ORSYS formation

Formation ANGULAR Développement Avancé

Chapitre 1

Animé par Michel BOCCIOLESI

Table des matières

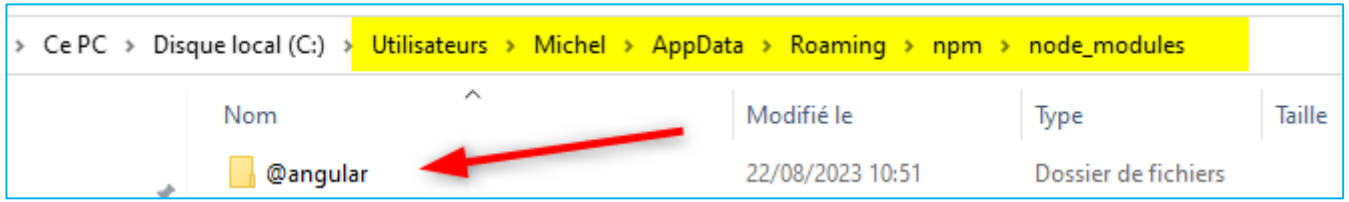
Récupérer les sources du projet de formation ici

Installation et configuration

https://nodejs.org
https://angular.io/
https://www.npmjs.com/package/@angular/cli

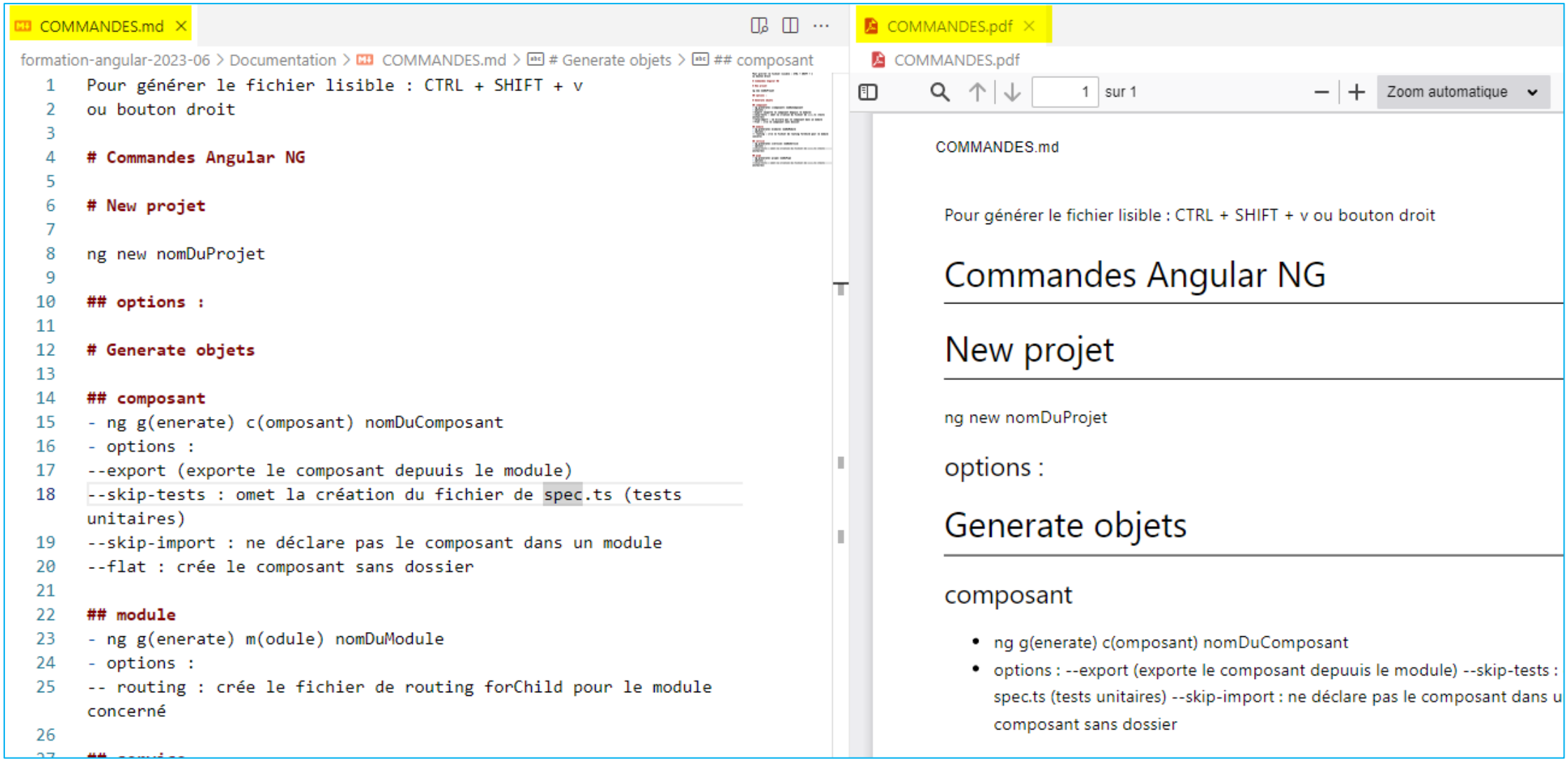
- npm install -g @angular/cli (global dans le path utilisateur)

  /home/michel (linux)



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

Documenter son projet



COMMANDES.md

formation-angular-2023-06 > Documentation > COMMANDES.md > # Generate objets > ## composant

```
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 depuuis le module)
--skip-tests : omet la création du fichier de spec.ts (tests unitaires)
--skip-import : ne déclare pas le composant dans un module
--flat : crée le composant sans dossier

## module
- ng g(enerate) m(odule) nomDuModule
- options :
-- routing : crée le fichier de routing forChild pour le module concerné
```

COMMANDES.pdf

COMMANDES.pdf

1 sur 1 — Zoom automatique

COMMANDES.md

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

Commandes Angular NG

New projet

ng new nomDuProjet

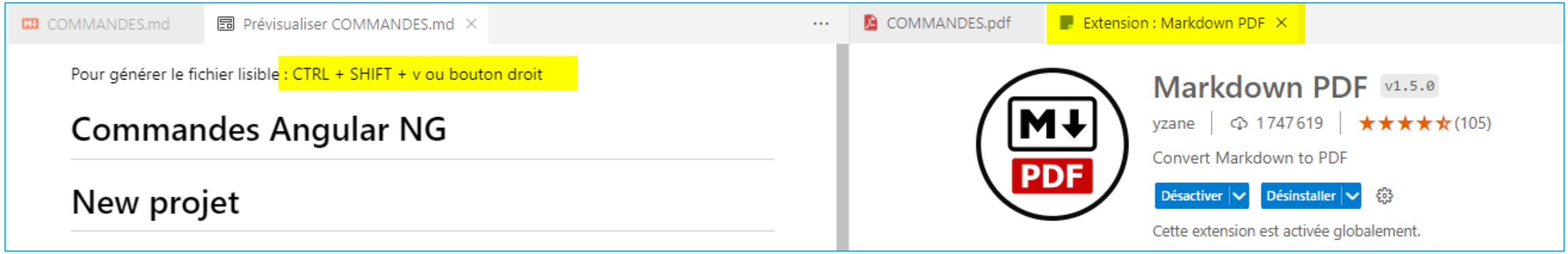
options :

Generate objets

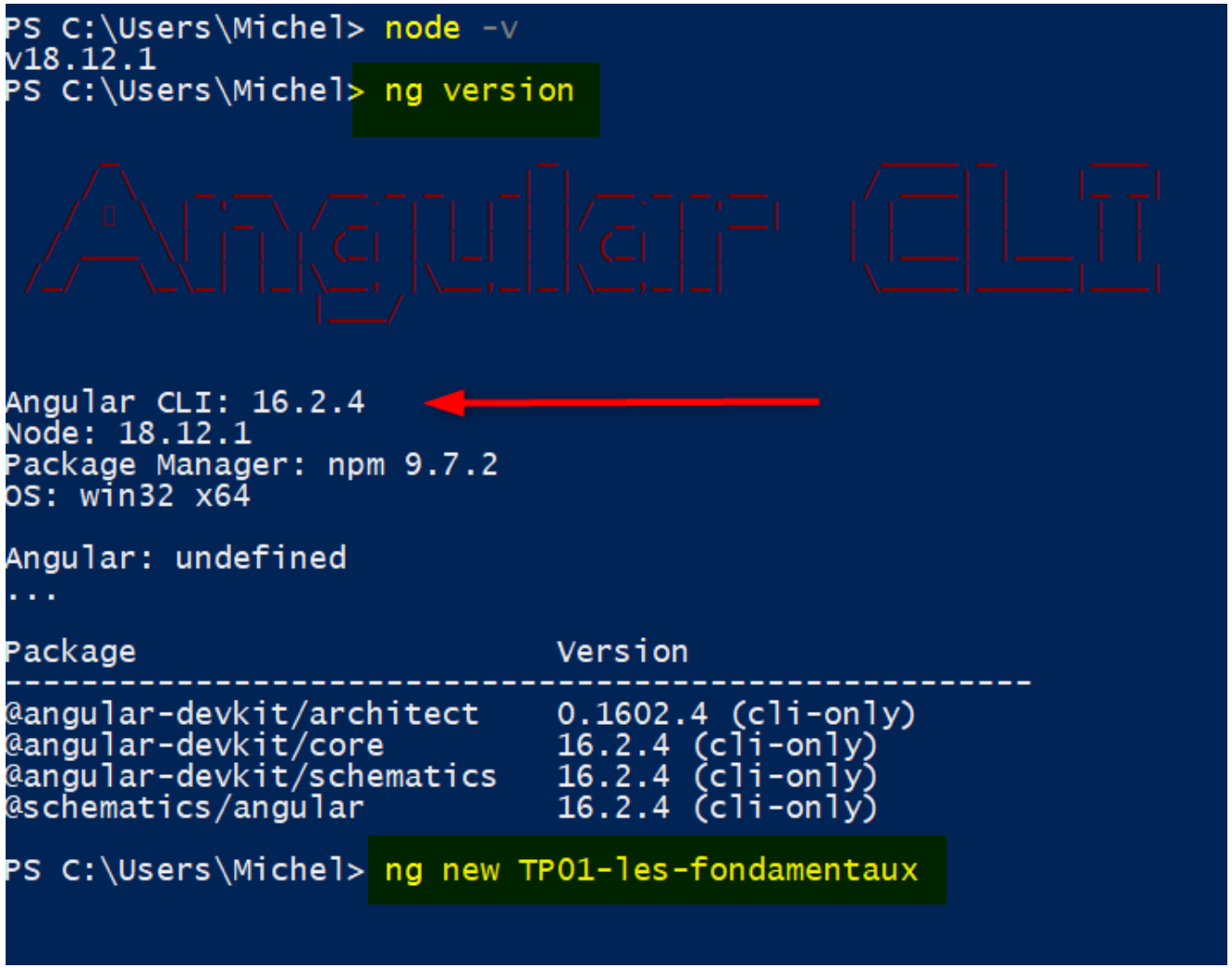
composant

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

Documenter son projet



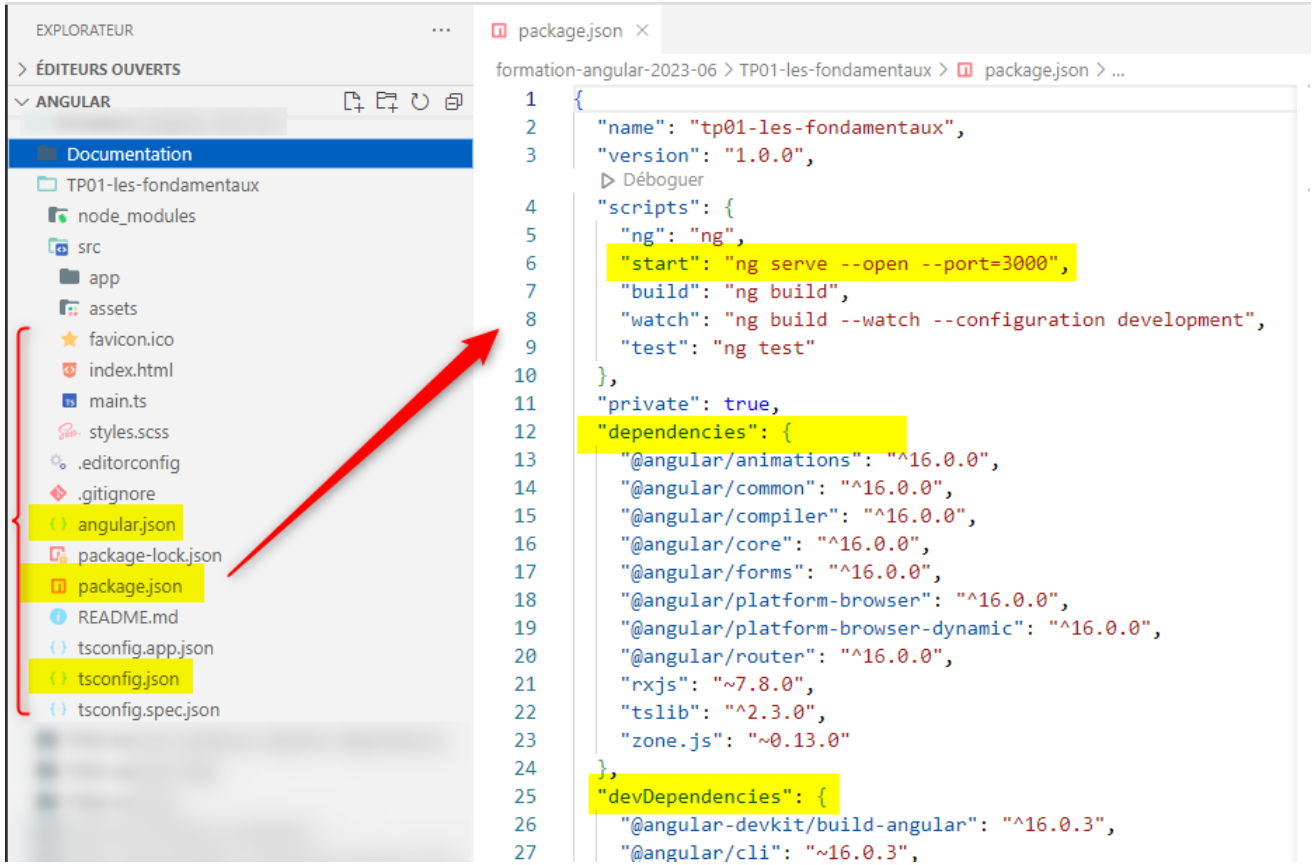
https://nodejs.org/fr



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

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*.

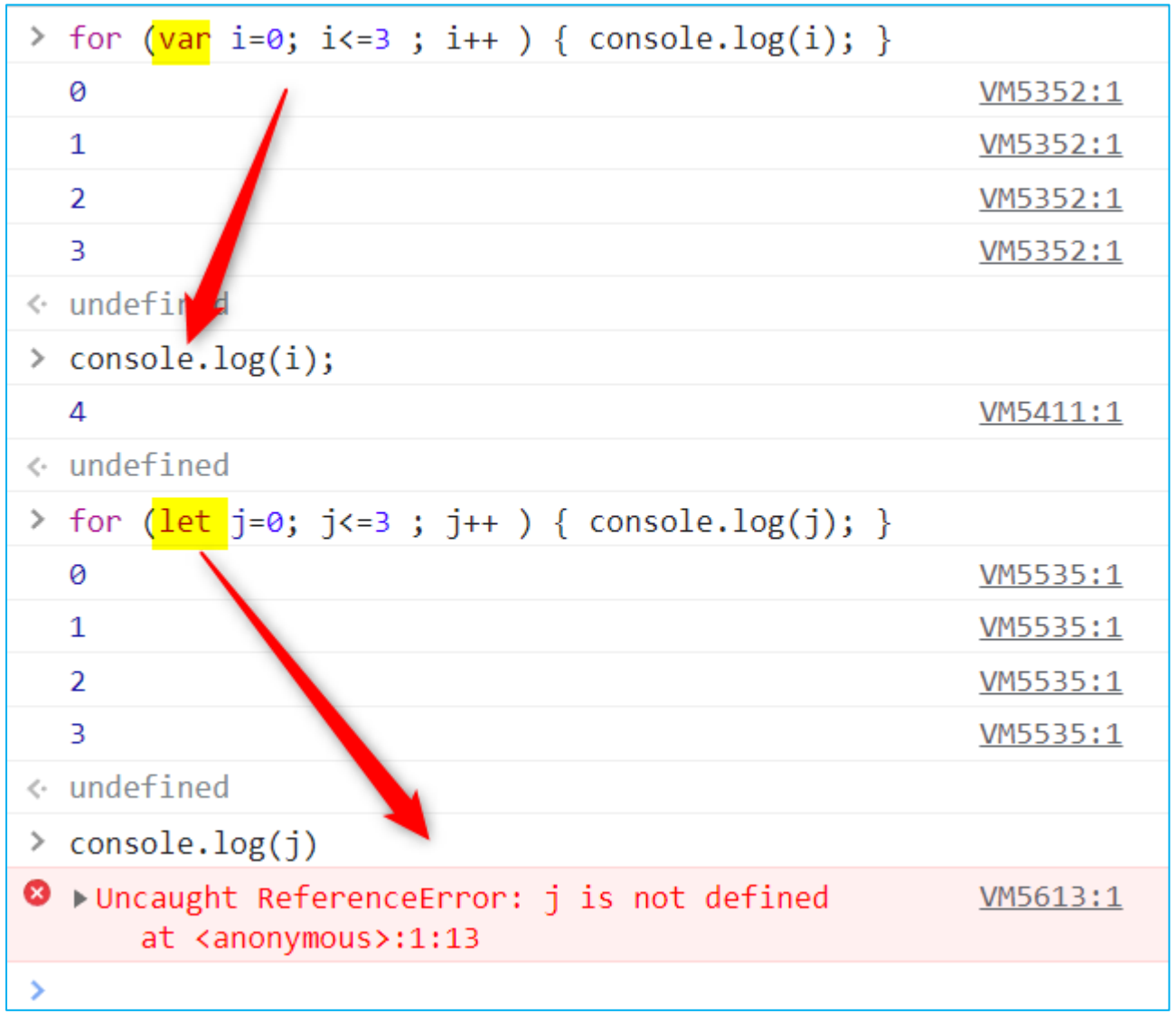
Les améliorations les plus importantes du langage concernent :

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

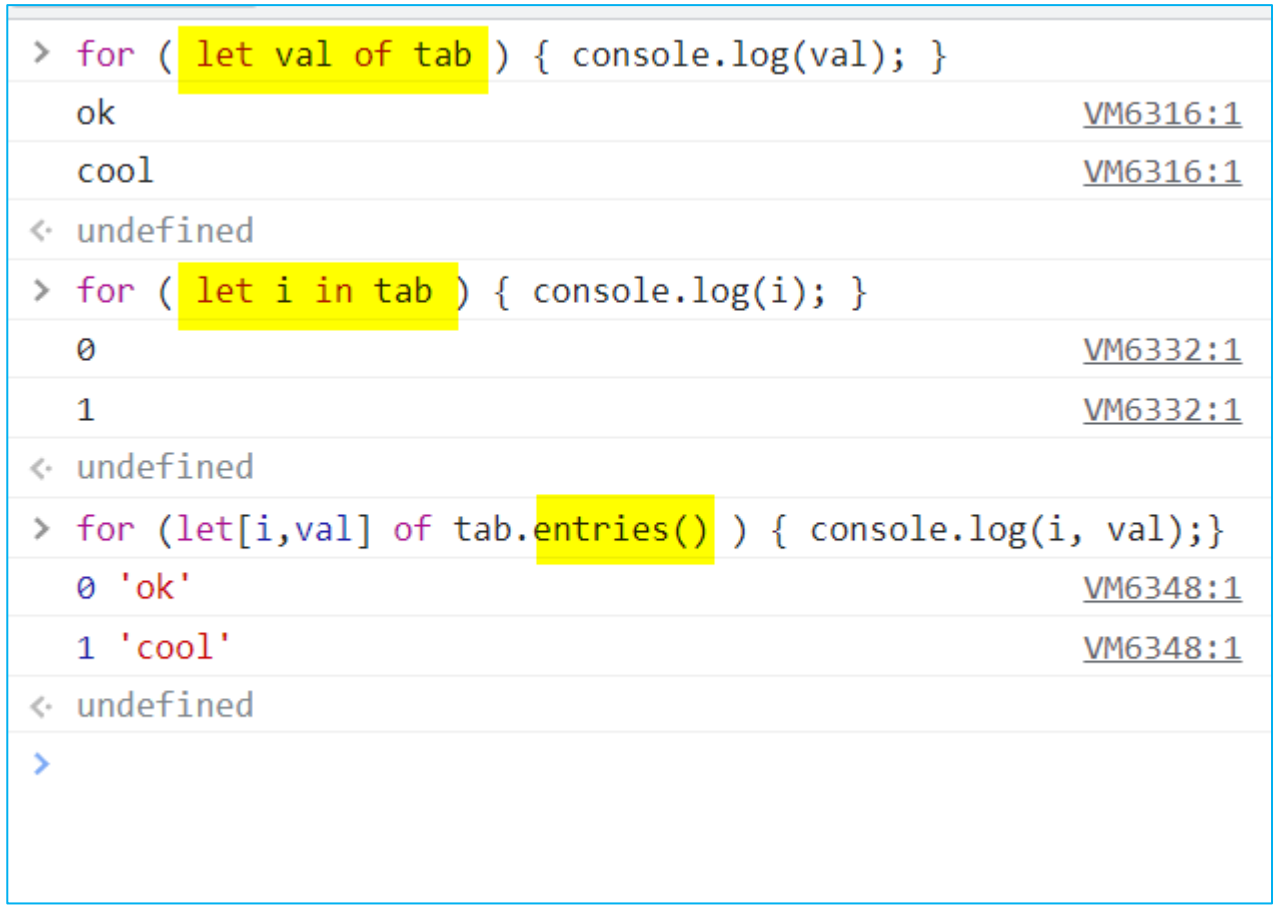
3. L'écriture simplifiée des FAT ARROWS
4. La string interpolation des variables avec ${maVariable} et la concatenation avec les back stits
5. Les itérateurs et les boucles FOR OF, FOR IN et FOR OF ENTRIES
6. Les spread operators pour déstructurer les tableaux
7. La notion de promesses afin de gérer les événements et les procédures asynchrones
8. Les objets et les CLASS*

```
<script src="js/index.js" type="module" defer></script>
```

Ecmascript 2015+



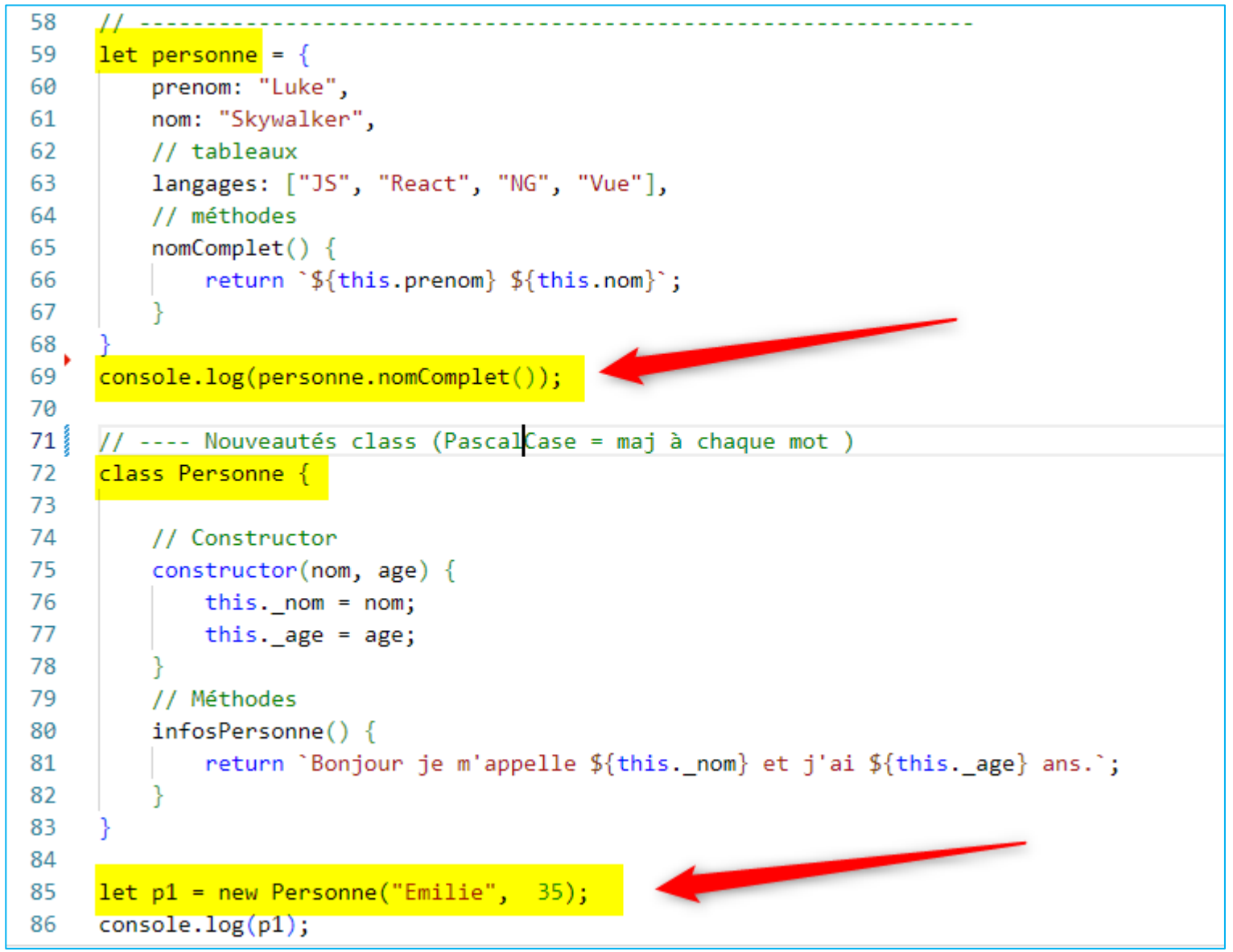
Ecmascript 2015+



Ecmascript 2015+

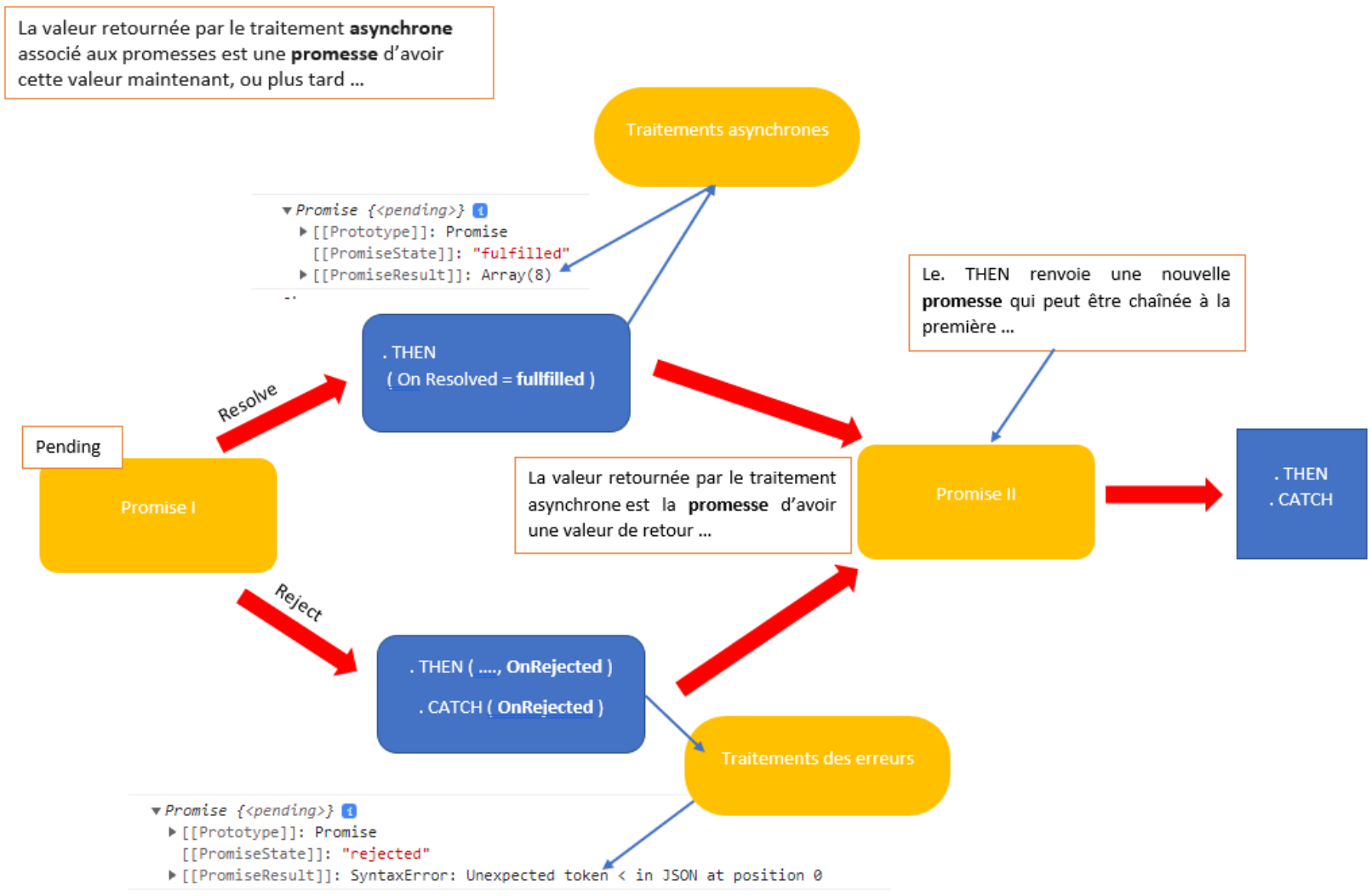
```
1   // nouveautés (fonctions)
2   function direBonjour(nom = "SkyWalker") {
3       // passage d'arguments par défaut
4       // console.log("Bonjour " + nom);
5
6       // string interpolation des variables
7       console.log(`Bonjour ${nom}`);
8       document.querySelector("#resultat").innerHTML = `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);
```

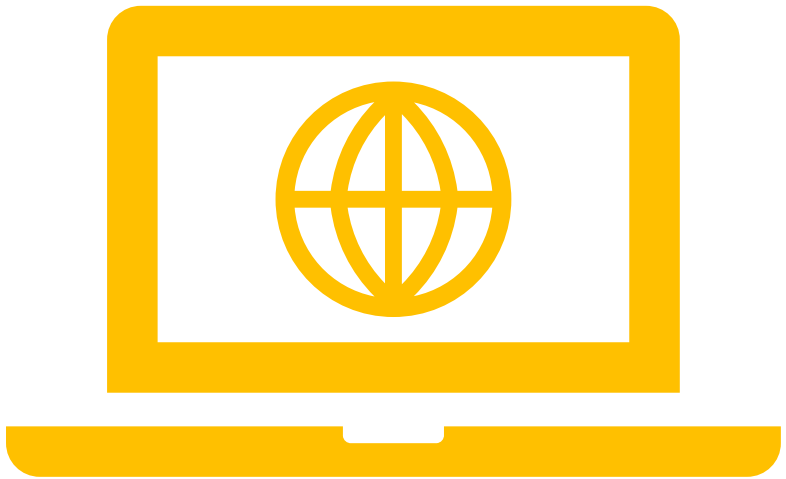
Ecmascript 2015+



Ecmascript 2015+

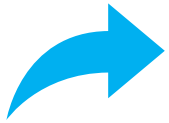
```
export const fnFilm = () => {
    // ES6 : fonction fetch
    // simplication d'écriture d'Ajax
    // amélioration du process JS ( Promises )

    //  1-  Request Query (requête Infos):
    // const _url = 'https://test.webjs.fr/films.json';
    const _url = 'https://dev.webjs.fr/films.json';
    // const _url='http://127.0.0.1:3000/json/films.json'; // en local avec un serveur json

    //  2-  Request Init Query
    //  définir la méthode (get ou post) ... ou patch ou put ou delete)
    const _method = 'get';
    // définir nos entêtes HTTP
    const _headers = new Headers();

    _headers.append('Content-Type', 'text/json'); //type mime (ex:image/png)
    // _headers.append('Access-Control-Allow-Origin', 'cors');

    //  3- Ecriture du FETCH
    // fetch renvoie une promesse
    fetch(
        _url,
        {
            method: _method,
            headers: _headers
        }
    ).then(
        // si on arrive à avoir une réponse du serveur
        // état onFullFilled
        (responseHTTP) => {
            // le then récupère une réponse 200 ok 404
            console.warn('On Fullfilled du 1er Then');
            console.log(responseHTTP);
            console.log(responseHTTP.body);
```

PWA



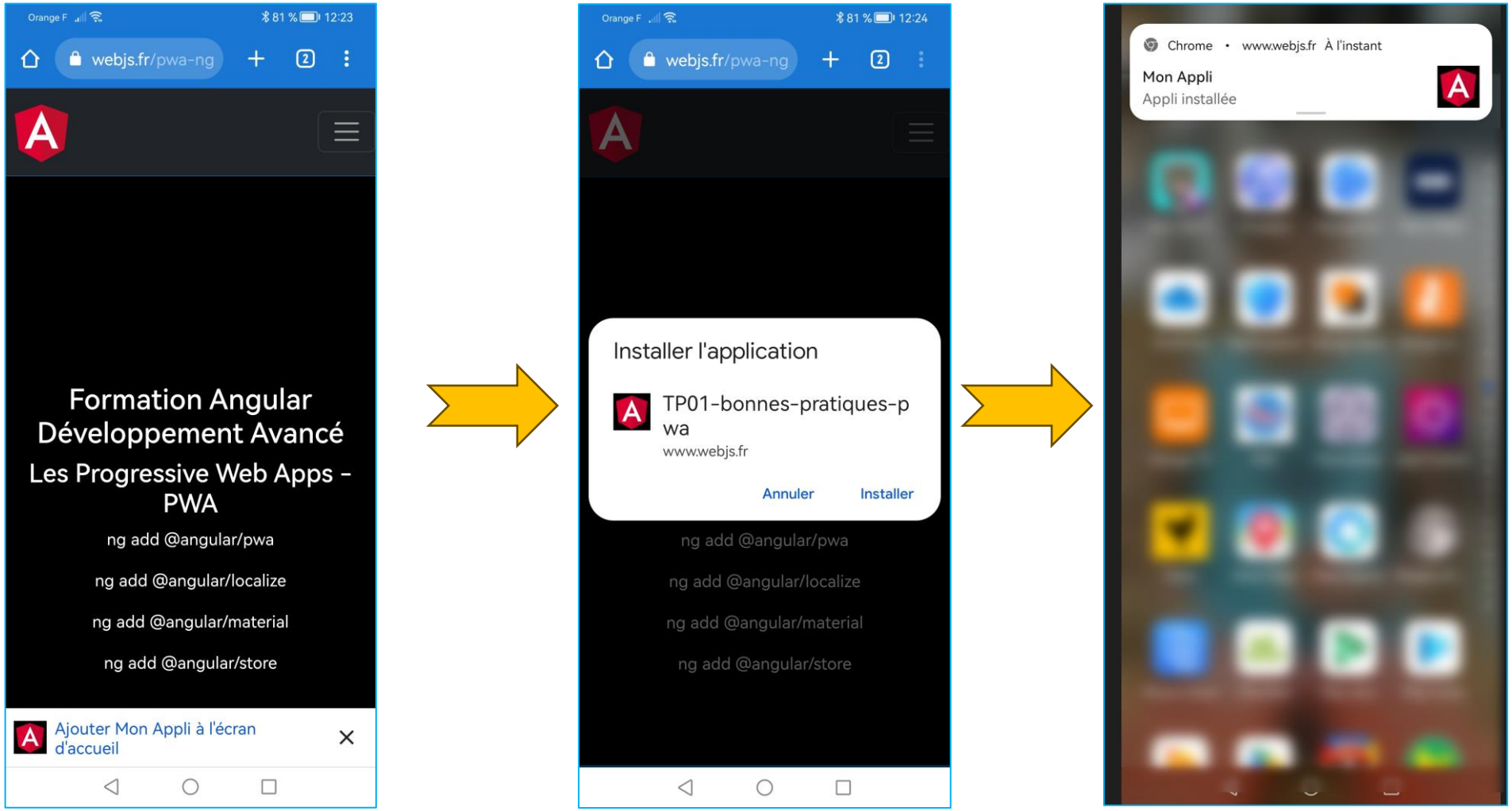
PWA – les Progressive Web Apps ?

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/

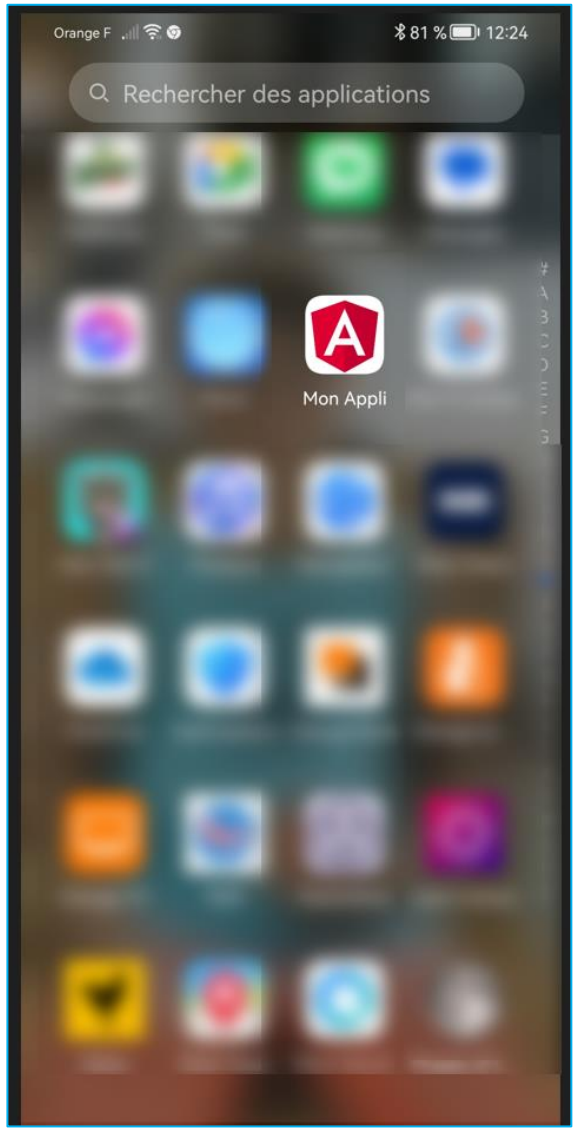
PWA – (le fichier web-manifest)



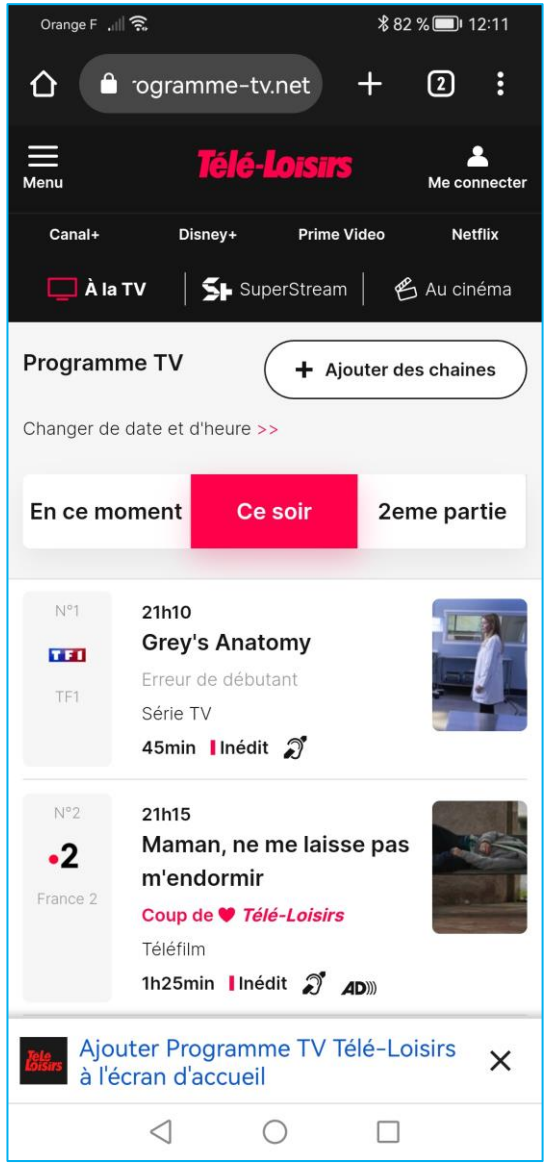
Références :

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

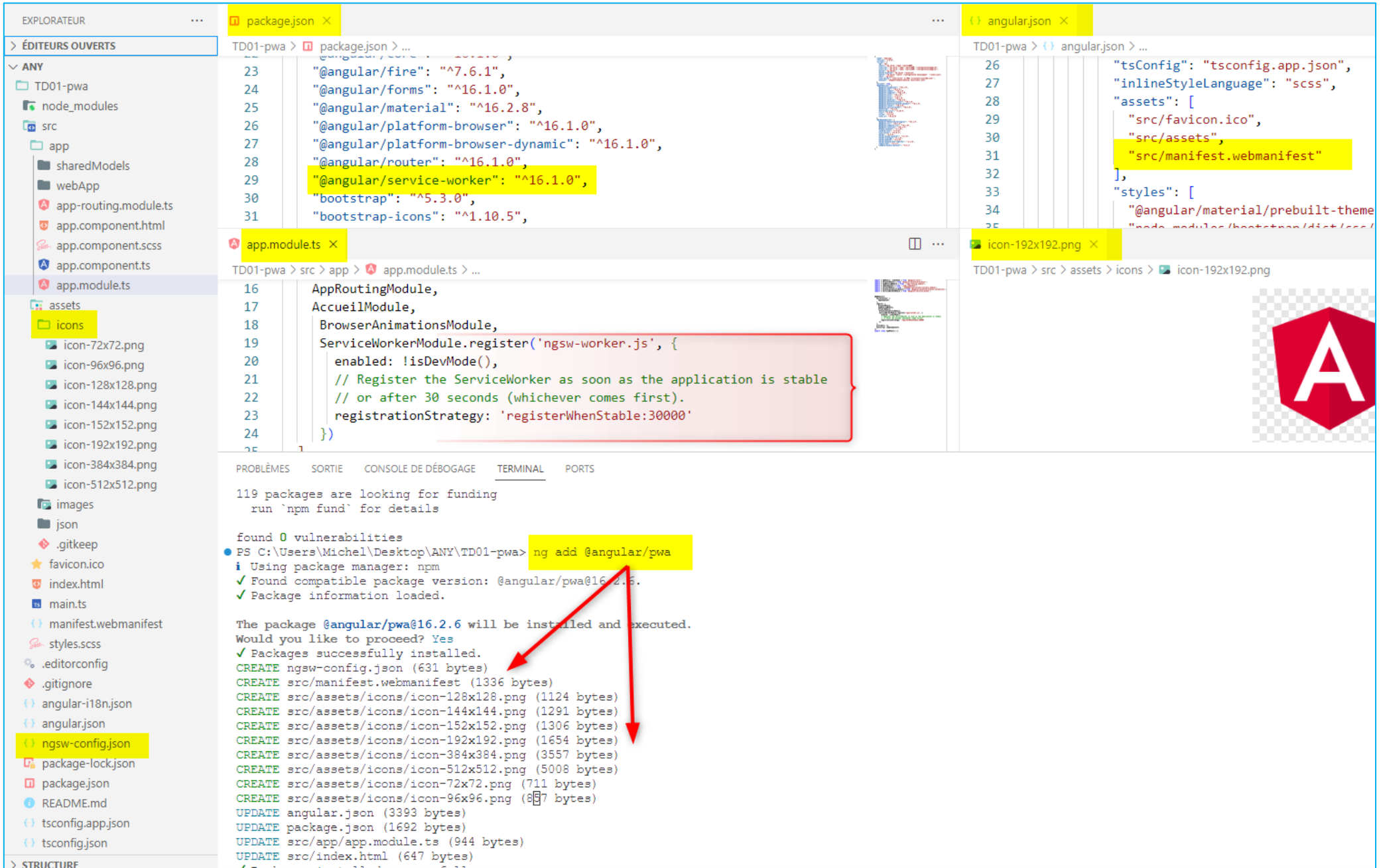
PWA – les Progressive Web Apps



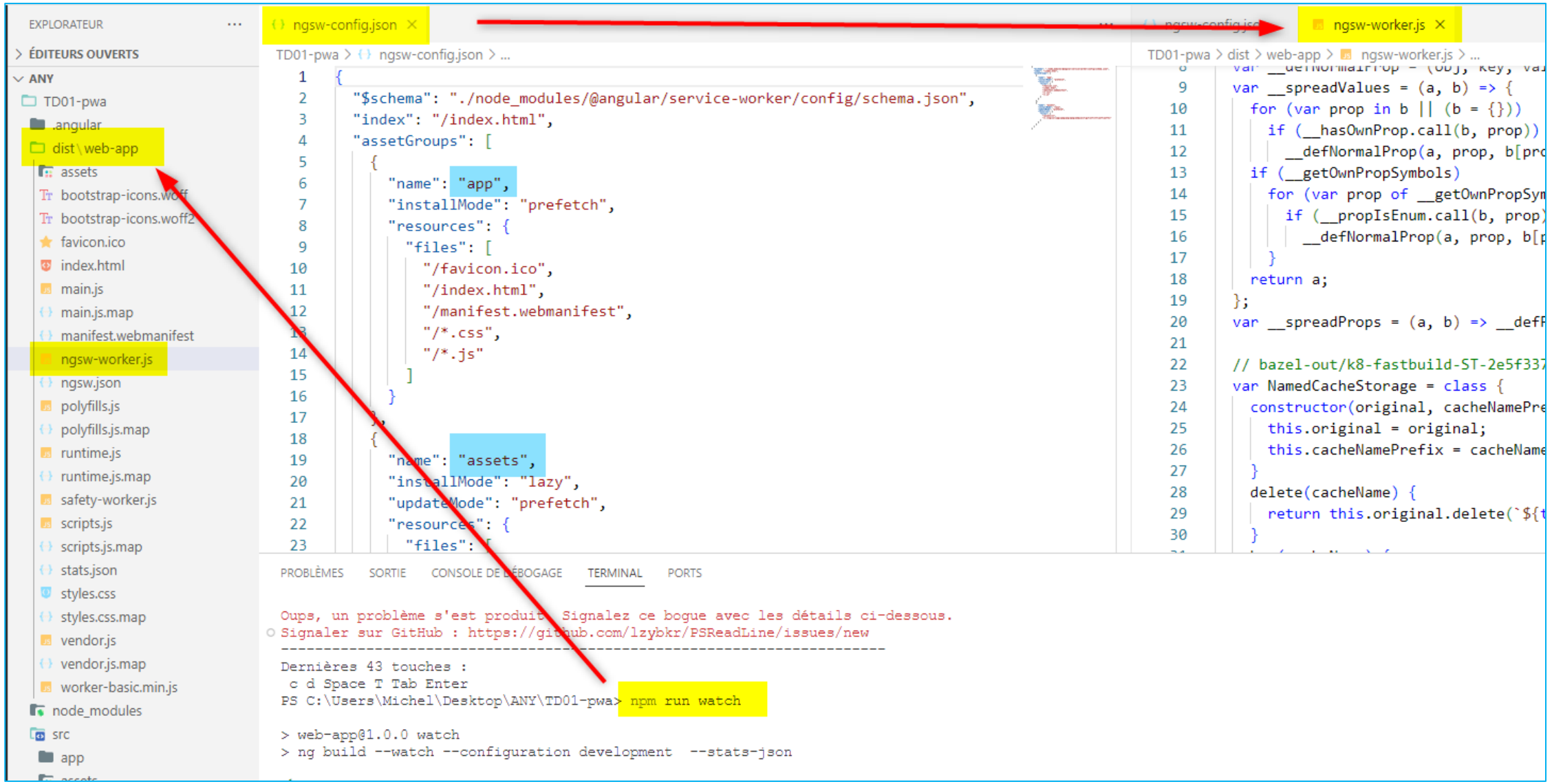
PWA – les Progressive Web Apps



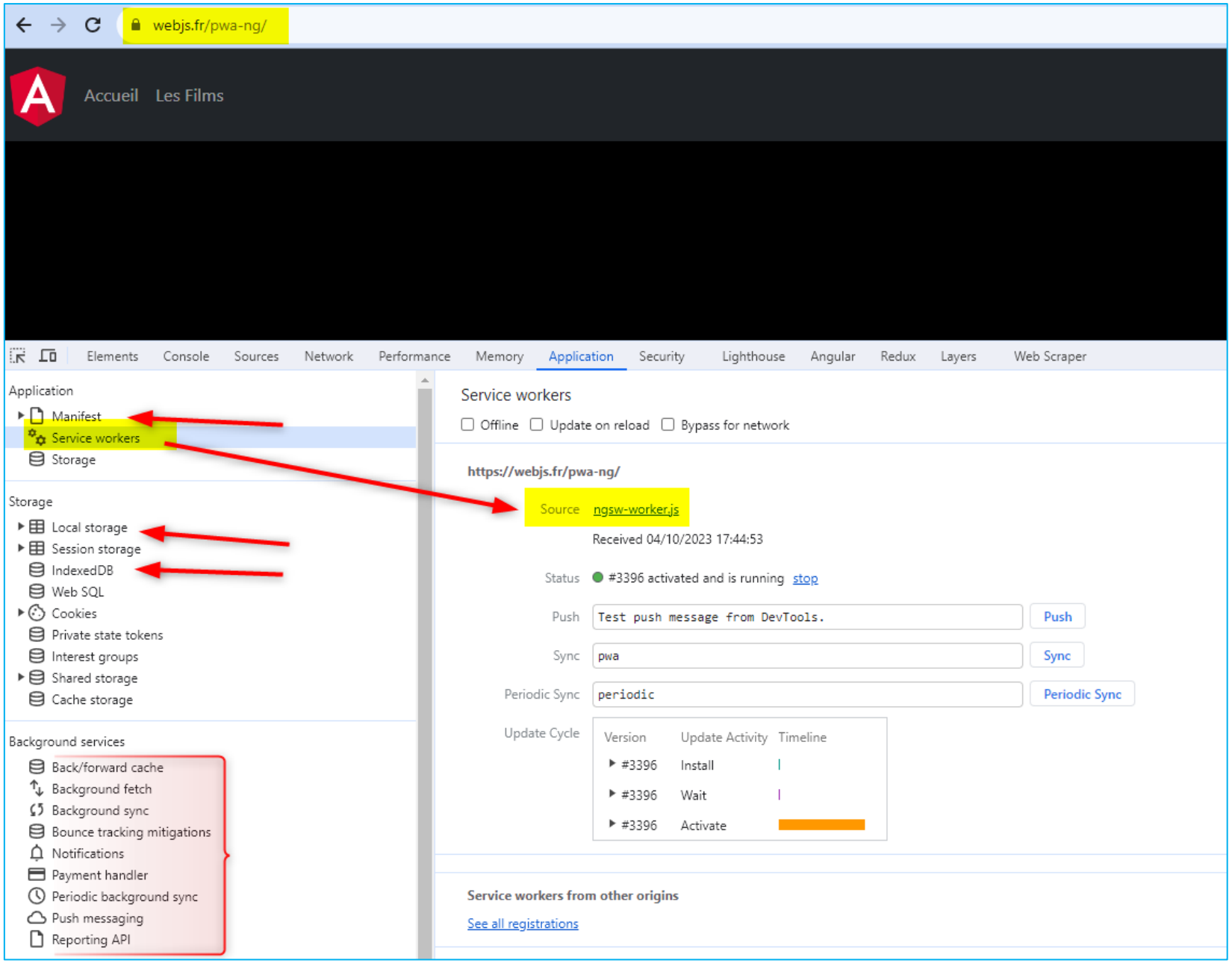
Commande d'installation et d'intégration de package : **ng add @angular/pwa**



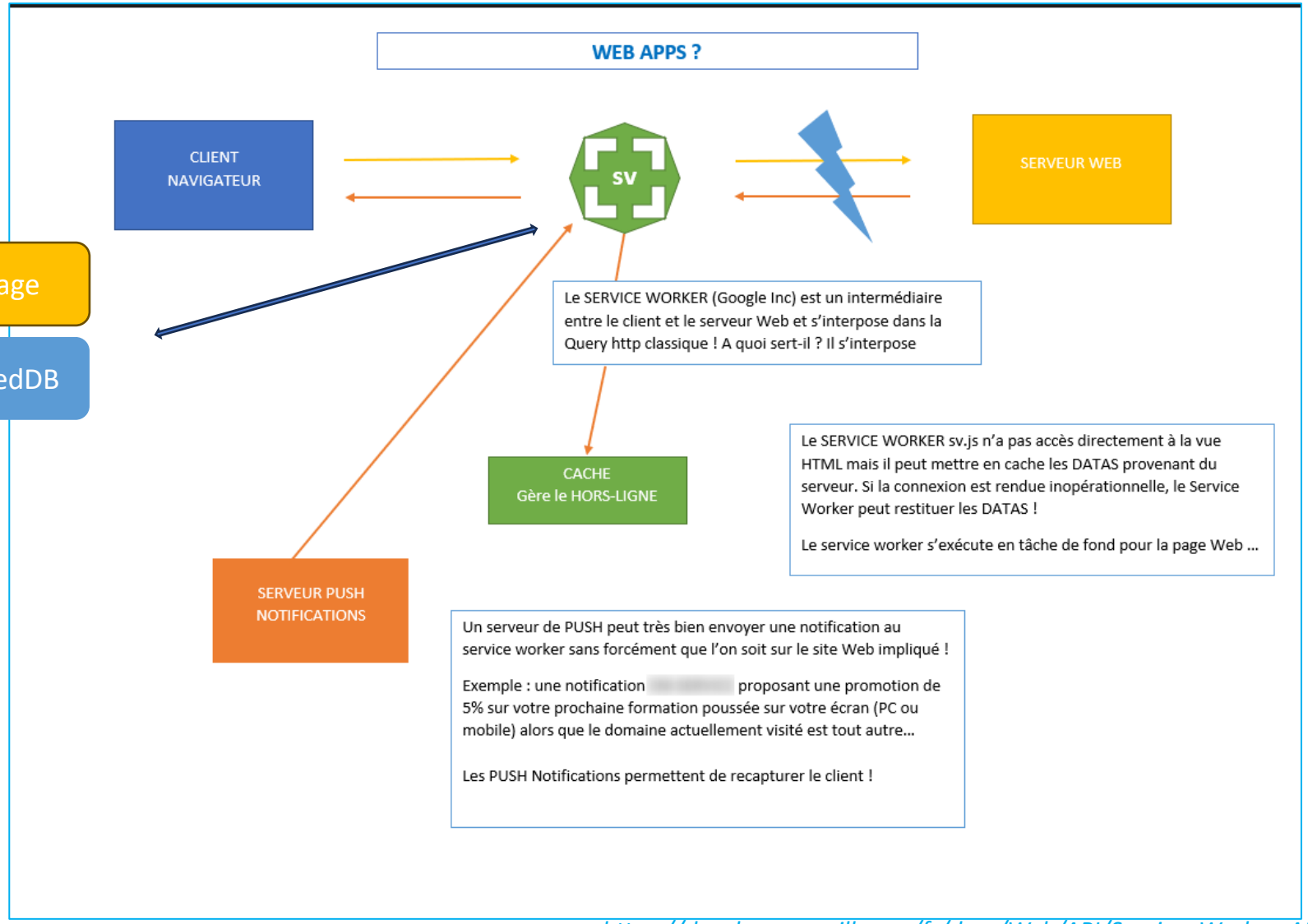
npm run build – npm run watch



PWA – les Progressive Web Apps



PWA – le Service Worker



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 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));
    }
};
```

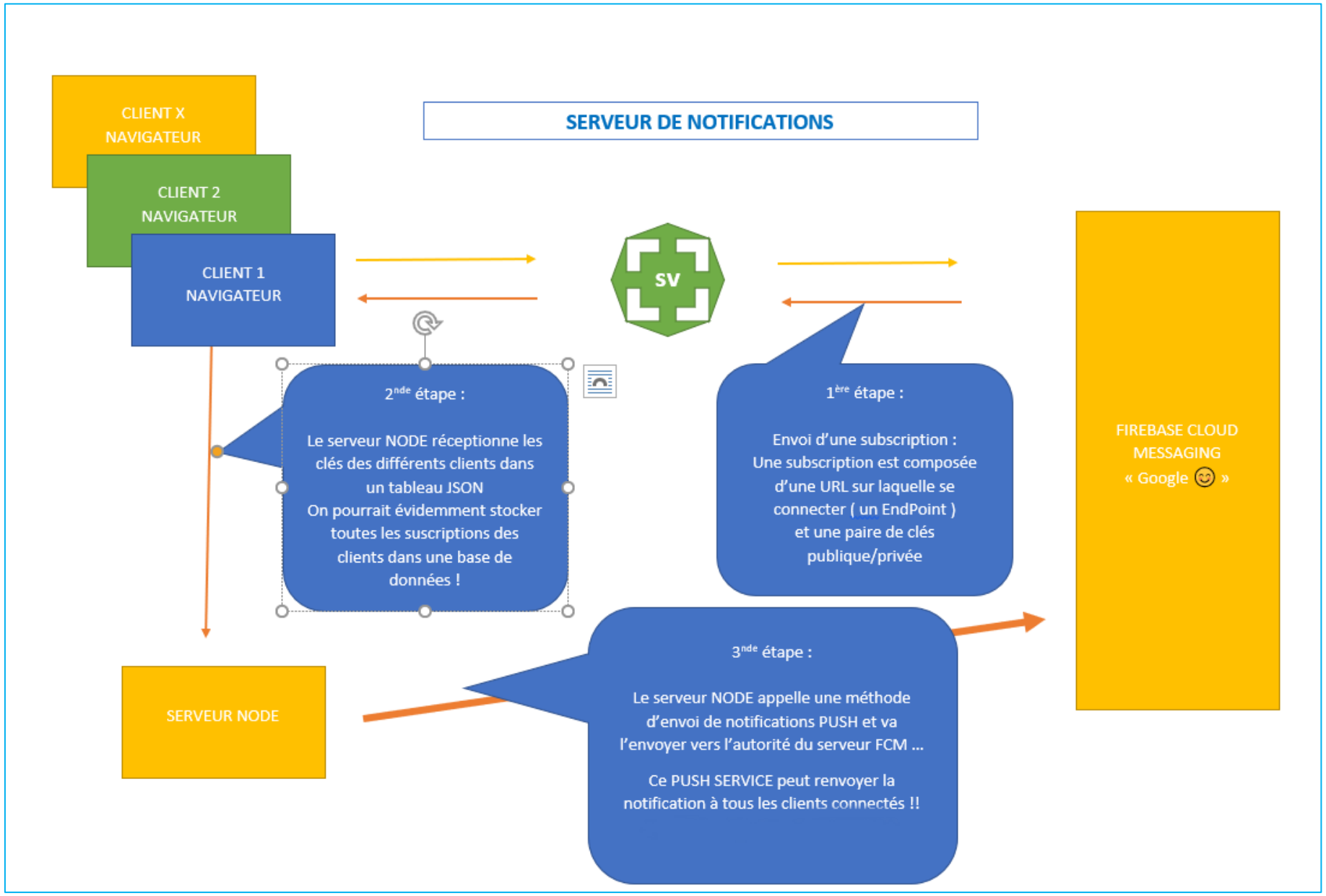
```
const delNotes = () => {
    localStorage.clear();
    eltUlListe.innerHTML='';
};
```

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');
    }
);
```

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

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

    //demander les permissions
    navigator.permissions.query(
        {
            name:'periodic-background-sync'
        }
    ).then(
        (e) => {
            const statusQueryPerm = e.state;
            console.warn(statusQueryPerm);
            if (statusQueryPerm==='granted') {
                console.log(`---> Periodic Background sync est autorisé`);
            }
        });
// ------------------------------------------------
navigator.serviceWorker.ready
.then(
    async registration => {
        await registration.periodicSync.register(
            'tag-periodic-sync',
            {
                // minInterval: 24 * 60 * 60 * 1000
                // 24 heures => daiy news
                //  exprimé en ms
                minInterval : 5
            }
            );
    }
);
}
```

PWA – les Progressive Web Apps

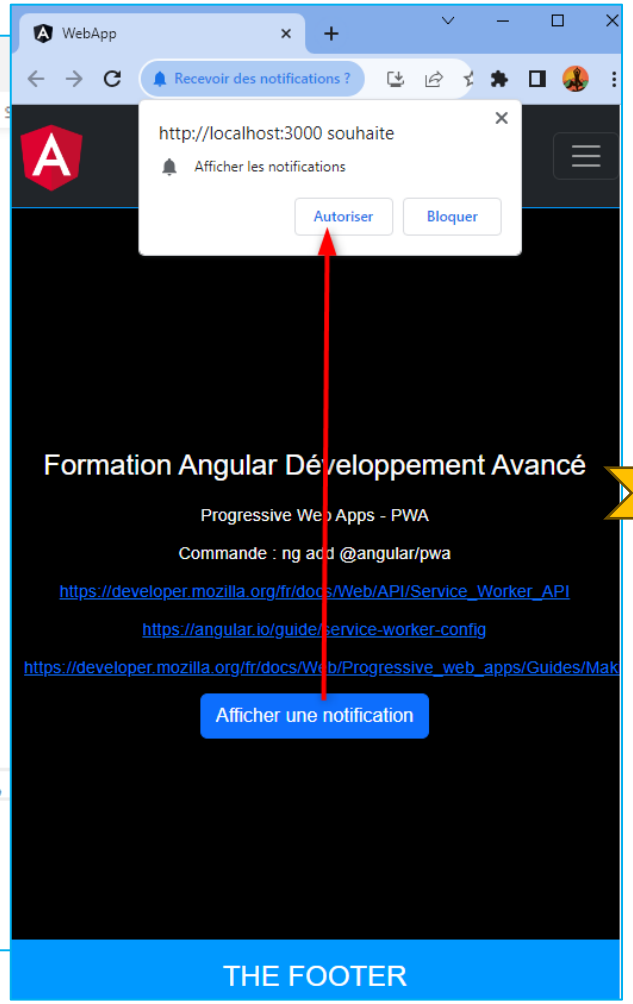


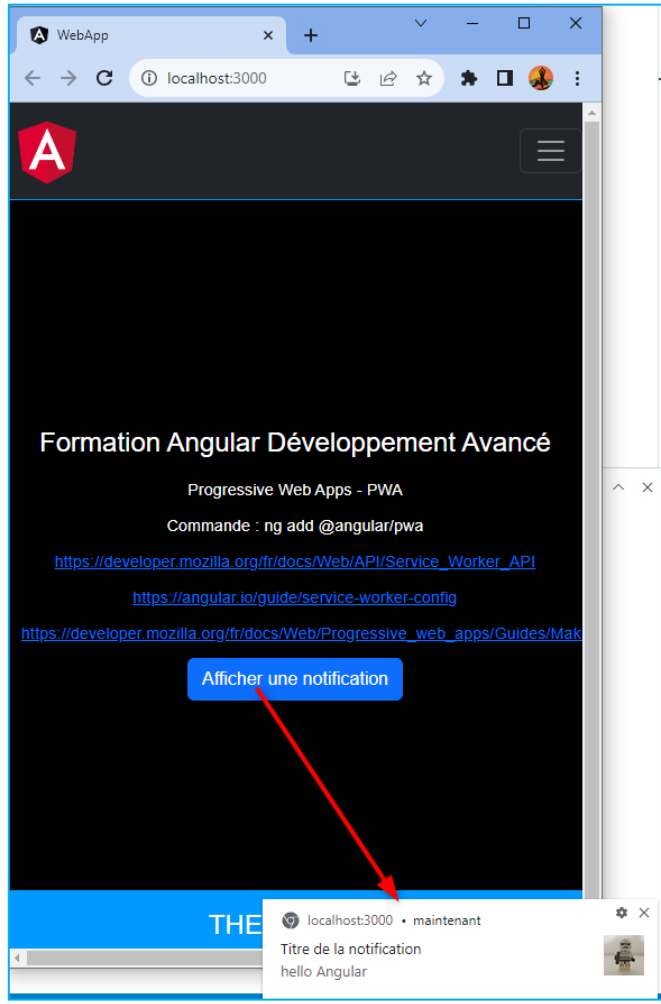
PWA – Notification

body.component.ts

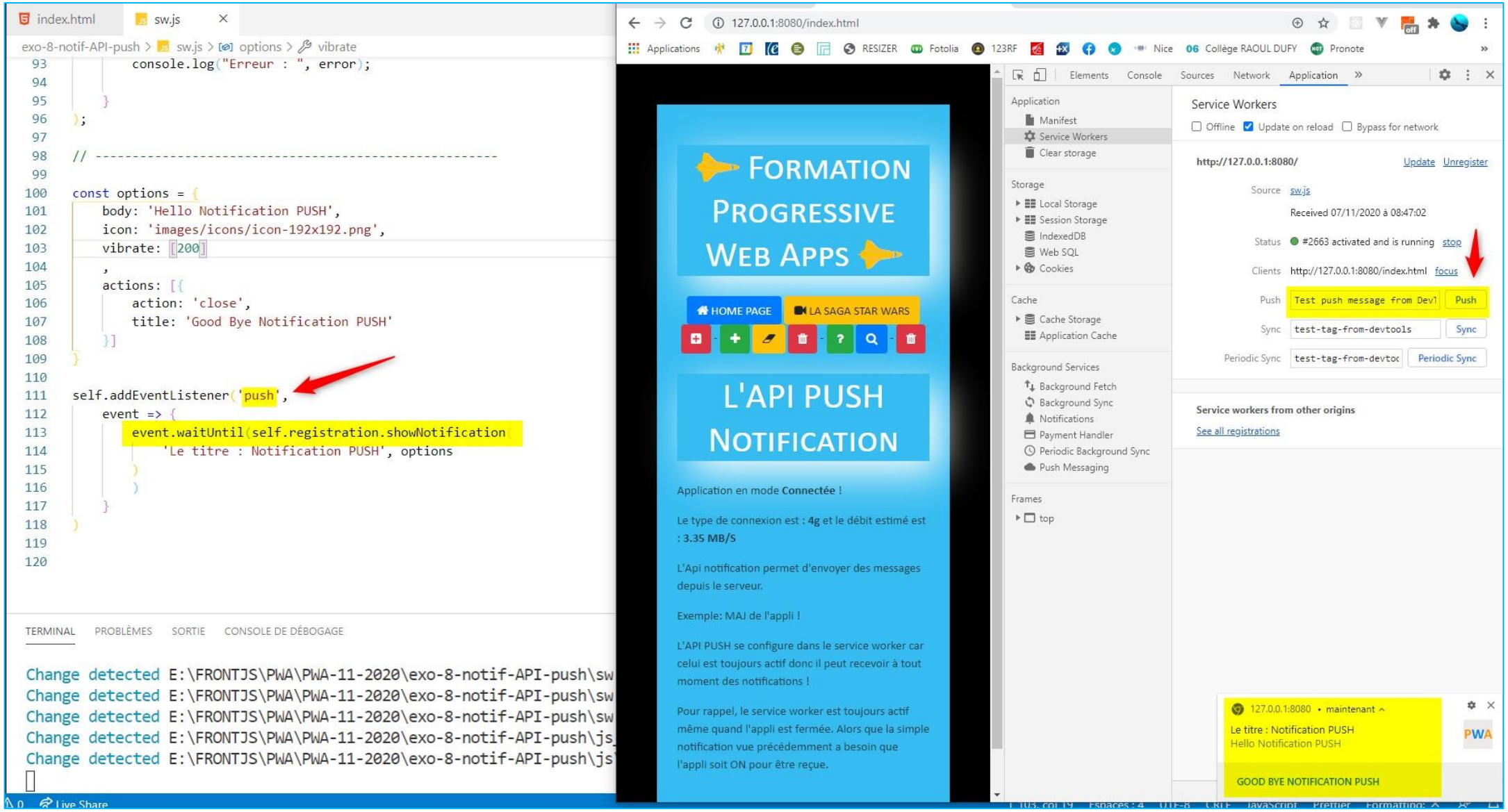
TD01-pwa > src > app > webApp > root > accueil > body > body.component.ts > BodyComponent

```
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',
              options);
39          }
40        }
41      );
```





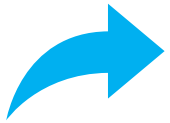
PWA – Notification



L’InternationalisatioN I18N

L’InternationalisatioN I18N

Comment traduire dans un projet Angular ?

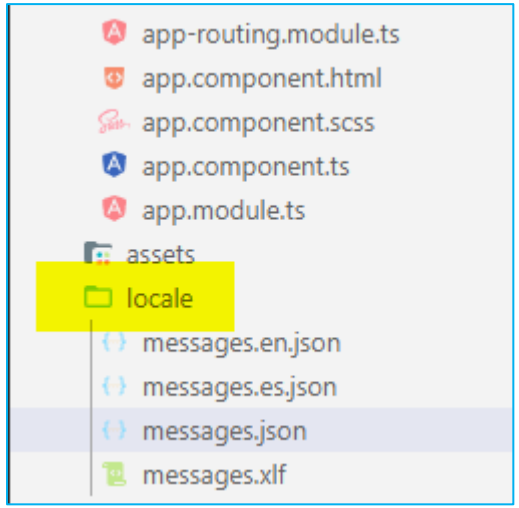


Commande d’installation et d’intégration de package : ng add @angular/localize

L'InternationalisatioN I18N

L'InternationalisatioN I18N

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



messages.en.json | messages.json

formation-any-2023-09 > TP02-i18n > src > locale > messages.json > ...

```
{
  "locale": "fr",
  "translations": {
    "div-vue": "{$START_PARAGRAPH}L'internationalisation I18N{$CLOSE_PARAGRAPH}{$S
    "4044202032648885552": "Bonjour Angular !",
    "slogan": "Angular est le meilleur Framework !",
    "4266238317959690326": "assets/images/drapeaux/fr/drapeau_fr.jpg",
    "4425774926678752758": "Mon Drapeau",
    "3194024123575565444": "www.angular.io",
    "8703060702766654522": "Doc Angular IO",
    "4435388665562277072": "Formation Angular Développement Avancé"
  }
}
```

L'InternationalisatioN I18N

Editer Angular.json

messages.en.json | angular.json 1

formation-any-2023-09 > TP02-i18n > angular.json > ...

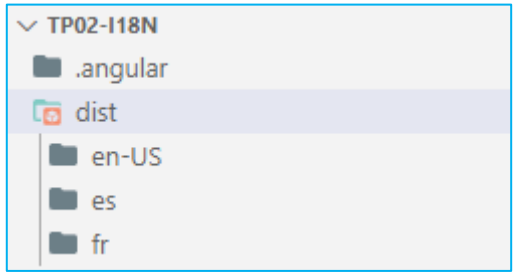
```
{
    "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
    "version": 1,
    "newProjectRoot": "projects",
    "projects": {
        "webApp": {
            "i18n": {
                "sourceLocale": "fr",
                "locales": {
                    "en-US": "src/locale/messages.en.json",
```

messages.en.json | angular.json 1

formation-any-2023-09 > TP02-i18n > angular.json > ...

```
                "node_modules/bootstrap-icons/font/bootstra
                "src/styles.scss"
            ],
            "scripts": [
                "node_modules/bootstrap/dist/js/bootstrap.m
            ]
        },
        "configurations": {
            "en": {
                "localize": [
                    "en-US"
                ]
            },
```

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 !

schema.json | angular.json | package.json | messages.es.json

angular.json > $schema

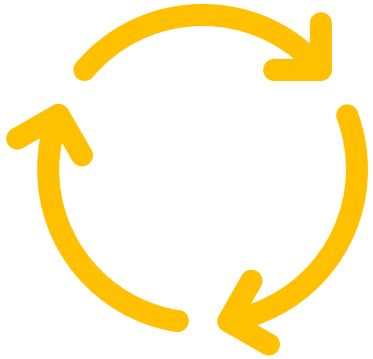
```
                        "sourceMap": true,
                        "namedChunks": true
                    }
                },
                "defaultConfiguration": "production"
            },
            "serve": {
                "builder": "@angular-devkit/build-angular:dev-server",
                "configurations": {
                    "production": {
                        "browserTarget": "webApp:build:production"
                    },
                    "development": {
                        "browserTarget": "webApp:build:development"
                    },
                    "page_en": {
                        "browserTarget": "webApp:build:en"
                    },
                    "page_es": {
                        "browserTarget": "webApp:build:es"
                    }
```

schema.json | angular.json | package.json | messages.es.json

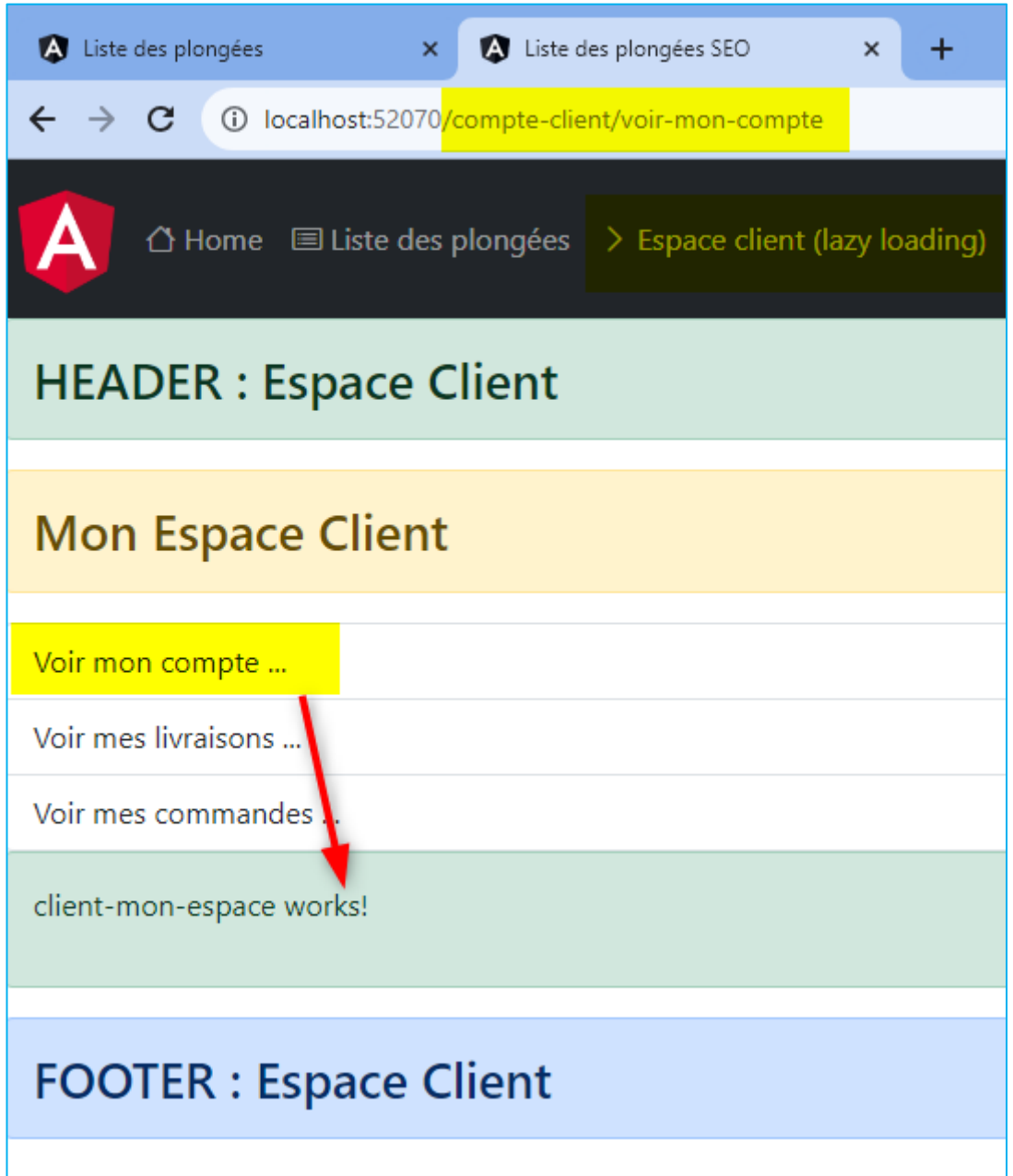
package.json > { } scripts

```
{
  "name": "web-app",
  "version": "1.0.0",
    Déboguer
  "scripts": {
    "ng": "ng",
    "start": "ng serve --open --port=3000",
    "start-en": "ng serve --open --port=3000 --configuration=page_en",
    "start-es": "ng serve --open --port=3000 --configuration=page_es",
    "build": "ng build",
    "build-localize": "ng build --localize",
    "watch": "ng build --watch --configuration development  --stats-json",
    "test": "ng test",
    "json-server": "json-server -p 3001 src/assets/json/films.json ",
    "json-server-post": "json-server -p 3002 src/assets/json/bdd.json",
    "analyze": "webpack-bundle-analyzer dist/stats.json"
  },
```

Routage Avancé
Lazy Loading
Optimisation du build

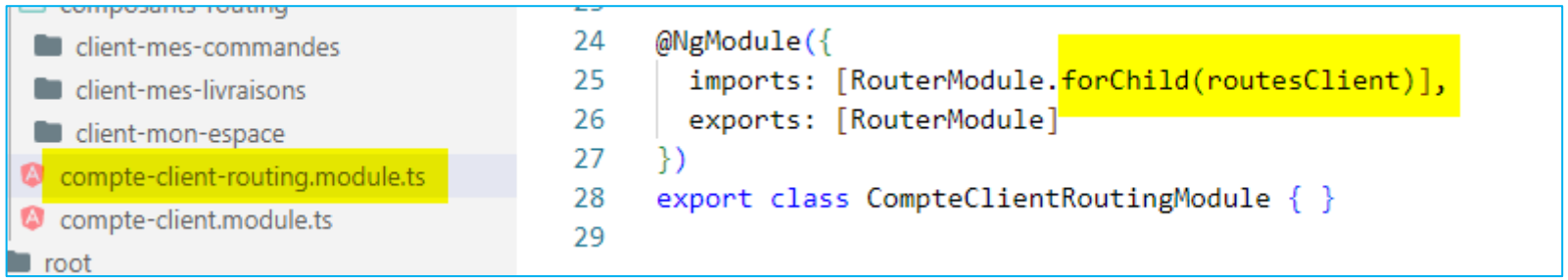


Routing : Concepts avancés – Le Lazy Loading



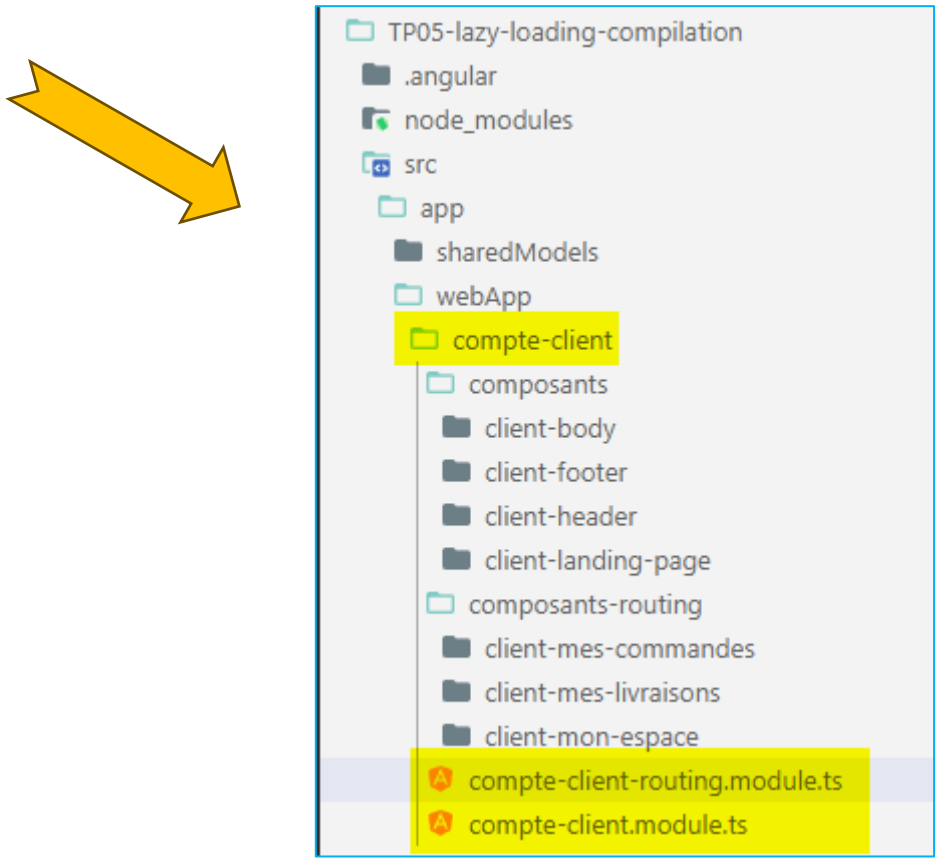
Routing : Concepts avancés – Le Lazy Loading

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



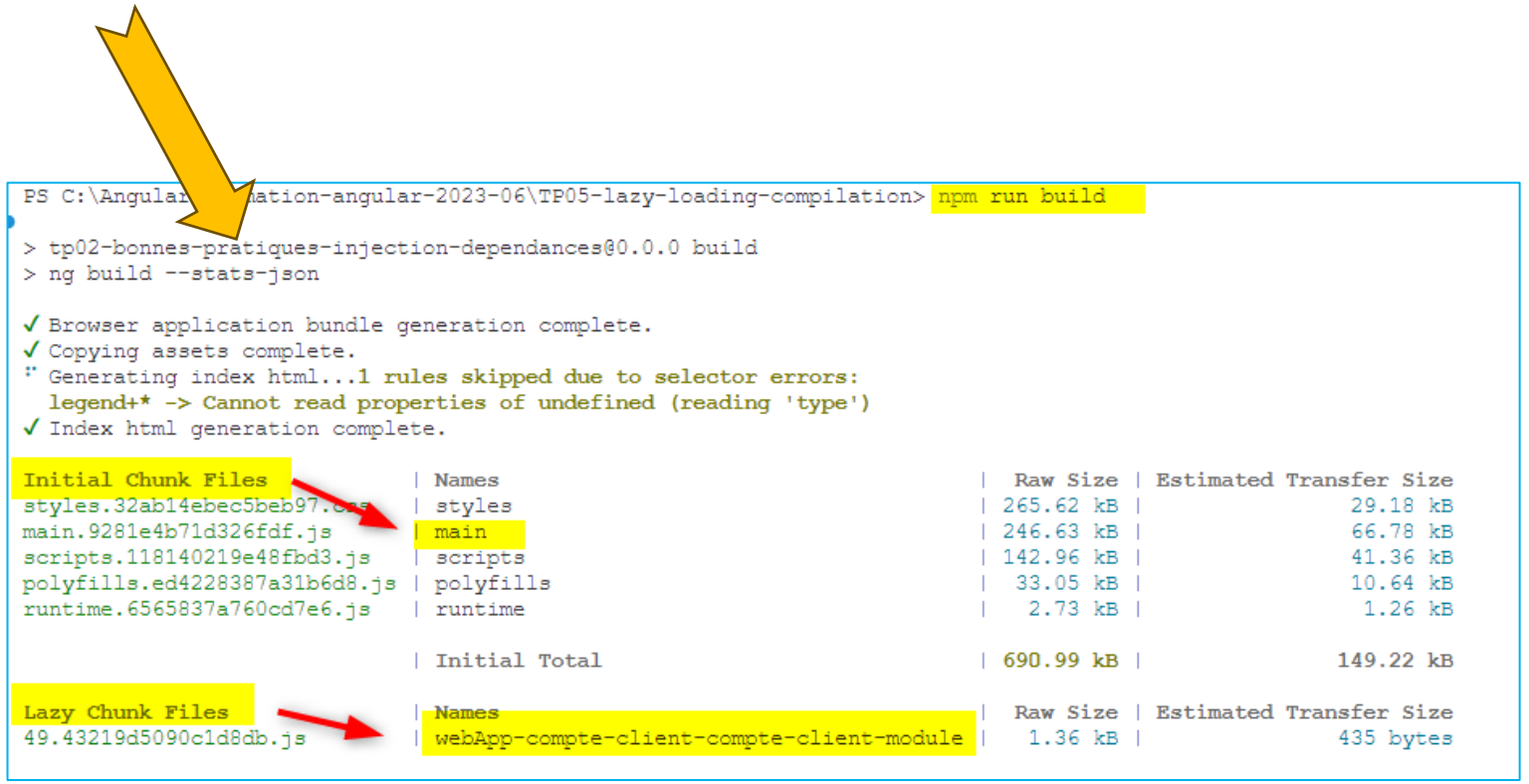
Routing : Concepts avancés – Le Lazy Loading

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



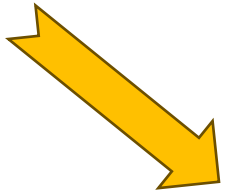
Routing : Concepts avancés – Le Lazy Loading

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.



Routing : Concepts avancés – Le Lazy Loading

Afin de bien analyser ces customisations « tuning » de compilation, mise en place d'une stratégie de lecture de statistiques avec le « webpack-bundle-analyzer »
https://www.npmjs.com/package/webpack-bundle-analyzer



package.json

P05-lazy-loading-compilation > package.json > ...

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

Le Lazy Loading : Mise en pratique
Etape #1 (no Lazy Loading)

client-landing-page.component.html

```
<div class="alert alert-primary">
    <app-client-header></app-client-he
    <hr>
    <app-client-body></app-client-body
    <hr>
    <app-client-footer></app-client-fo
</div>
```

client-body.component.html

```
<ul class="list-group">
    <li class="list-group-item">
        <a class="nav-link"
        [routerLink]="['mon-espace']"
        [routerLinkActive]="['lien-actif']"
        >Mon espace Client</a>
    </li>
    <li class="list-group-item">
        <a class="nav-link"
        routerLink="mes-commandes"
        [routerLinkActive]="['lien-actif']"
        >Mes commandes</a>
    </li>
    <li class="list-group-item">
        <a class="nav-link"
        [routerLink]="['mes-livraisons']"
        [routerLinkActive]="['lien-actif']"
        >Mes livraisons</a>
    </li>
</ul>

<p class="alert alert-success">
    <router-outlet></router-outlet>
</p>
```

accueil.module.ts

Formation-ANY-07-2023 > TP03-lazy-loading > src > app > webApp > root > accueil > accueil.module.t

```
// import { CompteClientModule } from '../../formation/compte-client/
@NgModule({
  declarations: [
    LandingPageComponent,
    HeaderComponent,
    BodyComponent,
    FooterComponent
  ],
  imports: [
    CommonModule,
    RouterModule,
    // nos modules
    FilmsModule,
    FormsReactiveModule,
    // CompteClientModule,
    RxjsModule
  ],
  exports:[
    LandingPageComponent,
    HeaderComponent,
    BodyComponent
```

compte-client-routing.module.ts

Formation-ANY-07-2023 > TP03-lazy-loading > src > app > webApp > formation > compte-client > compte-client-routing.module.ts > routesClient

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ClientMonEspaceComponent } from './composants-routing/client-mon-espace/client-mon-espace.c
import { ClientCommandesComponent } from './composants-routing/client-commandes/client-commandes.com
import { ClientLivraisonsComponent } from './composants-routing/client-livraisons/client-livraisons.

export const routesClient: Routes = [
// const routesClient: Routes = [

  { path: 'mon-espace', component: ClientMonEspaceComponent },
  { path: 'mes-commandes', component: ClientCommandesComponent },
  { path: 'mes-livraisons', component: ClientLivraisonsComponent },
];
```

app-routing.module.ts

Formation-ANY-07-2023 > TP03-lazy-loading > src > app > app-routing.module.ts > routes

```
import { ClientLandingPageComponent } from './webApp/formation/compte
// ------------------------------------------------
import { routesClient}  from  "./webApp/formation/compte-client/comp

const routes: Routes = [
  { path: '', component: BodyComponent },
  { path: 'accueil', component: BodyComponent },
  { path: 'liste-des-films', component: ListeDesFilmsComponent },

  // 1ère étape : sans lazy loading
  // raccrocher un module qui a son propre système de routage
  {
    path: 'compte-client',
    component: ClientLandingPageComponent,
    children: routesClient
  },
```

Le Lazy Loading : Mise en pratique
Etape #2 (Lazy Loading)

client-landing-page.component.html

```
<div class="alert alert-primary">
    <app-client-header></app-client-he
    <hr>
    <app-client-body></app-client-body
    <hr>
    <app-client-footer></app-client-fo
</div>
```

client-body.component.html

```
<ul class="list-group">
    <li class="list-group-item">
        <a class="nav-link"
        [routerLink]="['mon-espace']"
        [routerLinkActive]="['lien-actif']"
        >Mon espace Client</a>
    </li>
    <li class="list-group-item">
        <a class="nav-link"
        routerLink="mes-commandes"
        [routerLinkActive]="['lien-actif']"
        >Mes commandes</a>
    </li>
    <li class="list-group-item">
        <a class="nav-link"
        [routerLink]="['mes-livraisons']"
        [routerLinkActive]="['lien-actif']"
        >Mes livraisons</a>
    </li>
</ul>

<p class="alert alert-success">
    <router-outlet></router-outlet>
</p>
```

accueil.module.ts

```
// import { CompteClientModule } from '../../formation/compte-client/compte-client.

@NgModule({
  declarations: [
    LandingPageComponent,
    HeaderComponent,
    BodyComponent,
    FooterComponent
  ],
  imports: [
    CommonModule,
    RouterModule,
    // nos modules
    FilmsModule,
    FormsReactiveModule,
    // CompteClientModule,
    RxjsModule
```

compte-client-routing.module.ts

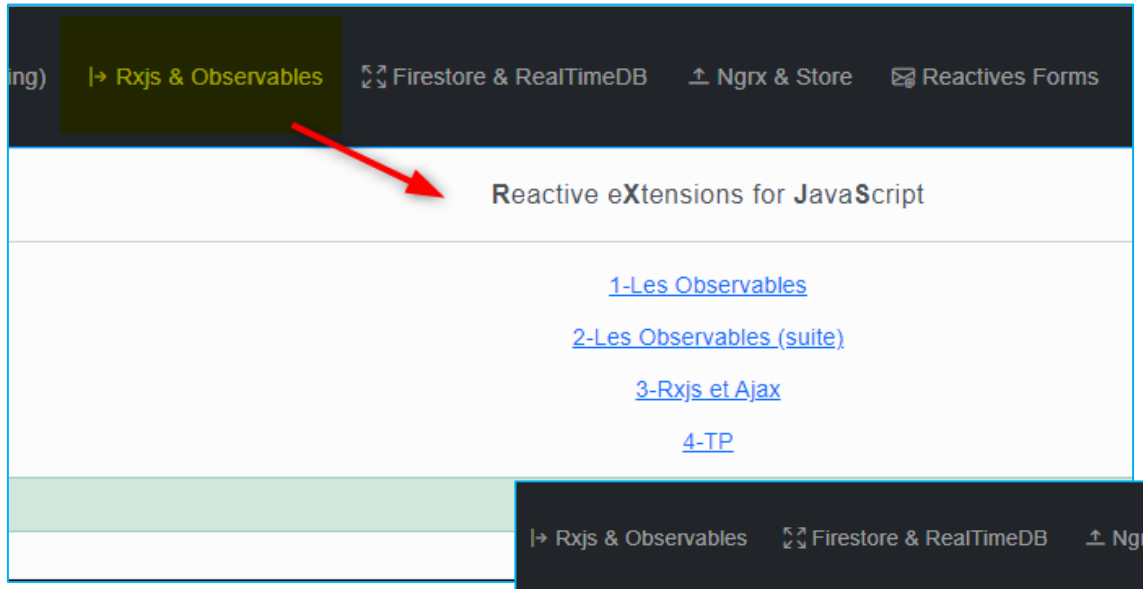
```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ClientMonEspaceComponent } from './composants-routing/client-mon-espace/client-mon-espace.c
import { ClientCommandesComponent } from './composants-routing/client-commandes/client-commandes.com
import { ClientLivraisonsComponent } from './composants-routing/client-livraisons/client-livraisons.

// export const routesClient: Routes = [
const routesClient: Routes = [

  { path: 'mon-espace', component: ClientMonEspaceComponent },
  { path: 'mes-commandes', component: ClientCommandesComponent },
  { path: 'mes-livraisons', component: ClientLivraisonsComponent },
];

@NgModule({
  imports: [RouterModule.forChild(routesClient)],
  exports: [RouterModule]
```

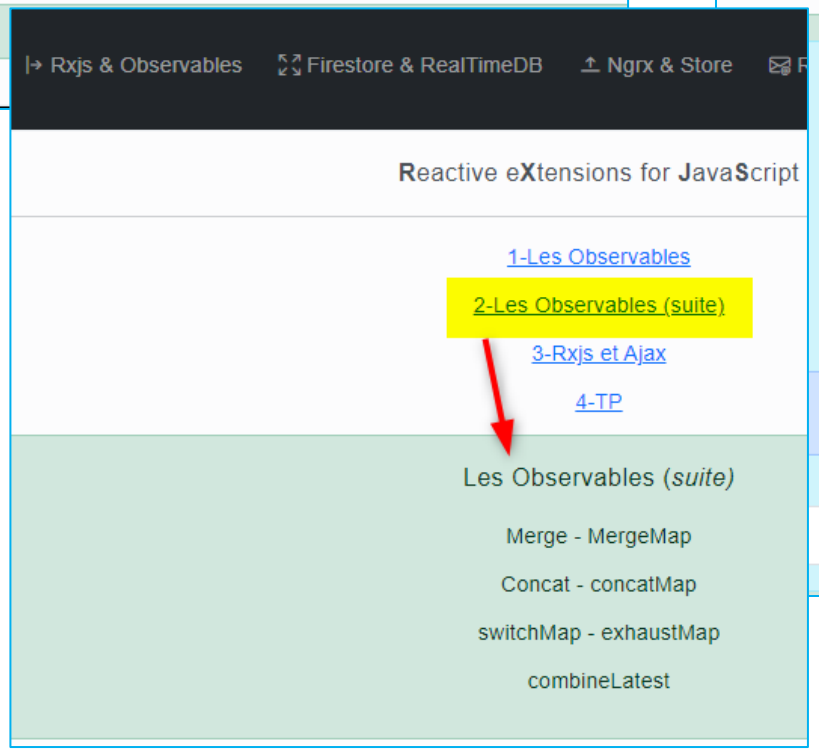
app-routing.module.ts

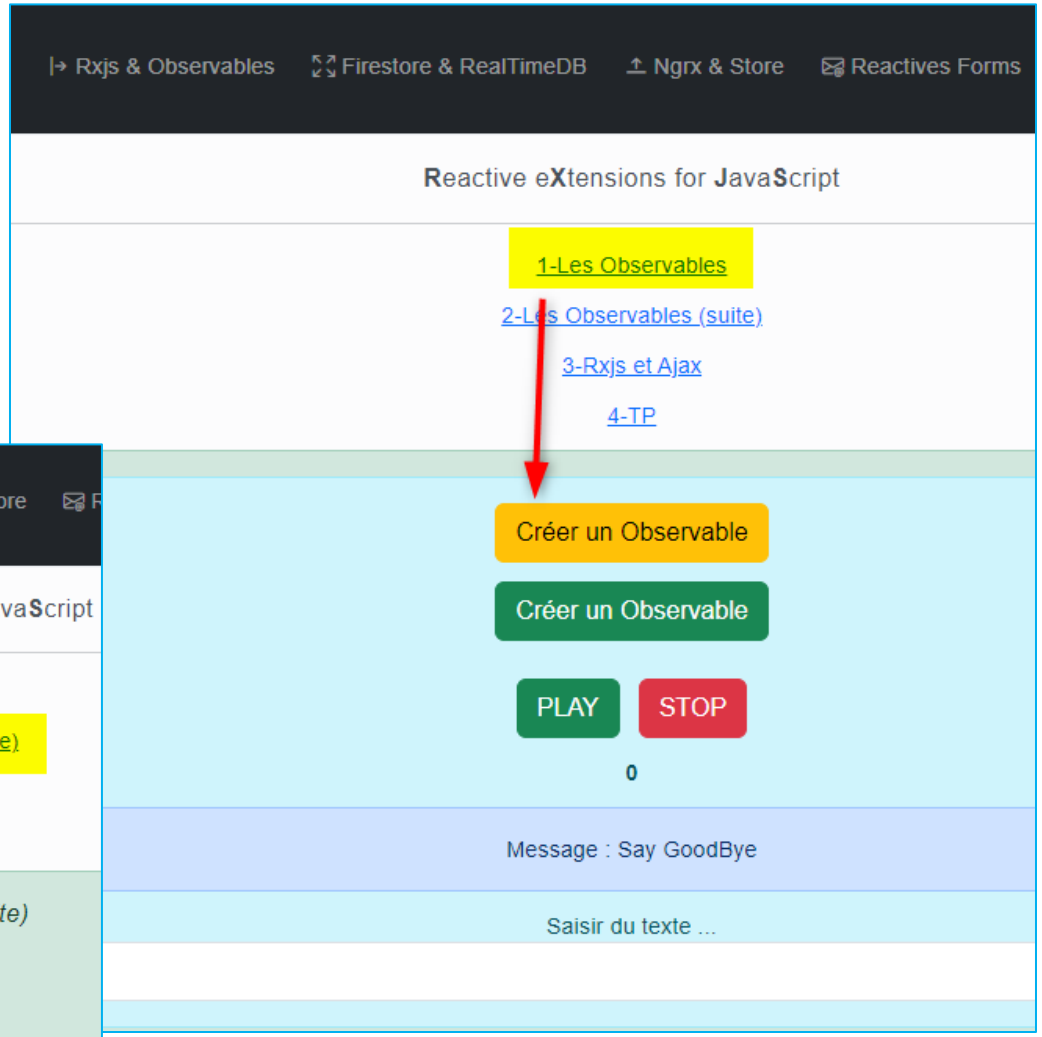
```
// 2ème étape : Avec lazy loading (promesses ES2015 😐)
// {
//   path: 'compte-client',
//   component: ClientLandingPageComponent,
//   loadChildren:
//   () => import('./webApp/formation/compte-client/compte-client.module')
//   .then(
//     (m) => {
//       return m.CompteClientModule
//     }
//   )
// },

// 3ème étape : Avec Lazy Loading (ES2017 : Async/await => asynchrone 😉)
{
  path: 'compte-client',
  component: ClientLandingPageComponent,
  loadChildren:
  async () => (await import('./webApp/formation/compte-client/compte-client.module'
  // syntaxic sugar
    data :{
      preload:true
    }

},
```

TP : Déployer une arborescence en Lazy Loading

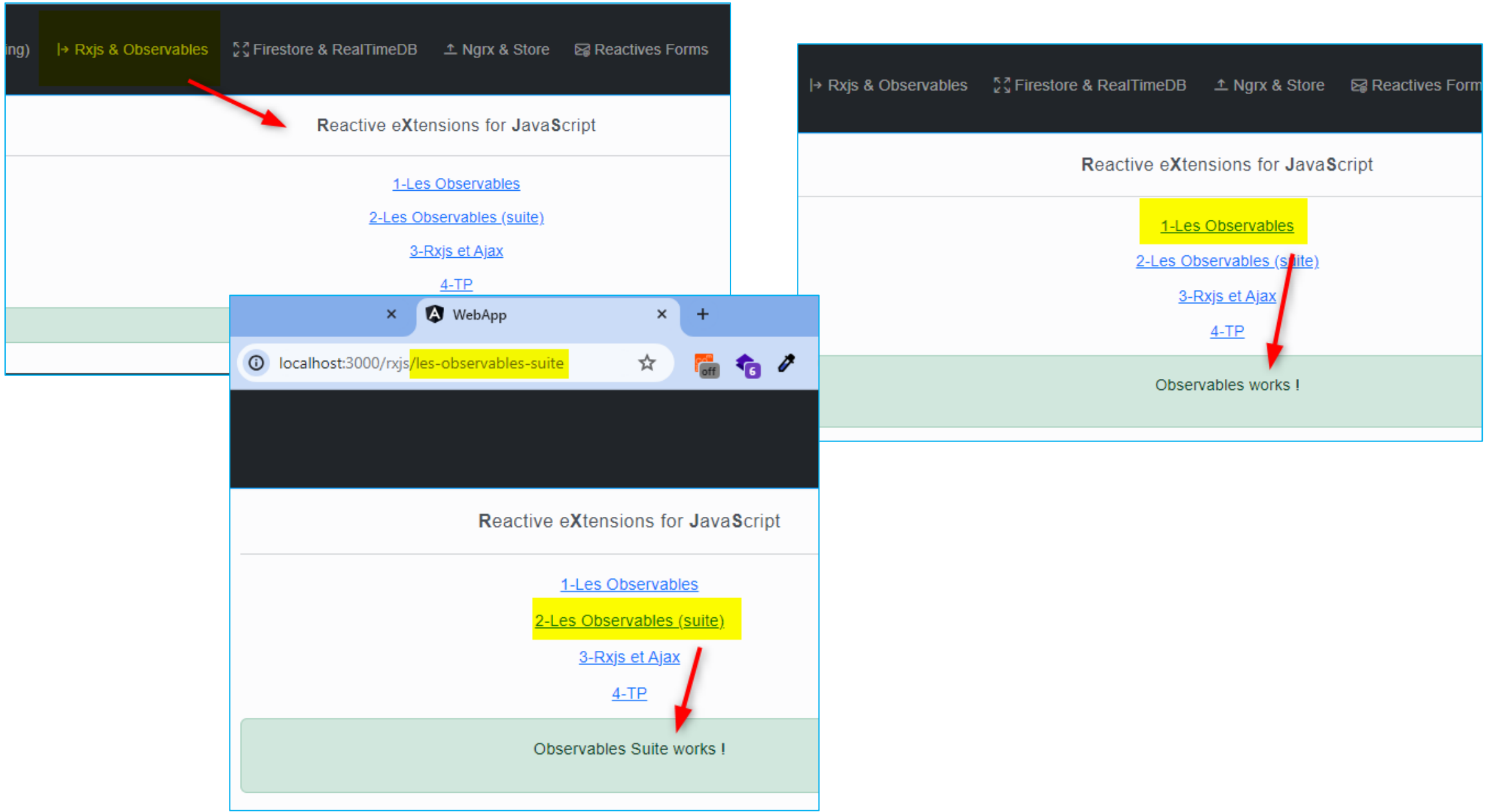
TP : déployer l'arborescence en Lazy Loading RXJS



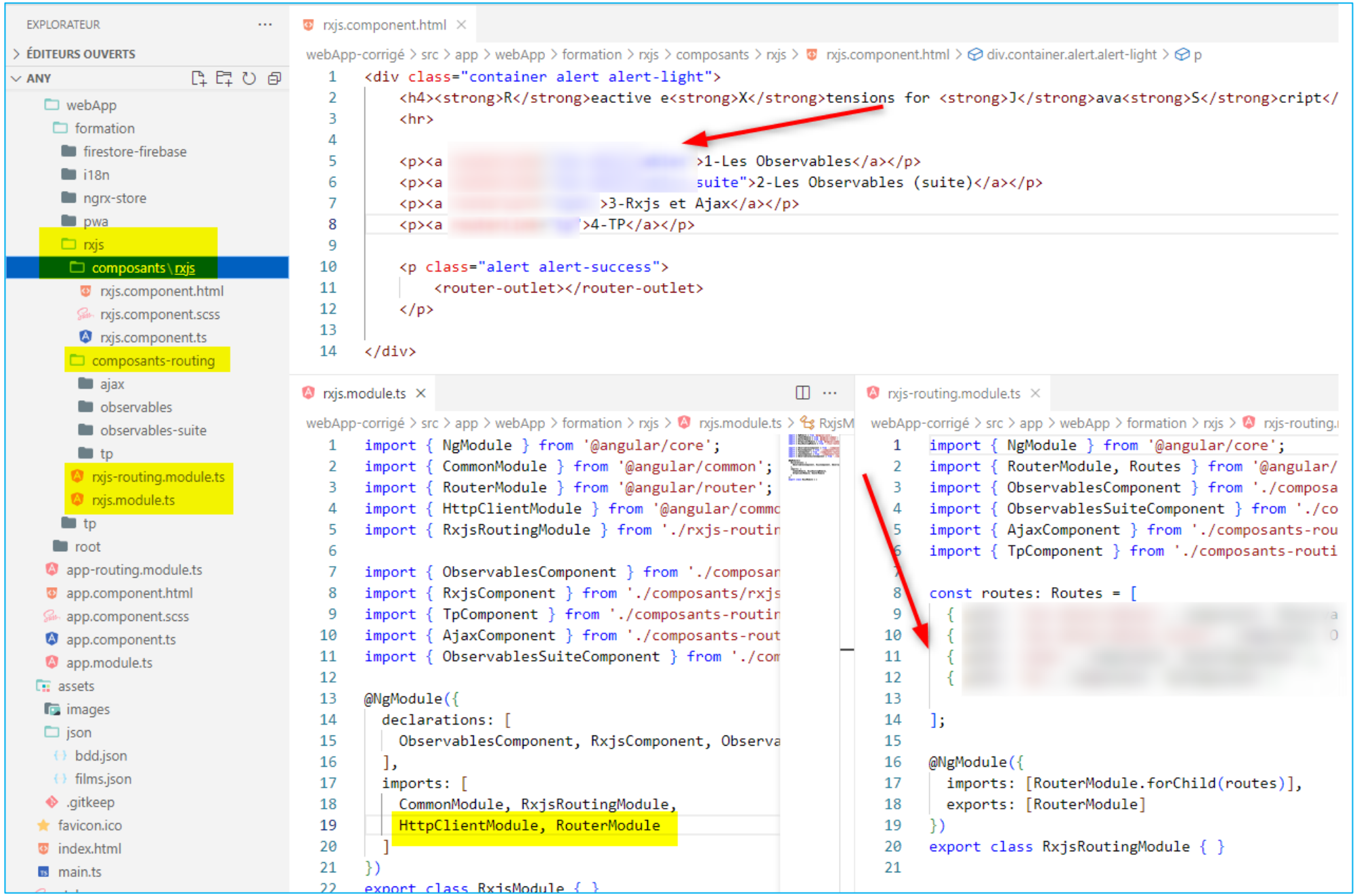




TP : déployer l'arborescence en Lazy Loading RXJS



TP : déployer l'arborescence en Lazy Loading RXJS



TP : Tester le mode lazy Loading en compilation

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