



Formation ANGULAR

Animé par Michel BOCCIOLESI

Table des matières

- Introduction :
 - Framework Progressif ou Orienté
 - Installation - configuration
 - Standalone Components VS Modules
- Notion de Réactivité Angular
Méthode « Value Based » (Zone.js)
Méthode Observable Based
Le signal NG16-17-18
- Les Web-components :
TS – HTML – CSS
Binding – décorateurs
composants parents-enfants
directives - pipes
- Les modules vs Standalone Component:
« structurels » - architecture de projet
« fonctionnels » - partie spécifique de l'application (compte-client, panier)
- Mise en place du projet « fil rouge » de la formation
Application des acquis
- Injection de dépendances (service Web ou autres)
- Les cycles de vie du composant
constructor – ngOnInit – la gestion des datas
- Le routage Angular « navigation et liens »
Concepts de base et avancés
- Communication entre composants parents et enfants
@Input() @Output()
- La programmation RXJS et les observables
- Les formulaires Angular
Reactive Forms vs Template
- Test Unitaires
Introduction à Karma et Jasmine
- Annexes :
 - Rappels EcmaScript 2015+
La révolution Javascript (rappels)
 - Le Framework Angular - Historique
Les versions marquantes 9 – 19

Framework Orienté ou Progressif



Copyright Michel BOCCIOLESI - ORSYS

Framework ou librairies ?

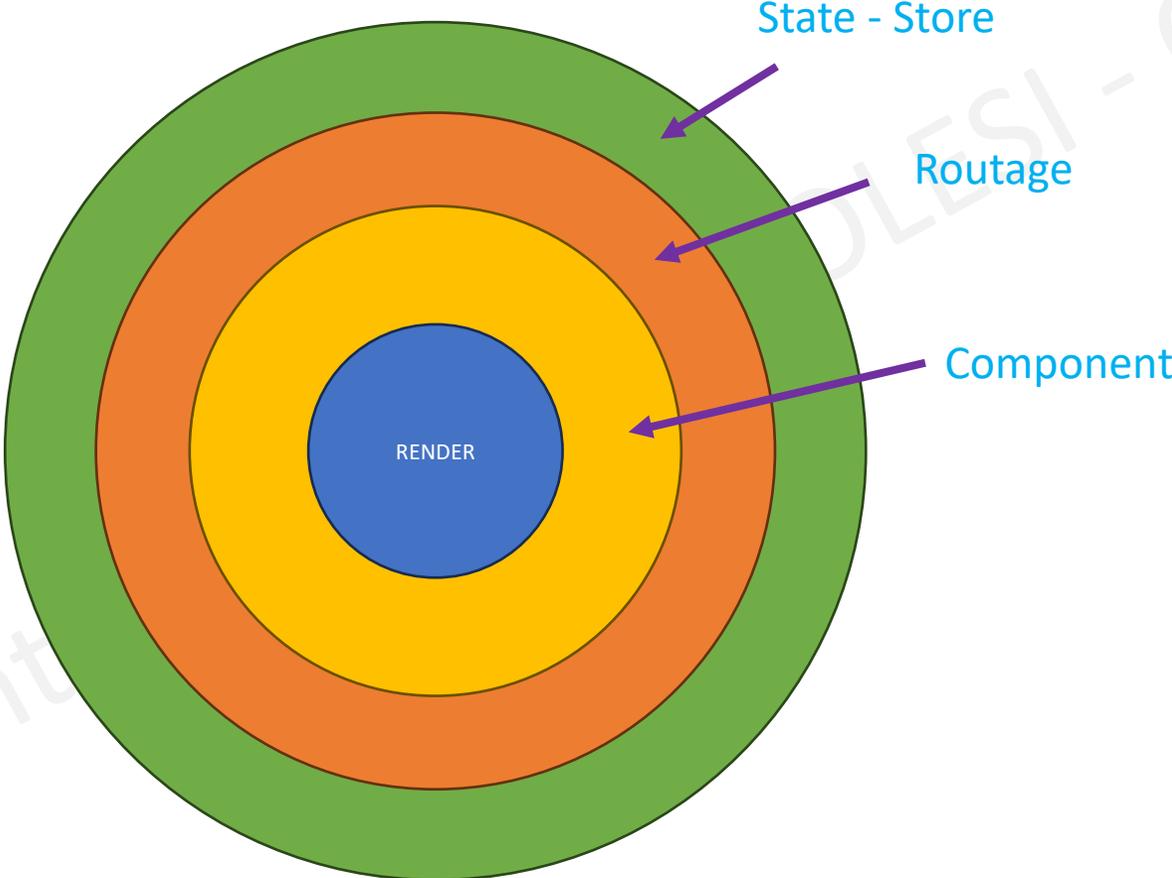
Dans le monde du développement Front, il existe plusieurs catégories de frameworks :

- [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** ¹ ». Il peut cependant être utilisé en tant que librairie. 🤔
Il possède ses propres outils de création [npm create vue](#) ou [Vite](#) ²

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

² 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^{er} 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

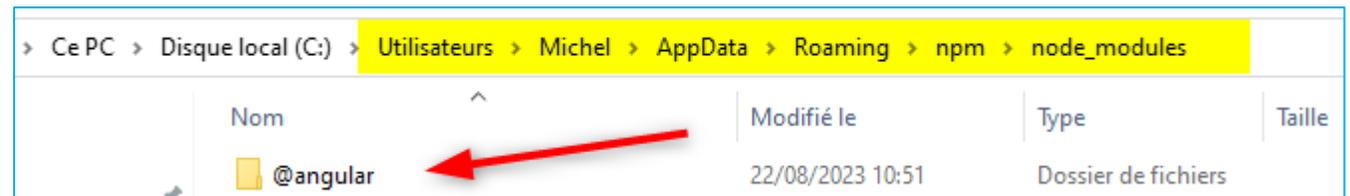
<https://nodejs.org>

<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^{ère} 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
```

- `npm install @angular/cli@12`
- `npx @angular/cli@14 new projet-ng-14`

```
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
```

```
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>
```

- 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.

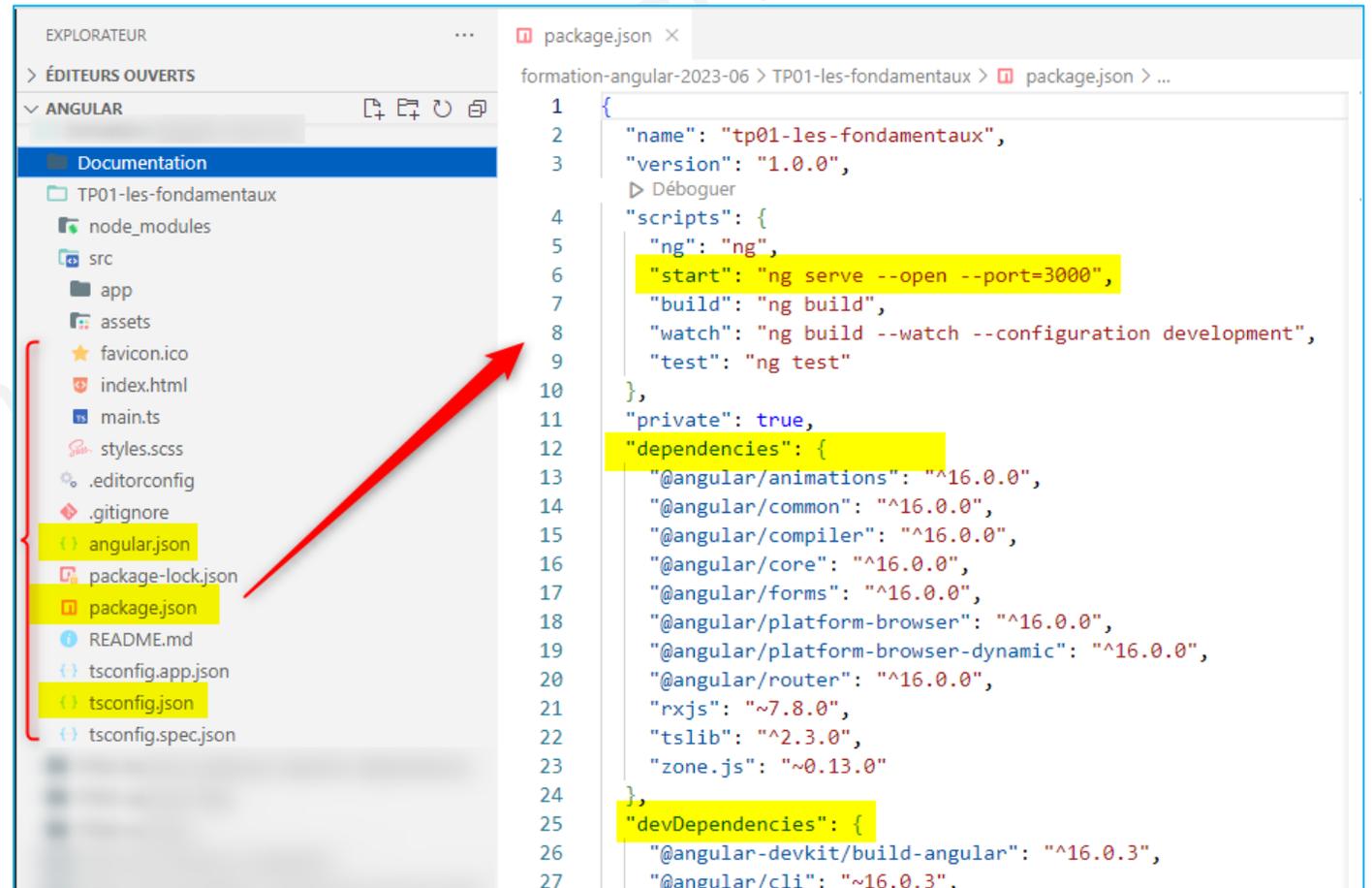
Il y a au moins 32k fichiers dans un simple projet Angular.

[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

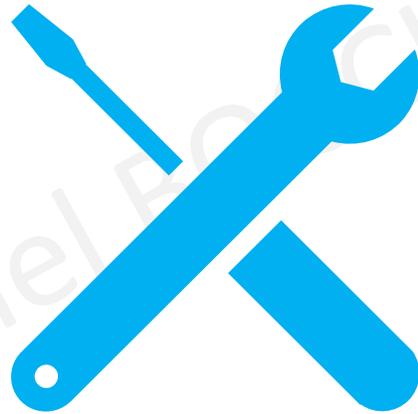
```
PS C:\Users\Formateur> npm init -y
Wrote to C:\Users\Formateur\package.json:

{
  "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
```



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- 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 Change¹** » ou « **Dirty Checking²** » 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

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.

1^{ER} PROJET ANGULAR



Copyright Michel ACCIOLESI - ORSYS

Prise en main de l'architecture d'un projet Angular avec Modules

```
angular.json > {} projects > {} TP01-les-fondamentaux > {} architect > {} build > {} options
9   "@schematics/angular:component": {
10     "style": "scss"
11   }
12 },
13 "root": "",
14 "sourceRoot": "src",
15 "prefix": "app",
16 "architect": {
17   "build": {
18     "builder": "@angular-devkit/build-angular:browser",
19     "options": {
20       "outputPath": "dist/tp01-les-fondamentaux",
21       "index": "src/index.html",
22       "main": "src/main.ts",
23       "polyfills": [
24         "zone.js"
25       ],
26       "scripts": [],
27       "styles": [
28         "src/styles.scss"
29       ],
30       "assets": [
31         "src/favicon.ico",
32         "src/assets"
33       ],
34       "vendorChunk": true,
35       "extractLicenses": false,
36       "buildOptimizer": true,
37       "fileReplacements": [
38         {
39           "replace": "src/environments/environment.ts",
40           "with": "src/environments/environment.prod.ts"
41         }
42       ],
43       "namedChunks": true
44     },
45     "configurations": {
46       "production": {
47         "fileReplacements": [
48           {
49             "replace": "src/environments/environment.ts",
50             "with": "src/environments/environment.prod.ts"
51           }
52         ],
53         "optimization": true,
54         "outputPath": "dist/tp01-les-fondamentaux",
55         "sourceMap": false,
56         "namedChunks": false,
57         "extractLicenses": true,
58         "vendorChunk": false
59       }
60     },
61     "defaultConfiguration": "production"
62   }
63 }

```

```
TP01-les-fondamentaux > src > main.ts
1 import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
2
3 // import LOCAL (import ES 2015)
4 import { AppModule } from './app/app.module';
5
6 // est le point d'entrée du projet
7 // charger ou bootstrapper le tout 1er module
8
9 platformBrowserDynamic().bootstrapModule(AppModule)
10 | // .catch(err => console.error(err));
11

```

```
TP01-les-fondamentaux > app > app.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6
7 // un décorateur transforme la class en objet NG (comp, module,
8 // service ...)
9 @NgModule({
10   declarations: [
11     // liste de tous les composants du module
12     AppComponent,
13     // PlanDeCoursComponent,
14     // DatesCoursComponent
15   ],
16   imports: [
17     // toutes les librairies nécessaires à ce module
18     BrowserModule,
19     AppRoutingModule
20   ],
21   providers: [
22     // services
23   ],
24   bootstrap: [
25     // quel est le 1er composant chargé ?
26     AppComponent
27   ]
28 })

```

```
TP01-les-fondamentaux > src > app > app.component.ts > ...
1 import { Component } from '@angular/core';
2
3 // un décorateur transforme la class en objet NG (comp, module,
4 @Component({
5   // props ou directives
6   selector: 'app-root',
7   templateUrl: './app.component.html',
8   styleUrls: ['./app.component.scss']
9 })
10
11 // export et class sont des mots ECMAScript 2015
12 export class AppComponent {
13
14   // propriétés
15   title = 'TP01-les-fondamentaux';
16 }
17

```

Copyright

Prise en main de l'architecture d'un projet Angular en standalone components

The image shows a Visual Studio Code editor window with the following structure:

- EXPLORATEUR (File Explorer):** Shows the project structure for 'NG19'. The 'src' folder is expanded, showing the 'app' subfolder. The 'app.config.ts' file is selected and highlighted in yellow.
- main.ts (Editor):** Contains the following code:

```
1 import { bootstrapApplication } from '@angular/platform-browser';
2 import { appConfig } from './app/app.config';
3 import { AppComponent } from './app/app.component';
4
5 bootstrapApplication(AppComponent, appConfig)
6   .catch((err) => console.error(err));
7
```
- app.config.ts (Editor):** Contains the following code:

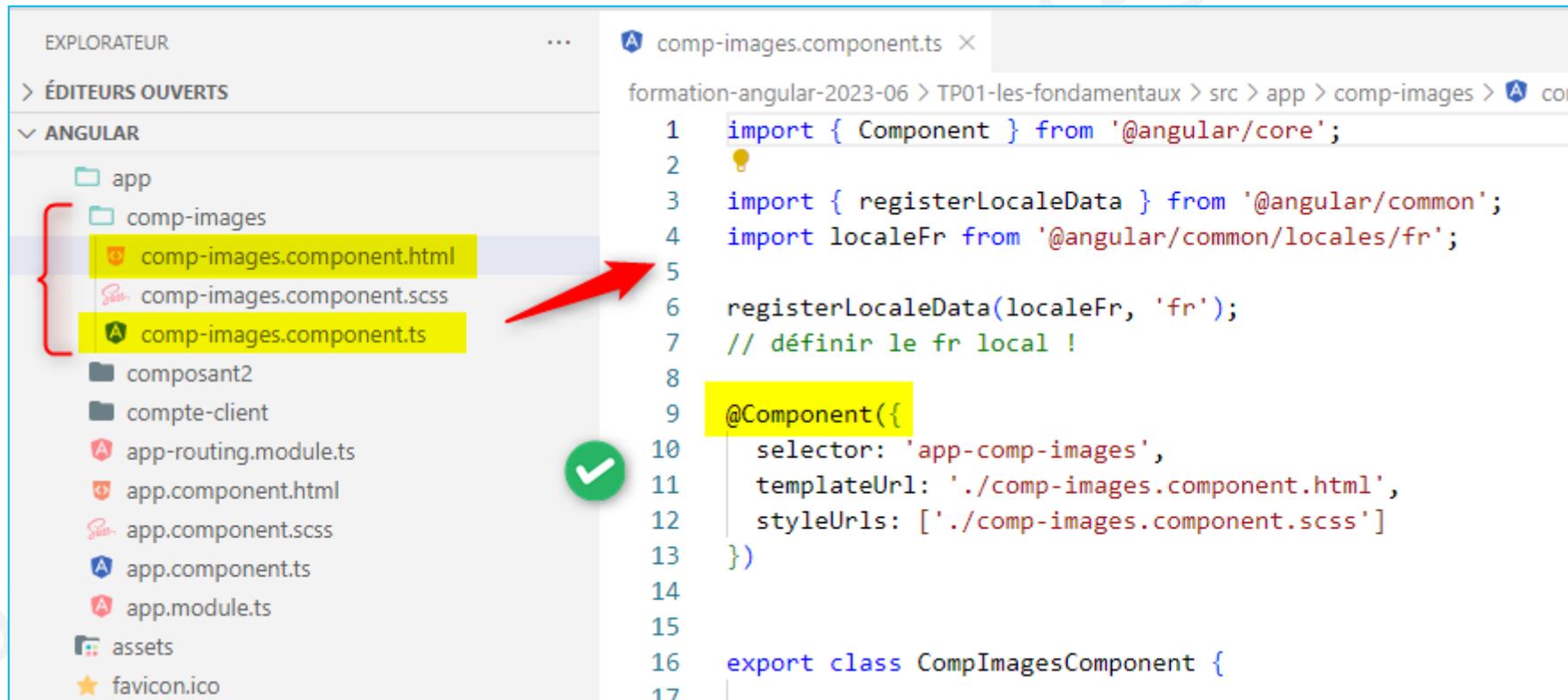
```
4 import { routes } from './app.routes';
5 import { provideClientHydration, withEventReplay } from '@angular/platform-browser';
6
7 export const appConfig: ApplicationConfig = {
8   providers: [
9     provideZoneChangeDetection({ eventCoalescing: true }),
10    provideRouter(routes),
11    provideClientHydration(withEventReplay())
12  ]
13};
```
- app.routes.ts (Editor):** Contains the following code:

```
1 import { Routes } from '@angular/router';
2
3 export const routes: Routes = [];
4
```

Intérêts du Framework Angular

Les web-components (composants web)

1 composant est constitué de 3 fichiers : TS, la vue HTML et le fichier de style CSS*



The screenshot shows an IDE interface with two main panes. On the left is the 'EXPLORATEUR' (File Explorer) showing a project structure under 'ANGULAR'. A folder named 'app' contains a sub-folder 'comp-images'. Inside 'comp-images', three files are listed: 'comp-images.component.html', 'comp-images.component.scss', and 'comp-images.component.ts'. A red bracket groups these three files, and a red arrow points from the 'comp-images.component.ts' file in the explorer to the code editor on the right. The code editor shows the content of 'comp-images.component.ts' with line numbers 1 through 17. The code includes imports for '@angular/core', '@angular/common', and '@angular/common/locales/fr'. It also shows the registration of locale data and the definition of the '@Component' decorator with properties for selector, templateUrl, and styleUrls. The component class 'CompImagesComponent' is partially visible at the bottom.

```
1 import { Component } from '@angular/core';
2
3 import { registerLocaleData } from '@angular/common';
4 import localeFr from '@angular/common/locales/fr';
5
6 registerLocaleData(localeFr, 'fr');
7 // définir le fr local !
8
9 @Component({
10   selector: 'app-comp-images',
11   templateUrl: './comp-images.component.html',
12   styleUrls: ['./comp-images.component.scss']
13 })
14
15
16 export class CompImagesComponent {
17
```

Intérêts du Framework Angular



Les web-components (composants web)

le composant web permet de travailler une partie de la page globale index.html tant au niveau des données (le TS) que de la vue HTML+CSS*

```
app.component.ts ×
formation-angular-2023-06 > TP01-les-fondamentaux > src > app > app.component.ts > ...
1 import { Component } from '@angular/core';
2
3 // un décorateur : son rôle : transformer la
  classe en ... (composant)
4
5 @Component({
6   selector: 'app-root',
7   templateUrl: './app.component.html',
8   styleUrls: ['./app.component.scss']
9 })
10
11 // une classe (ES2015) exportée
12 export class AppComponent {
13
14   // 1- déf des propriétés
15   public title: string = 'Angular';
16   public version: number;
17
18   // 2- constructeur de classe
19   constructor() {
20     this.version = 16;
21   }
22 }
23

app.component.html ×
formation-angular-2023-06 > TP01-les-fondamentaux > src > app > app.component.html > ...
1
2 <!-- string interpolation de variables {{ }} -->
3 <!-- Binding : le TS est lié à la vue HTML(template) -->
4 <!-- Single way Binding: TS => HTML -->
5 <h1>Formation {{title}} {{version}}</h1>
6
7
8 <!-- composant enfant de AppComponent -->
9 <app-composant2></app-composant2>
10
11 <hr>
12
13 <app-comp-images></app-comp-images>
14
15 <hr>
16 <h2>Espace Client ...</h2>
17 <app-landing-page></app-landing-page>
```

Composants et modules

Création de composants : composant2 – comp-images

- Le sélecteur de composant est placé dans le template HTML d'un composant « parent »
- Le composant appelé par le sélecteur devient le composant enfant.

Un module « structurel » ou « fonctionnel » déclare ses composants et les exporte afin que leurs contenus soient exploitables dans les vues de composants d'autres modules. (import-export)

Ressources disponibles :

<https://dev.webjss.fr/1-Angular/ressources-formation/>



Composants parents et enfants

The image shows a development environment with three code files and a browser preview:

- app.component.html**:

```
1 <!-- string interpolation de variables {{ }} -->
2 <!-- Binding : le TS est lié à la vue HTML(template) -->
3 <!-- Single way Binding: TS => HTML -->
4 <h1>Formation {{title}} {{version}}</h1>
5
6
7
8 <!-- composant enfant de AppComponent -->
9 <app-composant2></app-composant2>
10
11 <hr>
12
13 <app-comp-images></app-comp-images>
14
15 <hr>
16 <h2>Espace Client ...</h2>
17 <app-landing-page></app-landing-page>
```
- composant2.component.html**:

```
1 <h1>Composant #2</h1>
2
```
- comp-images.component.html**:

```
4 test IF -- boucles *ngIf *ngFor *ngSwitch .. -->
5
6
7 <!-- les directives s'écrivent (presque tout le temps) avec des crochets -->
8 <div *ngFor="let valeur of imagesArray" >
9   <span>
10     
19   <!-- la directive ngClass affiche une classe nommée si la condition se
20 </span>
21 </div>
```

The browser preview at localhost:3000 displays:

- TP01LesFondamentaux
- Formation Angular 16
- Composant #2
- Les Avatars ...
- A row of five circular images showing Star Wars characters: Stormtrooper, Chewbacca, Rey, Han Solo, and Darth Vader.

Red arrows indicate the flow of data and component rendering: from the `<app-composant2>` tag in `app.component.html` to the `Composant #2` header in the browser, and from the `<app-comp-images>` tag to the `Les Avatars ...` section.

Directives Binding Pipes

```
1 <h1>{{title | titlecase}}</h1>
2
3 <!-- structural directives Angular (déjà faites...)
4 test IF -- boucles *ngIf *ngFor *ngSwitch .. -->
5
6
7 <!-- les directives s'écrivent (presque tout le temps) avec des crochets -->
8 <div *ngFor="let valeur of imagesArray" >
9   <span>
10     
19     <!-- la directive ngClass affiche une classe nommée si la condition se réalise -->
20   </span>
21 </div>
22
23 <!-- *ngIf : affiche (ou pas) ce qui est contenu dans l'élément en fonction d'un paramètre-->
24 <div *ngIf="flag===true">
25   <p>{{formation}}</p>
26 </div>
27
28 <!-- *ngSwitchCase ==> gère plusieurs tests (conditions) -->
29 <div [ngSwitch]="codeCours">
30
31   <p *ngSwitchCase="'ng'">Formation Angular</p>
32   <p *ngSwitchCase="'react'">Formation React</p>
33   <p *ngSwitchDefault>Formation non trouvée.</p>
34
35 </div>
36
37 <p>{{maDate | date:'dd/M/Y H:m:s'}}</p>
38 <!-- <p>{{total | number:'1.0-2':'fr'}}</p> -->
39 <p>{{total | currency:'EUR':'symbol':'1.0-2':'fr'}}</p>
```

```
app.component.html ×
formation-angular-2023-06 > TP01-les-fondamentaux > src > app > app.component.html > ...
1
2 <!-- string interpolation de variables {{ }} -->
3 <!-- Binding : le TS est lié à la vue HTML(template) -->
4 <!-- Single way Binding: TS => HTML -->
5 <h1>Formation {{title}} {{version}}</h1>
6
7
8 <!-- composant enfant de AppComponent -->
9 <app-composant2></app-composant2>
10
11 <hr>
12
13 <app-comp-images></app-comp-images>
14
15 <hr>
16 <h2>Espace Client ...</h2>
17 <app-landing-page></app-landing-page>

app.module.ts ×
formation-angular-2023-06 > TP01-les-fondamentaux > src > app > app.module.ts > ...
1 import { NgModule } from '@angular/core';
2
3
4 import { BrowserModule } from '@angular/platform-browser';
5
6 // -----
7 // IMPORT LOCAL (ressources fichiers)
8 import { AppRoutingModule } from './app-routing.module';
9 import { AppComponent } from './app.component';
10 import { FormsModule } from '@angular/forms';
11 import { Composant2Component } from './composant2/composant2.component';
12 import { CompImagesComponent } from './comp-images/comp-images.component';
13 import { HeaderComponent } from './compte-client/composants/header/header.component';
14 import { CompteClientModule } from './compte-client/compte-client.module';
15
16 // un décorateur son rôle : transformer la classe en ... (module)
17 @NgModule({
18 // objet de description
19
20 })
21
22 compte-client.module.ts ×
formation-angular-2023-06 > TP01-les-fondamentaux > src > app > compte-client > compte-client.module.ts > ...
13 HeaderComponent,
14 FooterComponent,
15 LandingPageComponent
16 ],
17 imports: [
18   CommonModule
19 ],
20 exports: [
21   HeaderComponent,
22   BodyComponent,
23   FooterComponent,
24   LandingPageComponent
25 ]
26 })
27 export class CompteClientModule { }
28
```

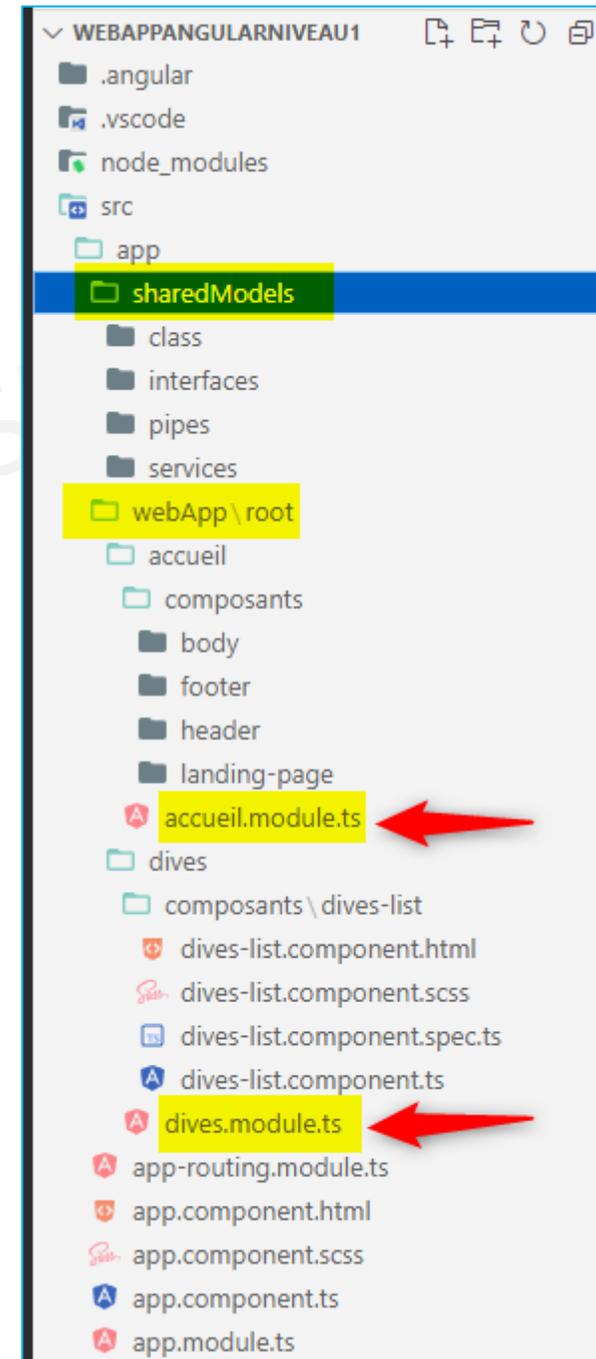
Export des composants depuis le module « compte-client »
Import du module « compte-client » dans « app-module »

Mise en place du TP fil rouge

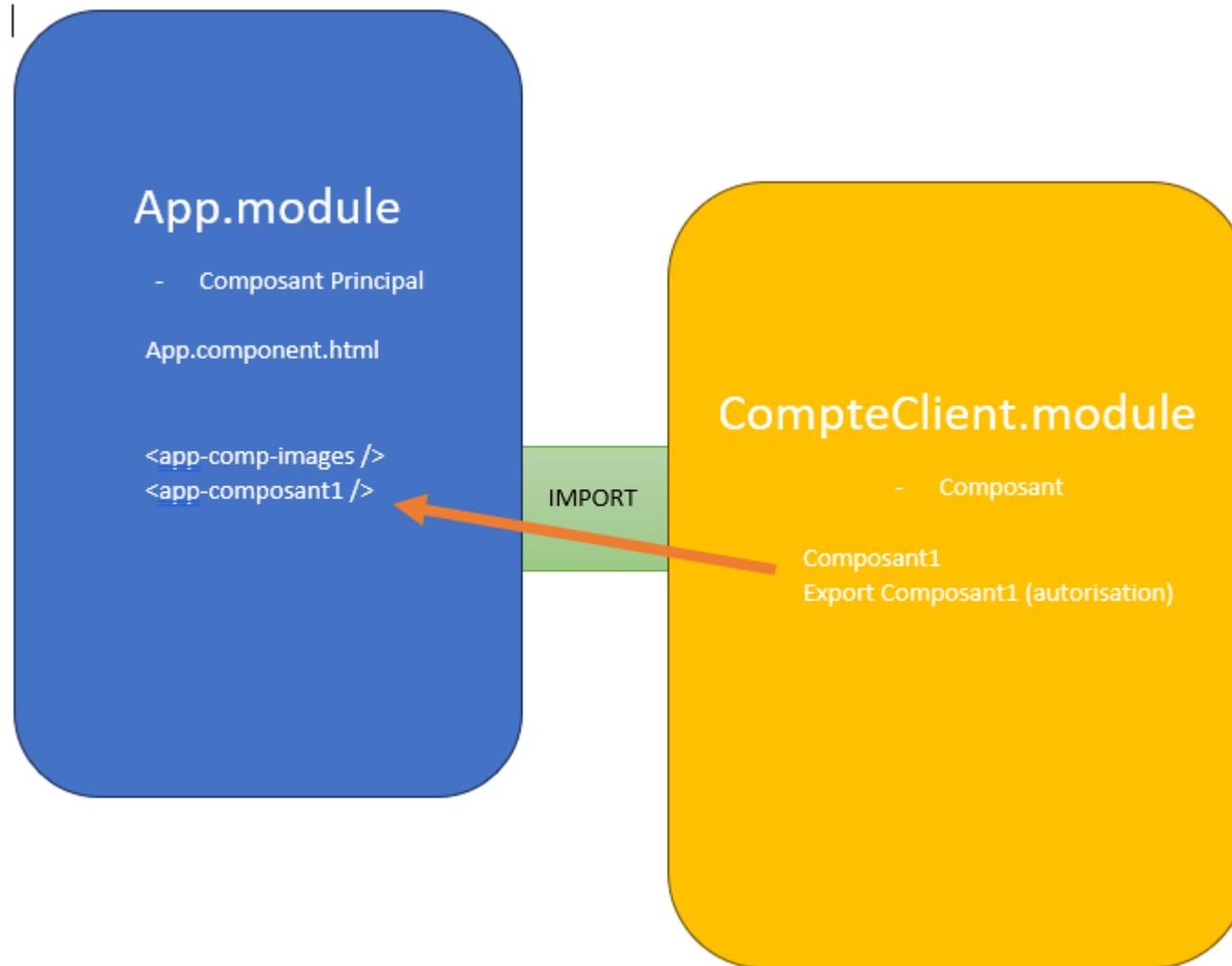
- Création d'un nouveau projet – avec le routage et SCSS
- Installation des dépendances Bootstrap et bootstrap-icons
<https://www.npmjs.com/package/bootstrap>
<https://www.npmjs.com/package/bootstrap-icons>
- Installation de la « dev » dépendance json-server
npm install -D json-server

```
9   "test": "ng test",  
10  "json-server": "json-server --watch src/assets/json/bdd.json -p 3001"  
11
```

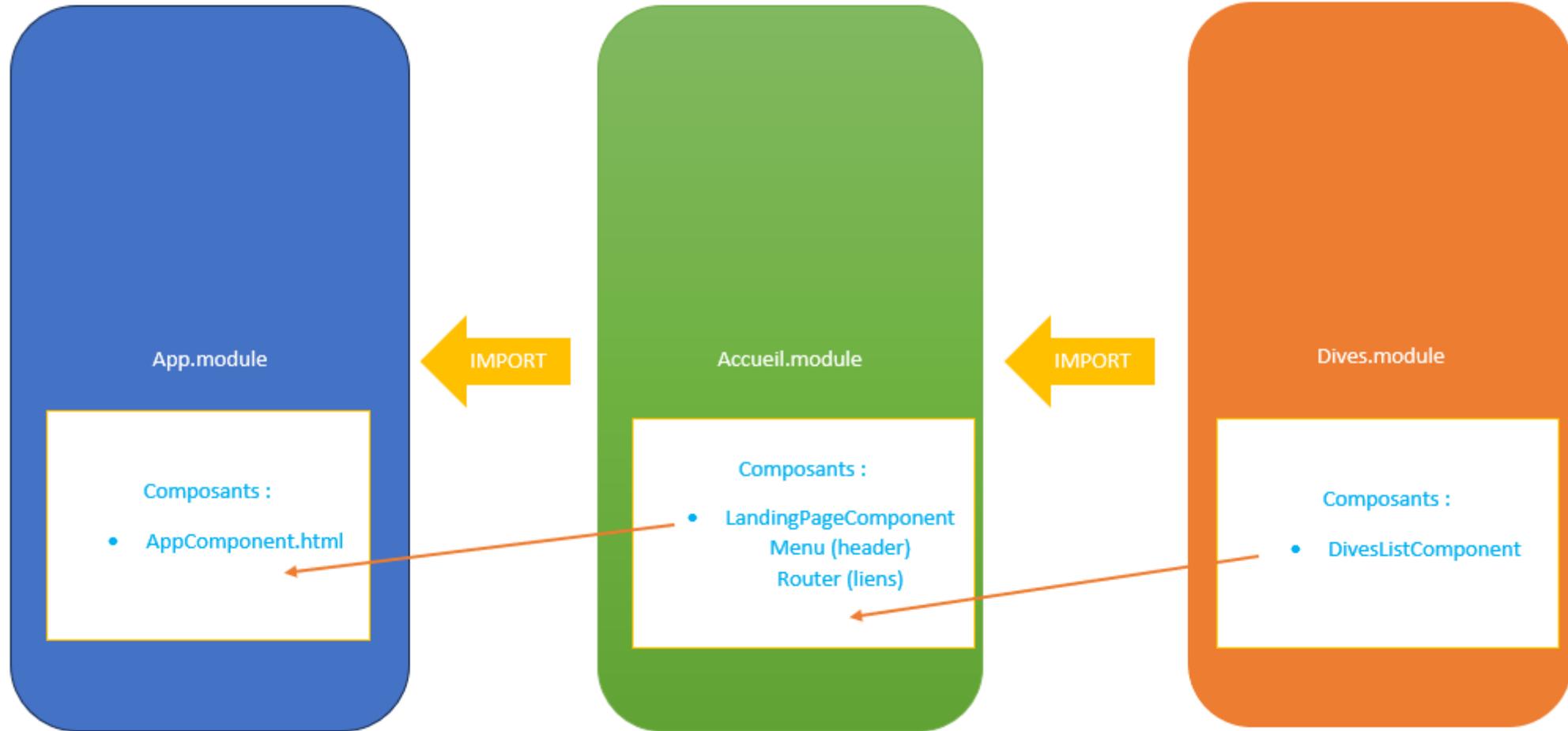
- Bonnes pratiques « structure et architecture » de projet
- injection de dépendances ...



Communication entre modules



Communication entre modules



Mise en place du TP fil rouge

```
PS C:\Users\Michel\Desktop\formation-angular-sparks-10-2023\TP03-services-HTTP> npm run json-server  
  
> tp02-bonnes-pratiques-injection-dependances@0.0.0 json-server  
> json-server --watch src/assets/json/bdd.json -p 3001  
  
\{^_^}/ hi!  
  
Loading src/assets/json/bdd.json  
Done  
  
Resources  
http://localhost:3001/dives  
http://localhost:3001/produits  
  
Home  
http://localhost:3001  
  
Type s + enter at any time to create a snapshot of the database  
Watching...
```



Copy

ORSYS

Mise en place du TP fil rouge

CVS

```
package.json U x header.component.html U ... angular.json U x
TP02-bonnes-pratiques > package.json > {} dependencies
1 {
2   "name": "tp02-bonnes-pratiques",
3   "version": "0.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve --open --port=3000",
7     "build": "ng build",
8     "watch": "ng build --watch --configuration developer",
9     "test": "ng test"
10  },
11  "private": true,
12  "dependencies": {
13    "@angular/animations": "^16.2.0",
14    "@angular/common": "^16.2.0",
15    "@angular/compiler": "^16.2.0",
16    "@angular/core": "^16.2.0",
17    "@angular/forms": "^16.2.0",
18    "@angular/platform-browser": "^16.2.0",
19    "@angular/platform-browser-dynamic": "^16.2.0",
20    "@angular/router": "^16.2.0",
21    "bootstrap": "^5.3.2",
22    "bootstrap-icons": "^1.11.1",
23    "rxjs": "~7.8.0",
24    "tslib": "^2.3.0",
25    "zone.js": "~0.13.0"
26  },
27  "devDependencies": {
28    "@angular-devkit/build-angular": "^16.2.0",
29    "@angular/cli": "~16.2.0",
30    "@types/jasmine": "~4.1.0",
31    "jasmine-core": "~4.6.0",
32    "karma": "~6.4.0",
33    "karma-chrome-launcher": "~3.1.0",
34    "karma-coverage": "~2.2.0",
35    "karma-jasmine": "~5.1.0",
36    "karma-jasmine-html-reporter": "~1.7.0",
37    "typescript": "~5.1.0"
38  }
39 }
40

TP02-bonnes-pratiques > angular.json > {} projects > {} TP02-bonnes-pratiques > {} architect > {} build > {}
16 "architect": {
17   "build": {
18     "builder": "@angular-devkit/build-angular:browser",
19     "options": {
20       "outputPath": "dist/tp02-bonnes-pratiques",
21       "index": "src/index.html",
22       "main": "src/main.ts",
23       "polyfills": [
24         "zone.js"
25       ],
26       "tsConfig": "tsconfig.app.json",
27       "inlineStyleLanguage": "scss",
28       "assets": [
29         "src/favicon.ico",
30         "src/assets"
31       ],
32       "styles": [
33         "node_modules/bootstrap/dist/css/bootstrap.min.css",
34         "node_modules/bootstrap-icons/font/bootstrap-icons.css",
35         "src/styles.scss"
36       ],
37       "scripts": [
38         "node_modules/bootstrap/dist/js/bootstrap.min.js"
39       ]
40     }
41   }
42 }
```

TP : single et double way binding

```
1 import { Component } from '@angular/core';
2
3 Codiumate: Options | Test this class
4 @Component({
5   selector: 'app-home-page',
6   templateUrl: './home-page.component.html',
7   styleUrls: ['./home-page.component.scss']
8 })
9 export class HomePageComponent {
10   // propriétés
11   public codeCours:string='VUE';
12
13   public result:string='ok';
14
15   // méthodes
16   Codiumate: Options | Test this method
17   public methodClic(){
18     this.result='Vous avez cliqué !';
19   }
20 }
21
```

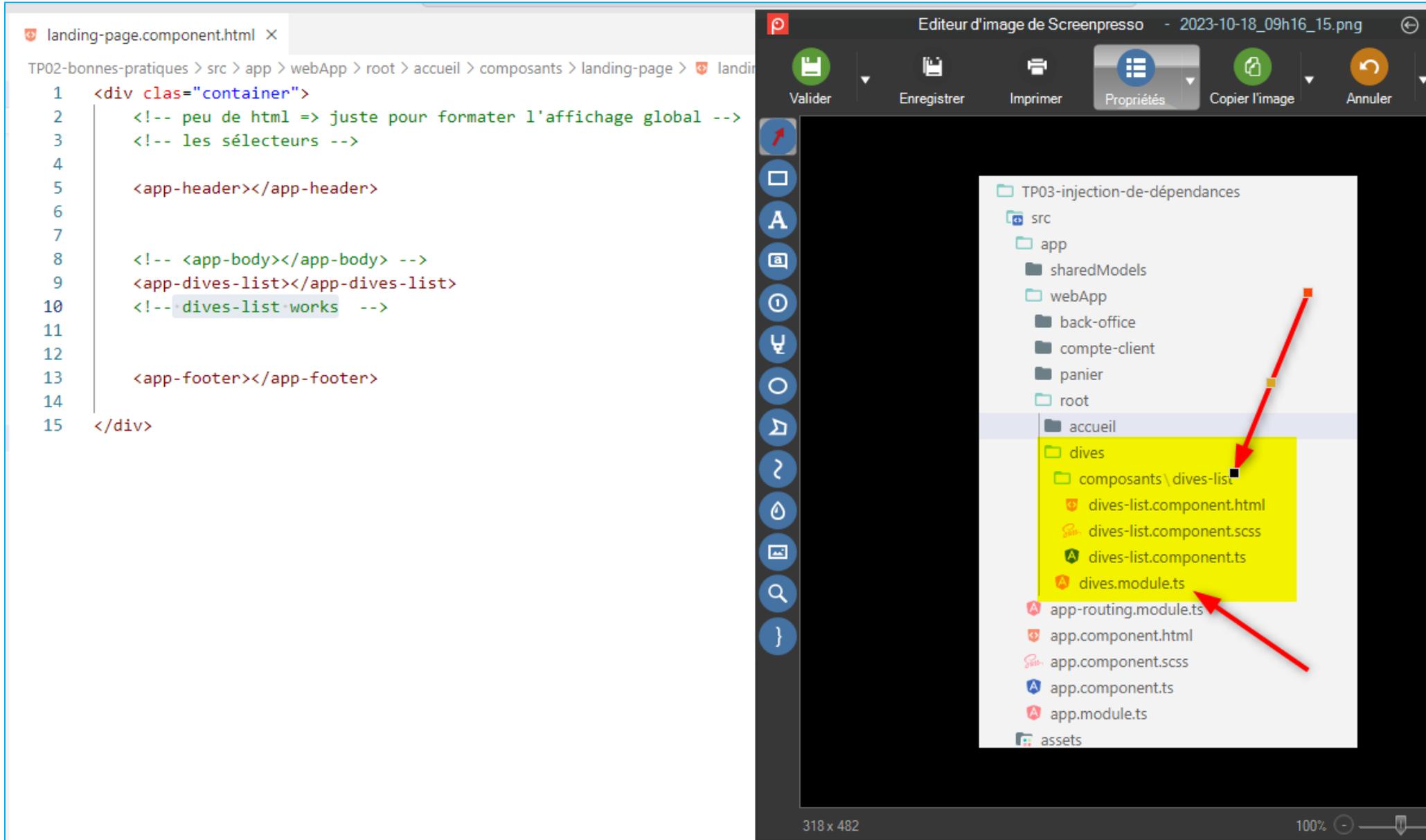
```
Go to component
1 <div class="alert alert-primary">
2   <h3>HOME PAGE</h3>
3
4   <!-- single way Binding : HTML => TS -->
5   <button class="btn btn-warning" (click)="methodClic...">
6     Cliquez pour tester le single way binding...
7   </button>
8
9   <p>{{result}}</p>
10
11   <!-- EXO :
12     *ngIf *ngSwitchCase
13     @if() Control flow NG>17+
14
15   si le codeCours est égal NG => Vous avez choisi
16   Formation Angular
17   Sinon ..... REACT => .... React
18   Sinon ..... VUE => ..... Vue
19   Autres => Choisir une formation
20   -->
21
22 </div>
```

TP :

```
home-page.component.ts x ...
app > webApp > root > accueil > composants > home-page > home-page
1 import { Component } from '@angular/core';
2
3 Codiumate: Options | Test this class
4 @Component({
5   selector: 'app-home-page',
6   templateUrl: './home-page.component.html',
7   styleUrls: ['./home-page.component.scss']
8 })
9 export class HomePageComponent {
10   // propriétés
11   public codeCours:string='VUE';
12
13   public result:string='';
14
15   // méthodes
16   Codiumate: Options | Test this method
17   public methodClic(){
18     this.result='Vous avez cliqué !';
19   }
20 }
21
```

```
home-page.component.html x
TP02 > src > app > webApp > root > accueil > composants > home-page > home-page.component.html > div.ale
Go to component
1 <div class="alert alert-primary">
2   <h3>HOME PAGE</h3>
3
4   <!-- single way Binding : HTML => TS -->
5
6   <button class="btn btn-primary" (click)="methodClic()">
7     Cliquez pour tester le single way binding...
8   </button>
9   <p>{{result}}</p>
10
11 <p></p>
12
13 <select class="form-control" [(ngModel)]="codeCours">
14   <option value="">Choisir pour tester le double way binding</option>
15   <option value="NG">Angular ...</option>
16   <option value="REACT">React ...</option>
17   <option value="VUE">Vue ...</option>
18 </select>
19
20 <p></p>
21
22 <div class="alert alert-warning">
23
24   @if (codeCours=== "NG") {
25     <span>Vous avez choisi Angular</span>
26   }
27   @else if (codeCours=== "REACT") {
28     <span>Vous avez choisi React</span>
29   }
30   @else if (codeCours=== "VUE") {
31     <span>Vous avez choisi Vue</span>
32   }
33   @else {
34     <span>Choisissez une formation</span>
35   }
36
```

TP : créer l'arborescence de dives



The image displays two side-by-side screenshots from a development environment.

The left screenshot shows a code editor with the file `landing-page.component.html` open. The code is as follows:

```
1 <div clas="container">
2   <!-- peu de html => juste pour formater l'affichage global -->
3   <!-- les sélecteurs -->
4
5   <app-header></app-header>
6
7
8   <!-- <app-body></app-body> -->
9   <app-dives-list></app-dives-list>
10  <!-- dives-list works -->
11
12
13  <app-footer></app-footer>
14
15 </div>
```

The right screenshot shows a file explorer window titled "Editeur d'image de Screenpresso - 2023-10-18_09h16_15.png". The file tree structure is:

- TP03-injection-de-dépendances
 - src
 - app
 - sharedModels
 - webApp
 - back-office
 - compte-client
 - panier
 - root
 - accueil
 - dives
 - composants \ dives-list
 - dives-list.component.html
 - dives-list.component.scss
 - dives-list.component.ts
 - dives.module.ts
 - app-routing.module.ts
 - app.component.html
 - app.component.scss
 - app.component.ts
 - app.module.ts
 - assets

Two red arrows point to the `dives-list` folder and the `dives.module.ts` file in the file explorer.

TP : créer l'arborescence de dives

The image shows a development environment with three code files and a browser preview. A red arrow points from the `<app-dives-list>` tag in `landing-page.component.html` to the `DivesModule` import in `accueil.module.ts`.

```
landing-page.component.html
1 <div class="container">
2   <!-- peu de html => juste pour formater l'affichage global -->
3   <!-- les sélecteurs -->
4
5   <app-header></app-header>
6
7
8   <!-- <app-body></app-body> -->
9   <app-dives-list></app-dives-list>
10  <!-- dives-list works -->
11
12
13  <app-footer></app-footer>
14
15 </div>
```

```
dives.module.ts
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { DivesListComponent } from './composants/dives-list';
4
5
6
7 @NgModule({
8   declarations: [
9     DivesListComponent
10  ],
11   imports: [
12     CommonModule
13  ],
14   exports: [
15     DivesListComponent
16  ]
17 })
18 export class DivesModule
19
```

```
accueil.module.ts
6 import { FooterComponent } from './composants/footer/footer.component';
7 import { DivesModule } from '../dives/dives.module';
8
9
10
11 @NgModule({
12   declarations: [
13     LandingPageComponent,
14     HeaderComponent,
15     BodyComponent,
16     FooterComponent
17  ],
18   imports: [
19     CommonModule,
20     // import de nos modules
21     DivesModule
22  ]
23 })
24 export class AccueilModule
25
```

The browser preview shows the Angular logo and the text "dives-list works!".

app.module.ts

```

1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { AccueilModule } from './webApp/accueil/accueil.module';
7
8 @NgModule({
9   declarations: [
10    AppComponent
11  ],
12  imports: [
13    BrowserModule,
14    AppRoutingModule,
15    AccueilModule
16  ],
17  providers: [],
18  bootstrap: [AppComponent]
19 })
20 export class AppModule { }
21

```

landing-page.component.html

```

1 <app-header></app-header>
2 <!-- <app-body></app-body -->
3 <app-dives-list></app-dives-list>
4 <app-footer></app-footer>

```



accueil.module.ts

```

1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { LandingPageComponent } from './composants/landing-page/landing-page.component';
4 import { HeaderComponent } from './composants/header/header.component';
5 import { FooterComponent } from './composants/footer/footer.component';
6 import { BodyComponent } from './composants/body/body.component';
7 import { DivesModule } from '../dives/dives.module';
8
9 @NgModule({
10  declarations: [
11    LandingPageComponent,
12    HeaderComponent,
13    FooterComponent,
14    BodyComponent
15  ],
16  imports: [
17    CommonModule,
18    DivesModule
19  ],
20  exports: [
21    LandingPageComponent,

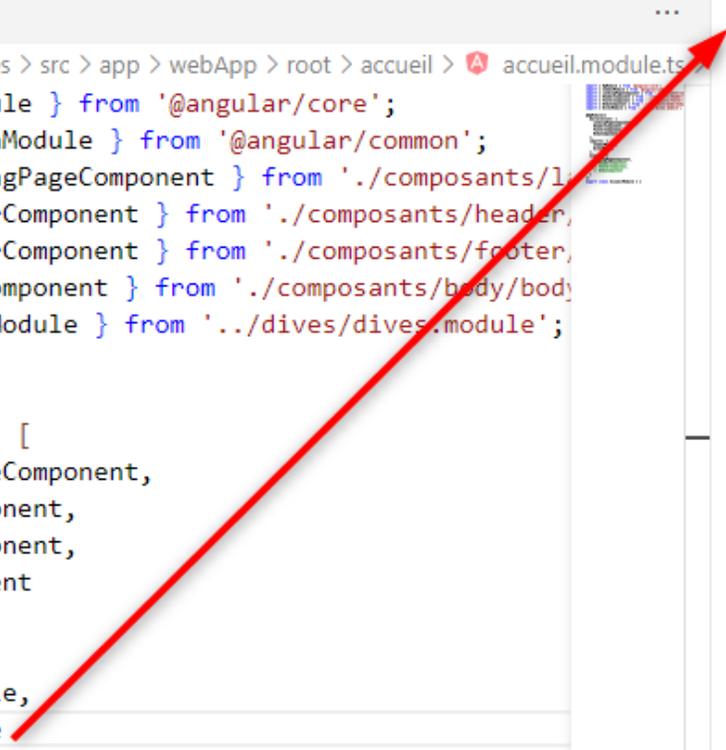
```

dives.module.ts

```

1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { DivesListComponent } from './dives-list/dives-list.component';
4
5 @NgModule({
6  declarations: [
7    DivesListComponent
8  ],
9  imports: [
10   CommonModule
11  ],
12  exports: [
13    DivesListComponent
14  ]
15 })
16 export class DivesModule { }
17
18

```



L'injection de dépendances Angular

Au moment de la création du composant (constructor)
Le composant reçoit un service ou une « fonctionnalité » requise

```
dives.service.ts | dives-list.component.ts
```

```
TP03-injection-de-dépendances > src > app > sharedModels > services > dives.service.ts > Injection-de-dépendances > src > app > webApp > root > dives > composants > dives-list >
```

```
81     "location": "Cozumel - Mexique - Départ de Pl",
82     "level": 2,
83     "description": "Plongée de niveau 2 FFESSM ou",
84     "latitude": 20.4317585,
85     "longitude": -86.9202746,
86     "evaluation": [
87       7,
88       9,
89       8
90     ],
91     "photo": "https://dev.webjts.fr/images/fond5.jpg",
92   };
93 };
94 }
95
96 // Méthodes
97 public getDives = () => {
98   return this.dives;
99 };
100 }
101 }
102
```

```
1 import { Component, OnInit } from '@angular/core';
2 import { DivesService } from 'src/app/sharedModels/services';
3
4 @Component({
5   selector: 'app-dives-list',
6   templateUrl: './dives-list.component.html',
7   styleUrls: ['./dives-list.component.scss']
8 })
9 export class DivesListComponent implements OnInit {
10
11   // 1- Props
12
13   // 2- Constructor
14   constructor(
15     // injection de dépendances requises
16     // au moment de la construction (création)
17     // du composant
18     private _service:DivesService
19   ) {
20     console.warn('Constructor');
21   }
22
23   // 3- Lifecycle
24   ngOnInit(): void {
25     console.warn('NgOnInit');
26     // Rôle : chargement des datas
27     console.table(this._service.getDives());
28
```

L'injection de dépendances

1^{er} exemple : le service

```
dives-list.component.ts x
dives.service.ts x

bApp > root > dives > composants > dives-list > dives-list.component.ts > DivesListComponent
16 // @ViewChild('btnPlus') eltBtn!: ElementRef;
17 @ViewChildren('btnPlus') eltBtnCollection!: QueryList<ElementRef>;
18
19 // 2- const
20 // 1'injection de dépendances permet de lier un service
21
22 constructor(
23     private _service: DivesService,
24     private _title: Title,
25     private _meta: Meta
26 ) {
27     console.warn('Constructor');
28     this.dives = [];
29     this._title.setTitle('Liste des Plongées SEO ....');
30     this._meta.addTags([
31         { name: 'description', content: 'Texte de description' },
32         { name: 'author', content: 'MB CREATION WEB' }
33     ]);
34 }
35
36 // -----
37 // 3- lifeCycle
38 // -----
39 ngOnInit(): void {
40     // Chargement des datas
41     console.warn('NgOnInit');
42     this._service.getDives()
43
44     // .subscribe(
45     //     // next seulement ...
46     //     (datas: Dives[]) => {
47     //         this.dives = datas;
48     //     }
49     // );
50
51     .subscribe({
52         next: // Call Back OK
53             (datas: Dives[]) => {
54                 this.dives = datas;
55             },
56         error:
57             e => console.log(e)
58         ,
59         complete:
60             () => console.log('Observer Complete')
61     })

```

```
TP03-services-http > src > app > sharedModels > services > dives.service.ts > DivesService > getDives
1 import { Injectable } from '@angular/core';
2 import { Dives } from '../class/dives';
3 import { HttpClient } from '@angular/common/http';
4 import { Observable, tap, map } from 'rxjs';
5
6 @Injectable({
7     providedIn: 'root'
8 })
9
10 export class DivesService {
11
12     // 1- props
13     private dives: Dives[];
14
15     // 2- const
16     constructor(private _http: HttpClient) {
17         this.dives = [];
18     }
19
20     // 3- Méthodes
21     public getDives = (): Observable<Dives[]> => {
22
23         const API_URL = 'http://localhost:3001/dives';
24         // const API_URL = 'https://dev.webjs.fr/dives.json';
25
26         return this._http.get<Dives[]>(API_URL).pipe(
27             // le pipe permet d'enchaîner les opérateurs RXJS
28             // 1 opérateur RXJS est une fonction qui modifie
29             // ou transforme la séquence(stream ou flux) de valeurs émises par l'observable
30
31             tap(
32                 // opérateur de debug => console
33                 (response: any) => {
34                     console.log('----- Réponse depuis le service : ', response);
35                 }
36             ),
37             map(
38                 (datas: Dives[]) => {
39                     return datas.filter(
40                         (data: Dives) => {
41                             // return data.id===5
42                             return data.id >=1
43                         }
44                     )
45                 }
46             )
47         )

```

Copy

Life Cycle du composant constructor & ngOnInit

ng g(enerate) s(ervice) dives

```
dives-list.component.ts 3  dives.service.ts X
TP02-bonnes-pratiques-injection-dependances > src > app > sharedModel
1 import { Injectable } from '@angular/core';
2 import { Dives } from '../class/dives';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7
8 export class DivesService {
9
10   // 1- props
11   private dives: Dives[];
12
13   // 2- const
14   constructor() {
15     this.dives = [
16       {
17         "id": 1,
18         "name": "Trou Aux Biches - L'Epave du
19         "location": "Ile Maurice - Trou aux B
```

```
constructor(
  private _service: DivesService,
  private _title: Title,
  private _meta: Meta
) {
  console.warn('Constructor');
  this.dives = [];
  this._title.setTitle('Liste des Plongées SEO ....');
  this._meta.addTags([
    { name: 'description', content: 'Texte de description ...' },
    { name: 'author', content: 'MB CREATION WEB' }
  ]);
}

// -----
// 3- lifeCycle
// -----
ngOnInit(): void {
  // Chargement des datas
  console.warn('NgOnInit');
  // console.table(this._service.getDives());
  // -- on passe les datas du service à la prop dives du composant
  this.dives = this._service.getDives();
}
```

TP : renseigner les valeurs manquantes

```
dives-list.component.ts 9+ x
1 import { Component, OnInit } from '@angular/core';
2 import { Dives } from 'src/app/sharedModels/class/dives';
3 import { DivesService } from 'src/app/sharedModels/services/dives-service';
4
5 @Component({
6   selector: 'app-dives-list',
7   templateUrl: './dives-list.component.html',
8   styleUrls: ['./dives-list.component.scss']
9 })
10 export class DivesListComponent implements OnInit {
11
12   // 1- Props
13   ~~~~~
14
15   // 2- Constructor
16   constructor(
17     // injection de dépendances requises
18     // au moment de la construction (création)
19     // du composant
20     private _service: DivesService
21   ) {
22     console.warn('Constructor');
23     ~~~~~
24   }
25
26   // 3- Lifecycle
27   ngOnInit(): void {
28     console.warn('NgOnInit');
29     // Rôle : chargement des datas
30     console.table(this._service.getDives());
31     ~~~~~
32
dives-list.component.html x
1 <h1 class="alert alert-primary">Liste des Plongées</h1>
2
3 <div class="row">
4   <div class="col-md-3" *ngFor="<div class="card">
5     <!-- récupérer chaque image -->
6     <img src="" class="card-img-top">
7     <div class="card-body">
8       <h3 class="card-title">Nom : </h3>
9       <p class="card-text">Description : </p>
10
11     <!-- cibler un élément du dom en template référence # -->
12     <button class="btn btn-primary" #btnPlus>En Savoir Plus</button>
13
14   </div>
15 </div>
16 </div>
17
18 </div>
19
20 </div>
21
22
```

The image shows a side-by-side comparison of a TypeScript component file and its HTML template. Red arrows indicate the mapping between code and HTML: one arrow points from the component's selector to the HTML's class attribute, another from the constructor to the HTML's *ngFor loop, and a third from the ngOnInit method to the HTML's button element. Yellow boxes highlight placeholder text (~~~~~) in the TypeScript file and corresponding placeholder text in the HTML template.

Life Cycle du composant

AfterViewInit – AfterViewChecked - OnDestroy

```
// -----  
//     cycles de vie  
// -----  
ngOnInit():void {  
    // exploitation des datas  
    console.warn('OnInit');  
    console.table(this._service.getDives());  
    this.dives=this._service.getDives();  
    console.log('dans le init : ', this.eltBtn);  
    // output => undefined  
}
```

```
// -----  
ngAfterViewInit(): void {  
    console.warn('ngAfterViewInit');  
    // quand le DOM (document object model) est chargé  
    // toute la vue HTML et tous ces éléments sont prêts  
    console.log('dans le After View Init : ', this.eltBtn);  
    // output => ElementRef {nativeElement: button.btn.btn-primary}  
    let texte='AfterViewInit';  
    // this.eltBtn.nativeElement.innerHTML = `Plus d'infos ${texte}`;  
  
    console.log(this.eltCollectionBtn);  
  
    this.eltCollectionBtn.forEach(  
        (elt:ElementRef) => {  
            elt.nativeElement.innerHTML=`Plus d'infos`;  
            // elt.nativeElement.className='btn btn-warning';  
            elt.nativeElement.classList.replace('btn-primary', 'btn-success');  
        }  
    );  
}
```

RXJS – Les Observables

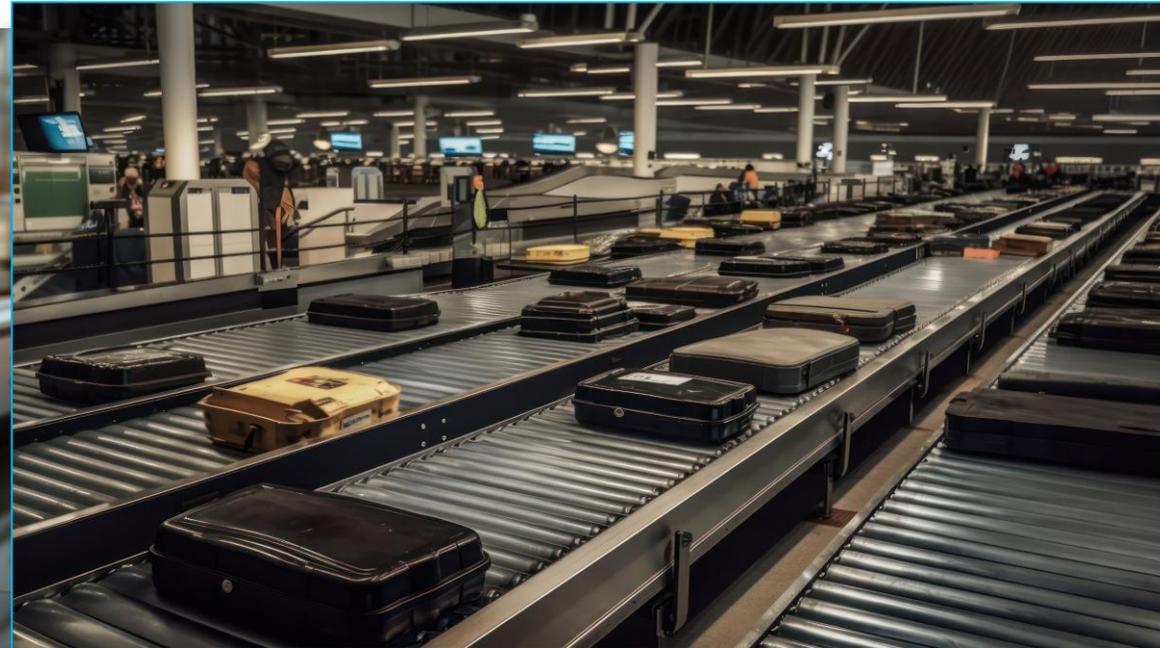
Niveau 1



Copyright Michel BOCCIOLESI - 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.



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)

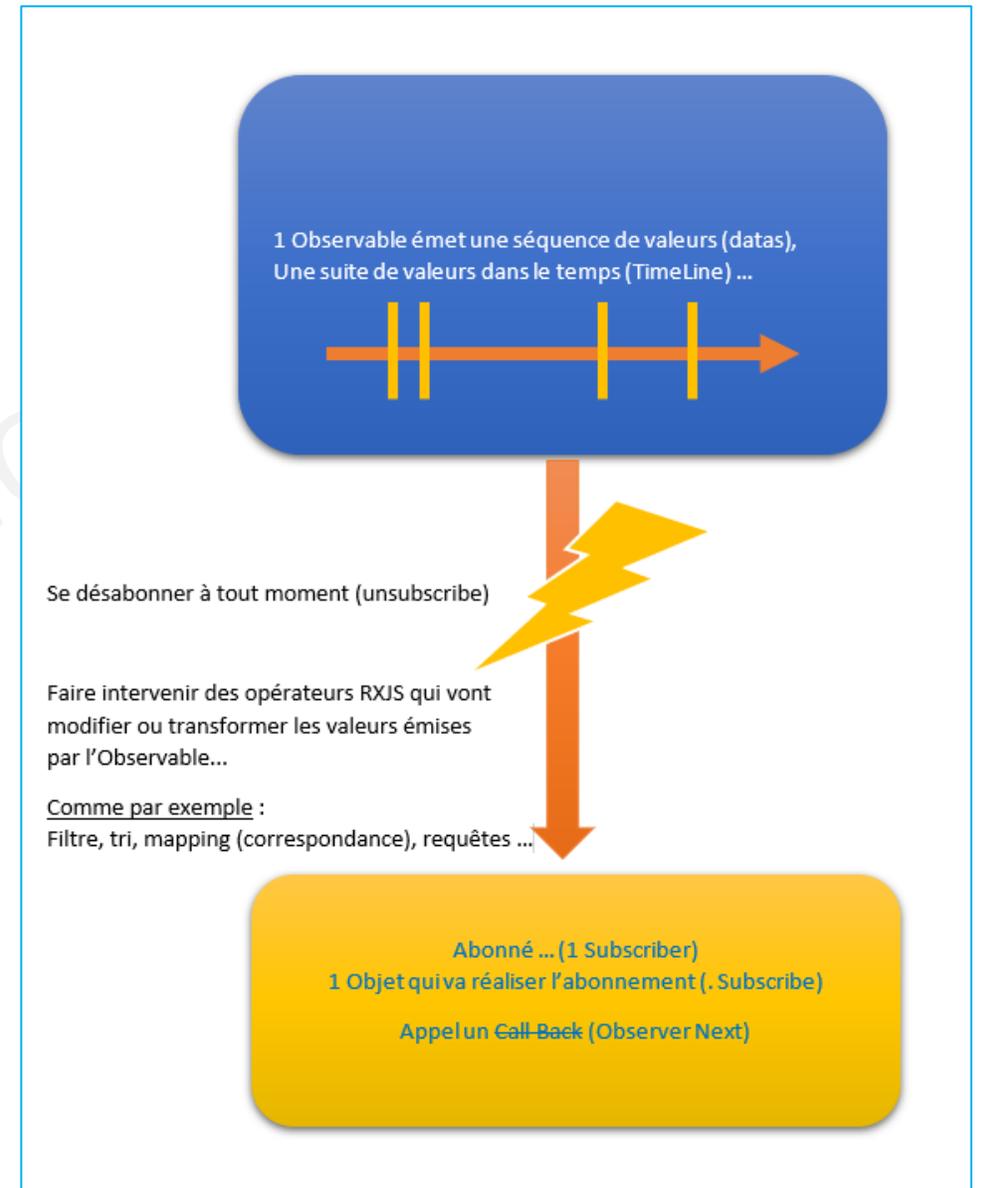
Opérateurs : des fonctions tap, map, merge, filter qui vont pouvoir modifier/transformer ce flux de valeurs émises par l'observable

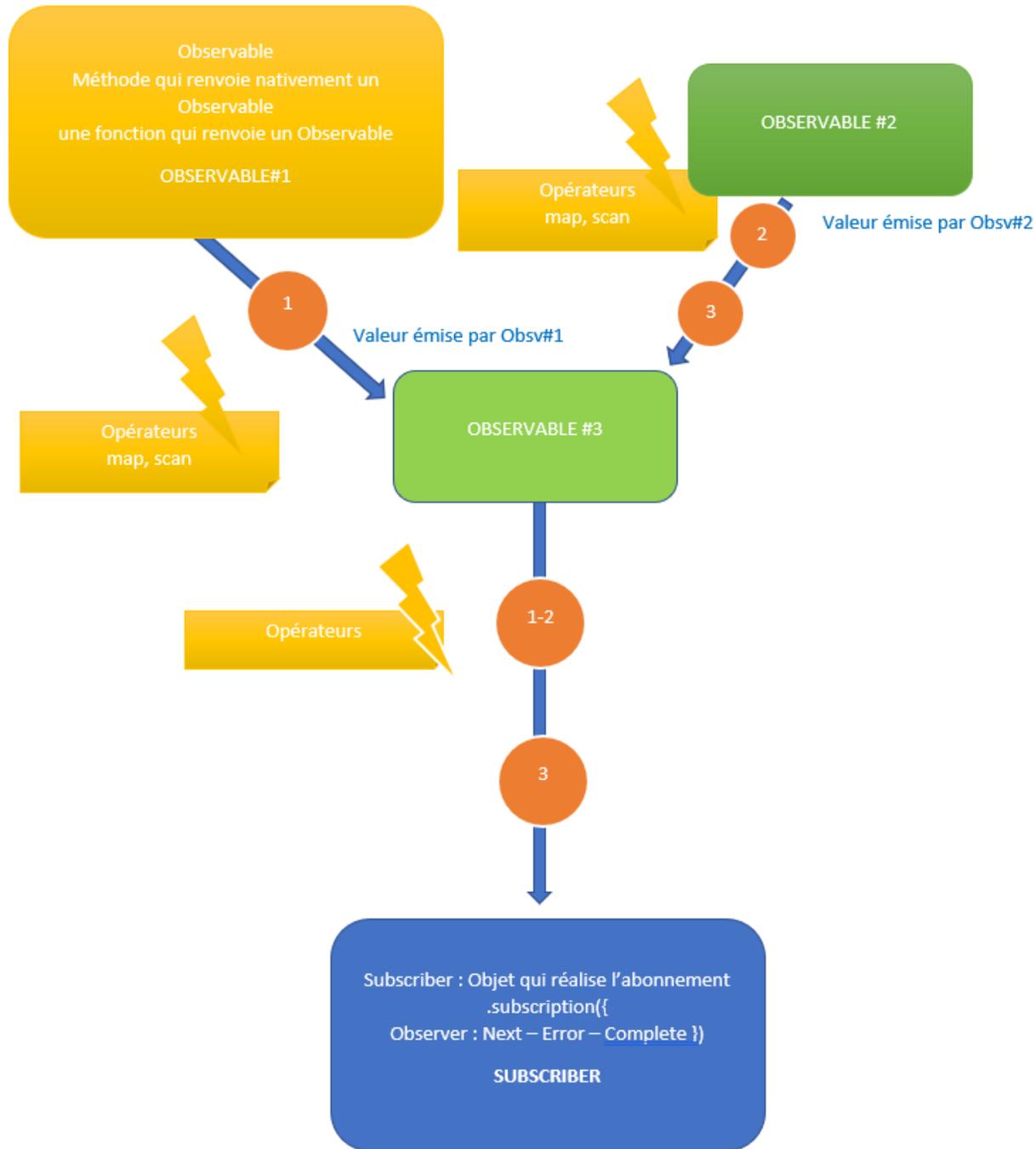
Dans le subscribe => Les observers (fcts de call back) qui se définissent dans le subscribe : Next - Error – Complete

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 recombinaison pour former une nouvelle séquence de valeurs !





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

Routage



Copyright Michel BOUZIIOLESI - ORSYS

Routing : Concepts de base

Un path (URL ou fragment d'URL, l'URI) est associé à un composant dans le fichier « app-routing.module.ts »

Le template du composant est chargée dans le router-outlet, lorsque ce path est invoqué.

Copyright Michel BOCCIOLESI - ORSYS

Routing : Concepts de base

The image shows a code editor with four files open, illustrating the flow of routing information in an Angular application:

- app-routing.module.ts** (top left):

```
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3
4 // import des composants
5 import { DivesListComponent } from './webApp/root/dives/composants/dives-list/';
6 import { BodyComponent } from './webApp/root/accueil/composants/body/body.comp';
7
8 const routes: Routes = [
9
10   { path: '', component: BodyComponent },
11   { path: 'liste-des-plongees', component: DivesListComponent }
12 ];
13
14 @NgModule({
15   imports: [RouterModule.forRoot(routes)],
16   exports: [RouterModule]
17 })
18 export class AppRoutingModule { }
```
- header.component.html** (top right):

```
4
5
6 <button class="navbar-toggler" type="button"
7   <span class="navbar-toggler-icon"></span>
8 </button>
9
10 <div class="collapse navbar-collapse" id="men
11   <ul class="navbar-nav">
12     <li class="nav-item">
13       <a class="nav-link"><i class="bi
14     </li>
15     <li class="nav-item">
16       <a class="nav-link"
17         routerLink="liste-des-plongees"
18     </li>
19     <li class="bi bi-card-list </li>
20   </ul>
21   <li class="nav-item">
22     <a class="nav-link"><i class="bi
23   </li>
24 </ul>
25 </div>
26 </nav>
```
- landing-page.component.html** (bottom left):

```
1 <!-- <div class="container"> -->
2   <app-header></app-header>
3
4   <div class="alert alert-warning"
5     <!-- comparé à un composant angular -->
6     <router-outlet></router-outlet>
7   </div>
8
9
10
11 <app-footer></app-footer>
```
- accueil.module.ts** (bottom right):

```
11
12 @NgModule({
13   declarations: [
14     LandingPageComponent,
15     HeaderComponent,
16     FooterComponent,
17     BodyComponent
18   ],
19   imports: [
20     CommonModule,
21     RouterModule,
```

Red arrows indicate the flow of information:

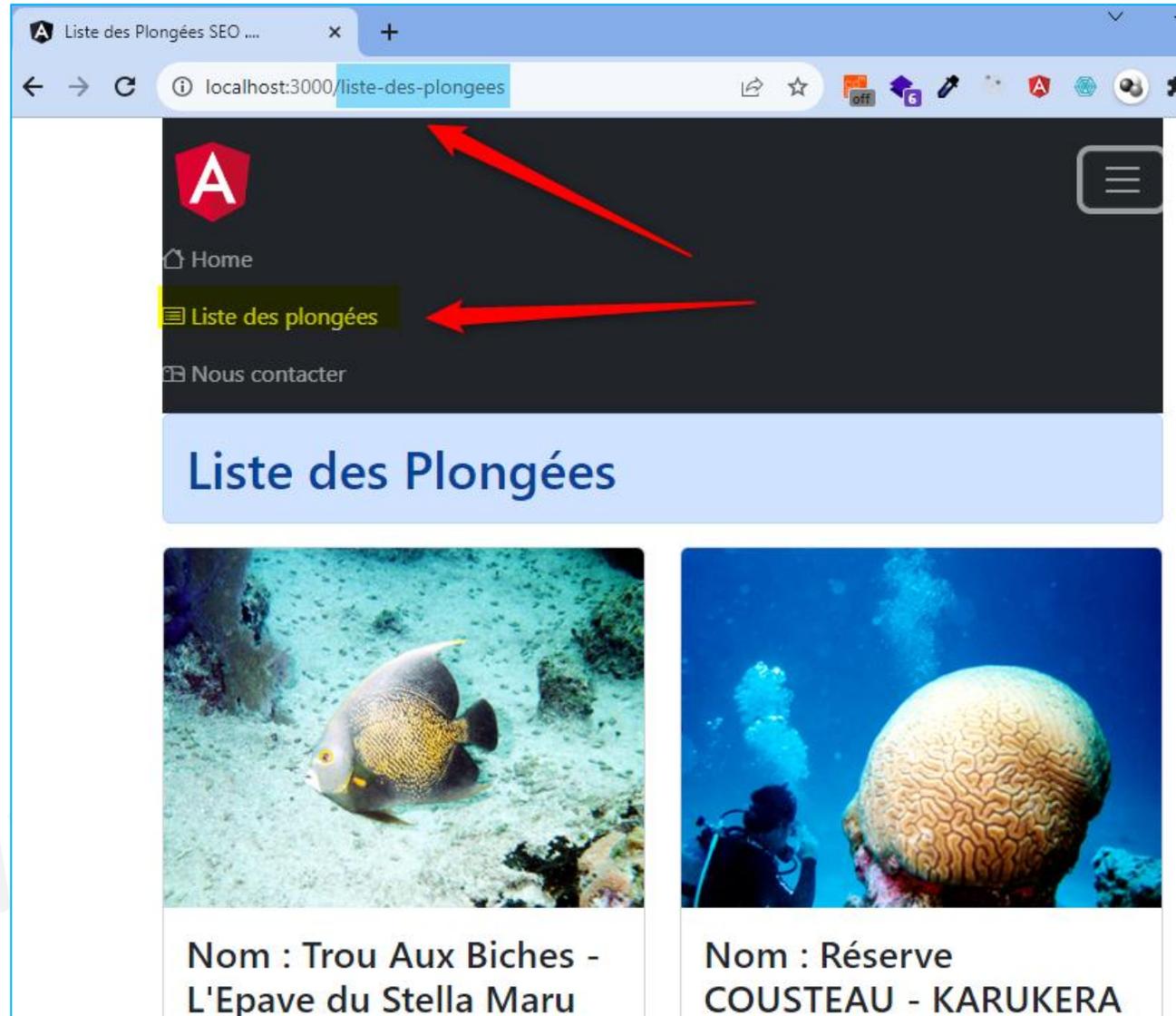
- From the `routerLink` attribute in the `header.component.html` to the `router-outlet` in `landing-page.component.html`.
- From the `routerLink` attribute in the `header.component.html` to the `component` property in the `app-routing.module.ts` routes array.

Routing : Concepts de base

```
app-routing.module.ts ×
TP04-routage > src > app > app-routing.module.ts > routes
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { BodyComponent } from '../webApp/root/accueil/composants/body/body.component';
4 import { DivesListComponent } from '../webApp/root/dives/composants/dives-list/dives-list.component';
5 import { Page404Component } from '../sharedModels/composants/page404.component';
6 import { ContactsComponent } from '../webApp/root/form/composants/contacts/contacts.component';
7 import { DivesDetailComponent } from '../webApp/root/dives/composants/dives-detail/dives-detail.component';
8
9 const routes: Routes = [
10   { path: '', component: BodyComponent },
11   // domain.tld/ (page de démarrage)
12   // { path: 'accueil', component: BodyComponent }, // domain.tld/accueil
13   { path: 'accueil', redirectTo: '', pathMatch: 'full' },
14   // { path: 'nouvelle-url', component: DivesListComponent },
15   // { path: 'ancienne-url-seo-déjà-référencée',
16   redirectTo: 'nouvelle-url', pathMatch: 'full' },
17   { path: 'contactez-nous', component: ContactsComponent },
18   { path: 'liste-des-plongees', component: DivesListComponent },
19   { path: 'liste-des-plongees/:id', component: DivesDetailComponent },
20
21   // Gestion des erreurs : toujours en dernier !!
22   // { path: '**', redirectTo: '', pathMatch: 'full' }
23   { path: '**', component: Page404Component }
24 ];
25
26 @NgModule({
27   imports: [RouterModule.forRoot(routes)],
28   exports: [RouterModule]
29 })
30 export class AppRoutingModule { }
31
```

```
header.component.html ×
TP04-routage > src > app > webApp > root > accueil > composants > header > header.component.html > nav.nav
1 <nav class="navbar navbar-expand-lg bg-dark navbar-dark sticky-top">
2   
3
4   <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-
5     <span class="navbar-toggler-icon"></span>
6   </button>
7
8   <div class="collapse navbar-collapse" id="menu">
9     <ul class="navbar-nav">
10      <li class="nav-item">
11        <a class="nav-link"
12          routerLink="accueil"
13          ><i class="bi bi-house"></i> Home</a>
14      </li>
15
16      <li class="nav-item">
17        <!-- <a class="nav-link"
18          [routerLink]="['liste-des-plongees']"
19          ><i class="bi bi-card-list"></i> Liste des plongées</a -->
20        <a class="nav-link"
21          routerLink="liste-des-plongees"
22          ><i class="bi bi-card-list"></i> Liste des plongées</a>
23      </li>
24
25      <li class="nav-item">
26        <a class="nav-link"
27          routerLink="contactez-nous"
28          ><i class="bi bi-mailbox"></i> Nous contacter</a>
29      </li>
30    </ul>
31  </div>
32 </nav>
```

Routing : Concepts de base



The screenshot shows a web browser window with the address bar displaying `localhost:3000/liste-des-plongees`. The page content includes a navigation menu with a red shield logo containing a white 'A'. The menu items are 'Home', 'Liste des plongées' (highlighted in yellow), and 'Nous contacter'. Below the menu is a light blue header with the text 'Liste des Plongées'. The main content area displays two cards, each with an underwater photograph and a caption. The first card shows a fish and is captioned 'Nom : Trou Aux Biches - L'Epave du Stella Maru'. The second card shows a diver and a brain coral and is captioned 'Nom : Réserve COUSTEAU - KARUKERA'. Two red arrows point from the 'Liste des plongées' menu item to the top-left corner of the page content area.

Localhost:3000/liste-des-plongees

Home

Liste des plongées

Nous contacter

Liste des Plongées



Nom : Trou Aux Biches - L'Epave du Stella Maru



Nom : Réserve COUSTEAU - KARUKERA

Routing : Concepts de base

```
landing-page.component.html x
src > app > webApp > root > accueil > composants > landing-page > landing-page.component.html
1 <div style="min-height:100vh;" class="alert alert-warning">
2 <app-header></app-header>
3 <!-- <app-body></app-body -->
4 <!-- <app-dives-list></app-dives-list -->
5
6 <!-- router-outlet primary -->
7 <div class="alert alert-primary">
8
9 <router-outlet></router-outlet>
10 </div>
11
12 <!-- router-outlet autres -->
13 <div class="alert alert-success">
14
15 <router-outlet name="outlet-2"></router-outlet>
16 </div>
17
18
19
20
21 </div>
22 <app-footer></app-footer>
```

```
app-routing.module.ts x
src > app > app-routing.module.ts > routes > canActivate
16 domain.tld/accueil
// { path:'accueil' , redirectTo:'',
pathMatch:'full'},
17 // { path:'nouvelle-url',
component:DivesListComponent},
18 // { path:'ancienne-url-seo-déjà-référencée',
redirectTo:'nouvelle-url', pathMatch:'full'},
19 { path: 'liste-des-plongees', component:
DivesListComponent },
20 { path: 'liste-des-plongees/:id', component:
DiveDetailComponent },
21
22 { path: 'routing-angular', component:
CompRoutingOutletComponent, outlet: 'outlet-2',
canActivate: [RouteGuardService] },
23
24 // 1ère manière de faire : charger un composant depuis
un module qui a son propre système de routage forChild
25 // {
26 // path: 'mon-compte-client',
27 // component: ClientLandingPageComponent,
28 // children: clientRoutes
29 // },
30
31 // 2nde manière optimisée : lazy loading avec ES2015
(promesses-import)
32 // {
33 // path: 'mon-compte-client',
34 // component: ClientLandingPageComponent,
35 // loadChildren:
36 // () => import('./webApp/compte-client/
compte-client.module').then(
```

```
header.component.html x
src > app > webApp > root > accueil > composants > header > header.component.html
16 Liste des plongées
17 </a>
18 </li>
19
20 <li class="nav-item">
21 <a class="nav-link" routerLink="mon-compt
22 </li>
23
24 <li class="nav-item">
25 <a class="nav-link" [routerLink]="[{
26 outlets: {
27 'outlet-2': 'routing-angular'
28 }]"
29 >
30
31 [skipLocationChange]="true"><i class="bi bi-m
32 </li>
33
34 <li class="nav-item">
35 <a class="nav-link"><i class="bi bi-mailb
36 </li>
37 </ul>
38 </div>
39 </nav>
```

Routing : Concepts avancés – QueryParams - Router



Nom : Trou Aux Biches - L'Epave du Stella Maru

Description : Plongée de niveau 2 FFESSM ou PADI au arge de la barrière de Corail, départ à 8h30. Profondeur -30 mètres. Rascasses Volantes, murènes ...

[Plus d'infos](#)



Liste des Plongées SEO ...

localhost:3000/liste-des-plongees

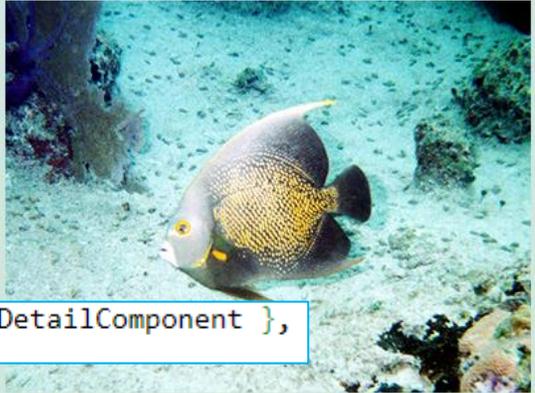
Plongée N° 1

Nom : Trou Aux Biches - L'Epave du Stella Maru

Description : Plongée de niveau 2 FFESSM ou PADI au arge de la barrière de Corail, départ à 8h30. Profondeur -30 mètres. Rascasses Volantes, murènes ...

Latitude : -20.0363

Longitude : 57.544700000000034



```
{ path: 'liste-des-plongees/:id', component: DivesDetailComponent },
```

Routing : Concepts avancés – QueryParams - Router

The image shows a code editor with two files open: `dives-list.component.html` and `dives-detail.component.ts`. The left sidebar shows a project structure for `TP04-routing`.

dives-list.component.html (lines 1-31):

```
1 <h1 class="alert alert-primary">Liste des Plongées</h1>
2
3 <div class="row">
4
5   <div class="col-md-6" *ngFor="let dive of dives">
6
7     <div class="card">
8       <!-- récupérer chaque image -->
9       
10      <div class="card-body">
11        <h3 class="card-title">Nom : {{dive.name}} </h3>
12        <p class="card-text">Description : {{dive.description}}</p>
13
14        <!-- <a
15         [routerLink]="[dive.id]"
16         [queryParams]="[dive.name]"
17         > -->
18        <!-- on interprète un des params de l'objet -->
19        <a
20         [routerLink]="[dive.id]"
21         [queryParams]="dive"
22         [skipLocationChange]="true"
23         >
24
25        <!-- on passe tout l'objet -->
26        <button class="btn btn-primary" #btnPlus>En Savoir Plus<
27
28      </div>
29    </div>
30  </div>
31 </div>
```

dives-detail.component.ts (lines 1-33):

```
1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute, Router } from '@angular/router';
3 import { Dives } from 'src/app/sharedModels/class/dives';
4
5 @Component({
6   selector: 'app-dives-detail',
7   templateUrl: './dives-detail.component.html',
8   styleUrls: ['./dives-detail.component.scss']
9 })
10 export class DivesDetailComponent implements OnInit {
11   // 1- props
12   public title: string;
13   public dive:Dives;
14
15   // 2- const
16   constructor(
17     private _routeActive: ActivatedRoute,
18     private _router:Router
19   ) {
20     this.title = '';
21     this.dive = new Dives();
22   }
23
24   // 3- lifeCycle
25   ngOnInit(): void {
26
27     const id = this._routeActive.snapshot.params['id'];
28     console.log(id);
29     this.title = `Plongée N° ${id}`;
30     // re-consommer le service ?
31     // ré-exploiter les datas stockées en storage ou en indexedDB
32     // Evidemment utiliser les fonctions de routage d'Angular ...
33 }
```

A red arrow points from the `[queryParams]="dive"` attribute in the HTML to the `private _routeActive: ActivatedRoute` property in the TypeScript constructor.

Routing : Concepts avancés – QueryParams - Router

The image shows a multi-pane IDE with the following content:

- divs-list.component.html**:

```
1 <h1 class="alert alert-primary">Liste des Plongées</h1>
2
3 <div class="row">
4
5   <div class="col-md-6" *ngFor="let dive of dives">
6
7     <div class="card">
8       <!-- récupérer chaque image -->
9       
10      <div class="card-body">
11        <h3 class="card-title">Nom : {{dive.name}} </h3>
12        <p class="card-text">Description : {{dive.description}}</p>
13
14        <!-- [routerLink]="[dive.id]" -->
15        <!-- Ecriture simplifiée du routerLink => routerLink="{{dive.id}}</h3>
16        <!-- routerLink="details/{{dive.id}}" -->
17        <!-- en chemin relatif -->
18        <button
19
20          routerLink="/liste-des-plongees/details/{{dive.id}}"
21          [queryParams]="dive"
22          [skipLocationChange]="true"
23
24          class="btn btn-primary" #btnPlus>En Savoir Plus</button>
25
26      </div>
27    </div>
28  </div>
```
- divs-details.component.ts**:

```
1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute } from '@angular/router';
3
4 @Component({
5   selector: 'app-divs-details',
6   templateUrl: './divs-details.component.html',
7   styleUrls: ['./divs-details.component.scss']
8 })
9 export class DivesDetailsComponent implements OnInit {
10
11   // -----
12   constructor(
13     private _routeActive: ActivatedRoute
14   ) {
15   }
16   // -----
17   ngOnInit() {
18     const id=this._routeActive.snapshot.params['param'];
19     console.clear();
20     console.log('ID récupéré : ', id);
21     // TP : passer l'ID à la Vue
22     // -----
23     this._routeActive.queryParams
24       .subscribe(
25         (params:any) => {
26           console.log('QueryParams : ', params);
27           // TP : passer l'objet récupéré à la vue
28         }
29       )
30   }
31 }
```
- divs-details.component.html**:

```
1 <h1>Détail de la plongée</h1>
2 <!-- TP afficher l'objet
3   passé par routing -->
```
- app-routing.module.ts**:

```
10 const routes: Routes = [
11
12   { path: '', component: BodyComponent }, // domain.tld/ (page de démar
13   { path: 'liste-des-plongees', component: DivesListComponent },
14
15   // { path:'accueil', component:BodyComponent}, // domain.tld/accdeil
16
17   // { path:'nouvelle-url', component:DivesListComponent},
18   // { path:'ancienne-url-seo-déjà-référencée', redirectTo:'nouvelle-url'
19
20   { path: 'liste-des-plongees/details/:param', component: DivesDetailsCon
21
22   { path: '**', component:Page404Component}
23
24 ]
```
- bdd.json**:

```
1 {
2   "dives": [
3     {
4       "id": 1,
5       "name": "Trou Aux Biches - L'Epave du Stella Maru",
6       "location": "Ile Maurice - Trou aux Biches - Nord/Ouest",
7       "level": 2,
8       "description": "Plongée de niveau 2 FFESSM ou PADI au arge de
9       "latitude": -20.0363,
10      "longitude": 57.544700000000034,
11      "evaluation": [
```
- divs.module.ts**:

```
1 import { NgModule } from '@
2 import { CommonModule } fro
3 import { DivesListComponent
4 import { HttpClientModule }
5 import { FormsModule } fro
6 import { RouterModule } fro
7
8
9
10
11 @NgModule({
12   declarations: [
```

Annotations and arrows in the image:

- Red arrows point from the `routerLink` and `[queryParams]` attributes in the HTML to the `private _routeActive: ActivatedRoute` property and the `ngOnInit` method in the TypeScript file.
- A blue dashed arrow points from the `params['param']` access in the TypeScript file to the `:param` placeholder in the routing configuration.

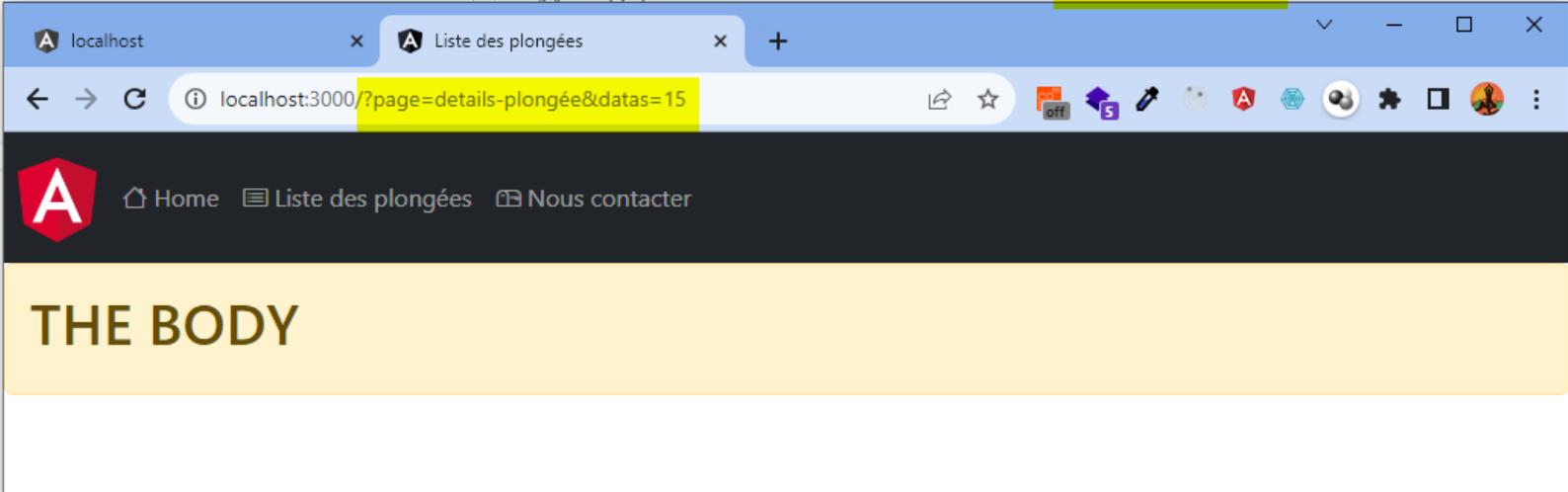
Routing : Concepts avancés – QueryParams - Router

```
dives-details.component.ts x
src > app > webApp > root > dives > composants > dives-details > dives-details.component.ts > DivesDetailsComponent > goBack
21 ngOnInit() {
22   const id = this._routeActive.snapshot.params['param'];
23   console.clear;
24   console.log('ID récupéré :', id);
25   // TP : passer l'ID à la Vue
26   // -----
27   this._routeActive.queryParams
28     .subscribe(
29     (params: any) => {
30       console.log('QueryParams : ', params);
31       // TP : passer l'objet récupéré à la vue
32       this.dive = params;
33     }
34   )
35 }
36 // -- Méthodes
37 public goBack = () => {
38   this._router.navigateByUr1('liste-des-plongees');
39 }
40
41 public goHome = () => {
42   this._router.navigate(
43     [''],
44     {
45       queryParams: {
46         page: 'details-plongée',
47         datas: 15
48       }
49     }
50 )

```

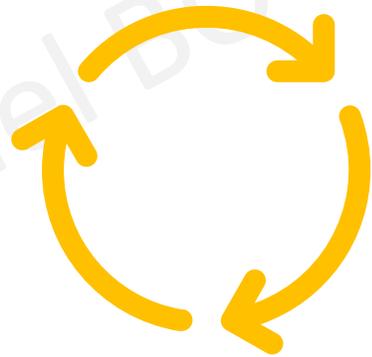
```
dives-details.component.html x
TP04-routage > src > app > webApp > root > dives > composants > dives-details > dives-details.component.html > p > button
1 <h1>Détail de la plongée N° {{dive.id}}</h1>
2 <!-- TP afficher l'objet passé par routage -->
3
4 <!-- Single way binding -->
5 <!-- TS => HTML : [directive] et {{string interpolation}} -->
6 <!-- HTML => TS : (event du DOM) par ex : (click)-->
7 <p>
8   <button class="btn btn-warning" (click)="goBack()">Go Back</button>
9
10  <button class="btn btn-danger" routerLink="/liste-des-plongees">RouterLink</button>
11 </p>
12 <p>
13  <button class="btn btn-success" (click)="goHome()">Go Home</button>

```



The browser screenshot shows a navigation menu with three items: Home, Liste des plongées, and Nous contacter. Below the menu is a large yellow rectangular area with the text "THE BODY". The browser's address bar shows the URL localhost:3000/?page=details-plongée&datas=15.

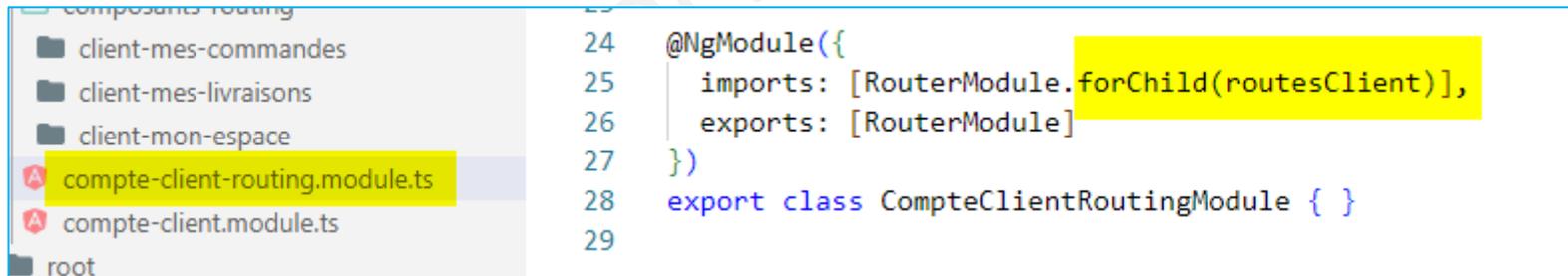
Routage Avancé
Lazy Loading
Optimisation du build



Copyright Michel BOCCIOLESI - ORSYS

Routing : Concepts avancés – Le Lazy Loading

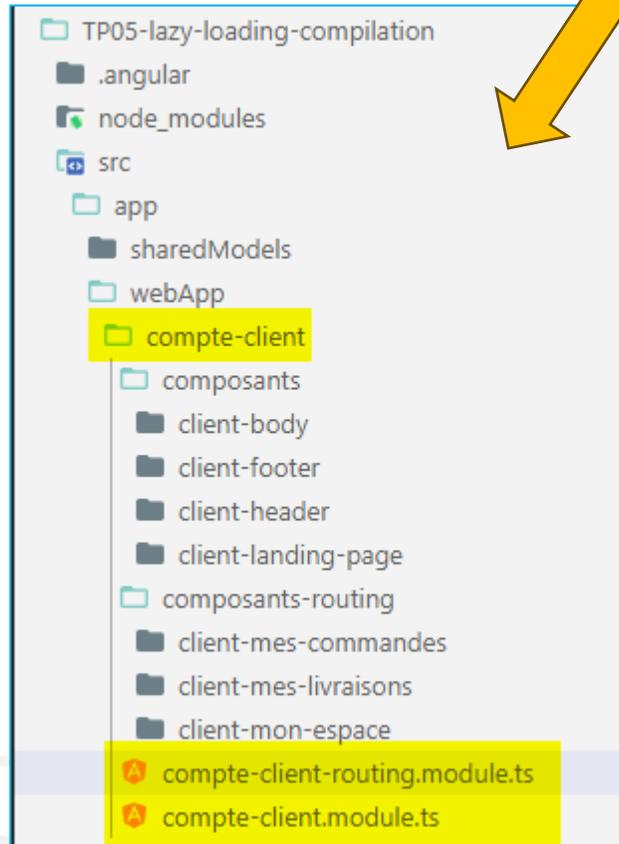
Créer le module compte-client avec l'option « --routing »
Cette option crée un module de routage spécifique
forChild



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

Routing : Concepts avancés – Le Lazy Loading

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



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

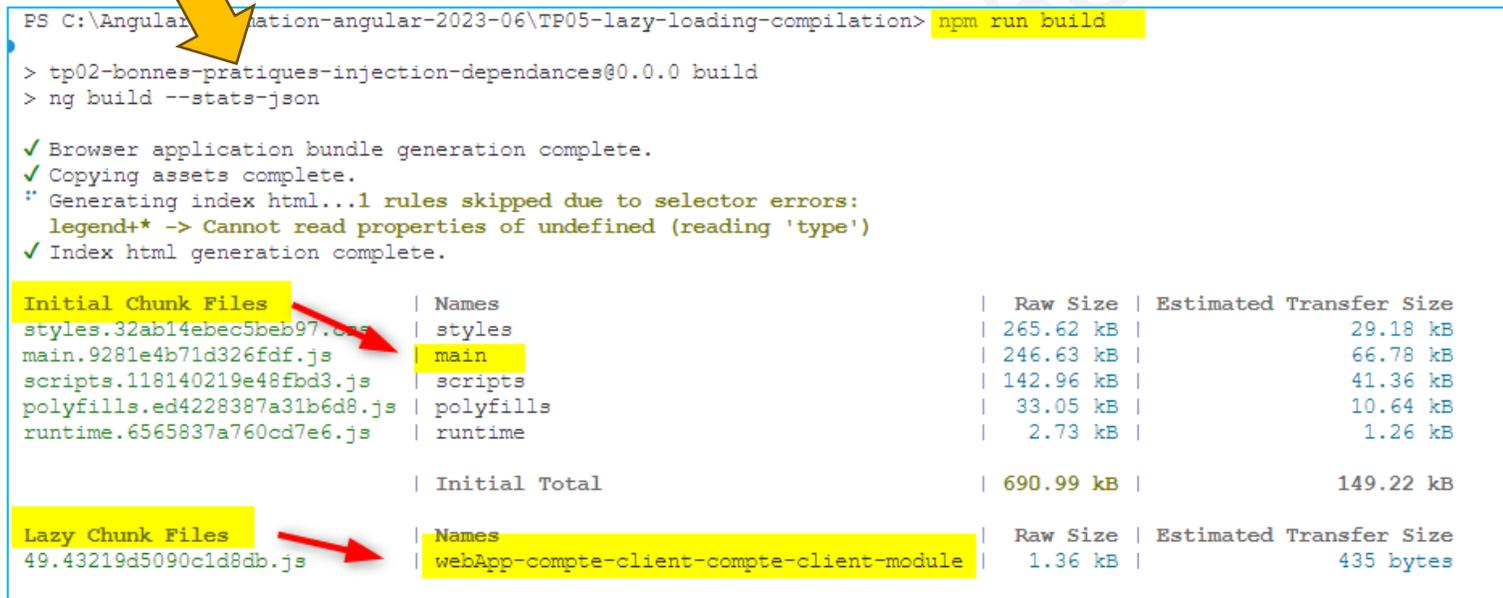
```
composants-routing
├── client-mes-commandes
├── client-mes-livraisons
├── client-mon-espace
├── compte-client-routing.module.ts
├── compte-client.module.ts
└── root
```

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

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-angulaire-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

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)

HEADER : Espace Client

Mon Espace Client

- Voir mon compte ...
- Voir mes livraisons ...
- Voir mes commandes ...

client-mon-espace works!

FOOTER : Espace Client

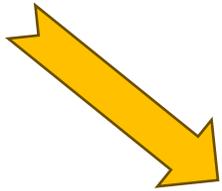
Copyright

ORSY

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)

The image shows a code editor with four panels displaying Angular code:

- 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
```
- 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 ];
```
- 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 })
35 export class AccueilModule {
36 }
37
```
- app-routing.module.ts**:

```
9 import { ClientLandingPageComponent } from './webApp/formation/compte-client/compte-client/
10
11 // -----
12 import { routesClient } from './webApp/formation/compte-client/compte-client/routing
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 ]
27
```

Red arrows indicate the following relationships:

- From the `CompteClientModule` import in `accueil.module.ts` to the `ClientLandingPageComponent` and `routesClient` in `compte-client-routing.module.ts`.
- From the `routesClient` array in `compte-client-routing.module.ts` to the `routesClient` import in `app-routing.module.ts`.
- From the `routesClient` array in `compte-client-routing.module.ts` to the `children: routesClient` property in `app-routing.module.ts`.

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 })
30 export class AccueilModule {}

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 ×
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 })
```

Le Lazy Loading : Standalone Components

1 seul fichier de routage 😊

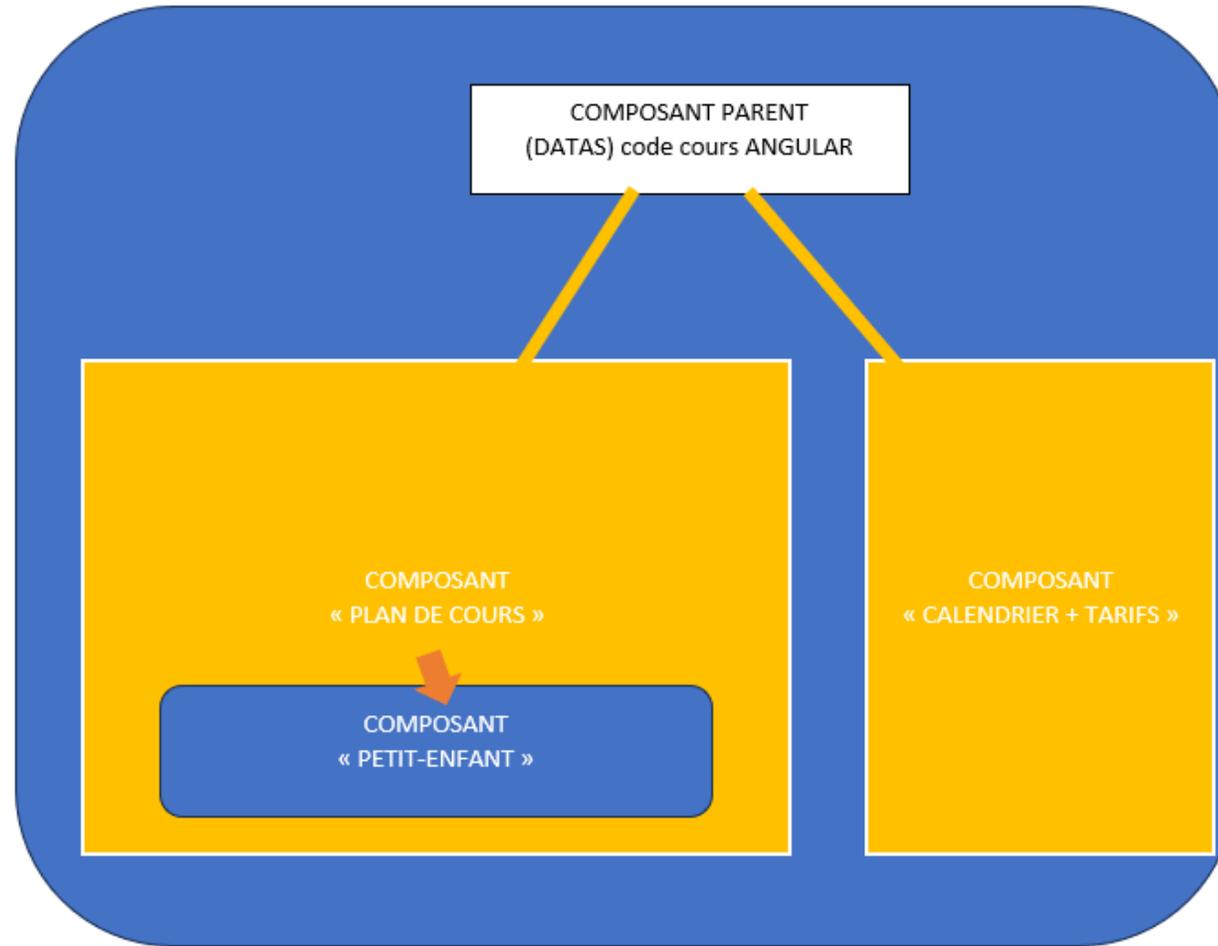
```
TS app.routes.ts ×
TP05-communication-composants > src > app > TS app.routes.ts > ...
1 import { Routes } from '@angular/router';
2 import { HomeComponent } from './webApp/accueil/home/home.component';
3 import { CommandesComponent } from './webApp/compte-client/components-routing/commandes/commandes.component';
4 import { LivraisonsComponent } from './webApp/compte-client/components-routing/livraisons/livraisons.component';
5 import { SignalsComponent } from './webApp/signals/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 { DivesDetailComponent } from './webApp/dives-list/components/dives-detail/dives-detail.component';
10 import { TpComposantsComponent } from './webApp/tp-composants/components/tp-composants/tp-composants.component';
11
12
13 export const routes: Routes = [
14   { path: '', component: HomeComponent, title: 'Angular 18 - StandAlone Components' },
15   { path: 'liste-des-plongees', component: DivesListComponent, title: 'Liste des plongees' },
16   { path: 'liste-des-plongees/:id', component: DivesDetailComponent },
17   { path: 'les-signals-angular-17', component: SignalsComponent, title: 'Les signals Angulars 17 - trop Cool' },
18   { path: 'tp-composants', component: TpComposantsComponent, title: '😎' },
19
20   // Avec le Lazy Loading Component 😊
21   {
22     path: 'compte-client', title: 'Mon espace client - Formation Angular',
23     loadComponent:
24       () => import('./webApp/compte-client/components/compte-client/compte-client.component')
25         .then(
26           (c) => c.EntryCompteClientComponent
27         ),
28     children: [
29       { path: 'mes-commandes', component: CommandesComponent },
30       { path: 'mes-livraisons', component: LivraisonsComponent },
31     ]
32   },
33   // TP ajouter le composant ContactsComponent en lazy Loading
34   // { path: 'nous-contactez', component: ContactsComponent, title: 'Wow Génial les formulaires NG 🚗' },
35
36   // {path:'**', redirectTo:'',title:'Page Introuvable !'}
37   { path: '**', component: Page404Component, title: 'Page Introuvable !' }
38 ];
39
```

Input - Output



Copyright Michel FOLCIERESI - ORSYS

Communication : composants parents et enfants



ORSYS

Cop.

Communication : composants parents et enfants

TP : déployer une architecture de composants dans la partie « dives » comme illustré ci-dessous

The diagram illustrates the deployment of a component architecture in the 'dives' section. It shows a file explorer on the left and a browser screenshot on the right.

File Explorer (Left):

- webApp
 - compte-client
 - panier
 - root
 - accueil
 - dives
 - composants
 - dives-detail
 - dives-list
 - composants-enfants** (highlighted in yellow)
 - dives-list-child

Files listed below 'dives-list-child':

- dives-list-child.component.html
- dives-list-child.component.scss
- dives-list-child.component.ts

Other files: dives.module.ts

Browser Screenshot (Right):

URL: localhost:3000/liste-des-plongees

Page Structure:

- Router Outlet #1(primary)
 - Liste des Plongées
 - Filtrer l'affichage
 - Filter
 - Panier
 - Nom : Trou Aux Biches - L'Epave du Stella Maru
 - Composant Enfant : Dives-List-Child
 - Lieu : Ile Maurice - Trou aux Biches - Nord/Ouest
 - Description : Plongée de niveau 2 FFESSM ou PADI au arge de la barrière de Corail, départ à 8h30. Profondeur -30 mètres. Rascasses Volantes, murènes ...
 - Nom : Réserve COUSTEAU - KARUKERA
 - Composant Enfant : Dives-List-Child
 - Lieu : DOM GUADELOUPE - Basse Terre
 - Description : Plongée de niveau 1 FFESSM ou PADI dans la baie de Bouillante, départ à 10h.Profondeur -10 à -25 mètres.

Red arrows point from the 'dives-list-child' component in the file explorer to the 'Composant Enfant : Dives-List-Child' sections in the browser. Yellow arrows point from the 'webApp' directory to the browser and from the 'dives' directory to the 'Liste des Plongées' section.

Communication : composants parents et enfants

TP : Chaque composant enfant généré par la boucle *ngFor est représenté dans une classe alert-warning.

Pour faire passer une données de la vue HTML du composant parent vers le TS du composant enfant, nous utilisons les `@Input()` qui sont toujours définis dans le composant enfant.

Cet input est utilisé par le parent comme une directive

```
<!-- Composant Enfant -->
<div class="alert alert-warning">

  <p>Composant Enfant : Dives-List-Child #2</p>
  <app-dives-list-child2
    [inputDive]="inputDive"
  ></app-dives-list-child2>

</div>
<!-- Composant Enfant -->
```

```
9  export class DivesListChildComponent {
10
11  // les @input et les @output permettent d'établir la communication entre comp parents et enfant(s)
12  // ils se définissent toujours dans l'enfant !
13
14  // 1- props
15  // décorateur  nom_du_décorateur:type
16  // le nom du décorateur (devient) une directive dans le parent
17  @Input() inputDive!:Dives;
18  @Input() inputInfos!:string;
19  // -----
20  @Output() outputEventDive:EventEmitter<any>;
21
```

The screenshot shows a web application interface. At the top, there is a navigation bar with a logo 'A' and links for 'Home', 'Liste des plongées', 'Nous contacter', and 'Se connecter (espace)'. Below this is a green header with the word 'Panier'. The main content area is titled 'Nom : Trou Aux Biches - L'Epave du Stella Maru'. It contains several light blue boxes with text: 'Composant Enfant : Dives-List-Child', 'Lieu : Ile Maurice - Trou aux Biches - Nord/Ouest', 'Description : Plongée de niveau 2 FFESSM ou PADI au arge de la barrière de Corail, départ à 8h30. Profondeur -30 mètres. Rascasses Volantes, murènes ...', 'Latitude : -20.0363', 'Longitude : 57.544700000000034', and 'Niveau : 2'. Below these is another 'Composant Enfant : Dives-List-Child #2' section which contains a photograph of a fish swimming underwater.

Communication : composants parents et enfants

The image displays a code editor with four files open, illustrating the communication between parent and child components in an Angular application. Red arrows highlight the data flow.

- dives-list.component.ts** (Parent Component):
 - Line 114: `// 4- méthodes`
 - Line 115: `public diveSelected = (e: any) => {`
 - Line 116: `console.log('depuis le parent $event => ',e);`
 - Line 117:
 - Line 118: `let isChecked=e.paramIsChecked;`
 - Line 119: `let diveSelect=e.paramDive;`
 - Line 120:
 - Line 121: `console.log(`Plongée : ${diveSelect.name} - Etat : ${isChecked}`);`
 - Line 122:
 - Line 123: `if (isChecked) {`
 - Line 124: `this.panier.push(diveSelect);`
 - Line 125: `console.table(this.panier);`
 - Line 126: `} else {`
 - Line 127: `let keyPanier = this.panier.indexOf(diveSelect);`
 - Line 128: `if (keyPanier >= 0) {`
 - Line 129: `this.panier.splice(keyPanier, 1);`
 - Line 130: `console.table(this.panier);`
 - Line 131: `}`
 - Line 132: `}`
 - Line 133:
 - Line 134: `}`
 - Line 135:
 - Line 136:
- dives-list-child.component.ts** (Child Component):
 - Line 9: `export class DivesListChildComponent {`
 - Line 10:
 - Line 11: `// les @input et les @output permettent d'établir la communication`
 - Line 12: `// ils se définissent toujours dans l'enfant !`
 - Line 13:
 - Line 14: `// 1- props`
 - Line 15: `// décorateur nom_du_décorateur:type`
 - Line 16: `// le nom du décorateur (devient) une directive dans le template`
 - Line 17: `@Input() inputDive!:Dives;`
 - Line 18: `@Input() inputInfos!:string;`
 - Line 19: `// -----`
 - Line 20: `@Output() outputEventDive:EventEmitter<any>;`
 - Line 21:
 - Line 22: `// 2- const`
 - Line 23: `constructor(){`
 - Line 24: `this.outputEventDive = new EventEmitter();`
 - Line 25: `}`
 - Line 26:
 - Line 27: `// 3- méthodes`
 - Line 28: `public choixDive = (e:any) => {`
 - Line 29: `console.clear();`
 - Line 30: `console.log(e.target.checked);`
- dives-list-child2.component.ts** (Child Component):
 - Line 1: `import { Component, Input } from '@angular/core';`
 - Line 2: `import { Dives } from 'src/app/sharedModels/class';`
 - Line 3:
 - Line 4: `@Component({`
 - Line 5: `selector: 'app-dives-list-child2',`
 - Line 6: `templateUrl: './dives-list-child2.component.html',`
 - Line 7: `styleUrls: ['./dives-list-child2.component.scss']`
 - Line 8: `})`
 - Line 9: `export class DivesListChild2Component {`
 - Line 10:
 - Line 11: `@Input() inputDive!:Dives;`
 - Line 12:
 - Line 13: `}`
 - Line 14:
- dives-list.component.html** (Parent Template):
 - Line 41: `<div class="card-body">`
 - Line 42: `<h4 class="card-title">Nom : {{dive.name}} </h4>`
 - Line 43: `<!-- Composant Enfant -->`
 - Line 44: `<div class="alert alert-warning">`
 - Line 45: `<p>Composant Enfant : Dives-List-Child</p>`
 - Line 46: `<app-dives-list-child`
 - Line 47: `[inputDive]="dive"`
 - Line 48: `[inputInfos]="infos"`
 - Line 49: `(outputEventDive)="diveSelected($event)"`
 - Line 50: `>>/app-dives-list-child`
 - Line 51: `</div>`
 - Line 52: `<!-- Composant Enfant -->`
 - Line 53:
 - Line 54:
 - Line 55:
 - Line 56:
- dives-list-child.component.html** (Child Template):
 - Line 14: `<div class="alert alert-warning">`
 - Line 15: `<p>Composant Enfant : Dives-List-Child #2</p>`
 - Line 16: `<app-dives-list-child2`
 - Line 17: `[inputDive]="inputDive"`
 - Line 18: `>>/app-dives-list-child2`
 - Line 19: `</div>`
 - Line 20: `<!-- Composant Enfant -->`
 - Line 21:
 - Line 22:
 - Line 23:
 - Line 24:
 - Line 25: `<p class="alert alert-primary">`
 - Line 26: `<input type="checkbox" (change)="choixDive($event)">`
 - Line 27: `</p>`
 - Line 28: `<!-- $event est l'objet global evenement -->`
 - Line 29:
- dives-list-child2.component.html** (Child2 Template):
 - Line 1: `<p class="alert alert-primary">`
 - Line 2: ``
 - Line 3: `</p>`
 - Line 4: `</p>`
 - Line 5:

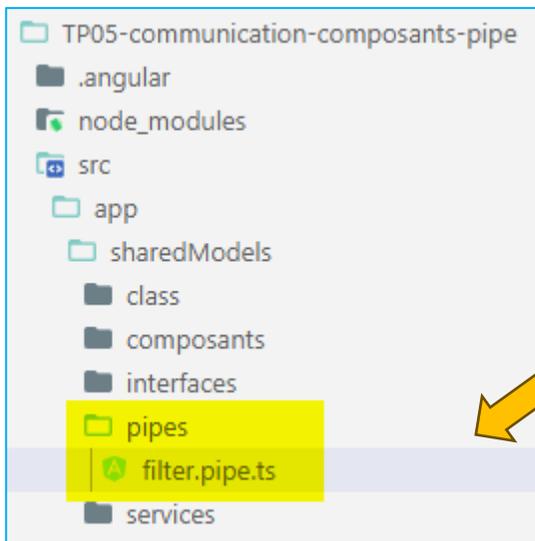
Red arrows indicate the flow of data: from the parent's `diveSelected` method to the child's `choixDive` method, and from the child's `inputDive` back to the parent's `diveSelected` method.

Pipes Angular



Copyright Michel POCCIOLESI - ORSYS

Les PIPES Angular



Créer ses propres Pipes avec la commande :

`ng g(enerate) p(ipe)`

Ce pipe va permettre de filtrer l'affichage des datas en fonction des caractères saisis.

Exemple : Agay



Les PIPES Angular

The image shows a code editor with three files open, illustrating the implementation and usage of an Angular pipe.

```
divides-list.component.ts
16 // -----
17 public dives: Dives[] = [];
18 @ViewChild('btnPlus') eltCollectionBtn!: QueryList<ElementRef>;
19 public infos:string='Orsys Formation ...';
20 public panier: Dives[] = [];
21 public texteSaisi:string='';
22
```

```
filter.pipe.ts
1 import { Pipe, PipeTransform } from '@angular/core';
2 import { Dives } from '../class/dives';
3
4 @Pipe({
5   name: 'filter'
6 })
7 // la méthode transform admet 2 paramètres
8 // le 1er : la data à transformer
9 // le second: le motif de transformation
10
11 // une date(data) | (nomduPipe) date:'dd/MM/y' (motif)
12 // myDate | date:'dd/MM/y HH:mm:ss'
13 // ... dives | filter:texteSaisi
14
15 export class FilterPipe implements PipeTransform {
16
17   transform(datas:Dives[], motif:string) {
18
19     if (motif==='') {
20       // on a rien saisi ou tout effacé
21       return datas;
22     }
23     else {
24       // on a saisi des caractères
25       return datas.filter(
26         (paramDive :Dives) => {
27           return paramDive.name.toLocaleLowerCase().includes(motif.
28             toLocaleLowerCase()) || paramDive.description.toLocaleLowerCase().
29             includes(motif.toLocaleLowerCase())
30         }
31       )
32     }
33   }
34 }
35
```

```
divides.module.ts
14 declarations: [
15   DivesListComponent,
16   DivesDetailComponent,
17   DivesListChildComponent,
18   DivesListChild2Component,
19   // *****
20   // on déclare AUSSI les pipes
21   FilterPipe
22 ],
23 imports: [
24   CommonModule,
25   HttpClientModule,
26   RouterModule,
27   FormsModule
28 ],
29 exports: [
30
```

```
divides-list.component.html
11 </div>
12 </div>
13 </div>
14
15 <!-- Filtre : PIPE -->
16 <div class="row">
17   <div class="alert alert-warning">
18     <h2>Filtre l'affichage</h2>
19     <!-- template Driven Forms => utilisation de la directive [ngModel] p
20     <!-- single Way Binding [ngModel] : TS => HTML -->
21     <!-- single Way Binding (ngModel) : HTML => TS -->
22     <!-- Double Way Binding -->
23     <!-- [()] banana in a box -->
24     <input type="text" class="form-control" [(ngModel)]="texteSaisi">
25   </div>
26 </div>
27
28 <div class="row">
29   <div class="col-md-3" *ngFor="let dive of dives | filter:texteSaisi">
30
31     <div class="card">
32
33       <div class="card-body">
34         <h3 class="card-title">Nom : {{dive.name}}</h3>
35
36       </div>
37     </div>
38   </div>
39 </div>
```

Annotations in the image:

- Yellow boxes highlight: `name: 'filter'`, `transform(datas:Dives[], motif:string)`, `FilterPipe`, `FormsModule`, `texteSaisi`, and `filter:texteSaisi`.
- Blue arrows point from the pipe name in the HTML template to the pipe definition in `filter.pipe.ts`.
- Blue arrows point from the pipe name in the HTML template to the `FilterPipe` class in `divides.module.ts`.

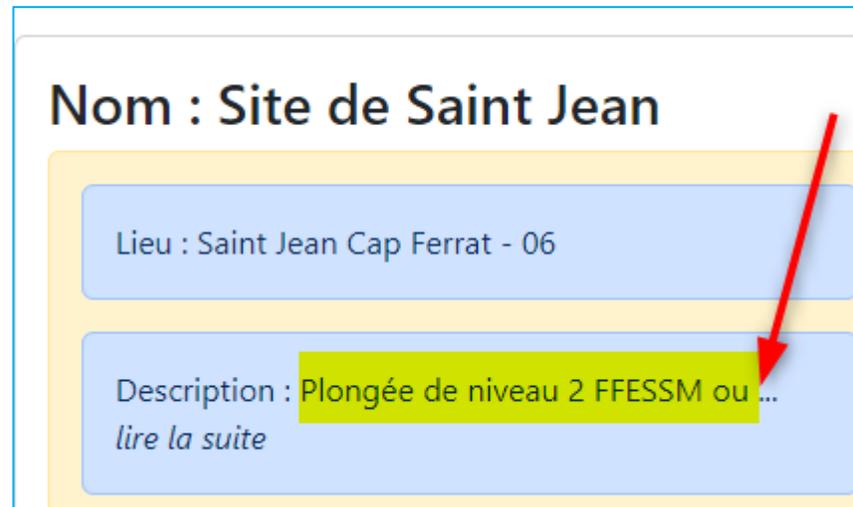


TP : Créer un pipe qui affiche seulement les 20 premiers caractères de la description avec la méthode javascript substring()

Nom : Site de Saint Jean

Lieu : Saint Jean Cap Ferrat - 06

Description : Plongée de niveau 2 FFESSM ou ...
lire la suite



Copyright

OLESI - ORSYS



TP : Créer un pipe qui affiche seulement les x premiers caractères (20 par ex) et ajoute un petit texte (ex : ... lire la suite) de la **description** avec la méthode javascript substring()

```
divides.module.ts
TP05-communication-composants-pipe > src > app > webApp > root > dives > dives.module.ts > DivesModule
11 import { VoirPlusPipe } from 'src/app/sharedModels/pipes/voir-plus.pipe';
12
13
14 @NgModule({
15   declarations: [
16     DivesListComponent,
17     DivesDetailComponent,
18     DivesListChildComponent,
19     DivesListChild2Component,
20     // *****
21     // on déclare AUSSI les pipes
22     FilterPipe,
23     VoirPlusPipe
24   ],
25   imports: [
26     CommonModule,
27     HttpClientModule,
28   ]
29 })
30 export class DivesModule {}
```

```
voir-plus.pipe.ts
TP05-communication-composants-pipe > src > app > sharedModels > pipes > voir-plus.pipe.ts > VoirPlusPipe > transform
1 import { Pipe, PipeTransform } from '@angular/core';
2
3 @Pipe({
4   name: 'voirPlus'
5 })
6 export class VoirPlusPipe implements PipeTransform {
7
8   transform(datas:string, nbCars:number) {
9     return `${datas.substring(0, nbCars)} ...`;
10  }
11
12 }
13
```

```
divides-list-child.component.html
communication-composants-pipe > src > app > webApp > root > dives > composants-enfant > dives-list-child > dives-list-child.component.html > p.alert
1 <p class="alert alert-primary">Lieu : {{inputDive.location}}</p>
2
3
4 <p class="alert alert-primary">Description : {{inputDive.description | voirPlus:30}} <em>lire la suite</em></p>
5
6
7 <p class="alert alert-primary">Latitude : {{inputDive.latitude}}</p>
8 <p class="alert alert-primary">Longitude : {{inputDive.longitude}}</p>
9 <p class="alert alert-primary">Niveau : {{inputDive.level}}</p>
10
11 <p class="alert alert-danger">{{inputInfos}}</p>
12
13 <!-- appel enfant -->
14 <app-dives-list-child2
15   [inputDive2]="inputDive"
16 ></app-dives-list-child2>
17
18
19
20 <p class="alert alert-primary">
21   <input type="checkbox" (change)="choixDive($event)">
22   <!-- $event = EVENT GLOBAL -->
23 </p>
24
```

```
PS C:\Users\Michel\Desktop\Formation-angular-alteca-10-2023\TP05-communication-composants-pipe\src\app\sharedModels\pipes> ng g p voir-plus --skip-tests --skip-import
```

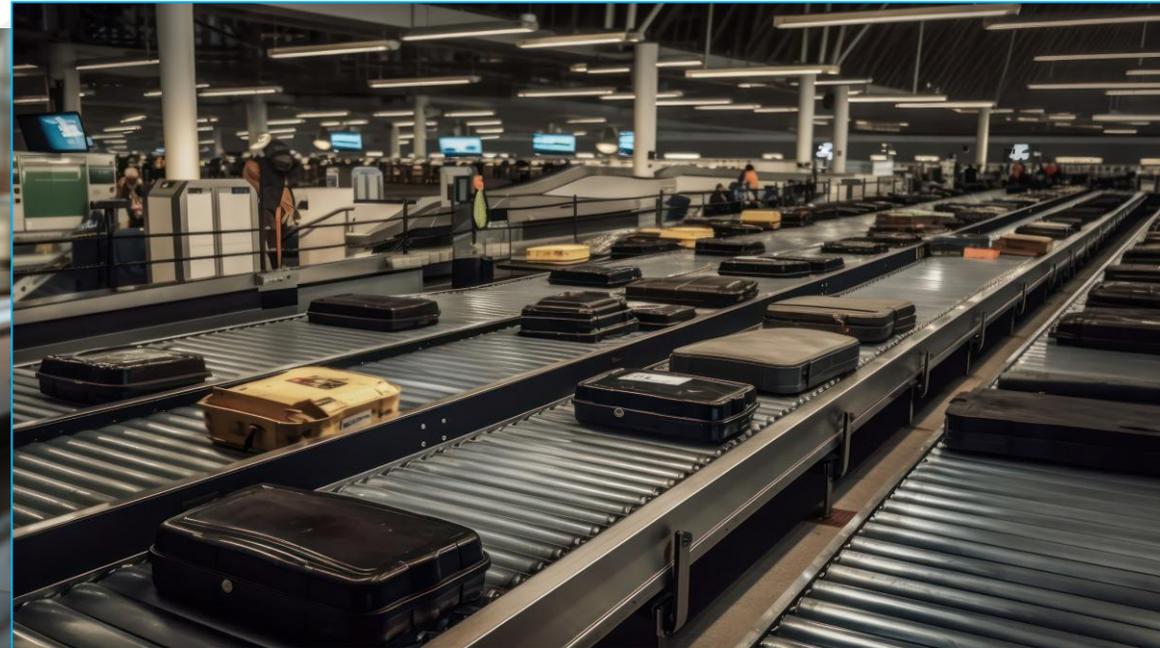
Rxjs – Observables
Approfondissement
Niveau 2



Copyright Michel FACCIOLESI - 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.



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)

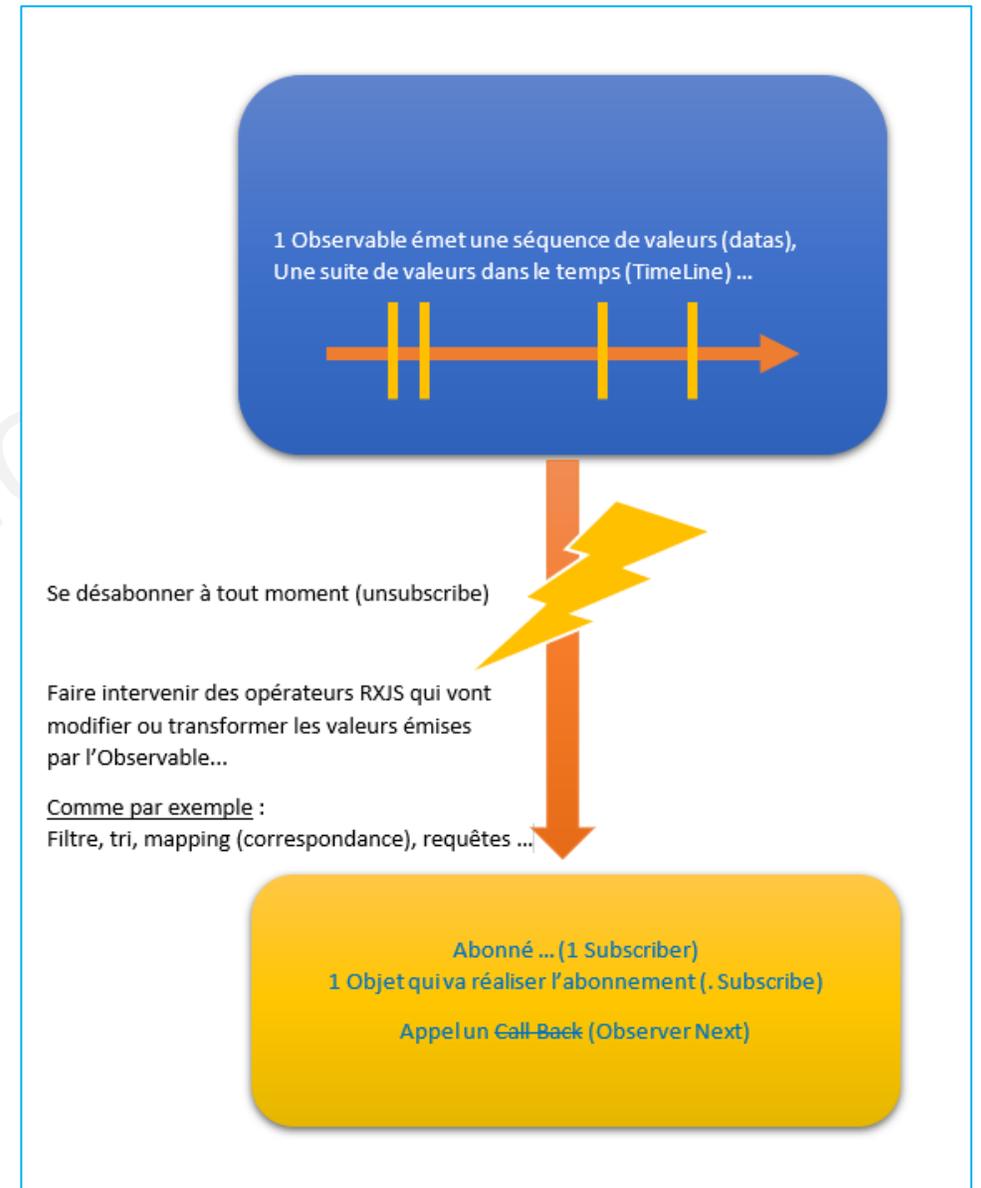
Opérateurs : des fonctions tap, map, merge, filter qui vont pouvoir modifier/transformer ce flux de valeurs émises par l'observable

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 !

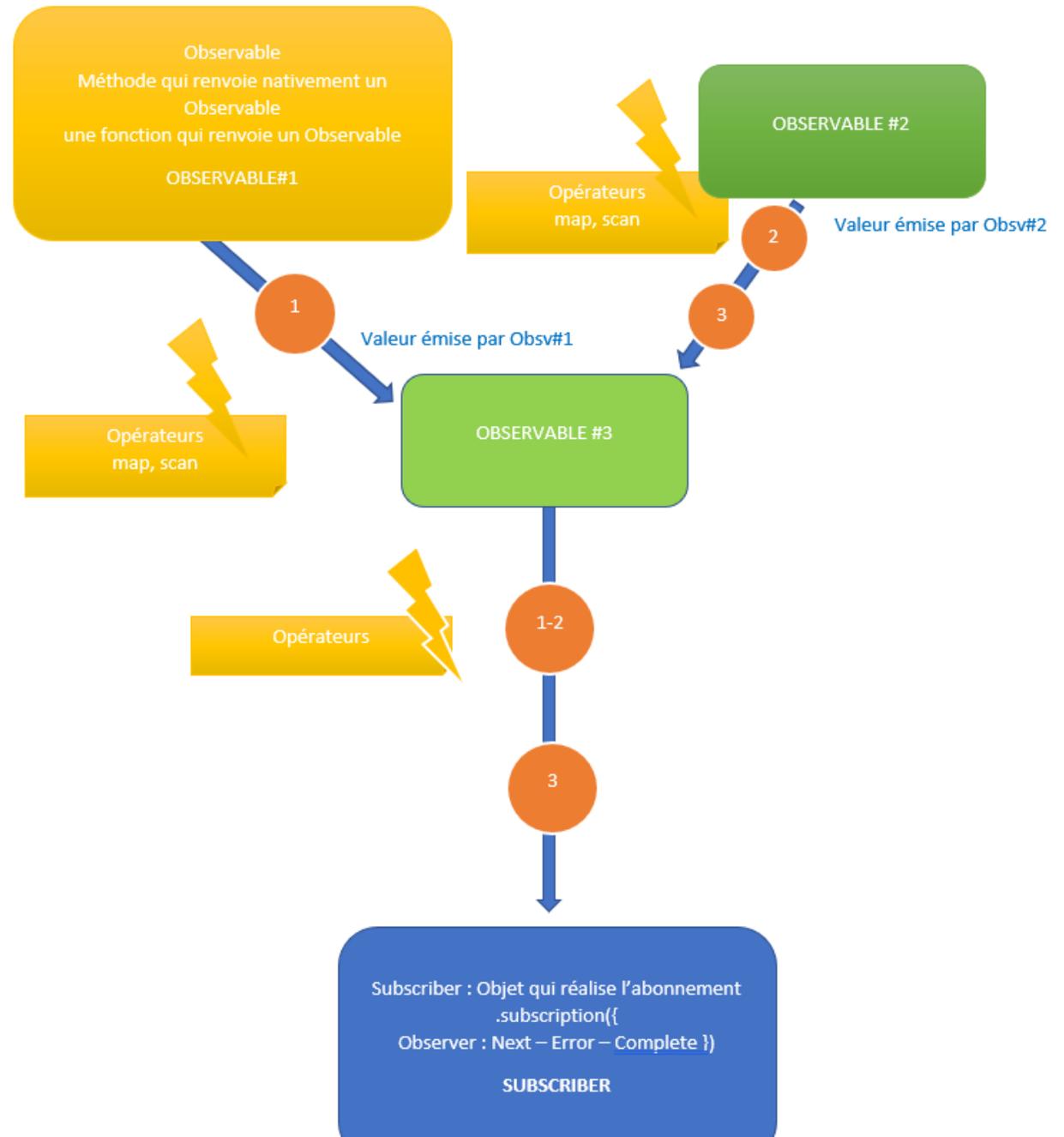
Les séquences de valeurs émises sont facilement annulables !

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



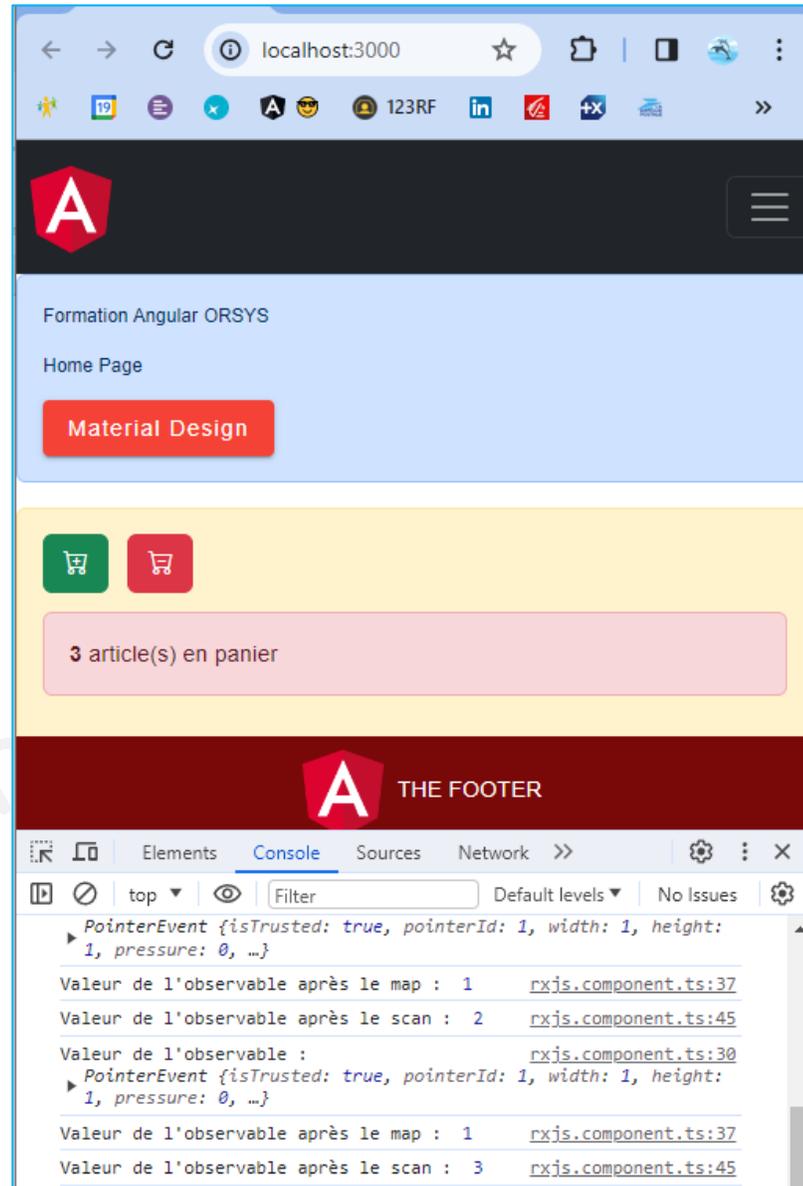
RXJS et les OBSERVABLES

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



Copyright Micho

RXJS et les OBSERVABLES



The screenshot shows a web browser at localhost:3000 displaying an Angular application. The page has a dark header with an 'A' logo and a menu icon. Below the header, there's a light blue section with the text 'Formation Angular ORSYS' and 'Home Page', and a red button labeled 'Material Design'. A yellow section contains two shopping cart icons and a pink box stating '3 article(s) en panier'. The footer is dark red with the 'A' logo and the text 'THE FOOTER'. The browser's developer console is open, showing a log of events:

```
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`;

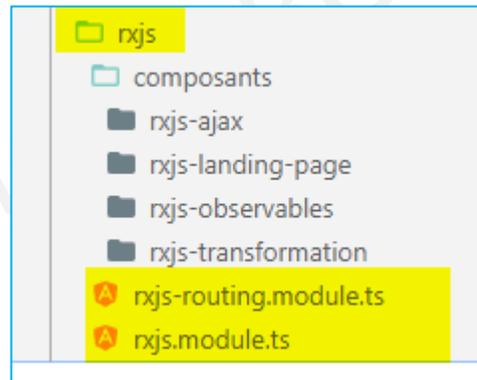
      if (valObs === 0) {
        // this.eltMoins.nativeElement.setAttribute('disabled', 'true');
        this.eltMoins.nativeElement.style.display = 'none';
      }
    }
  );

```

Copyright Michel

RXJS et les OBSERVABLES

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



```
<button class="btn btn-warning" (click)="createObservable1()">Créer un Observable</button>
<ul id="formations" class="list-group"></ul>

<p></p>

<button class="btn btn-success" (click)="createObservable2()">Créer un Observable</button>
<!-- template référence Angular # -->
<p></p>
<ul #formations2 class="list-group"></ul>
```

```
// 4- Methodes
public createObservable1: any = () => {

  const eltUl = <HTMLElement>document.querySelector('#formations');
  const formationsArray: string[] = ['NG15', 'REACT 17', 'PWA', 'XML', 'JSON',

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

  // abonnement à cet observable
  this.subArrayFormation = formation$.subscribe(
    // code OK success
    // la notion d'observer : ext - error - complete
    {
      next:
        (formation: string) => {
          console.log(formation);
          const eltLi = <HTMLElement>document.createElement('li');
          eltLi.innerHTML = formation;
          eltLi.className = 'list-group-item';
          eltUl.appendChild(eltLi);
        }
      ,
      error:
        (e) => {
          console.log(e);
        }
      ,
      complete:
        () => console.warn('Complete')
    }
  );
}
```

Copyright W

```

// -----
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),

```

```

128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
    delay(3000),
    filter(
      (formation: Formation) => {
        return formation.duree === 4;
      }
    ),
    tap(
      // observer NEXT
      (formation: Formation) => {
        console.warn(formation);
        const eltLi = <HTMLElement>document.createElement('li');
        eltLi.innerHTML = `${formation.cours} : ${formation.duree} jours.`;
        eltLi.className = 'list-group-item';
        // nativeElement est obligatoire ici car on récupère l'élément par Vie
        this.eltUl2.nativeElement.appendChild(eltLi);
      }
    ),
    catchError(
      // capturer les erreurs de l'observable
      (e: any): any => {
        console.log(e);
      }
    )
  )
  .subscribe(
    // // observer NEXT
    // (formation: Formation) => {
    //   console.log(formation);
    //   const eltLi = <HTMLElement>document.createElement('li');
    //   eltLi.innerHTML = `${formation.cours} : ${formation.duree} jours.`;
    //   eltLi.className = 'list-group-item';
    //   // nativeElement est obligatoire ici car on récupère l'élément par View
    //   this.eltUl2.nativeElement.appendChild(eltLi);
    // }
  )
  .unsubscribe();

```

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)

Angular est un framework « **orienté** de ce fait beaucoup de packages peuvent s'installer grâce à la commande « **ng add** »

ng add @angular/material

ng add @angular/localize (I18N : internationalisation)

ng add @angular/pwa (Progressive web app)

Ng add @ngrx/store (Le store Angular)

Les formulaires Angular (avec Matériel Design)

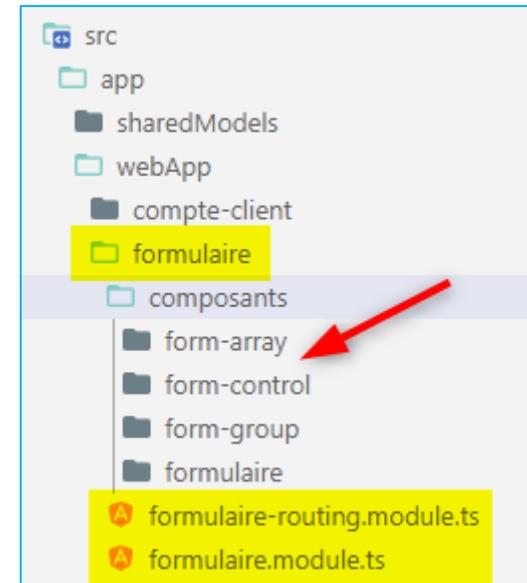
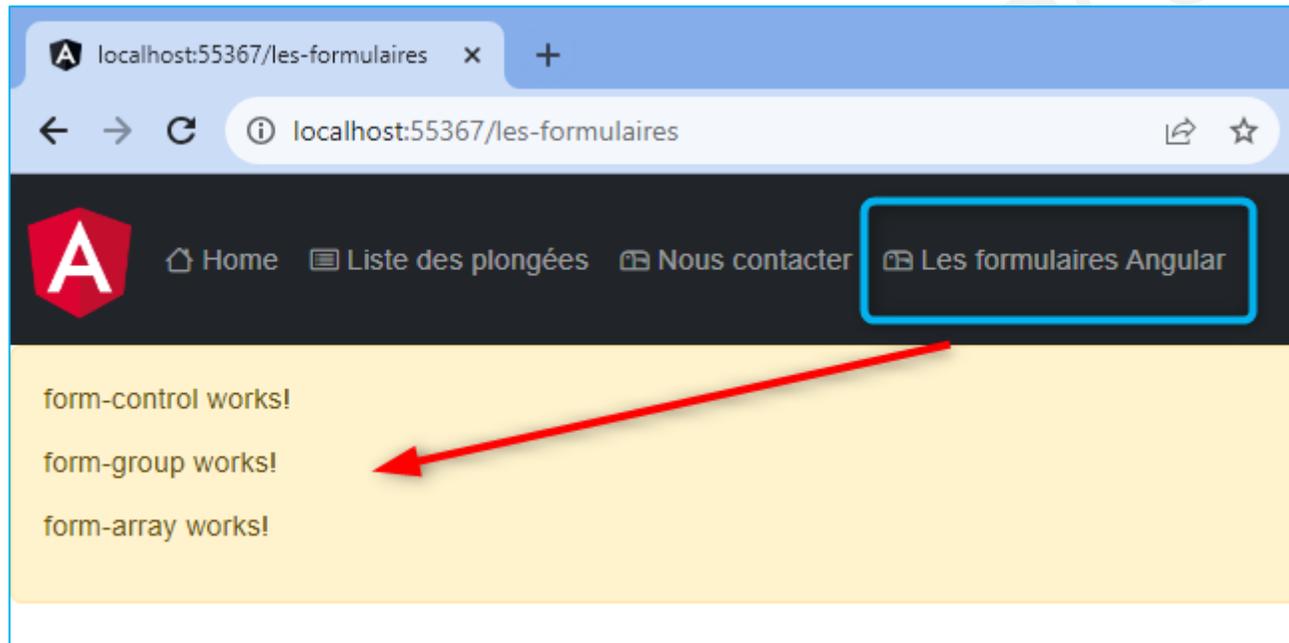
<https://dev.webjs.fr/1-Angular/ressources-formation/material.module.ts>

<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 (avec Matériel Design)

The image displays a code editor with three files open:

- formulaires.component.html** (lines 1-6):

```
1 <div class="alert alert-warning">
2   <app-form-control></app-form-control>
3   <app-form-group></app-form-group>
4   <app-form-array></app-form-array>
5 </div>
6
```
- header.component.html** (lines 16-38):

```
16 ><i class="bi bi-house"></i> Home</a>
17 </li>
18 <li class="nav-item">
19   <a class="nav-link"
20
21     routerLink="liste-des-plongees"
22
23   ><i class="bi bi-card-list"></i> Liste des plongées</a>
24 </li>
25 <li class="nav-item">
26   <a class="nav-link"
27     routerLink="contactez-nous"
28   ><i class="bi bi-mailbox"></i> Nous contacter</a>
29 </li>
30
31 <li class="nav-item">
32   <a class="nav-link"
33     routerLink="les-formulaires"
34   ><i class="bi bi-mailbox"></i> Les formulaires Angular</a>
35 </li>
36 </ul>
37 </div>
38 </nav>
```
- accueil.module.ts** (lines 6-27):

```
6 import { BodyComponent } from './composants/body/body.component';
7 import { DivesModule } from '../dives/dives.module';
8 import { RouterModule } from '@angular/router';
9 import { FormulairesModule } from '../../formulaires/formulaires.module';
10
11 @NgModule({
12   declarations: [
13     LandingPageComponent,
14     HeaderComponent,
15     FooterComponent,
16     BodyComponent
17   ],
18   imports: [
19     CommonModule,
20     RouterModule,
21     // nos propres modules
22     DivesModule,
23     FormulairesModule
24   ],
25   exports: [
26     LandingPageComponent,
27     // HeaderComponent
28   ]
29 })
30 export class AccueilModule {}
```
- app-routing.module.ts** (lines 4-21):

```
4 import { BodyComponent } from './webApp/root/accueil/composants/body/body.comp
5 import { ContactsComponent } from './webApp/root/contacts/composants/contacts/
6 import { Page404Component } from './sharedModels/composants/page404/page404.co
7 import { DivesDetailComponent } from './webApp/root/dives/composants/dives-det
8 import { FormulairesComponent } from './webApp/formulaires/composants/formulai
9
10 const routes: Routes = [
11   { path: '', component: BodyComponent }, // domain.tld
12   { path: 'accueil', component: BodyComponent },
13   { path: 'liste-des-plongees', component: DivesListComponent },
14   { path: 'contactez-nous', component: ContactsComponent },
15   { path: 'liste-des-plongees/details/:param', component: DivesDetailComponent
16   },
17   { path: 'les-formulaires', component: FormulairesComponent },
18
19   // page404
20   // { path: '**', redirectTo: '', pathMatch: 'full' },
21   { path: '**', component: Page404Component },
22 ]
```

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

```
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 > 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;nbsp;  
74     <mat-checkbox name="name_choix1" FormControlName="choix1"> Choix #1</mat-checkbox>  
75     <mat-checkbox name="name_choix2" FormControlName="choix2"> Choix #2</mat-checkbox>  
76     <p></p>  
77  
78     <mat-icon>settings</mat-icon>&nbsp;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;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-toggle>  
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  
68 // cycle de vies  
69 ngAfterViewInit() {  
  
formulaires.module.ts > ...  
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-label
56 <input ma
57 </mat-form-fi
58
59 <p></p>
60 </div>
61 </div>
62
63 <p></p>
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

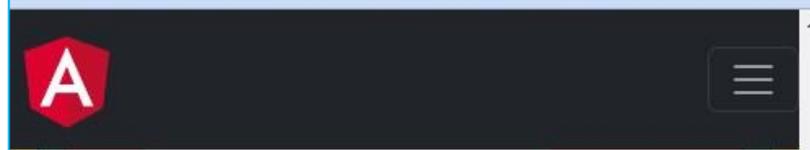
[webpack-dev-server] Server started: Hot Module Replacement disabled, Live Reloading enabled, Progress disabled, Overlay enabled. [index.js:485](#)

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: [form-group.component.ts:62](#)
 - cp: "3"
 - rue: "2"
 - ville: "2"
 - ▶ [[Prototype]]: Object
- ▼ Sous-Group (form) adresse : {rue: '2', ville: '2', cp: '3'} [form-group.component.ts:62](#)
 - 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
18   private _post: PostService
19 ) {
20 }
21 this.monForm = new FormGroup({
22 // déclaration de tous les controls de ce formulaire (formGroup)
```

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

```
bdd.json x
TP06-formulaires > src > assets > json > bdd.json > ...
76 "photo": "https://dev.webjss.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
10 export class PostService {
11
12   constructor(private _http: HttpClient) { }
13
14 // méthodes
15 public postForm = (form: FormGroup) => {
16   console.log('----- OBJET JS : ', form.value);
17
18   // --- Post sur un serveur json
19   // 1-url
20   const url: string = 'http://localhost:3001/messages';
21   // 2-body
22   const body = JSON.stringify(form.value);
23   console.warn('----- OBJET STRINGIFY : ', body);
24
25   // 3-headers
26   const myHeaders = new HttpHeaders({
27     'Content-Type': 'application/json',
28     'Access-Control-Allow-Origin': '*' // CORS
29   });
30   const options = { headers: myHeaders };
31   //-----
32
33   // envoi avec POST
34   this._http.post(url, body, options).subscribe(
35     (res: any) => console.log(res);
36   );
37
38 }
```

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>
```

Les Tests Unitaires



Copyright Michel ACCIOLESI - 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:

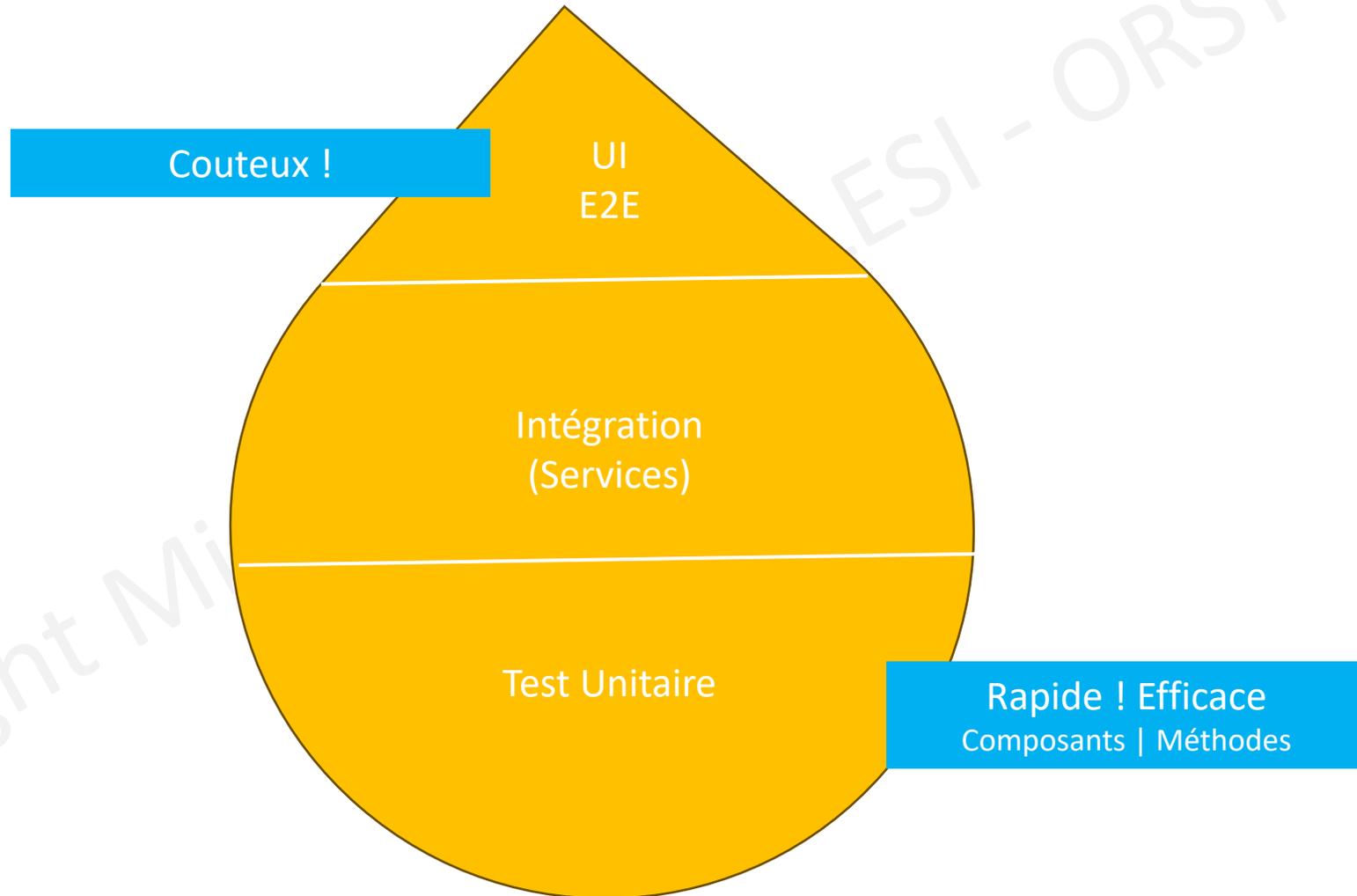
```
1 # Savoir documenter son projet
2
3 ## 1-Les tests Unitaires
4
5 ### Composants :
6 ### Pipes :
7 ### Directives :
8
9 ## 2-Les tests D'intégrations
10
11 ### Services :
12
13 ## 3-Les tests End To End
14
15 ### Mise en Prod :
```

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.

Le Test et la Documentation



1^{er} exemple : Apprendre de l'existant

The image shows a code editor with two panels. The left panel displays the source code for `app.component.ts` and `app.component.html`. The right panel displays the test suite `app.component.spec.ts`.

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

```
app.component.spec.ts
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
```

2nd 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 })
```

2nd 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) => {
```

2nd 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
=====
Statements : 88.88% ( 16/18 )
Branches   : 100% ( 0/0 )
Functions  : 71.42% ( 5/7 )
Lines      : 87.5% ( 14/16 )
=====
```

2nd 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 }
```

2nd 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 }
```

ANNEXES



Copyright Michelangelo MOLESI - ORSYS

Les versions marquantes !



Copyright Michel BACCIOLESI - ORSYS

Les versions marquantes Angular

- Version 9 (6 février 2020)
 - Nouveau compiler et nouveau moteur de rendu Ivy
 - Réduction notable de la taille des bundles (build)
 - Optimisation de la compilation (temps, debug) AOT
activé par défaut avec ng serve => remontée des erreurs en mode dev ...
 - composant YouTube
<https://github.com/angular/components/tree/main/src/youtube-player>
 - composant Google Maps
<https://github.com/angular/components/blob/main/src/google-maps/README.md>

EXPLORATEUR

ÉDITEURS OUVERTS

- package.json TP-NG9

ANGULAR-RELEASES

- TP-NG9
 - e2e
 - src
 - protractor.conf.js
 - tsconfig.json
 - src
 - .editorconfig
 - .gitignore
 - angular.json
 - browserslist
 - karma.conf.js
 - package.json
 - README.md
 - tsconfig.app.json
 - tsconfig.json
 - tsconfig.spec.json
 - tslint.json

package.json X

TP-NG9 > package.json > ...

```
1 {
2   "name": "tp-ng9",
3   "version": "0.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve",
7     "build": "ng build",
8     "test": "ng test",
9     "lint": "ng lint",
10    "e2e": "ng e2e"
11  },
12  "private": true,
13  "dependencies": {
14    "@angular/animations": "~9.1.13",
15    "@angular/common": "~9.1.13",
16    "@angular/compiler": "~9.1.13",
17    "@angular/core": "~9.1.13",
18    "@angular/forms": "~9.1.13",
19    "@angular/platform-browser": "~9.1.13",
20    "@angular/platform-browser-dynamic": "~9.1.13",
21    "@angular/router": "~9.1.13",
22    "rxjs": "~6.5.4",
23    "tslib": "^1.10.0",
24    "zone.js": "~0.10.2"
25  },
```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL PORTS

PS C:\Angular-releases> npx @angular/cli@9 new TP-NG9

ORSYS

Copy

EXPLORATEUR

ÉDITEURS OUVERTS

- tsconfig.json TP-NG9

ANGULAR-RELEASES

- TP-NG9
 - e2e
 - src
 - protractor.conf.js
 - tsconfig.json
 - src
 - .editorconfig
 - .gitignore
 - angular.json
 - browserslist
 - karma.conf.js
 - package.json
 - README.md
 - tsconfig.app.json
 - tsconfig.json
 - tsconfig.spec.json
 - tslint.json
 - ng-9.png

TP-NG9 > tsconfig.json > ...

```
1 {
2   "compileOnSave": false,
3   "compilerOptions": {
4     "baseUrl": "./",
5     "outDir": "./dist/out-tsc",
6     "sourceMap": true,
7     "declaration": false,
8     "downlevelIteration": true,
9     "experimentalDecorators": true,
10    "module": "esnext",
11    "moduleResolution": "node",
12    "importHelpers": true,
13    "target": "es2015",
14    "lib": [
15      "es2018",
16      "dom"
17    ]
18  },
19  "angularCompilerOptions": {
20    "fullTemplateTypeCheck": true,
21    "strictInjectionParameters": true
22  }
23 }
24
```

Les versions marquantes Angular

- Version 10 (24 juin 2020)

- Mode `strict` (typage obligatoire)
`ng new TP01 --strict`

Rappels :

- basic : pas de vérification de types
- full : les props utilisées dans la Vue doivent être correctement définies dans le TS
- nouveau datePicker dans Angular Matériel

ÉDITEURS OUVERTS

- package.json TP-NG10
- ANGULAR-RELEASES
 - TP-NG9
 - TP-NG10
 - e2e
 - src
 - .browserslistrc
 - .editorconfig
 - .gitignore
 - angular.json
 - karma.conf.js
 - package.json
 - README.md
 - tsconfig.app.json
 - tsconfig.json
 - tsconfig.spec.json
 - tslint.json
 - ng-9-config.png
 - ng-9.png

TP-NG10 > package.json > ...

```
1 {
2   "name": "tp-ng10",
3   "version": "0.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve",
7     "build": "ng build",
8     "test": "ng test",
9     "lint": "ng lint",
10    "e2e": "ng e2e"
11  },
12  "private": true,
13  "dependencies": {
14    "@angular/animations": "~10.2.4",
15    "@angular/common": "~10.2.4",
16    "@angular/compiler": "~10.2.4",
17    "@angular/core": "~10.2.4",
18    "@angular/forms": "~10.2.4",
19    "@angular/platform-browser": "~10.2.4",
20    "@angular/platform-browser-dynamic": "~10.2.4",
21    "@angular/router": "~10.2.4",
22    "rxjs": "~6.6.0",
23    "tslib": "^2.0.0",
24    "zone.js": "~0.10.2"
25  }
}
```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL PORTS

```
PS C:\Angular-releases> npx @angular/cli@10 new TP-NG10
```

Les versions marquantes Angular

- Version 11 (17 novembre 2020)

- Mode `strict` est proposé lors de la création de l'appli

```
PROBLÈMES  SORTIE  CONSOLE DE DÉBOGAGE  TERMINAL  PORTS

PS C:\Angular-releases> npx @angular/cli@11 new TP-NG11
Need to install the following packages:
  @angular/cli@11.1.2
Ok to proceed? (y) y
npm WARN deprecated @npmcli/move-file@1.1.2: This functionality has been moved to @npmcli/fs
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated source-map-codec@1.4.8: Please use @jridgewell/source-map-codec instead
npm WARN deprecated @npmcli/ci-detect@1.4.0: this package has been deprecated, use `ci-info` instead
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() i
/math-random for details.
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() i
/math-random for details.
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/
npm WARN deprecated @schematics/update@0.1101.2: This was an internal-only Angular package up through Angular v
pendency.
? Do you want to enforce stricter type checking and stricter bundle budgets in the workspace?
  This setting helps improve maintainability and catch bugs ahead of time.
  For more information, see https://angular.io/strict (y/N) 
```

Les versions marquantes Angular

- Version 12 (19 mai 2021)

- Abandon progressif de la compatibilité et support IE
validé avec NG13
- Fin du support de **e2e, protractor**..
- TSlint déprécié depuis 2019 est remplacé par ESLint
- script watch qui remplace et améliore build
- build est désormais le mode prod

Copyright Michael ACCIOLESI - ORSYS

EXPLORATEUR

ÉDITEURS OUVERTS

- × package.json TP-NG12

ANGULAR-RELEASES

- TP-NG9
- TP-NG10
- TP-NG11
- TP-NG12
 - node_modules
 - src
 - .browserslistrc
 - .editorconfig
 - .gitignore
 - angular.json
 - karma.conf.js
 - package-lock.json
 - package.json
 - README.md
 - tsconfig.app.json
 - tsconfig.json
 - tsconfig.spec.json
 - ng-9-config.png
 - ng-9.png
 - ng-10.png
 - ng-11.png

package.json

```
TP-NG12 > package.json > ...
1 {
2   "name": "tp-ng12",
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  },
11  "private": true,
12  "dependencies": {
13    "@angular/animations": "~12.2.0",
14    "@angular/common": "~12.2.0",
15    "@angular/compiler": "~12.2.0",
16    "@angular/core": "~12.2.0",
17    "@angular/forms": "~12.2.0",
18    "@angular/platform-browser": "~12.2.0",
19    "@angular/platform-browser-dynamic": "~12.2.0",
20    "@angular/router": "~12.2.0",
21    "rxjs": "~6.6.0",
22    "tslib": "^2.3.0",
23    "zone.js": "~0.11.4"
24  }
}
```

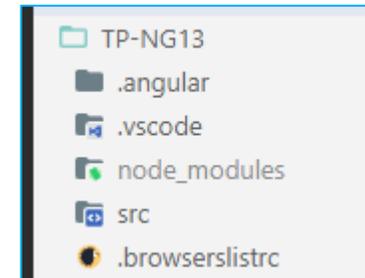
PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL PORTS

```
PS C:\Angular-releases> npx @angular/cli@12 new TP-NG12
```

Les versions marquantes Angular

- Version 13 (3 novembre 2021)

- Arrivée du cache .angular => accélération des temps de compilation
- démarrage en mode dév plus rapide
- Travail continu d'optimisation de IVY et ViewEngine

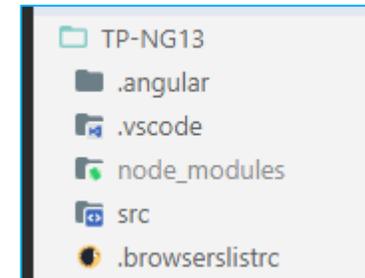


Copyright Michel BOCCIOLESI - ORSYS

Les versions marquantes Angular

- Version 14 (11 juin 2022)

- FormControl enfin typé
- composants standalone (*validé avec NG15*)
- optimisation des messages d'erreurs
- erreur banana in a box [()] ou [[]]



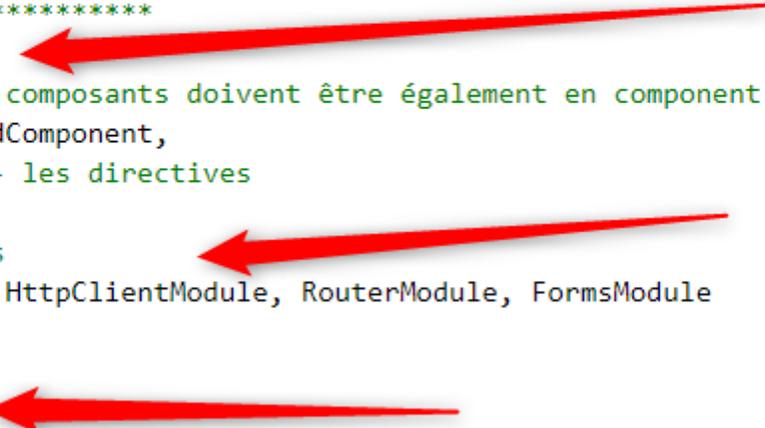
Copyright Michel BOCCIOLESI - ORSYS

Les versions marquantes Angular

- Version 15 (16 novembre 2022)

- composants standalone
 - Prop : standalone:true
- Omission des ngModules ?
- API autonomes ?

```
@Component({
  selector: 'app-root',
  // *****
  standalone: true,
  templateUrl: './dives-list.component.html',
  styleUrls: ['./dives-list.component.scss'],
  // *****
  imports: [
    // les autres composants doivent être également en component standalone !!
    DivesListChildComponent,
    // les pipes - les directives
    FilterPipe,
    // les modules
    CommonModule, HttpClientModule, RouterModule, FormsModule
  ],
  providers: [
    DivesService
  ]
})
```



Copyright Mich

Le mode Standalone Component

The screenshot displays an IDE with the following code:

```
main.ts
1 import { bootstrapApplication } from '@angular/platform-browser';
2 import { appConfig } from './app/app.config';
3 import { AppComponent } from './app/app.component';
4
5 bootstrapApplication(AppComponent, appConfig)
6   .catch((err) => console.error(err));
7
```

```
app.config.ts
1 import { ApplicationConfig, provideZoneChangeDetection } from '@angular/core';
2 import { provideRouter } from '@angular/router';
3
4 import { routes } from './app.routes';
5 import { provideAnimationsAsync } from '@angular/platform-browser/animations/async';
6
7 export const appConfig: ApplicationConfig = {
8   providers: [
9     provideZoneChangeDetection({ eventCoalescing: true }),
10    provideRouter(routes), provideAnimationsAsync()
11  ]};
12
```

```
app.component.ts
1 import { Component } from '@angular/core';
2 import { EntryRootComponent } from './webApp/entry-root.component';
3 import { RouterOutlet } from '@angular/router';
4
5 @Component({
6   selector: 'app-root',
7   standalone: true,
8   imports: [
9     EntryRootComponent,
10  ],
11  templateUrl: './app.component.html',
12  styleUrls: ['./app.component.scss']
13 })
14 export class AppComponent {
15   public title:string='TU';
16 }
17
```

```
app.routes.ts
8 export const routes: Routes = [
9   { path: '/', component: HomeComponent },
10  {
11    path: 'nous-contactez',
12    loadComponent:
13      () => import('./webApp/contacts/entry-contacts/entry-contacts.component')
14        .then(
15          (c) => c.EntryContactsComponent
16        )
17  },
18  // {
19  //   path: 'compte-client',
20  //   component: EntryCompteClientComponent,
21  //   children: [
22  //     { path: 'mes-commandes', component: CommandesComponent },
23  //     { path: 'mes-livraisons', component: LivraisonsComponent },
24  //   ]
25  // },
26  {
27    path: 'compte-client',
28    loadComponent:
29      () => import('./webApp/compte-client/components/entry-compte-client/entry-compte-client.component')
```

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 !



Copyright Michel POCQUOLESI - ORSYS



Angular 17

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

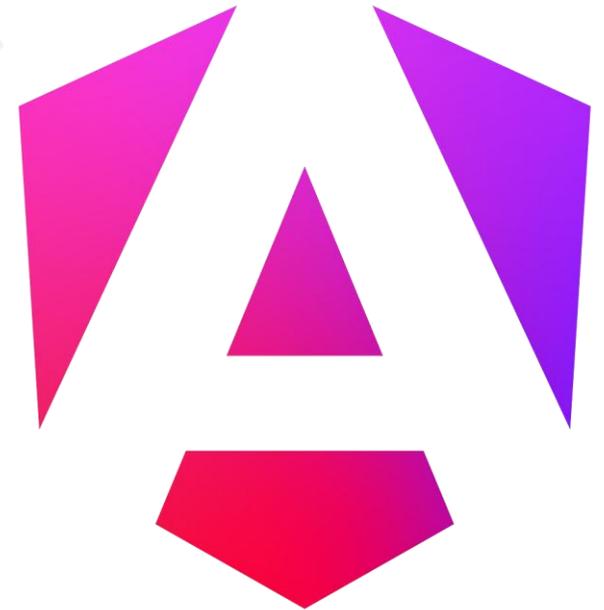


Copyright Michel BOCCIOLESI - ORSTIS

Angular 18

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

- Fin de Zone.js 😊
- Utilisation des signaux entre composants
- hydratation dans Angular DevTools



Copyright Michel BOCCIOLESI - ORSYS

Angular 19

19 novembre 2023

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



Copyright Miodio - ORSYS

Angular 19



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

Copyright Michel BOCCIOLESI - ORSYS

Angular 19



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

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

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

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

Copyright

Angular 19



- Material Theme mat.theme

```
1  @use '@angular/material' as mat;  
2  
3  html {  
4    @include mat.theme(  
5      color: mat.$azure-palette,  
6      typography: Roboto,  
7      density: 0,  
8    ));  
9  }  
10  
11
```

Copyright Mi

Rappels – Révisions Ecmascript



Copyright Michelangelo MOLESI - ORSYS

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
>
```

ORSY

Copyri

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+



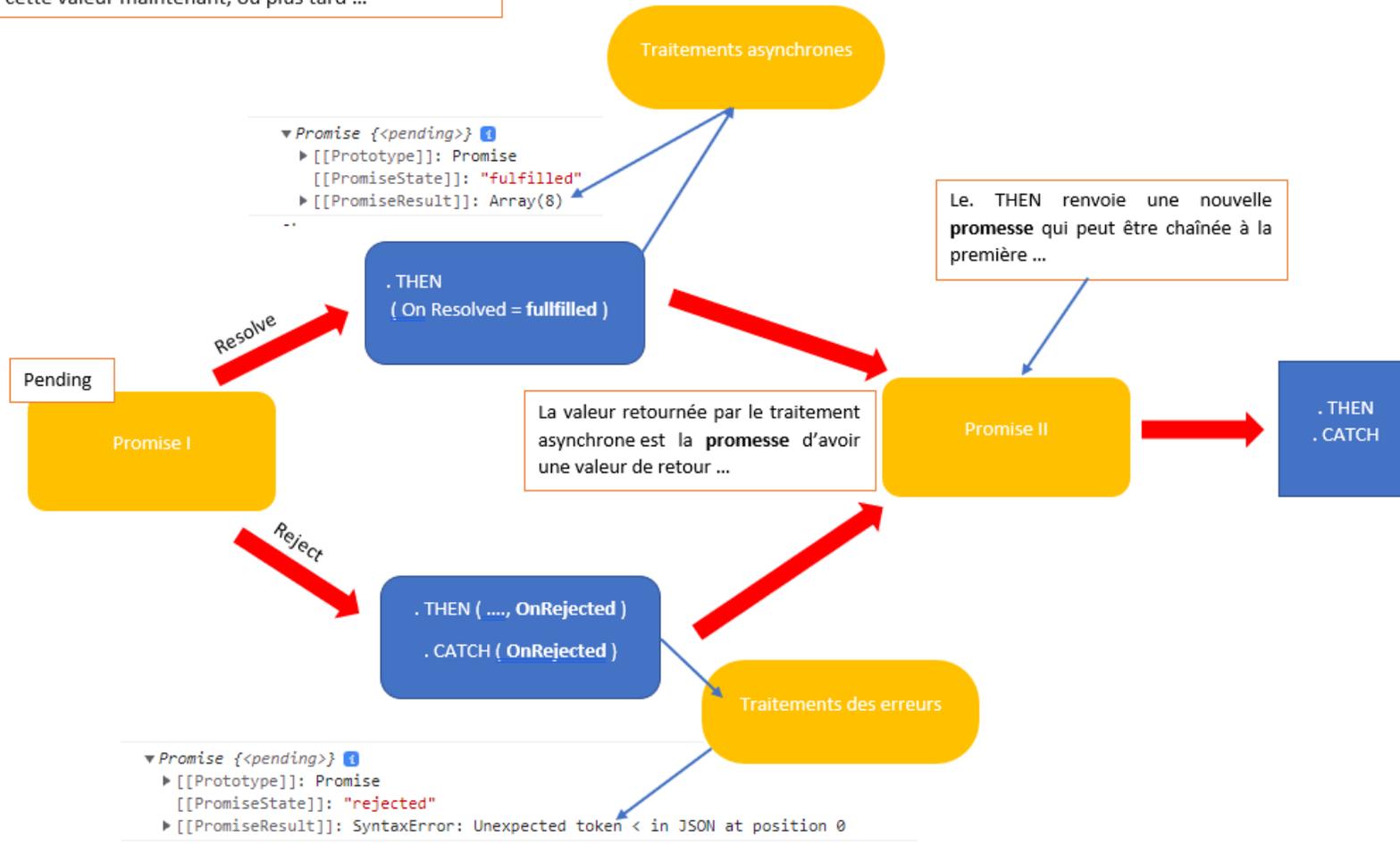
```
58 // -----  
59 let personne = {  
60   prenom: "Luke",  
61   nom: "Skywalker",  
62   // tableaux  
63   langages: ["JS", "React", "NG", "Vue"],  
64   // méthodes  
65   nomComplet() {  
66     return `${this.prenom} ${this.nom}`;  
67   }  
68 }  
69 console.log(personne.nomComplet());  
70  
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 ...



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);  
36    }  
37  )  
38 }
```

ORSYS

Copy



Formation ANGULAR

A bientôt.