



Formation ANGULAR Développement Avancé

Chapitre 1

Animé par Michel BOCCIOLESI

Table des matières

- « Nouveautés - Rappels » EcmaScript 2015+
La révolution Javascript (*rappels*)
- Documenter son projet – fichiers MD
- PWA – Progressive Web App
Comment une appli Web devient « progressivement » une appli Mobile ?
- I18N : L'Internationalisation
Comment traduire notre projet Angular ?
- Le routage avancé et le Lazy Loading
Concepts avancés de routage
Optimisation de la compilation
- La programmation RXJS et les observables
- Les formulaires Angular
Reactive Forms vs Template Driven Form
- Tests Unitaires
Karma & Jasmine
- Eco-système back Firebase
Base de données back-end en temps réel
- NGRX – la notion de Store dans Angular
aNGular Redux rXjs
- Le Framework Angular - Historique
Les versions marquantes 9 – 17
- Le « Detection Change Mode » Angular
Introduction au « signal NG16 – NG17 »

[Récupérer les sources du projet de formation ici](#)

Installation et configuration

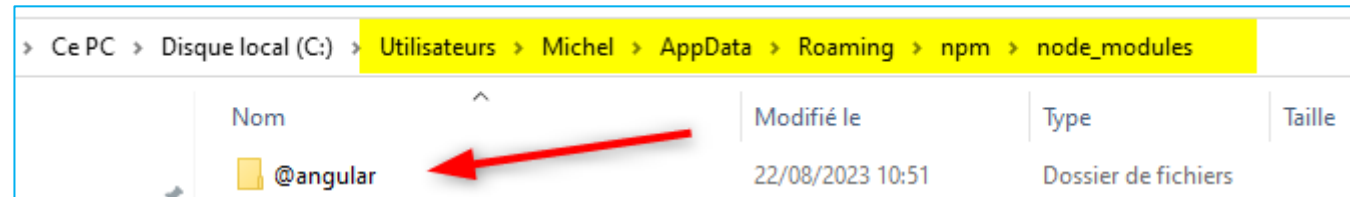
<https://nodejs.org>

<https://angular.io/>

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

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

/home/michel (linux)



Ce PC > Disque local (C:) > Utilisateurs > Michel > AppData > Roaming > npm > node_modules				
	Nom	Modifié le	Type	Taille
	@angular	22/08/2023 10:51	Dossier de fichiers	

- `npm install @angular/cli@12`
- `npm install @angular@15`

Documenter son projet



The image shows a side-by-side comparison of a Markdown file and its rendered PDF version. On the left, the VS Code editor displays the file `COMMANDES.md` with the following content:

```
1 Pour générer le fichier lisible : CTRL + SHIFT + v
2 ou bouton droit
3
4 # Commandes Angular NG
5
6 # New projet
7
8 ng new nomDuProjet
9
10 ## options :
11
12 # Generate objets
13
14 ## composant
15 - ng g(enerate) c(omposant) nomDuComposant
16 - options :
17 --export (exporte le composant depuis le module)
18 --skip-tests : omet la création du fichier de spec.ts (tests
19 unitaires)
20 --skip-import : ne déclare pas le composant dans un module
21 --flat : crée le composant sans dossier
22
23 ## module
24 - ng g(enerate) m(odule) nomDuModule
25 - options :
26 -- routing : crée le fichier de routing forChild pour le module
27 concerné
28
29 ## service
30 - ng g(enerate) s(ervice) nomDuService
31 - options :
32 --routing : crée le fichier de routing forChild pour le module
33 concerné
34 --flat : crée le service sans dossier
```

On the right, the rendered PDF document `COMMANDES.pdf` is shown. It features a search bar at the top with the text "1 sur 1" and a "Zoom automatique" button. The content is formatted with headings and lists:

COMMANDES.md

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

Commandes Angular NG

New projet

ng new nomDuProjet

options :

Generate objets

composant

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

Documenter son projet



COMMANDES.md

Prévisualiser COMMANDES.md

COMMANDES.pdf

Extension : Markdown PDF

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

Commandes Angular NG

New projet

Markdown PDF v1.5.0

yzane | 1747 619 | ★★★★★ (105)

Convert Markdown to PDF

Désactiver | Désinstaller | ⚙️

Cette extension est activée globalement.

<https://nodejs.org/fr>

```
PS C:\Users\Michel> node -v
v18.12.1
PS C:\Users\Michel> ng version
```

Angular CLI

```
Angular CLI: 16.2.4
Node: 18.12.1
Package Manager: npm 9.7.2
OS: win32 x64

Angular: undefined
...

Package                                  Version
-----
@angular-devkit/architect               0.1602.4 (cli-only)
@angular-devkit/core                    16.2.4 (cli-only)
@angular-devkit/schematics              16.2.4 (cli-only)
@schematics/angular                     16.2.4 (cli-only)
```

```
PS C:\Users\Michel> ng new TP01-les-fondamentaux
```

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.

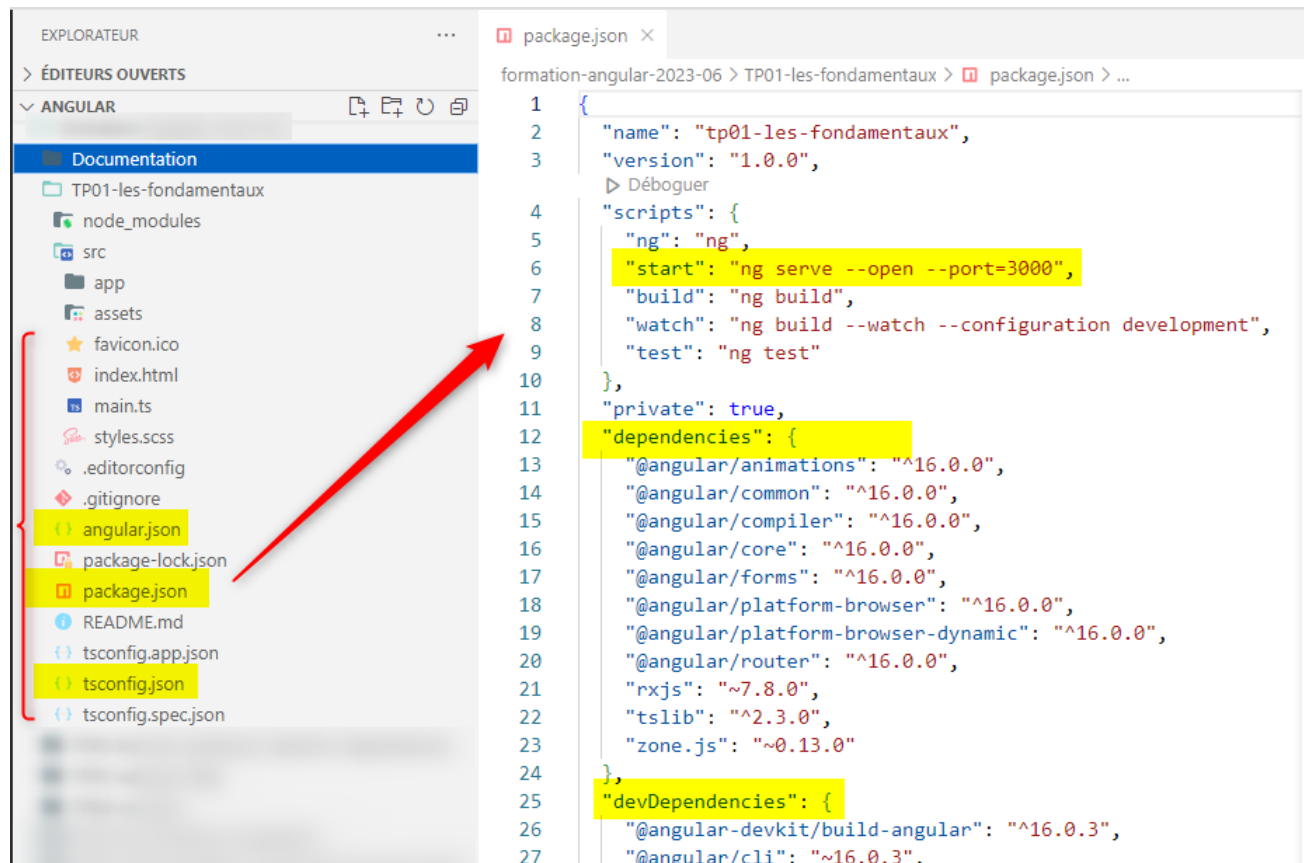
Il y a au moins 32k fichiers dans un simple projet Angular.

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",
```

Rappels – Révisions Ecmascript



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.*

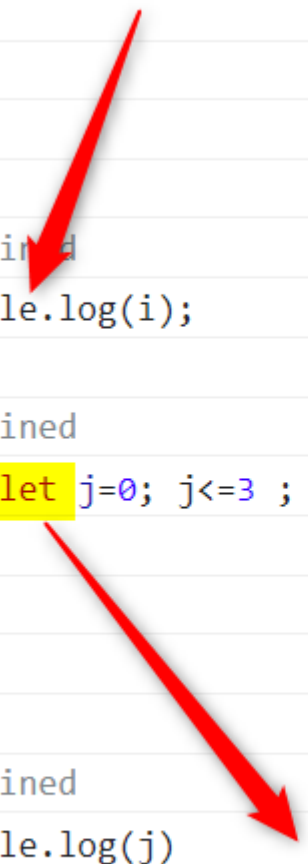
Les améliorations les plus importantes du langage concernent :

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
>
```



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  
>
```

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+

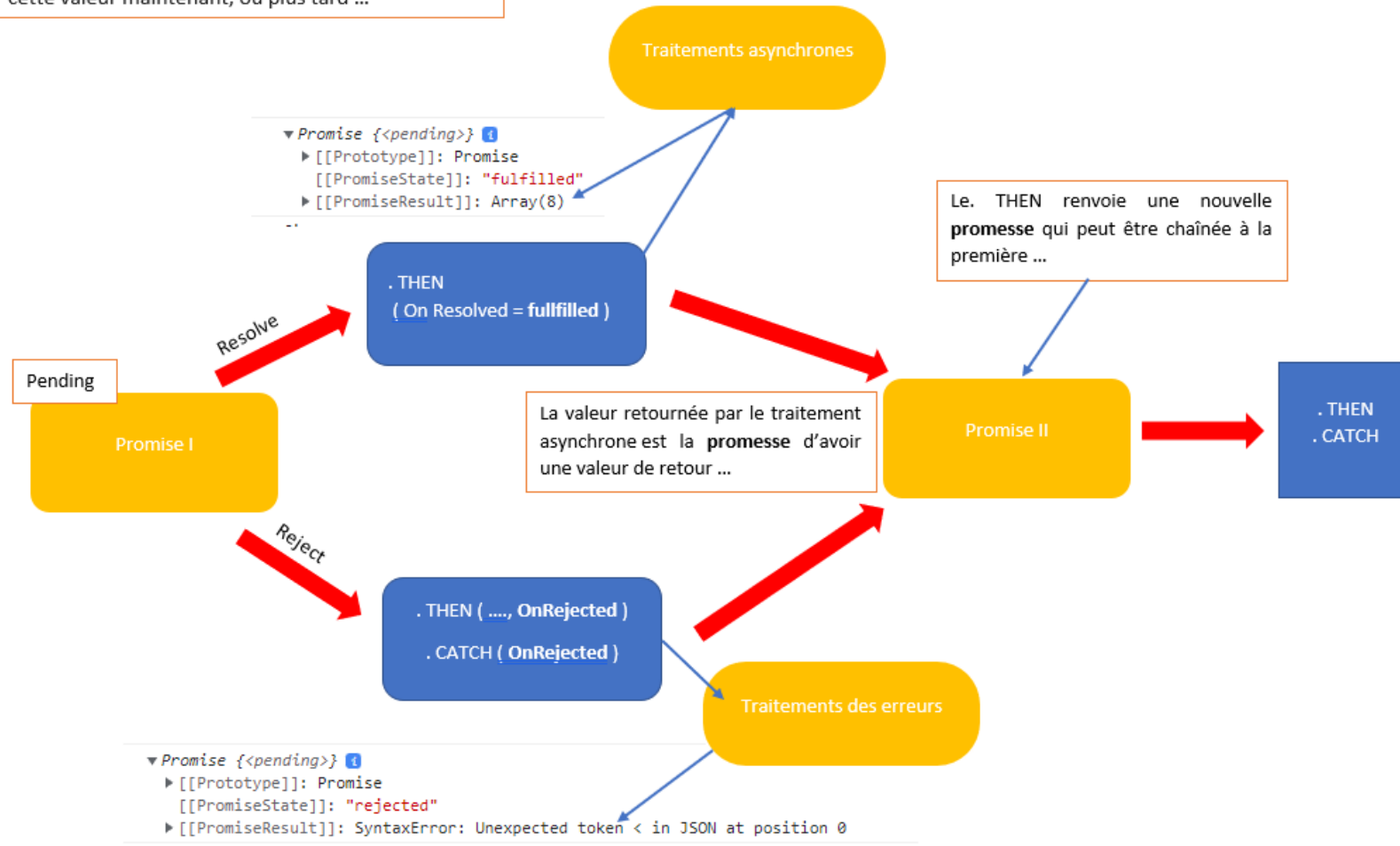


```
58 // -----
59 let personne = {
60   prenom: "Luke",
61   nom: "Skywalker",
62   // tableaux
63   langages: ["JS", "React", "NG", "Vue"],
64   // méthodes
65   nomComplet() {
66     return `${this.prenom} ${this.nom}`;
67   }
68 }
69 console.log(personne.nomComplet());
70
71 // ---- Nouveautés class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



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

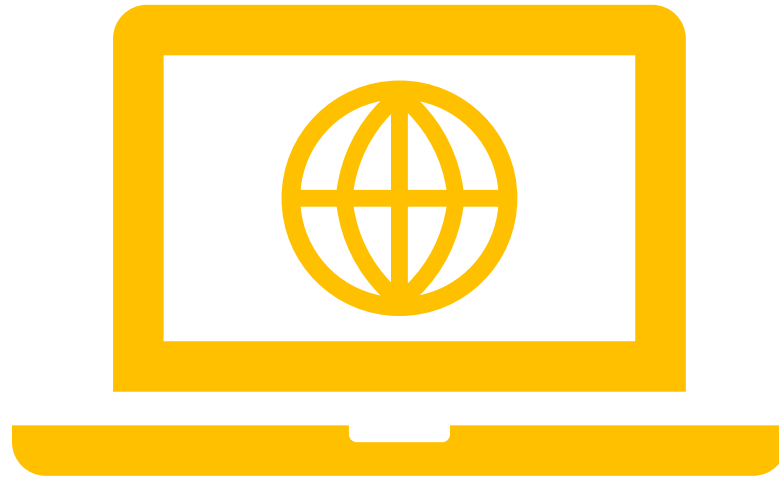


Ecmascript 2015+



```
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);
```

PWA



PWA – les Progressive Web Apps ?

Une Application Web ou une Application Mobile



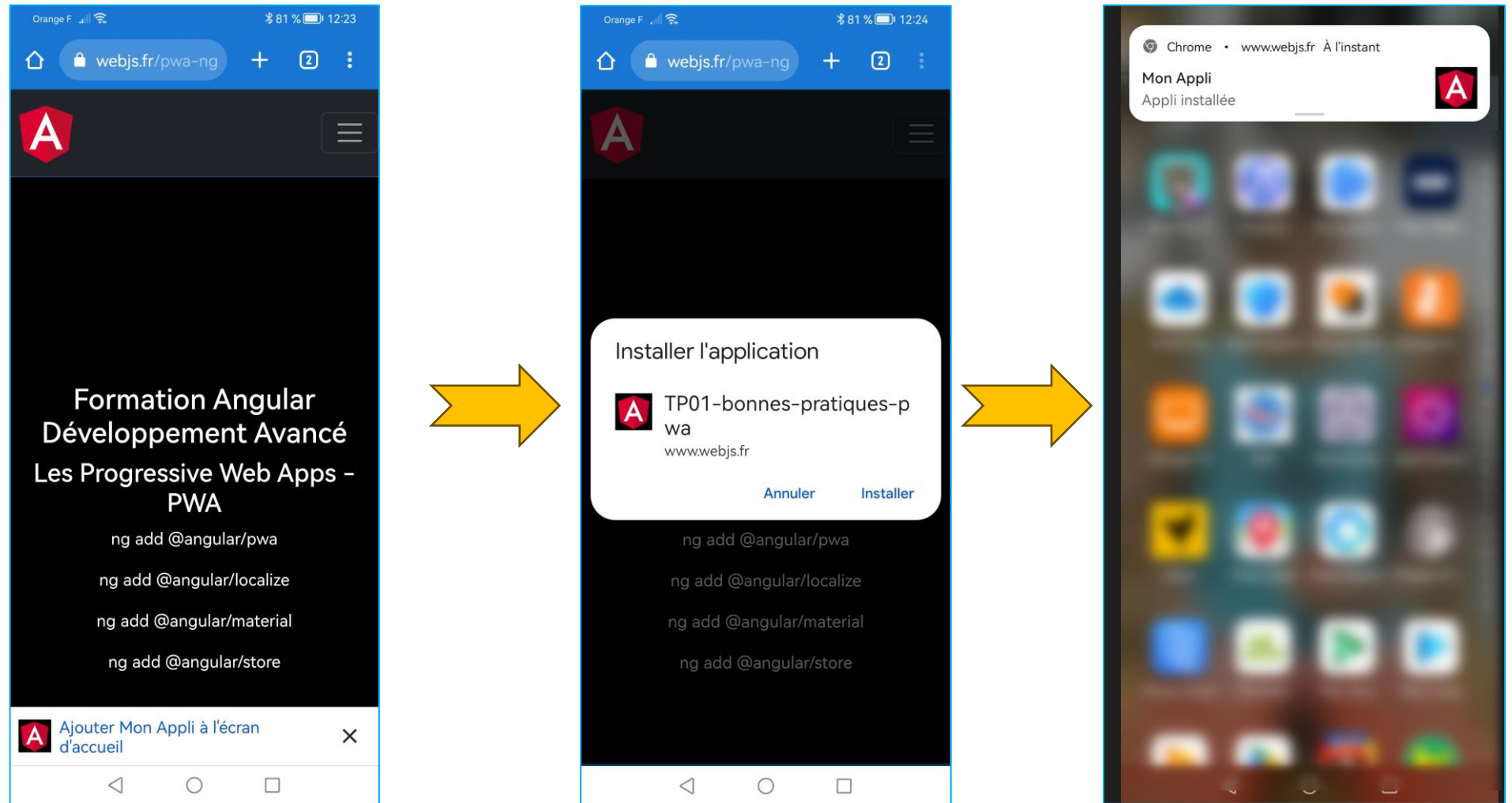
Mise en situation :

consulter cette appli depuis un smartphone :

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

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

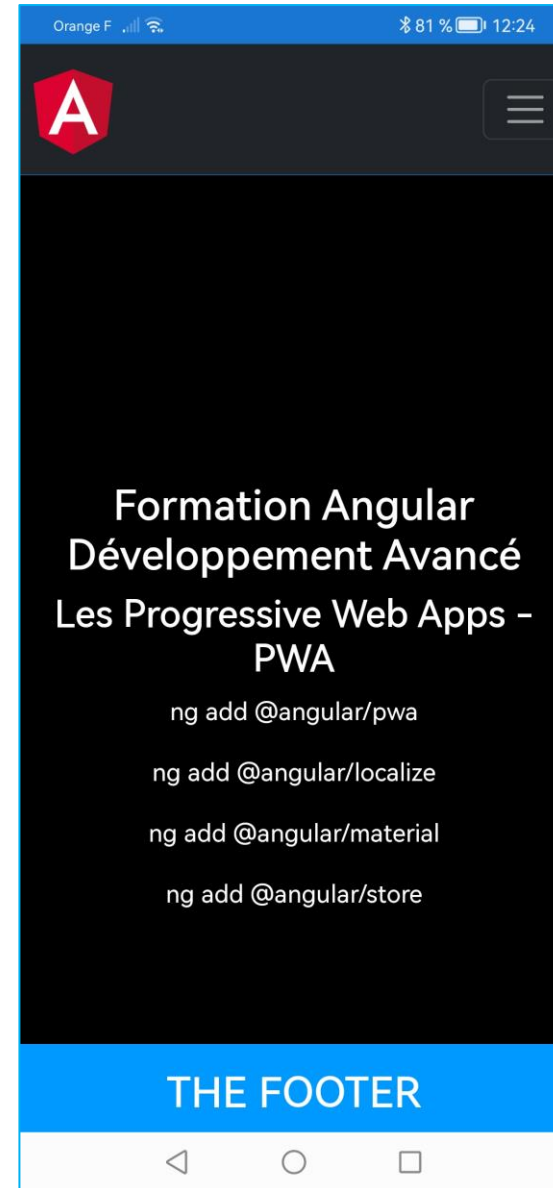
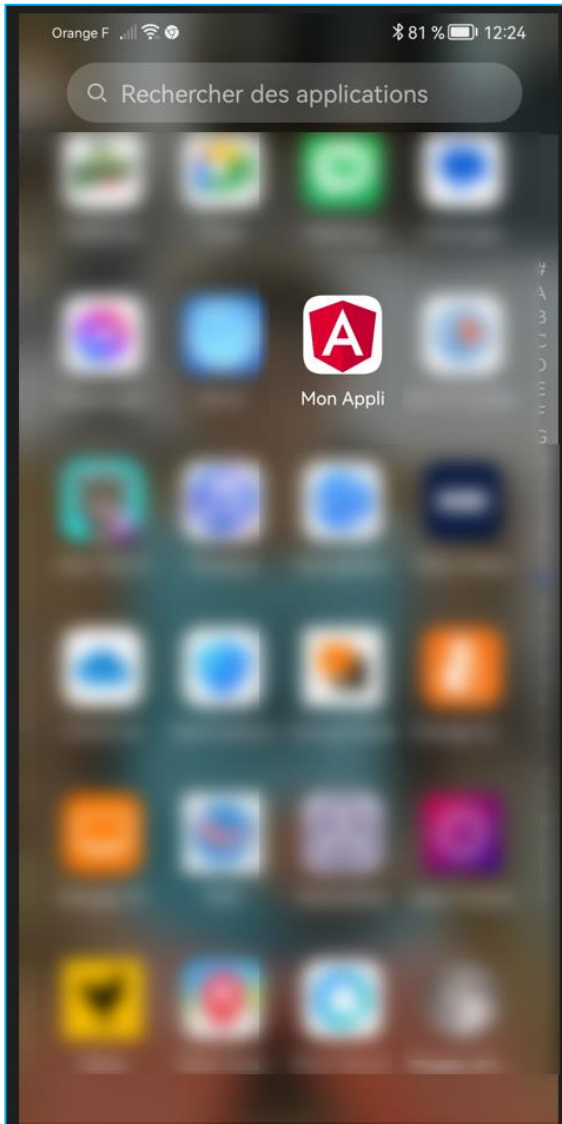
PWA – (le fichier web-manifest)



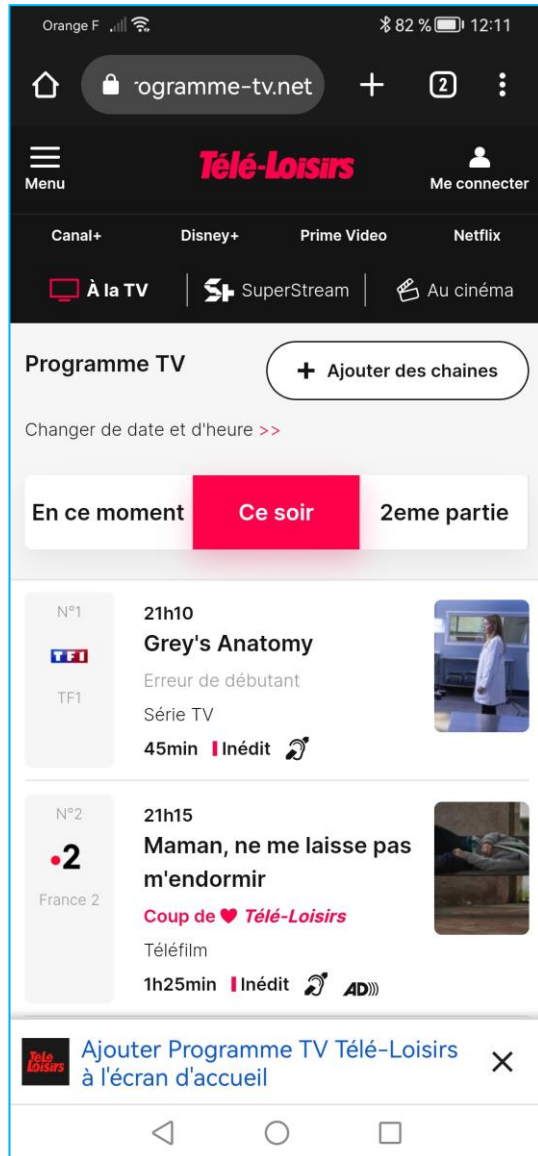
Références :

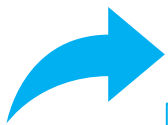
- https://developer.mozilla.org/fr/docs/Web/Progressive_web_apps/Guides/Making_PWAs_installable
- <https://developer.mozilla.org/fr/docs/Web/Manifest>

PWA – les Progressive Web Apps



PWA – les Progressive Web Apps





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

The screenshot displays the Visual Studio Code interface during the installation of the Angular PWA package. The file explorer on the left shows the project structure, including the `src/app` directory and the `assets` directory. The code editor shows the `package.json` file with the following dependencies:

```
"@angular/fire": "^7.6.1",
"@angular/forms": "^16.1.0",
"@angular/material": "^16.2.8",
"@angular/platform-browser": "^16.1.0",
"@angular/platform-browser-dynamic": "^16.1.0",
"@angular/router": "^16.1.0",
"@angular/service-worker": "^16.1.0",
"bootstrap": "^5.3.0",
"bootstrap-icons": "^1.10.5",
```

The `app.module.ts` file is also open, showing the `ServiceWorkerModule` registration:

```
AppRoutingModule,
AccueilModule,
BrowserAnimationsModule,
ServiceWorkerModule.register('ngsw-worker.js', {
  enabled: !isDevMode(),
  // Register the ServiceWorker as soon as the application is stable
  // or after 30 seconds (whichever comes first).
  registrationStrategy: 'registerWhenStable:30000'
}))
```

The terminal at the bottom shows the command `ng add @angular/pwa` being executed, with the following output:

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

found 0 vulnerabilities
PS C:\Users\Michel\Desktop\TD01-pwa> ng add @angular/pwa
i Using package manager: npm
✓ Found compatible package version: @angular/pwa@16.2.6.
✓ Package information loaded.

The package @angular/pwa@16.2.6 will be installed and executed.
Would you like to proceed? Yes
✓ Packages successfully installed.
CREATE ngsw-config.json (631 bytes)
CREATE src/manifest.webmanifest (1336 bytes)
CREATE src/assets/icons/icon-128x128.png (1124 bytes)
CREATE src/assets/icons/icon-144x144.png (1291 bytes)
CREATE src/assets/icons/icon-152x152.png (1306 bytes)
CREATE src/assets/icons/icon-192x192.png (1654 bytes)
CREATE src/assets/icons/icon-384x384.png (3557 bytes)
CREATE src/assets/icons/icon-512x512.png (5008 bytes)
CREATE src/assets/icons/icon-72x72.png (711 bytes)
CREATE src/assets/icons/icon-96x96.png (817 bytes)
UPDATE angular.json (3393 bytes)
UPDATE package.json (1692 bytes)
UPDATE src/app/app.module.ts (944 bytes)
UPDATE src/index.html (647 bytes)
```

npm run build – npm run watch

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

- EXPLORATEUR (Left):** Displays the file explorer for the project 'TD01-pwa'. The file 'ngsw-worker.js' is highlighted in the 'dist\web-app' folder.
- ÉDITEURS OUVERTS (Top):** Shows two open files: 'ngsw-config.json' and 'ngsw-worker.js'. A red arrow points from the 'ngsw-config.json' editor to the 'ngsw-worker.js' editor.
- ngsw-config.json (Middle):** Contains the following JSON configuration:

```
1 {
2   "$schema": "../node_modules/@angular/service-worker/config/schema.json",
3   "index": "/index.html",
4   "assetGroups": [
5     {
6       "name": "app",
7       "installMode": "prefetch",
8       "resources": {
9         "files": [
10          "/favicon.ico",
11          "/index.html",
12          "/manifest.webmanifest",
13          "/*.css",
14          "/*.js"
15        ]
16      }
17    },
18    {
19      "name": "assets",
20      "installMode": "lazy",
21      "updateMode": "prefetch",
22      "resources": {
23        "files": [
```
- ngsw-worker.js (Right):** Contains JavaScript code for the service worker, including a class for 'NamedCacheStorage'.
- TERMINAL (Bottom):** Shows the output of the 'npm run watch' command. It displays an error message: 'Oops, un problème s'est produit. Signalez ce bogue avec les détails ci-dessous.' and a link to report the issue on GitHub. Below the error, it shows the command 'npm run watch' and the output: 'web-app@1.0.0 watch' and 'ng build --watch --configuration development --stats-json'.

PWA – les Progressive Web Apps

The screenshot shows the Chrome DevTools Application tab for the URL `webjs.fr/pwa-ng/`. The interface is divided into three main sections: Application, Storage, and Background services.

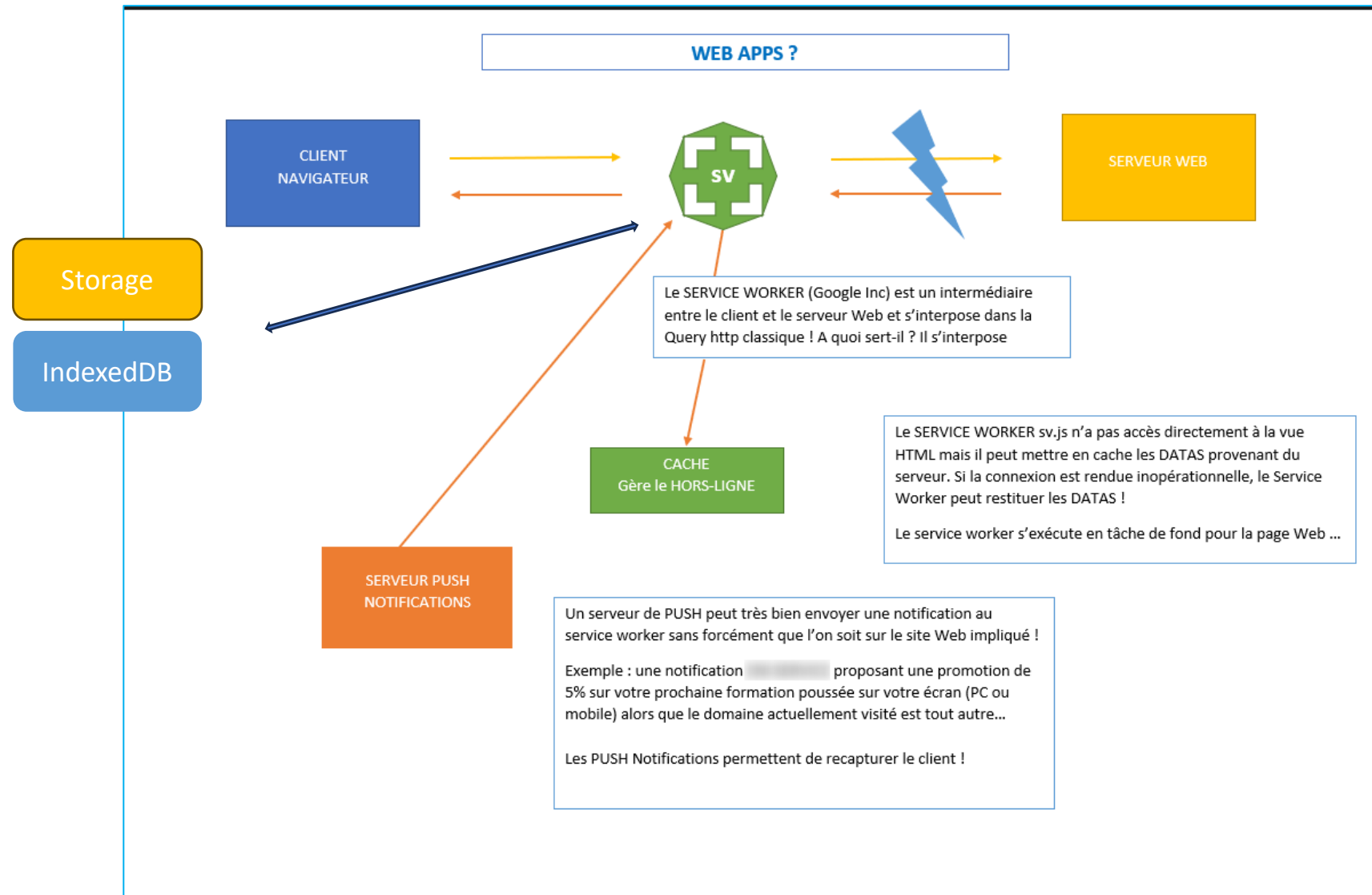
Application: The left sidebar shows the Application tree with `Service workers` highlighted. The right pane displays the service worker configuration for `https://webjs.fr/pwa-ng/`. The source is `ngsw-worker.js`, received on 04/10/2023 at 17:44:53. The status is `#3396 activated and is running` with a `stop` link. The push message is `Test push message from DevTools.` with a `Push` button. The sync type is `pwa` with a `Sync` button. The periodic sync is `periodic` with a `Periodic Sync` button. The update cycle table shows the following data:

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

Storage: The left sidebar shows the Storage tree with `Local storage`, `Session storage`, `IndexedDB`, `Web SQL`, `Cookies`, `Private state tokens`, `Interest groups`, `Shared storage`, and `Cache storage` listed.

Background services: The left sidebar shows the Background services tree with `Back/forward cache`, `Background fetch`, `Background sync`, `Bounce tracking mitigations`, `Notifications`, `Payment handler`, `Periodic background sync`, `Push messaging`, and `Reporting API` listed.

PWA – le Service Worker



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

PWA – le Storage

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

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

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

PWA – Background-Periodic-Sync

```
export const sync = () => {

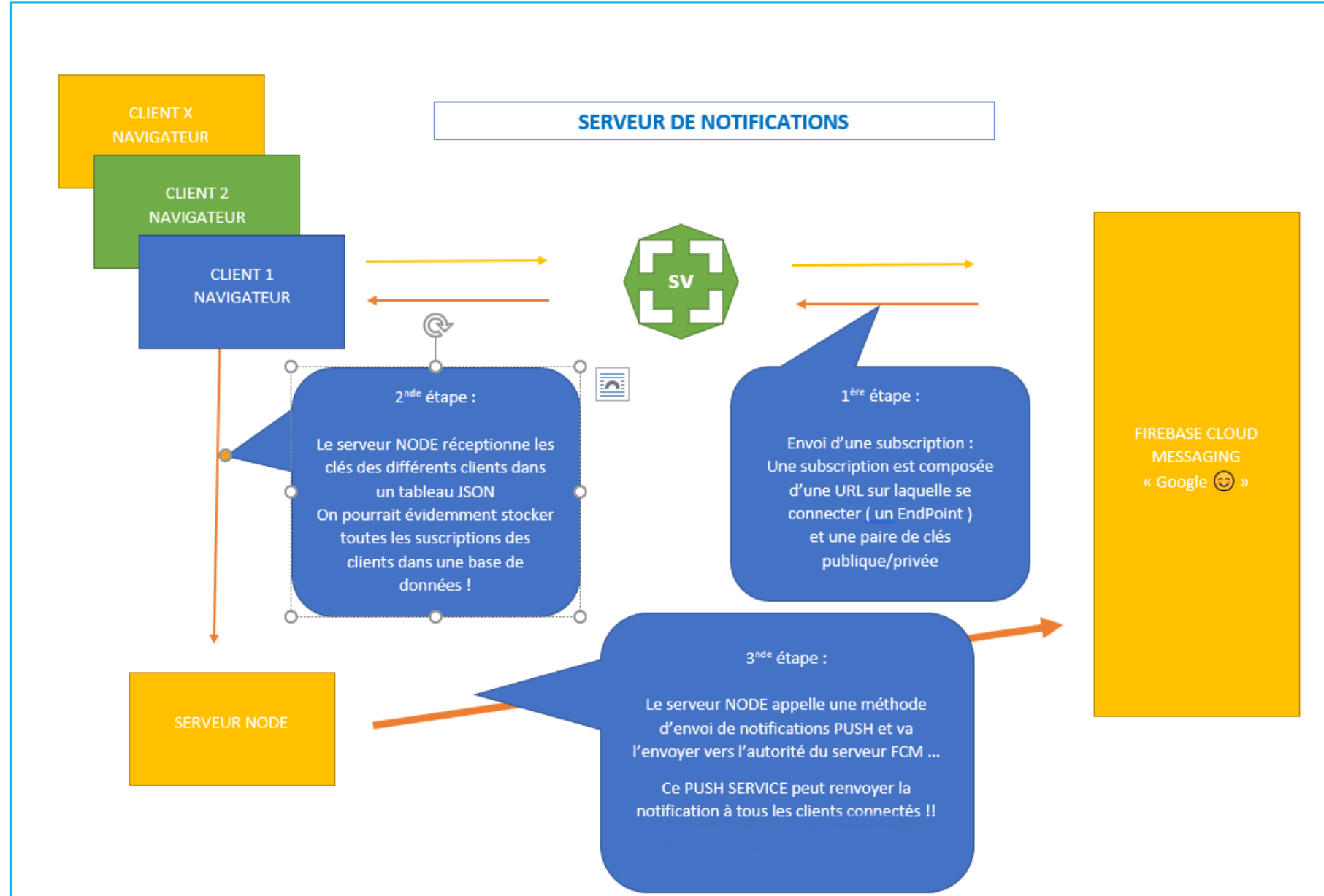
  console.clear();
  console.log('---SYNC---');

  //demander les permissions
  navigator.permissions.query(
    {
      name: 'background-sync'
    }
  ).then(
    (e) => {
      const statusQueryPerm = e.state;
      console.warn(statusQueryPerm);
      if (statusQueryPerm==='granted') {
        console.log(`---> Background -sync est autorisé`);
      }
    }
  );

  // -----
  navigator.serviceWorker.ready
  .then(
    (registration) => {
      registration.sync.register('tag-maj-cache');
    }
  );
};
```

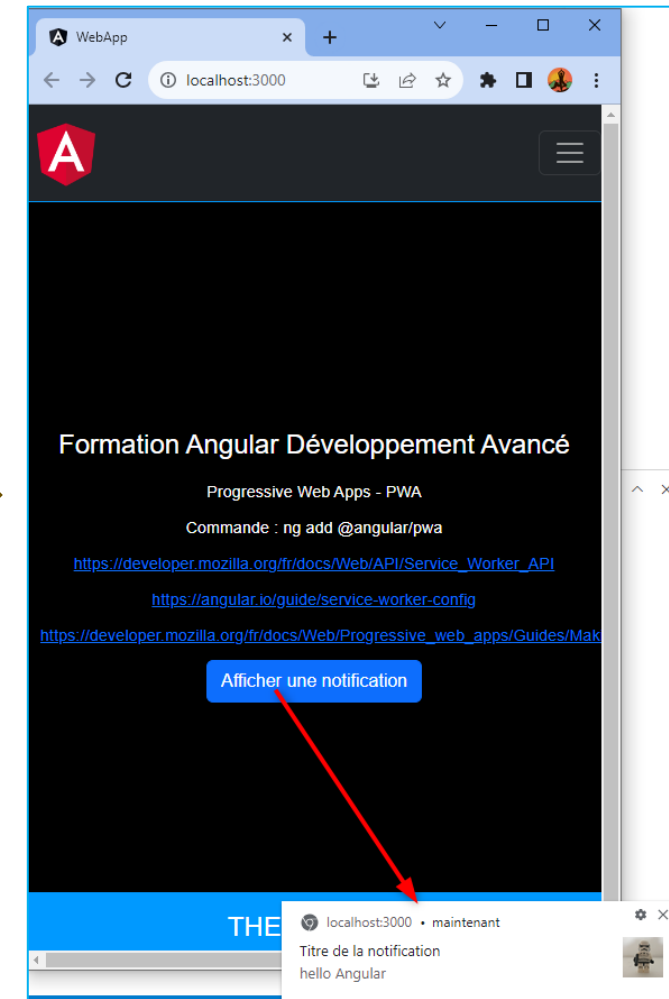
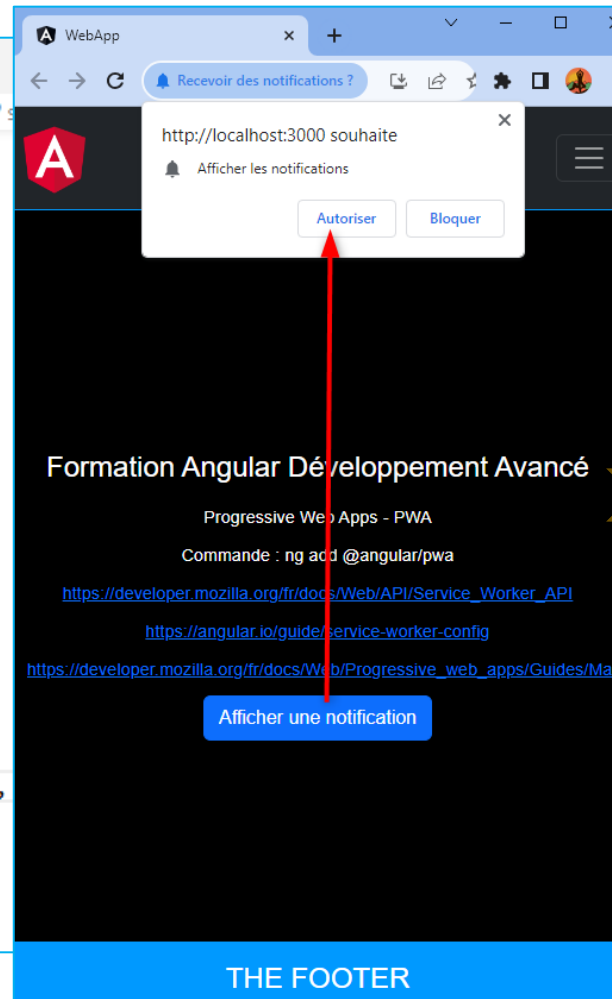
```
1 export const periodicSync = () => {
2
3   console.clear();
4   console.log('---Periodic SYNC---');
5
6   //demander les permissions
7   navigator.permissions.query(
8     {
9       name: 'periodic-background-sync'
10    }
11  ).then(
12    (e) => {
13      const statusQueryPerm = e.state;
14      console.warn(statusQueryPerm);
15      if (statusQueryPerm==='granted') {
16        console.log(`---> Periodic Background sync est autorisé`);
17      }
18    }
19  );
20
21  // -----
22  navigator.serviceWorker.ready
23  .then(
24    async registration => {
25      await registration.periodicSync.register(
26        'tag-periodic-sync',
27        {
28          // minInterval: 24 * 60 * 60 * 1000
29          // 24 heures => daiy news
30          // exprimé en ms
31          minInterval : 5
32        }
33      );
34    }
35  );
36 }
```

PWA – les Progressive Web Apps



PWA – Notification

```
body.component.ts x
TD01-pwa > src > app > webApp > root > accueil > body > body.component.ts > BodyComponent >
18
19
20 // ----- notif simple -----
21 public showNotification = () => {
22
23   if ('Notification' in window) {
24     console.log('Notifications autorisées');
25
26     Notification.requestPermission(
27       (e) => {
28         console.log(e);
29         if (e==='granted') {
30
31           const options={
32             body:'hello Angular',
33             lang: 'fr-FR',
34             icon: 'assets/images/avatars/avatar1.jpg',
35             vibrate: [200, 100, 200]
36           };
37
38           const notif = new Notification('Titre de la notification',
39             options);
40         }
41       }
42     );
43   }
44 }
```



PWA – Notification

index.html

sw.js

exo-8-notif-API-push > sw.js > options > vibrate

```
93     console.log("Erreur : ", error);
94
95   }
96 };
97
98 // -----
99
100 const options = {
101   body: 'Hello Notification PUSH',
102   icon: 'images/icons/icon-192x192.png',
103   vibrate: [200]
104 },
105 actions: [{
106   action: 'close',
107   title: 'Good Bye Notification PUSH'
108 }]
109 }
110
111 self.addEventListener('push',
112   event => {
113     event.waitUntil(self.registration.showNotification
114       'Le titre : Notification PUSH', options
115     )
116   }
117 )
118
119
120
```

TERMINAL

PROBLÈMES

SORTIE

CONSOLE DE DÉBOGAGE

Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw
Change detected E:\FRONTJS\PWA\PWA-11-2020\exo-8-notif-API-push\sw

127.0.0.1:8080/index.html

Applications

RESIZER

Fotolia

123RF

Nice

Collège RAOUL DUFY

Pronote

Elements

Console

Sources

Network

Application

Application

Manifest

Service Workers

Clear storage

Storage

Local Storage

Session Storage

IndexedDB

Web SQL

Cookies

Cache

Cache Storage

Application Cache

Background Services

Background Fetch

Background Sync

Notifications

Payment Handler

Periodic Background Sync

Push Messaging

Frames

top

Service Workers

☐ Offline ☒ Update on reload ☐ Bypass for network

http://127.0.0.1:8080/ [Update](#) [Unregister](#)

Source [sw.js](#)

Received 07/11/2020 à 08:47:02

Status ● #2663 activated and is running [stop](#)

Clients [http://127.0.0.1:8080/index.html](#) [focus](#)

Push [Test push message from Dev](#) [Push](#)

Sync [test-tag-from-devtools](#) [Sync](#)

Periodic Sync [test-tag-from-devto](#) [Periodic Sync](#)

Service workers from other origins

[See all registrations](#)

127.0.0.1:8080 • maintenant

Le titre : Notification PUSH

Hello Notification PUSH

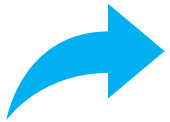
GOOD BYE NOTIFICATION PUSH

L'Internationalisation I18N



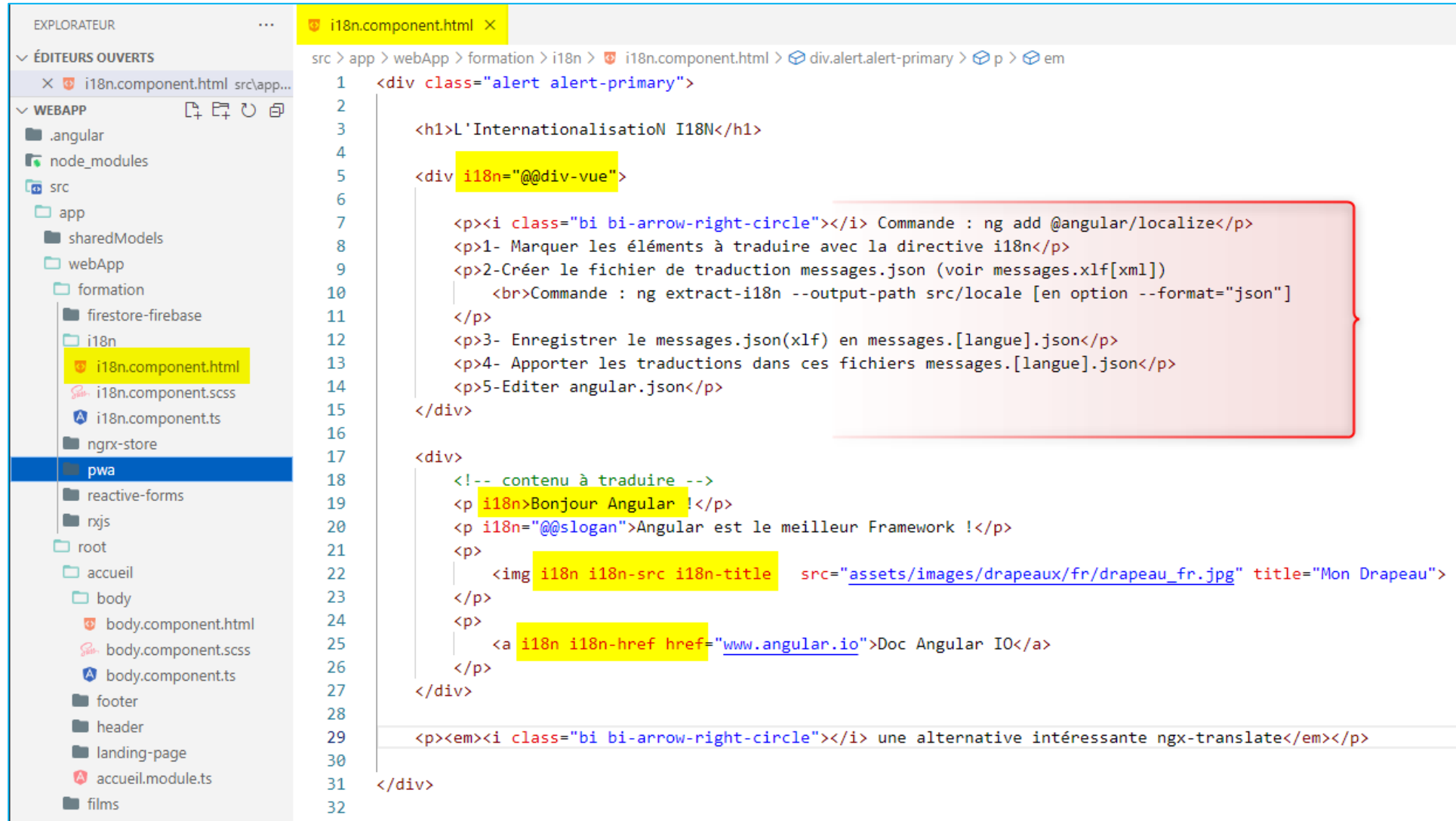
L'Internationalisation I18N

Comment traduire dans un projet Angular ?



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

L'Internationalisation I18N



The screenshot displays a code editor with a file explorer on the left and a code editor on the right. The file explorer shows the project structure, with the `i18n.component.html` file selected under the `formation > i18n` directory. The code editor shows the content of `i18n.component.html`, which includes a title, a list of instructions for internationalization, and a list of items to be translated.

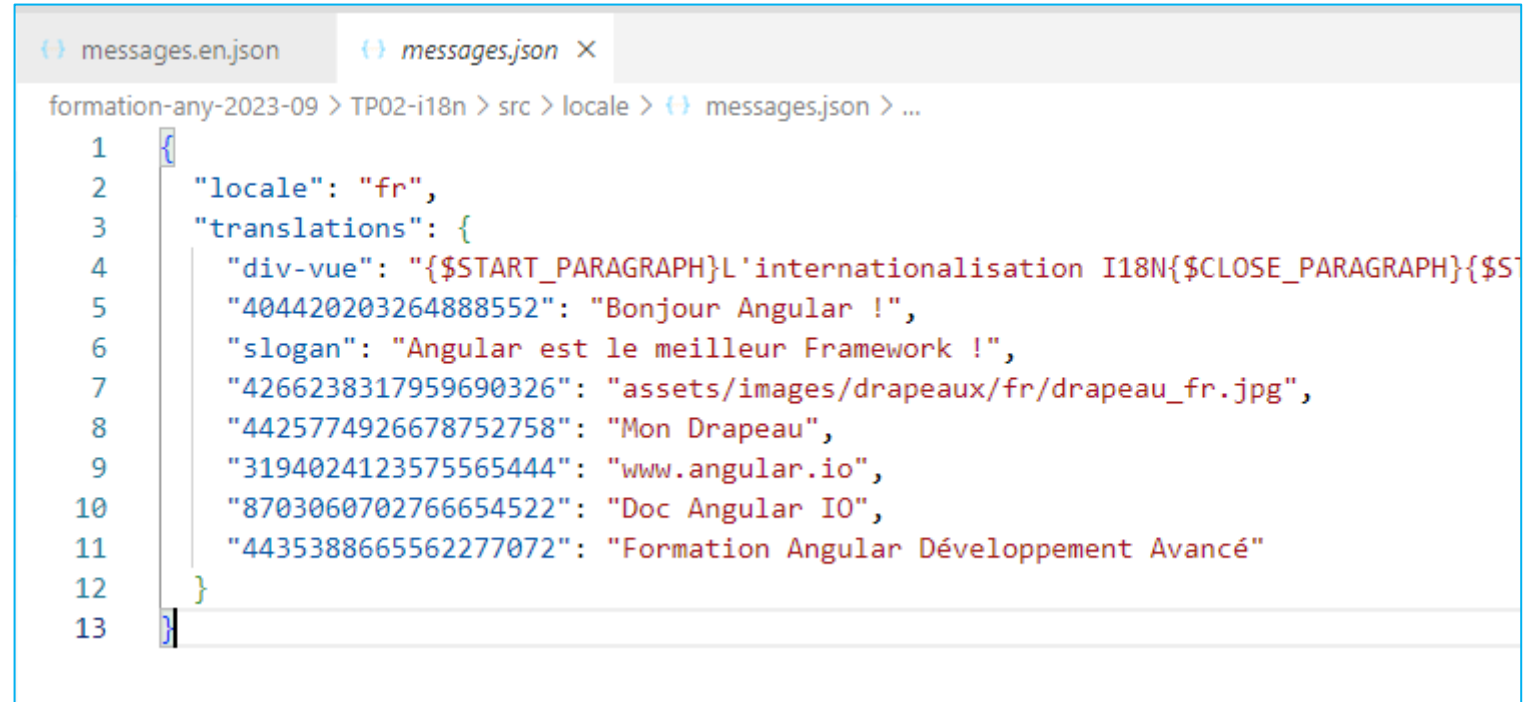
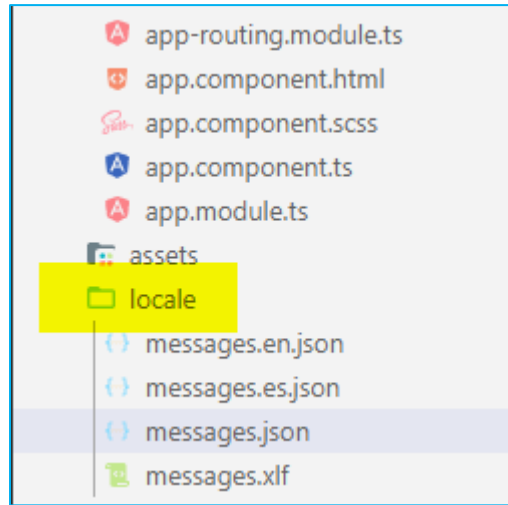
```
1 <div class="alert alert-primary">
2
3 <h1>L'Internationalisation I18N</h1>
4
5 <div i18n="@@div-vue">
6
7 <p><i class="bi bi-arrow-right-circle"></i> Commande : ng add @angular/localize</p>
8 <p>1- Marquer les éléments à traduire avec la directive i18n</p>
9 <p>2-Créer le fichier de traduction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

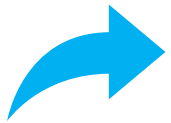
A red box highlights the instructions for internationalization, and a yellow box highlights the `i18n` directive and the `src` attribute of the `img` tag.

L'Internationalisation I18N



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





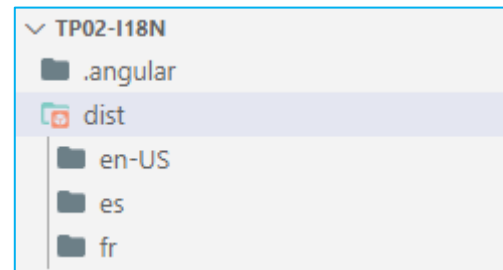
Editer Angular.json

L'Internationalisation I18N

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

```
messages.en.json  angular.json 1 X
formation-any-2023-09 > TP02-i18n > angular.json > ...
41   "node_modules/bootstrap-icons/font/bootstrap-icons.woff2": "src/styles.scss"
42   "src/styles.scss"
43   },
44   "scripts": [
45     "node_modules/bootstrap/dist/js/bootstrap.min.js"
46   ],
47   },
48   "configurations": {
49     "en": {
50       "localize": [
51         "en-US"
52       ]
53     },
```

ng build --localize



L'Internationalisation I18N

TP : Rajouter la traduction en espagnol

*Hola Angular !
Angular esta mejor !
assets/images/drapeaux/es/drapeau_es.jpg
Mi bandera*

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

L'Internationalisation I18N

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

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

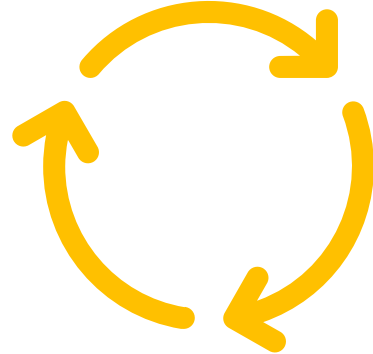
L'Internationalisation I18N

Option : afficher la traduction en mode serve !

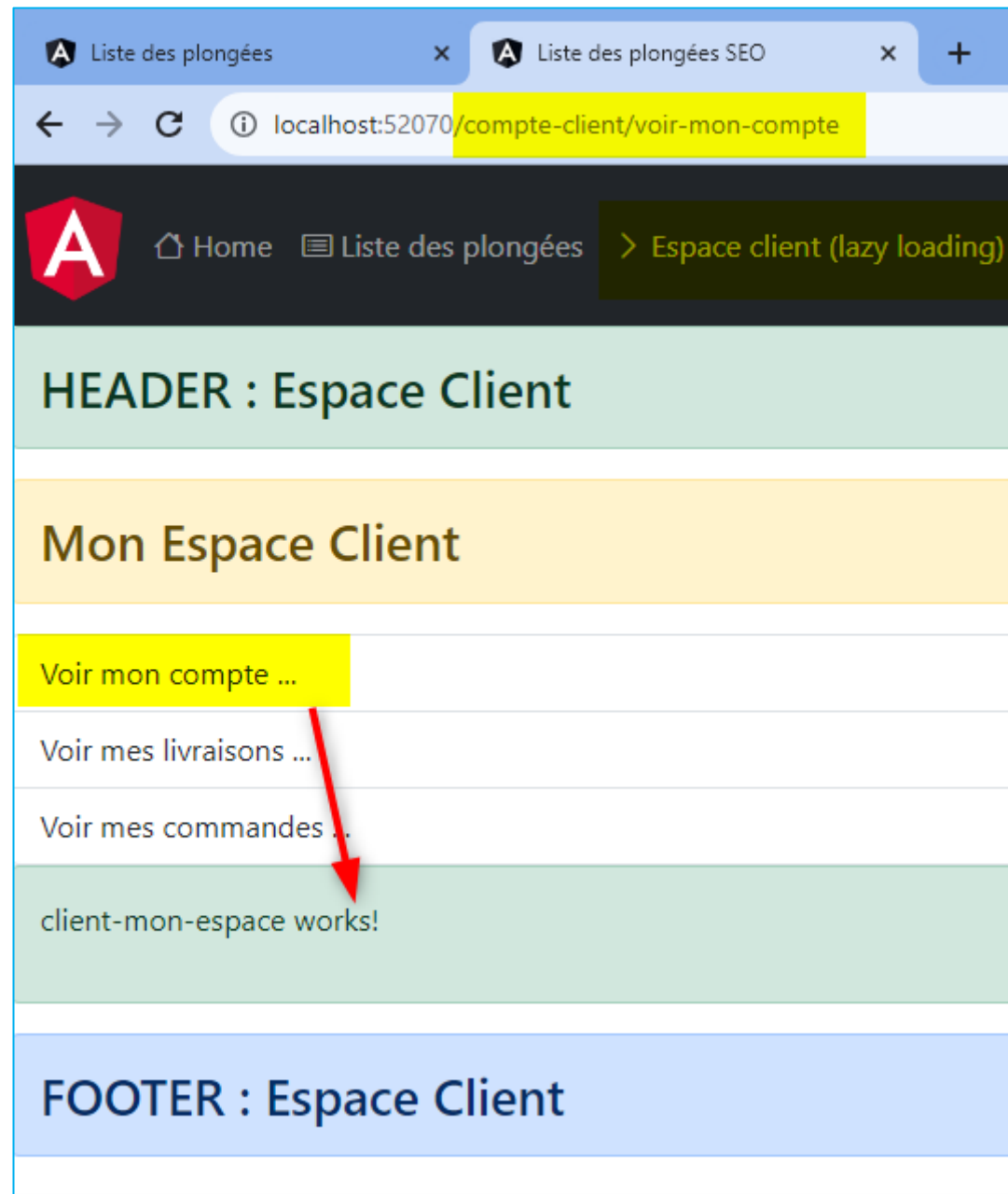
```
angular.json > $schema
78      extractLicenses : false,
79      "sourceMap": true,
80      "namedChunks": true
81    }
82  },
83  "defaultConfiguration": "production"
84 },
85 "serve": {
86   "builder": "@angular-devkit/build-angular:dev-server",
87   "configurations": {
88     "production": {
89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

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

Routage Avancé
Lazy Loading
Optimisation du build

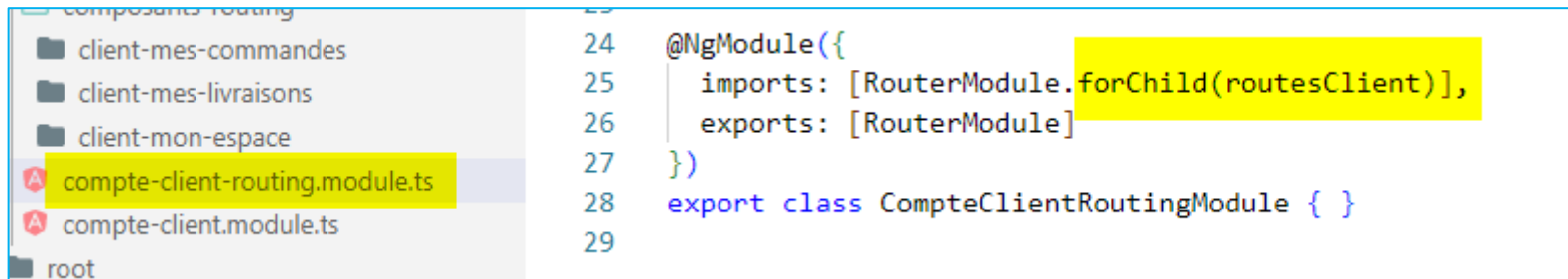


Routing : Concepts avancés – Le Lazy Loading



Routing : Concepts avancés – Le Lazy Loading

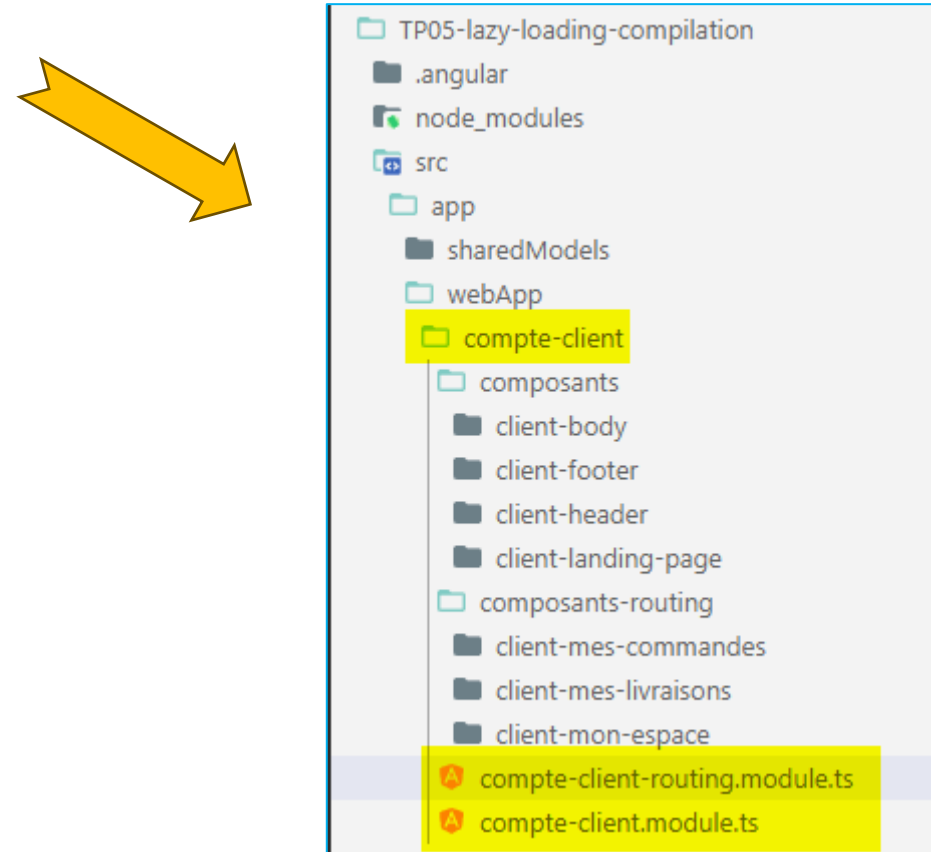
Créer le module compte-client avec l'option « --routing »
Cette option crée un module de routage spécifique
forChild



```
23
24 @NgModule({
25   imports: [RouterModule.forChild(routesClient)],
26   exports: [RouterModule]
27 })
28 export class CompteClientRoutingModule { }
29
```

Routing : Concepts avancés – Le Lazy Loading

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



Routing : Concepts avancés – Le Lazy Loading

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



```
PS C:\Angular\Injection-angular-2023-06\TP05-lazy-loading-compilation> npm run build

> tp02-bonnes-pratiques-injection-dependances@0.0.0 build
> ng build --stats-json

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

Initial Chunk Files	Names	Raw Size	Estimated Transfer Size
styles.32ab14ebec5beb97.css	styles	265.62 kB	29.18 kB
main.9281e4b71d326fdf.js	main	246.63 kB	66.78 kB
scripts.118140219e48fbd3.js	scripts	142.96 kB	41.36 kB
polyfills.ed4228387a31b6d8.js	polyfills	33.05 kB	10.64 kB
runtime.6565837a760cd7e6.js	runtime	2.73 kB	1.26 kB
	Initial Total	690.99 kB	149.22 kB

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

Routing : Concepts avancés – Le Lazy Loading

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

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



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

Le Lazy Loading : Mise en pratique

Etape #1 (no Lazy Loading)

The image shows a code editor with four files open, illustrating the setup for lazy loading in an Angular application. Red arrows highlight the relationships between the files.

- client-landing-page.component.html**: Contains the main structure of the landing page, including the header, body, and footer components.
- client-body.component.html**: Contains the main content area, including a list of links and a router outlet.
- compte-client-routing.module.ts**: Defines the routes for the 'compte-client' module, including 'mon-espace', 'mes-commandes', and 'mes-livraisons'.
- accueil.module.ts**: Defines the routes for the 'accueil' module, including 'accueil' and 'liste-des-films'.

Red arrows indicate the flow of module dependencies and routing configuration:

- From **accueil.module.ts** to **compte-client-routing.module.ts** (via the `CompteClientModule` import).
- From **compte-client-routing.module.ts** to **client-body.component.html** (via the `<router-outlet>` tag).
- From **client-body.component.html** to **client-landing-page.component.html** (via the `<app-client-body>` tag).

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-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
7
8 <li class="list-group-item">
9 <a class="nav-link"
10 routerLink="mes-commandes"
11 [routerLinkActive]="['lien-actif']"
12 >Mes commandes
13
14 <li class="list-group-item">
15 <a class="nav-link"
16 [routerLink]="['mes-livraisons']"
17 [routerLinkActive]="['lien-actif']"
18 >Mes livraisons
19
20
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

app-routing.module.ts

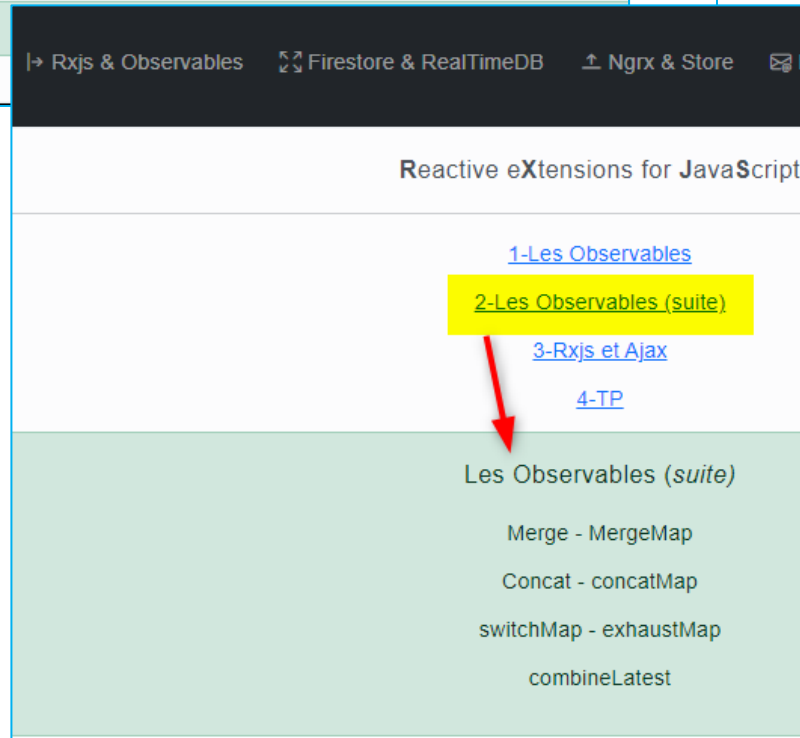
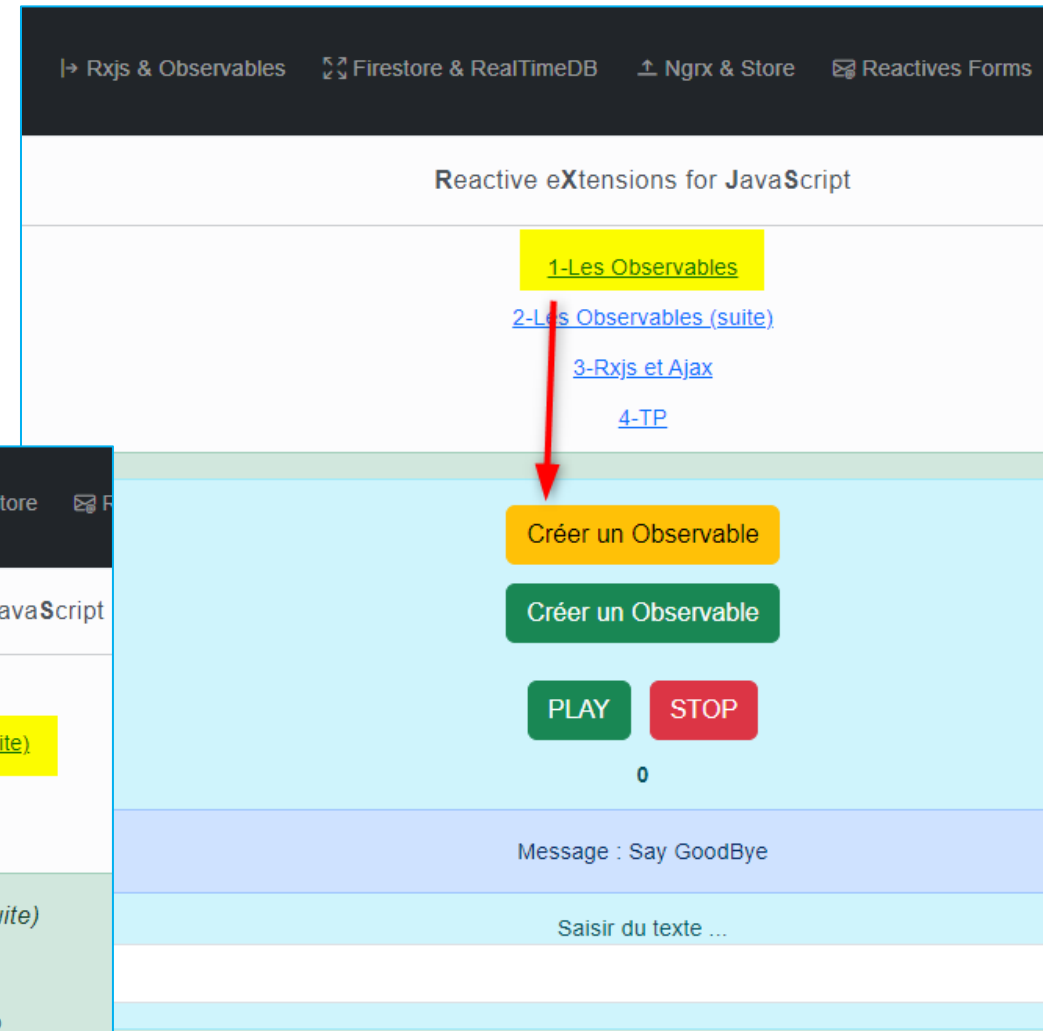
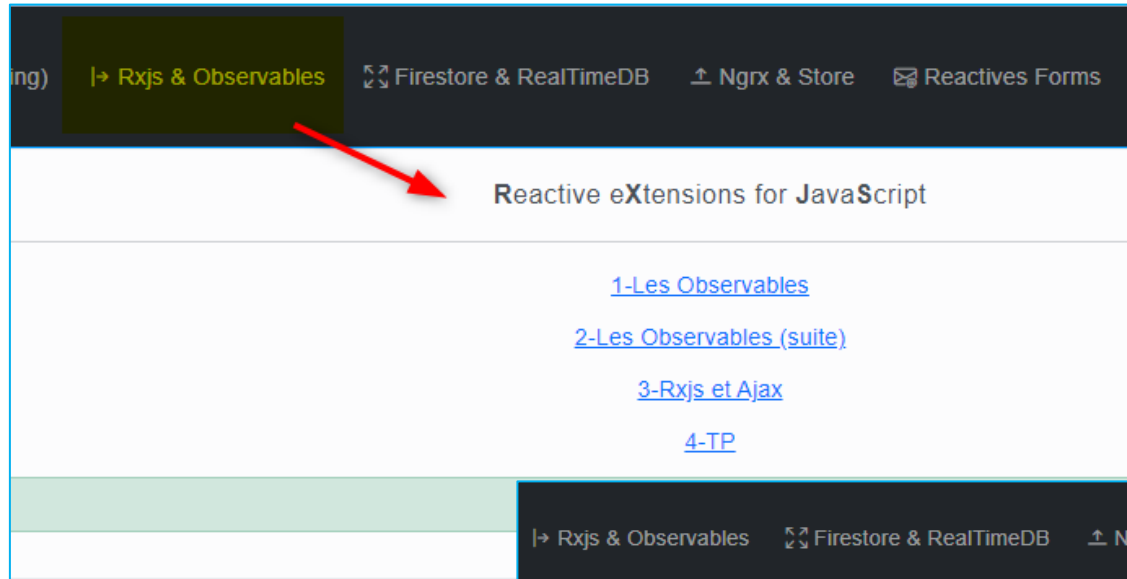
26
27 // 2ème étape : Avec lazy loading (promesses ES2015 😊)
28 // {
29 // path: 'compte-client',
30 // component: ClientLandingPageComponent,
31 // loadChildren:
32 // () => import('../webApp/formation/compte-client/compte-client.module')
33 // .then(
34 // (m) => {
35 // return m.CompteClientModule
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

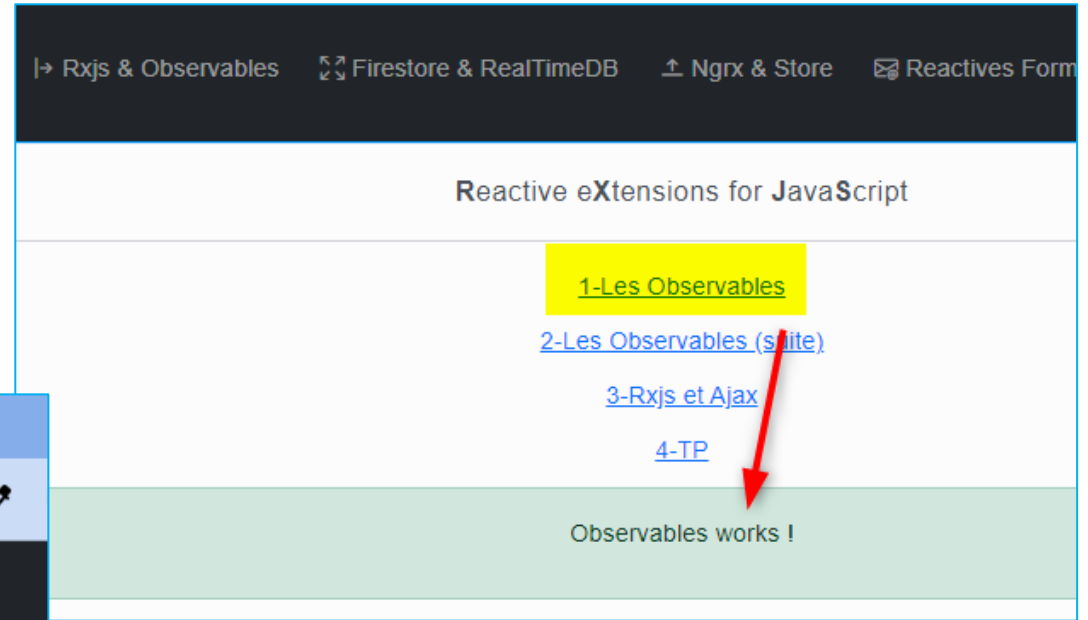
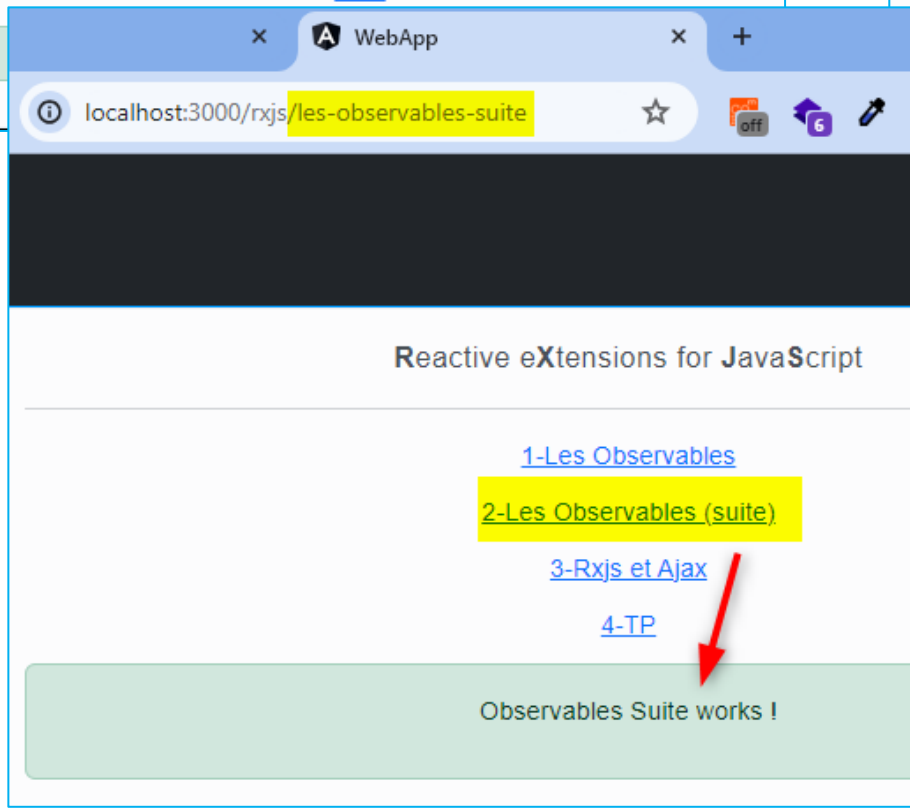
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ClientMonEspaceComponent } from './composants-routing/client-mon-espace/client-mon-espace.c
4 import { ClientCommandesComponent } from './composants-routing/client-commandes/client-commandes.com
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 })

TP : Déployer une arborescence en Lazy Loading

TP : déployer l'arborescence en Lazy Loading RXJS



TP : déployer l'arborescence en Lazy Loading RXJS



TP : déployer l'arborescence en Lazy Loading RXJS

The screenshot displays an IDE with the following components:

- EXPLORATEUR (File Explorer):** Shows the project structure. The `rxjs` folder is highlighted in yellow. Inside it, the `composants \ rxjs` folder is highlighted in blue. The `rxjs.component.html`, `rxjs.component.scss`, and `rxjs.component.ts` files are listed. The `composants-routing` folder is also highlighted in yellow, containing `ajax`, `observables`, `observables-suite`, and `tp` subfolders. The `rxjs-routing.module.ts` and `rxjs.module.ts` files are highlighted in yellow.
- rxjs.component.html:** Contains HTML code for a component. A red arrow points to the link for "1-Les Observables (suite)".

```
1 <div class="container alert alert-light">
2   <h4><strong>R</strong>eactive e<strong>X</strong>tensions for <strong>J</strong>ava<strong>S</strong>cript</h4>
3   <hr>
4   <p><a href="#">1-Les Observables</a></p>
5   <p><a href="#">1-Les Observables (suite)</a></p>
6   <p><a href="#">2-Les Observables (suite)</a></p>
7   <p><a href="#">3-Rxjs et Ajax</a></p>
8   <p><a href="#">4-TP</a></p>
9
10  <p class="alert alert-success">
11    <router-outlet></router-outlet>
12  </p>
13
14 </div>
```
- rxjs.module.ts:** Contains TypeScript code for the module. A red arrow points to the `RouterModule.forRoot` method.

```
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { RouterModule } from '@angular/router';
4 import { HttpClientModule } from '@angular/http';
5 import { RxjsRoutingModule } from './rxjs-routing.module';
6
7 import { ObservablesComponent } from './composants/observables/observables.component';
8 import { RxjsComponent } from './composants/rxjs/rxjs.component';
9 import { TpComponent } from './composants-tp/tp.component';
10 import { AjaxComponent } from './composants-ajax/ajax.component';
11 import { ObservablesSuiteComponent } from './composants-observables-suite/observables-suite.component';
12
13 @NgModule({
14   declarations: [
15     ObservablesComponent, RxjsComponent, ObservablesSuiteComponent,
16   ],
17   imports: [
18     CommonModule, RxjsRoutingModule,
19     HttpClientModule, RouterModule.forRoot([
20       { path: 'rxjs', loadChildren: './composants/rxjs/rxjs.module#RxjsModule' },
21     ])
22   ],
23 })
24 export class RxjsModule { }
```
- rxjs-routing.module.ts:** Contains TypeScript code for the routing module. A red arrow points to the `RouterModule.forChild` method.

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

TP : Tester le mode lazy Loading en compilation

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