



Angular 2+

Développer des applications Web

Support de cours

Réf. T44A-046





Module 3

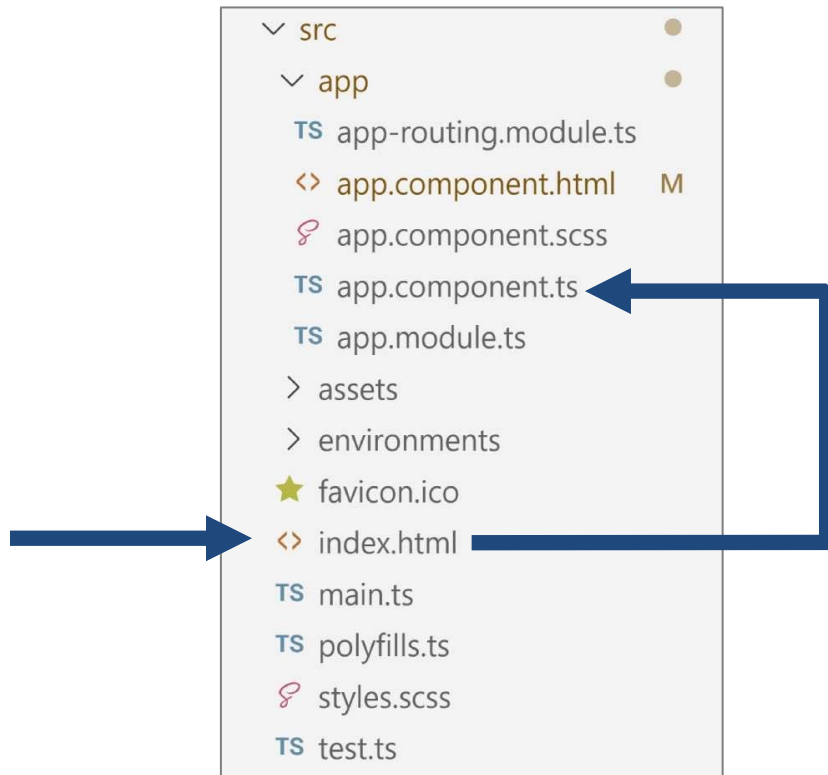
Les 3 notions de base

Contenu du module

- Présentation du composant principale (root)
- Notion 1 : L'Interpolation
- Notion 2 : La liaison par évènement
- Notion 3 : La liaison bidirectionnelle
- Builder son projet pour une mettre ligne

Qui appelle le composant principal : root ?

- Le point d'entrée de l'application est le fichier **index.html**
- Le fichier **index.html** appelle le composant principal **app.component.ts** sous la forme d'une balise



Le composant principale : Qui appelle le composant root ?

- Le point d'entrée de l'application est le fichier **index.html**
- Le fichier **index.html** appelle le composant principale **app.component.ts** sous forme de balise

```
<body>
  <app-root></app-root>
</body>
</html>
```

index.html

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {
  title = 'my-test';
}
```

app.component.ts

Le composant principale : Présentation

- Les décorateurs permettent de faire la liaison entre les **3 fichiers** du composant
- Et de définir le nom "de balise" pour l'appel du composant

Les décorateurs



```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {
  title = 'my-test';
}
```

app.component.ts

Notion 1 : L'interpolation

- L'interpolation ou "Double moustaches"
- Permet d'afficher des attributs ou ce que retourne une méthode TypeScript



app.component.ts

```
export class AppComponent {  
  fruit = 'pomme';  
  disBonjour(){  
    return 'Bonjour !';  
  }  
}
```

app.component.html

```
<h1>{{ fruit }}</h1>  
<h2>{{ disBonjour() }}</h2>
```

localhost:4200

pomme
Bonjour !

Notion 2 : La liaison par évènement

- La liaison par évènement **event binding**
- Permet d'appeler une méthode TypeScript
- Bonne pratique : préfixer avec **on** : *par exemple onCalculer()*


app.component.ts

```
export class AppComponent {  
  fruit = '';  
  onAffiche(){  
    this.fruit = 'kiwi';  
  }  
}
```


app.component.html

```
<button (click)="onAffiche()">GO</button>  
  
<h1>{{ fruit }}</h1>
```


Notion 3 : La liaison bidirectionnelle

- La liaison bidirectionnelle **2 ways binding** 
- Permet de relier un champ tel que input à un attribut
- Les échanges se font en temps réels
- Attention** : il faut déclarer **FormsModule** dans le fichier **app.module.ts**
- Moyen mémo-technique **Banana in the box**

app.module.ts



```
imports: [  
  FormsModule,  
  BrowserModule,  
  AppRoutingModule  
],
```

app.component.ts

```
export class AppComponent {  
  nom = '';  
  resultat = '';  
  onMettreEnMajuscule(){  
    this.resultat = this.nom.toUpperCase();  
    this.nom = ''; // vider le champ input  
  }  
}
```

app.component.html

```
<input [(ngModel)]="nom">  
  
<button (click)="onMettreEnMajuscule()">  
  GO  
</button>  
  
<h1>{{ resultat }}</h1>
```

Builder son projet pour le mettre en ligne

- Si l'application se trouve à la racine du site

```
ng build
```

- Si l'application se trouve dans un sous répertoire (ex : www.mon-site.fr/mon-projet)

```
ng build --base-href=/mon-projet
```

- Pour récupérer les fichiers buildés, prenez le contenu du répertoire **dist/premier-prj/**
- Et copier le contenu dans le répertoire mon-projet ou à la racine de votre hosting