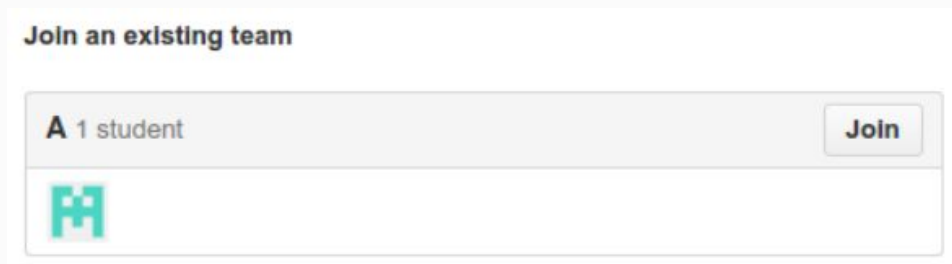
- Can be accessed from a **browser**
- Any web technology you prefer

# Requirements




# Code on Github Classroom

- Do **NOT** host your own repository
- Go to <https://classroom.github.com/group-assignment-invitations/4ea7e8e6af81ea0411d311073fda985b>
- Select your team if it is already in the list.

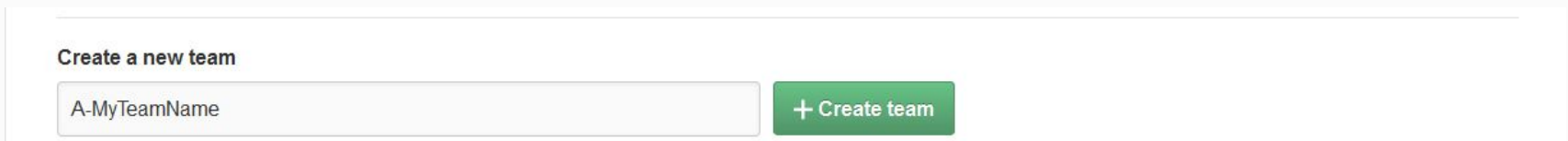


Join an existing team

A 1 student



- Otherwise create your team. Name it with its letter (A to F) followed by your own team name if you want



Create a new team

A-MyTeamName

# Guidelines

Don't forget about **project management** !

Define and share your **tasks** properly.

**Commits** should correspond to tasks in Jira

- Example : “MyTaskId: Fix map display bug on Safari”

Focus on...  
Google's API

# Google's Javascript APIs

Requests through **HTTP**

Need to authenticate requests with an **API key**

Great **documentation** with lots of examples

<https://developers.google.com/maps/documentation/javascript/>

# Getting an API key

- Direct link to get a key

- Google account required
- Create a new project
- Name the key
- Click “create”
- That’s it!

- Alternatively:

- Detailed instructions

Enregistrer l'application pour Google Maps JavaScript API Google Maps Geocoding API Google Maps Directions API Google Maps Distance Matrix API Google Maps Elevation API Google Places API Web Service dans la Console d'API Google

Console d'API Google vous permet de gérer votre application et de surveiller l'utilisation de l'API.

### Sélectionner un projet dans lequel votre application va être enregistrée

Vous pouvez utiliser un projet pour gérer l'ensemble de vos applications, ou vous pouvez créer un projet différent pour chaque application.

Créer un projet ▾

Continuer

### Clé API

Vous pouvez utiliser cette clé API dans ce projet avec n'importe quelle API compatible. Pour utiliser cette clé dans votre application, utilisez-la sous la forme du paramètre `key=API_KEY`.

#### Nom

Cle\_Maps\_Cours\_GL

#### ⚠ Restriction de clé

Cette clé ne fait l'objet d'aucune restriction. Pour éviter toute utilisation abusive et tout vol de quota, ajoutez des restrictions pour votre clé.

Une restriction de clé vous permet de spécifier les sites Web, les adresses IP ou les applications pouvant utiliser cette clé. [En savoir plus](#)

- Aucun
- Référents HTTP (sites Web)
- Adresses IP (serveurs Web, tâches Cron, etc.)
- Applications Android
- Applications iOS

Remarque : L'application de ce paramètre peut prendre jusqu'à cinq minutes.

Créer

Annuler

Focus on...  
AngularJS

# Disclaimer

This is **not** a programming course

We are **not** here to solve all your implementation issues

Using AngularJS is **not** mandatory

But we can help **a little** if you chose to

# Tooling

Jetbrain's Webstorm

30 days trial ...

... then get a **free license** with your unice email address at:

- <https://www.jetbrains.com/shop/eform/students>

Alternatives: SublimeText, etc.



# Base AngularJS (1) project

Already in your team github repository

Instructions for setup

Simple AngularJS app with basic dependencies

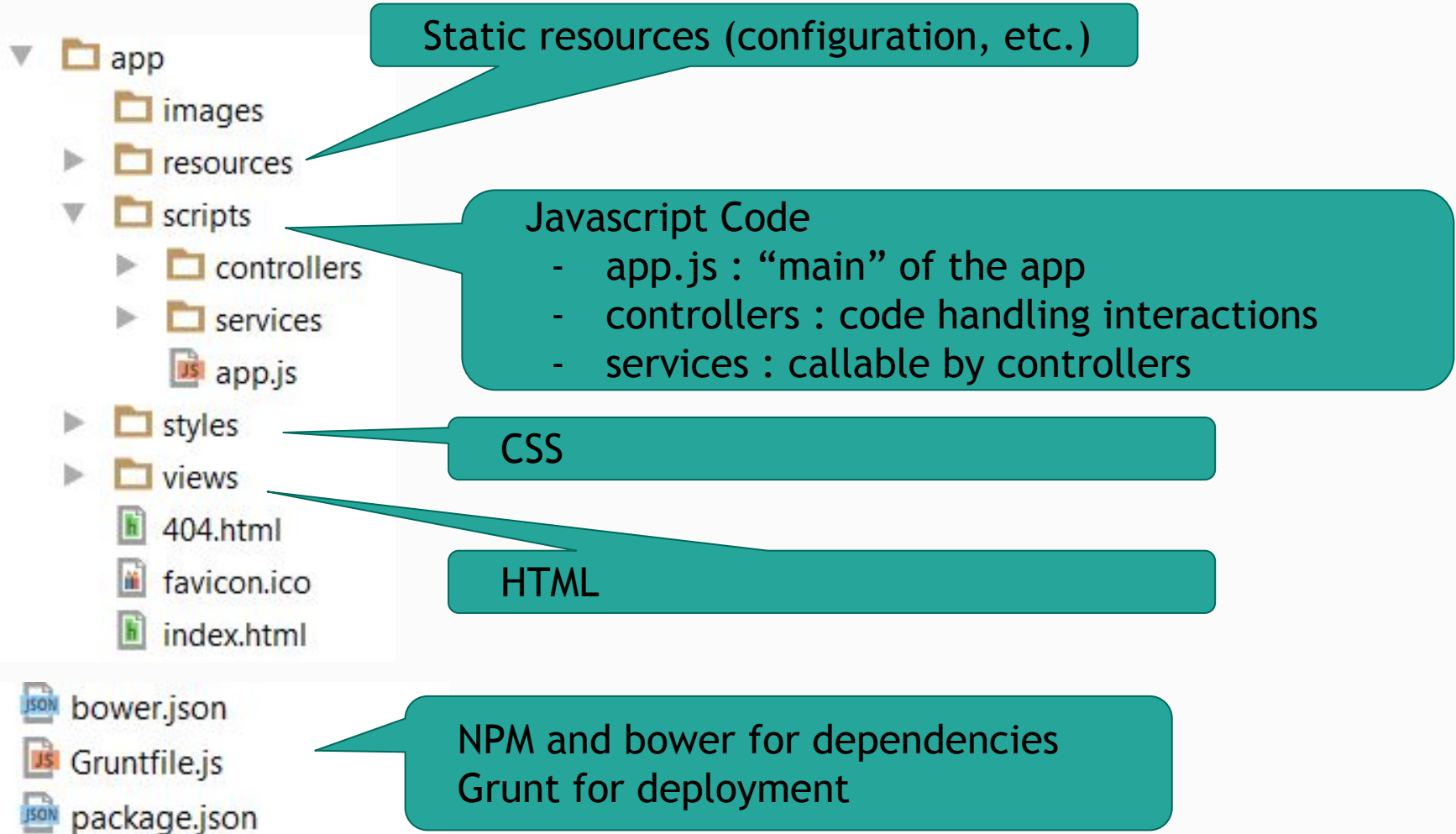
A JS file which queries the [shapes service](#)



VS



# Basic project structure



# App.js

Name of the app

angular

```
.module('geoCatchingApp', [  
  'ngRoute',  
  'ui.bootstrap',  
  'ngAnimate'])
```

Dependencies

- ngRoute: handle routing (urls)
- bootstrap: css
- ngAnimate: animations (collapse, etc.)



ng-anything = something defined by Angular

# App.js - routing

## angular

```
.module('geoCatchingApp', [  
  'ngRoute',  
  'ui.bootstrap',  
  'ngAnimate'])  
.config(function ($routeProvider) {  
  $routeProvider  
    .when('/', {  
      templateUrl:  
'views/main.html',  
      controller: 'MainCtrl'  
    })  
    .when('/about', {  
      templateUrl:  
'views/about.html'  
    })  
    .otherwise({  
      templateUrl: '404.html'  
    });  
});
```

# App.js - routing

```
angular
  .module('geoCat
    'ngRoute',
    'ui.bootstrap',
    'ngAnimate'])
  .config(function($routeProvider) {
    $routeProvider
      .when('/', {
        templateUrl:
'views/main.html',
        controller: 'MainCtrl'
      })
      .when('/about', {
        templateUrl:
'views/about.html'
      })
      .otherwise({
        templateUrl: '404.html'
      });
  });
```

Path

View file (HTML)

Associated controller (JS)

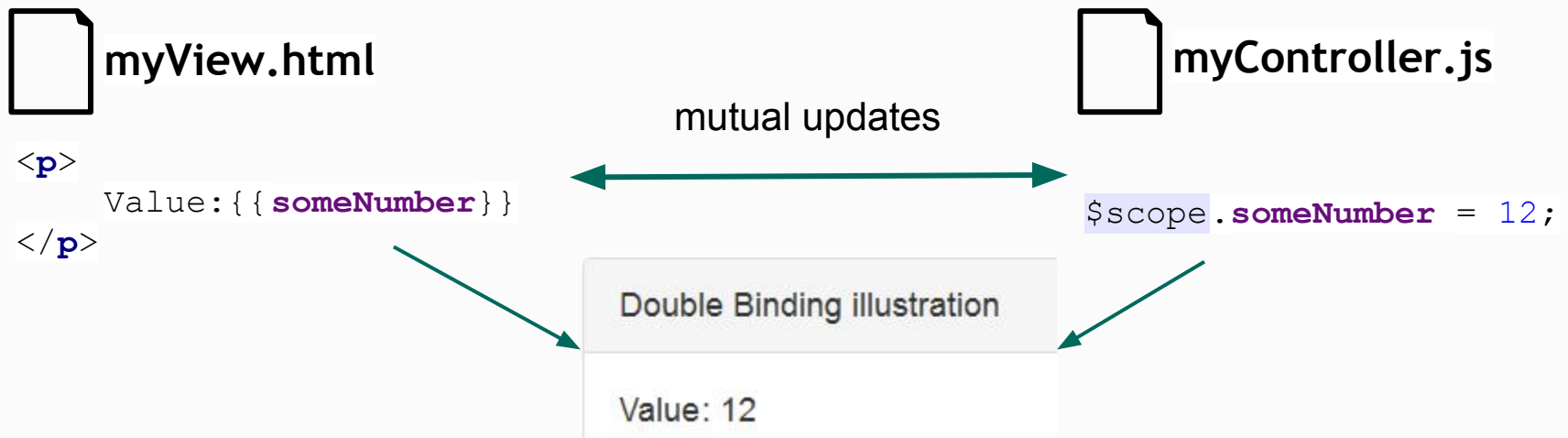
# MVVM: Model View View-Model

- Angular is **not** MVC
- **Double binding** between the model and the view



# Model View ViewModel

- Angular is **not** MVC
- **Double binding** between the model and the view





# Scope

- Contains the **model** for a given context
- A scope for all the app: **\$rootScope**  
=> **Global variables**
- A scope for each context: **\$scope**  
=> **Local variables**

# Views: Some useful directives

- ng-bind: binds view to element in the scope

```
<p ng-bind="someNumber"></p>
```

- ng-hide: hide element if condition is met
- ng-show: show element if condition is met

```
<p ng-show="someExpression">This is hidden if someExpression == false</ p>
```

- ng-if: create element if condition is met

```
<p ng-if="display">This is visible if display == true</ p>
```

- ng-class: set an element's class depending on the context

```
<p ng-class="{red:error}">This p has class 'red' if error == true</ p>
```

- ng-repeat: loop through an array to create DOM elements

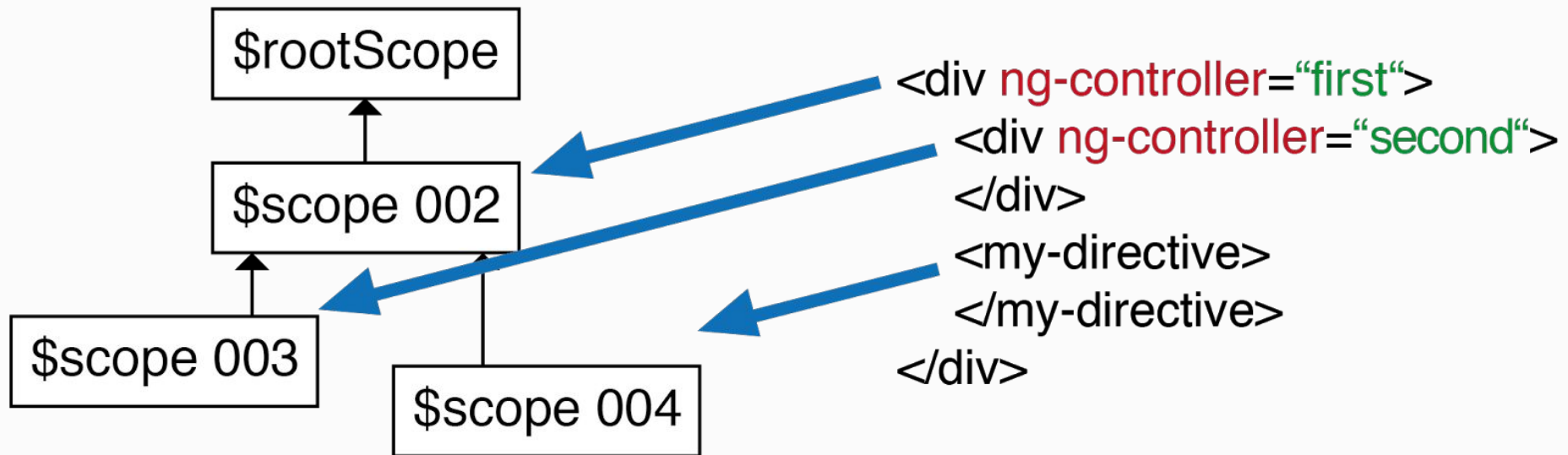
```
<ul> <li ng-repeat="elt in array">element {{elt.id}}</li> </ul>
```

- ng-controller: set a controller for an element

```
<div ng-controller="myController">...</div>
```

# Hierarchy of Scopes

## Scope Tree



# Javascript

- All .js files need to be added in index.html

```
<script src="scripts/app.js"></script>
```

app.js

```
<script src="scripts/controllers/main.js"></script>
```

```
<script src="scripts/controllers/...js"></script>
```

Controllers

```
<script src="scripts/services/ShapesWService.js"></script>
```

```
<script src="scripts/services/...js"></script>
```

Services

# Controllers and Services

- Controllers have **access to scope** and can **act on the view**
  - `$scope.variable = "value";`
  - `$window.alert("message");`
  - etc.
- Services **offer functions** that can be called from JS
  - Used to encapsulate operations in a separate component
  - Similar to classes/objects

# Declaring controllers and Services

id of the app

```
angular.module('geoCatchingApp')  
  .controller('MainCtrl', ['$scope', '$rootScope', 'ShapesWSClientService',  
    function ($scope, $rootScope, JsonLoaderService, ShapesWSClient) {  
      [Your code here]  
    }  
  ]  
});
```

id of the Controller

```
angular.module('geoCatchingApp')  
  .service('ShapesWSClientService', ['$http', '$q', function ($http, $q) {  
    [Your code here]  
  }  
]);
```

dependencies injection

# Dependencies injection

```
angular.module('geoCatchingApp')
```

```
.controller('MainCtrl', ['$scope', '$rootScope', 'ShapesWSClientService',
```

```
function ($scope, $rootScope, ShapesWSClient) {
```

```
    $scope.variable="test";
```

```
});
```

ids of the needed dependencies

The controller is a “big function”. Angular injects the dependencies as parameters

using the dependency \$scope



The name of the dependencies in the function does not need to be the same as the id like for ShapesWSClient, the injection is based on the declaration order.

# A little more on scopes

```
angular.module('geoCatchingApp')  
  .controller('MainCtrl', ['$scope', '$rootScope', 'ShapesWSClientService',  
    function ($scope, $rootScope, ShapesWSClient) {  
  
      $scope.variable="test";  
  
      $scope.operation = function(a) { ... }  
  
      function(a) { ... }  
  
    }]);
```

accessible from html

only accessible from JS



Expected for  
next week

# For October 16th 23:59

Create [your team](#) on Github Classroom

[Everyone](#) in the team joins

Commit and push your shape web-service [source code](#)

- no .class, no hidden files, no IDE settings, etc.
- use the .gitignore file (cf. versioning course).
- Create a tag “sprint-0” on master.

Detailed instructions in the [readme file](#) in your repo.

# Summary

# What's expected - Game

- Drawing of zones on the map
  - <https://developers.google.com/maps/documentation/javascript/shapes>
- User login and joining of a game
  - Call to the web-service that will be given next week
- Browser-based geolocation of the player
  - [http://www.w3schools.com/html/html5\\_geolocation.asp](http://www.w3schools.com/html/html5_geolocation.asp)
  - <https://developers.google.com/web/fundamentals/native-hardware/user-location/>
- Color differently the zone the player is in

# What's expected - release

- An assignment for your group on Github Classroom
- The repository should contain
  - A folder with your web interface code
  - A folder with the shapes web-service
  - A folder with a web-service to control the game
  - A readme file saying how to run the project and describing what is done and the contents on the repository
- Repository is queried by a script. **Any problem → 0**
- We only grade what's on the **master branch**.
- We get the **tag “sprint-1”**. **No tag → 0**
- Deadline is **November 13 at 23:59**. **Late → 0**

# What's expected - defense

November 15

10 minutes to present your work and a short demo

5 minutes of questions

## Minimal content

- Current state of the project
- Pros/Cons of your solution
- What's next

# Too much to do!!

- Start **small**
- Use **mocks**
- Share **tasks**
- Set **deadlines**



- The following are **not problems** (for now):
  - Interface doesn't work in some browsers
  - Network issues are not handled properly
  - No persistence

# For October 18th

Create your team on Github Classroom

Everyone in the team joins

Commit and push your shape web-service source code

- no .class, no hidden files, no IDE settings, etc.
- use the .gitignore file (cf. versioning course)





GO!!

