

course Esilv -  
Angular & Node Js

Advanced Web Features

arnaud.nauwynck@gmail.com

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Generating (Angular) Client from Swagger
  - Code Scaffolding
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- Final Words

# Reminder

**5 x CMs - Lessons**

**5 x TDs - Hands on**

**1 Project submission .... deadline 31 dec**

Reminder ...  
expectation on project submission for Exam

**Your project MUST include at least  
1 advanced feature**

# Outline

- Some Advanced features
- • AG-Grid
  - HighCharts
  - Generating (Angular) Client from Swagger
  - Code Scaffolding
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- Final Words

<https://www.ag-grid.com/>



# AG-Grid

Google ag-grid X |

Images Videos Npm React Angular News Example Vue Version

---

About 164,000,000 results (0.48 seconds)

 AG Grid  
<https://www.ag-grid.com> :

**Data Grid: AG Grid: High-Performance React Grid, Angular ...**

AG Grid is a feature rich datagrid designed for the major JavaScript Frameworks. Version 31. 0.  
Download v31 of the best Data Grid in the world now.

**Angular Data Grid**  
Our documentation will help you to get up and running with AG Grid ...

**React Data Grid: Documentation**  
Our documentation will help you to get up and running with AG Grid ...

**Demo**  
AG Grid is a feature-rich datagrid available in Community or ...

**Angular Data Grid: Quick Start**  
Angular Data GridQuick Start. angular logo · Container: for the ...

<https://ag-grid.com>

# Getting Started

The screenshot shows a web browser displaying the AG Grid Quick Start documentation at [ag-grid.com/angular-data-grid/getting-started/](https://ag-grid.com/angular-data-grid/getting-started/). The page has a dark blue header with the AG Grid logo, a search bar, and navigation links for Demo, Docs, API, Blog, Pricing, and a user profile icon.

The main content area features a sidebar on the left with a navigation menu:

- What's New
- OVERVIEW
- Get Started
  - Quick Start (selected)
  - Creating a Basic Grid
  - Video Tutorials
- Installation
- Guides
- Migration
- LAYOUT & STYLING
  - Design System
  - Theming
  - Styling
- CHARTING
  - Sparklines
  - Integrated Charts
  - Standalone Charts
- CORE FEATURES
  - Columns
  - Rows
  - Filtering
  - Selection
  - Rendering

The main content area displays the "Quick Start" section under "Angular Data Grid". It includes a heading "Quick Start", a sub-section "Create a grid in 60 Seconds", and a list of requirements:

- Container: for the grid's placement in your application.
- Styles: to define the grid's theme & dimensions.
- Row Data & Column Definitions: to define the data and how it should be displayed.

Below this, there are sections for "Install" (with a command line example) and "Create a Component" (with sample code). A sidebar on the right provides links to other documentation sections like "Quick Start", "Install", and "Create a Component".

```
npm install ag-grid-angular
```

```
import { Component } from '@angular/core';
import { AgGridModule } from 'ag-grid-angular'; // Angular Grid Logic
import { ColDef } from 'ag-grid-community'; // Column Definitions Interface

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [AgGridModule], // Add AG Grid Module to component
  styleUrls: ['./app.component.css'],
  template: ''
})

export class AppComponent {}
```

<https://ag-grid.com/example>

← → C ag-grid.com/example/

# AG Grid

Demo Documentation Pricing Blog

Data Size: 100000 Rows, 22 Co Theme: Alpine Filter: Filter any column... Take a video tour

Drag here to set row groups

Participant			Game of Choice		Performance >		Rating
<input type="checkbox"/> Name	Language	Country	Game Name	Bought	Bank Balance		
<input type="checkbox"/>		Malta	Tablut	X	\$79,428	★★	
<input type="checkbox"/>	French		Tablut	X	\$80,350		
<input type="checkbox"/>	Spanish		Tablut	X	\$67,068	★★★	
<input type="checkbox"/>	Spanish		Tablut	X	\$85,532	★★	
<input type="checkbox"/>	French		Tablut	X	\$23,417	★★★★	
<input type="checkbox"/>	Portuguese		Tablut	X	\$84,588	★	
<input type="checkbox"/>	French		Tablut	X	\$60,967	★★	
<input type="checkbox"/>	French		Tablut	X	\$2,692	★	

Pivot Mode

Search...

Participant Name Language Country Game of Choice Game Name Bought Performance Bank Balance

Row Groups Drag here to set row groups

Σ Values Drag here to aggregate

Rows: 1,470 of 100,000

# Did you notice? "Rows ... of 100,000"

<input type="checkbox"/> Chloe Hanagan	French	🇫🇷 France
<input type="checkbox"/> Ruby Greyson	French	🇫🇷 France
◀ [redacted]		
Rows: 1,470 of 100,000		

# AG-Grid performances

## Naive

create millions of  
<div></div>  
<div></div>  
<div></div>  
<div></div>  
<div></div>

Use Browser to scroll page



## Virtual Scrolling

create ~50 only if visible  
<div ..translateY(0px)>  
<div ..translateY(41px)>  
<div ..translateY(82px)>

change "div.y" position  
when scrolling  
Scroll

# Your-own-table vs AG-Grid



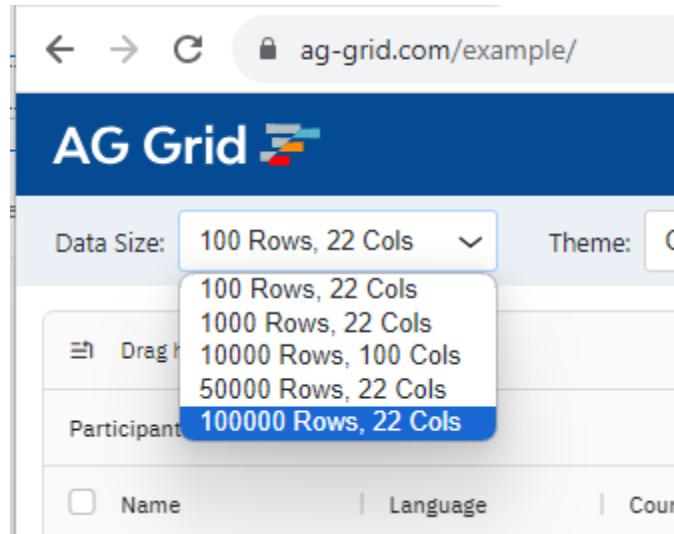
Name	Employee	Email	Phone	Website
Lazelle Ondra	Bret	Gwenethape@air	1-778-736-8091 x56440	Milegan.org
Ervin Howell	Aenean	Shawn@melissa.tv	016-652-6583 x91125	amavisio.net
Clementine Bauch	Sed	Horatio@yousicis.net	1-483-721-4447	rems.info
Patricia Lebsack	Karinae	Julliane@OConnorThury.org	483-110-9823 x739	kale.klk
Chelsey Dietrich	Kareem	Lois@Hildegard@remie.us	(231)914-1239	demos.info

1 2



```
<div *ngFor="let row of rows">
  <div class="row">{{row}}</div>
</div>
```

# Try online yourself



Then open Chrome DevTools

# Virtual Scroll

## <div "style="transform: translateY(82px); ">

The screenshot shows a web application interface with a grid component and its underlying DOM structure in the browser's developer tools.

**Grid Component:** On the left, there is a table-like grid with columns labeled "Participant", "Game of Choice", "Name", "Language", "Country", and "Game Name". The data rows represent various participants and their preferences.

Participant	Game of Choice	Name	Language	Country	Game Name
Andrew Connell	Ireland	Chess			
	Sweden	Bul			
Kevin Flanagan	Spanish	Uruguay	Rithmomachy		
Bricker McGee	French	France	Kalah		
Dimple Unalkat	Portuguese	Portugal	Game of the Generals		
Gil Lopes	Spanish	Colombia	Hare and Hounds		
Sophie Beckham	English	Ireland	Sugoroku		
Isabelle Black	French	France	Nine Men's Morris		
Emily Braxton	Maltese	Malta	Blockade		
Olivia Brennan	French	France	Patolli		
Lily Brock	Italian	Italy	YINSH		
Chloe Bryson	Greek	Greece	Downfall		
Isabella Cadwell	English	Ireland	Gipf		
Amelia Cage	English	Ireland	Shogi		
Taylor Clark	Chinese	China	Ma		

**DevTools Elements Tab:** On the right, the browser's developer tools are open, specifically the "Elements" tab. It shows the DOM structure of the grid. Several CSS styles are highlighted with red boxes:

- A row with the class "ag-row-even.ag-row-no-focus.ag-row-level-0.ag-row-position-absolute.ag-row-not-inlin...". It has a style of "transform: translateY(0px); height: 41px;" and "style="transform: translateY(0px); height: 41px;"
- A row with the class "ag-row-odd.ag-row-no-focus.ag-row.ag-row-level-0.ag-row-position-absolute.ag-row-not-inlin...". It has a style of "transform: translateY(41px); height: 41px;"
- A row with the class "ag-row-even.ag-row-no-focus.ag-row.ag-row-level-0.ag-row-position-absolute.ag-row-not-inlin...". It has a style of "transform: translateY(82px); height: 41px;"

The DevTools also show the "Console" tab at the bottom, which is currently empty.

# AG-grid virtual-scroll

The screenshot shows the DevTools Elements tab for the URL [www.ag-grid.com/example/](http://www.ag-grid.com/example/). A specific element, `<div class="ag-body-vertical-scroll-viewport" ref="eViewport" style="width: 21px; max-width: 21px; min-width: 21px;">`, is selected. The left sidebar displays the element's properties: `div.ag-body-vertical-scroll-viewport` with dimensions `21 × 661.25`, `Color #181D1F`, and `Font 14px "IBM Plex Sans", -apple-system, Blin...`. The `ACCESSIBILITY` section shows the element has `Name`, `Role`, and is `Keyboard-focusable`. The element itself contains a tree view of column names, including `Name`, `Language`, `Country`, `Game of Choice`, `Game Name`, `Bought`, `Performance`, `Bank Balance`, `Extra Info 1`, `Extra Info 2`, `Rating`, `Total Winnings`, `Monthly Breakd`, `Jan`, `Feb`, `Mar`, and `Apr`. Below the tree view, there is a `Row Groups` section. The bottom status bar shows the path: `div.ag-layout-normal > div.ag-root.ag-unselectable.ag-layout-normal > div.ag-body.ag-layout-normal > div.ag-body-vertical-scroll`.

```
<div class="ag-pinned-right-cols-container ag-hidden" role="presentation" aria-hidden="true" style="height: 4.1e+06px; width: 0px; max-width: 0px; min-width: 0px;"></div>
<!-- AG Row Container fullWidth --&gt;
&lt;div class="ag-full-width-container" role="presentation" style="height: 4.1e+06px;"&gt;&lt;/div&gt;
&lt;/div&gt;
<!-- AG Fake Vertical Scroll --&gt;
&lt;div class="ag-body-vertical-scroll" aria-hidden="true" style="width: 21px; max-width: 21px; min-width: 21px;"&gt;flex == $0
  &lt;div class="ag-body-vertical-scroll-viewport" ref="eViewport" style="width: 21px; max-width: 21px; min-width: 21px;"&gt;...&lt;/div&gt;
&lt;/div&gt;
<!-- AG Sticky Top --&gt;
&lt;div class="ag-sticky-top" role="presentation" style="height: 0px; top: 142px; width: calc(100% - 21px);"&gt;...&lt;/div&gt; flex
&lt;/div&gt;
<!-- AG Pinned Bottom --&gt;
&lt;div class="ag-floating-bottom" role="presentation" style="height: 0px; min-height: 0px;"&gt;...&lt;/div&gt;</pre>
```

# AG-Grid Features

---

## CORE FEATURES

- Columns
- Rows
- Filtering
- Selection
- Rendering
- Editing
- Client-Side Data
- Scrolling
- Interactivity

## ADVANCED FEATURES

- Group & Pivot
- Master Detail
- Accessories
- Server-Side Data
- Import & Export
- Components

Demo Example : Sorting, Export to Excel

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Generating (Angular) Client from Swagger
  - Code Scaffolding
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- Final Words

# HighCharts

<https://www.highcharts.com>

# google HighCharts

Google search results for "highchart".

About 108,000,000 results (0.37 seconds)

**Highcharts**  
https://www.highcharts.com

**Highcharts: Interactive charting library**  
The Only Charting Library You Need. The Highcharts library comes with all the tools you need to create reliable and secure data visualizations. Built on ...

Results from highcharts.com

**Highcharts demos**  
Check out Highcharts demos and examples to learn how to create ...

**Highcharts Documentation**  
Charting ... Our core library. Includes all standard chart ...

**Download**  
Installation - Highcharts file service - Chart Chooser - Forum - Demo

**Highcharts Core Charting Library**  
Highcharts, the core library of our product suite, is a pure ...

People also ask :



**Highcharts**  
Software

Highcharts is a software library for charting written in pure JavaScript, first released in 2009. The license is proprietary. It is free for personal/non-commercial uses and paid for commercial applications. [Wikipedia](#)

# <https://www.highcharts.com/demo>

Highcharts Demos

Products Demos Resources Support Blog About Try for Free See Pricing Home / Demos Themes

**CORE Line charts**

- Line charts
- Area charts
- Column and bar charts
- Pie charts
- Scatter and bubble charts
- Combinations
- Styled mode (CSS styling)
- Accessibility
- Audio charts
- Dynamic charts
- 3D charts
- Gauges
- Heat and tree maps
- More chart types

**HIGHCHARTS STOCK**

- General
- Chart types
- Various features
- Flags and Technical Indicators

**HIGHCHARTS MAPS**

- General
- Dynamic
- Input formats

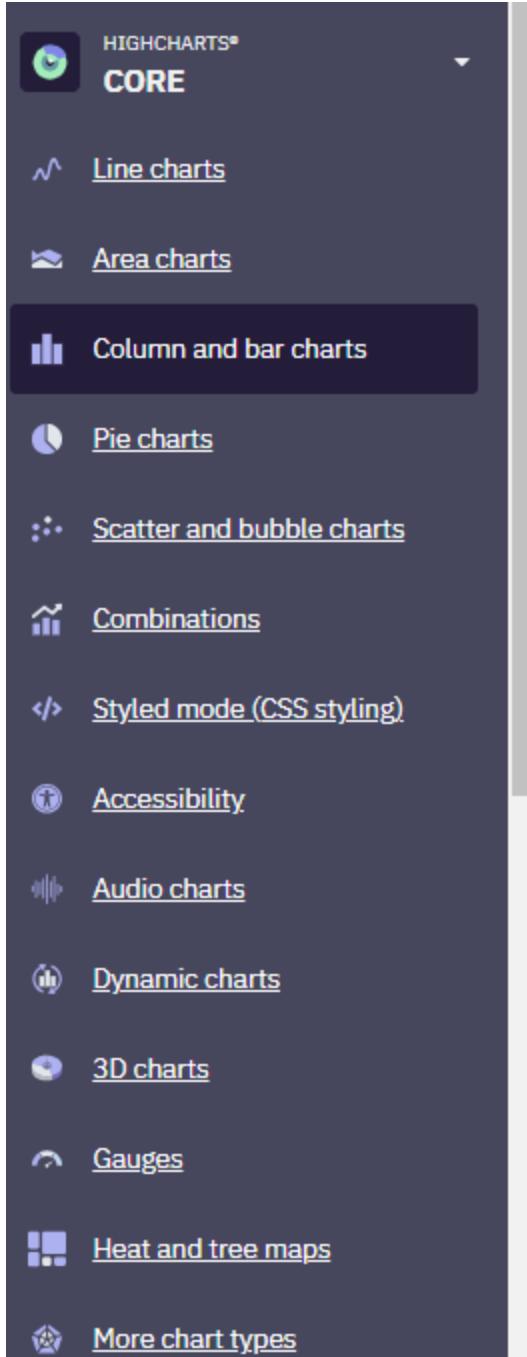
**Line charts**

- U.S Solar Employment Growth
- Monthly Average Temperature
- Monthly Average Temperature
- Atmosphere Temperature by Altitude
- Logarithmic axis demo
- United States of America's Inflation-related statistics
- Daily sessions at www.highcharts.com
- 2017 Tour de France Stage 5: Dole - Station des Rousses
- Line chart with 500k points
- Snow depth at Víkafjellet, Norway
- USD to EUR exchange rate over time
- Wind speed during a day

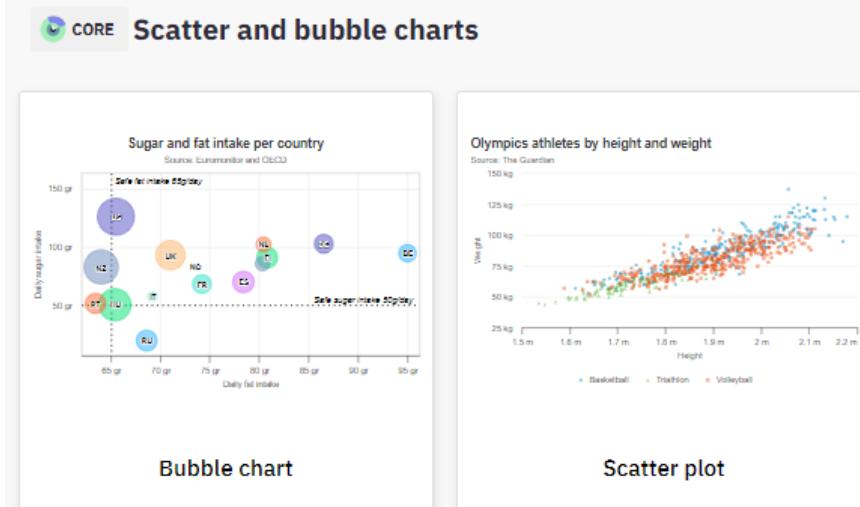
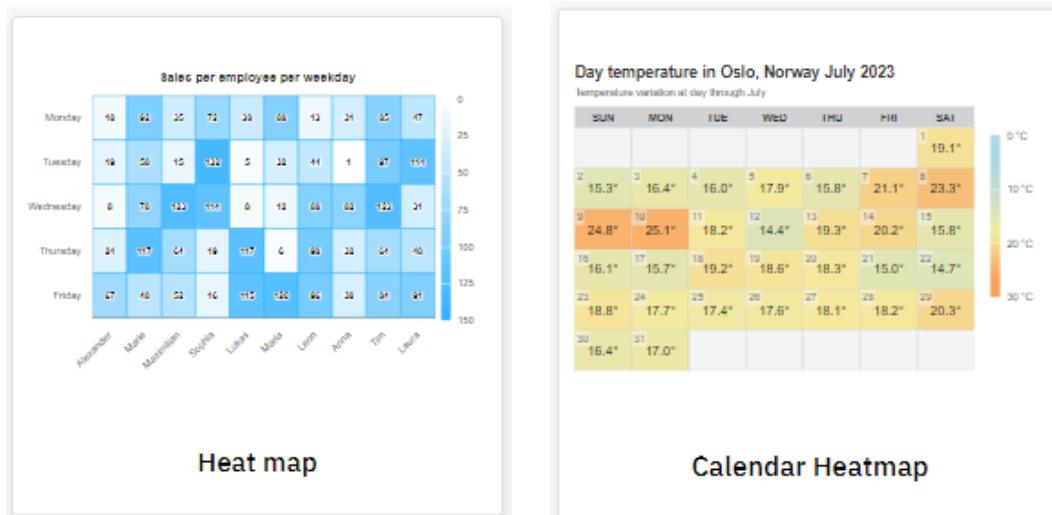
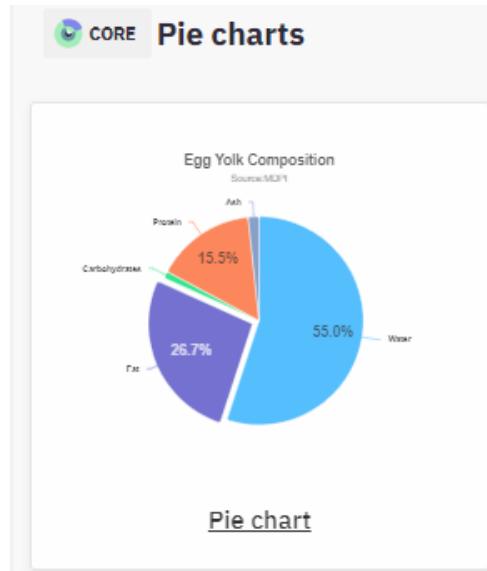
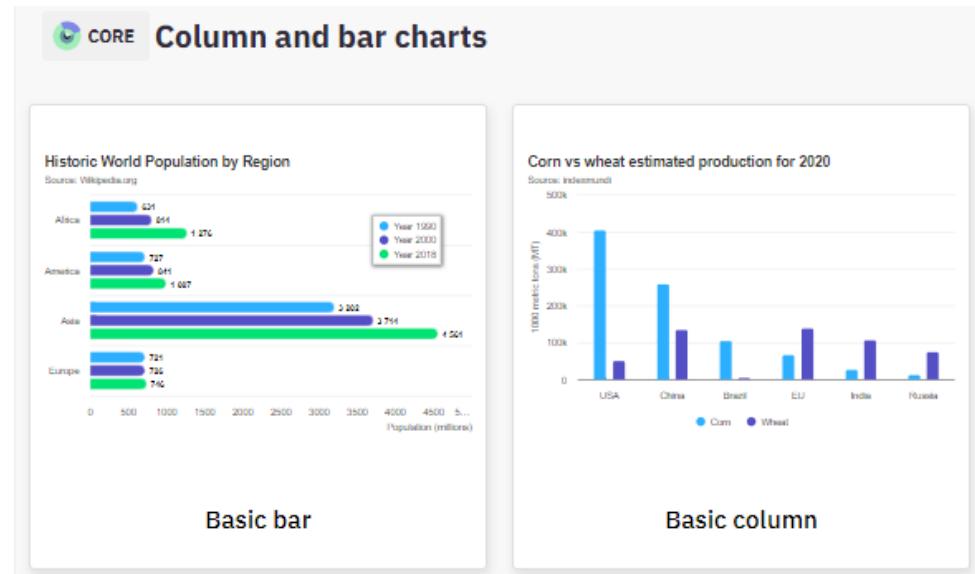
**Area charts**

- US and USSR nuclear stockpiles
- Greenhouse gases from Norwegian economic activity
- Countries/regions with highest Gt CO2-emissions
- Temperature variation by day

The Highcharts Demos page displays a grid of 16 different chart examples, each demonstrating a specific feature or dataset. The charts are organized into sections based on their type: Line charts, Area charts, and Combinations. Each chart includes a title, a brief description, and a link to its source. The 'Line charts' section contains 10 charts, including 'U.S Solar Employment Growth', 'Monthly Average Temperature' (Tokyo, Bergen), 'Atmosphere Temperature by Altitude', 'Logarithmic axis demo', 'United States of America's Inflation-related statistics', 'Daily sessions at www.highcharts.com', '2017 Tour de France Stage 5: Dole - Station des Rousses', 'Line chart with 500k points', 'Snow depth at Víkafjellet, Norway', 'USD to EUR exchange rate over time', and 'Wind speed during a day'. The 'Area charts' section contains 4 charts, including 'US and USSR nuclear stockpiles', 'Greenhouse gases from Norwegian economic activity', 'Countries/regions with highest Gt CO2-emissions', and 'Temperature variation by day'. The 'Combinations' section contains 2 charts, including 'Spline with symbols' and 'With data labels'. The sidebar on the left provides navigation links for different chart types and series, such as Area charts, Column and bar charts, Pie charts, Scatter and bubble charts, Combinations, and more.



# HighChart Features



<https://github.com/highcharts/highcharts-angular>

highcharts / highcharts-angular

Type ⌘ to search

Code Issues 9 Pull requests Actions Projects Security Insights

highcharts-angular Public Watch 31 Fork 115 Star 408

master 8 branches 25 tags Go to file Add file Code

 karolkolodziej	Changed name in the readme.	ce78091 4 days ago	224 commits
 .github/ISSUE_TEMPLATE	feat: Add issue templates. Closes #100.	5 years ago	
 e2e	updated to angular 11	3 years ago	
 highcharts-angular	chore(release): 4.0.0	3 weeks ago	
 js	feat: Added reference to @highcharts/map-collection and updated doc...	5 years ago	
 src	Updated highcharts.	last month	
 tasks	Updated producion build.	last year	
 .browserslistrc	Upadted demo to Angular 15.	last year	
 .editorconfig	Created branch for do-it-yourself.	6 years ago	
 .gitignore	Updated to Angular 13.	2 years ago	

About

Highcharts official integration for Angular

Readme View license Activity 408 stars 31 watching 115 forks Report repository

Releases 14

v4.0.0 Latest 3 weeks ago + 13 releases

# Getting Started

## npm install -s highcharts-angular

### Installing

Get package from NPM in your Angular app:

```
npm install highcharts-angular --save
```



In your app.module.ts add the HighchartsChartModule:

```
...
import { HighchartsChartModule } from 'highcharts-angular';

@NgModule({
  imports: [
    ...
    HighchartsChartModule
  ]
})
```



<https://github.com/highcharts/highcharts-angular#hello-world-demo>

## Hello world demo

To create a simple demo start with [installing](#).

Next for `app.component.ts`'s HTML template use:

```
<highcharts-chart
  [Highcharts]="Highcharts"
  [options]="chartOptions"

  style="width: 100%; height: 400px; display: block;"></highcharts-chart>
```

and export variables:

```
export class AppComponent {
  Highcharts: typeof Highcharts = Highcharts;
  chartOptions: Highcharts.Options = {
    series: [
      {
        data: [1, 2, 3],
        type: 'line'
      }
    ];
  ...
}
```

# Notice: Alternatives To HighCharts

**HTML <SVG>**

builtin features, but too low-level to use yourself

**HTML canvas**

**d3js**

also low-level and very complex

(redundant with Angular \*ngFor="let .. of ...")

**chart.js**

**& MANY others libraries**

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Generating (Angular) Client from Swagger
  - Code Scaffolding
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- Final Words

# Sharing (Partially) Codes / Same Technology for Backend and Frontend ?

Node  
**(reuse V8 !)**



Chrome V8  
JavaScript Engine



Chrome WebBrowser

Edge  
**(reuse V8 !)**



your-javascript-file-1.js  
your-javascript-file-2.js



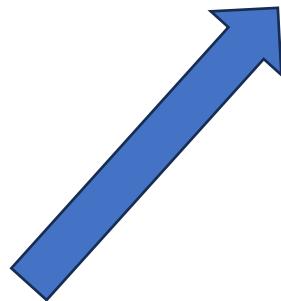
# Different Code ANYWAY !!

because Http Client != Http Server  
submit a Request != process a Request

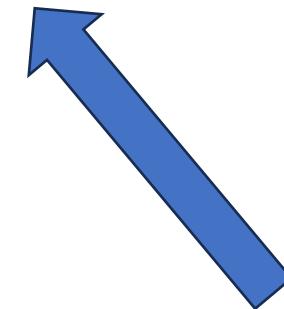
DTO classes can be fully shared

All Node-Express route methods must be translated to HttpClient

# Backend != FrontEnd



Long running Application Servers  
run on Linux  
in a DataCenter (OnPremise, AWS, Azure, ..)



Short interactive browsing  
run on your Windows Laptops  
or Android/iPhone SmartPhones

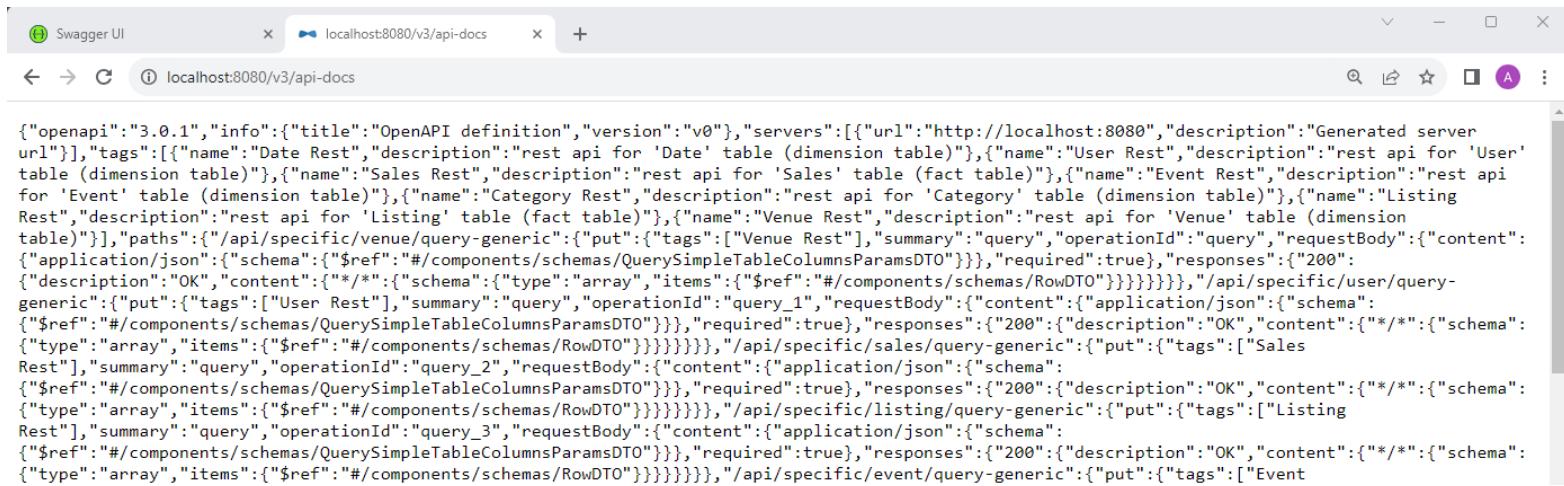
Most Probably in Java  
Because ... !!

On Chrome / FireFox / Edge /  
in JavaScript  
(because you have NO choice) !!

# Reminder OpenApi - swagger-ui.html

The screenshot shows the Swagger UI interface for an OpenAPI definition. The top navigation bar includes the title "Swagger" (Supported by SMARTBEAR), the URL "/v3/api-docs", and a green "Explore" button. Below the navigation is the main title "OpenAPI definition" with version "v0" and "OAS3" badges. A blue link below the title reads "/v3/api-docs". The "Servers" section contains a dropdown menu set to "http://localhost:8080 - Generated server url". The main content area is titled "Date Rest" and describes it as "rest api for 'Date' table (dimension table)". It lists two operations: a yellow-bordered "PUT /api/specific/date/query-generic query" and a blue-bordered "GET /api/specific/date list all". The "GET" operation has a "Parameters" section indicating "No parameters" and a "Try it out" button.

# Open Api description in json : /v3/api-docs



The screenshot shows a browser window titled "Swagger UI" with the URL "localhost:8080/v3/api-docs". The page displays a large amount of JSON code representing the OpenAPI specification for the "/v3/api-docs" endpoint. The JSON includes details about servers, tags, paths, and various API operations across different tables like Date, User, Sales, Event, Category, Listing, Venue, User, Sales, and Event.

```
{"openapi": "3.0.1", "info": {"title": "OpenAPI definition", "version": "v0"}, "servers": [{"url": "http://localhost:8080", "description": "Generated server url"}], "tags": [{"name": "Date Rest", "description": "rest api for 'Date' table (dimension table)"}, {"name": "User Rest", "description": "rest api for 'User' table (dimension table)"}, {"name": "Sales Rest", "description": "rest api for 'Sales' table (fact table)"}, {"name": "Event Rest", "description": "rest api for 'Event' table (dimension table)"}, {"name": "Category Rest", "description": "rest api for 'Category' table (dimension table)"}, {"name": "Listing Rest", "description": "rest api for 'Listing' table (fact table)"}, {"name": "Venue Rest", "description": "rest api for 'Venue' table (dimension table)"}], "paths": {"/api/specific/venue/query-generic": {"put": {"tags": ["Venue Rest"], "summary": "query", "operationId": "query", "requestBody": {"content": {"application/json": {"schema": {"$ref": "#/components/schemas/QuerySimpleTableColumnsParamsDTO"}}, "required": true}, "responses": {"200": {"description": "OK", "content": {"/*/*": {"schema": {"type": "array", "items": {"$ref": "#/components/schemas/RowDTO"}}}}}}}, "/api/specific/user/query-generic": {"put": {"tags": ["User Rest"], "summary": "query", "operationId": "query_1", "requestBody": {"content": {"application/json": {"schema": {"$ref": "#/components/schemas/QuerySimpleTableColumnsParamsDTO"}}, "required": true}, "responses": {"200": {"description": "OK", "content": {"/*/*": {"schema": {"type": "array", "items": {"$ref": "#/components/schemas/RowDTO"}}}}}}}, "/api/specific/sales/query-generic": {"put": {"tags": ["Sales Rest"], "summary": "query", "operationId": "query_2", "requestBody": {"content": {"application/json": {"schema": {"$ref": "#/components/schemas/QuerySimpleTableColumnsParamsDTO"}}, "required": true}, "responses": {"200": {"description": "OK", "content": {"/*/*": {"schema": {"type": "array", "items": {"$ref": "#/components/schemas/RowDTO"}}}}}}}, "/api/specific/listing/query-generic": {"put": {"tags": ["Listing Rest"], "summary": "query", "operationId": "query_3", "requestBody": {"content": {"application/json": {"schema": {"$ref": "#/components/schemas/QuerySimpleTableColumnsParamsDTO"}}, "required": true}, "responses": {"200": {"description": "OK", "content": {"/*/*": {"schema": {"type": "array", "items": {"$ref": "#/components/schemas/RowDTO"}}}}}}}, "/api/specific/event/query-generic": {"put": {"tags": ["Event Rest"], "summary": "query", "operationId": "query_4", "requestBody": {"content": {"application/json": {"schema": {"$ref": "#/components/schemas/QuerySimpleTableColumnsParamsDTO"}}, "required": true}, "responses": {"200": {"description": "OK", "content": {"/*/*": {"schema": {"type": "array", "items": {"$ref": "#/components/schemas/RowDTO"}}}}}}}}}
```

NOT to be read by Humans !!!!

either

=> in swagger-ui.html

OR

=> for Client Code Generators

# https://swagger.io

# Swagger Codegen

The screenshot shows the Swagger.io homepage. At the top, there's a navigation bar with links for "Why Swagger?", "Tools", "Resources", a search icon, "Sign In", and a "Try Free" button. Below the navigation, there's a large section for the "OpenAPI Specification". It features the title "OpenAPI Specification", the version "Version 3.1.0", and a note about key words like "MUST", "RECOMMENDED", and "NOT RECOMMENDED". It also mentions the BCP 14 RFC2119 RFC8174 standard and Apache License 2.0. A callout box highlights the "Pro" section which includes "SwaggerHub" (a collaborative platform for designing REST APIs), "SwaggerHub Enterprise" (a tool for standardizing APIs with projects, style checks, and reusable domains), and "SwaggerHub Explore" (an API explorer for evaluating functionality). To the right, there's a "Open Source" section featuring "Swagger Codegen" (a tool for generating server stubs and client SDKs from OpenAPI definitions), "Swagger Editor" (an API editor for OpenAPI and AsyncAPI specifications), and "Swagger UI" (an interactive UI for visualizing OpenAPI definitions). A sidebar on the right lists "Version 3.1.0", "Introduction", and "Table of Contents". At the bottom, there's a green button labeled "Explore all tools >".

swagger.io/specification/

Swagger  
Supported by SMARTBEAR

Why Swagger? Tools Resources

Sign In Try Free

## OpenAPI Specification

### Version 3.1.0

The key words "MUST", "MUST NOT", "REQUIRED", "RECOMMENDED", "NOT RECOMMENDED", "MAY" described in [BCP 14](#) [RFC2119](#) [RFC8174](#) when, and This document is licensed under [The Apache License 2.0](#).

## Introduction

The OpenAPI Specification (OAS) defines a standard, community-driven way to document APIs that is easy for humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection. When properly defined, a consumer can understand and interact with the remote service with a minimal amount of implementation logic.

Pro

SwaggerHub

Design & document all your REST APIs in one collaborative platform.

SwaggerHub Enterprise

Standardize your APIs with projects, style checks, and reusable domains.

SwaggerHub Explore

Instantly evaluate the functionality of any API

Open Source

Swagger Codegen

Generate server stubs and client SDKs from OpenAPI Specification definitions

Swagger Editor

API editor for designing APIs with the OpenAPI and AsyncAPI specifications.

Swagger UI

Visualize OpenAPI Specification definitions in an interactive UI.

Version 3.1.0

Introduction

Table of Contents

Definitions

Specification

Appendix A: Revision History

Explore all tools >

# Swagger CodeGen demo (maven plugin)

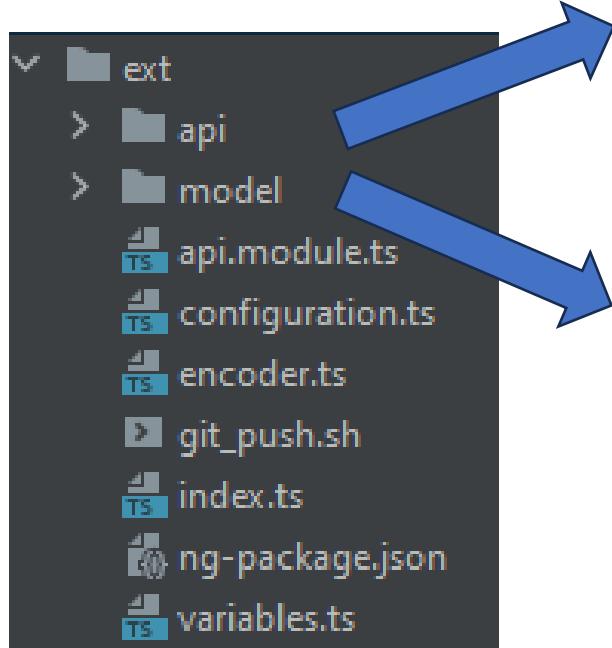
```
<plugin>
  <groupId>io.swagger.codegen.v3</groupId>
  <artifactId>swagger-codegen-maven-plugin</artifactId>
  <version>3.0.47</version>
  <configuration>
    <inputSpec>http://localhost:8080/v3/api-docs</inputSpec>
    <language>typescript-angular</language>
    <output>${basedir}/target/generated-typescript-angular16</output>
    <configOptions>
      <ngVersion>16</ngVersion>
    </configOptions>
  </configuration>
  <executions>
    <execution>
      <id>generate-swagger-typescript-angular-16</id>
      <phase>generate-sources</phase>
      <goals>
        <goal>generate</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

# Swagger CodeGen Demo

```
$ mvn swagger-codegen:generate
```

```
$ rm -rf ./ui/src/app/ext/*; rm -rf target/generated-typescript-angular16; mvn -Pswagger-gen swagger-codegen:generate; cp -rf target/generated-typescript-angular16/* ./ui/src/app/ext/
[INFO] Scanning for projects...
[INFO]
[INFO] -----< fr.an.test.spark-db-app:server >-----
[INFO] Building server 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- swagger-codegen-maven-plugin:3.0.47:generate (default-cli) @ server ---
[WARNING] Output directory does not exist, or is inaccessible. No file (.swagger-codegen-ignore) will be evaluated.
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\aliasedExprTextDTO.ts
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\arrayTypeDTO.ts
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\binaryOpExprDTO.ts
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\binaryTypeDTO.ts
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\booleanTypeDTO.ts
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\byteTypeDTO.ts
[INFO] writing file C:\arn\devPerso\test-spark-rest-server\server\target\generated-typescript-angular16\model\categoryDTO.ts
```

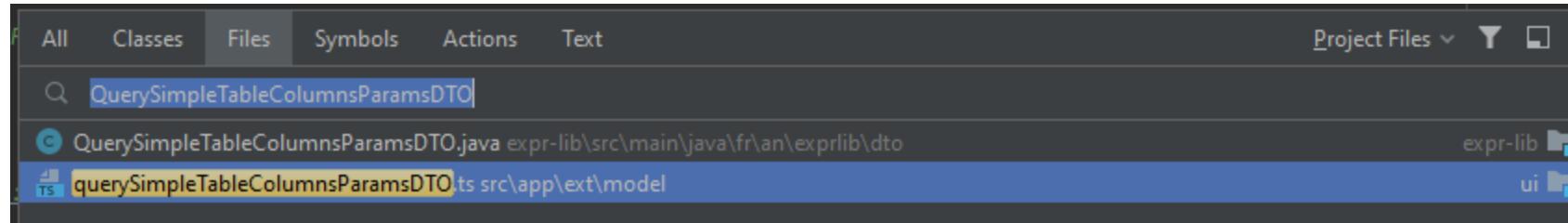
# Generated code for Angular-Typescript



contains all Services for http endpoints  
( = for @RestController in backend )  
each PUT/POST/GET/DELETE  
=> correspond to a typescript method

contains all interfaces describing datatypes  
( = for class \*DTO in backend )  
each JSON request body / response type  
=> correspond to a typescript interface

# The front-end code is ALWAYS IN-SYNC with backend code!



```
/**  
 * OpenAPI definition  
 * No description provided (generated by Swagger Codegen https://github.com/swagger-api/swagger-codegen)  
 *  
 * OpenAPI spec version: v0  
 *  
 *  
 * NOTE: This class is auto generated by the swagger code generator program.  
 * https://github.com/swagger-api/swagger-codegen.git  
 * Do not edit the class manually.  
 */
```

# Example CodeGen

in Java (SpringBoot) backend



Generated TypeScript for Angular

Rest Controller class

```
@RestController  
@RequestMapping(path = "/api/dispatch-table/{tableName}")  
public class DispatchGenericQueryRestController {
```



```
@Injectable()  
export class DispatchGenericQueryRestControllerService {  
  
    protected basePath: string = 'http://localhost:8080';  
    public defaultHeaders: HttpHeaders = new HttpHeaders();  
    public configuration: Configuration = new Configuration();
```

http PUT method endpoint

```
@PutMapping("/query-simple-cols")  
@Operation(summary = "query simple columns")  
public List<RowDTO> query(  
    @PathVariable("tableName") String tableName,  
    @RequestBody QuerySimpleTableColumnsParamsDTO req) {
```



```
public query7(body: QuerySimpleTableColumnsParamsDTO, tableName: string,  
            observe?: 'body', reportProgress?: boolean  
): Observable<Array<RowDTO>>;
```

(cf next slide for detail implementation)

DTO type for request body

```
@Data  
public class QuerySimpleTableColumnsParamsDTO {  
    public List<String> cols;  
    public int limitCount;  
}
```



```
export interface QuerySimpleTableColumnsParamsDTO {  
    cols?: Array<string>;  
    limitCount?: number;  
}
```

# (glory details : Typescript HttpClient generated code )

```
public query7(body: QuerySimpleTableColumnsParamsDTO, tableName: string, observe?: 'body', reportProgress?: boolean): Observable<Array<RowDTO>>;
public query7(body: QuerySimpleTableColumnsParamsDTO, tableName: string, observe?: 'response', reportProgress?: boolean): Observable<HttpResponse<Array<RowDTO>>>;
public query7(body: QuerySimpleTableColumnsParamsDTO, tableName: string, observe?: 'events', reportProgress?: boolean): Observable<HttpEvent<Array<RowDTO>>>;
no usages
public query7(body: QuerySimpleTableColumnsParamsDTO, tableName: string, observe: any = 'body', reportProgress: boolean = false ): Observable<any> {
    if (body === null || body === undefined) {
        throw new Error( message: 'Required parameter body was null or undefined when calling query7.' );
    }
    if (tableName === null || tableName === undefined) {
        throw new Error( message: 'Required parameter tableName was null or undefined when calling query7.' );
    }
    let headers : HttpHeaders = this.defaultHeaders;
    // to determine the Accept header
    let httpHeaderAccepts: string[] = [
        'application/json'
    ];
    const httpHeaderAcceptSelected: string | undefined = this.configuration.selectHeaderAccept(httpHeaderAccepts);
    if (httpHeaderAcceptSelected != undefined) {
        headers = headers.set('Accept', httpHeaderAcceptSelected);
    }
    // to determine the Content-Type header
    const consumes: string[] = [
        'application/json'
    ];
    const httpContentTypeSelected: string | undefined = this.configuration.selectHeaderContentType(consumes);
    if (httpContentTypeSelected != undefined) {
        headers = headers.set('Content-Type', httpContentTypeSelected);
    }
    return this.httpClient.request<Array<RowDTO>>( method: 'put', url: `${this.basePath}/api/dispatch-table/${encodeURIComponent(String(tableName))}/query-simple-cols` ,
        options: {
            body: body,
            withCredentials: this.configuration.withCredentials,
            headers: headers,
            observe: observe,
            reportProgress: reportProgress
        }
    );
}
```

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Generating (Angular) Client from Swagger
  - Code Scaffolding
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- Final Words

# Code Scafolding

\$ ng new

\$ ng g c = \$ ng generate component

Your first scafolding tools !  
a.k.a Code Generator

# More Scaffolding.. Yeoman

## <https://yeoman.io>

The screenshot shows the official Yeoman website at [yeoman.io](https://yeoman.io). The header features a navigation bar with links for "Using Yeoman", "Discovering generators", "Creating a generator", "Blog", and "Contributing". To the left of the main content area is the Yeoman logo, which is a cartoon character wearing a top hat and a monocle. The main title "THE WEB'S SCAFFOLDING TOOL FOR MODERN WEBAPPS" is displayed in large white text against a teal background. To the right of the title is a colorful illustration of three characters working on a large rocket engine: a woman in a lab coat, a man in a white coat, and a Yeoman character. Below the main section is a grey box containing instructions and a command-line example.

Get started and then [find a generator](#) for your webapp.  
Generators are available for [Angular](#), [Backbone](#), [React](#), [Polymer](#) and over [5600+ other projects](#).

One-line install using [npm](#):

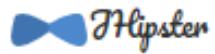
```
npm install -g yo
```

# More Scafolding ... JHipster

<https://jhipster.tech>

The screenshot shows the JHipster website homepage. The URL 'jhipster.tech' is visible in the browser's address bar. The page features a dark blue header with navigation links: LEARN, SPONSORS, ABOUT, OPTIONS, MODULES & BLUEPRINTS, JDL STUDIO, GITHUB, and YOUTUBE. A search bar is also present in the header. To the left of the main content area is a vertical sidebar with various icons, likely representing different tools or services. The main content area has a blue background. In the center, there is a white circular icon containing a cartoon illustration of a woman with blonde hair, wearing a black blazer over a white shirt and a blue bow tie, holding a coffee cup. Below this icon, the text 'Greetings, Java Hipster!' is displayed in a large, white, cursive font. A horizontal line separates this section from the explanatory text below. The explanatory text reads: 'JHipster is a development platform to quickly generate, develop, & deploy modern web applications & microservice architectures.' Another horizontal line follows. Further down, it says: 'JHipster is on [Open Collective](#), if you ❤️ JHipster consider becoming a [sponsor](#) or a [backer](#)'. At the bottom of the page, there are four dark blue footer cards with white text: a download icon followed by '115813 Downloads in last 30 days', a GitHub icon followed by '21001 Stars', a people icon followed by '600+ Contributors', and a small 'Search...' input field.

# JHipster in Few minutes > Technology Stack



- [Home](#)
- [JHipster Tech Board](#)
- [Release notes](#)
- [JHipster in a few minutes](#)
  - [Technology stack](#)
  - [Official JHipster slides](#)
  - [JHipster in 5 screenshots](#)
  - [Video tutorial \(15 minutes\)](#)
- [Online guides](#)
- [Companies using JHipster](#)
- [Showcase of JHipster apps](#)

## Technology stack

### Technology stack on the client side

Single Web page application:

- [Angular or React or Vue](#)
- [Responsive Web Design with Twitter Bootstrap](#)
- [HTML5 Boilerplate](#)
- Compatible with modern browsers (Chrome, FireFox, Microsoft Edge...)
- Full internationalization support
- Optional [Sass](#) support for CSS design
- Optional WebSocket support with Spring Websocket

With the great development workflow:

- Installation of new JavaScript libraries with [NPM](#)
- Build, optimization and live reload with [Webpack](#)
- Testing with [Jest](#) and [Protractor](#)

And what if a single Web page application isn't enough for your needs?

- Support for the [Thymeleaf](#) template engine, to generate Web pages on the server side

### Technology stack on the server side

A complete [Spring](#) application:

- [Spring Boot](#) for application configuration
- [Maven](#) or [Gradle](#) configuration for building, testing and running the application
- “development” and “production” profiles (both for Maven and Gradle)
- [Spring Security](#)
- [Spring MVC REST + Jackson](#)
- Optional WebSocket support with [Spring Websocket](#)
- [Spring Data JPA + Bean Validation](#)
- Database updates with [Liquibase](#)
- [Elasticsearch](#) support if you want to have search capabilities on the database
- [MongoDB](#) and [Couchbase](#) support if you'd rather use a document-oriented NoSQL database
- [Cassandra](#) support if you'd rather use a column-oriented NoSQL database
- [Kafka](#) and [Pulsar](#) support if you want to use a publish-subscribe interface

# JHipster Slides - Front End Technologies

<https://www.jhipster.tech/presentation/#/>

## Client-side technologies

YEOMAN, WEBPACK, ANGULAR, REACT, BOOTSTRAP

### NPM

- Fast and reliable dependency management
- Used to install and run all client-side tools

```
npm install -g generator-jhipster
```

## Yeoman

YEOMAN PROVIDES APPLICATION GENERATORS

- Hundreds of generators are available
- Mostly geared toward JavaScript front-end applications
- The top-rated generators have excellent quality

jhipster

## Angular and React

THE 2 MOST POPULAR JAVASCRIPT FRAMEWORKS

- Both Angular and React are supported by JHipster
- Powerful & easy to learn
- Data binding, form validation, i18n... all out of the box

# JHipster in 5 Screenshots: 1/5

The screenshot shows the JHipster website's homepage. On the left is a dark sidebar with a blue bowtie logo and a navigation menu. The menu items include: Home, JHipster Tech Board, Release notes, JHipster in a few minutes (with a dropdown arrow), Technology stack, Official JHipster slides, JHipster in 5 screenshots (which is highlighted with a white background), Video tutorial (15 minutes), Online guides, Companies using JHipster, Showcase of JHipster apps, Contributing (with a dropdown arrow), Setting up your environment (with a dropdown arrow), and Core JHipster tasks (with a dropdown arrow). On the right is the main content area. It features a large heading "Screenshots of the generated application" with a camera icon, followed by "The welcome screen". Below this is a screenshot of a web browser showing the "Development-Tmp v0.0.0" application. The browser header includes "Home", "Language", and "Account". The main content of the browser shows a cartoon character of a man with glasses and a bowtie, holding a coffee cup. The text "Welcome, Java Hipster!" is prominently displayed, along with "This is your homepage". A yellow callout box contains instructions: "If you want to **sign in**, you can try the default accounts:  
- Administrator (login="admin" and password="admin")  
- User (login="user" and password="user")." Another yellow callout box below it says "You don't have an account yet? **Register a new account**". Further down, there is a section titled "If you have any question on JHipster:" with a bulleted list: "JHipster homepage", "JHipster on Stack Overflow", "JHipster bug tracker", "JHipster public chat room", and "follow @java\_hipster on Twitter". At the bottom of the page, a footer note says "If you like JHipster, don't forget to give us a star on [GitHub](#)!". The very bottom of the page has a footer note "This is your footer".

# JHipster in 5 Screenshots: 2/5

The generated “Book” entity

This screenshot shows a web application interface for managing a 'Books' entity. The top navigation bar includes links for Home, Entities, Administration, Language, and Account. A red ribbon banner on the left says 'DevelopmentTmp v0.0.0'. On the right, there's a blue button to 'Create a new Book'. The main content area displays a table with three rows of book data. Each row includes columns for ID, Title, Description, Publication Date, Price, Author, and actions (View, Edit, Delete). The first book listed is 'Inferno' by Dan Brown, published on May 14, 2013, at a price of 45. The second book is 'King Lear' by William Shakespeare, published on Dec 30, 1604, at a price of 10. The third book is 'Gulliver's Travels' by Jonathan Swift, published on Oct 26, 1726, at a price of 15. At the bottom, it says 'Showing 1 - 3 of 3 items.' and has a page navigation bar with icons for back, forward, and search.

ID	Title	Description	Publication Date	Price	Author	Action
2	Inferno	Inferno is a 2013 mystery thriller novel by American author Dan Brown and the fourth book in his Robert Langdon series, following Angels & Demons, The Da Vinci Code and The Lost Symbol.	May 14, 2013	45	Dan Brown	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
4	King Lear	King Lear is a tragedy written by William Shakespeare. It depicts the gradual descent into madness of the title character.	Dec 30, 1604	10	William Shakespeare	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
5	Gulliver's Travels	It is Swift's best known full-length work, and a classic of English literature.	Oct 26, 1726	15	Jonathan Swift	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>

Showing 1 - 3 of 3 items.

# JHipster in 5 Screenshots: 3/5

One of the generated forms

The screenshot shows a web application interface for managing user settings. At the top, there's a dark header bar with a logo on the left and navigation links for Home, Entities, Administration, Language, and Account. A red diagonal banner on the left side of the main content area says "DevelopmentTmp v0.0.0". The main content is titled "User settings for [admin]". It contains four input fields: "First Name" with value "Julien", "Last Name" with value "Dubois", "E-mail" with placeholder "Your e-mail" and a red validation message "Your e-mail is required.", and "Language" with value "English". A blue "Save" button is at the bottom. At the very bottom of the page, there's a footer message: "This is your footer".

User settings for [admin]

First Name

Julien

Last Name

Dubois

E-mail

Your e-mail

Your e-mail is required.

Language

English

Save

This is your footer

# JHipster in 5 Screenshots: 4/5

The monitoring screen

The screenshot shows the JHipster monitoring interface. At the top, there's a navigation bar with links for Home, Entities, Administration, Language, and Account, along with a Refresh button. Below the navigation is a section titled "Application Metrics" which includes "JVM Metrics". This section displays various performance metrics with corresponding progress bars:

Memory	Threads (Total: 48)	Garbage collections
Total Memory (634M / 3,818M)	Runnable 15	Mark Sweep count 3
		Mark Sweep time 406ms
Heap Memory (511M / 3,818M)	Timed waiting (5)	Scavenge count 11
		Scavenge time 168ms
Non-Heap Memory (123M / 126M)	Waiting (28)	
	Blocked (0)	

Below the JVM metrics is a section for "HTTP requests (events per second)". It shows active requests (1) and total requests (23), followed by a table of request statistics:

Code	Count	Mean	Average (1 min)	Average (5 min)	Average (15 min)
Ok		0.35	1.14	2.32	2.63

# JHipster in 5 Screenshots: 5/5

## The log management screen

The screenshot shows the 'Logs' section of a JHipster application. At the top, there's a navigation bar with links for Home, Entities, Administration, Language, and Account. A red banner on the left says 'Development: Tmp v6.0'. Below the banner, the word 'Logs' is displayed in a large, bold font. A message indicates there are 1326 loggers. A 'Filter' input field contains the text 'springframework.security'. A table lists several loggers with their names and current log levels:

Name	Level
org.springframework.security	TRACE DEBUG INFO <b>WARN</b> ERROR
org.springframework.security.access	<b>TRACE</b> DEBUG INFO WARN ERROR
org.springframework.security.access.annotation	TRACE <b>DEBUG</b> INFO WARN ERROR
org.springframework.security.access.annotation.SecuredAnnotationSecurityMetadataSource	TRACE DEBUG INFO <b>WARN</b> ERROR
org.springframework.security.access.expression	TRACE DEBUG <b>INFO</b> WARN ERROR
org.springframework.security.access.expression.DenyAllPermissionEvaluator	TRACE DEBUG <b>INFO</b> WARN ERROR
org.springframework.security.access.expression.method	TRACE DEBUG INFO <b>WARN</b> <b>ERROR</b>

# JHipster Demo (case 1/ with local installation)

```
$ npm install -g generator-jhipster
```

```
$ jhipster  
.... prompt....
```

```
$ jhipster add ...  
$ jhipster jdl <<your-file>>.jdl
```

# case 2 / with JHipster Online

<https://start.jhipster.tech/generate-application>

The screenshot shows the JHipster Online application generation interface. On the left, there's a sidebar with links for 'Create Application', 'Create Azure Application', 'JDL Studio', and 'JHipster Statistics'. The main area is titled 'Application generation' and 'Project configuration'. It includes fields for 'Application name' (set to 'jhipsterSampleApplication'), 'Repository name' (set to 'jhipster-sample-application'), and a dropdown for 'Application type' (set to 'Monolithic application (recommended for simple projects)'). Under 'Server side options', the 'Default Java package name' is set to 'com.mycompany.mvann'. At the bottom, there are 'Download as Zip file' and 'Reset' buttons.

JHipster Online v2.26.0

JHipster Online is the best place to generate **JHipster** applications, with no installation required!

No more command line: fill an easy-to-use Web form, JHipster Online generates a complete **JHipster** application on your GitHub account.

Check out **JHipster statistics** to have an overview of what JHipster users are building.

**Create Application**

**Create Azure Application**

**JDL Studio**

**JHipster Statistics**

**Application generation**

Project configuration

Application name

jhipsterSampleApplication

Repository name

jhipster-sample-application

Application type

Which type of application would you like to create?

Monolithic application (recommended for simple projects)

Server side options

What is your default Java package name?

com.mycompany.mvann

Download as Zip file

Reset

# <https://start.jhipster.tech/jdl-studio>

start.jhipster.tech/jdl-studio/

## JDL-Studio

Please sign in for more features! [Home](#) | [Logout](#) [Download](#) [Upload](#) [Help](#) [Feedback](#)

```
1
2 entity Region {
3     regionName String
4 }
5
6 entity Country {
7     countryName String
8 }
9
10 // an ignored comment
11 /** not an ignored comment */
12 entity Location {
13     streetAddress String,
14     postalCode String,
15     city String,
16     stateProvince String
17 }
18
19 entity Department {
20     departmentName String required
21 }
22
23 /**
24 * Task entity.
25 * @author The JHipster team.
26 */
```

The diagram illustrates the JDL entities and their associations:

- Language** (enum): FRENCH, ENGLISH, SPANISH.
- JobHistory**: startDate, endDate, language.
- Department**: departmentName (required).
- Location**: streetAddress, postalCode, city, stateProvince.
- Employee**: firstName, lastName, email, phoneNumber, hireDate, salary, commissionPct.
- Job**: jobTitle, minSalary, maxSalary.
- Task**: title, description.
- Country**: countryName.
- Region**: regionName.

Relationships:

- Language** is associated with **JobHistory**.
- JobHistory** is associated with **Department** (multiplicity 1..1).
- Department** is associated with **Location** (multiplicity 1..1) and **Employee** (multiplicity 1..\*).
- Location** is associated with **Country** (multiplicity 1..4).
- Employee** is associated with **Job** (multiplicity 1..\*).
- Job** is associated with **Task** (multiplicity \*..\*).
- Country** is associated with **Region** (multiplicity 1..1).

# Step 1/ Compile backend

\$ mvn package

```
$ mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.mycompany.myapp:jhipster-sample-application >-----
[INFO] Building Jhipster Sample Application 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----

[INFO] The original artifact has been renamed to C:\web\demo-jhipster\jhipster-sample-application-0.0.1-SNAPSHOT.jar.original
[INFO]
[INFO] --- modernizer-maven-plugin:2.7.0:modernizer (modernizer) @ jhipster-sample-application ---
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  6.574 s
[INFO] Finished at: 2023-12-02T18:08:06+01:00
[INFO] -----
```

# Step 2/ Run backend

```
$ mvn spring-boot:run -Pdev
```

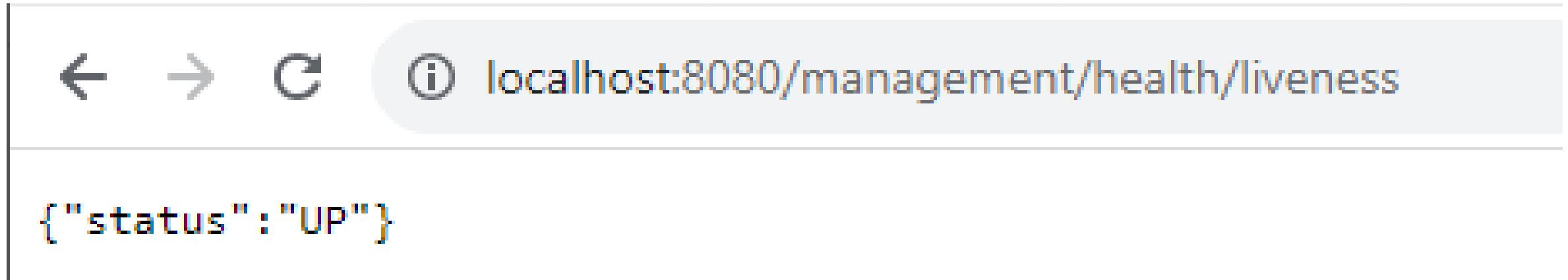
```
$ mvn spring-boot:run -Pdev
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.mycompany.myapp:jhipster-sample-application >-----
[INFO] Building Jhipster Sample Application 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> spring-boot-maven-plugin:3.1.5:run (default-cli) > test-compile @ jhipster-sample-application >>>
```

```
2023-12-02T18:13:20.679+01:00  INFO 13436 --- [  restartedMain] c.m.myapp.JhipsterSampleApplicationApp      : Started
n 6.889 seconds (process running for 7.296)
2023-12-02T18:13:20.684+01:00  INFO 13436 --- [  restartedMain] c.m.myapp.JhipsterSampleApplicationApp      :
-----
Application 'jhipsterSampleApplication' is running! Access URLs:
Local:          http://localhost:8080/
External:       http://192.168.0.10:8080/
Profile(s):    [dev, api-docs]
-----
```

## Step 3/ Test Backend

http://localhost:8080/management/health

```
$ curl http://localhost:8080/management/health/liveness
{"status": "UP"}
```



# Step 4/ compile FrontEnd

\$ npm install  
\$ npm start

```
$ npm install  
[          ] \ idealTree:jhipsterSampleApplication: sill idealTree buildDeps
```

```
added 2162 packages, and audited 2163 packages in 4m
```

```
306 packages are looking for funding  
  run `npm fund` for details
```

```
8 moderate severity vulnerabilities
```

# Step 5/ Start the FrontEnd

## \$ npm start

```
$ npm start
```

```
> jhipster-sample-application@0.0.1-SNAPSHOT start
> ng serve --hmr
```

```
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
```

```
✓ Compiled successfully.
Notifications are disabled
Reason: DisabledByGroupPolicy Please make sure that the app id is set correctly.
Command Line: C:\web\demo-jhipster\jhipsterSampleApplication\node_modules\node-notifier\vendor\snoreToast\snoretoast
notifierPipe-25256b7e-89fb-4008-9ccb-d78e197f65e4 -p C:\web\demo-jhipster\jhipsterSampleApplication\webpack\logo-jh
" -t "Jhipster Sample Application"
[Browsersync] Proxying: http://localhost:4200
[Browsersync] Access URLs:
-----
Local: http://localhost:9000
External: http://192.168.0.10:9000
-----
UI: http://localhost:3001
UI External: http://localhost:3001
-----
```

# Step 6/ open Browser

## http://localhost:4200

A screenshot of a web browser window displaying the JHipster Sample Application homepage at localhost:9000. The page features a dark header with the application name and a navigation bar with icons for download, search, and account. On the left, there's a cartoon illustration of a hipster man with a beard, wearing a beret, a plaid shirt, and brown suspenders. The main content area has a yellow background. It displays a welcome message, information about default accounts, a link to register a new account, and links for further assistance. At the bottom, there's a footer section.

localhost:9000

JhipsterSampleApplication vDEV Home Account

## Welcome, Java Hipster! (Jhipster Sample Application)

This is your homepage

If you want to [sign in](#), you can try the default accounts:

- Administrator (login="admin" and password="admin")
- User (login="user" and password="user").

You don't have an account yet? [Register a new account](#)

If you have any question on JHipster:

- [JHipster homepage](#)
- [JHipster on Stack Overflow](#)
- [JHipster bug tracker](#)
- [JHipster public chat room](#)
- [follow @jhipster on Twitter](#)

If you like JHipster, don't forget to give us a star on [GitHub](#)!

This is your footer

# Step 7/ login as "admin/admin" or "user/user"

The screenshot shows a web application interface. At the top is a dark header bar with the following navigation items from left to right: a house icon labeled "Home", a grid icon labeled "Entities" with a dropdown arrow, a user icon labeled "Administration" with a dropdown arrow, and a person icon labeled "Account" with a dropdown arrow. Below this is the main content area. On the left, there is a large heading "Welcome, Java Hipster! (Jhipster Sample Application)" and a subtext "This is your homepage". On the right, a vertical sidebar menu is open, listing several options with corresponding icons: "User management" (people icon), "Metrics" (gauge icon), "Health" (heart icon), "Configuration" (cog icon), "Logs" (log icon), "API" (api icon), and "Database" (database icon). A green footer bar at the bottom contains the text "You are logged in as user \"admin\"".

Welcome, Java Hipster! (Jhipster Sample Application)

This is your homepage

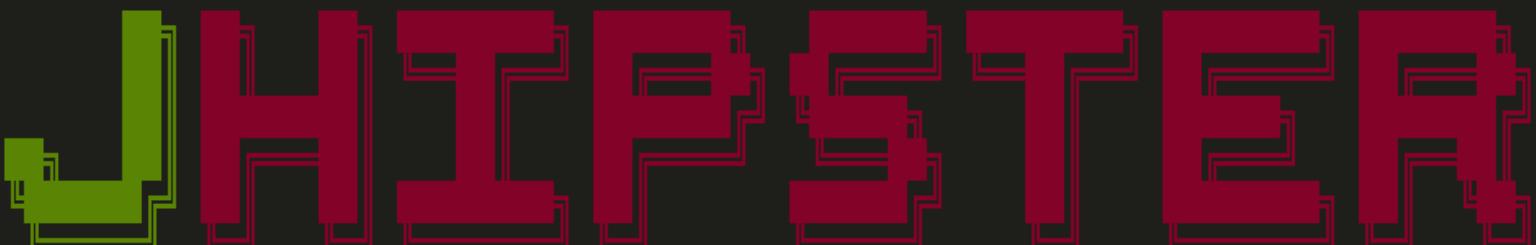
You are logged in as user "admin".

- User management
- Metrics
- Health
- Configuration
- Logs
- API
- Database

# Step 8/ import jdl ... recompile & restart

## \$ jhipster jdl \${jdlFile}

```
$ jhipster jdl ..../jhipster-jdl.jdl
WARNING! Since JHipster v8, the jhipster command will not use the locally installed generator-jhipster.
If you want to execute the locally installed generator-jhipster, run: npx jhipster


https://www.jhipster.tech
Welcome to JHipster v8.0.0

INFO! Generating jdls ..../jhipster-jdl.jdl
INFO! Generating entities Region Country Location Department Task Employee Job JobHistory
```

```
create src\test\java\com\mycompany\myapp\web\rest\LocationResourceIT.java
create src\test\java\com\mycompany\myapp\domain\EmployeeTestSamples.java
create src\test\java\com\mycompany\myapp\domain\JobTest.java
create src\main\webapp\app\entities\enumerations\language.model.ts
create src\main\webapp\app\entities\region\service\region.service.ts
create src\main\webapp\app\entities\country\update\country-update.component.spec.ts
create src\main\webapp\app\entities\region\region.model.ts
create src\main\webapp\app\entities\country\detail\country-detail.component.ts
create src\main\webapp\app\entities\region\service\region.service.spec.ts
```

# Step 9/ navigate Entities Table - View - EditForm

The screenshot illustrates the JHipster Sample Application interface. On the left, a sidebar menu lists entities: Region, Country, Location, Department, Task, Employee, Job, and Job History. The 'Department' item is selected and highlighted with a black background. The main content area shows a table of 'Departments' with columns: ID, Department Name, and Location. The table contains 6 rows, each with a 'View', 'Edit', and 'Delete' button. Below the table, a modal dialog titled 'Create or edit a Department' is open. It contains fields for 'ID' (set to 1), 'Department Name' (set to 'dept 1'), and 'Location' (set to 1). At the bottom of the dialog are 'Cancel' and 'Save' buttons.

Home Entities Administration Account

- \* Region
- \* Country
- \* Location
- \* Department**
- \* Task
- \* Employee
- \* Job
- \* Job History

JHipsterSampleApplication vDEV

Departments

ID	Department Name	Location
1	boastfully fumbling supposing	
2	ponder finally unfortunately	
3	rosy bubbly inasmuch	
4	plus arid	
5	sour	
6	ick boohoo obvious	

Create or edit a Department

ID  
1

Department Name  
dept 1

Location  
1

Cancel Save

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Code Scaffolding
  - Generating (Angular) Client from Swagger
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- Final Words

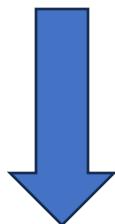
Security ... should not be done half way

Leave it to Experts

# Security Basics

## Encryption

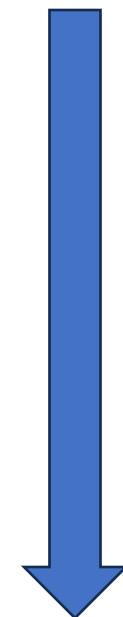
Confidentiality  
No Man-In-The-Middle



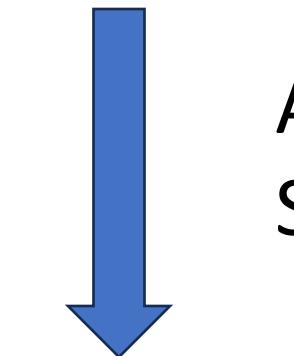
**https://**  
**certificate Trust**

## Authentication

Delegated  
Authentication



**user  
password**



**OAuth2  
flows**



**JWT  
token  
(Cookie)**

## Authorization

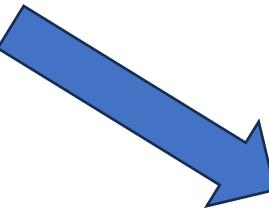
relation in code  
Resource  
-> Permission  
-> Group -> Users



## Http Headers

"Authorization=Basic <user:pass>"  
"Authorization=Bearer <token>"

```
$ curl -v -u user:pass http://localhost:8080
```



password in Base64 (~clear text) on Http Header

# Demo "Basic <Base64(user:password)>"

```
$ curl -v -u user:user http://localhost:8080/
* Trying [::1]:8080...
* Connected to localhost (::1) port 8080
* Server auth using Basic with user 'user'
> GET / HTTP/1.1
> Host: localhost:8080
> Authorization: Basic dXNlcjp1c2Vy
> User-Agent: curl/8.4.0
> Accept: */*
>
< HTTP/1.1 200 OK
```

```
$ echo dXNlcjp1c2Vy | base64 -d
user:user
```

# Demo: Un-authenticated

```
$ curl -v http://localhost:8080/management/liveness
*   Trying ::1:8080...
*   Connected to localhost (::1) port 8080
> GET /management/liveness HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/8.4.0
> Accept: */*
>
< HTTP/1.1 401 Unauthorized
```

# Demo ... 401

( no password, bad password, bad user, or unauthorized )

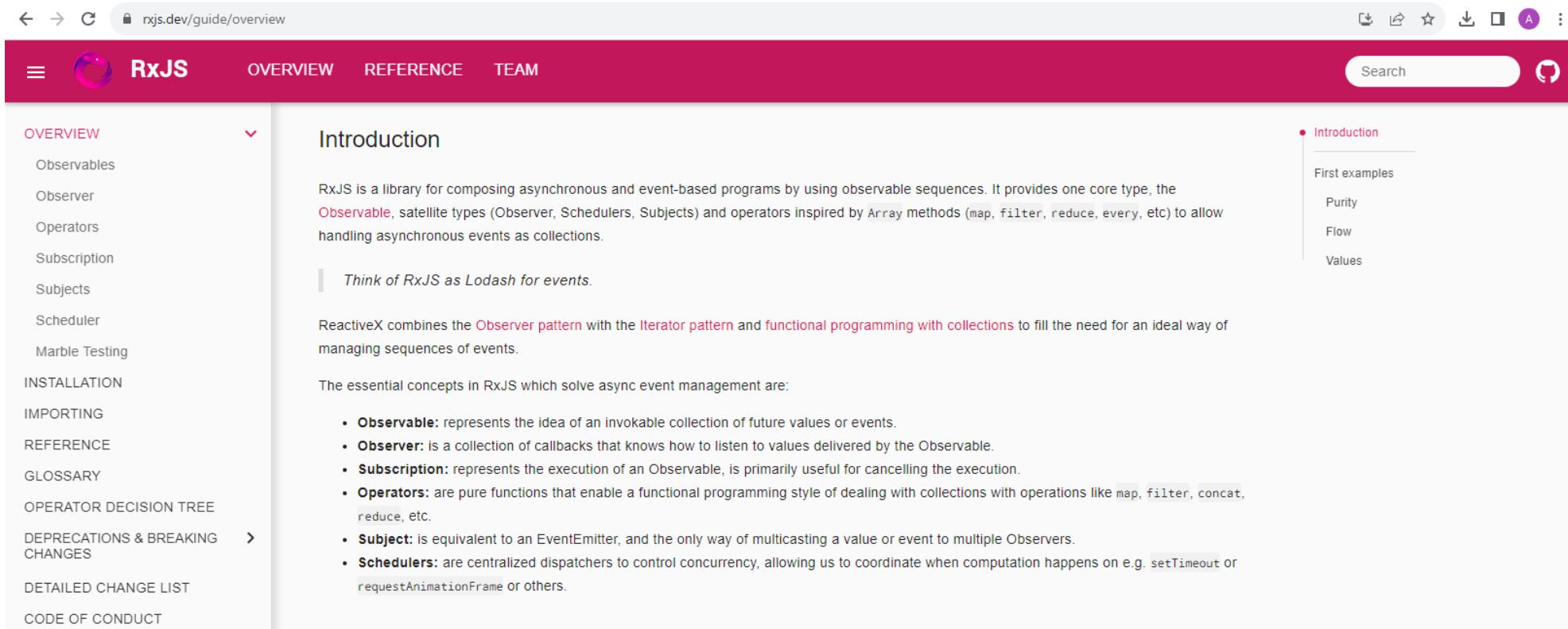
```
$ curl -v -u user:pass http://localhost:8080/management/liveness
*   Trying [:1]:8080...
* Connected to localhost ([:1]) port 8080
* Server auth using Basic with user 'user'
> GET /management/liveness HTTP/1.1
> Host: localhost:8080
> Authorization: Basic dXNlcjpwYXNz
> User-Agent: curl/8.4.0
> Accept: */*
>
< HTTP/1.1 401 Unauthorized
```

Demo of Login in Jhipster  
( if time ok ? )

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Code Scaffolding
  - Generating (Angular) Client from Swagger
  - Security: Cookie, Login, OAuth2, JWT, ...
  - • RxJs, Auto completion, ng-prime
- Final Words

# RxJs : Asynchronous programming with Observable & Subscribe, is ~~Complex~~ Powerful

A screenshot of the RxJS guide overview page. The page has a red header with the RxJS logo and navigation links for Overview, Reference, and Team. A search bar is on the right. The main content area has a sidebar with links like Observables, Observer, Operators, etc. The main content starts with an introduction about RxJS being a library for composing asynchronous and event-based programs using observable sequences. It mentions Observables, Observers, Subscriptions, and Operators. It compares RxJS to Lodash for events and ReactiveX. It lists essential concepts: Observable, Observer, Subscription, Operators, Subject, and Schedulers. A sidebar on the right shows a tree structure with Introduction at the top, followed by First examples, Purity, Flow, and Values.

rxjs.dev/guide/overview

**RxJS** OVERVIEW REFERENCE TEAM

Search

OVERVIEW

- Observables
- Observer
- Operators
- Subscription
- Subjects
- Scheduler
- Marble Testing

INSTALLATION

IMPORTING

REFERENCE

GLOSSARY

OPERATOR DECISION TREE

DEPRECATIONS & BREAKING CHANGES

DETAILED CHANGE LIST

CODE OF CONDUCT

## Introduction

RxJS is a library for composing asynchronous and event-based programs by using observable sequences. It provides one core type, the [Observable](#), satellite types (Observer, Schedulers, Subjects) and operators inspired by Array methods (`map`, `filter`, `reduce`, `every`, etc) to allow handling asynchronous events as collections.

*Think of RxJS as Lodash for events.*

ReactiveX combines the [Observer pattern](#) with the [Iterator pattern](#) and [functional programming with collections](#) to fill the need for an ideal way of managing sequences of events.

The essential concepts in RxJS which solve async event management are:

- **Observable**: represents the idea of an invokable collection of future values or events.
- **Observer**: is a collection of callbacks that knows how to listen to values delivered by the Observable.
- **Subscription**: represents the execution of an Observable, is primarily useful for cancelling the execution.
- **Operators**: are pure functions that enable a functional programming style of dealing with collections with operations like `map`, `filter`, `concat`, `reduce`, etc.
- **Subject**: is equivalent to an `EventEmitter`, and the only way of multicasting a value or event to multiple Observers.
- **Schedulers**: are centralized dispatchers to control concurrency, allowing us to coordinate when computation happens on e.g. `setTimeout` or `requestAnimationFrame` or others.

# Example RxJs Operator: map()

rxjs.dev/api/operators/map

RxJS OVERVIEW REFERENCE TEAM

OVERVIEW INSTALLATION IMPORTING REFERENCE GLOSSARY OPERATOR DECISION TREE DEPRECATIONS & BREAKING CHANGES DETAILED CHANGE LIST CODE OF CONDUCT stable (v7.8.1)

API / rxjs/operators map FUNCTION STABLE OPERATOR

Applies a given project function to each value emitted by the source Observable, and emits the resulting values as an Observable.

`map<T, R>(project: (value: T, index: number) => R, thisArg?: any): OperatorFunction<T, R>`

**Parameters**

project	<code>(value: T, index: number) =&gt; R</code>	The function to apply to each value emitted by the source Observable. The index parameter is the number <i>i</i> for the <i>i</i> -th emission that has happened since the subscription, starting from the number 0.
thisArg	any	Optional. Default is <code>undefined</code> . An optional argument to define what <code>this</code> is in the project function.

**Returns**

`OperatorFunction<T, R>: A function that returns an Observable that emits the values from the source Observable transformed by the given project function.`

**Description**

Like `Array.prototype.map()`, it passes each source value through a transformation function to get corresponding output values.

```
graph LR; 1((1)) -- "map(x => 10 * x)" --> 10((10)); 2((2)) -- "map(x => 10 * x)" --> 20((20)); 3((3)) -- "map(x => 10 * x)" --> 30((30))
```

Sample usage:

"unchecked JSON" DTO interface  
-map->  
type-safe "Model" class

# Some Object-Oriented Reasons to re-define class on client-side for DTO interface

```
export interface RowDTO {  
  cols?: Array<any>;  
}
```



```
export class RowModel {  
  private readonly cols: any[]; // immutable  
  1 usage  
  public constructor(src: RowDTO) {  
    this.cols = src?.cols || []; // sanity check input  
  }  
  no usages  
  col(idx: number): string { return this.cols[idx]; }  
  no usages  
  strCol(idx: number): string { return <string> this.cols[idx]; }  
}
```

```
const json: RowDTO[] = [  
  { cols: [ 1, 'test' ]},  
  { cols: [ 2, 'test' ]}  
];
```



```
const model: RowModel[] = [  
  new RowModel(json[0]),  
  new RowModel(json[1]),  
];  
const model2 :RowModel[] = json.map(x :RowDTO => new RowModel(x));
```

# Calling RxJs: .pipe(map(x => transform(x)))

```
query(req: QuerySimpleTableColumnsParamsDTO, tableName: string): Observable<RowModel[]> {
  return this.restService.query7(req, tableName).pipe(
    map( project: (dto: RowDTO[]) => dto!.map(element : RowDTO  => new RowModel(element)))
  );
}
```

# Other Example RxJs operator: debounceTime(millis)

rxjs.dev/api/index/function/debounceTime

RxJS OVERVIEW REFERENCE TEAM

OVERVIEW Observables Observer Operators Subscription Subjects Scheduler Marble Testing INSTALLATION IMPORTING REFERENCE GLOSSARY OPERATOR DECISION TREE DEPRECATIONS & BREAKING CHANGES DETAILED CHANGE LIST CODE OF CONDUCT

stable (v7.8.1)

debounceTime FUNCTION STABLE OPERATOR

Emits a notification from the source Observable only after a particular time span has passed without another source emission.

`debounceTime<T>(dueTime: number, scheduler: SchedulerLike = asyncScheduler): MonoTypeOperatorFunction<T>`

Parameters

dueTime	number	The timeout duration in milliseconds (or the time unit determined internally by the optional scheduler) for the window of time required to wait for emission silence before emitting the most recent source value.
scheduler	SchedulerLike	Optional. Default is <code>asyncScheduler</code> . The <code>SchedulerLike</code> to use for managing the timers that handle the timeout for each value.

Returns

`MonoTypeOperatorFunction<T>`: A function that returns an Observable that delays the emissions of the source Observable by the specified `dueTime`, and may drop some values if they occur too frequently.

Description

*It's like `delay`, but passes only the most recent notification from each burst of emissions.*

debounceTime(20)

debounceTime delays notifications emitted by the source Observable, but drops previous pending delayed emissions if a new notification arrives on the source Observable. This operator is similar to the `debounce` operator, but it only delays the first notification emitted on the source Observable if another notification arrives on the source Observable before the specified time interval has passed.

# example debounceTime() ?

Doing Auto-completion -> trigger http Call(s)  
but NOT for each char typed

```
yourAngularField$ : Observable<string> = ....
```

```
yourAngularField$  
  .pipe(debounce(300))  
  // ... more rxjs operators needed (cf next slide)  
  .subscribe(x => {  
    // handle field change, example for auto-completion  
  });
```

... cf Also .distinctUntilChanged()  
and switchMap() !!

<https://stackblitz.com/edit/autocomplete-suggestions-seach-with-debouncing?file=src%2Fapp%2Fapp.component.ts>

```
this.bookForm.controls["searchtext"].valueChanges
  .pipe(
    debounceTime(1000),
    distinctUntilChanged((curr, prev) => {
      return curr.toLowerCase() === prev.toLowerCase();
    }),
    switchMap(text => {
      this.searchstring.push(text);
      if (
        this.searchstring[this.searchstring.length - 1] !==
        this.searchstring[this.searchstring.length - 2]
      ) {
        return this.bookservic.getBookBySearchText(text);
      } else {
        return [];
      }
    })
  )
  .subscribe(res => {
    this.books = res;
 ));
}
```

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Code Scaffolding
  - Generating (Angular) Client from Swagger
  - Security: Cookie, Login, OAuth2, JWT, ...
  - • RxJs, Auto completion, ng-prime
- Final Words

# NG-Prime

Doing Rich web components is Difficult (and not funny)

Leave it to Experts !!

Re-use code, do not re-invent the Wheel

# https://primeng.org

The screenshot displays the PrimeNG website at <https://primeng.org>. The page features a header with the PrimeNG logo and navigation links for "Sakai Free Admin Template" and "View Demo". Below the header, there's a large banner with the text "The Most Complete UI Suite for Angular". A call-to-action button "Get Started →" and a "Give a Star" button with a yellow star icon are visible. The main content area is filled with various PrimeNG UI components, including:

- A form with fields for "Amount" (\$24.00) and "Beneficiary" (Select a User).
- A radio button group for "Account" with "Savings" selected.
- A date input field.
- A chart showing quarterly data from Q1 to Q4.
- A slider component with "Angular" and "TypeScript" tags and a toggle switch between "Styled" and "Unstyled" modes.
- A product card for "Sneaker Premium Quality" priced at \$990, featuring an image of the shoe and an "Add to Cart" button.
- A user profile card for "Amanda Williams" (Administrator) with links to "Dashboard" and "Control Panel".
- An inbox card with 3 messages, labeled "Inbox View Messages".
- A navigation bar with "Home" and "Calendar" links.

# <https://github.com/primefaces/primeng>

Screenshot of the GitHub repository page for `primefaces / primeng`.

The repository is public and has 3 branches and 244 tags.

Recent commits:

- cagataycivici Fixed link (e8620e4, 8 hours ago)
- .github Update github bug report description (2 months ago)
- .vscode Refactor (last year)
- api-generator Update apidoc & generate docs (5 days ago)
- ani Optimize images and refactor landing (9 months ago)

**About**

The Most Complete Angular UI Component Library

[primeng.org](http://primeng.org)

components charts angular typescript  
ui mit datatable datagrid

[Readme](#)

**README.md**

npm package 17.0.0-rc.1 downloads 1.4M/month NodeJS CI failing chat 701 online Prime Discussions 567

**PRIME NG**  
The Most Complete UI Library for Angular



Website

Visit the [PrimeNG Website](#) for general information, demos and documentation.

Production - primeng-ssr-test 7 hours ago  
Production - primeng 8 hours ago  
+ more deployments

Languages

Language	Percentage
JavaScript	60.2%
TypeScript	38.6%
Other	1.2%

# (alternatives to) Prime-NG

ng-bootstrap components (but very low level, not so rich behaviors)

tons of npm libraries ... but you have a patchwork of CSS styles

Google Material Design WebComponent library

your own company already has some super cool toolkit ?

# Outline

- Some Advanced features
  - AG-Grid
  - HighCharts
  - Code Scaffolding
  - Generating (Angular) Client from Swagger
  - Security: Cookie, Login, OAuth2, JWT, ...
  - RxJs, Auto completion, ng-prime
- • Final Words

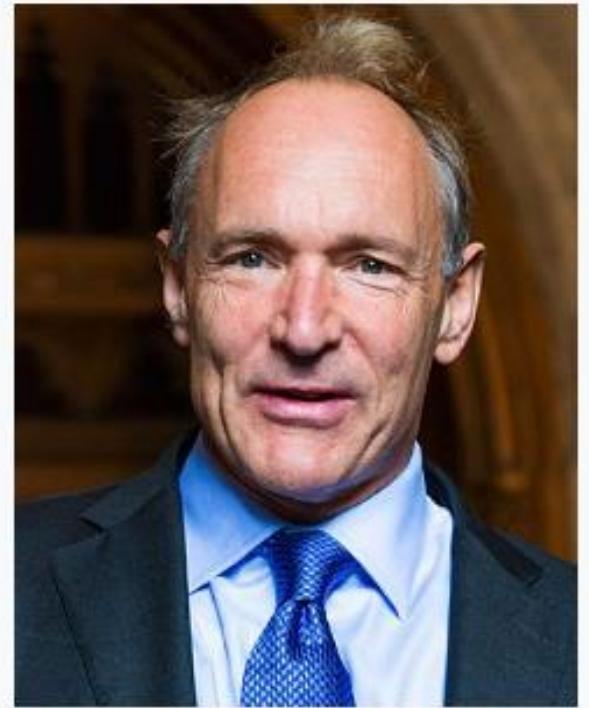
# Final Words

## Thank you Tim

Sir

**Tim Berners-Lee**

OM KBE FRS FREng FRSA DFBCS RDI



Berners-Lee in 2014

<b>Born</b>	Timothy John Berners-Lee 8 June 1955 (age 68) <a href="#">London</a> , England
<b>Other names</b>	TimBL TBL
<b>Education</b>	<a href="#">The Queen's College, Oxford</a> (BA)
<b>Known for</b>	Invention of the <a href="#">World Wide Web</a>

# Render to Cesar what belongs to Cesar

**Brendan Eich** (/[aɪk](#)/; born July 4, 1961)<sup>[1]</sup> is an American computer programmer and technology executive. He created the [JavaScript programming language](#) and co-founded the [Mozilla](#) project, the [Mozilla](#)



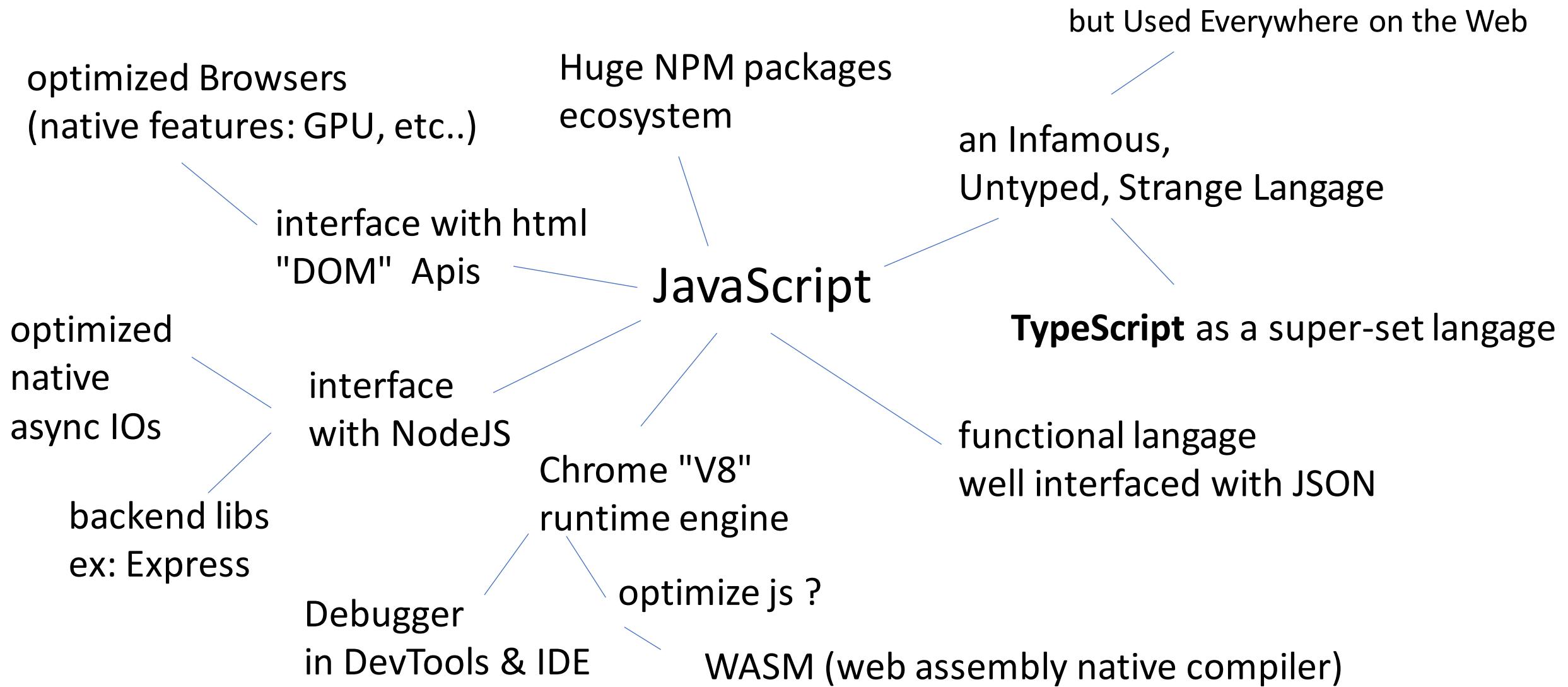
Eich in 2012

<b>Born</b>	4 July 1961 (age 62) Pittsburgh, Pennsylvania, US
<b>Alma mater</b>	<a href="#">University of Illinois Urbana-Champaign</a> <a href="#">Santa Clara University</a>
<b>Website</b>	<a href="http://brendaneich.com">brendaneich.com</a> ↗

# Web Fundamental



# JavaScript MindMap



# TypeScript - JavaScript Schizofrenia Troller

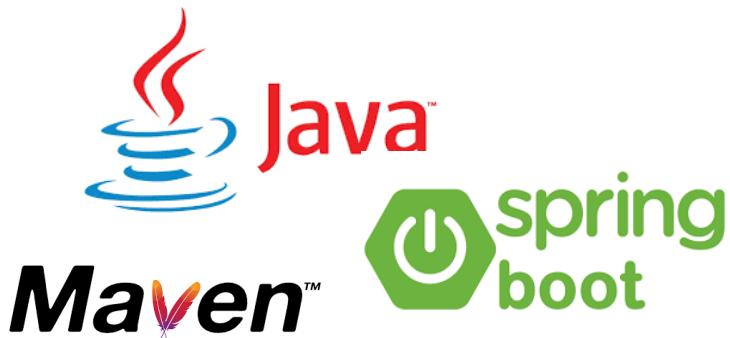
**TS** TypeScript



**JavaScript**  
 JS

# Backend - Frontend Schizofrenia Troller

Back-end Developper

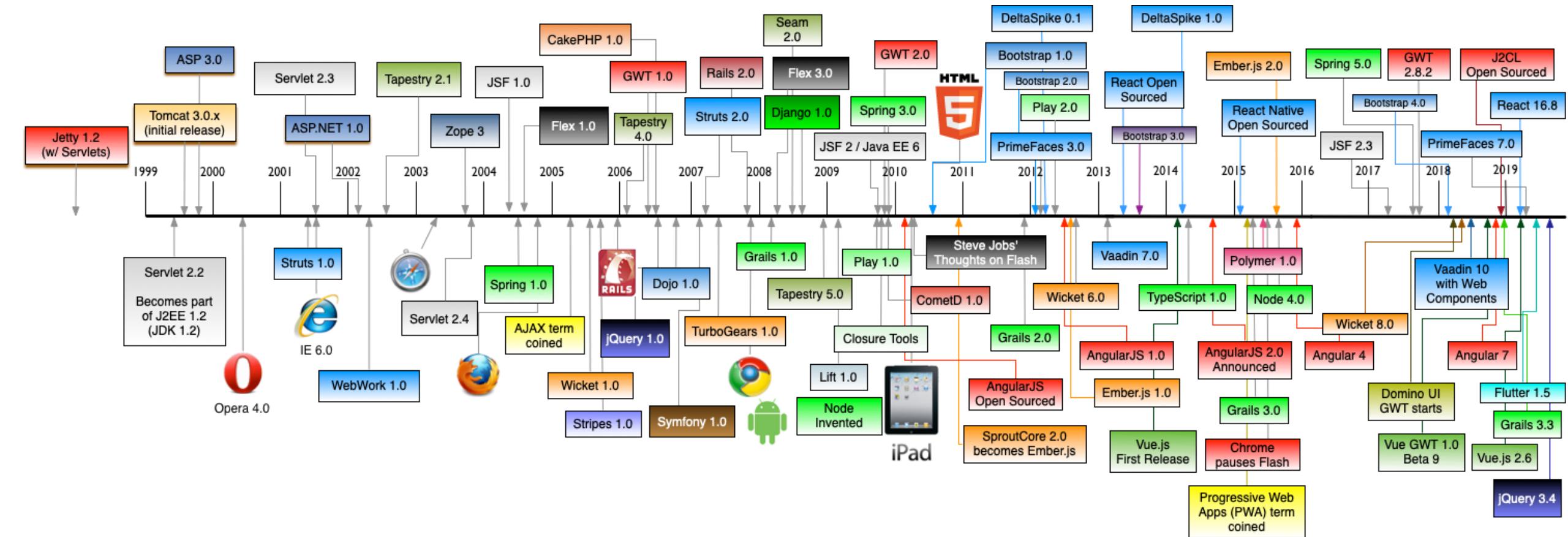


Front-end Developper



Full-Stack Developper

# Web Frameworks ... 1999 to 2020



# Web Frameworks - 2023 and Future ?



# Conclusion

Frameworks change every year

The Web will live long : Html + Javascript + Css

# Questions

arnaud.nauwynck@gmail.com