



# Formation ANGULAR Développement Avancé

Animé par Michel BOCCIOLESI

# Table des matières

- **Introduction :**
  - Framework Progressif ou Orienté
  - Installation - configuration
  - Le Framework Angular - Historique  
Les versions marquantes **15 - 20**
  - Standalone Components VS Modules
- **Le Lazy Loading – Deferrable Views**  
Concepts avancés de routage  
Optimisation de la compilation
- **Notion de Réactivité Angular**  
Méthode « Value Based » (Zone.js)  
Méthode Observable Based  
Le signal NG17
- **La programmation RXJS et les observables**  
Approfondissement
- **Le Signal**  
La réactivité (Angular >=17)
- **Les formulaires Angular**  
Reactive Forms vs Template Driven Form  
Compilation de composants
- **Tests Unitaires**  
Karma & Jasmine  
Documenter son projet – fichiers MD
- **I18N : L'Internationalisation**  
Comment traduire notre projet Angular ?
- **SSR : Server Side Rendering**
- **Annexes :**
  - Rappels EcmaScript 2015+
  - Avenir de Zone.js (Zoneless NG20) et Change Detection Mode
  - PWA – Progressive Web App
  - NGRX : Notion de Store dans Angular

# Application finale réalisée

- <https://dev.webjs.fr/1-Angular/website2>
- Projet source à télécharger – cahier d'exercices :  
<https://dev.webjs.fr/1-Angular/CAHIERS/>

Copyright Michel BOCCIOLESI - ORSYS

# Framework Orienté ou Progressif



Copyright Michel BOCCIOLESI - ORSYS

## Framework ou librairies ?

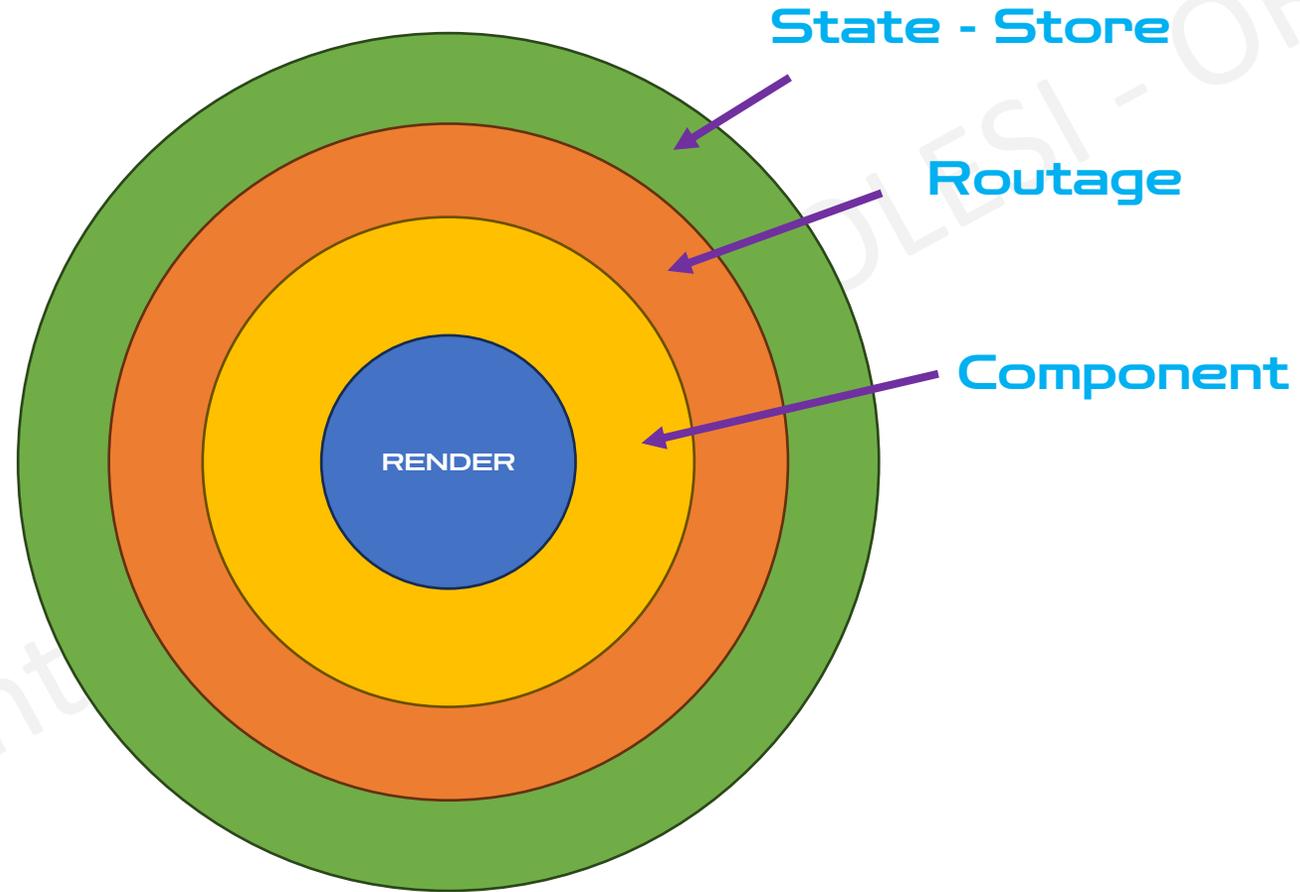
Dans le monde du développement Front, il existe plusieurs catégories de frameworks :

- [Angular](#) (2016) est un framework « **orienté** »
- [React](#) (2013) est une librairie mais peut être piloté et utilisé par d'autres framework (« [NextJS](#) » ou « [Vite](#) »)
- [Vue](#) (2014) est un framework « **progressif** <sup>1</sup> ». Il peut cependant être utilisé en tant que librairie. 🤔  
Il possède ses propres outils de création [npm create vue](#) ou [Vite](#) <sup>2</sup>

<sup>1</sup> Le projet peut être simple initialement et peut facilement évoluer vers un projet beaucoup plus complexe, grâce à sa modularité

<sup>2</sup> Vite a été développé en **2020 par Evan You** le créateur de [Vue, Vite et Vitest](#)

# Framework progressif



Copyright

OLESI-ORSYS

## Histoire du Web

- **2003** : Création du WHATWG
- **2007** : HTML5 et les API JS – le tout 1<sup>er</sup> smartphone est présenté
- **2010** : Responsive Web Design
- **2011** : Bibliothèque CSS Bootstrap
- **2015** : le développement Front devient très important, les frameworks, le webpack sont de plus en plus utilisés

## EcmaScript : la normalisation de Javascript

- **1999** : version 3
- **2009** : version 5 (la 4 n'avait existé)
- **2015**: version 6
- 2016
- 2017
- 2018
- 2019
- **ES Next**

# Installation et configuration



Copyright Michel CIOLESI - ORSYS

# Installation et configuration

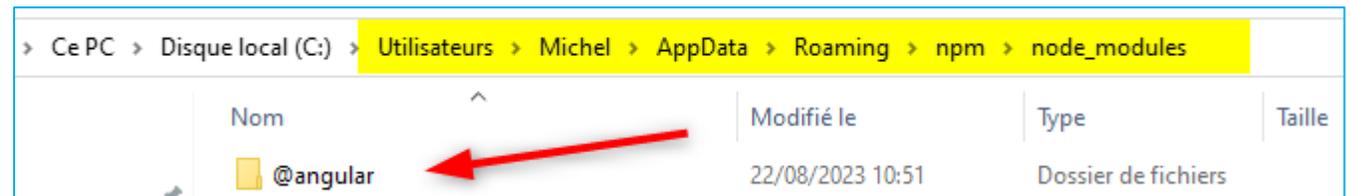
<https://nodejs.org>

<https://angular.dev/>

<https://www.npmjs.com/package/@angular/cli>

- `npm install -g @angular/cli` (global dans le path utilisateur)

`/home/michel (linux)`



# Node.JS

Tous ces frameworks utilisent [NODEJS](#) (2009), un serveur javascript développé par Google.

[NodeJS](#) permet de travailler son projet Web en tant que [WEBPACK](#).

C'est beaucoup plus facile de structurer, d'architecturer un projet avec un webpack.

NB : Même en Vanilla, on peut développer avec un [Webpack](#). La toute 1<sup>ère</sup> étape est de créer un projet avec la commande **npm init (-y)**

<https://nodejs.org/fr>

```
Windows PowerShell
PS C:\Users\Michel> ng version

Angular CLI
-----
Angular CLI: 18.2.5
Node: 20.12.1
Package Manager: npm 10.8.1
OS: win32 x64

Angular: undefined
...

Package                                Version
-----
@angular-devkit/architect              0.1802.5 (cli-only)
@angular-devkit/core                   18.2.5 (cli-only)
@angular-devkit/schematics             18.2.5 (cli-only)
@schematics/angular                    18.2.5 (cli-only)

PS C:\Users\Michel>
```

Copyright

Depuis la version 18, le mode standalone est proposé par défaut, si vous souhaitez utiliser le mode module 🙄  
ajouter l'option -- standalone=false

```
ng new projet-avec-module --standalone=false
```

```
PS C:\Users\Michel> ng new projet-standalone-ng-18  
Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS  
Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? no
```

```
PS C:\Users\Michel> npx @angular/cli@14 new projet-angular-14
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? CSS
```

- npx @angular/cli@14 new nomDuProjet

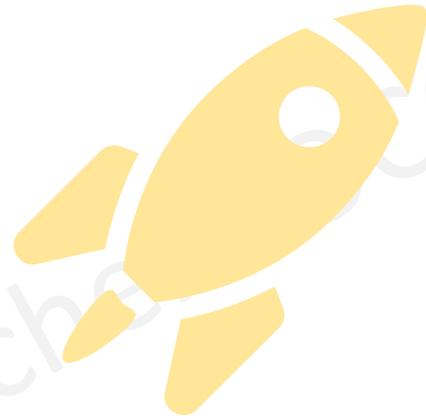
```
PS C:\Users\Michel\projet-angular-14> ls

Répertoire : C:\Users\Michel\projet-angular-14

Mode                LastWriteTime         Length Name
----                -
d-----            30/09/2024    10:37      .vscode
d-----            30/09/2024    10:38    node_modules
d-----            30/09/2024    10:37      src
-a-----            30/09/2024    600     .browserslistrc
-a-----            30/09/2024    274     .editorconfig
-a-----            30/09/2024    548     .gitignore
-a-----            30/09/2024   2977    angular.json
-a-----            30/09/2024   1434    karma.conf.js
-a-----            30/09/2024  479274  package-lock.json
-a-----            30/09/2024   1050    package.json
-a-----            30/09/2024   1070    README.md
-a-----            30/09/2024    287    tsconfig.app.json
-a-----            30/09/2024    863    tsconfig.json
-a-----            30/09/2024    333    tsconfig.spec.json

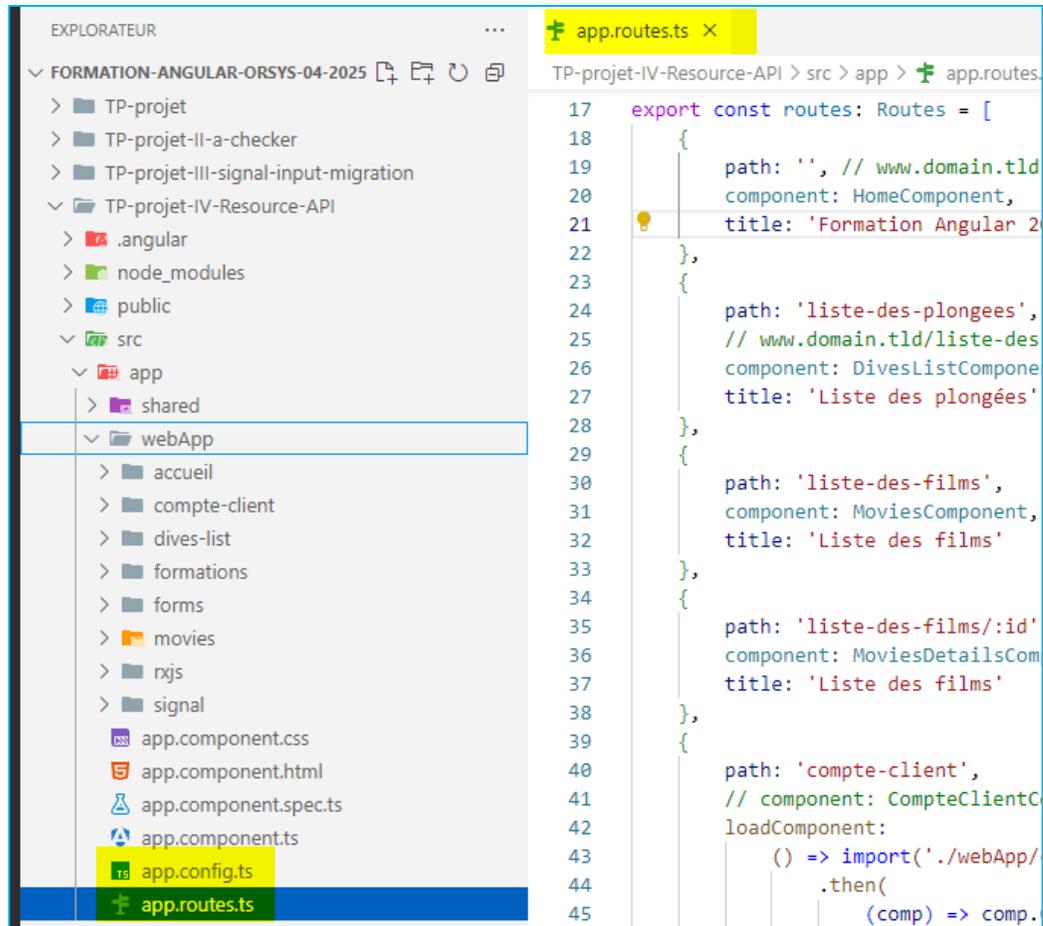
PS C:\Users\Michel\projet-angular-14>
```

# Les versions marquantes 15-20



Copyright Michaël ACCIOLESI - ORSYS

# Version 15 : Le mode Standalone



```
17 export const routes: Routes = [
18   {
19     path: '', // www.domain.tld
20     component: HomeComponent,
21     title: 'Formation Angular 2
22   },
23   {
24     path: 'liste-des-plongees',
25     // www.domain.tld/liste-des
26     component: DivesListCompon
27     title: 'Liste des plongées'
28   },
29   {
30     path: 'liste-des-films',
31     component: MoviesComponent,
32     title: 'Liste des films'
33   },
34   {
35     path: 'liste-des-films/:id'
36     component: MoviesDetailsCom
37     title: 'Liste des films'
38   },
39   {
40     path: 'compte-client',
41     // component: CompteClientC
42     loadChildren:
43       () => import('./webApp/
44         .then(
45           (comp) => comp.
```

## Version 15 (16 novembre 2022)

- Plus **besoin** des modules
- Le composant est **autonome**
- Il importe tout ce dont il a besoin
- Le **routage** est beaucoup plus **simple** ( un seul fichier à la racine du projet)
- L'application ne **bootstrap** plus un **module** mais un **composant** « **point d'entrée** »

# Les versions marquantes Angular

- Version 16 (12 mai 2023)

- Les **Signals**
- Change Detection Change
- Encore besoin de **Zone.JS** ?
- Avenir de **RXJS** ?



Copyright Michel BOCCIOLESI - ORSYS



# Angular 17 - La renaissance !

Version 17 (8 novembre 2023) <https://angular.dev/>

- Le control flow est totalement revu  
`*ngIf` et `*ngFor` sont remplacés par `@if` et `@for`
- Webpack est remplacé par ESBUILD (ng server et nb build iront 2 fois plus vite !)
- SSR par défaut (Server Side Rendering => SEO ...)
- Lazy Loading (Vues différées) `@defer` `@placeholder`
- Signal est abouti en tant qu'API de réactivité



# Angular 18

Version 18 (23 mai 2024) <https://angular.dev/>

- Utilisation des signaux entre composants
- hydratation dans Angular DevTools
- 18.1 : @let 🔄 @let movies = movies\$ | async | filter ..

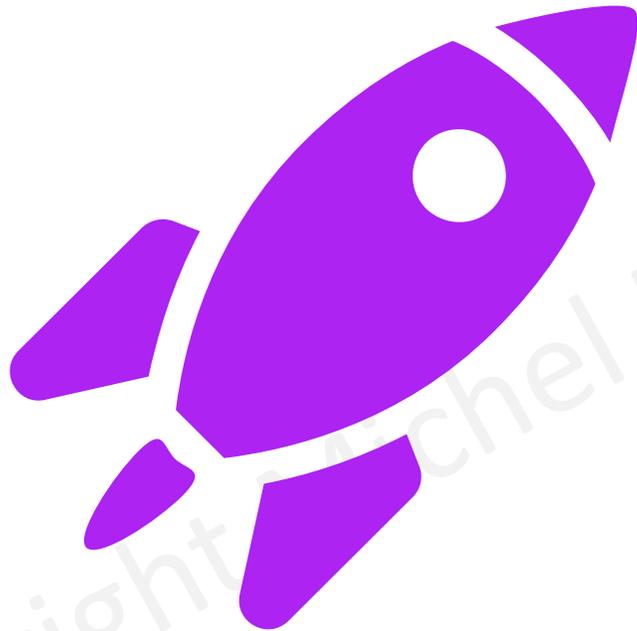


Copyright Michel BOCCIOLESI - ORSYS

# Angular 19

19 novembre 2024

<https://angular.dev/reference/releases>



- **LinkedSignal**  
<https://angular.dev/guide/signals/linked-signal#>
- **Ressource**  
<https://angular.dev/guide/signals/resource#resource-loaders>

New Refactoring -> Input & Output -> Signal Use

# Angular 19

- New Refactoring -> @ViewChild => viewChild Signal

```
ng g @angular/core:signal-queries-migration
```

- New Refactoring -> Input & Output -> Signal Use

```
ng g @angular/core:signal-input-migration
```

```
ng g @angular/core:output-migration
```



# Angular 19 – API Resource



Copyright Michel POCQUOLESI - ORSYS

## API Resource Pourquoi ?

class httpClient Angular 2+

- `private _http = inject(httpClient)`
- `this._http.get ...`

?

Angular 19

- `resource()`
- `rxResource()`
- `httpResource()`

Copyright Mich

### Zone.JS

- Détecte les changes UI (Change Detection Mode)
- Voit les datas changées depuis l'Observable
- Met à jour l'UI

### UI

#### Liste des Films

- Episode I
- Episode II
- ...

### Model – State – Datas

```
Movies$ = Observable<[Movie]>  
Movies$.subscribe()
```

### httpClient

- reçoit les datas
- émet les datas à l'Observable créé

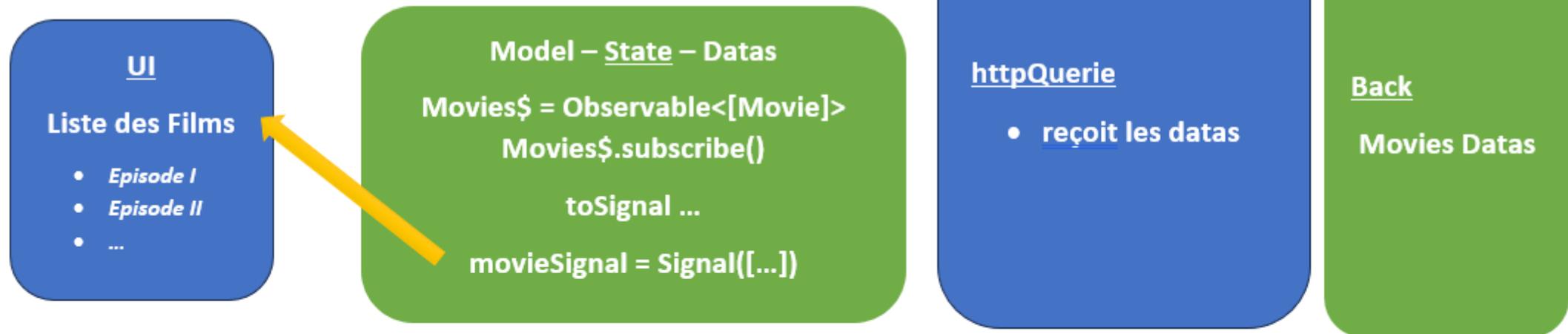
### Back

Movies Datas

## SIGNAL & RESOURCE API

### Zone.JS

- Détecte les changements (Change Detection Mode)
- Voit les données grâce à l'Observable
- Met à jour l'UI



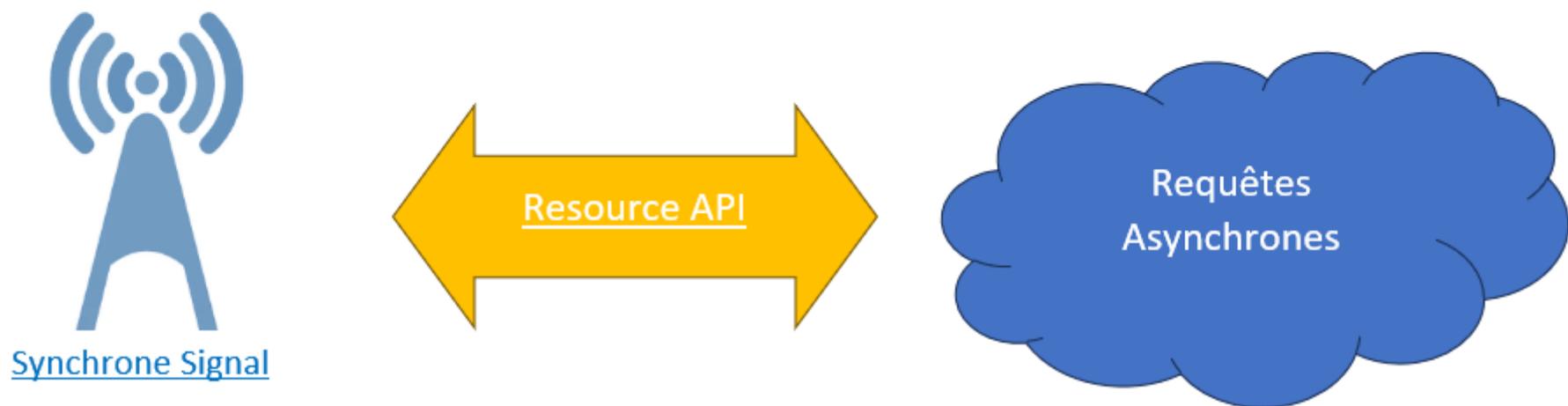
Le [signal \(SYNCHRONE\)](#) n'utilise plus Zone.JS et peut mettre à jour directement l'UI. Mais Angular 19+ a introduit une solution encore plus pratique ([Resource API](#)), qui est une sorte de pont entre la requête asynchrone http et le signal qui est synchrone ! Plus besoin de Subscription, les datas peuvent être reçues directement par le resource signal.

## SIGNAL & RESOURCE API



Le **signal (SYNCHRONE)** n'utilise plus Zone.JS et peut mettre à jour directement l'UI. Mais Angular 19+ a introduit une solution encore plus pratique (**Resource API**), qui est une sorte de pont entre la requête asynchrone http et le signal qui est synchrone ! Plus besoin de Subscription, les datas peuvent être reçues directement par le resource signal.

## Resource API



Le but de Resource API est de récupérer directement les datas dans un signal, sans utiliser Zone.JS. Resource API offre 3 méthodes d'utilisation au choix :

- resource() : utilise le fetch de javascript et les promesses
- rxResource() : utilise la programmation RXJS, les operators et les Observables (map, mergeMap, forkJoin)
- httpResource() : Angular 19.2 : plus facile et plus souple, utilise une syntaxe de Query http,

## TP : Mise en Pratique

[Home Page](#)[Compte Client](#)[Liste des plongées](#)[Liste des films Star Wars](#)[Les Observables](#)[Le Signal](#)[Resource](#)[Contacts](#)[← Go Back](#)

### Liste des Formations

Développement Front

Formation	Durée	Catégorie
Angular - les fondamentaux	4	Développement Front
Angular - concepts avancés	3	Développement Front
Javascript	4	Développement Front
React	3	Développement Front
Vue.JS	5	Développement Front
HTML & JS	5	Développement Front

src &gt; app &gt; shared &gt; services &gt; formations.service.ts &gt; FormationsService

```
1 import { computed, inject, Injectable, resource, signal } from '@angular/core';
2 import { HttpClient, HttpResponse } from '@angular/common/http';
3 import { FORMATIONS_API_URL } from '@shared/env/globals';
4 import { Formation } from '@shared/interfaces/Formation';
5 import { setErrorMessage } from '@shared/errors/error-http-msg';
6
7
8 @Injectable({ providedIn: 'root' })
9
10 export class FormationsService {
11
12   private _url = FORMATIONS_API_URL;
13   public formationModels = signal<string[]>([
14     'Programmation', 'Développement Front', 'Développement Back', 'Frontend', 'Backend'
15   ]);
16   public formationSelected = signal<string>('');
17   // -----
18
19   // 1- resource() avec paramètres
20
21   private formationsResource = resource({
22     request: this.formationSelected,
23     loader: (param) => fetch(`${this._url}?category=${param.request}`)
24       .then(
25         (res: any) => {
26           console.log(res);
27           return res.json() as Promise<Formation[]>
28         }
29       )
30   });
31
32   });
33
34   public formations = this.formationsResource.value;
35
36   public error = computed(
37     () => this.formationsResource.error() as HttpResponse
38   );
39   public errorMessage = computed(
40     () => setErrorMessage(this.error(), 'Formations'));
41
42   public isLoading = this.formationsResource.isLoading;
43
44 }
45
```

src &gt; app &gt; webApp &gt; formations &gt; formations.component.ts &gt; FormationsComponent

```
1 import { AfterViewChecked, Component, inject } from '@angular/core';
2 import { FormsModule } from '@angular/forms';
3 import { FormationsService } from '@app/shared/services/formations.service';
4
5 @Component({
6   selector: 'app-formations',
7   imports: [FormsModule],
8   templateUrl: './formations.component.html',
9   styleUrls: ['./formations.component.css']
10 })
11 export class FormationsComponent {
12
13   private _FormationsService = inject(FormationsService);
14
15   // Signals envoyés au Template HTML
16   public formationModels = this._FormationsService.formationModels;
17   public formationsSelected = this._FormationsService.formationSelected;
18
19   public formations = this._FormationsService.formations;
20   public errorMessage = this._FormationsService.error;
21   public isLoading = this._FormationsService.isLoading;
22
23
24 }
```

## Le template HTML

```
formations.component.html x
src > app > webApp > formations > formations.component.html > div.container.alert.alert-warning > div
Go to component
1 <div class="container alert alert-warning">
2   <h3 class="text-primary">Liste des Formations</h3>
3
4   @if (isLoading()) {
5     <div>... loading formations</div>
6   }
7   @else if (errorMessage){
8     <div style='color: red'>An error occurred: {{ errorMessage() }}</div>
9   }
10  @else {
11    <div>
12      <select [(ngModel)]="formationsSelected" class="form-control">
13
14        <option value="" selected>Choisir une formation</option>
15
16        @for(formation of formationModels(); track formation) {
17          <option>{{ formation}}</option>
18        }
19      </select>
20    </div>
21  </div>
22
23  <hr>
24
```

```
formations.component.html x
src > app > webApp > formations > formations.component.html > div.container.alert.ale
1 <div class="container alert alert-warning">
22
23 <hr>
24
25 @if (formations()) {
26 <table class="table ">
27 <thead>
28 <tr>
29 <th scope="col">Formation</th>
30 <th scope="col">Durée</th>
31 <th scope="col">Catégorie</th>
32 </tr>
33 </thead>
34
35 <tbody>
36 @for (f of formations(); track f) {
37 <tr>
38 <td {{ f.cours }} </td>
39 <td {{ f.duration }} </td>
40 <td {{ f.category }} </td>
41 </tr>
42 }
43 </tbody>
44 </table>
45 }
46 @else {
47 <div>Pas de formations</div>
48 }
49
50 }
51 </div>
```

ORSYS

Copyright

## Avec RxResource (Observables et opérateurs RXJS)

```
formations.service.ts X
src > app > shared > services > formations.service.ts > FormationsService

9
10 @Injectable({ providedIn: 'root' })
11
12 export class FormationsService {
13
14   private _url = FORMATIONS_API_URL;
15   public formationModels = signal<string[]>([
16     'Programmation', 'Développement Front', 'Développement Back', 'Front CSS
17   ]);
18   public formationSelected = signal<string>('');
19   // -----
20
21   // 1- resource() avec paramètres :
22   // private formationsResource = resource({
23   //   request: this.formationSelected,
24   //   loader: (param) => fetch(`${this._url}?category=${param.request}`)
25   //     .then(
26   //       (res: any) => {
27   //         console.log(res);
28   //         return res.json() as Promise<Formation[]>;
29   //       }
30   //     )
31   // });
32
33   // 2- rxResource() avec paramètres
34   private _http = inject(HttpClient)
35   private formationsResource = rxResource({
36     request: this.formationSelected,
37     loader: (param) => this._http.get<Formation>(`${this._url}?category=${pa
38     .pipe(
39       tap(
40         (valObs) => console.log(valObs)
41       )
42     )
43   });
44
45   // -----
46   public formations = this.formationsResource.value
47
48   public error = computed(
49     () => this.formationsResource.error() as HttpResponse
50   )
51   public errorMessage = computed(
```

The screenshot shows a web application with a dark header containing a logo and a menu icon. The main content area has a yellow background and is titled "Liste des Formations". It features a search input field with the text "Programmation". Below the search field is a table with three columns: "Formation", "Durée", and "Catégorie". The table contains three rows of data:

Formation	Durée	Catégorie
C#	4	Programmation
C	5	Programmation
C++	5	Programmation

At the bottom of the page, there is a dark footer with the text "THE FOOTER - Formation Angular".

The browser's developer console is open, showing the following logs:

```
Angular is running in development mode. core.mjs:20750
formations.service.ts:40
▶ (15) [{"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}, {"..."}]
formations.service.ts:40
▼ (3) [{"..."}, {"..."}, {"..."}]
  ▶ 0: {id: '13', cours: 'C#', codeCours: 'C001', duration: 4, category: 'Programmation'}
  ▶ 1: {id: '14', cours: 'C', codeCours: 'C002', duration: 5, category: 'Programmation'}
  ▶ 2: {id: '15', cours: 'C++', codeCours: 'C003', duration: 5, category: 'Programmation'}
```

## Avec httpResource (Angular 19.2)

```
formations.service.ts × package-lock.json
src > app > shared > services > formations.service.ts > FormationsService > formationsResource
12 export class FormationsService {
13     // 1- rxResource() avec paramètres
14     // private _http = inject(HttpClient)
15     // private formationsResource = rxResource({
16     //     request: this.formationSelected,
17     //     loader: (param) => this._http.get<Formation>(`${this._url}?category=${param}`)
18     //     .pipe(
19     //         tap(
20     //             (valObs) => console.log(valObs)
21     //         )
22     //     )
23     // });
24
25     // 3- httpResource() avec paramètres (Juste avec l'URL 😊)
26     // private formationsResource = httpResource<Formation>(
27     //     () => `${this._url}?category=${this.formationSelected()}`);
28
29     // 4- httpResource() avec paramètres dans un OBJET
30     private formationsResource = httpResource<Formation>({
31         url: this._url,
32         method: 'get',
33         headers: { accept: 'application/json' },
34         params: { category: this.formationSelected(), }
35     });
36
37     // -----
38     public formations = this.formationsResource.value
39
40     public error = computed(
41         () => this.formationsResource.error() as HttpErrorResponse
42     )
43
44     public errorMessage = computed(
45         () => setErrorMessage(this.error(), 'Formations'));
46
47     public isLoading = this.formationsResource.isLoading;
48 }
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
```

# Angular 20

28 Mai 2025

<https://angular.dev/reference/releases>



```
PS C:\Users\Michel> ng new NG20
✓ Do you want to create a 'zoneless' application without zone.js (Developer Preview)? Yes
✓ Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS
✓ Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? Yes
CREATE NG20/angular.json (2470 bytes)
CREATE NG20/package.json (1150 bytes)
CREATE NG20/README.md (1526 bytes)
CREATE NG20/tsconfig.json (1026 bytes)
```



## Zoneless

```
app.config.ts M X
TP19 > src > app > app.config.ts > appConfig > providers
1 import { ApplicationConfig, provideZoneChangeDetection } from '@angular/
2 import { provideRouter } from '@angular/router';
3
4 import { routes } from './app.routes';
5
6 export const appConfig: ApplicationConfig = {
7   providers: [
8     provideZoneChangeDetection({ eventCoalescing: true }),
9     provideRouter(routes)
10  ]
11 };
12
```

```
app.config.ts M X
TP20 > src > app > app.config.ts > ...
1 import { ApplicationConfig, provideBrowserGlobalErrorListeners
2 import { provideRouter } from '@angular/router';
3
4 import { routes } from './app.routes';
5
6 export const appConfig: ApplicationConfig = {
7   providers: [
8     provideBrowserGlobalErrorListeners(),
9     // provideZoneChangeDetection({ eventCoalescing: true }),
10    provideZonelessChangeDetection(), // NG20
11    provideRouter(routes)
12  ]
13 };
14
```

```
app.component.ts M X
TP19 > src > app > app.component.ts > ...
1 import { NgFor } from '@angular/common';
2 import { afterRender, Component, signal } from '@angular/core';
3
4
5 @Component({
6   selector: 'app-root',
7   imports: [NgFor],
8   templateUrl: './app.component.html',
9   styleUrls: ['./app.component.css']
10 })
11 export class AppComponent {
12   // 1-props
13   public title: string = 'Les Images';
14   public imagesArray: string[] = ['avatar1.jpg', 'avatar2.jpg', 'avatar3.jpg'];
15   // Zoneless
16   public titleSignal = signal<string>('Vive le Signal');
17
18
19   constructor() {
20     afterRender(
21       () => {
22         // news NG20
23         console.log('AfterRender =>', this.title);
24       }
25     );
26   }
27 }
```

```
app.ts M X
TP20 > src > app > app.ts > App > constructor > afterEveryRender() callback
1 import { NgFor } from '@angular/common';
2 import { afterEveryRender, Component, signal } from '@angular/core';
3
4
5 @Component({
6   selector: 'app-root',
7   imports: [NgFor],
8   templateUrl: './app.html',
9   styleUrls: ['./app.css']
10 })
11 export class App {
12   // 1-props
13   public title: string = 'Les Images';
14   public imagesArray: string[] = ['avatar1.jpg', 'avatar2.jpg', 'avatar3.jpg'];
15   // Zoneless
16   public titleSignal = signal<string>('Vive le Signal');
17
18
19   constructor(){
20     afterEveryRender(
21       () => {
22         // news NG20
23         console.log('AfterEveryRender =>', this.title);
24       }
25     );
26   }
27 }
```

Les structural directives \*ngIf \*ngFor \*ngSwitch sont « deprecated »  
« Removed » in V22 😞



```
package.json M  app.ts M x  app.html M x
src > app > app.ts > App
1  import { NgFor } from '@angular/common';
2  import { Component } from '@angular/core';
3
4
5  @Component({
6  selector: 'app-root',
7  imports: [NgFor],
8  templateUrl:
9  styleUrls:
10 })
11 export class
12 // 1-props
13 public tit
14 public ima
    'NgFor' est déprécié. ts(6385)
    common_module.d-Cpp8wYHt.d.ts(613, 4): La déclaration a été marquée
    ici comme étant dépréciée.
    (alias) class NgFor<T, U extends NgIterable<T> = i0.NgIterable<T>>
    import NgFor
    A structural directive that renders a template for each item in a collection. The directive
```

```
src > app > app.html > span > img
Go to component
1  <span *ngFor="let image of imagesArray">
2  
3  </span>
```

Copyright

# Migration entre versions majeures



Copyright Michel ACCIOLESI - ORSYS

- Pour mettre à jour une version majeure d'Angular 🤗, suivons les conseils d'Angular 😊
- <https://angular.dev/update-guide>
- `ng update @angular/core @angular/cli` (upgrade d'une version)
- `ng update @ngrx/store`

## Update Guide

Select the options that match your update

Angular versions

From v.  To v.

 **Warning:** Be sure to follow the guide below to migrate your application to the new version. You can't run `ng update` to update Angular applications more than one major version at a time.

Pour mettre à jour  
une version majeure  
d'Angular 😊, suivons  
les conseils d'Angular  
😊

## Guide to update your Angular application v14.0 → v17.0 for basic applications

### Before you update

*You don't need to do anything before moving between these versions.*

### Update to the new version

*Review these changes and perform the actions to update your application.*

- Make sure that you are using a supported version of node.js before you upgrade your application. Angular v15 supports node.js versions: 14.20.x, 16.13.x and 18.10.x. [Read further](#)
- Make sure that you are using a supported version of TypeScript before you upgrade your application. Angular v15 supports TypeScript version 4.8 or later. [Read further](#)
- In the application's project directory, run `ng update @angular/core@15 @angular/cli@15` to update your application to Angular v15.
- In your application's `tsconfig.json` file, remove `enableIvy`. In v15, Ivy is the only rendering engine so `enableIvy` is not required.
- Make sure that all `ActivatedRouteSnapshot` objects have a `title` property. In v15, the `title` property is a required property of `ActivatedRouteSnapshot`. [Read further](#)
- In v15, `relativeLinkResolution` is not configurable in the Router. It was used to opt out of an earlier bug fix that is now standard. [Read further](#)
- Update instances of `TestBed.inject()` that use an `InjectFlags` parameter to use an `InjectOptions` parameter. The `InjectFlags` parameter of `TestBed.inject()` is deprecated in v15. [Read further](#)
- Using `providedIn: 'any'` for an `@Injectable` or `InjectionToken` is deprecated in v15. [Read further](#)

```
PS C:\Users\Formateur\Desktop\TP-Standalone-ANY-10-2024-Solutions> ng update @angular/core@19 @angular/cli@19
The installed Angular CLI version is outdated.
Installing a temporary Angular CLI versioned 19.0.1 to perform the update.
Using package manager: npm
Collecting installed dependencies...
Found 40 dependencies.
Fetching dependency metadata from registry...
  Updating package.json with dependency @angular-devkit/build-angular @ "19.0.1" (was "18.2.7")...
  Updating package.json with dependency @angular/cli @ "19.0.1" (was "18.2.7")...
  Updating package.json with dependency @angular/compiler-cli @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/localize @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency ng-packagr @ "19.0.1" (was "18.2.1")...
  Updating package.json with dependency typescript @ "5.6.3" (was "5.4.5")...
  Updating package.json with dependency @angular/animations @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/common @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/compiler @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/core @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/forms @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/platform-browser @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/platform-browser-dynamic @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/router @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency @angular/service-worker @ "19.0.0" (was "18.2.6")...
  Updating package.json with dependency zone.js @ "0.15.0" (was "0.14.10")...
UPDATE package.json (1768 bytes)
✓ Cleaning node modules directory
✓ Installing packages
** Executing migrations of package '@angular/cli' **

> Update '@angular/ssr' import paths to use the new '/node' entry point when 'CommonEngine' is detected.
  Migration completed (No changes made).

> Update the workspace configuration by replacing deprecated options in 'angular.json' for compatibility with the latest Angular CLI changes.
  Migration completed (No changes made).

** Optional migrations of package '@angular/cli' **

This package has 1 optional migration that can be executed.
Optional migrations may be skipped and executed after the update process, if preferred.

Select the migrations that you'd like to run (Press <space> to select, <a> to toggle all, <i> to invert selection, and <enter> to proceed)
> [use-application-builder] Migrate application projects to the new build system. (https://angular.dev/tools/cli/build-system-migration)
```

UPDATE src/app/app.component.ts (420 bytes)  
UPDATE src/app/webApp/tests/pipes/see-more.pipe.ts (682 bytes)  
Migration completed (46 files modified).

> Updates ExperimentalPendingTasks to PendingTasks.  
Migration completed (No changes made).

**\*\* Optional migrations of package '@angular/core' \*\***

This package has 1 optional migration that can be executed.  
Optional migrations may be skipped and executed after the update process, if preferred.

Select the migrations that you'd like to run (Press <space> to select, <a> to toggle all, <i> to invert selection, and <enter> to proceed)

> [provide-initializer] Replaces `APP\_INITIALIZER`, `ENVIRONMENT\_INITIALIZER` & `PLATFORM\_INITIALIZER` respectively with `provideAppInitializer`, `provideEnvironmentInitializer` & `providePlatformInitializer`.

# Webpack

Le webpack est un environnement de développement NODEJS, composé de plusieurs fichiers de configuration permettant de « customiser » le développement (serve) et le build (compilation)

- package.json
- angular.json
- tsconfig.json
- répertoire node\_modules \*

- node\_modules \* contient toutes les librairies (dépendances) nécessaires au projet webpack.

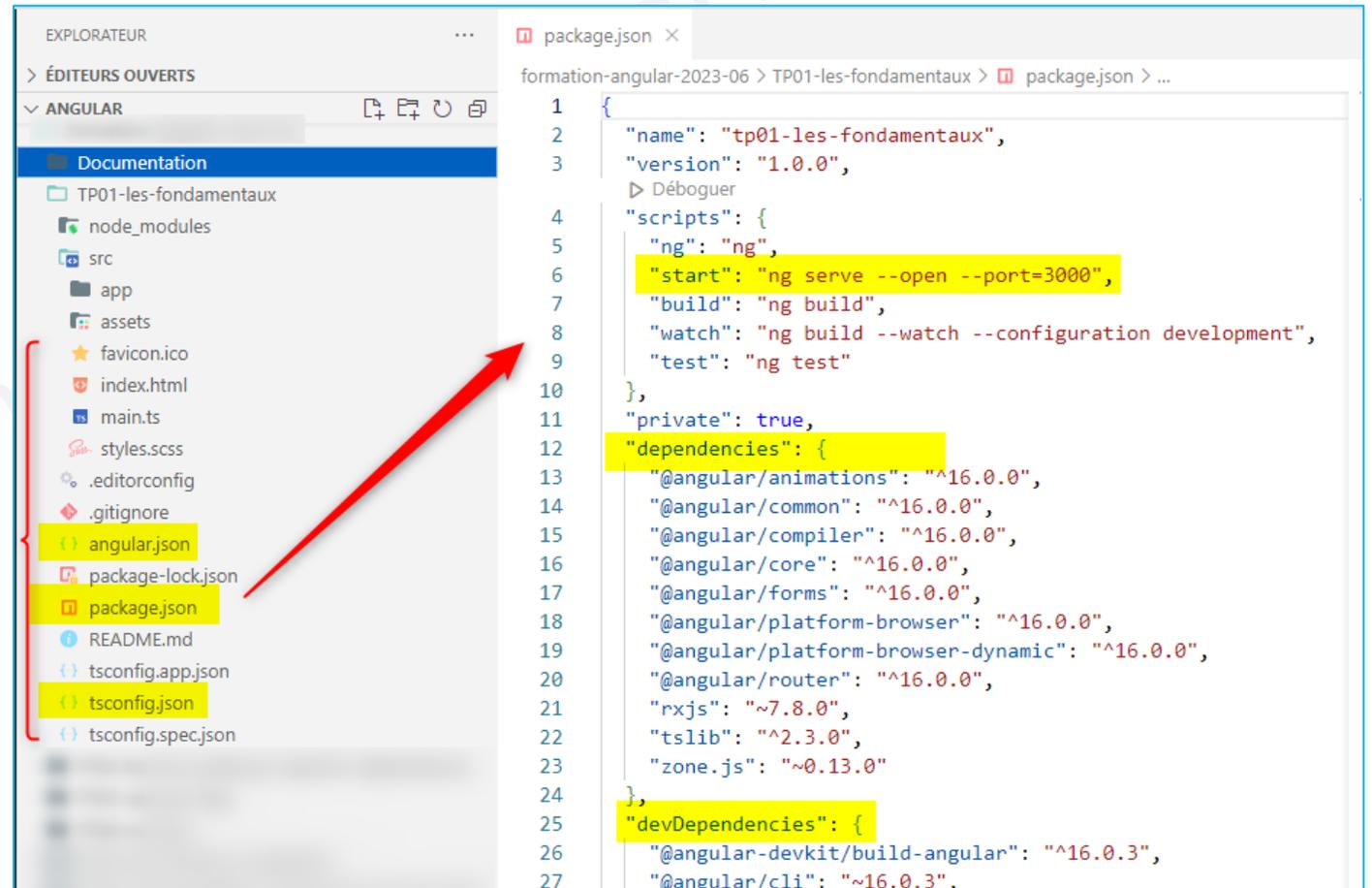
Il y a au moins 32k fichiers dans un simple projet Angular.

[npm init -y](#) : crée un projet webpack

[npm update](#) : met à jour les paquets selon les règles du package.json

^ : mise à jour minor + patch

~ : mise à jour patch



```
1 {
2   "name": "tp01-les-fondamentaux",
3   "version": "1.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve --open --port=3000",
7     "build": "ng build",
8     "watch": "ng build --watch --configuration development",
9     "test": "ng test"
10  },
11  "private": true,
12  "dependencies": {
13    "@angular/animations": "^16.0.0",
14    "@angular/common": "^16.0.0",
15    "@angular/compiler": "^16.0.0",
16    "@angular/core": "^16.0.0",
17    "@angular/forms": "^16.0.0",
18    "@angular/platform-browser": "^16.0.0",
19    "@angular/platform-browser-dynamic": "^16.0.0",
20    "@angular/router": "^16.0.0",
21    "rxjs": "~7.8.0",
22    "tslib": "^2.3.0",
23    "zone.js": "~0.13.0"
24  },
25  "devDependencies": {
26    "@angular-devkit/build-angular": "^16.0.3",
27    "@angular/cli": "~16.0.3",
```

# Créer un Webpack

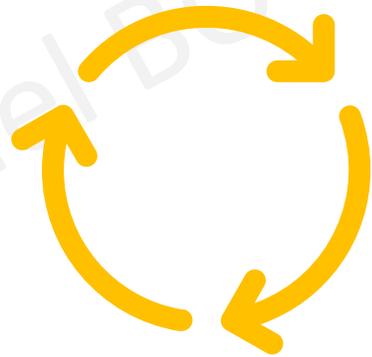
```
PS C:\Users\Formateur> npm init -y
Wrote to C:\Users\Formateur\package.json:

{
  "name": "formateur",
  "version": "1.0.0",
  "description": "ok",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "json-server": "^1.0.0-beta.3"
  },
  "devDependencies": {},
  "keywords": []
}
```

```
PS C:\Users\Formateur> npm create vite
Need to install the following packages:
create-vite@5.5.5
Ok to proceed? (y) y
✓ Project name: ... vite-project
? Select a framework: » - Use arrow-keys. Return to submit.
> Vanilla
  Vue
  React
  Lit
  Svelte
  Solid
  Qwik
  Angular
  Others
```



Routage Avancé  
Lazy Loading  
Optimisation du build



Copyright Michel BOCCIOLESI - ORSYS

# Routing : Concepts avancés – Le Lazy Loading

Browser tabs: Liste des plongées, Liste des plongées SEO

Address bar: localhost:52070/compte-client/voir-mon-compte

Navigation: Home, Liste des plongées, Espace client (lazy loading)

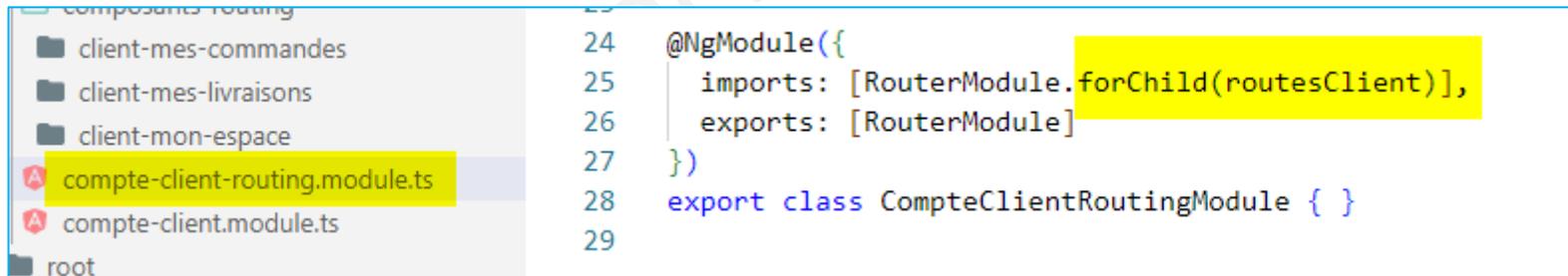
Page Content:

- HEADER : Espace Client
- Mon Espace Client
- Voir mon compte ...
- Voir mes livraisons ...
- Voir mes commandes ...
- client-mon-espace works!
- FOOTER : Espace Client

A red arrow points from the 'Voir mon compte ...' link to the 'client-mon-espace works!' message.

## Routing : Concepts avancés – Le Lazy Loading

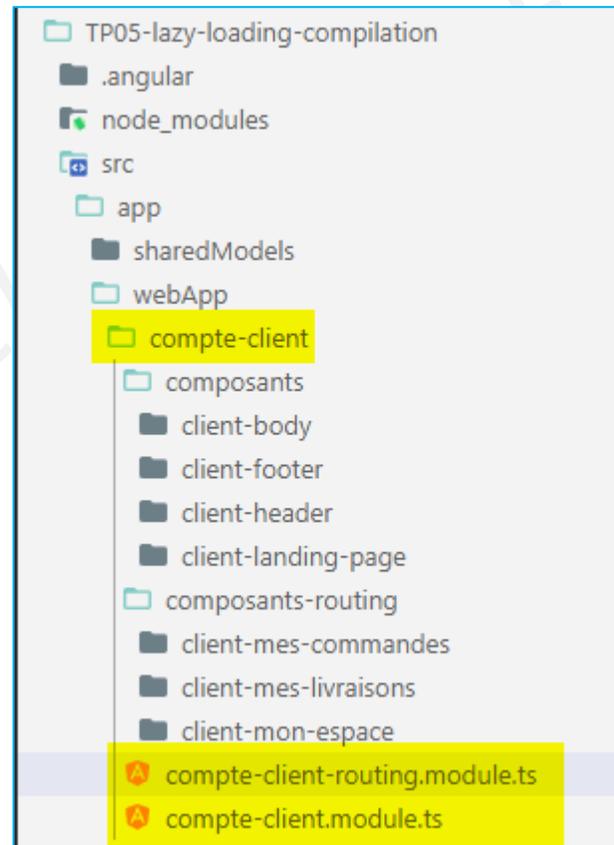
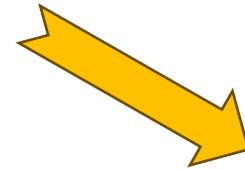
Créer le module compte-client avec l'option « --routing »  
Cette option crée un module de routage spécifique  
forChild



```
24 @NgModule({
25   imports: [RouterModule.forChild(routesClient)],
26   exports: [RouterModule]
27 })
28 export class CompteClientRoutingModule { }
29
```

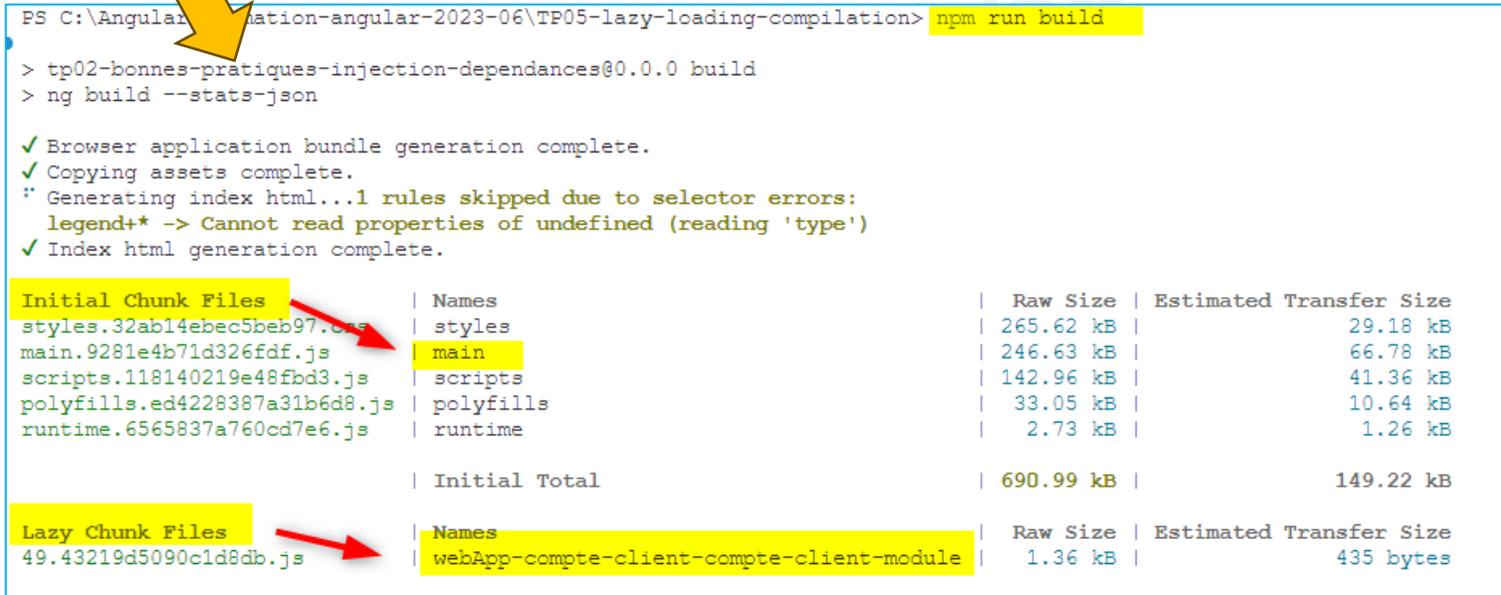
# Routing : Concepts avancés – Le Lazy Loading

TD : déployer une architecture de module et composants « compte-client »  
comme ci-dessous



# Routing : Concepts avancés – Le Lazy Loading

Le Lazy Loading permet de charger tout un module (compilé dans un fichier extrait du main.js), seulement quand l'utilisateur sollicitera ce module par une action de navigation.  
Les fichiers de build (compilation) sont donc dégroupés et le projet final « buildé » est optimisé.



```
PS C:\Angular\Injection-angular-2023-06\TP05-lazy-loading-compilation> npm run build
> tp02-bonnes-pratiques-injection-dependances@0.0.0 build
> ng build --stats-json

✓ Browser application bundle generation complete.
✓ Copying assets complete.
* Generating index html...1 rules skipped due to selector errors:
  legend+* -> Cannot read properties of undefined (reading 'type')
✓ Index html generation complete.

Initial Chunk Files | Names | Raw Size | Estimated Transfer Size
styles.32ab14ebec5beb97.css | styles | 265.62 kB | 29.18 kB
main.9281e4b71d326fdf.js | main | 246.63 kB | 66.78 kB
scripts.118140219e48fbd3.js | scripts | 142.96 kB | 41.36 kB
polyfills.ed4228387a31b6d8.js | polyfills | 33.05 kB | 10.64 kB
runtime.6565837a760cd7e6.js | runtime | 2.73 kB | 1.26 kB

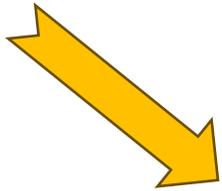
| Initial Total | 690.99 kB | 149.22 kB

Lazy Chunk Files | Names | Raw Size | Estimated Transfer Size
49.43219d5090c1d8db.js | webApp-compte-client-compte-client-module | 1.36 kB | 435 bytes
```

# Routing : Concepts avancés – Le Lazy Loading

Afin de bien analyser ces customisations « tuning » de compilation, mise en place d'une stratégie de lecture de statistiques avec le « webpack-bundle-analyzer »

<https://www.npmjs.com/package/webpack-bundle-analyzer>



```
package.json ×
P05-lazy-loading-compilation > package.json > ...
5   "ng": "ng",
6   "start": "ng serve --open --port=3000",
7   "build": "ng build --stats-json",
8   "watch": "ng build --watch --configuration development --stats-json",
9   "test": "ng test",
10  "json-server": "json-server --watch src/assets/json/dives.json -p 3001",
11  "analyze": "webpack-bundle-analyzer dist/stats.json"
12  },
```

# Le Lazy Loading : Mise en pratique

## Etape #1 (no Lazy Loading)

```
client-landing-page.component.html
1 <div class="alert alert-primary">
2   <app-client-header></app-client-he
3   <hr>
4   <app-client-body></app-client-body
5   <hr>
6   <app-client-footer></app-client-f
7 </div>

client-body.component.html
1 <ul class="list-group">
2   <li class="list-group-item">
3     <a class="nav-link"
4       [routerLink]="['mon-espace']"
5       [routerLinkActive]="['lien-actif']"
6     >Mon espace Client</a>
7   </li>
8   <li class="list-group-item">
9     <a class="nav-link"
10      routerLink="mes-commandes"
11      [routerLinkActive]="['lien-actif']"
12     >Mes commandes</a>
13   </li>
14   <li class="list-group-item">
15     <a class="nav-link"
16       [routerLink]="['mes-livraisons']"
17       [routerLinkActive]="['lien-actif']"
18     >Mes livraisons</a>
19   </li>
20 </ul>
21
22 <p class="alert alert-success">
23   <router-outlet></router-outlet>
24 </p>
25

accueil.module.ts
11 // import { CompteClientModule } from '../..formation/compte-client,
12
13 @NgModule({
14   declarations: [
15     LandingPageComponent,
16     HeaderComponent,
17     BodyComponent,
18     FooterComponent
19   ],
20   imports: [
21     CommonModule,
22     RouterModule,
23     // nos modules
24     FilmsModule,
25     FormsReactiveModule,
26     // CompteClientModule,
27     RxjsModule
28   ],
29   exports: [
30     LandingPageComponent,
31     HeaderComponent,
32     BodyComponent
33   ]
34 })

compte-client-routing.module.ts
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ClientMonEspaceComponent } from './composants-routing/client-mon-espace/client-mon-espace.c
4 import { ClientCommandesComponent } from './composants-routing/client-commandes/client-commandes.com
5 import { ClientLivraisonsComponent } from './composants-routing/client-livraisons/client-livraisons.
6
7 export const routesClient: Routes = [
8   // const routesClient: Routes = [
9
10   { path: 'mon-espace', component: ClientMonEspaceComponent },
11   { path: 'mes-commandes', component: ClientCommandesComponent },
12   { path: 'mes-livraisons', component: ClientLivraisonsComponent },
13 ];

app-routing.module.ts
9 import { ClientLandingPageComponent } from './webApp/formation/compte-client
10
11 // -----
12 import { routesClient } from './webApp/formation/compte-client/compte-client-routing.module
13
14 const routes: Routes = [
15   { path: '', component: BodyComponent },
16   { path: 'accueil', component: BodyComponent },
17   { path: 'liste-des-films', component: ListeDesFilmsComponent },
18
19   // 1ère étape : sans lazy loading
20   // raccrocher un module qui a son propre système de routage
21   {
22     path: 'compte-client',
23     component: ClientLandingPageComponent,
24     children: routesClient
25   },
26 ]
```

# Le Lazy Loading : Mise en pratique

## Etape #2 (Lazy Loading)

```
client-landing-page.component.html ×
1 <div class="alert alert-primary">
2   <app-client-header></app-client-he
3   <hr>
4   <app-client-body></app-client-body
5   <hr>
6   <app-client-footer></app-client-fc
7 </div>

client-body.component.html ×
1 <ul class="list-group">
2   <li class="list-group-item">
3     <a class="nav-link"
4       [routerLink]="['mon-espace']"
5       [routerLinkActive]="['lien-actif']"
6     >Mon espace Client</a>
7   </li>
8   <li class="list-group-item">
9     <a class="nav-link"
10      routerLink="mes-commandes"
11      [routerLinkActive]="['lien-actif']"
12    >Mes commandes</a>
13  </li>
14  <li class="list-group-item">
15    <a class="nav-link"
16      [routerLink]="['mes-livraisons']"
17      [routerLinkActive]="['lien-actif']"
18    >Mes livraisons</a>
19  </li>
20 </ul>
21
22 <p class="alert alert-success">
23   <router-outlet></router-outlet>
24 </p>
25

accueil.module.ts ×
11 // import { CompteClientModule } from '../formation/compte-client/compte-client.
12
13 @NgModule({
14   declarations: [
15     LandingPageComponent,
16     HeaderComponent,
17     BodyComponent,
18     FooterComponent
19   ],
20   imports: [
21     CommonModule,
22     RouterModule,
23     // nos modules
24     FilmsModule,
25     FormsReactiveModule,
26     // CompteClientModule,
27     RxjsModule
28   ]
29 })

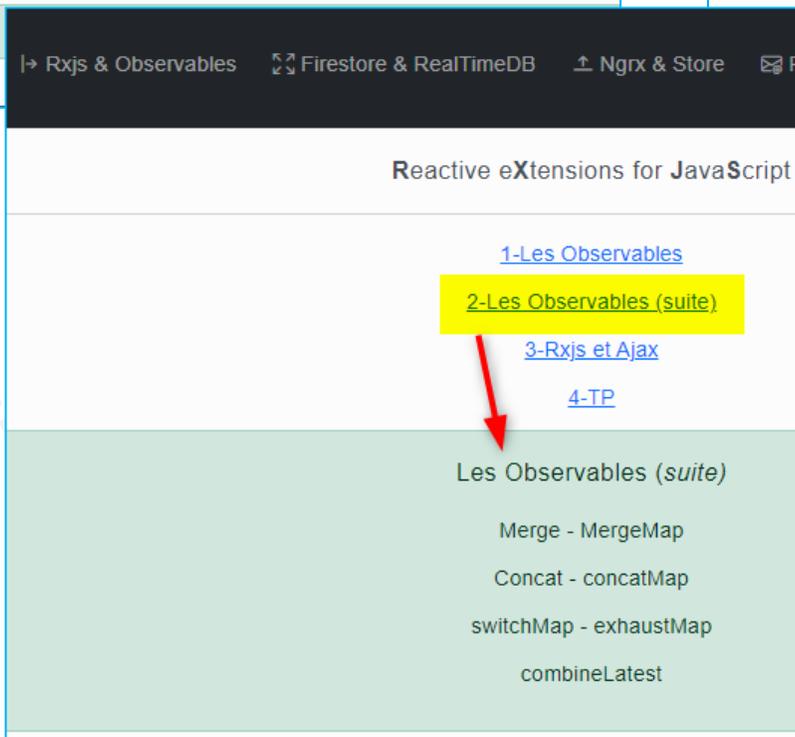
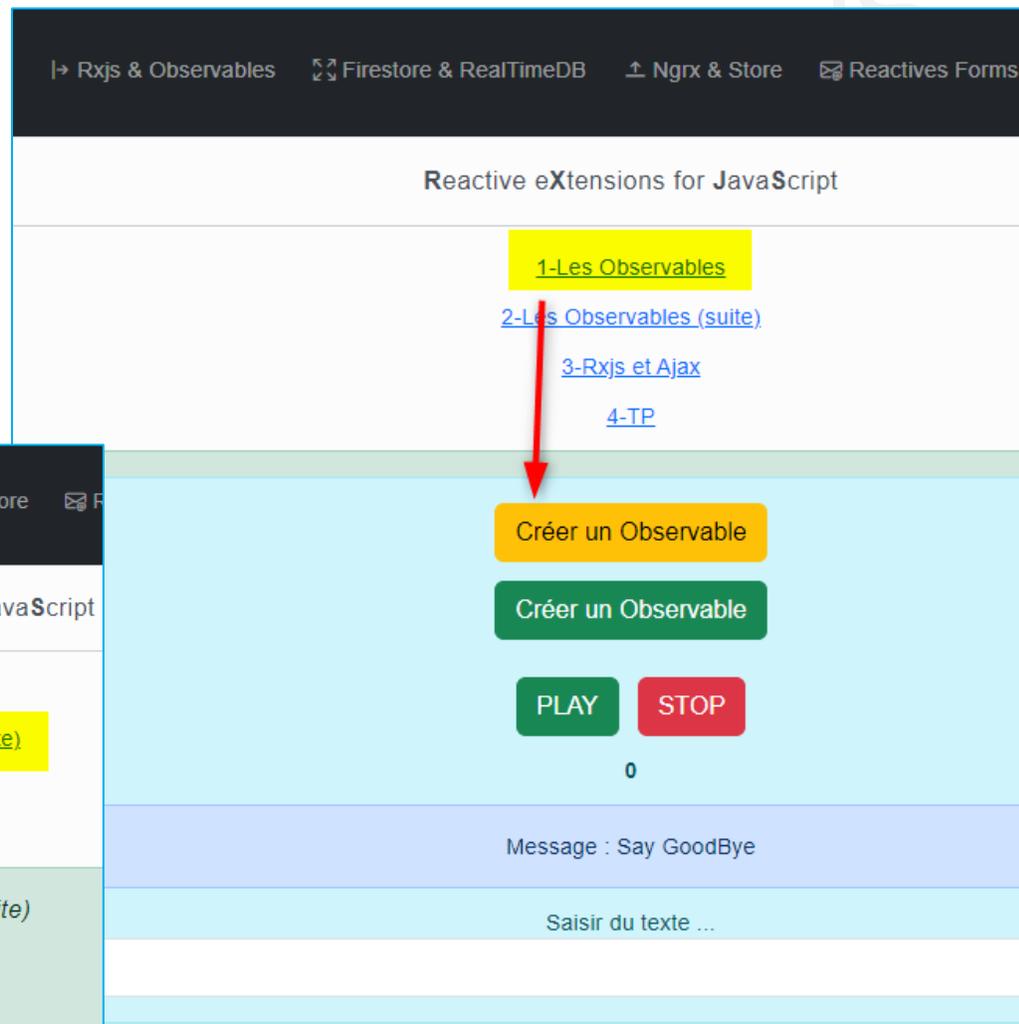
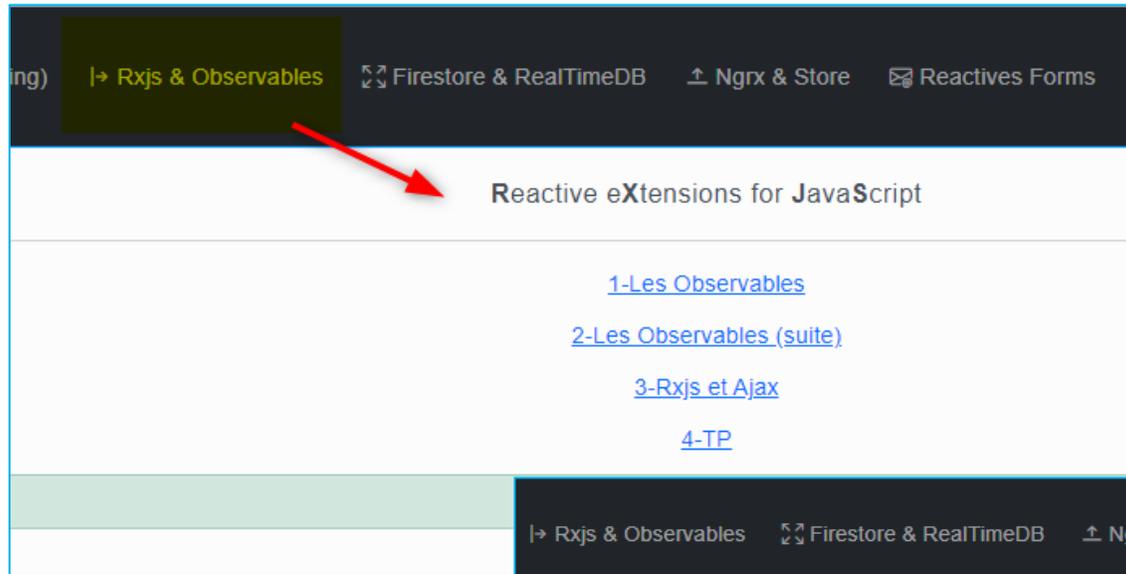
app-routing.module.ts ×
26
27 // 2ème étape : Avec lazy loading (promesses ES2015 😊)
28 // {
29 //   path: 'compte-client',
30 //   component: ClientLandingPageComponent,
31 //   loadChildren:
32 //     () => import('../webApp/formation/compte-client/compte-client.module')
33 //   .then(
34 //     (m) => {
35 //       return m.CompteClientModule
36 //     }
37 //   )
38 // },
39
40 // 3ème étape : Avec Lazy Loading (ES2017 : Async/await => asynchrone 😊)
41 {
42   path: 'compte-client',
43   component: ClientLandingPageComponent,
44   loadChildren:
45     async () => (await import('../webApp/formation/compte-client/compte-client.module'
46 // syntactic sugar
47   data :{
48     preload:true
49   }
50 }
51 },
```

```
compte-client-routing.module.ts ×
Formation-ANY-07-2023 > TP03-lazy-loading > src > app > webApp > formation > compte-client > compte-client-routing.module.ts > routesClient
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ClientMonEspaceComponent } from '../composants-routing/client-mon-espace/client-mon-espace.c
4 import { ClientCommandesComponent } from '../composants-routing/client-commandes/client-commandes.com
5 import { ClientLivraisonsComponent } from '../composants-routing/client-livraisons/client-livraisons.
6
7 export const routesClient: Routes = [
8   const routesClient: Routes = [
9
10   { path: 'mon-espace', component: ClientMonEspaceComponent },
11   { path: 'mes-commandes', component: ClientCommandesComponent },
12   { path: 'mes-livraisons', component: ClientLivraisonsComponent },
13 ];
14
15 @NgModule({
16   imports: [RouterModule.forChild(routesClient)],
17   exports: [RouterModule]
18 })
```

TP : Déployer une arborescence en Lazy Loading

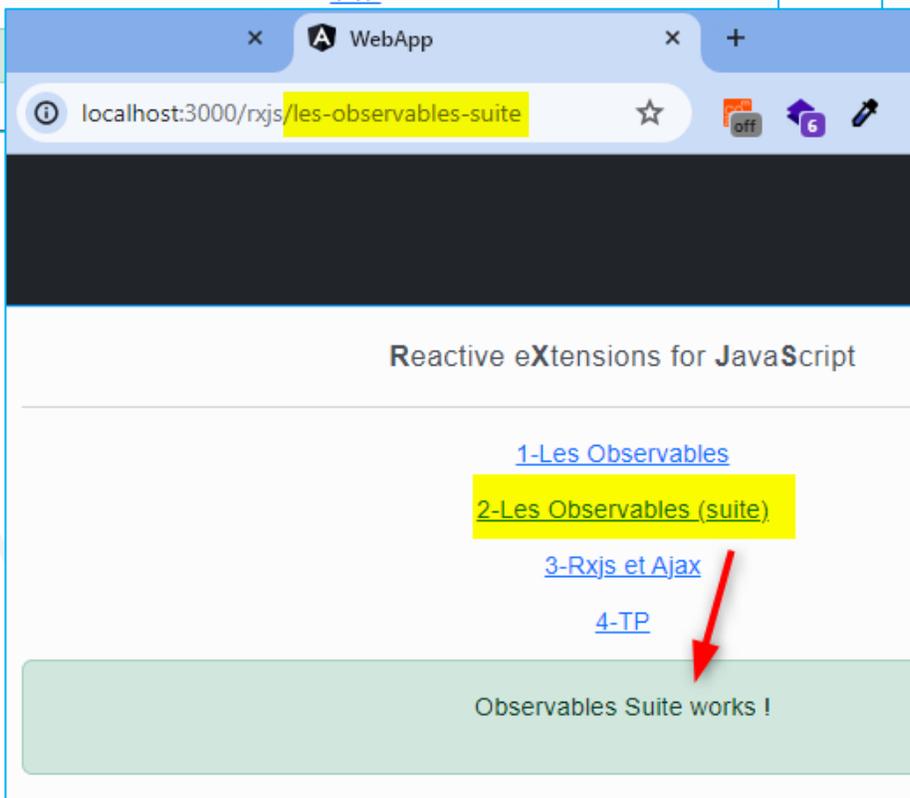
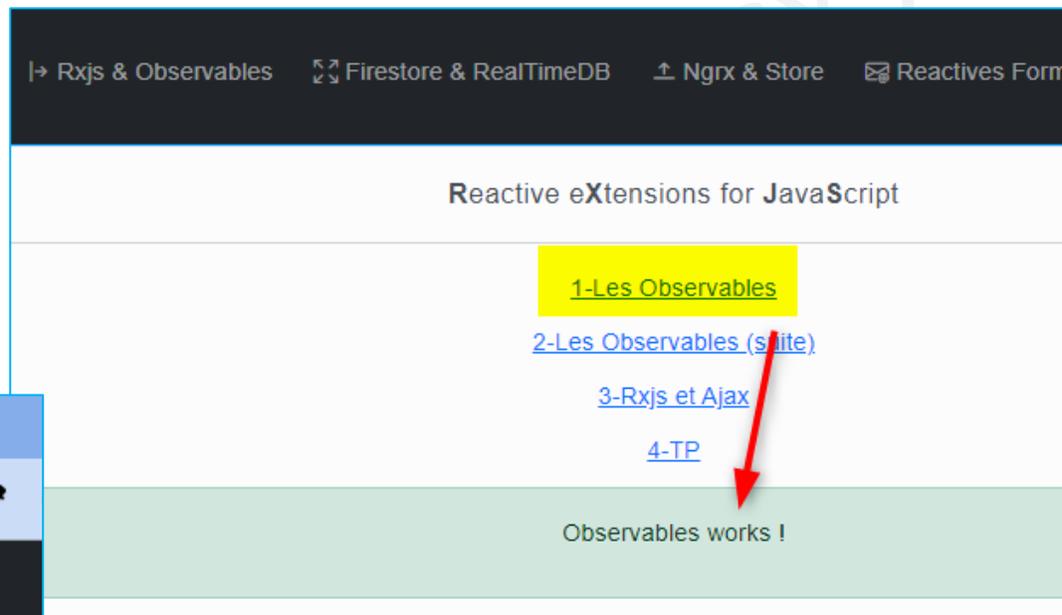
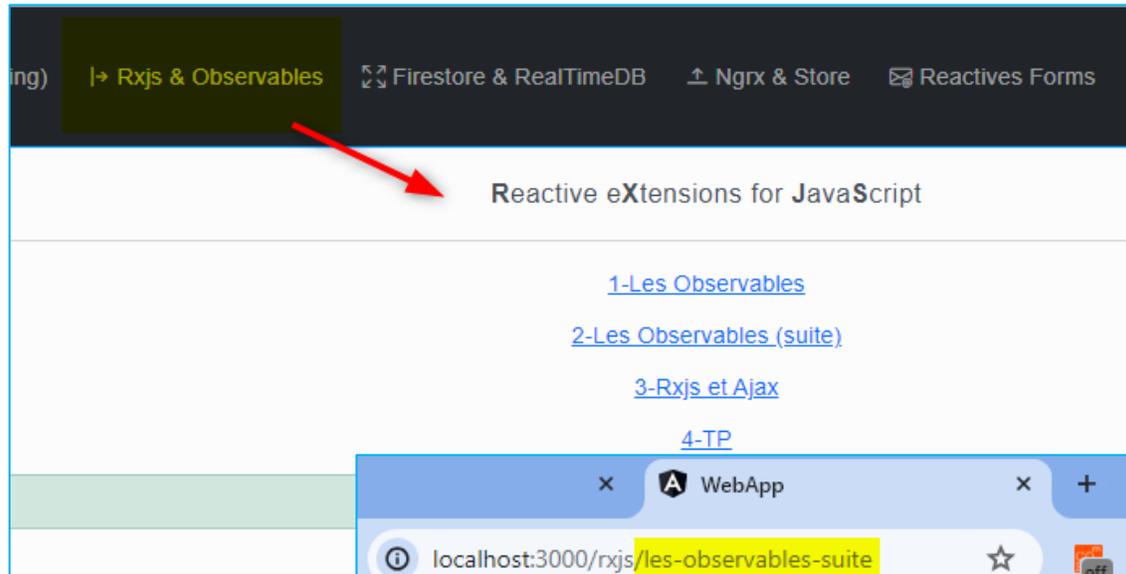
Copyright Michel BOCCICCESI - ORSYS

## TP : déployer l'arborescence en Lazy Loading RXJS



Copyr.

## TP : déployer l'arborescence en Lazy Loading RXJS



# TP : déployer l'arborescence en Lazy Loading RXJS

The image displays a development environment with the following components:

- EXPLORATEUR (File Explorer):** Shows a project structure with folders like 'webApp', 'formation', and 'rxjs'. The 'rxjs' folder is highlighted in yellow, and its sub-folder 'composants' is highlighted in blue.
- rxjs.component.html:** Contains a list of links for navigation:

```
1 <div class="container alert alert-light">
2   <h4><strong>R</strong>eactive e<strong>X</strong>tensions for <strong>J</strong>ava<strong>S</strong>cript</h4>
3   <hr>
4   <p><a href="#">1-Les Observables</a></p>
5   <p><a href="#">2-Les Observables (suite)</a></p>
6   <p><a href="#">3-Rxjs et Ajax</a></p>
7   <p><a href="#">4-TP</a></p>
8 </div>
```
- rxjs.module.ts:** Shows the main module configuration with imports for various Angular services and components:

```
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { RouterModule } from '@angular/router';
4 import { HttpClientModule } from '@angular/common/http';
5 import { RxjsRoutingModule } from './rxjs-routing.module';
6
7 import { ObservablesComponent } from './composants/observables';
8 import { RxjsComponent } from './composants/rxjs';
9 import { TpComponent } from './composants-tp';
10 import { AjaxComponent } from './composants-ajax';
11 import { ObservablesSuiteComponent } from './composants-observables-suite';
12
13 @NgModule({
14   declarations: [
15     ObservablesComponent, RxjsComponent, ObservablesSuiteComponent, AjaxComponent, TpComponent
16   ],
17   imports: [
18     CommonModule, RxjsRoutingModule, HttpClientModule, RouterModule
19   ]
20 })
21 export class RxjsModule { }
```
- rxjs-routing.module.ts:** Shows the routing configuration, including the routes array:

```
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ObservablesComponent } from './composants/observables';
4 import { ObservablesSuiteComponent } from './composants-observables-suite';
5 import { AjaxComponent } from './composants-ajax';
6 import { TpComponent } from './composants-tp';
7
8 const routes: Routes = [
9   { path: '', redirectTo: '1', pathMatch: 'full' },
10  { path: '1', component: ObservablesComponent },
11  { path: '2', component: ObservablesSuiteComponent },
12  { path: '3', component: AjaxComponent },
13  { path: '4', component: TpComponent }
14 ];
15
16 @NgModule({
17   imports: [RouterModule.forChild(routes)],
18   exports: [RouterModule]
19 })
20 export class RxjsRoutingModule { }
```

app.config.ts

src > app > app.config.ts > appConfig > providers

```
3
4 import { routes } from './app.routes';
5 import { provideAnimationsAsync } from '@angular/platform-browser/animations/async';
6 import { provideHttpClient } from '@angular/common/http';
7
8 export const appConfig: ApplicationConfig = {
9   providers: [
10     provideZoneChangeDetection({ eventCoalescing: true }),
11     provideRouter(routes),
12     provideAnimationsAsync(),
13     provideHttpClient()]
14 };
15
```

app.routes.ts

src > app > app.routes.ts > ...

```
1 import { Routes } from '@angular/router';
2 import { HomeComponent } from './webApp/accueil/components/home/home.component';
3 import { CommandesComponent } from './webApp/compte-client/components-routing/commandes/commandes.component';
4 import { LivraisonsComponent } from './webApp/compte-client/components-routing/livraisons/livraisons.component';
5 import { SignalsComponent } from './webApp/signals/components/signals/signals.component';
6 import { DivesListComponent } from './webApp/dives-list/components/dives-list/dives-list.component';
7 import { ContactsComponent } from './webApp/contacts/components/contacts/contacts.component';
8 import { Page404Component } from './shared/shared-components/page404.component';
9 import { RxjsComponent } from './webApp/rxjs/components/rxjs/rxjs.component';
10
11
12 export const routes: Routes = [
13   { path: '', component: HomeComponent, title: 'Angular 18 - StandAlone Components' },
14   { path: 'liste-des-plongees', component: DivesListComponent, title: 'Liste des plongees' },
15   { path: 'les-signals-angular-17', component: SignalsComponent, title: 'Les signals Angulars 17 - trop Cool' },
16   { path: 'les-observables', component: RxjsComponent, title: 'Rxjs et les observables' },
17
```

```
export const routes: Routes = [
  { path: '', component: HomeComponent, title: 'Angular 18 - StandAlone Components' },
  { path: 'liste-des-plongees', component: DivesListComponent, title: 'Liste des plongees' },
  { path: 'les-signals-angular-17', component: SignalsComponent, title: 'Les signals Angulars 17 - trop Cool' },
  { path: 'les-observables', component: RxjsComponent, title: 'Rxjs et les observables' },

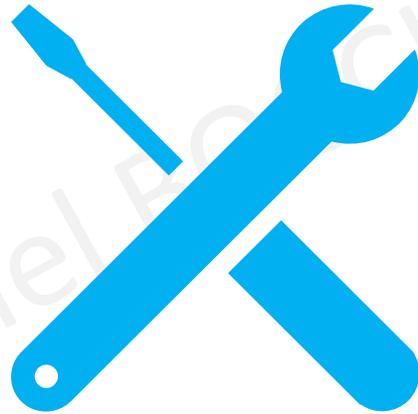
  // Avec le Lazy Loading Component 😊
  {
    path: 'compte-client', title: 'Mon espace client - Formation Angular',
    loadChildren:
      () => import('./webApp/compte-client/components/compte-client/compte-client.component')
        .then(
          (c) => c.CompteClientComponent
        ),
    children: [
      { path: 'mes-commandes', component: CommandesComponent },
      { path: 'mes-livraisons', component: LivraisonsComponent },
    ]
  }
]
```

TP : Tester le mode lazy Loading en compilation

Y'a-t-il vraiment un fichier « lazy chunk file » créé ?

Copyright Michel BOCCIOLESI - ORSYS

# La réactivité en Javascript



Copyright Michel BOCCIOLESI - ORSYS

# Réactivité de Javascript

Javascript n'est pas du tout réactif ... 🤔

Finalement quel Problème doit on résoudre ?

Tous les Frameworks modernes essaient de résoudre ce même problème

Copyright Michel BOCCIOLESI - ORSYS

## La notion de réactivité en Front End

	A	B
1	Montant HT	150
2	Tva	0,2
3	Montant TTC	=B1*B2+B1

```
index.html × script.js ×  
1 let montantHT=150;  
2 const TVA=0.2;  
3 let total=montantHT*TVA + montantHT;  
4 console.log(total);  
5  
6 let montantHt=200;  
7 console.log('Est ce réactif ? : ', total, '😞😞');  
8
```

Console ×

```
180  
Est ce réactif ? : 180 😞😞
```

# La notion de réactivité en Front End

Comment implémenter cela dans nos applis JS ? (Google Sheets l'a déjà fait avec du « *fichier Excel* » en JS)

3 Méthodes pour implémenter la réactivité dans les Frameworks JS :

## 1- Le fonctionnement de base d'Angular : Value Based :

L'état actuel d'un composant est stockée dans une zone mémoire précise (store, composant.ts, local Storage, etc ...)

le Framework utilise le mode « **Detection Change<sup>1</sup>** » ou « **Dirty Checking<sup>2</sup>** » pour savoir si la valeur a changé car il ne peut pas l'observer, il parcourt l'ensemble des composants pour vérifier les différents états et mettre à jour le DOM. La librairie actuelle\* Zone.js fait ce travail de vérification.

Avantages : aucunes connaissances du « **core** » **Zone.js** n'est requise et c'est extrêmement simple à utiliser pour le dév. Le DOM est automatiquement MAJ.

Inconvénients: Performances 🤔 🤖, des quantités de traitements sont faits pour mettre à jour les Vues HTML. Pour améliorer cela, il faut devenir expert en **Zone** et utiliser le **change detector** et le **onPush** !

1 : Pour Angular

2 : utilisé dans d'autres framework

# La notion de réactivité en Front End

3 Méthodes pour implémenter la réactivité dans les Frameworks JS :

## 2- La méthode **Observable Based**:

Cette méthode est beaucoup plus performante que la « Value based » mais demande une solide investigation et des temps d'intégration et de compréhension conséquents.

La notion d'abonnement permet de mettre à jour le DOM uniquement lorsqu'on en aura besoin !

\*Rappel :

1 « Observable » est un Objet qui émet une séquence de valeurs

1 « Observer » observe l'observable et réagit à l'arrivée de **nouvelles** valeurs

« Subject » et « Behaviour Subject (avec valeur initiale) » sont en même temps Observables et Observers

:

Le « Subject » est **multidiffusion** par rapport à l'observable qui est **monodiffusion**

# La notion de réactivité en Front End

3 Méthodes pour implémenter la réactivité dans les Frameworks JS :

## 3- La méthode **Signal()**:

Introduit dans la version 16, validé avec la version 17, le signal va révolutionner la réactivité dans Angular !

Le signal représente l'état d'un composant, d'une propriété, d'une valeur spécifique.

Lorsque le signal est mis à jour, les parties de l'application qui dépendent de ce signal peuvent être informées et modifier leurs propres données.

(mécanisme d'optimisation que l'on reconnaît dans le Store)

Lorsque une valeur change, le signal émet une information et l'application peut se mettre à jour de manière sélective.

# Reactive eXtensions for JavaScript

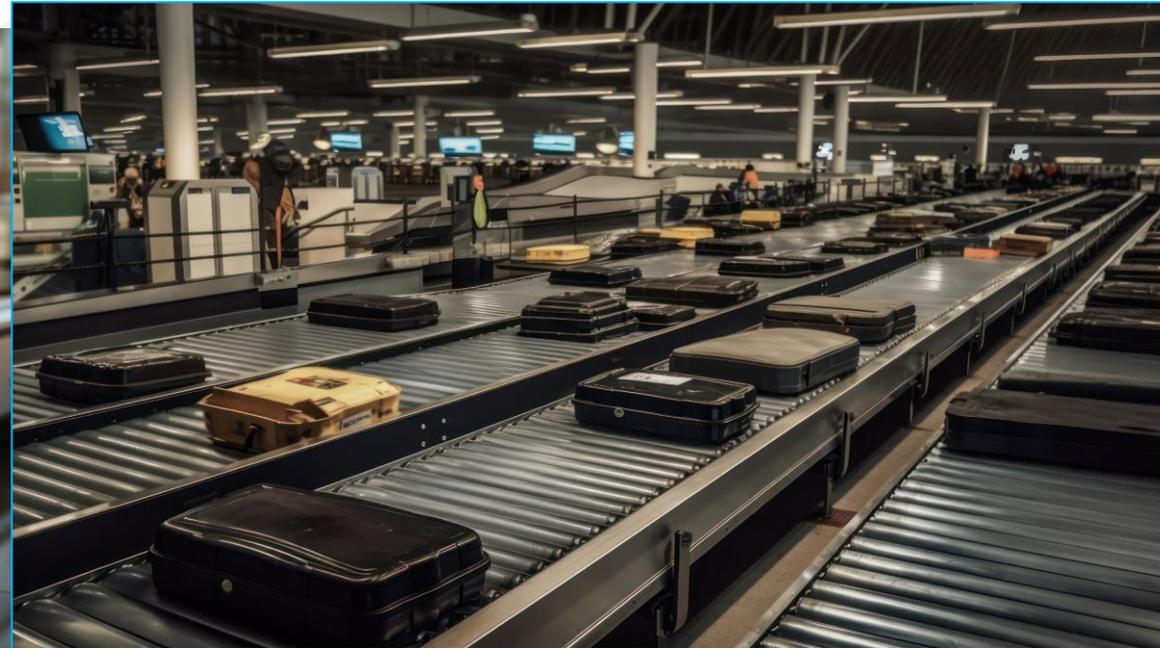
## Les Observables



Copyright Michelangelo Sciolesi - ORSYS

# RXJS et les OBSERVABLES

- 1 Observable est un objet qui émet une suite (une séquence) de valeurs (datas) dans le temps.  
Exemple : une requête HTTP
- On pourrait comparer cette suite de valeurs émises aux valises et bagages qui défilent sur un tapis roulant
- Chaque bagage a une destination mais va peut être croiser d'autres bagages qui n'ont pas la même destination mais empruntent le même tapis
- Ces valeurs peuvent être reçues par un poste de tri, interceptées, regroupées en fonction de critères et réaiguillées dynamiquement.
- On peut également stopper le tapis roulant
- Cela offre beaucoup de souplesse dans le traitement des données et la gestion des requêtes asynchrones, la gestion des multiples interactions utilisateurs



# RXJS et les OBSERVABLES

<https://rxjs-dev.firebaseapp.com/api>

Observable = 1 objet typé qui émet des valeurs dans le temps ==> forme une séquence de valeurs ou un Stream ou un flux de datas émises

Subscriber = 1 abonné qui avec la méthode .subscribe() peut recevoir ces valeurs émises(Stream)

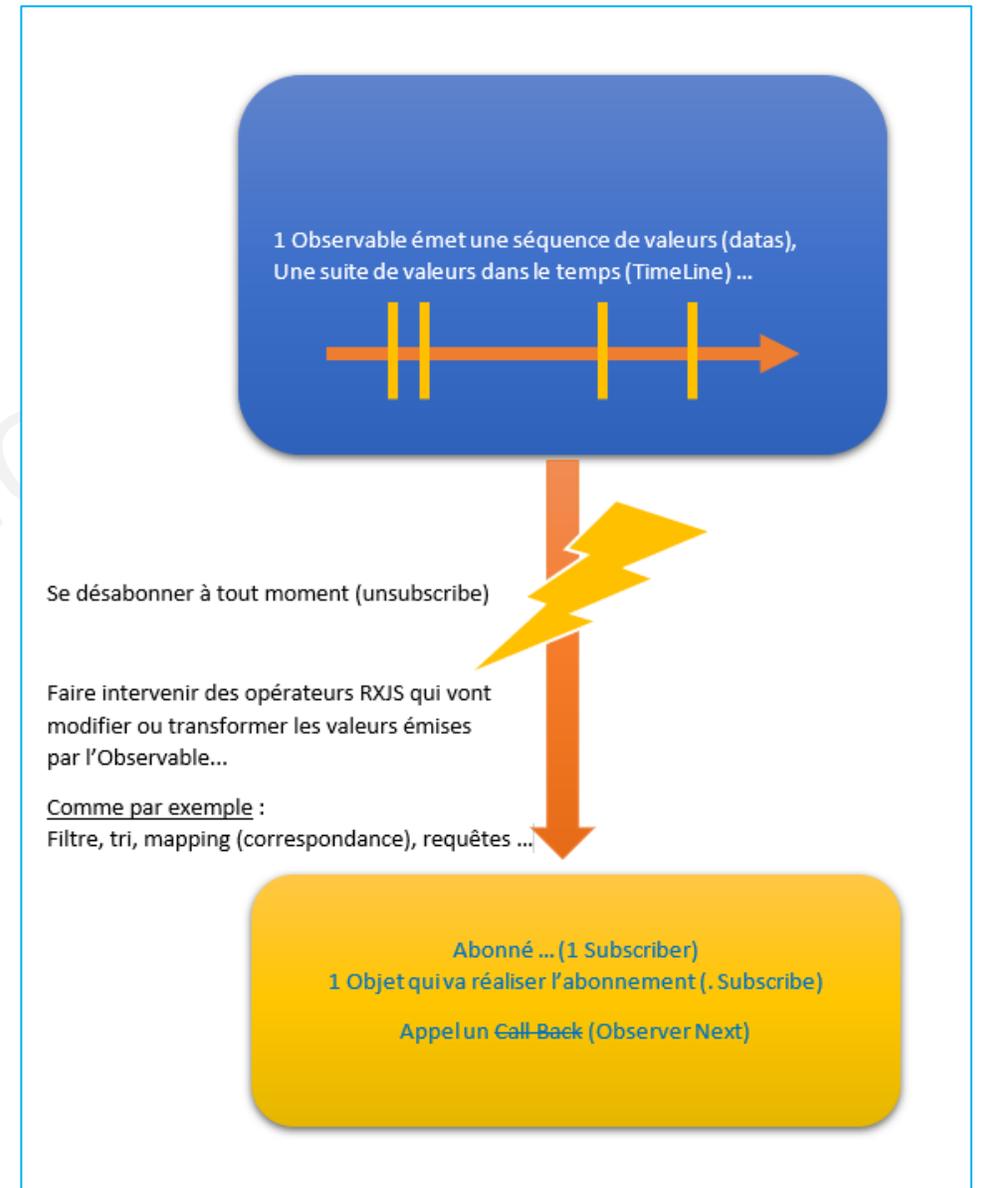
Opérateurs : des fonctions tap, map, merge, filter qui vont pouvoir modifier/transformer ce flux de valeurs émises par l'observable

Dans le subscribe => Les observers (fcts de call back) qui se définissent dans le subscribe : Next - Error – Complete

La gestion des erreurs est mieux optimisée encore avec les Observables car on peut se **désabonner** !

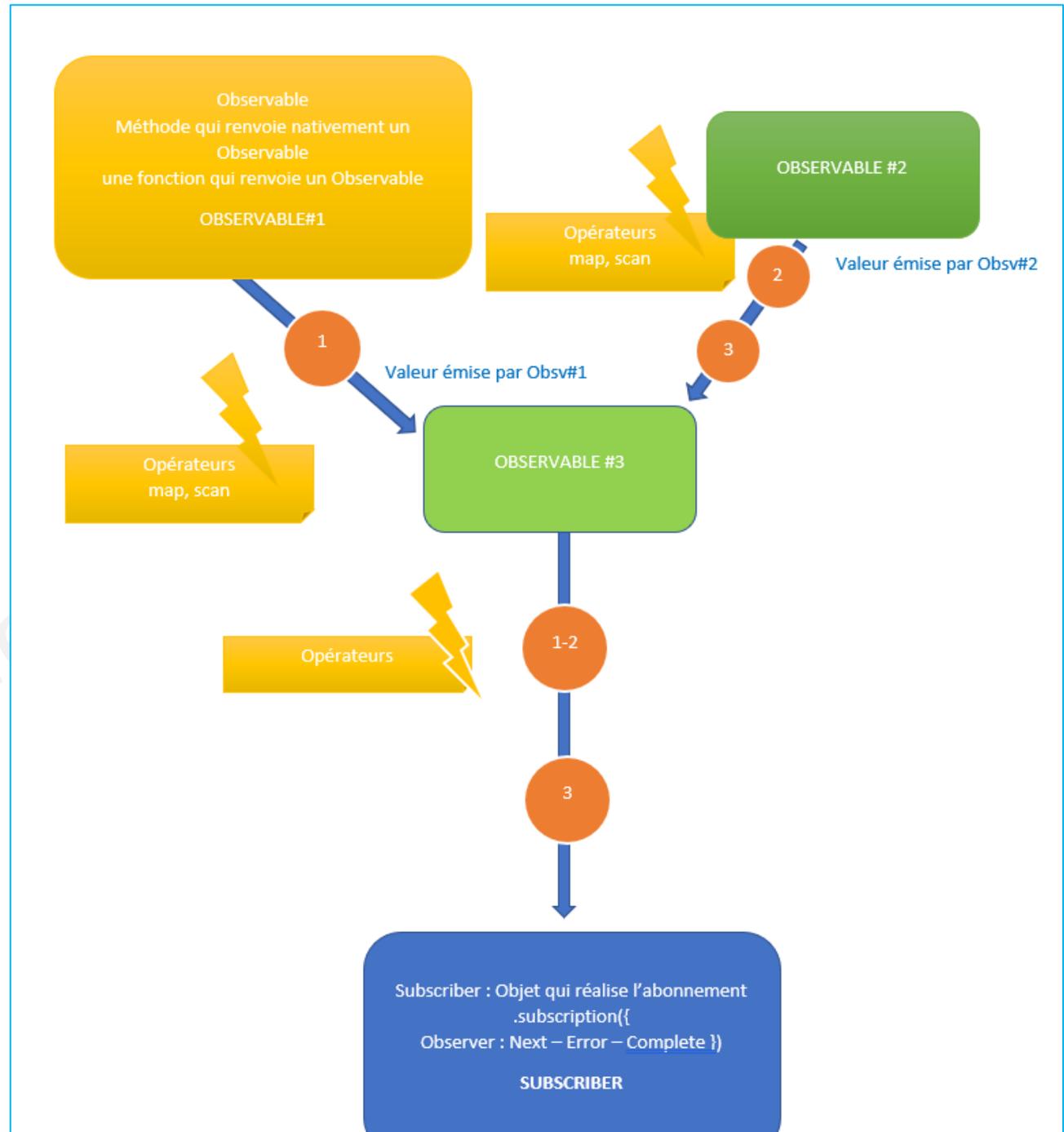
Les séquences de valeurs émises sont facilement **annulables** !

On peut facilement les **recomposer** ou les **recombinaer** pour former une nouvelle séquence de valeurs !

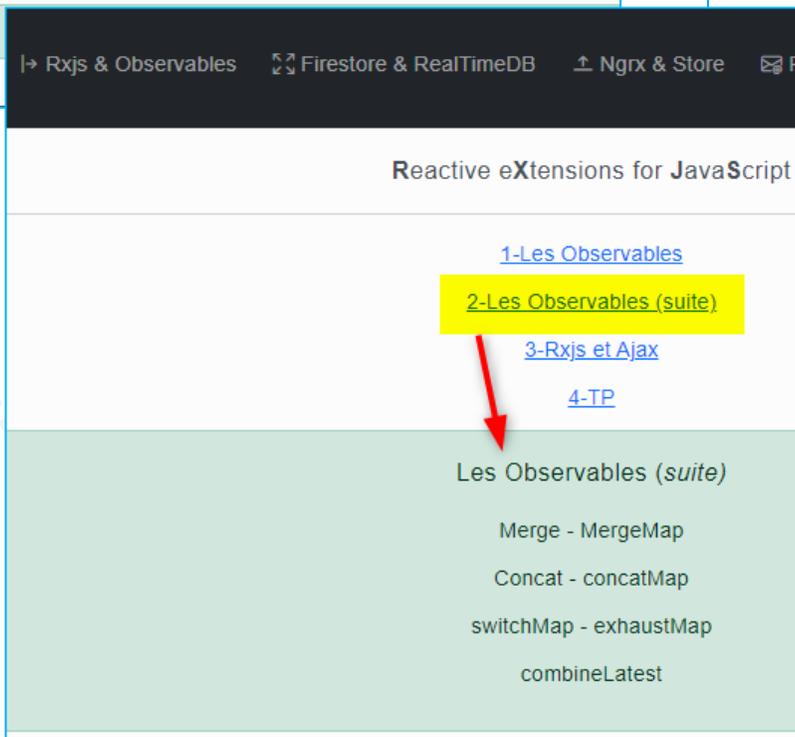
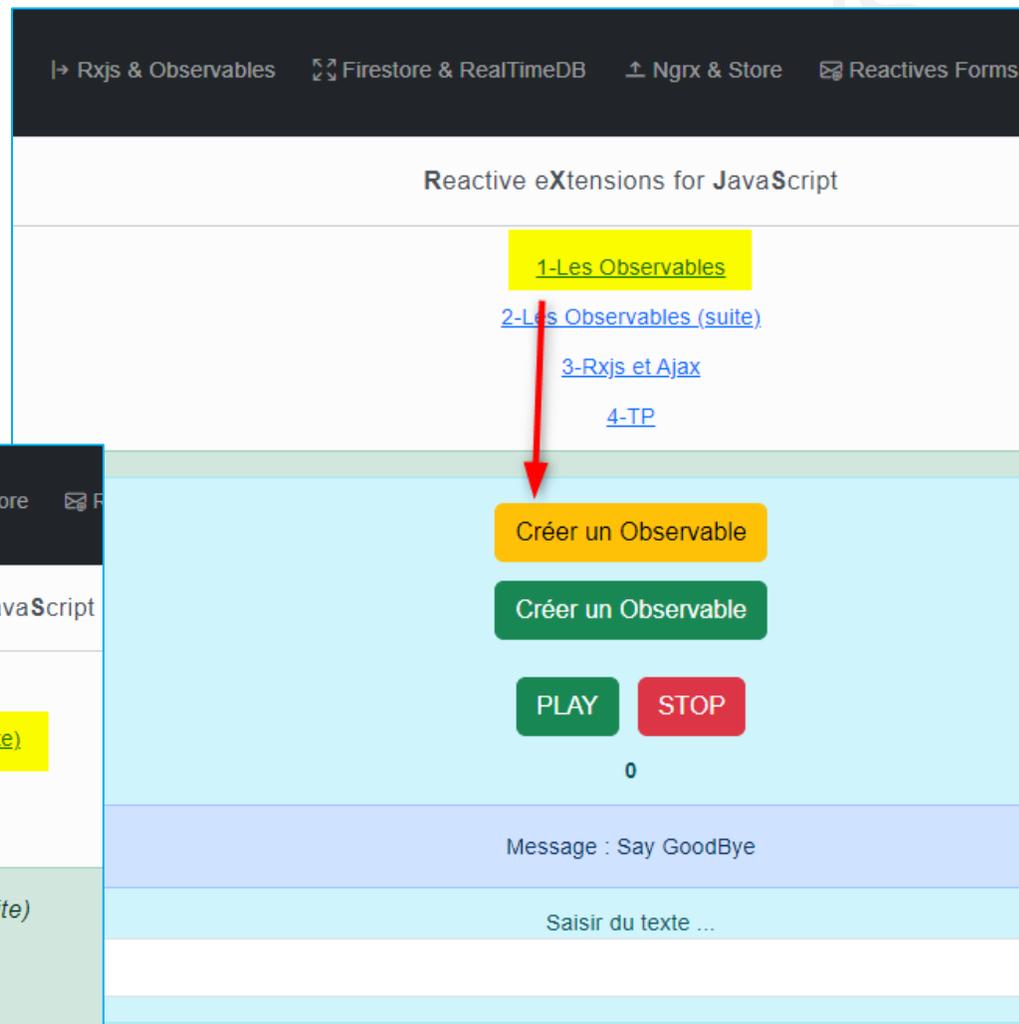
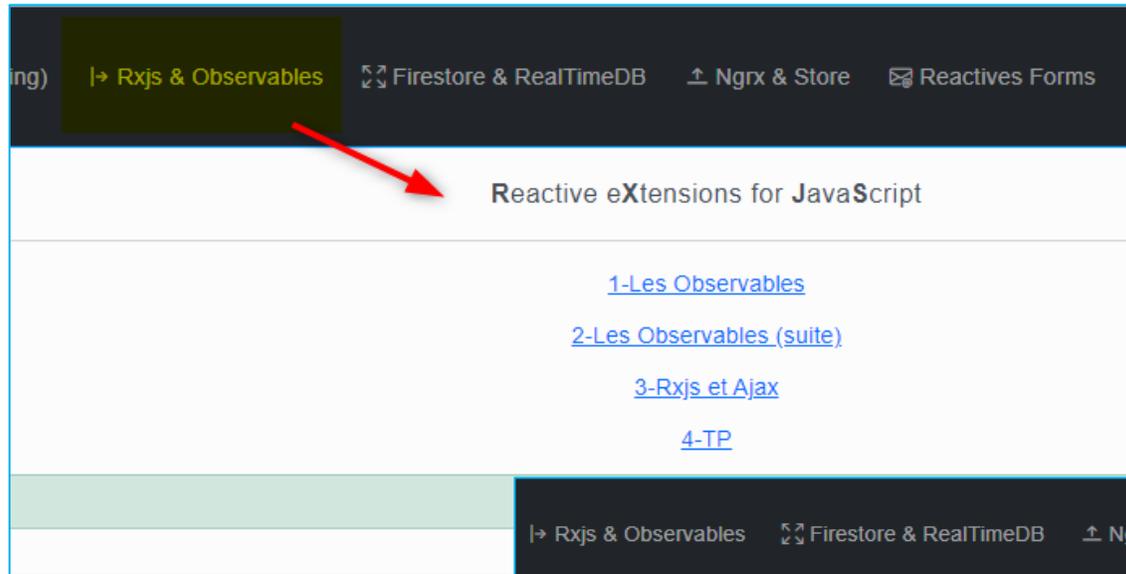


# RXJS et les OBSERVABLES

On peut facilement les **recomposer**  
ou les **recombiner** pour former  
une nouvelle séquence de valeurs !



## TP : déployer l'arborescence en Lazy Loading RXJS



Copyr.

# Exercice #1



observables.component.html

```
p-corrigé > src > app > webApp > formation > rxjs > composants-routing > observables > observables.component.html > div.alert.ale
<div class="alert alert-info">
  <button class="btn btn-warning" (click)="createObservable1()">Créer un Observable</button>
  <ul #formations1 class="list-group"></ul>
  <p></p>

  <button class="btn btn-success" (click)="createObservable2()">Créer un Observable</button>
  <p></p>

  <ul #formations2 class="list-group"></ul>
  <p></p>

  <button class="btn btn-success" (click)="play()">PLAY</button>
  &nbsp;
  <!-- <button class="btn btn-danger" (click)="stop()">STOP</button> -->
  <button class="btn btn-danger" #stop >STOP</button>

  <p></p>
  <p><strong>{{temps}}</strong></p>

  <p></p>
  <p class="alert alert-primary">Message : {{ infos$ | async }}</p>

  <p></p>
  <label for="">Saisir du texte ...</label>
  <input type="text" class="form-control" #texte|
</div>
```

Reactive eXtensions for JavaScript

[1-Les Observables](#)

[2-Les Observables \(suite\)](#)

[3-Rxjs et Ajax](#)

[4-TP](#)

Créer un Observable

NG16

Créer un Observable

NG 16 : 4 jours.

VUE : 4 jours.

PLAY STOP

0

Message : Say GoodBye

Saisir du texte ...

## Exercice #1



observables.component.ts

```
// Création de l'observable
const formation$: Observable<string> = from(formationsArray);
console.log(formation$);

this.subscription = formation$
  .pipe(
    first()
  ).subscribe({
    // la notion d'Observer : next | error | complete
    next:
      (formation: string) => {
        console.log(formation);
        const eltLi = <HTMLElement>document.createElement('li');
        eltLi.innerHTML = formation;
        eltLi.className = 'list-group-item';
        this.eltU11.nativeElement.appendChild(eltLi);
      },
    // error
    error:
      (e) => {
        console.error(e);
      },
    // complete
    complete:
      () => {
        console.warn('Complete');
      }
  });
}
```

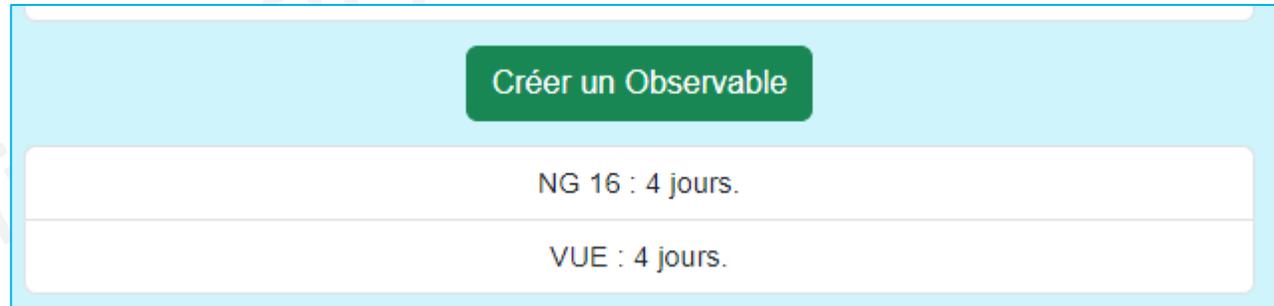
## Exercice #1



observables.component.ts

TP : créer l'observable associée à la méthode createObservable2

- seules les formations de 4 jours seront affichées
- gérer l'observer Next dans le tap



```
// -----  
public createObservable2: any = () => {  
  // tableau à 2 entrées  
  const formationsArray2: Formation[] = [  
    { cours: 'NG 16', duree: 4 },  
    { cours: 'REACT 17', duree: 3 },  
    { cours: 'VUE*', duree: 2 },  
    { cours: 'VUE', duree: 4 },  
    { cours: 'ECMA SCRIPT*', duree: 2 },  
    { cours: 'TYPESCRIPT', duree: 5 }  
  ];  
  // console.table(formationsArray2);  
  // -----  
  // création d'un observable SANS le nommer  
  from(formationsArray2)  
  .pipe(  
    // permet d'enchaîner les opérateurs RXJS  
    tap(  
      // observer NEXT  
      (formation: Formation) => console.table(formation)  
    ),  
    // first(),  
    // last(),  
    // first(  
    //   // prédicat === patern  
    //   (formation: Formation) => {  
    //     return formation.duree === 2 ;  
    //   }  
    // ),  
    delay(3000),  
  )  
}
```



Copyright

RSYS

## Exercice #2



### Observables-suite.component

```
ngOnInit() {  
  // Merge : une simple émission en parallèle  
  // -----  
  const observableA$ = interval(1000)  
    .pipe(take(3), map(  
      (valObs) => `ObsA ${valObs}`  
    ));  
  const observableB$ = interval(1000)  
    .pipe(take(3), map(  
      (valObs) => `ObsB ${valObs}`  
    ));  
  
  merge(observableA$, observableB$)  
    // émission en parallèle  
    .subscribe(  
      (valObs) => console.log('merge', valObs)  
    );  
  
  // Concat : une émission non plus en parallèle mais séquentielle  
  // le 1er observable doit avoir terminé ou envoyer complete  
  // avant que le second puisse être interprété  
  // -----  
  
  concat(observableA$, observableB$)  
    // émission après complete  
    .subscribe(  
      (valObs) => console.warn('concat', valObs)  
    );  
}
```

Les fonctions de regroupement et de recombinaison de séquences

merge ObsA 0
merge ObsB 0
▲ ▶ concat ObsA 0
merge ObsA 1
merge ObsB 1
▲ ▶ concat ObsA 1
merge ObsA 2
merge ObsB 2
▲ ▶ concat ObsA 2
▲ ▶ concat ObsB 0
▲ ▶ concat ObsB 1
▲ ▶ concat ObsB 2

## Exercice #2



### Observables-suite.component

### Les fonctions de regroupement et de recombinaison de séquences

```
// -----  
// map - mergeAll - mergeMap - concatmap  
// -----  
const API_URL = 'http://localhost:3001/formations';  
  
of(1, 2, 3)  
  .pipe(  
    map(valObs => this._http.get<any[]>(`${API_URL}/${valObs}`)),  
    mergeAll()  
  )  
  .subscribe(  
    (valObs) => console.log('map + mergeMap : ', valObs)  
  )  
// -----  
of(1, 2, 3)  
  .pipe(  
    mergeMap(valObs => this._http.get<any[]>(`${API_URL}/${valObs}`))  
  )  
  .subscribe(  
    (valObs) => console.log('mergemap : ', valObs)  
  );
```

```
map + mergeMap : ▶ {id: 1, cours: 'Angular - Les fondamentaux', codeCours: 'AGU', duration: 4}  
map + mergeMap : ▶ {id: 2, cours: 'Angular - concepts avancés', codeCours: 'ANY', duration: 3}  
map + mergeMap : ▶ {id: 3, cours: 'Javascript', codeCours: 'DHL', duration: 4}  
mergemap : ▶ {id: 1, cours: 'Angular - Les fondamentaux', codeCours: 'AGU', duration: 4}  
mergemap : ▶ {id: 2, cours: 'Angular - concepts avancés', codeCours: 'ANY', duration: 3}  
mergemap : ▶ {id: 3, cours: 'Javascript', codeCours: 'DHL', duration: 4}
```

## Exercice #2



Observables-suite.component

Les fonctions de regroupement et de recombinaison de séquences

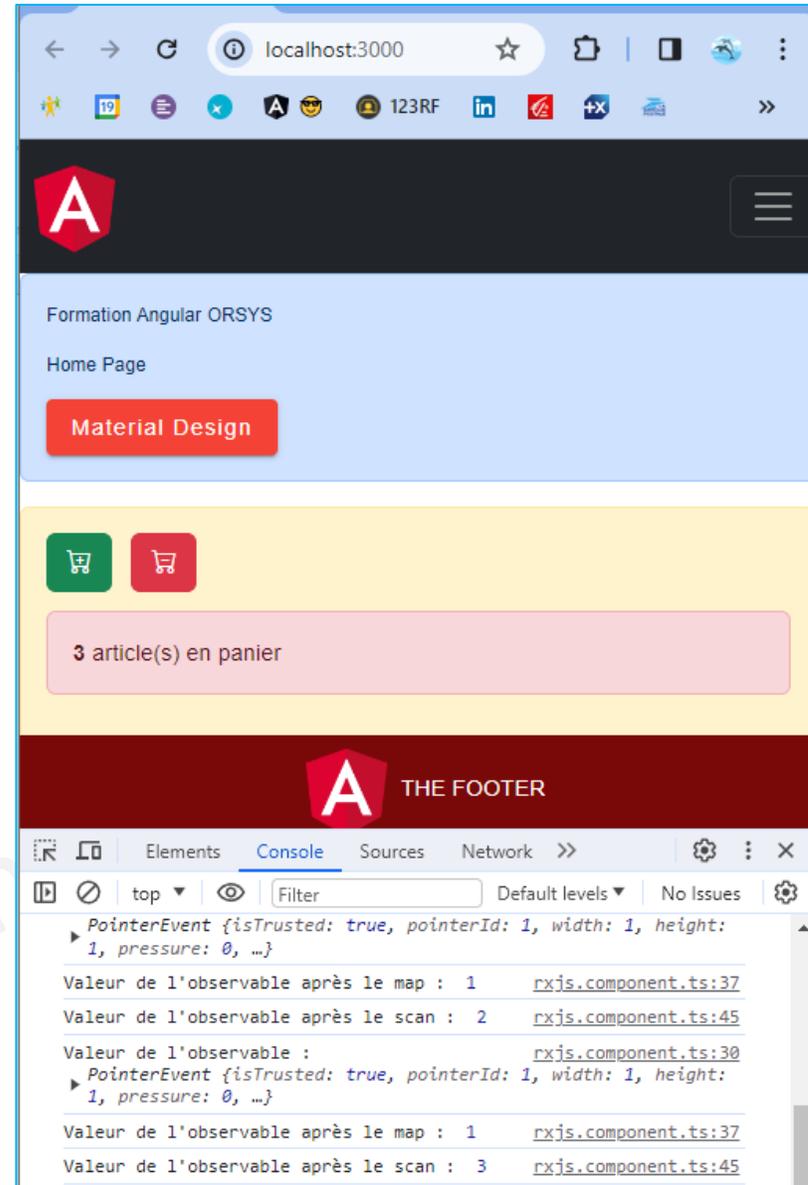
```
▶ switchmap : ▶ {id: 3, cours: 'Javascript', codeCours: 'DHL', duration: 4}  
▶ exhaustmap : ▶ {id: 1, cours: 'Angular - Les fondamentaux', codeCours: 'AGU', duration: 4}
```

```
// -----  
of(1, 2, 3)  
  .pipe(  
    switchMap(valObs => this._http.get<any[]>(`${API_URL}/${valObs}`))  
  )  
  .subscribe(  
    (valObs) => console.warn('switchmap : ', valObs)  
  );
```

```
of(1, 2, 3)  
  .pipe(  
    exhaustMap(valObs => this._http.get<any[]>(`${API_URL}/${valObs}`))  
  )  
  .subscribe(  
    (valObs) => console.warn('exhaustmap : ', valObs)  
  );
```

Intéressant : les requêtes précédentes ou suivantes sont stoppées

# TP (Regroupement avec combinelatest)



The screenshot displays a web browser window at localhost:3000. The page features a dark header with an Angular logo and a navigation menu. Below the header, there is a light blue section titled "Formation Angular ORSYS" with a "Home Page" link and a red "Material Design" button. A yellow section contains two shopping cart icons and a pink notification box stating "3 article(s) en panier". The footer is dark red with the Angular logo and the text "THE FOOTER".

The browser's developer console is open, showing the following log entries:

```
PointerEvent {isTrusted: true, pointerId: 1, width: 1, height: 1, pressure: 0, ...}
Valeur de l'observable après le map : 1 rxjs.component.ts:37
Valeur de l'observable après le scan : 2 rxjs.component.ts:45
Valeur de l'observable : rxjs.component.ts:30
  PointerEvent {isTrusted: true, pointerId: 1, width: 1, height: 1, pressure: 0, ...}
Valeur de l'observable après le map : 1 rxjs.component.ts:37
Valeur de l'observable après le scan : 3 rxjs.component.ts:45
```

# RXJS et les OBSERVABLES

The image shows a development environment with three main components: a code editor on the left, a code editor in the middle, and a browser on the right.

**Left Code Editor (body.component.html):**

```
1 <div class="alert alert-primary">
2
3 <h3>Formation Angular ORSYS</h3>
4 <h2>Home Page</h2>
5 <button mat-raised-button color="warn">Material Design</button>
6
7
8 <div class="alert alert-warning">
9
10 <app-rxjs></app-rxjs>
11
12
```

**Middle Code Editor (rxjs.component.ts):**

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-rxjs',
5   templateUrl: './rxjs.component.html',
6   styleUrls: ['./rxjs.component.scss']
7 })
8 export class RxjsComponent {
9
10 }
11
```

**Bottom Code Editor (rxjs.component.html):**

```
1 <button class="btn btn-success" #plus><i class="bi bi-plus">
2   </i>
3 <button class="btn btn-danger" #moins><i class="bi bi-minus">
4   </i>
5 </button>
6
7 <p class="alert alert-danger" #panier></p>
8
9
```

**Browser (localhost:3000):**

The browser displays the rendered application. It features a dark navigation bar with a white 'A' logo. Below the navigation bar, there is a light blue header area containing the text "Formation Angular ORSYS" and "Home Page". A prominent red button labeled "Material Design" is centered in this header. Below the header, there is a yellow section containing two shopping cart icons (one green, one red) and a light pink search bar.

```
// -----  
// création de l'observable #1 : PLUS  
// -----  
const plus$ = fromEvent(this.eltPlus.nativeElement, 'click')  
  .pipe(  
    startWith(0),  
    tap(  
      (valObs) => console.log(`Valeur de l'observable : `, valObs)  
      // output : PointerEvent {isTrusted: true ...  
    ),  
    map(  
      () => { return 1 }  
    ),  
    tap(  
      (valObs) => console.log(`Valeur de l'observable après le map : `,  
        valObs)  
      // output : 1  
    ),  
    scan(  
      (accu: number, nouvelleValeur: number) => { return accu + nouvelleValeur }  
    ),  
    tap(  
      (valObs) => {  
        console.log(`Valeur de l'observable après le scan : `, valObs);  
        // output : 2 - 3 - 4 - 5 ....  
        this.eltMoins.nativeElement.style.display = 'inline';  
      }  
    )  
  )  
  .subscribe(  
    (valObs) => this.eltPanier.nativeElement.innerHTML = `${valObs}`  
    </strong> article(s) en panier`  
  );
```

```

// -----
// création de l'observable #2 : MOINS
// -----

const moins$ = fromEvent(this.eltMoins.nativeElement, 'click')
  .pipe(
    startWith(0),
    map(
      () => { return 1 }
    ),
    scan(
      (accu: number, nelleValeur: number) => { return accu + nelleValeur }
    )
  )
// .subscribe(
//   (valObs) => this.eltPanier.nativeElement.innerHTML = `${valObs}
</strong> article(s) en panier`
// );

```

```

// -----
// Re-Composition des 2 observables : Combinaison - Fusion(merge)
// -----

combineLatest([plus$, moins$])
  .pipe(
    map(
      ([valObs1, valObs2]) => {
        return valObs1 - valObs2;
      }
    )
  )
  .subscribe(
    (valObs) => {
      this.eltPanier.nativeElement.innerHTML = `${valObs}</strong>
article(s) en panier`;

      if (valObs === 0) {
        // this.eltMoins.nativeElement.setAttribute('disabled', 'true');
        this.eltMoins.nativeElement.style.display = 'none';
      }
    }
  );

```

Copyright Michel

# TP (Requêtes Ajax : forkJoin)

Comptes) Compte Client(Lazy Loading) Rxjs & Observables Firestore & RealTimeDB Ngrx & Store Reactives Forms Unit Tests

## Reactive eXtensions for JavaScript

[1-Les Observables](#)  
[2-Les Observables \(suite\)](#)  
**[3-Rxjs et Ajax](#)**  
[4-TP](#)

[getdatas](#)

Titre	Description
Episode I : La Menace Fantôme	La République Galactique est en pleine ébullition. La taxation des routes commerciales reliant les systèmes éloignés provoque la cupide Fédération du Commerce et ses redoutables vaisseaux de guerre imposent un blocus à la petite planète Naboo.Face à ce Congrès de la République s'enlise dans des débats sans fin, le Chancelier Suprême charge en secret deux Chevaliers Jedi, gard galaxie, de résoudre le conflit ...
Episode II : L'Attaque des Clones	L'agitation règne au Sénat Galactique. Des milliers de systèmes solaires ont annoncé leur intention de quitter la République. Con mené par le Comte Dooku, les Chevaliers Jedi, en nombre limité, ont du mal à maintenir la paix et l'ordre dans la galaxie. La sén Naboo, revient au Sénat Galactique participer à un vote crucial sur la création d'une Armée de la République pour aider les Jedi c
Episode III : La Revanche des Sith	C'est la guerre ! La République croule sous les attaques de l'impitoyable Sith, le Comte Dooku. Il y a des héros dans les deux ca audace stupéfiante, le Général Grievous, diabolique chef droïde, est entré dans la capitale pour enlever le Chancelier Palpatine, Alors que l'Armée Séparatiste Droïde tente de quitter la capitale assiégée avec son précieux otage, deux Chevaliers Jedi mènent secourir le Chancelier captif...

# TP (Requêtes Ajax : forkJoin)

```
ajax.component.html x rxjs.module.ts
webApp-corrige > src > app > webApp > formation > rxjs > composants-routing > ajax > ajax.component.html >
1 <button class="btn btn-primary" (click)="getDatas()">getdatas</button>
2
3 <table mat-table [dataSource]="films" class="mat-elevation-z8">
4
5   <ng-container matColumnDef="title">
6     <th mat-header-cell *matHeaderCellDef> Titre </th>
7     <td mat-cell *matCellDef="let element"> {{element.title}} </td>
8   </ng-container>
9
10
11   <ng-container matColumnDef="desc">
12     <th mat-header-cell *matHeaderCellDef> Description </th>
13     <td mat-cell *matCellDef="let element"> {{element.desc}} </td>
14   </ng-container>
15
16
17   <tr mat-header-row *matHeaderRowDef="displayedColumns"></tr>
18   <tr mat-row *matRowDef="let row; columns: displayedColumns;"></tr>
19
20 </table>
```

```
// -----
public getDatas = () => {

  const URL1 = 'http://localhost:3001/films1';
  const URL2 = 'http://localhost:3001/films2';

  forkJoin({
    rst1:ajax.getJSON(URL1) as Observable<Films>,
    rst2:ajax.getJSON(URL2) as Observable<Films>
  })
  .pipe(
    tap(
      (datas:any) => {
        this.films = datas.rst1.concat(datas.rst2);
        console.table(this.films)
      }
    )
  )
}

// -----
// forkJoin({
//   rst1:this._http.get(URL1) as Observable<Films>,
//   rst2:this._http.get(URL2) as Observable<Films>
// })
// .pipe(
//   tap(
//     (datas:any) => {
//       this.films = datas.rst1.concat(datas.rst2);
//       console.table(this.films)
//     }
//   )
// )
// )
.subscribe()
}
```

# Subject et BehaviourSubject

Un simple Observable est un objet en lecture seule. Il n'est pas possible d'invoquer l'événement ou le changement de valeur ...

**Exemple** : le client HTTP utilise un *Observable* pour exposer l'événement à une requête HTTP.

```
<div class="row alert alert-primary">  
  <div class="col-md-3" *ngFor="let objfilm of Movies$ | async">  
    <div class="card">
```

```
movies.service.ts X  
src > app > shared > services > movies.service.ts > MovieService > getMovies$  
1  import { Injectable } from '@angular/core';  
2  import { HttpClient } from '@angular/common/http';  
3  import { map, Observable, tap } from 'rxjs';  
4  import { Movie } from '../class/movie';  
5  
6  // décorateur  
7  @Injectable({ providedIn: 'root'})  
8  
9  export class MovieService {  
10   // const (injection de dépendances)  
11   constructor(private _http: HttpClient) { }  
12  
13   // méthodes  
14   public getMovies$(): Observable<Movie[]> {  
15  
16     // const API_URL = 'http://localhost:3001/films1';  
17     const API_URL = 'https://dev.webjs.fr/JSON_API/films.json';  
18  
19     return this._http.get<Movie[]>(API_URL)  
20     .pipe(tap((responseHTTP: any) => { console.log('Depuis le  
21     service : ', responseHTTP); })),  
22     // delay(4000),  
23     map((datas: Movie[]) => {  
24       return datas.filter(  
25         (data: Movie) => {  
26           // return data.id==5;  
27           return data.id >= 1;  
28         }  
29       )  
30     }  
31   }  
}
```

```
export class MovieBehaviourService {
```

```
private moviesSubject = new BehaviorSubject<Movie[]>([]);  
public movies = this.moviesSubject.asObservable();
```

```
// asObservable permet de retourner un observable à partir d'un Subject  
// En empêchant les composants abonnés d'appeler la méthode next sur l'observable retourné  
// Ainsi uniquement le Subject initial peut être utilisé pour diffuser de nouvelles séquences de valeurs  
// Et pourra notifier les abonnés des MAJ 😊
```

```
constructor(private _http: HttpClient) { }
```

```
public getMovies() {
```

```
  // const API_URL = 'http://localhost:3001/films1';  
  const API_URL = 'https://dev.webjs.fr/JSON_API/films.json';
```

```
  this._http.get<Movie[]>(API_URL)
```

```
    .pipe(  
      catchError(  
        () => {  
          this.moviesSubject.error('Une erreur est survenue !');  
          return [];  
        }  
      ),  
      map(  
        (movies) => { return movies }  
      )  
    )
```

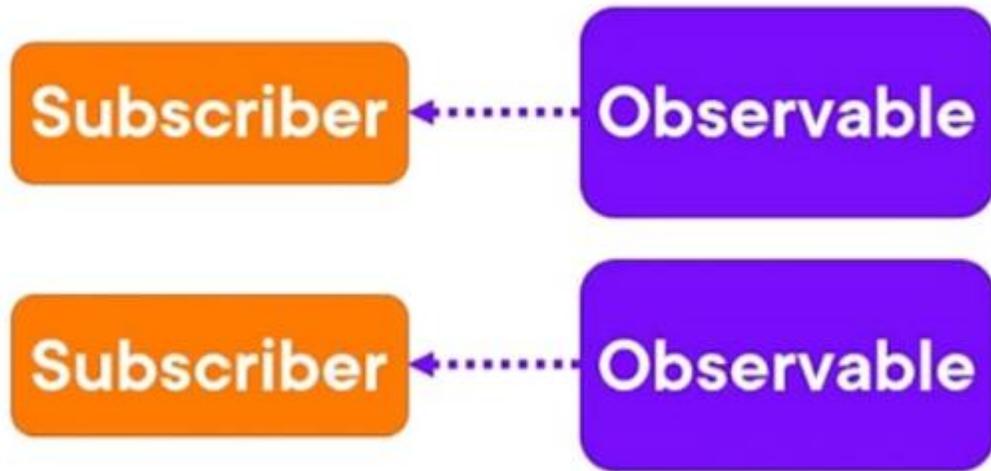
```
    .subscribe(  
      (movies) => this.moviesSubject.next(movies)  
      // next met à jour l'état courant de l'instance de Behaviour Subject  
      // la différence entre Subject et behaviourSubject est le fait que behaviour initialise une 1ère valeur par défaut  
      // cette valeur est renvoyée aux composants qui s'abonnent même après le 1er envoi de datas !  
      // même si les composants s'abonnent plus tard !  
    )
```

A l'opposé et contrairement aux *Observables*, les *Subjects* peuvent émettre des événements (valeurs) aux « observers » abonnés à l'aide de la fonction `next()`.

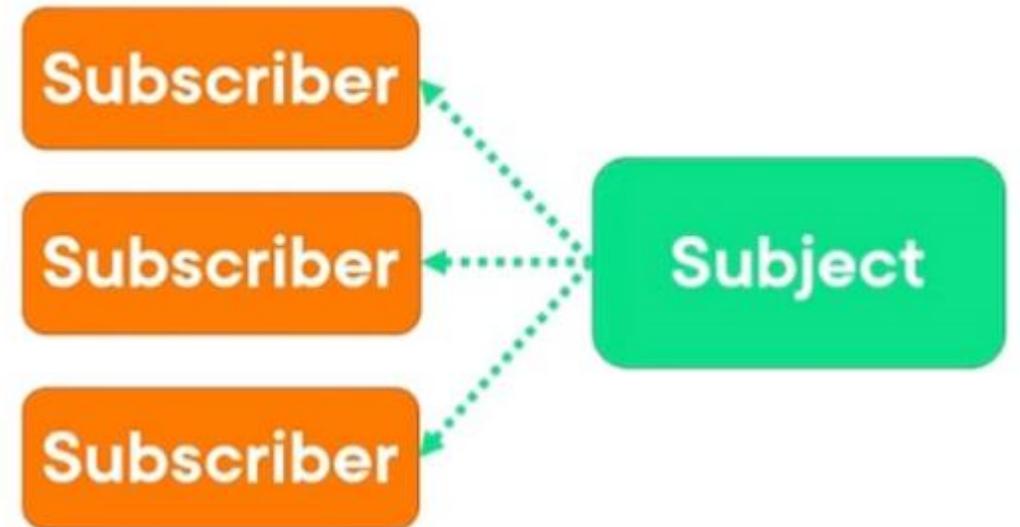
Les modifications créés dans le *Subject*, avec la fonction `next()`, pourront être écoutées avec la fonction `subscribe()` ou le pipe `async !`.

`asObservable` peut « **déguiser un *Subject* en *Observable*** » pour masquer son comportement de *Subject* et ne montrer que sa capacité à s'abonner aux modifications.

## UNICAST VS MULTICAST



L'Observable est normalement UNICAST



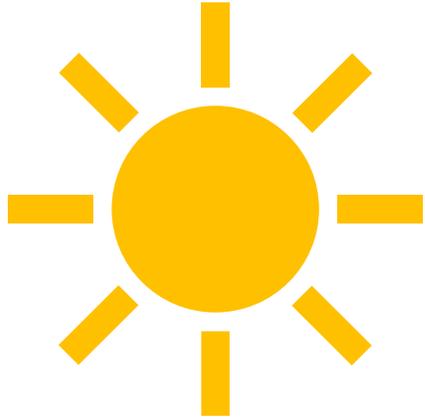
Le Subject est MULTICAST

## UNICAST VS MULTICAST

L'observable unicast ne peut être écouté que par un seul subscriber (l'observer) . Si un nouvel observer vient écouter, un nouvel Observable est créé et les anciennes valeurs sont ... perdues

L'observable multicast peut être écouté par plusieurs subscribers (observers) . Si un nouvel observer vient écouter, les valeurs déjà émises peuvent être reçues 😊

## Hot VS Cold Observables



La création de l'observable active la source des valeurs qui peuvent être émises sans subscribers ! L'Hot Observable est généralement MULTICAST

```
productSubject = new Subject<number>();  
this.productSubject.next(19);
```



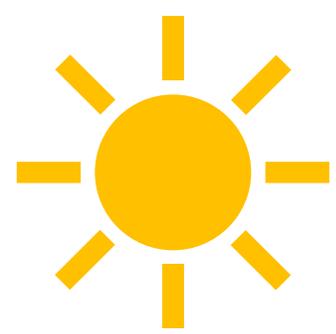
N'émet rien tant qu'il n'y a pas de subscribers donc d'abonnement (.subscribe | async ) ! L'abonnement active une Source UNICAST

```
Movies$ =this._http.get<Movies[]>(_url)
```

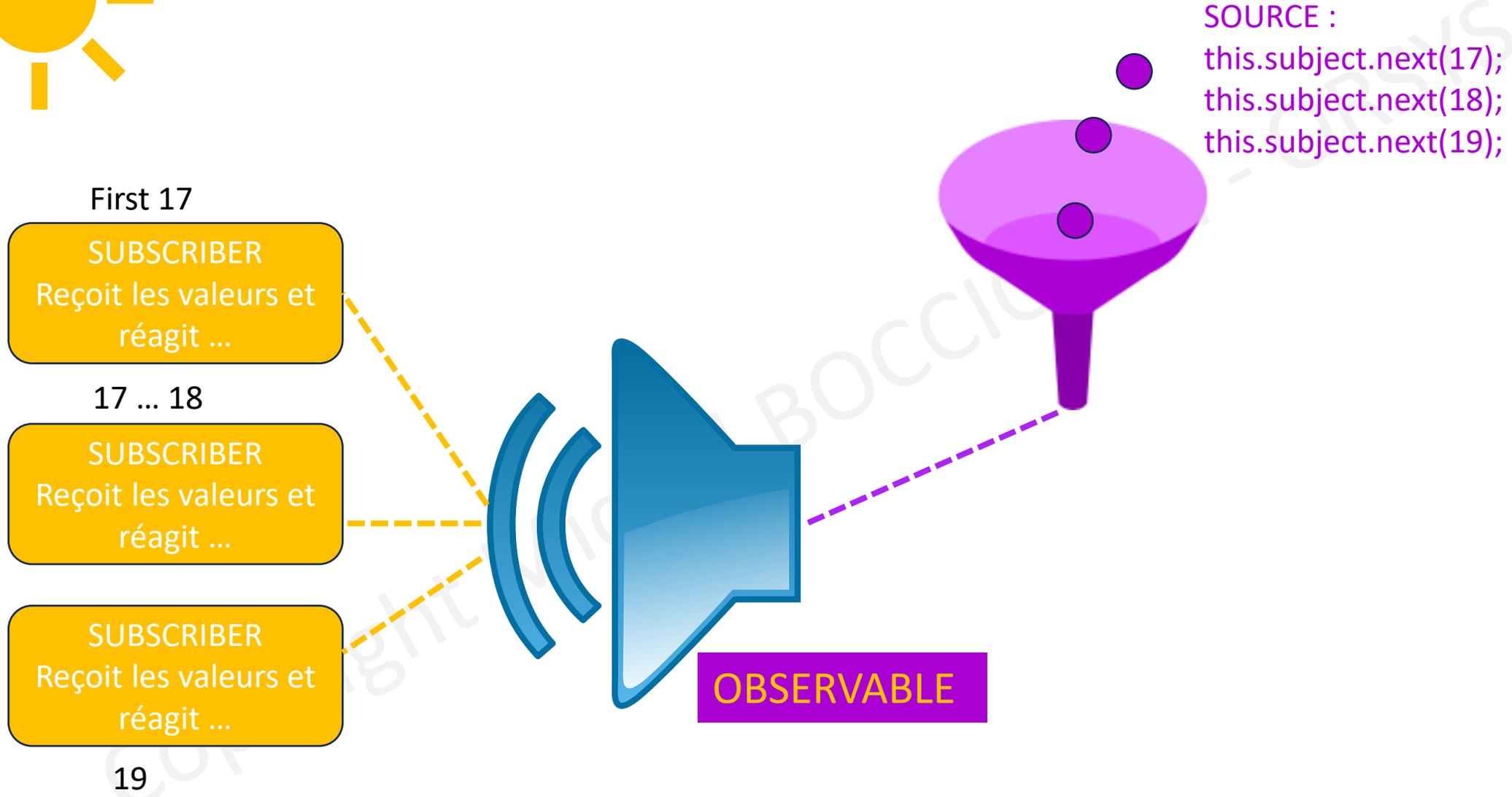


# COLD UNICAST





# HOT MULTICAST



# Le Signal



Copyright Michel BÉCICIOLESI - ORSYS

# La notion de réactivité en Front End

Finalemment quel Problème doit on résoudre ?

Tous les Frameworks modernes essaient de résoudre ce même problème

	A	B
1	Montant HT	150
2	Tva	0,2
3	Montant TTC	=B1*B2+B1

```
index.html × script.js ×  
1 let montantHT=150;  
2 const TVA=0.2;  
3 let total=montantHT*TVA + montantHT;  
4 console.log(total);  
5  
6 let montantHt=200;  
7 console.log('Est ce réactif ? : ', total, '😞😞');  
8  
Console ×  
180  
Est ce réactif ? : 180 😞😞
```

# La notion de réactivité en Front End

Comment implémenter cela dans nos applis JS ? (Google Sheets l'a déjà fait avec du « *fichier Excel* » en JS)

3 Méthodes pour implémenter la réactivité dans les Frameworks JS :

## 1- La méthode Value Based :

L'état actuel d'un composant est stockée dans une zone mémoire précise (store, composant.ts, local Storage, etc ...)

le Framework utilise le mode « **Detection Change<sup>1</sup>** » ou « **Dirty Checking<sup>2</sup>** » pour savoir si la valeur a changé car il ne peut pas l'observer, il parcourt l'ensemble des composants pour vérifier les différents états et mettre à jour le DOM. La librairie actuelle\* Zone.js fait ce travail de vérification.

Avantages : aucunes connaissances du « **core** » **Zone.js** n'est requise et c'est extrêmement simple à utiliser pour le dev. Le DOM est automatiquement MAJ.

Inconvénients: Performances 🤔 🤖, des quantités de traitements sont faits pour mettre à jour les Vues HTML. Pour améliorer cela, il faut devenir expert en **Zone** et utiliser le **change detector** et le **onPush** !

1 : Pour Angular

2 : utilisé dans d'autres framework

# La notion de réactivité en Front End

3 Méthodes pour implémenter la réactivité dans les Frameworks JS :

## 2- La méthode **Observable Based**:

Cette méthode est beaucoup plus performante que la « Value based » mais demande une solide investigation et des temps d'intégration et de compréhension conséquents.

La notion d'abonnement permet de mettre à jour le DOM uniquement lorsqu'on en aura besoin !

\*Rappel :

1 « Observable » est un Objet qui émet une séquence de valeurs

1 « Observer » observe l'observable et réagit à l'arrivée de **nouvelles** valeurs

« Subject » et « Behaviour Subject (avec valeur initiale) » sont en même temps Observables et Observers

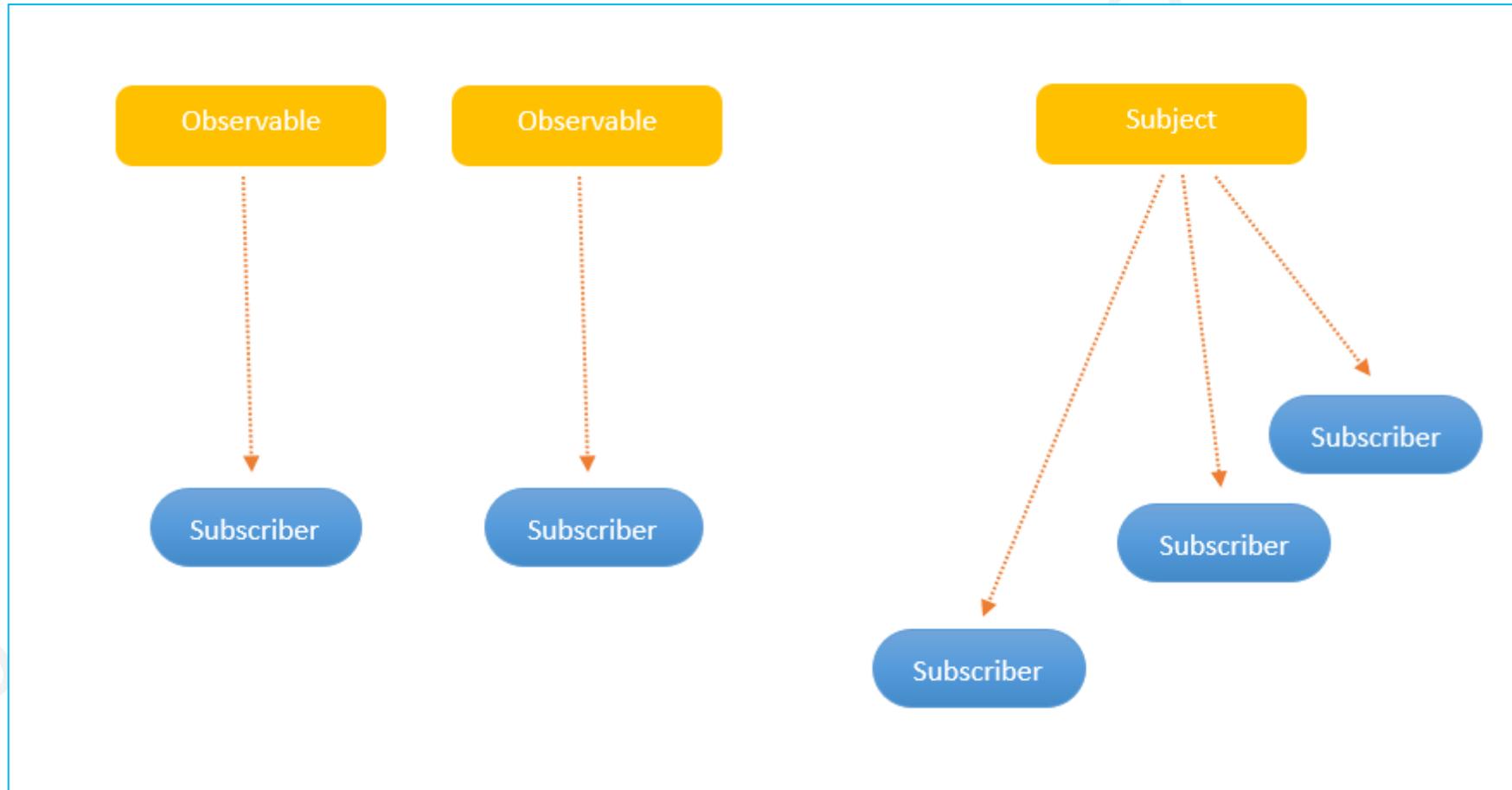
:

Le « Subject » est **multidiffusion** par rapport à l'observable qui est **monodiffusion**

# La notion de réactivité en Front End

Le « Subject » est multidiffusion par rapport à l'observable qui est monodiffusion, il faut user de complexité pour recevoir les valeurs du même Observable par plusieurs « Subscribers » 😬

Plusieurs Subscriber peuvent s'attacher au Subject et recevoir les valeurs émises. 😊



# Angular 17 – Le Signal!



Copyright Michel POCQUOLESI - ORSYS

# Le Signal

Meilleures **performances** en termes d'exécution !

Le signal réduit la **complexité du parcours de l'arbre des composants parents-enfants**

La vérification du changement d'affichage dans le composant, devient de plus en plus fine et étroitement liée au composant concerné. La **granularité** du « Change Detection Mode » est nettement améliorée.

Zone.js utilisé depuis le début d'Angular pour détecter les changements dans les composants va être progressivement arrêté ! (pour rappel, le principe d'une Zone permet d'exécuter du code dans le contexte Angular et hors contexte Angular) 🤖

Les signaux peuvent interagir avec RXJS et apporter de la **complémentarité** !

On peut passer facilement d'un Observable à un signal et vice-versa

toSignal() toObservable()

# Le Signal

Le signal représente l'état d'un composant, d'une propriété , d'une valeur spécifique.

Lorsque le signal est mis à jour, les parties de l'application qui dépendent de ce signal peuvent être informées et modifier leurs propres données.  
(mécanisme d'optimisation que l'on reconnaît dans le Store)

Lorsque une valeur change, le signal émet une information et l'application peut se mettre à jour de manière sélective.

<https://angular.dev/guide/signals>

1- Le signal permet de réduire le nombre de mise à jour effectuées lors du "Change Detection Mode" d'Angular !

La réactivité est performante ! Plus besoin de Zone.js ! Zone va ou est en train d'être abandonné ! 🌟📍

Interopérabilité avec RXJS : méthodes toSignal() et toObservable()

2- Le signal est une valeur réactive utilisée pour représenter un état ou une donnée changeante dans une application.

- signal() : crée une propriété de type signal
- set : écrase la valeur du signal et en crée une nouvelle
- update: prend la dernière valeur du signal et la transforme
- computed : permet de calculer des valeurs dérivées à partir du signal
- effects : observer les changements de valeur du signal et crée un "effet d'observation call bacjk d'action"



2 étoiles attribuée(s)

**Note :** 4/10 😊 2 (étoiles attribuée(s))



Home Page

Introduction

Compte Client

Angular datas

Rxjs

Formulaires

Le Signal

HOME PAGE - Bienvenue dans la formation Angular - Cahier d'exercices

I ❤️ Signal !!

Mettre à jour le signal ?

Change Detection Mode mettra t-il à jour la simple prop ? 😞

Mettre à jour une simple prop ?

## Child Home Component

Ce composant reçoit un service en injection de dépendances "SignalService" qui lui fournit un signal grâce à la méthode signal avec la méthode **updateSignal()** ! Le composant 🛎 signal lui aussi s'abonne au service et reçoit en temps réel la

**Signal MAJ** : Le Signal 😊 C'est Cool !

**Prop MAJ** : Prop Classique!

[Home Page](#)[Introduction ▾](#)[Compte Client](#)[Angular datas ▾](#)[Rxjs](#)[Formulaires ▾](#)[Le Signal](#)[Les News NG17+](#)[← Go Back](#)

## Signal Angular 16+

Je suis un composant enfant ( *ou très éloigné ...*), je reçois ( *m'abonne 😊*) au **service signalService** en injection de dépendances et peut accéder avec beaucoup plus de réactivités aux datas !

**Signal** : Le Signal 😊 C'est Cool !

# Le Signal Angular

- La primitive Signal()
- Computed
- Méthode Set() Update() Mutate()

```
//signal

public signalNb = signal(3);
// la primitive signal = lecture ou en ecriture

// calculé à partir d'autre signaux
public infos = computed(
  () => `${this.signalNb()} étoiles attribuée(s) !` )

//viewChildren
@ViewChild('plus') eltPlus!: ElementRef<HTMLElement>;
@ViewChild('moins') eltMoins!: ElementRef<HTMLElement>;

ngAfterViewInit() {
  // -----
  fromEvent(this.eltPlus.nativeElement, 'click').pipe(
    tap( () => { this.signalNb.update( (valSignal) => valSignal + 1 ) } )
  ).subscribe()
  // -----
  fromEvent(this.eltMoins.nativeElement, 'click').pipe(
    tap( () => { this.signalNb.update( (valSignal) => valSignal - 1 ) } )
  ).subscribe()
}
```

# Les Formulaires



Copyright Michele Accioli - ORSYS

# Les formulaires Angular (avec Matériel Design)

- <https://material.angular.io/components/categories>
- <https://fonts.google.com/icons>

```
HTML TS CSS
<section>
  <div class="example-label">Basic</div>
  <div class="example-button-row">
    <button mat-button>Basic</button>
    <button mat-button color="primary">Primary</button>
    <button mat-button color="accent">Accent</button>
    <button mat-button color="warn">Warn</button>
    <button mat-button disabled>Disabled</button>
    <a mat-button href="https://www.google.com/" target="_blank">Link</a>
  </div>
```

```
HTML TS CSS
<mat-form-field>
  <mat-label>Input</mat-label>
  <input matInput>
</mat-form-field>
<mat-form-field>
  <mat-label>Select</mat-label>
  <mat-select>
    <mat-option value="one">First option</mat-option>
    <mat-option value="two">Second option</mat-option>
  </mat-select>
</mat-form-field>
<mat-form-field>
  <mat-label>Textarea</mat-label>
  <textarea matInput></textarea>
</mat-form-field>
```

Material Components CDK Guides

## Button

Autocomplete  
Badge  
Bottom Sheet  
**Button**  
Button toggle  
Card  
Checkbox  
Chips  
Core  
Datepicker  
Dialog  
Divider

OVERVIEW API EXAMPLES

Angular Material buttons are native `<button>` or `<a>` elements enhanced with Material Design style.

Basic buttons

Basic	Basic	Primary	Accent	Warn	Disabled	Link
Raised	Basic	Primary	Accent	Warn	Disabled	Link
Stroked	Basic	Primary	Accent	Warn	Disabled	Link
Flat	Basic	Primary	Accent	Warn	Disabled	Link
Icon	⋮	🏠	☰	❤	📄	

## Les formulaires Angular (avec Matériel Design)

`ng add @angular/material`

*ng add @angular/localize*

*ng add @angular/pwa*

Copyright Michel BOCCIOLESI - ORSYS

# Les formulaires Angular (avec Matériel Design)

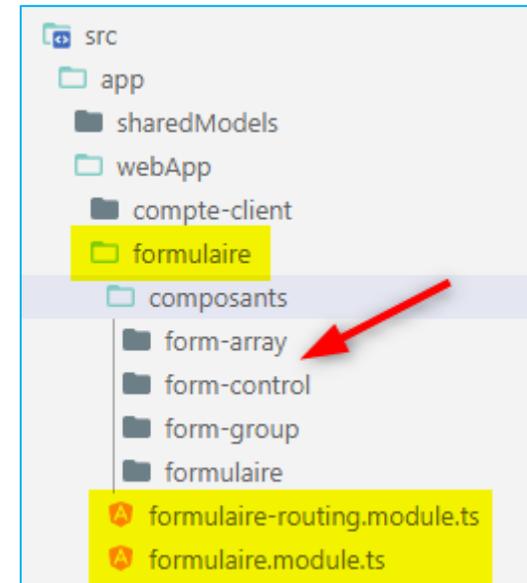
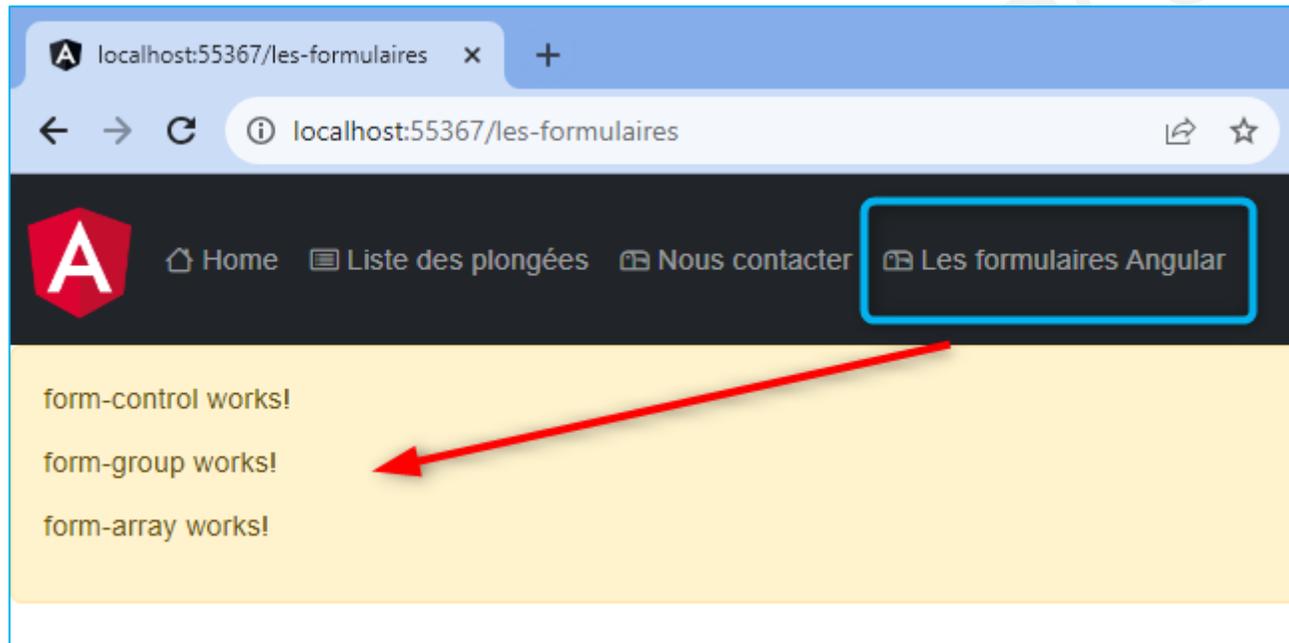
<https://dev.webjs.fr/1-Angular/ressources-formation/material.module.ts>

<https://dev.webjs.fr/1-Angular/ressources-formation/material-all.module.ts>

TP : mise en place d'une architecture de projet avec le module « formulaires » comme illustré ci-dessous.

Ressources en ligne (*corrigé en ligne*) :

<https://dev.webjs.fr/1-Angular/ressources-formation/reactive-forms.zip>





# Les formulaires Angular

- Mise en lumière des différences importantes entre le « Template Driven Form » et le « Reactive Form »
- Choix de l'utilisation
- Contexte d'utilisation – validation – contrôle de saisies de données – traitement des données saisies

Copyright Michel BOCCIOLESI - ORSYS



La class FormControl

 Votre prénom :\*

- Touched : false
- Pristine : true
- Dirty : false
- Valid : false

**Le champ prénom est requis !**

⚙ Le champ est requis !

[La Class form-control](#)

```
form-control.component.ts x
TP06-formulaires > src > app > webApp > formulaires > composants > form-control > form-control.c
1 import { Component } from '@angular/core';
2 import { FormControl, Validators } from '@angular/forms';
3
4 @Component({
5   selector: 'app-form-control',
6   templateUrl: './form-control.component.html',
7   styleUrls: ['./form-control.component.scss']
8 })
9 export class FormControlComponent {
10
11   // props
12   public prenom: FormControl<string | null>;
13
14   // const
15   constructor() {
16     this.prenom = new FormControl(
17       //val par défaut
18       '',
19       // Validators
20       [
21         Validators.required,
22         Validators.minLength(3),
23         Validators.maxLength(10)
24       ]
25     );
26   }
27
28 }
29

form-control.component.html x
TP06-formulaires > src > app > webApp > formulaires > composants > form-control > form-control.component.html >
1 <h5>La class formControl</h5>
2
3   <mat-icon>perm_identity</mat-icon>
4
5   <mat-form-field>
6     <mat-label>Votre prénom :</mat-label>
7     <input type="text" matInput [formControl]="prenom">
8   </mat-form-field>
9
10 <!-- tout ce qui suit peut être utilisé autant dans les "templates driven forms" d
11
12 <div>
13   <p>- Touched : {{prenom.touched}}</p>
14   <p>- Pristine : {{prenom.pristine}}</p>
15   <p>- Dirty : {{prenom.dirty}}</p>
16   <p>- Valid : {{prenom.valid}}</p>
17 </div>
18
19
20 <p [hidden]="!prenom.touched && !prenom.valid" style="color: lightgreen">
21   Le champ prénom est validé !
22 </p>
23
24 <p [hidden]="prenom.touched || prenom.valid" style="color: red">
25   Le champ prénom est requis !
26 </p>
27
28 <div *ngIf="prenom.touched && prenom.valid; then blockOk else blockNotOk"></div>
29
30   <ng-template #blockOk> Le champ est validé !</ng-template>
31   <ng-template #blockNotOk> Le champ est requis !</ng-template>
32
33 <hr>
34
```

[Home](#)
[Liste des plongées](#)
[Nous contacter](#)
[Les Formulaires](#)

## Les Forms groups

Votre prénom :\*  
 Zoé

- Prénom (value) : Zoé  
 - Prénom (Touched) : true  
 - Prénom (Pristine) : false  
 - Prénom (Dirty) : true  
 - Prénom (Valid) : true

Votre nom :

Votre email :\*  
 z.angular@tech.lead

Saisir votre rue :

Saisir votre ville :\*  
 New York

Saisir votre cp :

Allez à la suite ...  
 Une autre partie du formulaire à compléter ...

```

form-group.component.html
TP06-formulaires > src > app > webApp > formulaires > composants > form-group > form-group.component.html > div.a
1 <div class="alert alert-success">
2
3 <h2>Les Forms groups</h2>
4 <form [formGroup]="monForm">
5   <!-- <form [formGroup]="monForm" (submit)="onSubmit()" -->
6
7   <mat-icon id="icone-prenom">face</mat-icon>&nbsp;
8   <mat-form-field>
9     <mat-label>Votre prénom :</mat-label>
10    <input type="text" matInput formControlName="prenom">
11  </mat-form-field>
12  <hr>
13
14  <div class="alert alert-warning">
15    - Prénom (value) : {{monForm.controls['prenom'].value}}
16    <br> - Prénom (Touched) : {{monForm.controls['prenom'].touched}}
17    <br> - Prénom (Pristine) : {{monForm.controls['prenom'].pristine}}
18    <br> - Prénom (Dirty) : {{monForm.controls['prenom'].dirty}}
19    <br> - Prénom (Valid) : {{monForm.controls['prenom'].valid}}
20  </div>
21  <hr>
22
23  <p></p>
24  <mat-icon>emoji_people</mat-icon>&nbsp;
25  <mat-form-field>
26    <mat-label>Votre nom :</mat-label>
27    <input type="text" matInput formControlName="nom">
28  </mat-form-field>
29  <p></p>
30
31  <p></p>
32  <mat-icon>email</mat-icon>&nbsp;
33  <mat-form-field>
34    <mat-label>Votre email :</mat-label>
35    <input type="text" matInput formControlName="email">
36  </mat-form-field>
37  <p></p>
38
39  <div formGroupName="adresse" class="alert alert-danger">
40    <div>
41      <mat-icon>traffic</mat-icon>&nbsp;
42      <mat-form-field>
43        <mat-label>Saisir votre rue :</mat-label>
44        <input matInput type="text" formControlName="rue">
45      </mat-form-field>
46    <p></p>
  
```

La class form-group

# La Class FormGroup

```
export class FormGroupComponent {  
  public monForm: FormGroup;  
  constructor(  
    private _post: PostService,  
    private _fb: FormBuilder) {  
  
    this.monForm = this._fb.group({  
      // collection des controls  
      prenom: [  
        // val par défaut définie comme(as) une string ou null  
        // 'valeur_saisie' as string | null,  
        null as string | null,  
        [  
          Validators.required,  
          Validators.minLength(5)  
        ]  
      ],  
      // -----  
      nom: [  
        null as string | null,  
        Validators.required  
      ],  
      email: [null as string | null,  
        [  
          Validators.pattern('^[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,}$'),  
          Validators.required  
        ]  
      ],  
      adresse: this._fb.group({  
        rue: [null as string | null, Validators.required],  
        ville: [null as string | null, Validators.required],  
        cp: ['']  
      })  
    });  
  }  
}
```

```
// méthodes  
public onSubmit = () => {  
  console.log('group (form) principal : ', this.monForm.value);  
  console.log('sous-group (form) adresse : ', this.monForm.controls['adresse'].value);  
  this._post.postForm(this.monForm);  
}
```

TP :

1- récupérer le HTML : <https://dev.webjss.fr/1-Angular/ressources-formation/form-group.component.zip>

2- Mettre en place la validation TS

3- Créer le service « post.service.ts » qui poste les datas sur un json-server

"json-server": "json-server --watch src/assets/json/bdd.json -p 3001"

Base de données JSON : <https://dev.webjss.fr/1-Angular/ressources-formation/json.zip>

Corrigé en ligne : <https://dev.webjss.fr/1-Angular/ressources-formation/post.service.zip>

# TP : ajouter un sous-form-group (choix1, choix2, message, dateDebut

```
nxjs.component.ts  form-group.component.html  TP06-formulaires > src > app > webApp > formulaires > composants > form-group > form-group.component.html  TP06-formulaires > src > app > webApp > formulaires > composants > form-group > form-group.component.html  TP06-formulaires > src > app > webApp > formulaires > composants > form-group > form-group.component.html  TP06-formulaires > src > app > webApp > formulaires > formulaires.module.ts > ...
```

```
61     <mat-label>Saisir votre cp :</mat-label>
62     <input matInput type="text" FormControlName="cp">
63   </mat-form-field>
64   <p></p>
65 </div>
66 </div>
67
68
69 <div FormGroupName="tp" class="alert alert-danger">
70
71   <p></p>
72   <mat-label>Votre choix</mat-label>
73   <mat-icon>question_mark</mat-icon>&nbsp;
74   <mat-checkbox name="name_choix1" FormControlName="choix1"> Choix #1
75   <mat-checkbox name="name_choix2" FormControlName="choix2"> Choix #2
76   <p></p>
77
78   <mat-icon>settings</mat-icon>&nbsp;
79   <mat-form-field>
80     <mat-label>Votre message :</mat-label>
81     <textarea matInput FormControlName="message"></textarea>
82   </mat-form-field>
83
84   <p></p>
85   <mat-icon>calendar_today</mat-icon>&nbsp;
86   <mat-form-field appearance="fill">
87     <mat-label>Choisir une date</mat-label>
88     <input matInput [matDatepicker]="date" FormControlName="dateDebut"
89
90     <mat-datepicker-toggle matSuffix [for]="date"></mat-datepicker-t
91     <mat-datepicker #date></mat-datepicker>
92   </mat-form-field>
93   <p></p>
94 </div>
95
96
97 <div [hidden]="!monForm.controls['adresse'].valid">
98
99   Allez à la suite ...
100   <p>Une autre partie du formulaire à compléter ...</p>
101
```

```
48   ],
49   },
50   ),
51   // *****
52   adresse: new FormGroup({
53     rue: new FormControl<string | null>(null),
54     ville: new FormControl<string | null>(null, Validators.required),
55     cp: new FormControl<string | null>(null),
56   }),
57   // TP
58   // *****
59   tp: new FormGroup({
60     choix1: new FormControl<boolean | null>(true),
61     choix2: new FormControl<boolean | null>(false),
62     message: new FormControl<string | null>(null, Validators.required),
63     dateDebut: new FormControl<string | null>(null),
64   })
65 }
66
67 // cycle de vies
68 ngAfterViewInit() {
69
```

```
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { FormulairesComponent } from './composants/formulaires/formulaires.component';
4 import { FormControlComponent } from './composants/form-control/form-control.component';
5 import { FormGroupComponent } from './composants/form-group/form-group.component';
6 import { FormArrayComponent } from './composants/form-array/form-array.component';
7
8 // ---- Matériel Design -----
9 import { MatIconModule } from '@angular/material/icon';
10 import { MatInputModule } from '@angular/material/input';
11 import { MatButtonModule } from '@angular/material/button';
12 import { MatDatepickerModule } from '@angular/material/datepicker';
13 import { MatNativeDateModule } from '@angular/material/core';
14 import { MAT_DATE_LOCALE } from '@angular/material/core';
15 import { MatCheckboxModule } from '@angular/material/checkbox';
16 // ---- Matériel Design -----
```

# La Class formGroup

```
// + : 1 ou plusieurs fois
// * : 0 ou plusieurs fois
// {val min , val max } : pour répéter
Validators.pattern('^[a-z0-9._-]+@[a-z0-9.-]+\.[a-z]{2,}$'),
Validators.required
]
),
// *****
adresse: new FormGroup({
  rue: new FormControl<string | null>(null),
  ville: new FormControl<string | null>(null),
  cp: new FormControl<string | null>(null),
})
});

// méthodes
public onSubmit = () => {
  console.log('Group (form) principal : ', this.monForm.value);
  console.log('Sous-Group (form) adresse : ', this.monForm.controls['adresse'].value);
}
}
```

```
50 <p></p>
51 </div>
52 <div>
53 <mat-icon>tr
54 <mat-form-fie
55 <mat-labe
56 <input ma
57 </mat-form-fi
58
59 <p></p>
60 </div>
61 </div>
62
63
64 <p></p>
65
66 <button mat-raised-bu
67 <mat-icon>
68 </button>
69
70 <p></p>
71
72 </form>
73
```

1 Issue: 1

[webpack-dev-server] Server started: Hot Module Replacement disabled, Live Reloading enabled, Progress disabled, Overlay enabled. [index.js:485](#)

Angular is running in development mode. [core.mjs:25499](#)

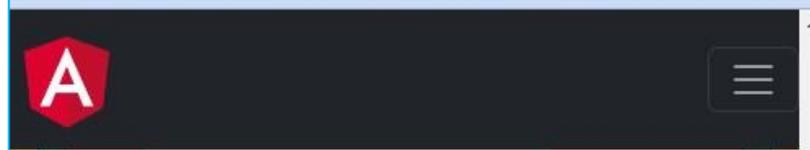
Group (form) principal : [form-group.component.ts:61](#)  
▼ {prenom: 'mbo', nom: 'gfg', email: 'mbo@free.fr', adresse: {...}}

- ▼ adresse:
  - cp: "3"
  - rue: "2"
  - ville: "2"
  - ▶ [[Prototype]]: Object
- ▶ [[Prototype]]: Object

Sous-Group (form) adresse : [form-group.component.ts:62](#)  
▼ {rue: '2', ville: '2', cp: '3'} 1

- cp: "3"
- rue: "2"
- ville: "2"
- ▶ [[Prototype]]: Object

Copyright



Allez à la suite ...  
Une autre partie du formulaire à compléter ...

submit ...

```
{ "prenom": "Michel", "nom": "B", "email": "mbo@free.fr",  
"adresse": { "rue": "des Lilas", "ville": "Nice", "cp": "06200" },  
"tp": { "choix1": true, "choix2": true, "message": "OK",  
"dateDebut": "2023-10-19T22:00:00.000Z" } }
```

```
----- OBJET JS : post.service.ts:16  
{prenom: 'Michel', nom: 'B', email: 'mbo@free.fr', adresse:  
  {...}, tp: {...}}  
  adresse: {rue: 'des Lilas', ville: 'Nice', cp: '06200'}  
    email: "mbo@free.fr"  
    nom: "B"  
    prenom: "Michel"  
    tp: {choix1: true, choix2: true, message: 'OK', dateDebut: Fr  
    [[Prototype]]: Object  
----- OBJET STRINGIFY : post.service.ts:23  
{ "prenom": "Michel", "nom": "B", "email": "mbo@free.fr", "adresse":  
  { "rue": "des Lilas", "ville": "Nice", "cp": "06200" }, "tp":  
  { "choix1": true, "choix2": true, "message": "OK", "dateDebut": "2023-10-  
  19T22:00:00.000Z" } }
```

```
[  
  {  
    "nameNom": "Bruce Wayne",  
    "nameEmail": "bruce@gothamcity.com",  
    "nameMessage": "Alfred je reviens ...",  
    "id": 1  
  },  
  {  
    "nameNom": "Ahsoka",  
    "nameEmail": "a.tano@hyperspace.com",  
    "nameMessage": "Sabine tu as été ma padawan",  
    "id": 1  
  },  
  {  
    "prenom": "Michel",  
    "nom": "B",  
    "email": "mbo@free.fr",  
    "adresse": {  
      "rue": "des Lilas",  
      "ville": "Nice",  
      "cp": "06200"  
    },  
    "tp": {  
      "choix1": true,  
      "choix2": true,  
      "message": "OK",  
      "dateDebut": "2023-10-19T22:00:00.000Z"  
    },  
    "id": 2  
  }  
]
```

Copy

```
form-group.component.ts x
src > app > webApp > formulaires > composants > form-group > form-group.component.ts > FormGroupComponent > constructor
14 public monForm: FormGroup,
15 // constructeur
16 constructor(
17     private _post: PostService
18 ) {
19     this.monForm = new FormGroup({
20         // déclaration de tous les controls de ce formulaire (formGroup)
21     });
22 }
```

```
form-group.component.ts x
src > app > webApp > formulaires > composants > form-group > form-group.component.ts > FormGroupComponent > onSubmit
98 }
99 }
100 // Méthodes
101 public onSubmit = () => {
102     console.log('Groupe principal : ', this.monForm.value);
103     console.log('Sous Groupe : ', this.monForm.controls['adresse'].value);
104     console.log('Sous Groupe : ', this.monForm.controls['tp'].value);
105 }
106 this._post.postForm(this.monForm);
107 // TP
108 }
109 }
110 }
111 }
```

```
bdd.json x
TP06-formulaires > src > assets > json > bdd.json > ...
76 "photo": "https://dev.webjts.fr/images/fond5.jpg"
77 }
78 ],
79 "messages": [
80 {
81     "nameNom": "Bruce Wayne",
82     "nameEmail": "bruce@gothamcity.com",
83     "nameMessage": "Alfred je reviens ...",
84     "id": 1
85 }
```

```
post.service.ts x
TP06-formulaires > src > app > sharedModels > services > post.service.ts > PostService > post
1 import { HttpClient, HttpHeaders } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { FormGroup } from '@angular/forms';
4
5 @Injectable({
6     providedIn: 'root'
7 })
8
9 export class PostService {
10     constructor(private _http: HttpClient) {
11     }
12 // méthodes
13 public postForm = (form: FormGroup) => {
14     console.log('----- OBJET JS : ', form.value);
15
16     // --- Post sur un serveur json
17     // 1-url
18     const url: string = 'http://localhost:3001/messages';
19     // 2-body
20     const body = JSON.stringify(form.value);
21     console.warn('----- OBJET STRINGIFY : ', body);
22
23     // 3-headers
24     const myHeaders = new HttpHeaders({
25         'Content-Type': 'application/json',
26         'Access-Control-Allow-Origin': '*' // CORS
27     });
28     const options = { headers: myHeaders };
29     //-----
30
31     // envoi avec POST
32     this._http.post(url, body, options).subscribe(
33         (res: any) => console.log(res);
34     );
35 }
36 }
37 }
38 }
```

# La Class formArray

Récupérer les ressources en ligne :

<https://dev.webjs.fr/1-Angular/ressources-formation/reactive-forms.zip>

Form Array - Form Builder

Donner un nom au cursus de formation  
Web

Ajouter une formation

» +

N° : 0  
Formation\* Angular Niveau Perfectionnement

N° : 1  
Formation\* React Niveau Expert

N° : 2  
Formation\* Niveau

N° : 3  
Formation\* Niveau

- ORSYS

Copyright

# La Class formArray

Récupérer les ressources en ligne :

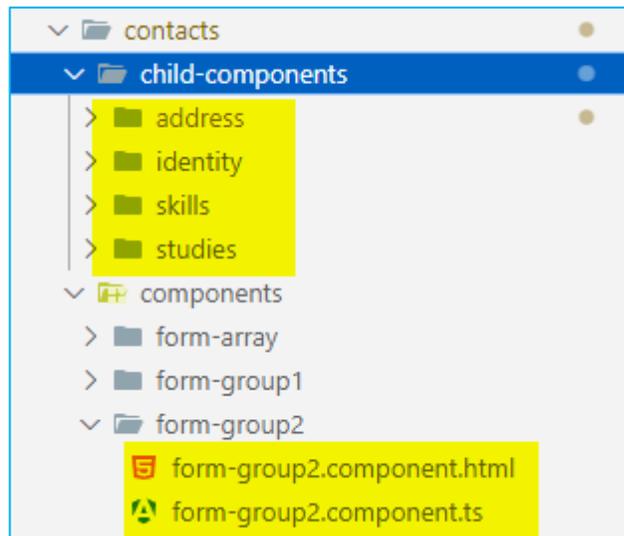
<https://dev.webjs.fr/1-Angular/ressources-formation/reactive-forms.zip>

```
export class FormArrayComponent {  
  
  public cursus: FormGroup;  
  // -----  
  constructor(private _fb: FormBuilder) {  
  
    this.cursus = this._fb.group({  
      nomCursus: '',  
      // définition du 2nd champ qui va être un tableau(ensemble) de formulaire  
      formationsArray: this._fb.array([])  
    });  
  }  
  // un getter qui va nous retourner le formationsArray  
  get getFormations(): FormArray {  
    return this.cursus.get('formationsArray') as FormArray;  
  }  
  // ----- Méthodes -----  
  public addFormation = () => {  
    this.getFormations.push(this.newFormation());  
  }  
  // méthode qui crée un nouveau formgroup  
  // avec la formation et le niveau  
  
  private newFormation = (): FormGroup<any> => {  
    return this._fb.group({ // définition du formGroup  
      formation: [null as string | null, Validators.required],  
      niveau: ''  
    });  
  }  
  // -----  
  public delFormation = (i: number) => {  
    this.getFormations.removeAt(i);  
  }  
}
```

```
<h3>Form Array - Form Builder</h3>  
<form [formGroup]="cursus">  
  <p>  
    <mat-form-field>  
      <mat-label>Donner un nom au cursus de formation :</mat-label>  
      <input matInput type="text" FormControlName="nomCursus">  
    </mat-form-field>  
  </p>  
  <p>Ajouter une formation </p>  
  <p>  
    <mat-icon>double_arrow</mat-icon>&nbsp;<br>  
    <button mat-mini-fab type="button" (click)="addFormation()">  
      <mat-icon>add</mat-icon>  
    </button>  
  </p>  
  
  <div formArrayName="formationsArray">  
    <div *ngFor="let formation of getFormations.controls; let i=index">  
      N° : {{i}}  
      <div [formGroupName]="i">  
  
        <mat-form-field>  
          <mat-label>Formation</mat-label>  
          <input matInput type=" text" FormControlName="formation">  
        </mat-form-field>&nbsp;<br>  
  
        <mat-form-field>  
          <mat-label>Niveau</mat-label>  
          <mat-select FormControlName="niveau">  
            <mat-option value="niveau1">Niveau 1</mat-option>  
            <mat-option value="niveau2">Perfectionnement</mat-option>  
            <mat-option value="niveau3">Expert</mat-option>  
          </mat-select>  
        </mat-form-field>  
  
        <mat-icon (click)="delFormation(i)">delete</mat-icon>  
      </div>  
    </div>  
  </div>  
</form>
```

## TP

Améliorer le form-group réalisé en 1<sup>ère</sup> partie de formation. Le but étant d'avoir un formulaire principal qui appelle des child-components.

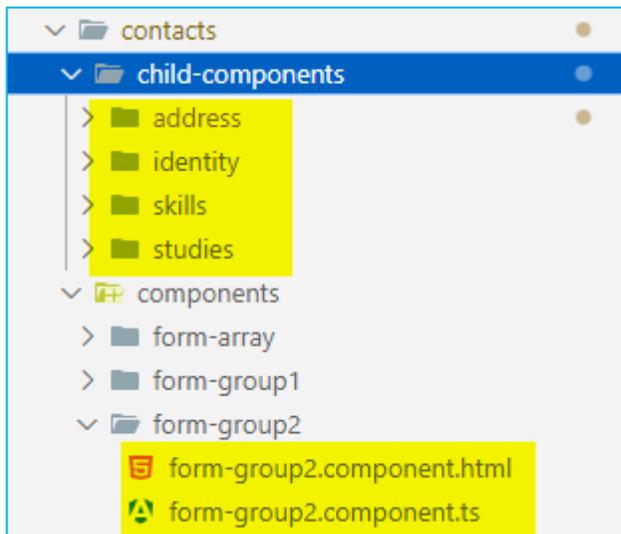


Le composant principal passera en Input [parentFormGroup] le nom du FormGroup

```
form-group2.component.html x
src > app > webApp > contacts > components > form-group2 > form-group2.component.html
1 <div class="container">
2
3
4
5
6
7 <hr>
8 <form [formGroup]="monForm" (submit)="onSubmit()">
9
10   <div class="alert alert-light">
11     <app-identity [parentFormGroup]="monForm" #identity />
12   </div>
13
14   <!-- ----- -->
15
16   <div [hidden]="!monForm.controls['identity'].valid">
17     <div class="alert alert-light">
18       <app-address [parentFormGroup]="monForm" #address />
19     </div>
20   </div>
21
22   <!-- ----- -->
23
24   <div [hidden]="!monForm.controls['address'].valid">
25     <div class="alert alert-light">
26       <app-studies [parentFormGroup]="monForm" #studies />
27     </div>
28   </div>
29
30   <!-- ----- -->
31
32   <div [hidden]="!monForm.controls['studies'].valid">
33     <div class="alert alert-light">
34       <app-skills [parentFormGroup]="monForm" #skills />
35     </div>
36   </div>
37
38   <!-- ----- -->
```

## TP

Améliorer le form-group réalisé en 1<sup>ère</sup> partie de formation. Le but étant d'avoir un formulaire principal qui appelle des child-components.



```
// 1-props
public monForm: FormGroup;
@ViewChild('identity') eltIdentity!:any;
@ViewChild('adress') eltAddress!:any;

// 2-const
constructor(private _fb:FormBuilder, _service: PostService) {
  this.monForm = this._fb.group({
  })
}
public onSubmit = () => {
  console.log(this.monForm.value);
  console.log(this.monForm.controls['identity'].value);
  console.log(this.monForm.controls['address'].value);
  // this._service.postForm(this.monForm);
}
}
```

Le composant principal passera en Input [parentFormGroup] le nom du FormGroup

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.  
Phase 1 : Compilation et publication sur NODE.JS

The screenshot displays the Visual Studio Code interface. On the left, the Explorer pane shows a project structure with a folder named 'address-form' containing a 'lib' subfolder. The file 'address-form.component.ts' is selected. The main editor shows the code for this component, which imports '@angular/core' and defines an '@Component' with a selector 'lib-address-form', standalone: true, and a template containing 'address-form works!'. The component is exported as 'AddressFormComponent'.

At the bottom, the Terminal pane shows the output of the command 'ng generate library address-form'. The output lists the creation of various files and the successful installation of packages.

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'lib-address-form',
5   standalone: true,
6   imports: [],
7   template: `
8     <p>
9       address-form works!
10    </p>
11  `,
12   styles: ``
13 })
14 export class AddressFormComponent {
15 }
16 }
17
```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE **TERMINAL** PORTS

- PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024> ng generate library address-form
- CREATE projects/address-form/ng-package.json (167 bytes)
- CREATE projects/address-form/package.json (228 bytes)
- CREATE projects/address-form/README.md (1057 bytes)
- CREATE projects/address-form/tsconfig.lib.json (490 bytes)
- CREATE projects/address-form/tsconfig.lib.prod.json (412 bytes)
- CREATE projects/address-form/tsconfig.spec.json (449 bytes)
- CREATE projects/address-form/src/public-api.ts (142 bytes)
- CREATE projects/address-form/src/lib/address-form.component.spec.ts (651 bytes)
- CREATE projects/address-form/src/lib/address-form.component.ts (257 bytes)
- CREATE projects/address-form/src/lib/address-form.service.spec.ts (399 bytes)
- CREATE projects/address-form/src/lib/address-form.service.ts (149 bytes)
- UPDATE angular.json (6182 bytes)
- UPDATE tsconfig.json (1043 bytes)
- ✓ Packages installed successfully.
- PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024> □

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.

The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORATEUR (Left):** Shows a file tree for a project named 'TP-PROJET-STANDALONE-ANY-PARTICIPANTS-09-2024'. The 'dist' folder is expanded to show 'address-form', which contains 'esm2022', 'fesm2022', 'lib', and 'package.json'. A red arrow points from the 'package.json' file in the tree to the code editor.
- Code Editor (Center):** Displays the content of 'package.json' for the 'address-form' package. The 'name' field is set to '@mbo06255/address-form' and the 'version' is '0.0.1'. A red arrow points to the version field with the handwritten note 'Renommer avec le nom du compte !'. The 'dependencies' section includes 'tslib' and 'fesm2022/address-form.mjs'.
- TERMINAL (Bottom):** Shows the command 'ng build address-form' being executed. The output includes:
  - Building entry point 'address-form'
  - Build status: ✓ Compiling with Angular sources in Ivy partial compilation mode. ✓ Generating FESM bundles. ✓ Copying assets. ✓ Writing package manifest. ✓ Built address-form
  - Build path: - from: C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\projects\address-form - to: C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\dist\address-form
  - Build at: 2024-10-04T06:47:26.920Z - Time: 2573ms

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.

The image shows a VS Code editor window on the left and a browser window on the right. The VS Code editor displays the `package.json` file for the `address-form` package. The terminal below shows the command `npm publish --access=public` being executed, followed by a series of `npm notice` messages detailing the package's metadata and the successful upload to the npm registry. A red arrow points from the terminal output to the browser window, which displays the `Authentication Successful` message on the `npmjs.com` website.

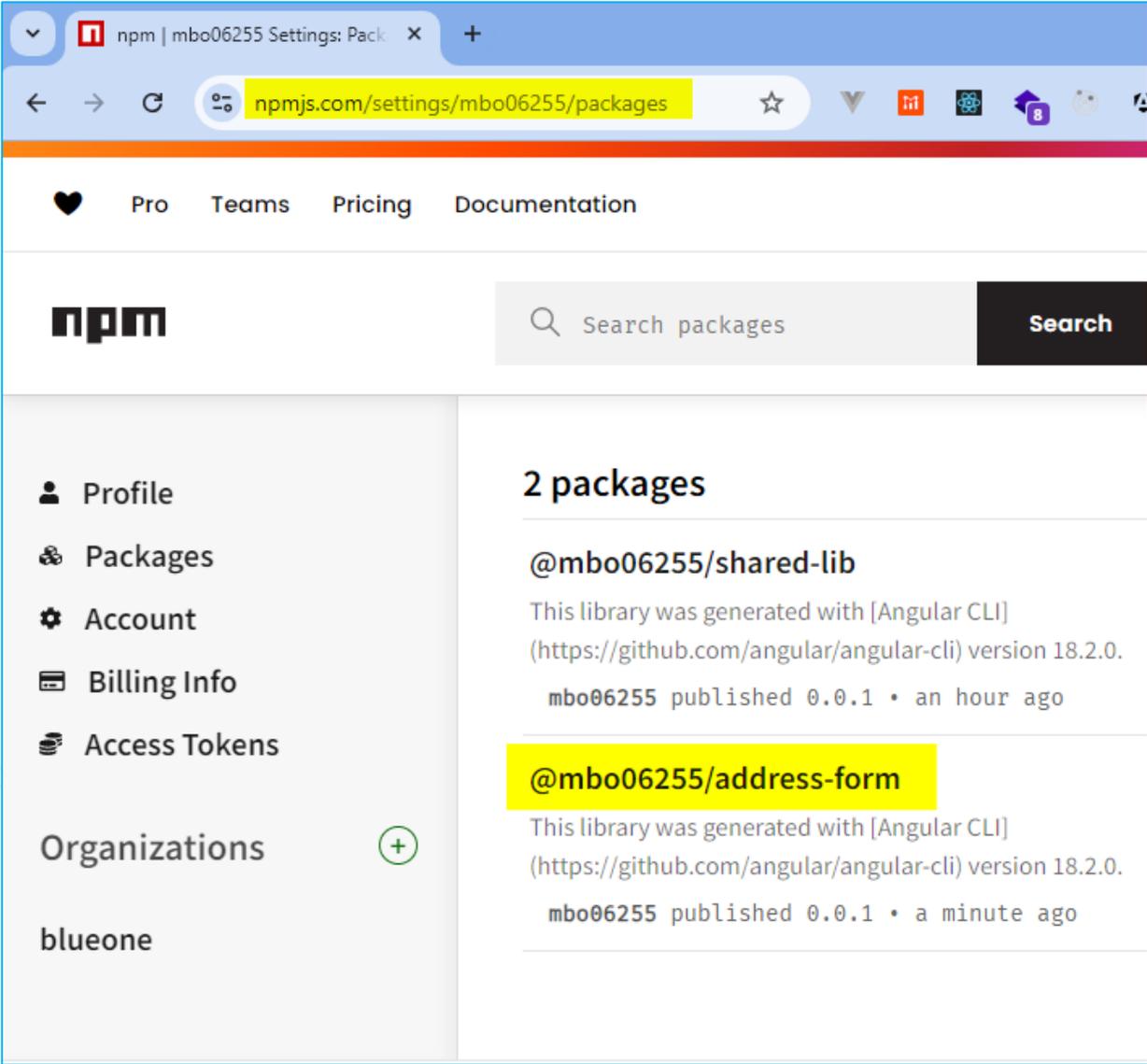
```
dist > address-form > package.json > name
1  {
2    "name": "@mbo06255/address-form",
3    "version": "0.0.1",
4    "peerDependencies": {
5      "@angular/common": "^18.2.0",
6      "@angular/core": "^18.2.0"
7    },
8    "dependencies": {
9      "tslib": "^2.3.0"
10   },
11   "sideEffects": false,
12   "module": "fesm2022/address-form.mjs",
13   "typings": "index.d.ts",
14   "exports": {
15     "./package.json": {
16       "default": "./package.json"
17     },
18   },
19 }
```

```
PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\dist\address-form> npm publish --access=public
npm notice
npm notice   @mbo06255/address-form@0.0.1
npm notice Tarball Contents
npm notice 1.1kB README.md
npm notice 504B esm2022/address-form.mjs
npm notice 11.3kB esm2022/lib/address-form.component.mjs
npm notice 545B esm2022/public-api.mjs
npm notice 6.2kB fesm2022/address-form.mjs
npm notice 5.4kB fesm2022/address-form.mjs.map
npm notice 117B index.d.ts
npm notice 527B lib/address-form.component.d.ts
npm notice 563B package.json
npm notice 46B public-api.d.ts
npm notice Tarball Details
npm notice name: @mbo06255/address-form
npm notice version: 0.0.1
npm notice filename: mbo06255-address-form-0.0.1.tgz
npm notice package size: 6.7 kB
npm notice unpacked size: 26.3 kB
npm notice shasum: f59b4412181b60d5ac0513fadceab174f420e9ce
npm notice integrity: sha512-2jvenx2maHdgX[... ]I9RV3qJehNj+A==
npm notice total files: 10
npm notice
npm notice Publishing to https://registry.npmjs.com with tag latest and public access
Authenticate your account at:
https://www.npmjs.com/auth/cli/180f655e-0f66-459c-9987-d79c9e807c61
Press ENTER to open in the browser...
+ @mbo06255/address-form@0.0.1
PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\dist\address-form >
```

The browser window shows the `npmjs.com` website with the `Authentication Successful` message. The message includes a lock icon and the text: "You can close this tab and return to your command prompt". A red arrow points from the terminal output to this message.

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.

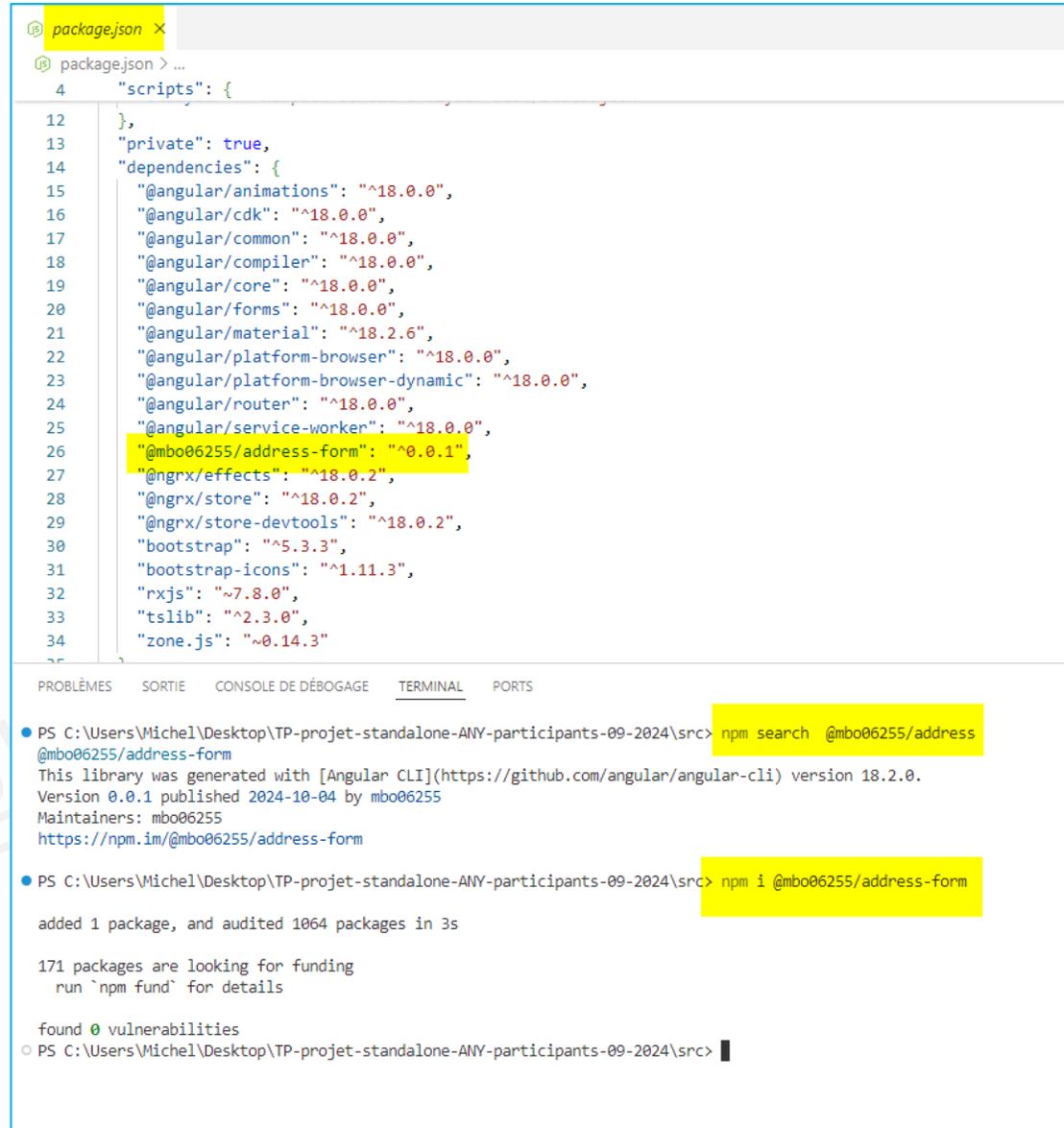


The screenshot shows the npmjs.com settings page for user mbo06255. The browser address bar is `npmjs.com/settings/mbo06255/packages`. The page features a navigation menu with links for Pro, Teams, Pricing, and Documentation. Below the navigation is the npm logo and a search bar for packages. A sidebar on the left contains links for Profile, Packages, Account, Billing Info, Access Tokens, Organizations, and blueone. The main content area displays a list of 2 packages:

- @mbo06255/shared-lib**  
This library was generated with [Angular CLI] (https://github.com/angular/angular-cli) version 18.2.0.  
mbo06255 published 0.0.1 • an hour ago
- @mbo06255/address-form** (highlighted in yellow)  
This library was generated with [Angular CLI] (https://github.com/angular/angular-cli) version 18.2.0.  
mbo06255 published 0.0.1 • a minute ago

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.  
Phase 2 : Déploiement depuis NODE.JS



The image shows a code editor window with a file named `package.json`. The content of the file is as follows:

```
4   "scripts": {  
12 },  
13 "private": true,  
14 "dependencies": {  
15   "@angular/animations": "^18.0.0",  
16   "@angular/cdk": "^18.0.0",  
17   "@angular/common": "^18.0.0",  
18   "@angular/compiler": "^18.0.0",  
19   "@angular/core": "^18.0.0",  
20   "@angular/forms": "^18.0.0",  
21   "@angular/material": "^18.2.6",  
22   "@angular/platform-browser": "^18.0.0",  
23   "@angular/platform-browser-dynamic": "^18.0.0",  
24   "@angular/router": "^18.0.0",  
25   "@angular/service-worker": "^18.0.0",  
26   "@mbo06255/address-form": "^0.0.1",  
27   "@ngrx/effects": "^18.0.2",  
28   "@ngrx/store": "^18.0.2",  
29   "@ngrx/store-devtools": "^18.0.2",  
30   "bootstrap": "^5.3.3",  
31   "bootstrap-icons": "^1.11.3",  
32   "rxjs": "~7.8.0",  
33   "tslib": "^2.3.0",  
34   "zone.js": "~0.14.3"
```

Below the code editor is a terminal window with the following output:

```
PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\src> npm search @mbo06255/address-form  
@mbo06255/address-form  
This library was generated with [Angular CLI](https://github.com/angular/angular-cli) version 18.2.0.  
Version 0.0.1 published 2024-10-04 by mbo06255  
Maintainers: mbo06255  
https://npm.im/@mbo06255/address-form  
  
PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\src> npm i @mbo06255/address-form  
  
added 1 package, and audited 1064 packages in 3s  
  
171 packages are looking for funding  
run `npm fund` for details  
  
found 0 vulnerabilities  
PS C:\Users\Michel\Desktop\TP-projet-standalone-ANY-participants-09-2024\src>
```

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.  
Phase 2 : Déploiement depuis NODE.JS

The screenshot displays a development environment with three main panels:

- Left Panel (home.component.ts):** Shows the TypeScript code for the HomeComponent. Lines 9-10 and 20-21 are highlighted in yellow, showing imports for SharedLibComponent, SharedLibService, and the AddressComponent.
- Middle Panel (home.component.html):** Shows the HTML template. Lines 36-39 are highlighted in yellow, showing the inclusion of the app-address component within a warning alert.
- Right Panel (Browser):** Shows the application running on localhost:3000. It features a navigation bar, a 'Child Home Component' section with a form for 'Choisir le coefficient de réduction', and a 'Composant redistribuable' section with an 'Adresse' form containing three input fields: 'Saisir votre rue', 'Saisir votre ville' (marked as required), and 'Saisir votre cp'. A red arrow points from the highlighted code in the middle panel to the 'Saisir votre ville' input field in the browser.

At the bottom, a terminal window shows the output of the application bundle generation, listing files like chunk-M3NT5YE4.js and their sizes, followed by the message: 'Application bundle generation complete. [3.951 seconds]'.

## TP2

Compiler le form-group « address » afin de le réexploiter facilement dans ce projet ou d'autres projets.  
Phase 2 : Déploiement depuis NODE.JS

```
form> npm config set registry https://registry.npmjs.com
```

Copyright Michel B... - ORSYS

# Les Tests



Copyright Michel BACCIOLESI - ORSYS

# Le Test

Le test (ou le Test Driven Development) permet de :

- Documenter le projet au fur et à mesure de sa création
- Faciliter les migrations entre versions majeures
- Simplifier la recherche d'erreurs
- Isoler l'objet de son contexte et de tester son intégration (Mock)

The image shows a screenshot of the Visual Studio Code interface. On the left is the Explorer sidebar showing a project structure under 'WEBAPP'. The 'test-unitaires' folder is expanded, and 'documentation.md' is highlighted. The central editor shows the content of 'documentation.md' with the following text:

```
1 # Savoir documenter son projet
2
3 ## 1-Les tests Unitaires
4
5 ### Composants :
6 ### Pipes :
7 ### Directives :
8
9 ## 2-Les tests D'intégrations
10
11 ### Services :
12
13 ## 3-Les tests End To End
14
15 ### Mise en Prod :
```

On the right, the 'Prévisualiser documentation.md' window shows a rendered version of the markdown file with the following structure:

- Savoir documenter son projet
- 1-Les tests Unitaires
- Composants :
- Pipes :
- Directives :
- 2-Les tests D'intégrations
- Services :
- 3-Les tests End To End
- Mise en Prod :

Two red arrows point from the 'documentation.md' file in the Explorer to the code editor and the preview window, respectively.

# Documenter son projet



The image shows a side-by-side comparison of a markdown file and its rendered PDF version. The left pane shows the source code in a code editor, and the right pane shows the rendered PDF document.

**COMMANDES.md (Source Code):**

```
1 Pour générer le fichier lisible : CTRL + SHIFT + v
2 ou bouton droit
3
4 # Commandes Angular NG
5
6 # New projet
7
8 ng new nomDuProjet
9
10 ## options :
11
12 # Generate objets
13
14 ## composant
15 - ng g(enerate) c(omposant) nomDuComposant
16 - options :
17 --export (exporte le composant depuis le module)
18 --skip-tests : omet la création du fichier de spec.ts (tests
unitaires)
19 --skip-import : ne déclare pas le composant dans un module
20 --flat : crée le composant sans dossier
21
22 ## module
23 - ng g(enerate) m(odule) nomDuModule
24 - options :
25 -- routing : crée le fichier de routing forChild pour le module
concerné
26
27 ## service
```

**COMMANDES.pdf (Rendered Output):**

Pour générer le fichier lisible : CTRL + SHIFT + v ou bouton droit

## Commandes Angular NG

---

### New projet

---

ng new nomDuProjet

options :

### Generate objets

---

composant

- ng g(enerate) c(omposant) nomDuComposant
- options : --export (exporte le composant depuis le module) --skip-tests : spec.ts (tests unitaires) --skip-import : ne déclare pas le composant dans un composant sans dossier

# Documenter son projet



The screenshot shows the Visual Studio Code interface with the Markdown PDF extension installed. The left pane displays the content of the 'COMMANDES.md' file, which includes the text 'Commandes Angular NG' and 'New projet'. A yellow highlight is placed over the text 'CTRL + SHIFT + v ou bouton droit'. The right pane shows the extension's details, including its icon, name 'Markdown PDF', version 'v1.5.0', author 'yzane', download count '1747619', and a 5-star rating. The extension is currently active globally, as indicated by the text 'Cette extension est activée globalement.' and the 'Désactiver' button.

COMMANDES.md | Prévisualiser COMMANDES.md | ... | COMMANDES.pdf | Extension : Markdown PDF

Pour générer le fichier lisible : CTRL + SHIFT + v ou bouton droit

## Commandes Angular NG

---

## New projet

---

### Markdown PDF v1.5.0

yzane | 1747619 | ★★★★★ (105)

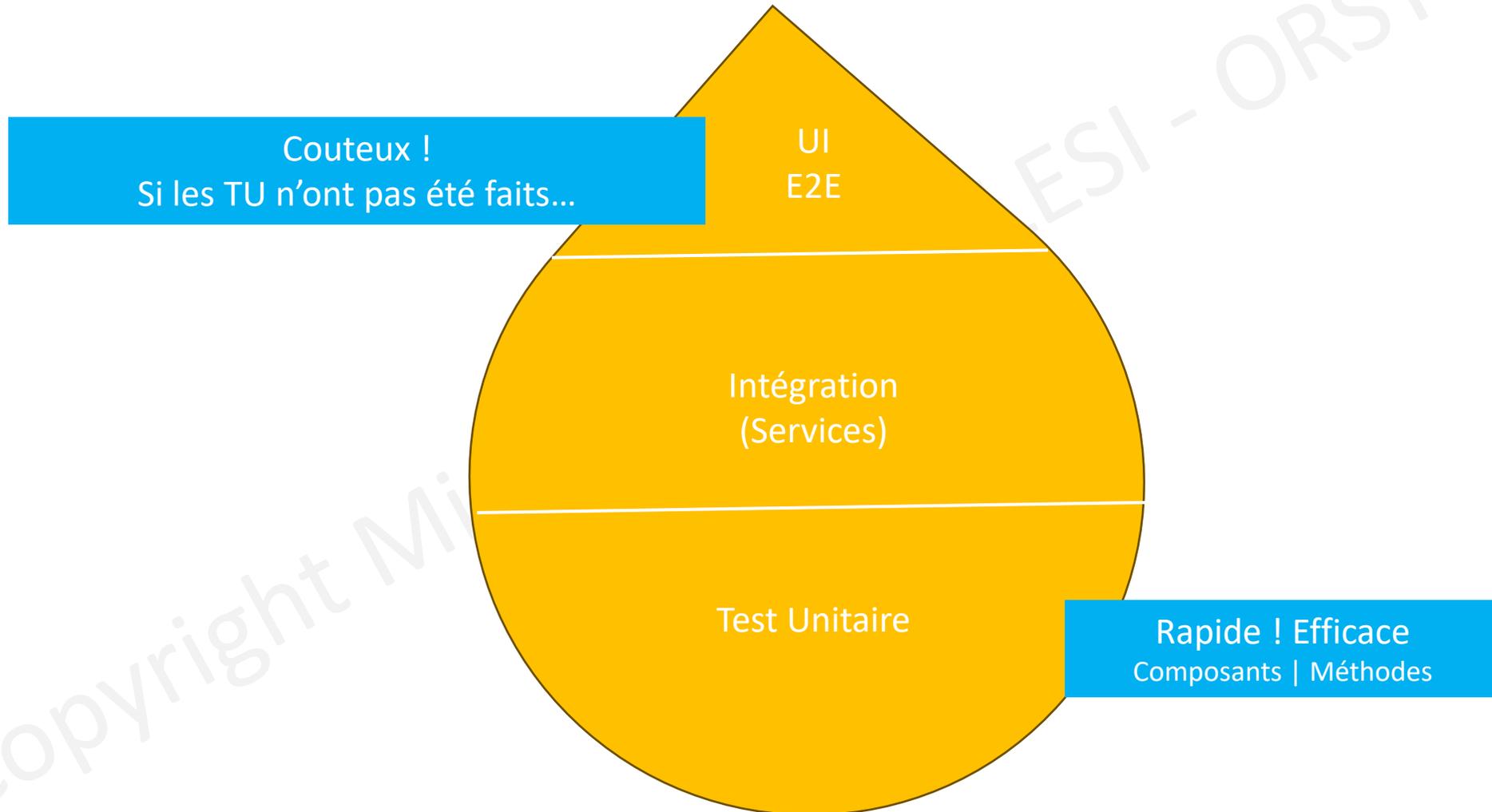
Convert Markdown to PDF

Désactiver | Désinstaller | ⚙️

Cette extension est activée globalement.

Copyright Miché

# Le Test et la Documentation



# 1<sup>er</sup> exemple : Apprendre de l'existant

```
app.component.ts x
TP07-tests-unitaires > src > app > app.component.ts > AppComponent > titl
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-root',
5   templateUrl: './app.component.html',
6   styleUrls: ['./app.component.css']
7 })
8 export class AppComponent {
9   title = 'TU';
10 }
11

app.component.html x
TP07-tests-unitaires > src > app > app.component.html > h1
1 <h1>{{title}}</h1>

app.component.spec.ts x
TP07-tests-unitaires > src > app > app.component.spec.ts > describe('AppComponent') callback
1 import { TestBed } from '@angular/core/testing';
2 import { AppComponent } from './app.component';
3
4 // -----
5 // describe permet de créer un Objet de Test Unitaire
6 // -----
7 describe('AppComponent', () => {
8   beforeEach(() => TestBed.configureTestingModule({
9     declarations: [AppComponent],
10  }));
11
12 // it est une fonction qui crée une spécification
13 it('should create the app', () => {
14   const fixture = TestBed.createComponent(AppComponent);
15   const app = fixture.componentInstance;
16   // une spéc (it) attend un retour (expectation)
17   expect(app).toBeTruthy(); // matchers Jasmine
18 });
19
20
21 it(`should have as title 'TP07-tests-unitaires'`, () => {
22   const fixture = TestBed.createComponent(AppComponent);
23   const composant = fixture.componentInstance;
24   expect(composant.title).toEqual('TU');
25 });
26
27
28 it('should render title', () => {
29   const fixture = TestBed.createComponent(AppComponent);
30   fixture.detectChanges();
31   const compiled = fixture.nativeElement as HTMLElement;
32   expect(compiled.querySelector('h1')?.textContent).toContain('TU');
33 });
34 });
35
```

# 2<sup>nd</sup> exemple : Injection de services

```
warn.service.ts × TP07-tests-unitaires > src > app > services > warn.service.ts > WarnService
1 import { Injectable } from '@angular/core';
2
3 @Injectable({
4   providedIn: 'root'
5 })
6 export class WarnService {
7   // -----
8   // méthodes
9   // -----
10  public warnMessage = (msg: string) => {
11    console.warn(`Le message est : ${msg}`);
12  }
13 }
```

```
operations.service.ts × TP07-tests-unitaires > src > app > services > operations.service.ts > OperationsService > ajouter
1 import { Injectable } from '@angular/core';
2 import { WarnService } from './warn.service';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class OperationsService {
8
9   constructor(
10    private _warn: WarnService
11  ) { }
12
13   // -----
14   // méthodes
15   // -----
16  public ajouter = (nb1: number, nb2: number) => {
17    this._warn.warnMessage('Ajout OK');
18    // this._warn.warnMessage('Ajout OK');
19    return nb1 + nb2;
20  }
21
22  public soustraire = (nb1: number, nb2: number) => {
23    this._warn.warnMessage('Soustraction OK');
24    return nb1 - nb2;
25  }
26 }
```

```
operations.service.spec.ts × TP07-tests-unitaires > src > app > services > operations.service.spec.ts > describe('Test Injection de dépendances') ca
1 import { OperationsService } from './operations.service';
2 import { WarnService } from './warn.service';
3
4 // 1- Création de l'objet de Test
5 describe('Test Injection de dépendances', () => {
6
7   // 2- liste des spécifications
8
9   // it (xit - fit)
10  it('Ajouter 2 nombres OK ?', () => {
11
12    const operations = new OperationsService(new WarnService);
13    const resultat = operations.ajouter(10, 5);
14    // 3- expectation => ce que l'on attend
15    expect(resultat).withContext('--- Erreur Expectation ---').toBe(15);
16  });
17
18  // 4- duplication du premier IT
19
20  // it (xit - fit)
21  it('Soustraire 2 nombres OK ?', () => {
22
23    const operations = new OperationsService(new WarnService);
24    const resultat = operations.soustraire(10, 5);
25    // 3- expectation => ce que l'on attend
26    expect(resultat).withContext('--- Erreur Expectation ---').toBe(5);
27  });
28
29
30
31
32
33 })
```

# 2<sup>nd</sup> exemple : Injection de services

```
warn.service.ts x
TP07-tests-unitaires > src > app > services > warn.service.ts > WarnService > logErreur
1 import { Injectable } from '@angular/core';
2
3 @Injectable({
4   providedIn: 'root'
5 })
6 export class WarnService {
7   // -----
8   // méthodes
9   // -----
10  public warnMessage = (msg: string) => {
11    console.warn(`Le message est : ${msg}`);
12  }
13  // ---
14  public logErreur = (msg: string) => {
15    console.error(`L'erreur est : ${msg}`);
16  }
17 }

operations.service.spec.ts x
TP07-tests-unitaires > src > app > services > operations.service.spec.ts > ...
1 import { OperationsService } from './operations.service';
2 import { WarnService } from './warn.service';
3
4 // 1- Création de l'objet de Test
5 describe('Test Injection de dépendances', () => {
6
7   // 2- liste des spécifications
8
9   // it (xit - fit)
10  it('Ajouter 2 nombres OK ?', () => {
11
12    // 5- Mocker WarnService (Jasmine Spies)
13    const MockWarn = jasmine.createSpyObj('WarnService', ['warnMessage', 'logErreur']);
14    const operations = new OperationsService(MockWarn);
15    const resultat = operations.ajouter(10, 5);
16
17    // 3- expectation => ce que l'on attend
18    expect(resultat).withContext('--- Erreur Expectation ---').toBe(15);
19  });
20
21 // 4- duplication du premier IT
22
23 // it (xit - fit)
24 it('Soustraire 2 nombres OK ?', () => {
25
26    // 6- reproduire aussi ici le MockWarn
27    const MockWarn = jasmine.createSpyObj('WarnService', ['warnMessage', 'logErreur']);
28    const operations = new OperationsService(MockWarn);
29    const resultat = operations.soustraire(10, 5);
30    // 3- expectation => ce que l'on attend
31    expect(resultat).withContext('--- Erreur Expectation ---').toBe(5);
32  });
33
34 });
35
36
37 }

operations.service.ts x
src > app > services > operations.service.ts > OperationsService > ajouter
1 import { Injectable } from '@angular/core';
2 import { WarnService } from './warn.service';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class OperationsService {
8
9   constructor(
10    private _warn: WarnService
11  ) { }
12
13  // -----
14  // méthodes
15  // -----
16  public ajouter = (nb1: number, nb2: number) => {
17    this._warn.warnMessage('Ajout OK');
18    // this._warn.warnMessage('Ajout OK');
19    return nb1 + nb2;
20  }
21
22  public soustraire = (nb1: number, nb2: number) => {
```

# 2<sup>nd</sup> exemple : Injection de services

```
warn.service.ts
1 import { Injectable } from '@angular/core';
2

operations.service.spec.ts
1 import { OperationsService } from './operations.service';
2 import { WarnService } from './warn.service';
3
4 // 1- Création de l'objet de Test
5 describe('Test Injection de dépendances', () => {
6
7   // 2- liste des spécifications
8
9   // it (xit - fit)
10  it('Ajouter 2 nombres OK ?', () => {
11
12    // 5- Mocker WarnService (Jasmine Spies)
13    const MockWarn = jasmine.createSpyObj('WarnService', ['warnMessage', 'logErreur']);
14    const operations = new OperationsService(MockWarn);
15    const resultat = operations.ajouter(10, 5);
16
17    // 3- expectation => ce que l'on attend
18    expect(resultat).withContext('--- Erreur Expectation ---').toBe(15);
19    // 7- ajout d'une autre expectation
20    expect(MockWarn.warnMessage).toHaveBeenCalledTimes(1);
21  });
22
23  // 4- duplication du premier IT
24
25
operations.service.ts
1 import { Injectable } from '@angular/core';
2 import { WarnService } from './warn.service';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class OperationsService {
8
9   constructor(
10    private _warn: WarnService
11  ) { }
12
13  // -----
14  // méthodes
15  // -----
16  public ajouter = (nb1: number, nb2: number) => {
17    this._warn.warnMessage('Ajout OK');
18    this._warn.warnMessage('Ajout OK');
19    return nb1 + nb2;
20  }
}

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL PORTS
Coverage summary
Statements : 88.23% ( 15/17 )
Branches : 100% ( 0/0 )
Functions : 71.42% ( 5/7 )
Lines : 86.66% ( 13/15 )
✓ Browser application bundle generation complete.
Chrome 118.0.0.0 (Windows 10) Test Injection de dépendances Ajouter 2 nombres OK ? FAILED
Expected spy WarnService.warnMessage to have been called once. It was called 2 times.
at <Jasmine>
at UserContext.apply (src/app/services/operations.service.spec.ts:20:34)
at _ZoneDelegate.invoke (node_modules/zone.js/fesm2015/zone.js:368:26)
at ProxyZoneSpec.onInvoke (node_modules/zone.js/fesm2015/zone-testing.js:273:39)
at _ZoneDelegate.invoke (node_modules/zone.js/fesm2015/zone.js:367:52)
Chrome 118.0.0.0 (Windows 10): Executed 6 of 6 (1 FAILED) (0.031 secs / 0.017 secs)
TOTAL: 1 FAILED, 5 SUCCESS
Coverage summary
Statements : 88.88% ( 16/18 )
Branches : 100% ( 0/0 )
Functions : 71.42% ( 5/7 )
Lines : 87.5% ( 14/16 )
```

# 2<sup>nd</sup> exemple : Injection de services

```
warn.service.ts ×
TP07-tests-unitaires > src > app > services > warn.service.ts > WarnService > logErreur
1 import { Injectable } from '@angular/core';
2
3 @Injectable({
4   providedIn: 'root'
5 })
6 export class WarnService {
7   // -----
8   // méthodes
9   // -----
10  public warnMessage = (msg: string) => {
11    console.warn(`Le message est : ${msg}`);
12  }
13  // ---
14  public logErreur = (msg: string) => {
15    console.error(`L'erreur est : ${msg}`);
16  }
17 }
```

```
operations.service.spec.ts ×
TP07-tests-unitaires > src > app > services > operations.service.spec.ts > describe('Test Injection de dépendances') callback
1 import { OperationsService } from './operations.service';
2 import { WarnService } from './warn.service';
3
4 // Création de l'objet de Test
5 describe('Test Injection de dépendances', () => {
6
7   // -----
8   // Zone de déclarations de variables - const - objets
9   // -----
10  let MockWarn:any;
11  let operations:any;
12
13  // -----
14  // traitement spécifiques communs (beforeEach - AfterEach - beforeAll)
15  // -----
16  beforeEach(()=> {
17    MockWarn = jasmine.createSpyObj('WarnService', ['warnMessage', 'logErreur']);
18    operations = new OperationsService(MockWarn);
19  });
20
21  // -----
22  // liste des spécifications
23  // -----
24
25  it('Ajouter 2 nombres OK ?', () => {
26    const resultat = operations.ajouter(10, 5);
27    expect(resultat).withContext('--- Erreur Expectation ---').toBe(15);
28    expect(MockWarn.warnMessage).toHaveBeenCalledTimes(1);
29  });
30
31  it('Soustraire 2 nombres OK ?', () => {
32    const resultat = operations.soustraire(10, 5);
33    expect(resultat).withContext('--- Erreur Expectation ---').toBe(5);
34  });
35 })
```

```
operations.service.ts ×
TP07-tests-unitaires > src > app > services > operations.service.ts > OperationsService > ajouter
1 import { Injectable } from '@angular/core';
2 import { WarnService } from './warn.service';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class OperationsService {
8
9   constructor(
10    private _warn: WarnService
11  ) { }
12
13  // -----
14  // méthodes
15  // -----
16  public ajouter = (nb1: number, nb2: number) => {
17    this._warn.warnMessage('Ajout OK');
18    // this._warn.warnMessage('Ajout OK');
19    return nb1 + nb2;
20  }
21
22  public soustraire = (nb1: number, nb2: number) => {
23    this._warn.warnMessage('Soustraction OK');
24    return nb1 - nb2;
25  }
26 }
```

# 2<sup>nd</sup> exemple : Injection de services

```
warn.service.ts ×
TP07-tests-unitaires > src > app > services > warn.service.ts > WarnService > warnMess
1 import { Injectable } from '@angular/core';
2
3 @Injectable({
4   providedIn: 'root'
5 })
6 export class WarnService {
7   // -----
8   // méthodes
9   // -----
10  public warnMessage = (msg: string) => {
11    console.warn(`Le message est : ${msg}`);
12  }
13  // ---
14  public logErreur = (msg: string) => {
15    console.error(`L'erreur' est : ${msg}`);
16  }
17 }

operations.service.spec.ts ×
TP07-tests-unitaires > src > app > services > operations.service.spec.ts > describe('Test Injection de dépendances') callb
1 import { TestBed } from "@angular/core/testing";
2 import { OperationsService } from "../operations.service";
3 import { WarnService } from "../warn.service";
4
5 // Création de l'objet de Test
6 describe('Test Injection de dépendances', () => {
7   // -----
8   // Zone de déclarations de variables - const - objets
9   // -----
10  let MockWarn: any;
11  let operations: any;
12  // -----
13  // traitement spécifiques communs (beforeEach - AfterEach - beforeEach)
14  // -----
15  beforeEach(() => {
16    MockWarn = jasmine.createSpyObj('WarnService', ['warnMessage', 'logErreur']);
17    // -----
18    // Création d'un Objet complet de test et de simulation : TESTBED
19    // Créons une vraie injection de dépendances avec les providers
20    // -----
21    TestBed.configureTestingModule({
22      providers: [
23        // injection de dépendances
24        OperationsService,
25        {
26          provide: WarnService,
27          // on n'utilise pas la méthode de WarnService
28          // mais la méthode du Mock ('warnMessage', 'logErreur')
29          useValue: MockWarn
30        }
31      ]
32    });
33    // -----
34    operations = TestBed.inject(OperationsService);
35    // -----
36  });
37  // -----
38  // liste des spécifications
39  // -----
40  it('Ajouter 2 nombres OK ?', () => {
41    const resultat = operations.ajouter(10, 5);
42    expect(resultat).withContext('--- Erreur Expectation ---').toBe(15);
43    expect(MockWarn.warnMessage).toHaveBeenCalledTimes(1);
44  });
45
```

```
operations.service.ts ×
TP07-tests-unitaires > src > app > services > operations.service.ts > OperationsService >
13 // -----
14 // méthodes
15 // -----
16 public ajouter = (nb1: number, nb2: number) => {
17   this._warn.warnMessage('Ajout OK');
18   // this._warn.warnMessage('Ajout OK');
19   return nb1 + nb2;
20 }
21
22 public soustraire = (nb1: number, nb2: number) => {
23   this._warn.warnMessage('Soustraction OK');
24   return nb1 - nb2;
25 }
26 }
```

# TP : Rédiger un TU

Rédiger le test unitaire du pipe see-more.pipe.ts

Le pipe seeMorePipe permet grâce à la méthode substring de javascript de n'afficher qu'une partie du texte.

```
see-more.pipe.ts X
src > app > shared > pipes > see-more.pipe.ts > ...
1  import { Pipe, PipeTransform } from '@angular/core';
2
3  @Pipe({
4    name: 'seeMore',
5    standalone:true
6  })
7  export class SeeMorePipe implements PipeTransform {
8
9    transform(datas: string, nbCar: number, texte: string) {
10
11      return `${datas.substring(0, nbCar)} ${texte}`;
12    }
13  }
```

# TP : Rédiger un TU

Rédiger le test unitaire du pipe see-more.pipe.ts

Le pipe seeMorePipe permet grâce à la méthode substring de javascript de n'afficher qu'une partie du texte.

En le codant, les dev n'ont pas forcément penser à plusieurs cas :

- Le nombre de caractères est absent ?
- La position initiale n'est pas fournie ?
- Le texte n'est pas fourni par le service ?

```
<p class="alert alert-primary">Description : {{dive.description | seeMore:15:'lire la suite'}}</p>
```

Rédiger le test unitaire permettant de gérer ces cas de figure et par conséquence modifier le fichier see-more.pipe.ts

# L'Internationalisation I18N



Copyright Michel BOUJOLESI - ORSYS

# L'Internationalisation I18N

Comment traduire dans un projet Angular ?



Commande d'installation et d'intégration de package : `ng add @angular/localize`

Copyright Michel BOCCIOLESI - ORSYS

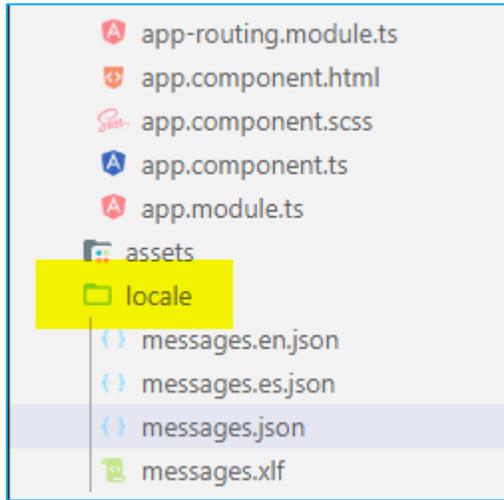
# L'Internationalisation I18N

```
EXPLORATEUR
ÉDITEURS OUVERTS
  i18n.component.html src\app...
WEBAPP
  .angular
  node_modules
  src
    app
      sharedModels
      webApp
        formation
          firestore-firebase
          i18n
            i18n.component.html
            i18n.component.scss
            i18n.component.ts
          ngx-store
          pwa
          reactive-forms
          rxjs
        root
          accueil
          body
            body.component.html
            body.component.scss
            body.component.ts
          footer
          header
          landing-page
            accueil.module.ts
          films

src > app > webApp > formation > i18n > i18n.component.html > div.alert.alert-primary > p > em
1 <div class="alert alert-primary">
2
3 <h1>L'Internationalisation I18N</h1>
4
5 <div i18n="@@div-vue">
6
7 <p><i class="bi bi-arrow-right-circle"></i> Commande : ng add @angular/localize</p>
8 <p>1- Marquer les éléments à traduire avec la directive i18n</p>
9 <p>2-Créer le fichier de traduction messages.json (voir messages.xlf[xml])
10 | <br>Commande : ng extract-i18n --output-path src/locale [en option --format="json"]
11 </p>
12 <p>3- Enregistrer le messages.json(xlf) en messages.[langue].json</p>
13 <p>4- Apporter les traductions dans ces fichiers messages.[langue].json</p>
14 <p>5-Editer angular.json</p>
15 </div>
16
17 <div>
18 <!-- contenu à traduire -->
19 <p i18n>Bonjour Angular !</p>
20 <p i18n="@@slogan">Angular est le meilleur Framework !</p>
21 <p>
22 | 
23 </p>
24 <p>
25 | <a i18n i18n-href href="www.angular.io">Doc Angular IO</a>
26 </p>
27 </div>
28
29 <p><em><i class="bi bi-arrow-right-circle"></i> une alternative intéressante ngx-translate</em></p>
30
31 </div>
32
```

# L'Internationalisation I18N

Commande : `ng extract-i18n --output-path src/locale --format="json"`



```
messages.en.json  messages.json X
formation-any-2023-09 > TP02-i18n > src > locale > messages.json > ...
1  {
2    "locale": "fr",
3    "translations": {
4      "div-vue": "{$START_PARAGRAPH}L'internationalisation I18N{$CLOSE_PARAGRAPH}{{$S
5      "404420203264888552": "Bonjour Angular !",
6      "slogan": "Angular est le meilleur Framework !",
7      "4266238317959690326": "assets/images/drapeaux/fr/drapeau_fr.jpg",
8      "4425774926678752758": "Mon Drapeau",
9      "3194024123575565444": "www.angular.io",
10     "8703060702766654522": "Doc Angular IO",
11     "4435388665562277072": "Formation Angular Développement Avancé"
12   }
13 }
```

# L'Internationalisation I18N

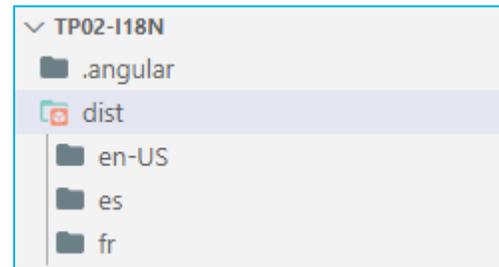


Editer Angular.json

```
messages.en.json  angular.json 1 x
formation-any-2023-09 > TP02-i18n > angular.json > ...
1 {
2   "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
3   "version": 1,
4   "newProjectRoot": "projects",
5   "projects": {
6     "webApp": {
7       "i18n": {
8         "sourceLocale": "fr",
9         "locales": {
10          "en-US": "src/locale/messages.en.json",
```

```
messages.en.json  angular.json 1 x
formation-any-2023-09 > TP02-i18n > angular.json > ...
41   "node_modules/bootstrap-icons/font/bootstrap
42   "src/styles.scss"
43   },
44   "scripts": [
45     "node_modules/bootstrap/dist/js/bootstrap.m
46   ]
47   },
48   "configurations": {
49     "en": {
50       "localize": [
51         "en-US"
52       ]
53     },
```

ng build --localize



Copyright Michel

# L'Internationalisation I18N

TP : Rajouter la traduction en espagnol

*Hola Angular !  
Angular esta mejor !  
assets/images/drapeaux/es/drapeau\_es.jpg  
Mi bandera*

- TP :
- Changer le dossier de compilation en build
  - Utiliser une cible ES2017
  - désactiver ou activer le mode Strict

Copyright Michel BOCCIOLESI - ORSYS

# L'Internationalisation I18N

Comment traduire non pas la vue mais les données qui proviennent du TS ?

```
constructor() {  
  // this.title=$localize `Formation Angular Développement Avancé`;  
  this.title=$localize `:@@id:Formation Angular Développement Avancé`;  
}
```

Copyright Mich

# L'Internationalisation I18N

Option : afficher la traduction en mode serve !

```
angular.json > $schema
78     extractLicenses : false,
79     "sourceMap": true,
80     "namedChunks": true
81   }
82 },
83   "defaultConfiguration": "production"
84 },
85   "serve": {
86     "builder": "@angular-devkit/build-angular:dev-server",
87     "configurations": {
88       "production": {
89         "browserTarget": "webApp:build:production"
90       },
91       "development": {
92         "browserTarget": "webApp:build:development"
93       },
94       "page_en": {
95         "browserTarget": "webApp:build:en"
96       },
97       "page_es": {
98         "browserTarget": "webApp:build:es"
99       }
100     }
  }
```

```
package.json > {} scripts
1 {
2   "name": "web-app",
3   "version": "1.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve --open --port=3000",
7     "start-en": "ng serve --open --port=3000 --configuration=page_en",
8     "start-es": "ng serve --open --port=3000 --configuration=page_es",
9     "build": "ng build",
10    "build-localize": "ng build --localize",
11    "watch": "ng build --watch --configuration development --stats-json",
12    "test": "ng test",
13    "json-server": "json-server -p 3001 src/assets/json/films.json ",
14    "json-server-post": "json-server -p 3002 src/assets/json/bdd.json",
15    "analyze": "webpack-bundle-analyzer dist/stats.json"
16  },
  }
```

# Le Server Side Rendering « SSR »



Copyright Michel BOCCIOLESI ORSYS

## SSR ou CSR

Le « **S**erver **S**ide **R**endering » existe depuis les années 1990 !

La page est rendue depuis le back et est affichée dans le navigateur du client.

Début 2010, les premiers « Webpack » et les « **S**ingle **P**age **A**pplication » font leur arrivée dans le monde du développement Web.

On parle alors de **C**lient **S**ide **R**endering, puisque le projet final contient seulement une page index.html qui fait elle même appel à des « chunk files javascript » (*des fichiers JavaScripts buildés*).

## SSR ou CSR

Pourquoi revient-on depuis peu au SSR ?

- Il y aura de meilleures performance en SEO, bien que les moteurs dont Google garantit que les pages CSR seront correctement indexées, cela nécessite plus de temps mais les moteurs peuvent le faire !
- Le temps de chargement du contenu est un peu plus rapide, car on doit attendre que le contenu total de l'applis soit chargé (NB: le lazy loading solutionne cette problématique ...)
- React, Vue et Angular l'implémentent facilement !

# SSR ou CSR

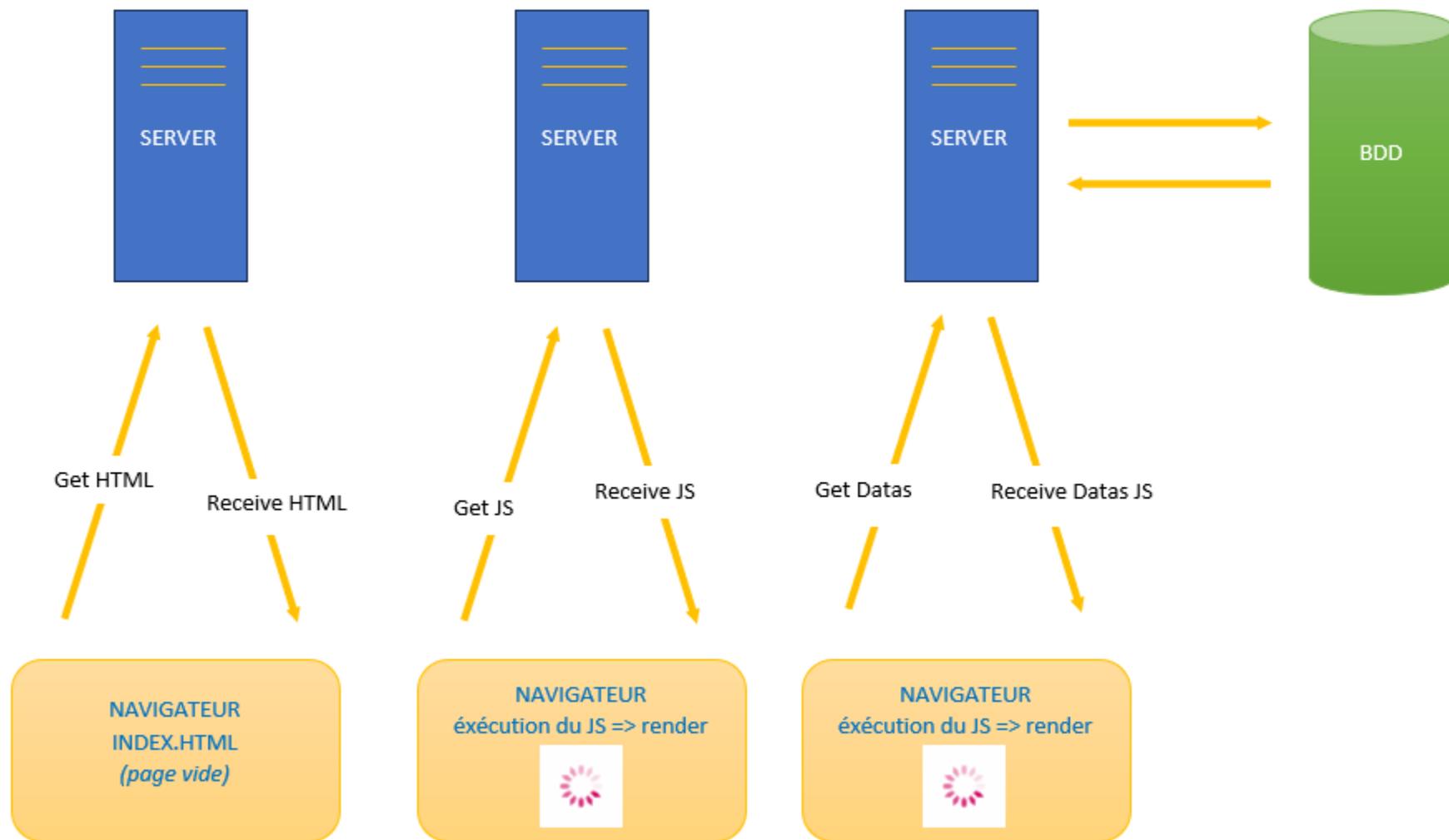
## Pourquoi revient-on depuis peu au SSR ?

- Il y aura de meilleures performance en SEO, bien que les moteurs dont Google garantit que les pages CSR seront correctement indexées, cela nécessite plus de temps mais les moteurs peuvent le faire !
- Le temps de chargement du contenu est un peu plus rapide, car on doit attendre que le contenu total de l'applis soit chargé (NB: le lazy loading solutionne cette problématique ...)
- React, Vue et Angular l'implémentent facilement !

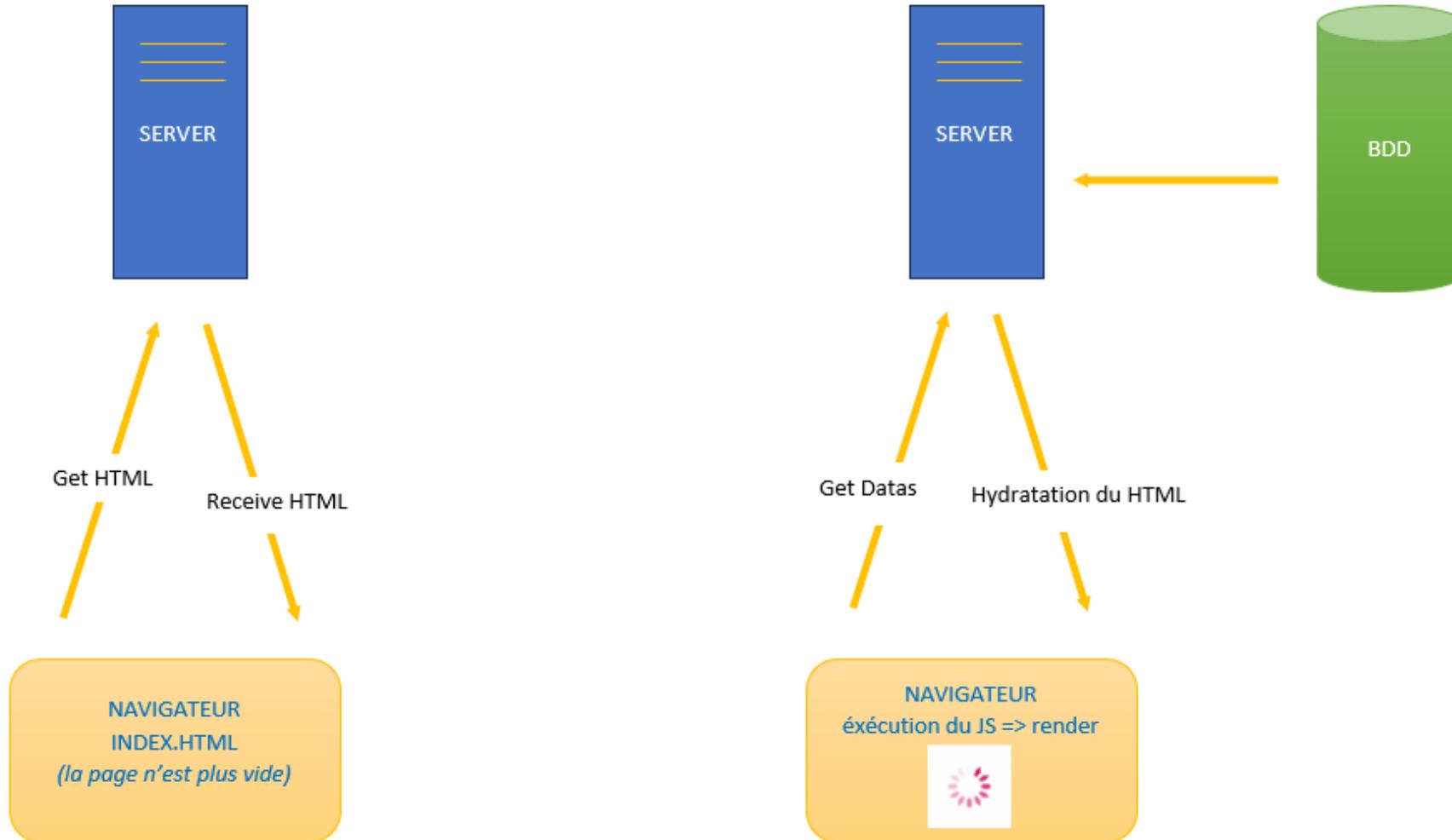
## Inconvénients :

- La charge « back » est évidemment plus couteuse
- Il est nécessaire d'implémenter Node.js car ce n'est plus un simple hébergement web statique !

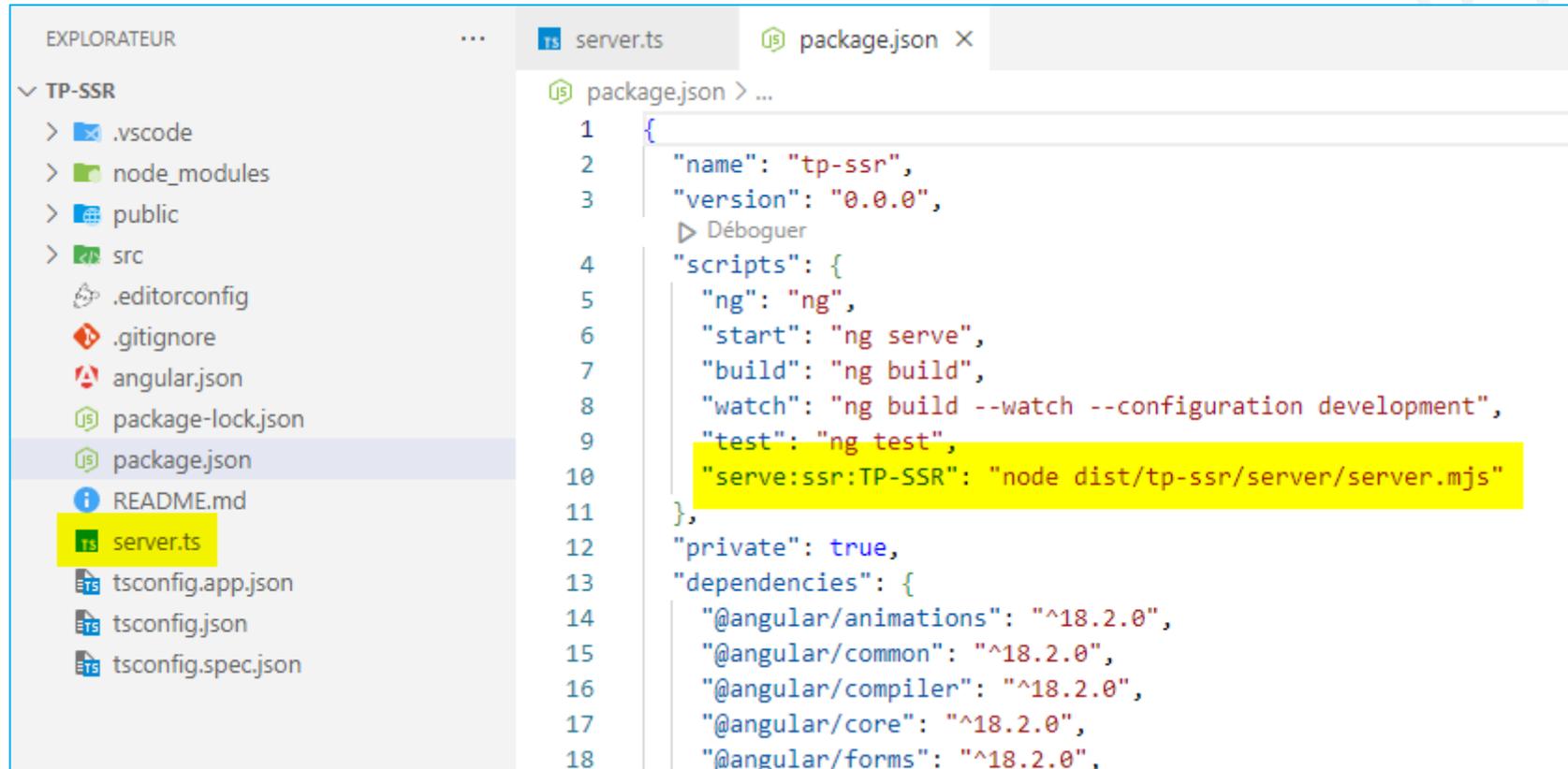
CSR : Client-Side Rendering | Single Page Application : Le rendu se fait côté client



## SSR : Server-Side Rendering



Mise en pratique : <https://angular.dev/guide/ssr>



```
1 {
2   "name": "tp-ssr",
3   "version": "0.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve",
7     "build": "ng build",
8     "watch": "ng build --watch --configuration development",
9     "test": "ng test",
10    "serve:ssr:TP-SSR": "node dist/tp-ssr/server/server.mjs"
11  },
12  "private": true,
13  "dependencies": {
14    "@angular/animations": "^18.2.0",
15    "@angular/common": "^18.2.0",
16    "@angular/compiler": "^18.2.0",
17    "@angular/core": "^18.2.0",
18    "@angular/forms": "^18.2.0",
```

EXPLORATEUR

- TP-SSR
  - .vscode
  - node\_modules
  - public
    - favicon.ico
  - src
    - app
      - index.html
      - main.server.ts
      - main.ts
      - styles.css
    - .editorconfig
    - .gitignore
    - angular.json
    - package-lock.json
    - package.json
    - README.md
    - server.ts
    - tsconfig.app.json
    - tsconfig.json
    - tsconfig.spec.json

server.ts package.json main.server.ts X

```
src > main.server.ts > ...
1 import { bootstrapApplication } from '@angular/platform-browser';
2 import { AppComponent } from './app/app.component';
3 import { config } from './app/app.config.server';
4
5 const bootstrap = () => bootstrapApplication(AppComponent, config);
6
7 export default bootstrap;
8
```

```
8   "watch": "ng build --watch --configuration development",
9   "test": "ng test",
10  "ssr": "node dist/tp-ssr/server/server.mjs"
11  },
```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL PORTS

● PS C:\Users\Michel\Desktop\TP-SSR> npm run build

```
> tp-ssr@0.0.0 build
> ng build
```

#### Browser bundles

Initial chunk files	Names	Raw size	Estimated transfer size
main-YFAVBIXK.js	main	219.77 kB	59.76 kB
polyfills-FFHMD2TL.js	polyfills	34.52 kB	11.28 kB
styles-5INURTSO.css	styles	0 bytes	0 bytes
	Initial total	254.29 kB	71.05 kB

#### Server bundles

Initial chunk files	Names	Raw size
server.mjs	server	1.11 MB
chunk-WHPWUB66.mjs	-	528.79 kB
polyfills.server.mjs	polyfills.server	268.60 kB
chunk-YFYEK1WGK.mjs	-	19.16 kB
chunk-5XUXGTUW.mjs	-	2.55 kB
render-utils.server.mjs	render-utils.server	1.46 kB
main.server.mjs	main.server	149 bytes

Lazy chunk files	Names	Raw size
chunk-VCVCHJZV.mjs	xhr2	12.07 kB

#### Prerendered 1 static route.

Application bundle generation complete. [13.800 seconds]

Output location: C:\Users\Michel\Desktop\TP-SSR\dist\tp-ssr

○ PS C:\Users\Michel\Desktop\TP-SSR> npm run ssr

```
> tp-ssr@0.0.0 ssr
> node dist/tp-ssr/server/server.mjs
```

Node Express server listening on http://localhost:4000

# Annexes



Copyright Michel POCQUOLESI - ORSYS

# Rappels – Révisions Ecmascript\*



\* Si besoin

# Ecmascript 2015+



Ecmascript, la normalisation du langage Javascript a révolutionné Javascript en 2015 (ES6) en le dotant de fonctionnalités et spécificités dignes d'un « langage de haut niveau ».

Cette transformation totale et globale devenait nécessaire au vu des besoins des développements front web et mobile. *Angular utilise Ecmascript.*

Les améliorations les plus importantes du langage concernant :

1. La définition des variables et leur portée : VAR, LET et CONST
2. La possibilité de créer des modules de code exportable et importable : IMPORT, EXPORT
3. L'écriture simplifiée des FAT ARROWS `<script src="js/index.js" type="module" defer></script>`
4. La string interpolation des variables avec `${maVariable}` et la concatenation avec les back stits
5. Les itérateurs et les boucles FOR OF, FOR IN et FOR OF ENTRIES
6. Les spread operators pour déstructurer les tableaux
7. La notion de promesses afin de gérer les événements et les procédures asynchrones
8. Les objets et les CLASS\*

# Ecmascript 2015+



```
> for (var i=0; i<=3 ; i++ ) { console.log(i); }
0 VM5352:1
1 VM5352:1
2 VM5352:1
3 VM5352:1
< undefined
> console.log(i);
4 VM5411:1
< undefined
> for (let j=0; j<=3 ; j++ ) { console.log(j); }
0 VM5535:1
1 VM5535:1
2 VM5535:1
3 VM5535:1
< undefined
> console.log(j)
✖ Uncaught ReferenceError: j is not defined VM5613:1
  at <anonymous>:1:13
>
```

- ORSYS

Copy

# Ecmascript 2015+



```
> for ( let val of tab ) { console.log(val); }
ok VM6316:1
cool VM6316:1
< undefined
> for ( let i in tab ) { console.log(i); }
0 VM6332:1
1 VM6332:1
< undefined
> for (let[i,val] of tab.entries() ) { console.log(i, val);}
0 'ok' VM6348:1
1 'cool' VM6348:1
< undefined
>
```

Copyright

ORSYS

# EcmaScript 2015+



```
1 // nouveautés (fonctions)
2 function direBonjour(nom = "SkyWalker") {
3     // passage d'arguments par défaut
4     // console.log("Bonjour " + nom);
5
6     // string interpolation des variables
7     console.log(`Bonjour ${nom}`);
8     document.querySelector("#resultat").innerHTML = `Bonjour ${nom}`;
9 }
10
11 const direBonjour2 = (prenom, nom) => {
12     console.log(`Bonjour ${prenom} ${nom}`);
13     document.querySelector("#resultat").innerHTML = `Bonjour ${prenom} ${nom}`;
14 }
15
16 direBonjour(); // javascript s'écrit en camelCase majuscule à chaque mot sauf le 1er
17 direBonjour2("Dark", "Vador");
18
```

# Ecmascript 2015+



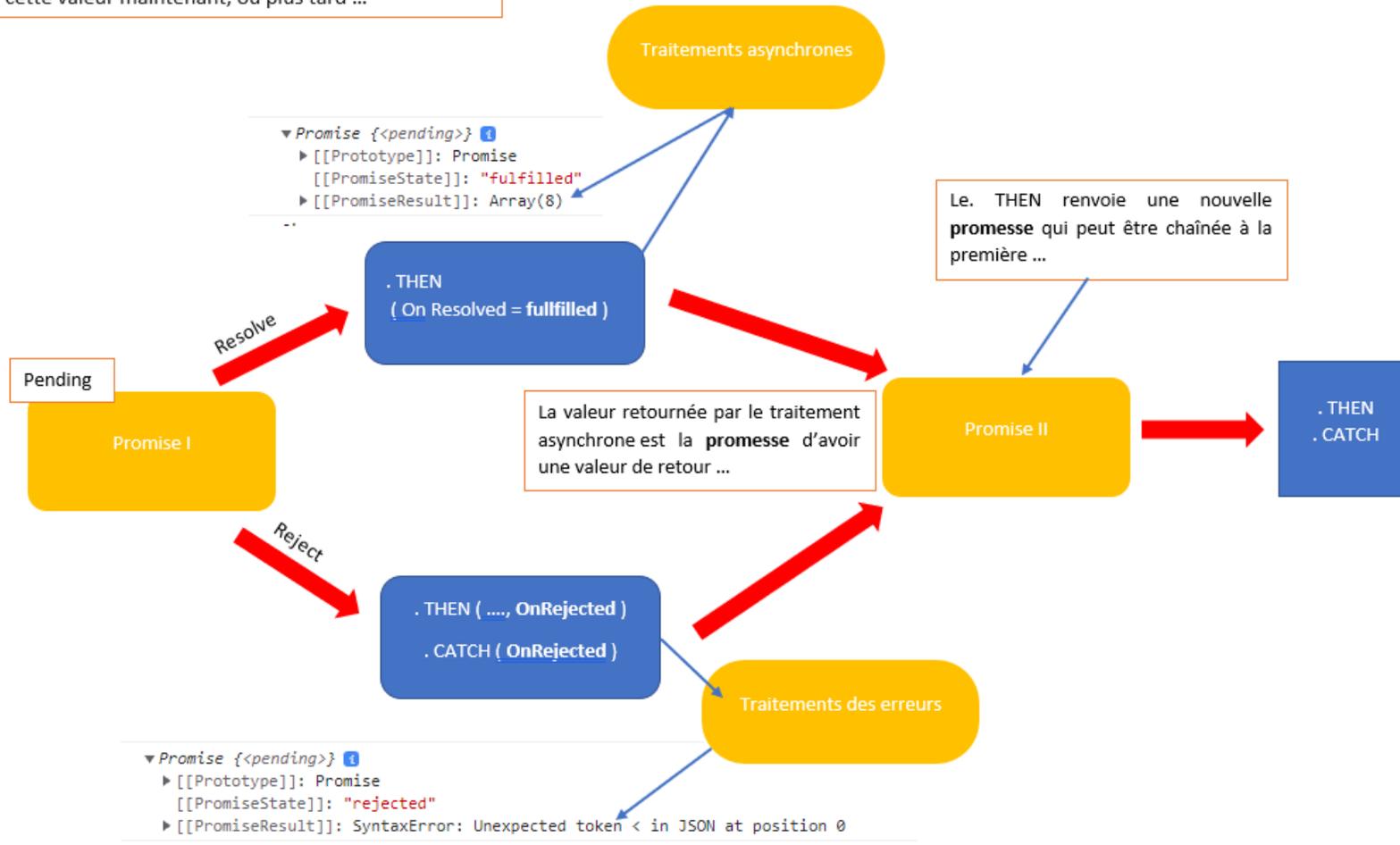
```
58 // -----  
59 let personne = {  
60   prenom: "Luke",  
61   nom: "Skywalker",  
62   // tableaux  
63   langages: ["JS", "React", "NG", "Vue"],  
64   // méthodes  
65   nomComplet() {  
66     return `${this.prenom} ${this.nom}`;  
67   }  
68 }  
69 console.log(personne.nomComplet());  
70  
71 // ---- Nouveautés class (PascalCase = maj à chaque mot )  
72 class Personne {  
73  
74   // Constructor  
75   constructor(nom, age) {  
76     this._nom = nom;  
77     this._age = age;  
78   }  
79   // Méthodes  
80   infosPersonne() {  
81     return `Bonjour je m'appelle ${this._nom} et j'ai ${this._age} ans.`;  
82   }  
83 }  
84  
85 let p1 = new Personne("Emilie", 35);  
86 console.log(p1);
```

# Ecmascript 2015+



VS

La valeur retournée par le traitement **asynchrone** associé aux promesses est une **promesse** d'avoir cette valeur maintenant, ou plus tard ...



Co

# Ecmascript 2015+



```
1 export const fnFilm = () => {
2   // ES6 : fonction fetch
3   // simplification d'écriture d'Ajax
4   // amélioration du process JS ( Promises )
5
6   // 1- Request Query (requête Infos):
7   // const _url = 'https://test.webjs.fr/films.json';
8   const _url = 'https://dev.webjs.fr/films.json';
9   // const _url='http://127.0.0.1:3000/json/films.json'; // en local avec un serveur json
10
11  // 2- Request Init Query
12  // définir la méthode (get ou post) ... ou patch ou put ou delete)
13  const _method = 'get';
14  // définir nos entêtes HTTP
15  const _headers = new Headers();
16
17  _headers.append('Content-Type', 'text/json'); //type mime (ex:image/png)
18  // _headers.append('Access-Control-Allow-Origin', 'cors');
19
20  // 3- Ecriture du FETCH
21  // fetch renvoie une promesse
22  fetch(
23    _url,
24    {
25      method: _method,
26      headers: _headers
27    }
28  ).then(
29    // si on arrive à avoir une réponse du serveur
30    // état onFullFilled
31    (responseHTTP) => {
32      // le then récupère une réponse 200 ok 404
33      console.warn('On Fullfilled du 1er Then');
34      console.log(responseHTTP);
35      console.log(responseHTTP.body);
```

ORSYS

Copy

# Le Detection Change Mode (value based) Zone.JS ?



Copyright Michele Rocciocolesi - ORSYS

# Le Detection Change Mode Angular

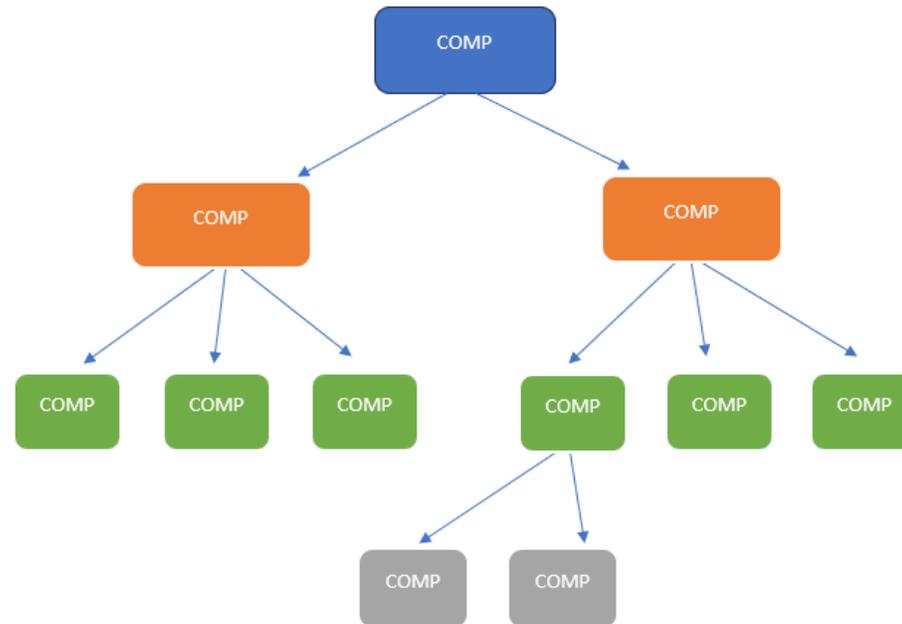
Lorsque des propriétés d'entrée (**Input**) ou des événements d'entrée sont **déclenchés**, Angular met à jour **l'arbre des composants pour détecter d'éventuels changements** à apporter aux affichages des Vues.

Modifier le comportement par défaut du « Change Detection Mode » d'Angular peut être utile lorsque des composants « rendent » (affichent) des données qui ne changent pas souvent.

Le nombre de cycles de change detection sera réduit et les performances de l'application en seront augmentées.

# Le Detection Change Mode Angular

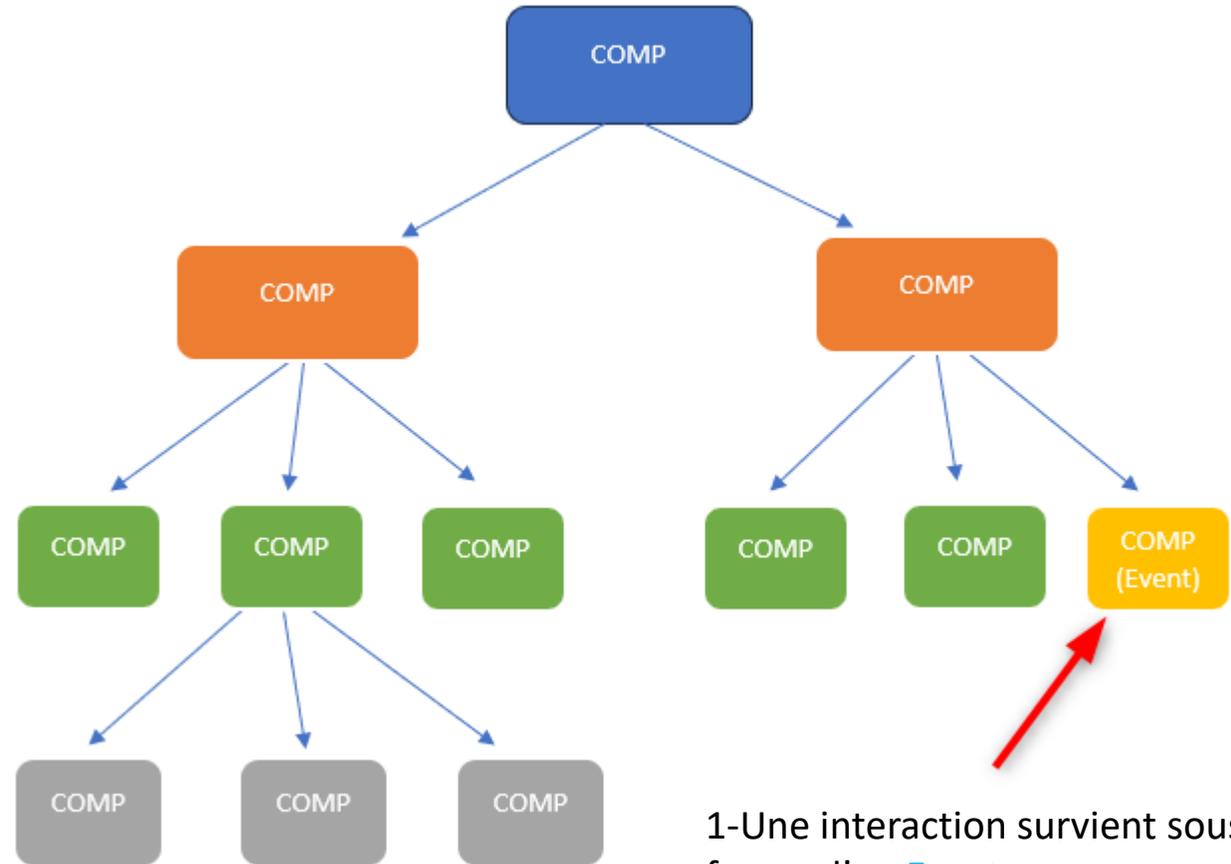
Comment Angular met à jour l'affichage des composants imbriqués par des Inputs, dans une stratégie composants parents-enfants



# Le Detection Change Mode par défaut

2-Angular parcourt tout l'arbre des composants pour afficher un changement de valeur sur les composants

```
@Component({  
  selector: 'app-liste-des-films',  
  templateUrl: './liste-des-films.component.html',  
  styleUrls: ['./liste-des-films.component.scss'],  
  changeDetection: ChangeDetectionStrategy.Default,  
})
```



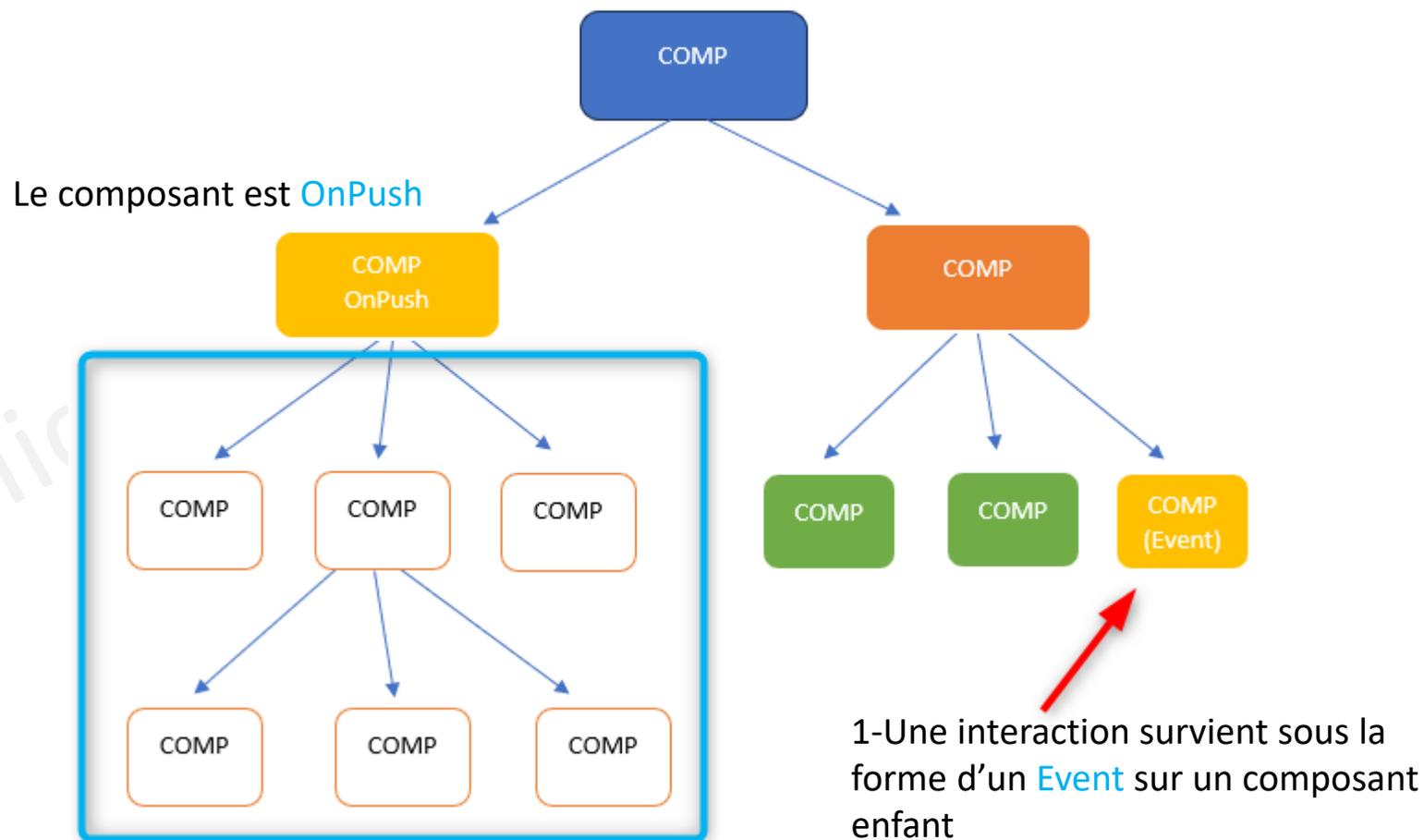
1-Une interaction survient sous la forme d'un **Event** sur un composant enfant

# Le Detection Change Mode onPush

Angular **limite** le nombre de vérifications sur les enfants

Si la référence de l'objet (valeur...) passée en Input ne change pas, le composant enfant ne doit pas être modifié.

Angular **arrête son parcours...**



# La class ChangeDetectorRef

On peut **forcer** la mise à jour de l'affichage de la Vue d'un composant de **manière explicite**, sans attendre (ou dépendre) du déclenchement des valeurs d'entrées du composant.

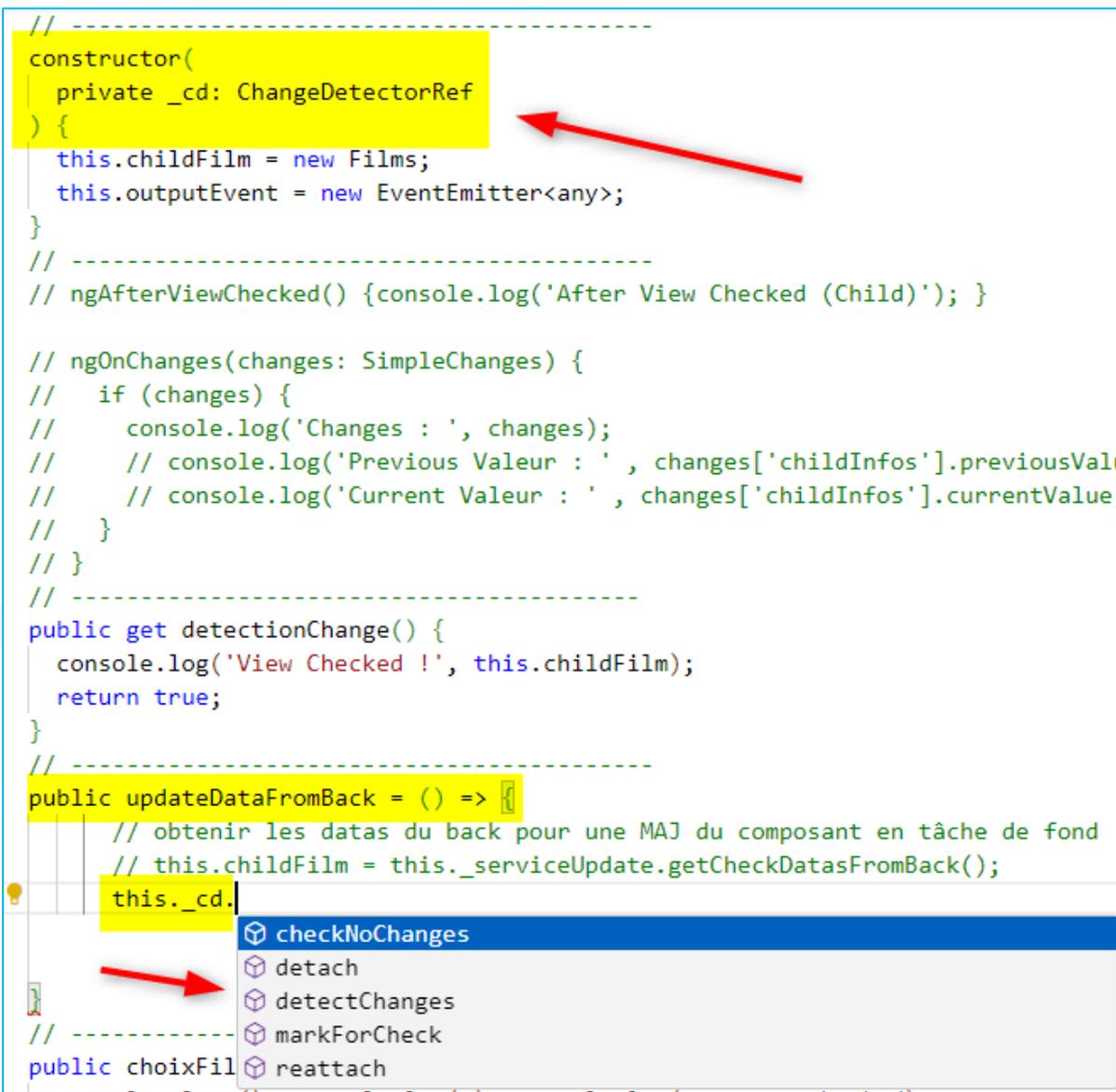
```
films-details.component.ts x
webApp2 > src > app > webApp > root > films > composants > films-details > films-details.component.ts > FilmsDe
4  @Component({
5    selector: 'app-films-details',
6    templateUrl: './films-details.component.html',
7    styleUrls: ['./films-details.component.scss'],
8    changeDetection: ChangeDetectionStrategy.OnPush
9  })
10 export class FilmsDetailsComponent {
11
12   @Input() childFilm: Films;
13   // @Input() set childInfos(value: string) { // console.log(value); };
14   @Output() outputEvent: EventEmitter<any>;
15
16   // -----
17   constructor(
18     private _cd: ChangeDetectorRef
19   ) {
20     this.childFilm = new Films;
21     this.outputEvent = new EventEmitter<any>;
22   }
}
```

# La class ChangeDetectorRef

« **markForCheck** » indique que le composant doit être vérifié lors du prochain parcours de l'arbre des composants, même si la stratégie de mise à jour de « change detection » est **OnPush**

```
// -----  
public updateDataFromBack = () => {  
    // obtenir les datas du back pour un MAJ du composant en tâche de fond  
    // this.childFilm = this._serviceUpdate.getCheckDdatasFromBack();  
    this._cd.markForCheck();  
}
```

```
// -----  
constructor(  
    private _cd: ChangeDetectorRef  
) {  
    this.childFilm = new Films;  
    this.outputEvent = new EventEmitter<any>;  
}  
// -----  
// ngAfterViewChecked() {console.log('After View Checked (Child)'); }  
  
// ngOnChanges(changes: SimpleChanges) {  
//     if (changes) {  
//         console.log('Changes : ', changes);  
//         // console.log('Previous Valeur : ', changes['childInfos'].previousValue);  
//         // console.log('Current Valeur : ', changes['childInfos'].currentValue);  
//     }  
// }  
// -----  
public get detectionChange() {  
    console.log('View Checked !', this.childFilm);  
    return true;  
}  
// -----  
public updateDataFromBack = () => {  
    // obtenir les datas du back pour une MAJ du composant en tâche de fond  
    // this.childFilm = this._serviceUpdate.getCheckDdatasFromBack();  
    this._cd.  
// -----  
public choixFil
```



- checkNoChanges
- detach
- detectChanges
- markForCheck
- reattach

# La class ChangeDetectorRef

```
6   templateUrl: './films-details.component.html',
7   styleUrls: ['./films-details.component.scss'],
8   changeDetection: ChangeDetectionStrategy.OnPush
9 })
0 export class FilmsDetailsComponent {
1
2   @Input() childFilm: Films;
3   // @Input() set childInfos(value: string) { // console.log(value); };
4   @Output() outputEvent: EventEmitter<any>;
5
6   // -----
7   constructor(
8     private _cd: ChangeDetectorRef
9   ) {
10    this.childFilm = new Films;
11    this.outputEvent = new EventEmitter<any>;
12  }
13  // -----
14  // ngAfterViewChecked() {console.log('After View Checked (Child)'); }
15
16  // ngOnChanges(changes: SimpleChanges) {
17  //   if (changes) {
18  //     console.log('Changes : ', changes);
19  //     // console.log('Previous Valeur : ', changes['childInfos'].previousValue);
20  //     // console.log('Current Valeur : ', changes['childInfos'].currentValue);
21  //   }
22  // }
23  // -----
24  public get detectionChange() {
25    console.log('View Checked !', this.childFilm);
26    return true;
27  }
28  // -----
29  public updateDataFromBack = () => {
30    // obtenir les datas du back pour une MAJ du composant en tâche de fond
31    // this.childFilm = this._serviceUpdate.getCheckDatasFromBack();
32    this._cd.markForCheck();
33  }
34  }
```

PWA



Copyright Michel BOUCCICULESI - ORSYS

## PWA – les Progressive Web Apps ?

*Une Application Web ou une Application Mobile*

Mise en situation :

consulter cette appli depuis un smartphone :

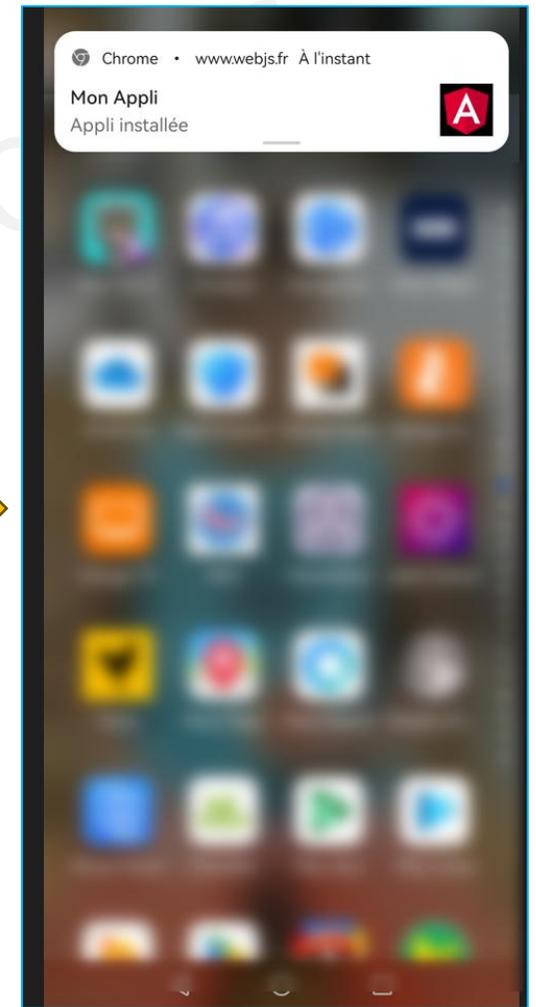
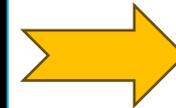
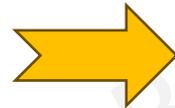
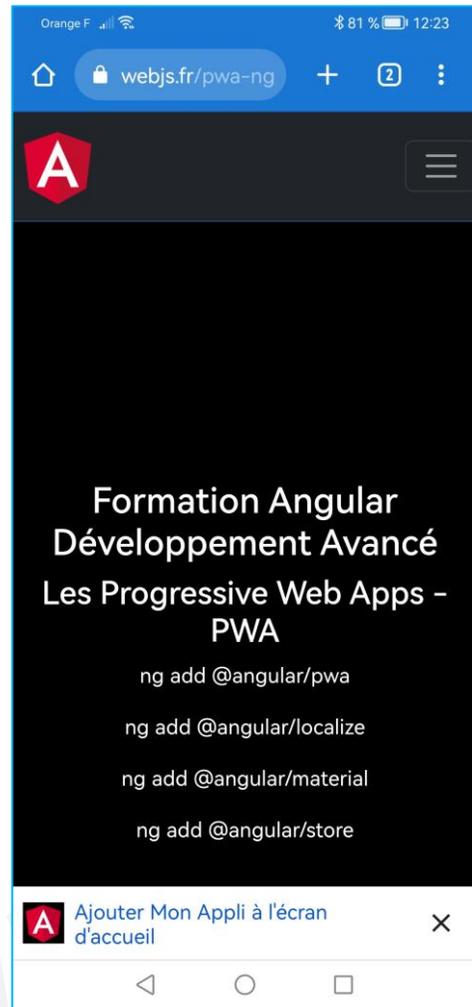
<https://webjs.fr/pwa-ng/>

<https://www.webjs.fr/pwa-vanilla/>



Copyright Michel BOCCIOLESI - ORSYS

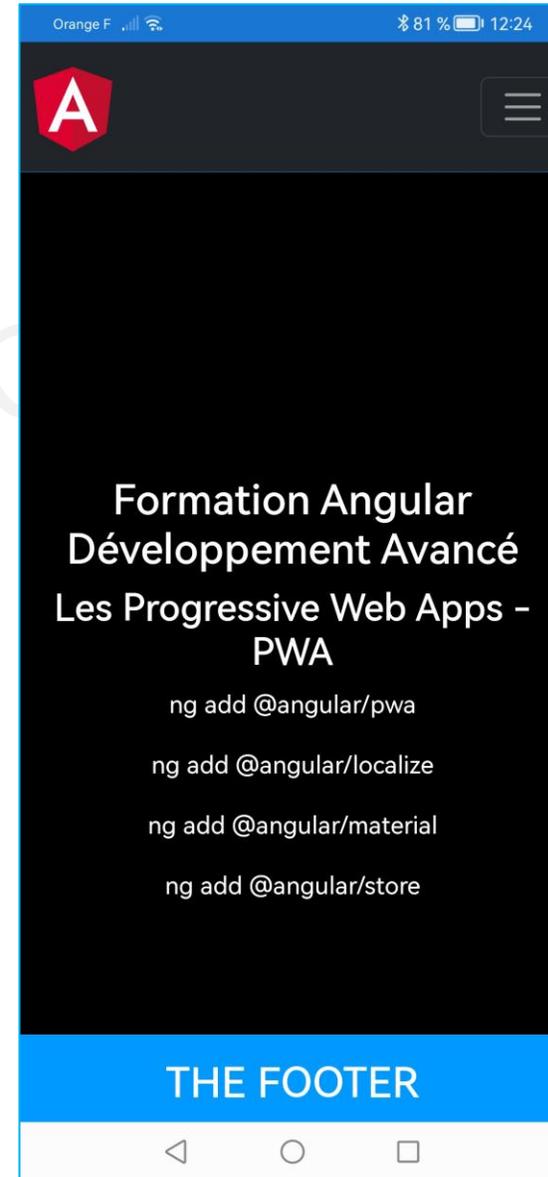
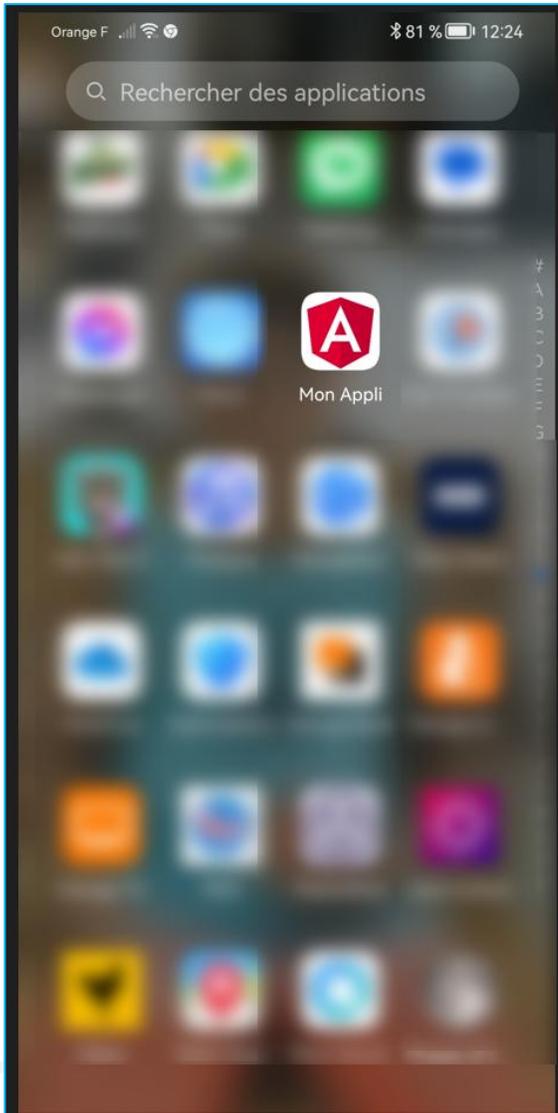
# PWA – (le fichier web-manifest)



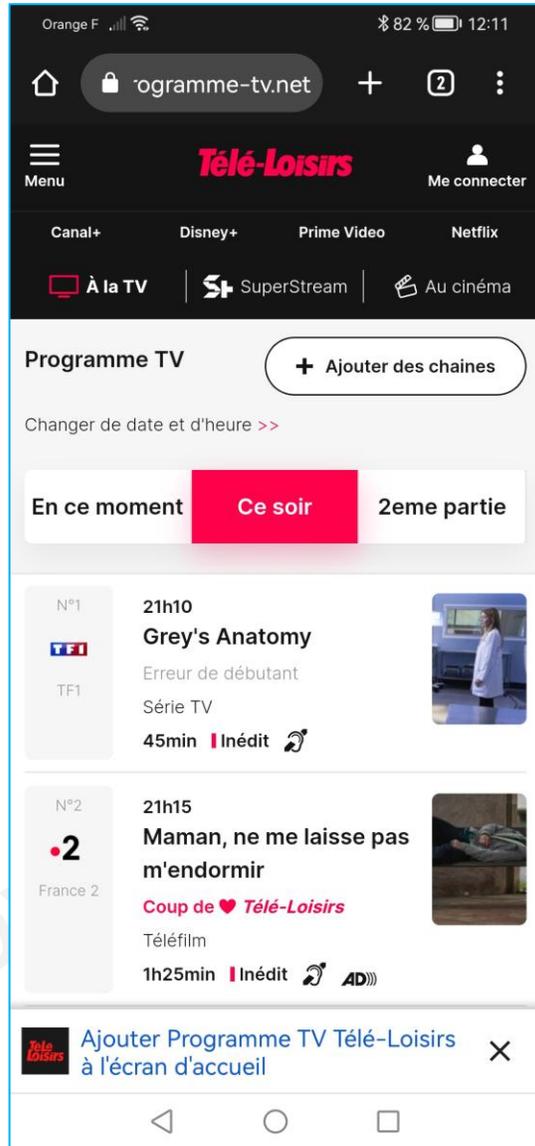
## Références :

- [https://developer.mozilla.org/fr/docs/Web/Progressive\\_web\\_apps/Guides/Making\\_PWAs\\_installable](https://developer.mozilla.org/fr/docs/Web/Progressive_web_apps/Guides/Making_PWAs_installable)
- <https://developer.mozilla.org/fr/docs/Web/Manifest>

# PWA – les Progressive Web Apps



# PWA – les Progressive Web Apps





# Commande d'installation et d'intégration de package : ng add @angular/pwa

The screenshot shows the Visual Studio Code interface with the following components:

- File Explorer (Left):** Shows the project structure for 'TD01-pwa', including folders like 'node\_modules', 'src', and 'assets', and files like 'app.module.ts', 'app.component.html', and 'ngsw-config.json'.
- Code Editor (Middle):** Displays the contents of 'package.json', 'app.module.ts', and 'angular.json'. The 'package.json' file lists dependencies for '@angular/fire', '@angular/forms', '@angular/material', '@angular/platform-browser', '@angular/platform-browser-dynamic', '@angular/router', '@angular/service-worker', 'bootstrap', and 'bootstrap-icons'. The 'app.module.ts' file shows the configuration for the ServiceWorkerModule, including the 'registerWhenStable:30000' strategy. The 'angular.json' file shows the configuration for the application, including the 'assets' and 'styles' arrays.
- Terminal (Bottom):** Shows the execution of the command 'ng add @angular/pwa'. The output indicates that the package is successfully installed and executed, listing the files created and updated, such as 'ngsw-config.json', 'src/manifest.webmanifest', and various icon files.

# npm run build – npm run watch

The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORATEUR (File Explorer):** Shows the project structure. The folder `dist \ web-app` is highlighted in yellow. A red arrow points from this folder to the `ngsw-worker.js` file in the editor.
- ÉDITEURS OUVERTS (Open Editors):** Three editor windows are open:
  - `ngsw-config.json`: Contains JSON configuration for the service worker. The `"name": "app"` and `"name": "assets"` fields are highlighted in blue.
  - `ngsw-worker.js`: Contains JavaScript code for the service worker.
  - `ngsw-config.json`: A second instance of the configuration file.
- TERMINAL:** Shows an error message and the command `npm run watch`. The command is highlighted in yellow. The error message is: `Oops, un problème s'est produit. Signalez ce bogue avec les détails ci-dessous. Signaler sur GitHub : https://github.com/lzybkr/PSReadLine/issues/new`

# PWA – les Progressive Web Apps

The screenshot shows the Chrome DevTools Application panel for a Progressive Web App (PWA) at `webjs.fr/pwa-ng/`. The browser's address bar shows the URL. The Application panel is open, displaying the following sections:

- Application:** Manifest, Service workers (highlighted in yellow), Storage.
- Storage:** Local storage, Session storage, IndexedDB, Web SQL, Cookies, Private state tokens, Interest groups, Shared storage, Cache storage.
- Background services:** Back/forward cache, Background fetch, Background sync, Bounce tracking mitigations, Notifications, Payment handler, Periodic background sync, Push messaging, Reporting API.

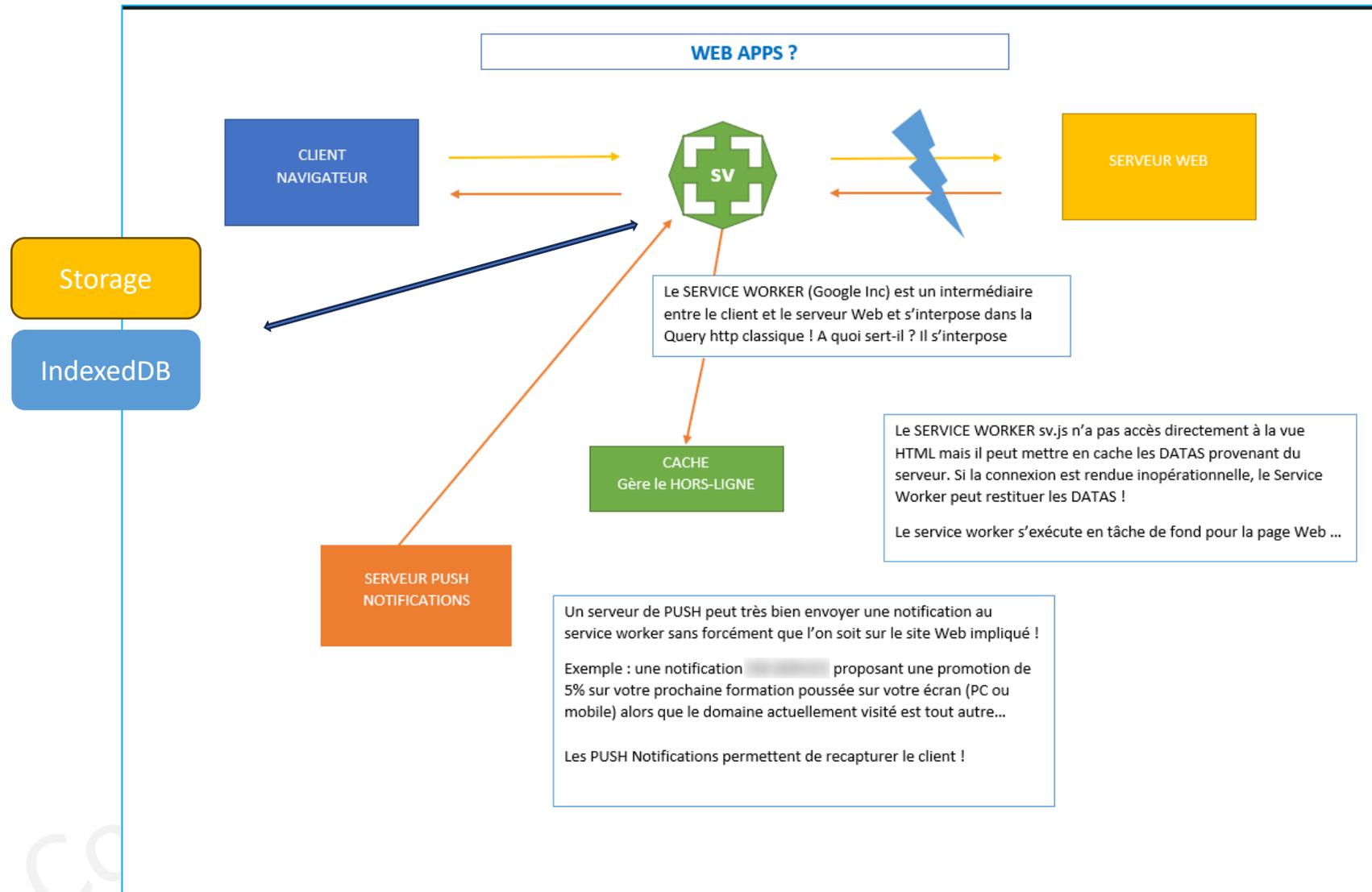
The Service workers section is expanded, showing the following details for `https://webjs.fr/pwa-ng/`:

- Source:** `ngsw-worker.js` (highlighted in yellow).
- Received:** 04/10/2023 17:44:53.
- Status:** #3396 activated and is running. [stop](#)
- Push:** Test push message from DevTools. [Push](#)
- Sync:** pwa. [Sync](#)
- Periodic Sync:** periodic. [Periodic Sync](#)
- Update Cycle:**

Version	Update Activity	Timeline
▶ #3396	Install	
▶ #3396	Wait	
▶ #3396	Activate	██████████

At the bottom, there is a section for **Service workers from other origins** with a link to [See all registrations](#).

# PWA – le Service Worker



[https://developer.mozilla.org/fr/docs/Web/API/Service Worker API](https://developer.mozilla.org/fr/docs/Web/API/Service_Worker_API)  
<https://angular.io/guide/service-worker-config>

# PWA – le Storage

```
const addToStorage = (objet) => {  
  
  console.log(JSON.stringify(objet)); // output {'':''}  
  const nbDocs = localStorage.length;  
  localStorage.setItem(nbDocs, JSON.stringify(objet));  
};
```

```
const delNotes = () => {  
  localStorage.clear();  
  eltUlliste.innerHTML='';  
};
```

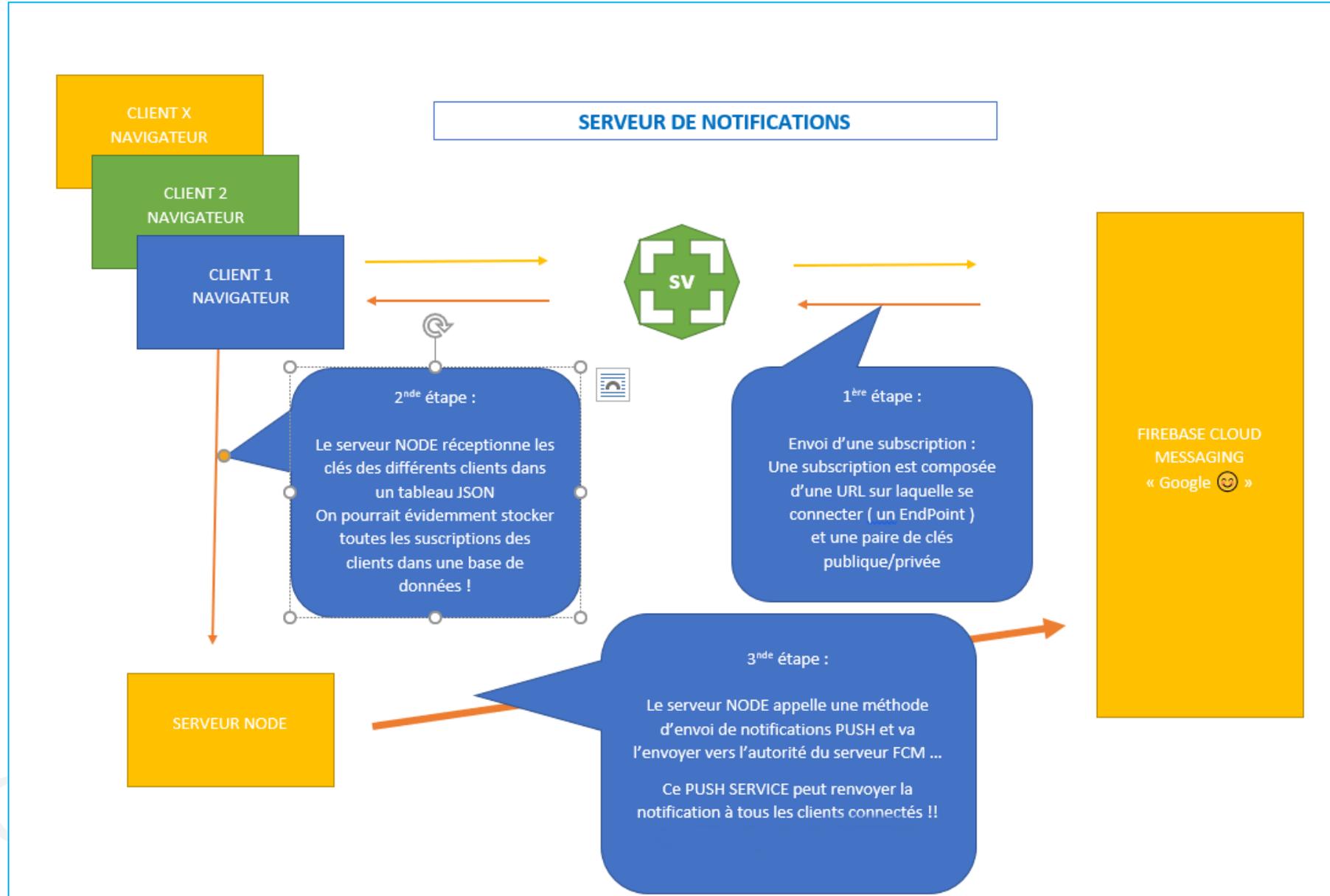
```
const showNotesFromStorage = () => {  
  for (let i=0; i < localStorage.length ; i++) {  
    console.log(localStorage.key(i) ,localStorage.getItem(i) );  
    const noteStorage = localStorage.getItem(i);  
  }  
  // For OF  
  // https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Global\_Objects/Object/entries  
  for (const [key, value] of Object.entries(localStorage)) {  
    console.log(key,value);  
    addDom(JSON.parse(value));  
  }  
};
```

# PWA – Background-Periodic-Sync

```
export const sync = () => {  
  
  console.clear();  
  console.log('---SYNC---');  
  
  //demander les permissions  
  navigator.permissions.query(  
    {  
      name: 'background-sync'  
    }  
  ).then(  
    (e) => {  
      const statusQueryPerm = e.state;  
      console.warn(statusQueryPerm);  
      if (statusQueryPerm==='granted') {  
        console.log(`---> Background -sync est autorisé`);  
      }  
    }  
  );  
  
  // -----  
  navigator.serviceWorker.ready  
  .then(  
    (registration) => {  
      registration.sync.register('tag-maj-cache');  
    }  
  );  
};
```

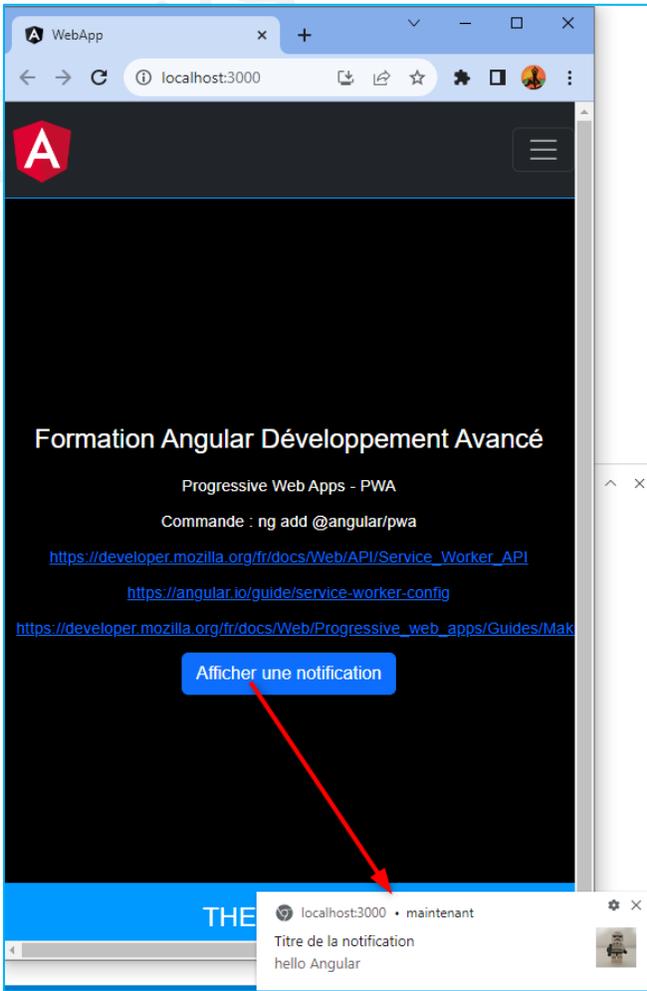
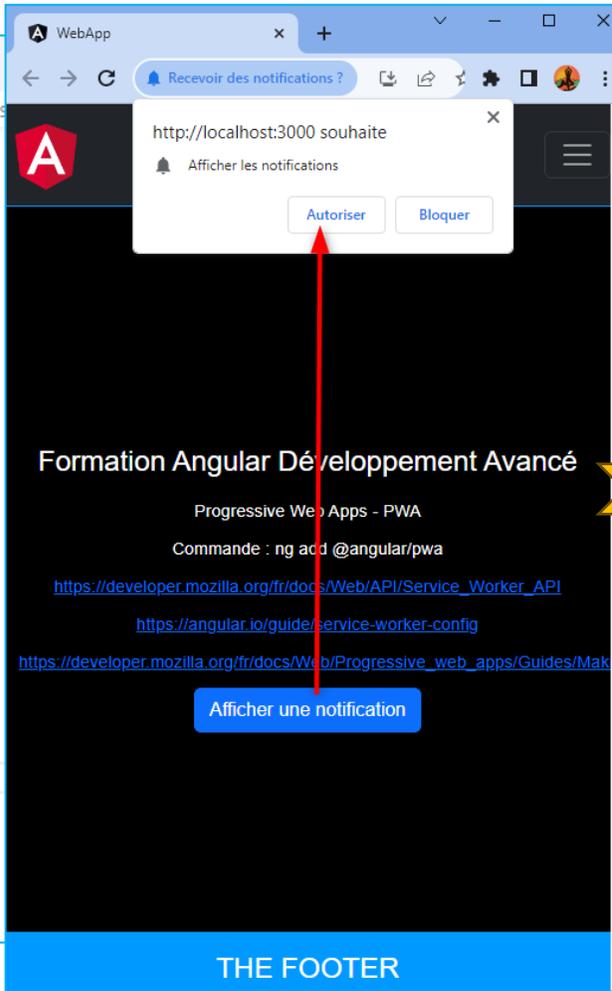
```
1 export const periodicSync = () => {  
2  
3   console.clear();  
4   console.log('---Periodic SYNC---');  
5  
6   //demander les permissions  
7   navigator.permissions.query(  
8     {  
9       name: 'periodic-background-sync'  
10    }  
11  ).then(  
12    (e) => {  
13      const statusQueryPerm = e.state;  
14      console.warn(statusQueryPerm);  
15      if (statusQueryPerm==='granted') {  
16        console.log(`---> Periodic Background sync est autorisé`);  
17      }  
18    }  
19  );  
20 // -----  
21 navigator.serviceWorker.ready  
22 .then(  
23   async registration => {  
24     await registration.periodicSync.register(  
25       'tag-periodic-sync',  
26       {  
27         // minInterval: 24 * 60 * 60 * 1000  
28         // 24 heures => daiy news  
29         // exprimé en ms  
30         minInterval : 5  
31       }  
32     );  
33   }  
34 };
```

# PWA – les Progressive Web Apps



# PWA – Notification

```
body.component.ts x
TD01-pwa > src > app > webApp > root > accueil > body > body.component.ts > BodyComponent >
10
19
20 // ----- notif simple -----
21 public showNotification = () => {
22
23   if ('Notification' in window) {
24     console.log('Notifications autorisées');
25
26     Notification.requestPermission(
27       (e) => {
28         console.log(e);
29         if (e==='granted') {
30
31           const options={
32             body:'hello Angular',
33             lang: 'fr-FR',
34             icon: 'assets/images/avatars/avatar1.jpg',
35             vibrate: [200, 100, 200]
36           };
37
38           const notif = new Notification('Titre de la notification',
39             options);
40         }
41       });
42
```



# PWA – Notification

```
index.html sw.js  
exo-8-notif-API-push > sw.js > [options] > vibrate  
93 console.log("Erreur : ", error);  
94 }  
95 }  
96 };  
97 // -----  
98 // -----  
99  
100 const options = {  
101   body: 'Hello Notification PUSH',  
102   icon: 'images/icons/icon-192x192.png',  
103   vibrate: [200]  
104 },  
105   actions: [{  
106     action: 'close',  
107     title: 'Good Bye Notification PUSH'  
108   }]  
109 }  
110  
111 self.addEventListener('push',  
112   event => {  
113     event.waitUntil(self.registration.showNotification(  
114       'Le titre : Notification PUSH', options  
115     ))  
116   })  
117 }  
118 }  
119 }  
120 }  
  
TERMINAL PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE  
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw  
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw  
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw  
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw  
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw
```

FORMATION  
PROGRESSIVE  
WEB APPS

HOME PAGE LA SAGA STAR WARS

L'API PUSH  
NOTIFICATION

Application en mode Connectée !  
Le type de connexion est : 4g et le débit estimé est : 3.35 MB/S  
L'Api notification permet d'envoyer des messages depuis le serveur.  
Exemple: MAJ de l'appli !  
L'API PUSH se configure dans le service worker car celui est toujours actif donc il peut recevoir à tout moment des notifications !  
Pour rappel, le service worker est toujours actif même quand l'appli est fermée. Alors que la simple notification vue précédemment a besoin que l'appli soit ON pour être reçue.

Service Workers  
Update on reload  
http://127.0.0.1:8080/  
Source sw.js  
Received 07/11/2020 à 08:47:02  
Status #2663 activated and is running stop  
Clients http://127.0.0.1:8080/index.html focus  
Push Test push message from Dev! Push  
Sync test-tag-from-devtools Sync  
Periodic Sync test-tag-from-devtools Periodic Sync

127.0.0.1:8080 • maintenant  
Le titre : Notification PUSH  
Hello Notification PUSH  
GOOD BYE NOTIFICATION PUSH

# NGRX – La notion de STORE



aNGular – Redux - rXjs

Lazy Loading)

↳ Rxjs & Observables

↕ Firestore & RealTimeDB

⬆️ Ngrx & Store

✉️ Formulaires Angular

🔍 Unit Tests

⚙️ TP & Exos

Un **store** représente l'état global de l'application en lecture seule  
Une source unique de vérité ne pouvant pas être modifiée  
Un objet javascript comme une banque de données, une base de données mais dans le front !

Commandes d'intégration :

```
ng add @ngrx/store
```

```
ng add @ngrx/effects
```

```
devTools : ng add @ngrx/store-devtools
```

Extension Chrome - firefox : <https://chrome.google.com/webstore/detail/redux-devtools/lmhkpbekcpmknkioeibfkpmmfiblj?hl=fr>

Charger les films

Composant Store #2

# Installation

```
package.json > ...
18   "@angular/animations": "^16.1.0",
19   "@angular/cdk": "^16.2.8",
20   "@angular/common": "^16.1.0",
21   "@angular/compiler": "^16.1.0",
22   "@angular/core": "^16.1.0",
23   "@angular/fire": "^7.6.1",
24   "@angular/forms": "^16.1.0",
25   "@angular/material": "^16.2.8",
26   "@angular/platform-browser": "^16.1.0",
27   "@angular/platform-browser-dynamic": "^16.1.0",
28   "@angular/router": "^16.1.0",
29   "@ngrx/effects": "^16.3.0",
30   "@ngrx/store": "^16.3.0",
31   "@ngrx/store-devtools": "^16.3.0",
32   "bootstrap": "^5.3.0",
33   "bootstrap-icons": "^1.10.5",
34   "rxfire": "^6.0.3",
35   "rxjs": "~7.8.0",
36   "tslib": "^2.3.0",
37   "zone.js": "~0.13.0"
38 },
39 "devDependencies": {
40   "@angular-devkit/build-angular": "^16.1.3",
41   "@angular/cli": "~16.1.3",
42   "@angular/compiler-cli": "^16.1.0",
43   "@angular/localize": "^16.1.3",
44   "@types/jasmine": "~4.3.0",
45   "jasmine-core": "~4.6.0",

```

```
src > app > app.module.ts > AppModule
9   import { StoreDevtoolsModule } from '@ngrx/store-devtools';
10
11  // import { rootReducer } from './webApp/formation/ngrx-store/reducers/rootReducer';
12  // import { metaReducersX } from './webApp/formation/ngrx-store/reducers/metaReducers';
13  // import { appEffects } from './webApp/formation/ngrx-store/effects/appEffects';
14
15  @NgModule({
16    declarations: [AppComponent],
17    imports: [BrowserModule, AppRoutingModule, AccueilModule, BrowserModule,
18             // NGRx - Store
19             StoreModule.forRoot(
20               {
21                 // rootReducer: formate le store (modifie)
22                 // nom à notre state (root): reducer principal
23                 // root: rootReducer | STATE_NAME: rootReducer
24               },
25               { // metaReducer
26                 // metaReducers: metaReducersX
27               }
28             ),
29             EffectsModule.forRoot([
30               // appEffects
31             ]),
32             StoreDevtoolsModule.instrument({ maxAge: 25, logOnly: !isDevMode() })
33          ],
34    providers: [],
35    bootstrap: [AppComponent]
36  })

```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE **TERMINAL** PORTS

```

✓ Found compatible package version: @ngrx/effects@16.3.0.
✓ Package information loaded.

The package @ngrx/effects@16.3.0 will be installed and executed.
Would you like to proceed? Yes
✓ Packages successfully installed.
UPDATE src/app/app.module.ts (750 bytes)
UPDATE package.json (1735 bytes)
✓ Packages installed successfully.
● PS C:\Users\Michel\Desktop\ANY\webApp> ng add @ngrx/store-devtools
i Using package manager: npm
✓ Found compatible package version: @ngrx/store-devtools@16.3.0.
✓ Package information loaded.

```

# NGRX pourquoi ?

Lorsqu'il y a beaucoup d'interactions entre composants et composants-enfants,  
Lorsque le projet devient très conséquent en termes de développement,  
Lorsque le projet grandit et se construit au fur et à mesure ...

Le projet est de plus en plus difficile à maintenir.

On peut rencontrer des effets de bords non désirés.

La structure « complexe » du projet rend de plus en plus difficile les opérations d'amélioration et d'enrichissement.

**BEST** NOUVELLE EDITION

## Formation : Angular, maîtriser le Framework Front-End de Google

concepts de développement

★★★★☆ 4,5 / 5

Angular est le framework javascript de référence de Google. Il utilise tous les standards du Web. Il offre des performances accrues avec une conception modulaire adaptée à la mobilité ainsi qu'une amélioration de la productivité de vos équipes de développement. Angular bénéficie immédiatement d'un écosystème riche et d'une communauté toujours plus grande.

### Objectifs pédagogiques

À l'issue de la formation, le participant sera en mesure de :

- Organiser, modulariser et tester ses développements JavaScript

Inter Intra Sur mesure

Cours pratique en présentiel ou en classe à distance

Réf. AGU

🕒 4j - 28h00 *Pauses-café et déjeuners offerts*

Dates, lieux et Inscription

[Nous contacter](#)

> Formations > Technologies numériques > Technologies Web > Développement Front-End > Formation Angular, développement avancé

**BEST**

## Formation : Angular, développement avancé

★★★★☆ 4,3 / 5

Vous découvrirez en profondeur les bonnes pratiques de développement des applications Angular avec les dernières version du framework Angular et le moteur de rendu optimisé Ivy. Vous apprendrez à maîtriser le FormBuilder pour des formulaires réactifs ainsi que la génération de tests unitaires.

### Objectifs pédagogiques

À l'issue de la formation, le participant sera en mesure de :

- Savoir utiliser les décorateurs Angular
- Architecturer les applications web complexes
- Intégrer les outils de documentation et les tests unitaires
- Développer et intégrer des bibliothèques de composants

Inter Intra Sur mesure

Cours pratique en présentiel ou en classe à distance

Réf. ANY

🕒 3j - 21 *Pauses-café et déjeuners offerts*

Dates, lieux et Inscription

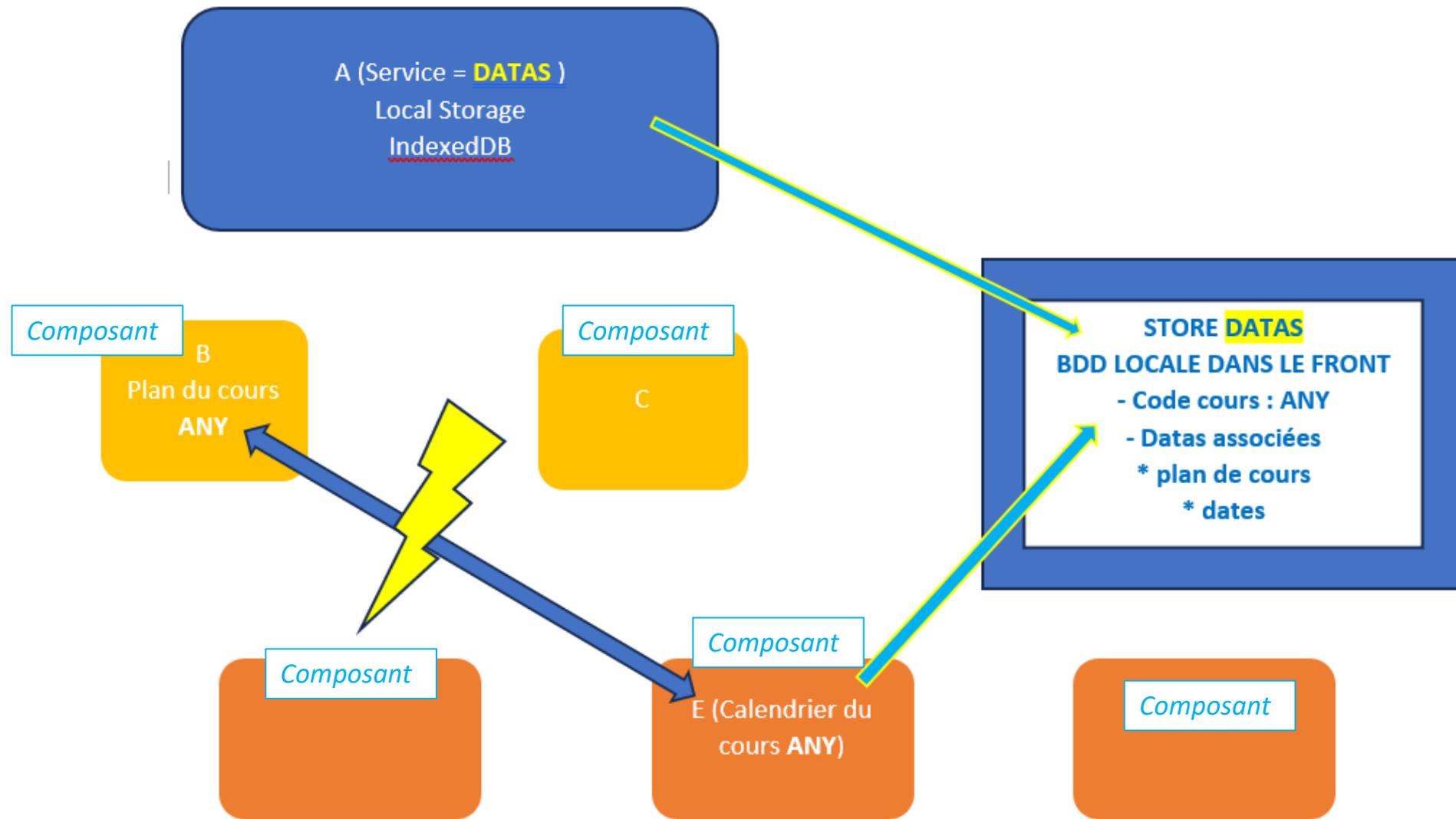
[Nous contacter](#)

# NGRX la solution

Au lieu de traverser tout l'héritage des Input et des Output des composants parents-enfants, le composant peut s'inscrire auprès du Store et dispatcher une Action.

C'est-à-dire demander quelque chose ou demander à faire quelque chose.

L'accès est direct sans passer par la structure parfois « trop rigide » des web-components imbriqués.

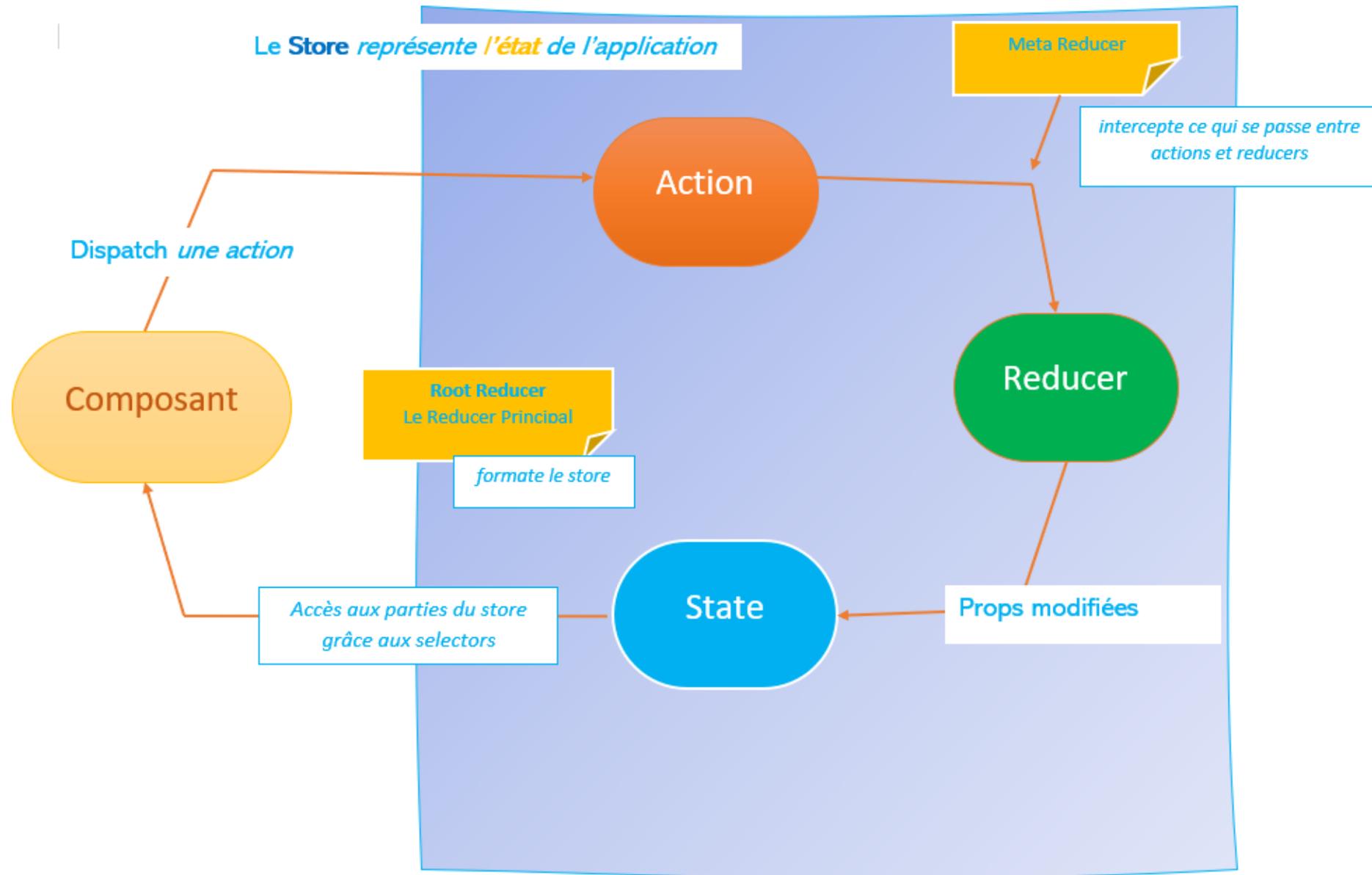


1 Composant s'inscrit auprès du STORE !

Dispatcher des actions (demande à faire qq chose avec les datas enregistrées dans le store)

# NGRX Principe de fonctionnement

- 1- Le composant dispatche une action
- 2- Celle-ci doit être listée dans le store
- 3- Un reducer réceptionne l'action « dispatchée » et se charge de son exécution
- 4- Le reducer modifie le state de l'application
- 5- Le composant dispatchant l'action demande à accéder à une partie du store grâce aux selectors
- 6- Un nouveau rendu HTML est proposé à tous les composants ayant accès à ce selector !



# NGRX SideEffect - BackEnd

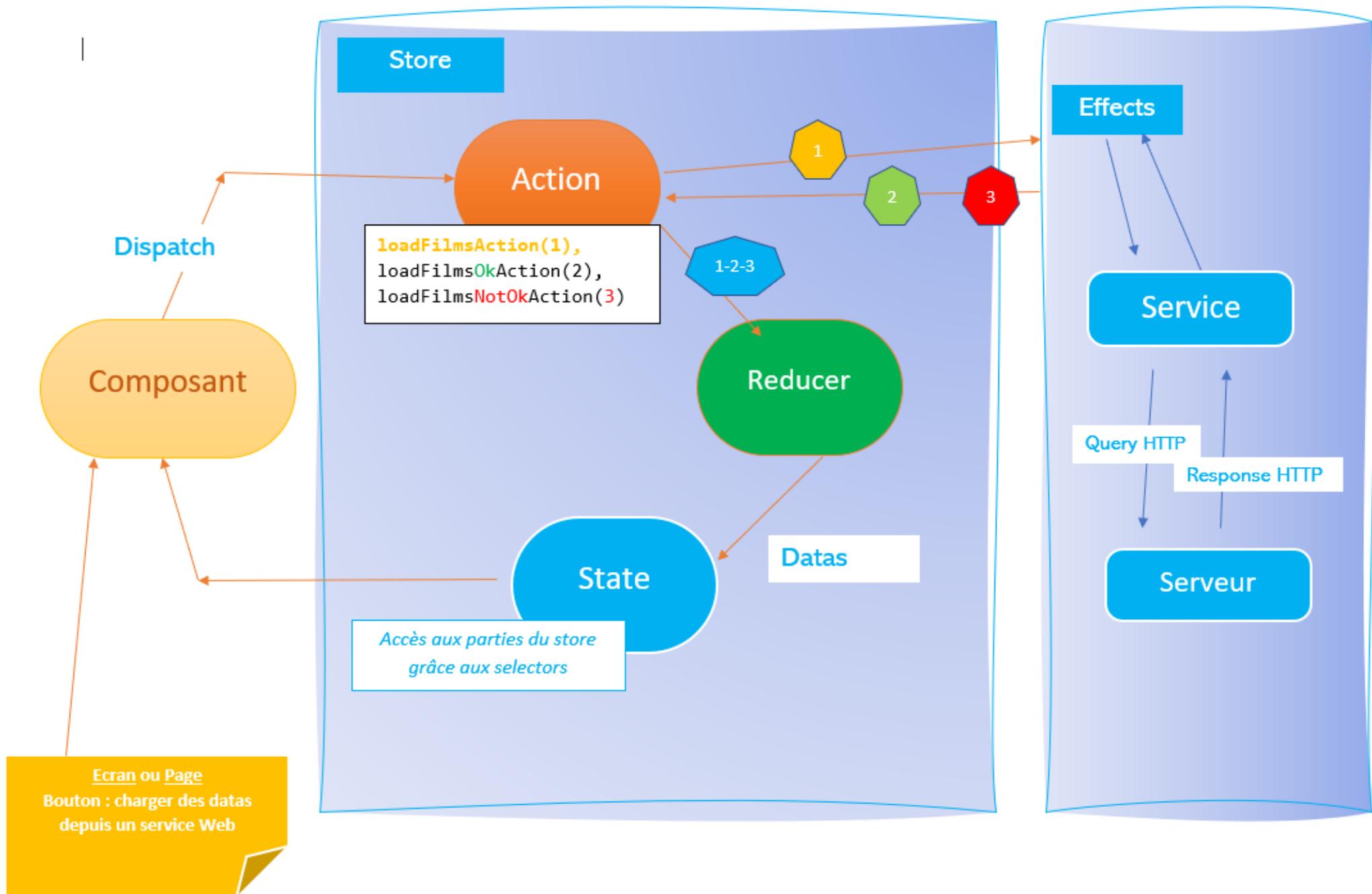
Les [sideEffects](#) permettent de connecter le back via des services web à notre application Angular-STORE.

Le schéma peut « paraître » complexe au 1<sup>er</sup> abord mais il est logique dans la conception de la « fonction pure » du Reducer !

Pour rappel, une fonction pure ne peut retourner que les paramètres reçus en appel (en entrée)

```
<!--  
  fct impure  
  let c=2  
function X (a,b) {  
  return a+b + c  
}  
  
fct pure  
function X (a,b) {  
  return a+b  
} -->
```

# NGRX



# Root Reducer

```
root-reducers.ts x
Formation-ANY-07-2023 > TP07-NGRX > src > app > webApp > formation > ngx-store > reducers > root-reducers.ts > [e] root > TP07-NGRX > src > app > webApp > formation > ngx-store > reducers > root-reducers.ts
1 import { createReducer, on } from "@ngrx/store";
2 import { changenameAction, initAction, loadFilmsAction, loadFilmsNotOkAction } from "src/app/sharedModels/models/class/actions";
3 import { Films } from "src/app/sharedModels/models/class/films";
4
5 // const INITIAL_STATE = {}; // à l'init => en prod
6 // ----- OPTIMISATION -----
7 // export const STATE_NAME = 'appli';
8 export interface RootState {
9   appName: string;
10  actor: {
11    name: string;
12    isJedi: boolean;
13  }
14  film?: Films;
15  films?: Films[],
16  loaded?: boolean
17 }
18 // ----- OPTIMISATION -----
19 const INITIAL_STATE: RootState = {
20   appName: 'Formation Angular NGRX',
21   actor: {
22     name: '',
23     isJedi: false
24   }
25 };
26 export const rootReducer = createReducer(
27   // state initial
28   INITIAL_STATE,
29   // ons => 'on' sur une action
30   // définir les actions invoquées
31   on(
32     initAction,
33     // tâche ou le job que va faire le reducer ...
34     // sur les props du store représenté par un state
35     (state) => {
36       // le reducer est capable de prendre qu'une partie du store(state)
37       return {
38         ...state, // ... spread operators ES2015 (déstructure notre state)
39         actor: {
40           ...state.actor, // copie de la valeur de actor
41           isJedi: true // param modifié
42         }
43       };
44     },
45     // autre on
46     on(
47       changenameAction,
48       (state, props) => {
49         return {
50           ...state,
51           actor: {
52             ...state.actor,
53             name: props.paramNameActionCOMP
54           }
55         };
56       }
57     ),
58     // autre on
59     on(loadFilmsAction,
60       (state) => {
61         return {
62           ...state,
63           loaded: false
64         };
65       }
66     ),
67     // autre on
68     on(loadFilmsOkAction,
69       (state, props) => {
70         return {
71           ...state,
72           films: props.films,
73           loaded: true
74         };
75       }
76     ),
77     // autre on
78     on(loadFilmsNotOkAction,
79       (state, props) => {
80         return {
81           ...state,
82           loaded: true,
83           KO: true,
84           erreurDesc: props.err
85         };
86       }
87     )
88   );
89 );
```

# Meta Reducer

```
root-reducers.ts selectors.ts actions.ts effects.ts meta-reducers.ts X
Formation-ANY-07-2023 > TP07-NGRX > src > app > webApp > formation > ngrx-store > reducers > meta-reducers.ts > [log] > <fun
1 // affiche en console les différentes étapes
2 // qui se déroulent dans le store entre les actions et les reducers...
3
4 import { ActionReducer, MetaReducer } from "@ngrx/store"
5
6 // actionReducer représente la combinaison de l'action et du reducer
7 // nous donne le résultat de ce qui se passe entre l'action et le reducer
8
9 const log = (paramReducer: ActionReducer<any>) => {
10
11   return(state:any, action:any) => {
12
13     const currentState = paramReducer(state, action);
14
15     console.groupCollapsed(action.type);
16
17     console.log('Etat précédent : ', state);
18     console.warn('Action : ', action);
19     console.log('Etat suivant : ', currentState);
20
21     console.groupEnd();
22
23     return currentState;
24   }
25 }
26 // ----- export le metaReducer -----
27 export const metaReducersX: MetaReducer[] = [log];
```

# selectors

```
root-reducers.ts selectors.ts X
tion-ANY-07-2023 > TP07-NGRX > src > app > webApp > formation > ngrx-store > selectors > sel
1 // on va créer des parties du store qu'on veut select
2 // pour le smettre à dispo des composants
3
4 import { createSelector } from "@ngrx/store";
5
6 const selectRootState = (state:any) => {
7   return state.root;
8 }
9
10 export const selectorGetActor = createSelector(
11   selectRootState,
12   (state:any) => state.actor
13 )
14
15 export const selectorGetFilms = createSelector(
16   // on prend une partie du state
17   selectRootState, // a partir de quoi on va filtrer ce que l
18   (state:any) => state.films
19 );
20 export const selectorGetLoaded = createSelector(
21   selectRootState,
22   (state:any) => state.loaded
23 );
24 export const selectorGetKO = createSelector(
25   selectRootState,
26   (state: any) => state.KO
27 );
```

# selectors

```
root-reducers.ts selectors.ts X
tion-ANY-07-2023 > TP07-NGRX > src > app > webApp > formation > ngrx-store > selectors > sel
1 // on va créer des parties du store qu'on veut select
2 // pour le smettre à dispo des composants
3
4 import { createSelector } from "@ngrx/store";
5
6 const selectRootState = (state:any) => {
7   return state.root;
8 }
9
10 export const selectorGetActor = createSelector(
11   selectRootState,
12   (state:any) => state.actor
13 )
14
15 export const selectorGetFilms = createSelector(
16   // on prend une partie du state
17   selectRootState, // a partir de quoi on va filtrer ce que l
18   (state:any) => state.films
19 );
20 export const selectorGetLoaded = createSelector(
21   selectRootState,
22   (state:any) => state.loaded
23 );
24 export const selectorGetKO = createSelector(
25   selectRootState,
26   (state: any) => state.KO
27 );
```

# actions

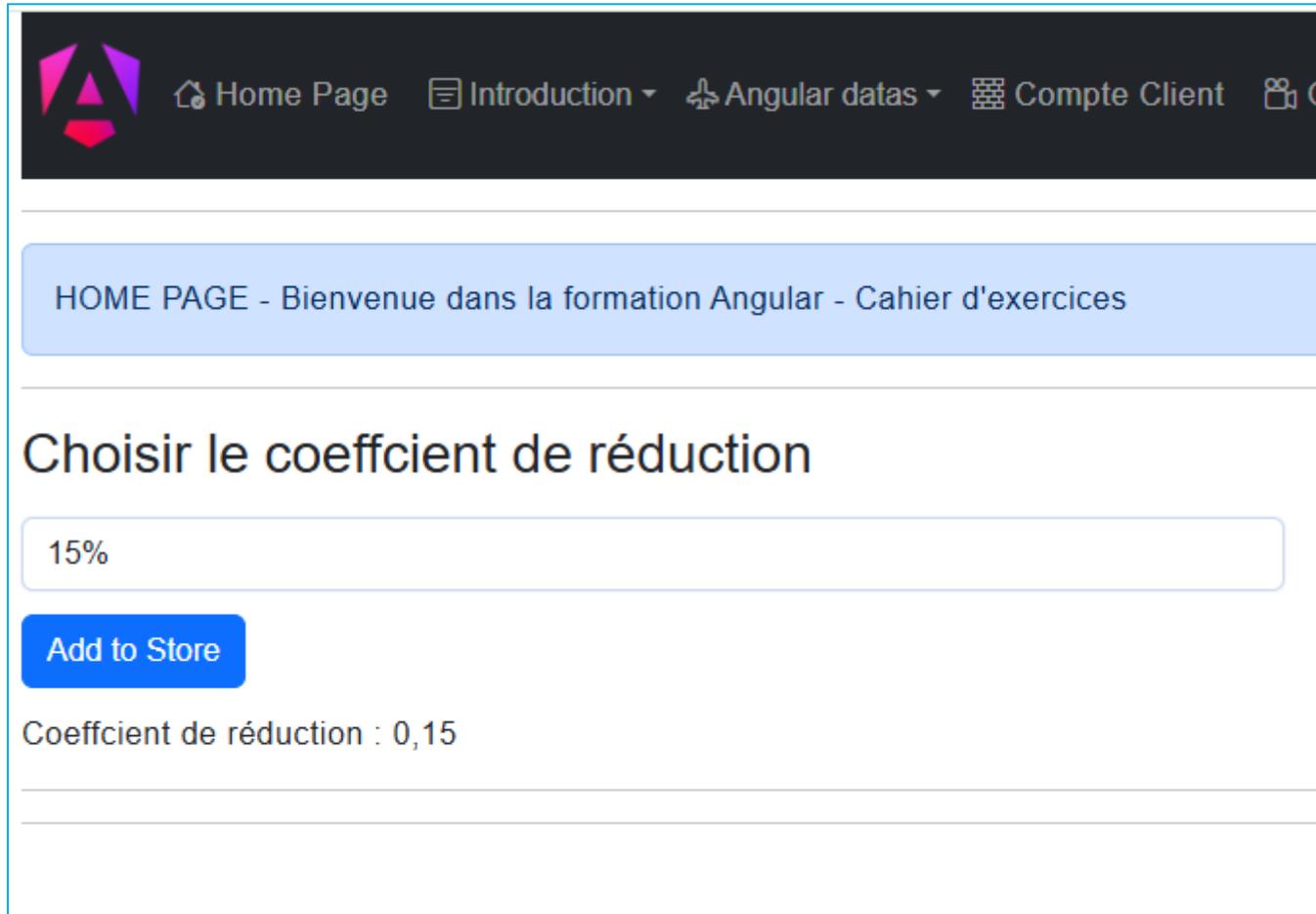
```
root-reducers.ts selectors.ts actions.ts X
Formation-ANY-07-2023 > TP07-NGRX > src > app > webApp > formation > ngrx-store > actions > act
1 import { createAction, props } from "@ngrx/store";
2 import { Films } from "src/app/sharedModels/models/class/films";
3
4 export const createAction = createAction(
5   // nom de l'action
6   '[ROOT] Init Action'
7 );
8
9 export const changenameAction = createAction(
10  // nom de l'action
11  '[ROOT] Change Name Actor Action',
12  // props à faire ajouter par le reducer dans le state
13  props<{paramNameActionCOMP:string}>()
14 );
15
16 export const loadFilmsAction = createAction(
17  // nom de l'action
18  '[FILMS] Ask Load Films Action'
19 );
20
21 export const loadFilmsOkAction = createAction(
22  '[FILMS DATAS] Load films OK',
23  props<{films:Films[]}>()
24 );
25
26 export const loadFilmsNotOkAction = createAction(
27  '[FILMS DATAS] Load films Not OK',
28  props<{err:any}>()
29 );
```

```
root-reducers.ts selectors.ts actions.ts effects.ts x
Formation-ANY-07-2023 > TP07-NGRX > src > app > webApp > formation > ngrx-store > effects > effects.ts > appEffects > loadFilms$
6 import * as compActions from '../actions/actions';
7 import { Films } from "src/app/sharedModels/models/class/films";
8
9 @Injectable()
10
11 export class appEffects {
12
13
14   constructor(private _serviceEffects: NgrxEffectsService, private _actions: Actions) { }
15
16   // création d'un Observable
17
18   loadFilms$: Observable<any> = createEffect(
19     () => {
20       // détection des actions invoquées
21       return this._actions.pipe(
22         tap(
23           (action) => {
24             console.log('***** Actions : ', action);
25           }
26         ),
27         ofType(
28           // ofType peut filtrer la liste de ce qu'on lui passe
29           compActions.loadFilmsAction
30         ),
31         mergeMap(
32           action => this._serviceEffects.getFilms$()
33           // on récupère les datas : films
34           .pipe(
35             map(
36               (films:Films[]) => {
37                 return compActions.loadFilmsOkAction({films})
38                 // c'est l'effect qui passe des datas à l'action
39               }
40             ),
41             catchError(
42               (err) => {
43                 // console.log(err);
44                 return of(compActions.loadFilmsNotOkAction({err}));
45               }
46             )
47           )
48         )
49       )
50     }
51 }
```

Copyright

SYS

# NGRX TP

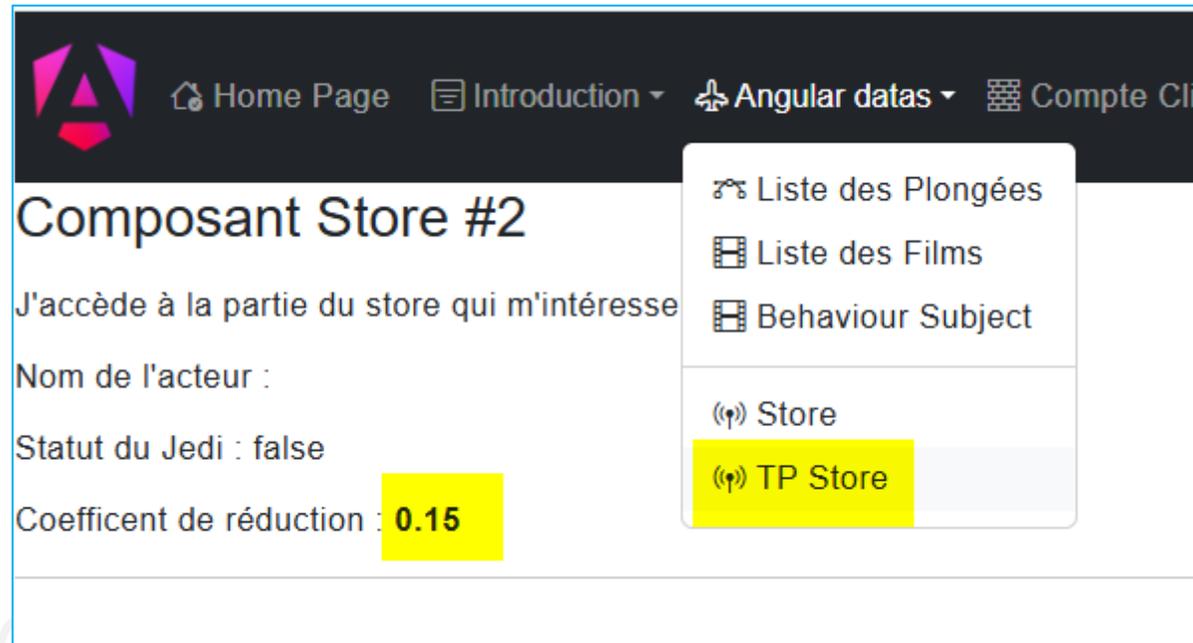


The screenshot shows a web application interface with a dark navigation bar at the top containing a logo and menu items: Home Page, Introduction, Angular datas, and Compte Client. Below the navigation bar is a light blue header with the text "HOME PAGE - Bienvenue dans la formation Angular - Cahier d'exercices". The main content area features a heading "Choisir le coefficient de réduction" followed by a text input field containing "15%". Below the input field is a blue button labeled "Add to Store". At the bottom of the main content area, the text "Coefficient de réduction : 0,15" is displayed.

Ajouter sur la home-page un composant qui va permettre de valider un coefficient de réduction dans le store.

Il faudra ensuite le réexploiter et l'afficher, soit dans la même vue dans un autre composant soit dans le composant TP-Store.

# NGRX TP



The screenshot shows a web application interface. At the top, there is a navigation bar with a logo and links for 'Home Page', 'Introduction', 'Angular datas', and 'Compte Cli'. Below the navigation bar, the main content area is titled 'Composant Store #2'. The text 'J'accède à la partie du store qui m'intéresse' is displayed. Below this, there are three data points: 'Nom de l'acteur :', 'Statut du Jedi : false', and 'Coefficient de réduction : 0.15'. A dropdown menu is open on the right side, listing 'Liste des Plongées', 'Liste des Films', and 'Behaviour Subject' in the top section, and 'Store' and 'TP Store' in the bottom section. The 'TP Store' option is highlighted in yellow.

Home Page Introduction Angular datas Compte Cli

## Composant Store #2

J'accède à la partie du store qui m'intéresse

Nom de l'acteur :

Statut du Jedi : false

Coefficient de réduction : **0.15**

- Liste des Plongées
- Liste des Films
- Behaviour Subject
- Store
- TP Store**

# NGRX TP

Choisir le coefficient de réduction

Add to Store

Coefficient de réduction : 0,15

THE FOOTER - Formation Angular - Michel Bocciolesi - Orsys Formation

The screenshot shows the Redux DevTools interface. The top bar includes navigation icons and menu items: Éléments, Console, Sources, Réseau, Performances, Mémoire, Appli, Sécurité, Lighthouse, Enregistreur, and Informations sur les performances. Below this, there are tabs for 'Actions' and 'Settings'. The 'Actions' tab is active, showing a list of actions with a search filter 'filter...'. The actions list includes:

Action	Time
@ngrx/store/init	10:44:18.24
@ngrx/effects/init	+00:00.02
[TP] Coeff Action	+00:02.11
[TP] Coeff Action	+00:02.26

The right panel shows the 'State' tab with a tree view. The state structure is:

```
state -- root --> {
  appName,
  name,
  isJedi,
  coe: "0.15"
}
```

The 'coe' property is highlighted in yellow in the original image.

# NGRX TP

```
<div>
  <h5>Choisir le coefficient de réduction</h5>
  <select class="form-control" style="width:50%" [(ngModel)]="coeff" (change)="addToStore()">
    <option value="">Choisir le coefficient de réduction</option>
    <option value="0.1">10%</option>
    <option value="0.15">15%</option>
    <option value="0.2">20%</option>
  </select>
  <p></p>

  <button class="btn btn-primary" (click)="addToStore()">Add to Store</button>
  <p></p>
  <p>Coefficient de réduction : {{ coeff | number:'1.0-2' }}</p>
</div>
```

# NGRX TP

```
//-----  
public coeff: number = 0;  
_store = inject(Store);  
  
public addToStore() {  
  this._store.dispatch(  
    coeffAction({  
      coeffStore: this.coeff  
    })  
  );  
}  
//-----
```

```
// TP  
  
export const coeffAction = createAction(  
  // nom de l'action  
  '[TP] Coeff Action',  
  // props à faire ajouter par le reducer dans le state  
  props<{coeffStore:number}>()  
);
```

```
// TP  
  
export const selectorGetCoeff = createSelector(  
  selectRootState,  
  (state:any) => state.coeff  
);
```

```
,  
// TP COEFF  
// un autre ON sur une autre action  
on(  
  coeffAction,  
  (state, props) => {  
    return {  
      ...state,  
      coeff: props.coeffStore  
    }  
  }  
)  
)
```

# NGRX TP

comp-store2.component.html ×

src > app > webApp > ngrx-store > components > comp-store2 > comp-store2.component.html > h5

Go to component

```
1 <h5>Composant Store #2</h5>
2 <p>J'accède à la partie du store qui m'intéresse ...</p>
3 <p>Nom de l'acteur : {{(actor$ | async).name}} </p>
4 <p>Statut du Jedi : {{(actor$ | async).isJedi}}</p>
5 <p>Coefficient de réduction : <strong> {{coeff$ | async }}</strong></p>
6
```

comp-store2.component.ts ×

src > app > webApp > ngrx-store > components > comp-store2 > comp-store2.component

```
14 export class CompStore2Component {
16     public actor$: Observable<any> = new Observable()
17     public coeff$: Observable<number> = new Observable();
18     constructor(private _store: Store) { }
19
20     ngOnInit() {
21         // -----Avec les selectors :-) -----
22         this.actor$ = this._store
23         .pipe(
24             select(selectorGetActor)
25         );
26         this.coeff$ = this._store.pipe(
27             select(selectorGetCoeff)
28         )

```

Copyright



# Formation ANGULAR

A bientôt.