ACADÉMIE ANGULAR

Classe Virtuelle | Avril 2020

Laurent Duveau

Qui suis-je ?

Laurent Duveau

- Montréal!
- Fondateur de l'Académie Angular
- Formation en classe de 2 jours
- **189 workshops** en 4 ans 

- Montréal, Boston, Québec, Toronto, Vancouver, Ottawa, Calgary, Winnipeg, Rimouski, London, Copenhague, Helsinki, Punta Cana…
- @AngularAcademy

Logistique

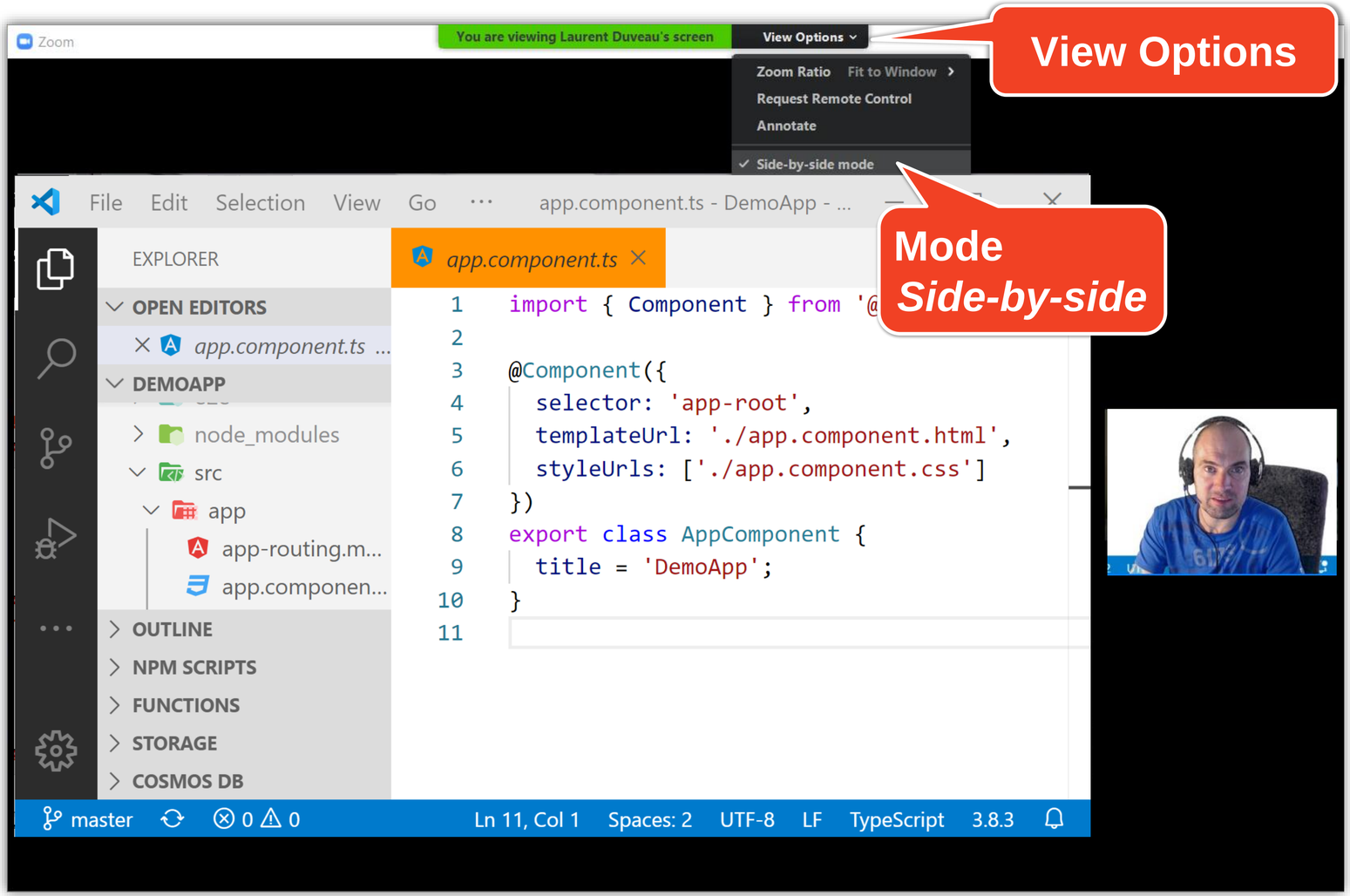
- 2 jours
- 8:30 - 16:30
- Pause Lunch: 1 heure
- Pauses de 15min le matin et l’après midi

Utiliser Zoom

Partage des slides

Partage des slides

Partage du code



Cette formation ne sera **PAS** enregistrée.

Perdu dans les coding labs?

- Si vous avez *déjà* un repo Git local
  - Sinon, le créer avec: `> git init`

Dans le dossier *DemoApp*!

- Connecter votre projet local avec le repo git de l'instructeur:

```
> git remote add origin https://github.com/ldex/Angular-Academy-190.git
```

Perdu dans les coding labs?

- Chaque fois que vous souhaitez obtenir le code du formateur (attendre après son *commit*):

```
> git reset --hard
> git pull origin master
```

Dans le dossier *DemoApp*!

(Cela remplacera tout code que vous avez fait par le code du formateur)

Perdu dans les coding labs?

- Si vous ne voulez pas perdre votre code, vous pouvez créer un dossier **séparé** à la place :

```
> git clone https://github.com/ldex/Angular-Academy-190.git DemoAppFromGit
```

- Ensuite, chaque fois que vous souhaitez synchroniser ce dossier avec le code du formateur :

```
> git reset --hard
> git pull origin master
```

Dans le dossier *DemoAppFromGit*!

Pré-requis

Ce que vous êtes supposé savoir…

- HTML
- CSS
- JavaScript

Attentes

- Vous ne serez pas nécessairement un expert en 2 jours!
- Comprendre tous les concepts de Angular et TypeScript
- **Apprentissage par la pratique!** Bâtir une application complète, en partant de zéro
- Nous allons aussi voir le setup, les outils et les meilleures pratiques

Agenda

- TypeScript
- Setup et outillage
- Introduction à Angular
- Composants
- Syntaxe des templates
- Services et Injection de dependence
- Intro à la Programmation Réactive avec RxJS (Observables)
- Requêtes asynchrones à une API REST (HTTP)
- Implémenter un cache local
- Pagination dans une liste
- Débogger une app
- Modules
- Déploiement
- Navigation et Routeur
- Lazy loading
- Formulaires et validation
- Authentification et sécurité
- Meilleures Pratiques

Atelier!

Slides…

…Code!

Slides…

…Code!

Slides…

…Code!

Questions?

Questions?

Questions?

Questions ?
Feedback ?

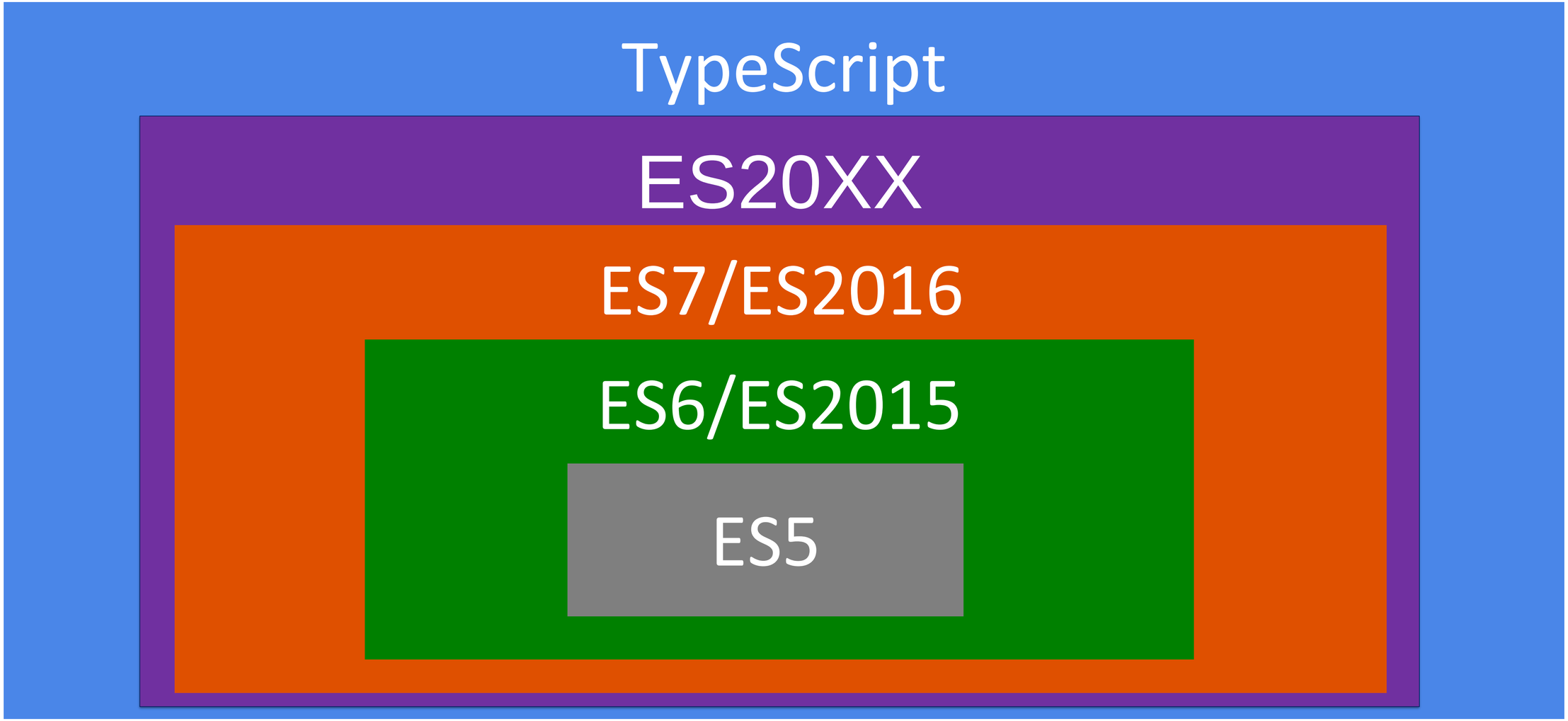
Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

TypeScript

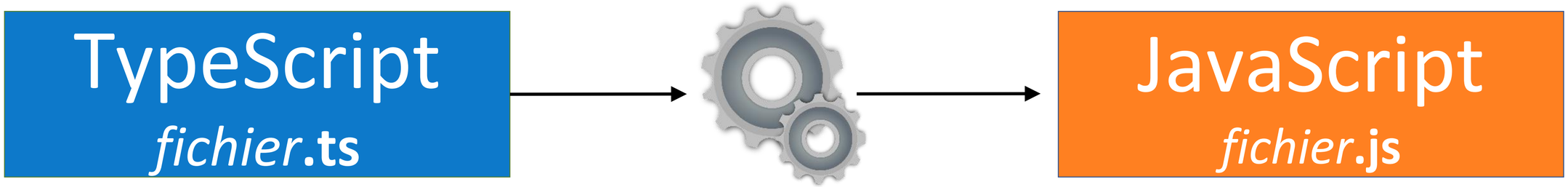
Qu'est-ce que TypeScript?

- TypeScript est un surensemble **typé** de JavaScript qui **compile** vers du JavaScript
- Par Microsoft, **Open Source**
- C’est comme coder en JavaScript, mais sans la douleur…

TypeScript est un surensemble de JavaScript



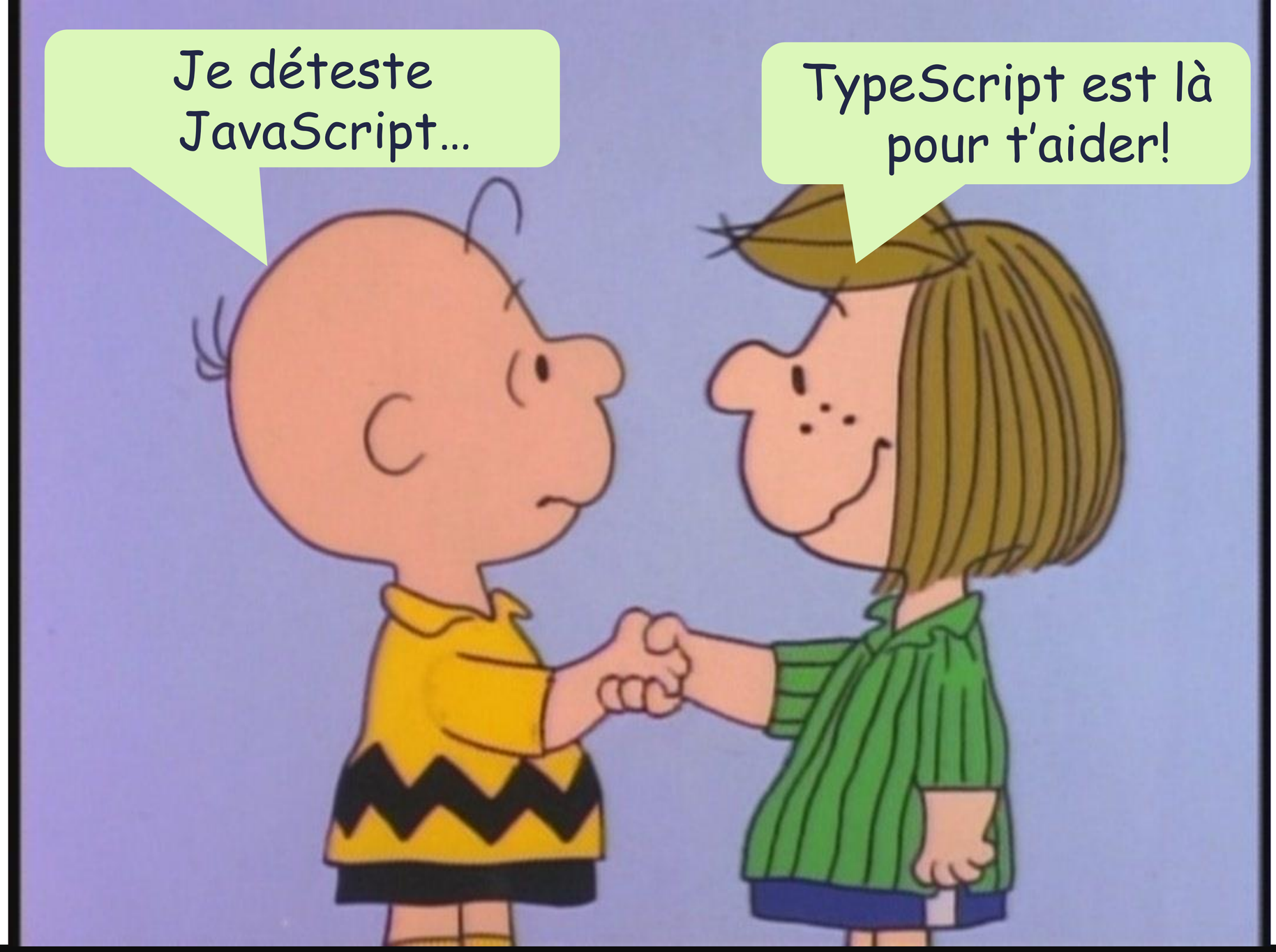
Comment ça marche?



Compilateur TypeScript

“Transpiler”

Génère du code ES5/ES6/…


Je déteste JavaScript…
TypeScript est là pour t'aider!

Annotations de type

```
let price: number = 60;
let isDone: boolean = true;
let name: string = "Angular Academy";
let list1: number[] = [1, 2, 3];
let list2: Array<number> = [17, 99, 42];
let sortedList = list2.sort((n1,n2) => n1 - n2);

function add(x: number, y: number): number {
    return x + y;
}
let res = add(18, "5");
```

Génériques!

Erreur TypeScript
dans l'IDE!

Classe, Interface, …

```
class Game {

  constructor(private user: Person) {
  }

  get User(): Person {
    return this.user;
  }

  set User(val: Person) {
    this.user = val;
  }

  start() {
    console.log(this.user.sayHi());
  }
}
```

constructeur

promotion du user comme membre local

propriété get/set

méthode

```
interface Person {
    name : string;
    age?: number;
    sayHi:() => string;
}
```

? optionnel

Classe, Interface, …

```
class Game {
  user: Person;

  constructor(_user: Person) {
    this.user = _user;
  }

  get User(): Person {
    return this.user;
  }

  set User(val: Person) {
    this.user = val;
  }

  start() {
    console.log(this.user.sayHi());
  }
```

```
interface Person {
    name : string;
    age?: number;
    sayHi:() => string;
}
```

DÉMONSTRATION / ATELIER

“Angular technically doesn't require TypeScript kind of like technically a car doesn't require brakes.” *– Joe Eames*

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Bien débuter

(Setup, configuration et outillage)

Fichiers partagés

- Labs/ ← Pour les exercices!
- angular.pdf ← Cette présentation!
- setup-fr.pdf ← Procédure de setup

http://angular.ac/fichiers

Récupérez les fichiers!

Angular CLI! (*Command Line Interface*)

- Mise en route rapide!

```
> npm install -g @angular/cli
> ng new DemoApp --routing
> cd DemoApp
> ng serve
```

Paramètres

+ génération de composants, services, routes, …

https://angular.io/cli

“La CLI Angular facilite la **création et gestion** d'une application qui suit les **meilleures pratiques**, *right out of the box*” *– Moi*

Dans un terminal ou command prompt:

```
> ng v
```

Dans le dossier de votre projet:

```
> ng serve -o
```

DÉMONSTRATION / ATELIER

IDEs et éditeurs de code

- Sublime Text
- Atom
- Brackets
- WebStorm
- Visual Studio
- …

Visual Studio Code

- **VS Code**… éditeur de code populaire!
- Gratuit, Open Source
- Versions Windows, Mac et Linux!
- HTML5, JavaScript, CSS, LESS avec NodeJs ou ASP.NET
- Éditeur de code riche dans un outil léger et très rapide
- Débogage, déploiement
- Intégration Git
- Nombreuses extensions

Je vous encourage à me suivre avec VS Code!

code.visualstudio.com

DÉMONSTRATION / ATELIER

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

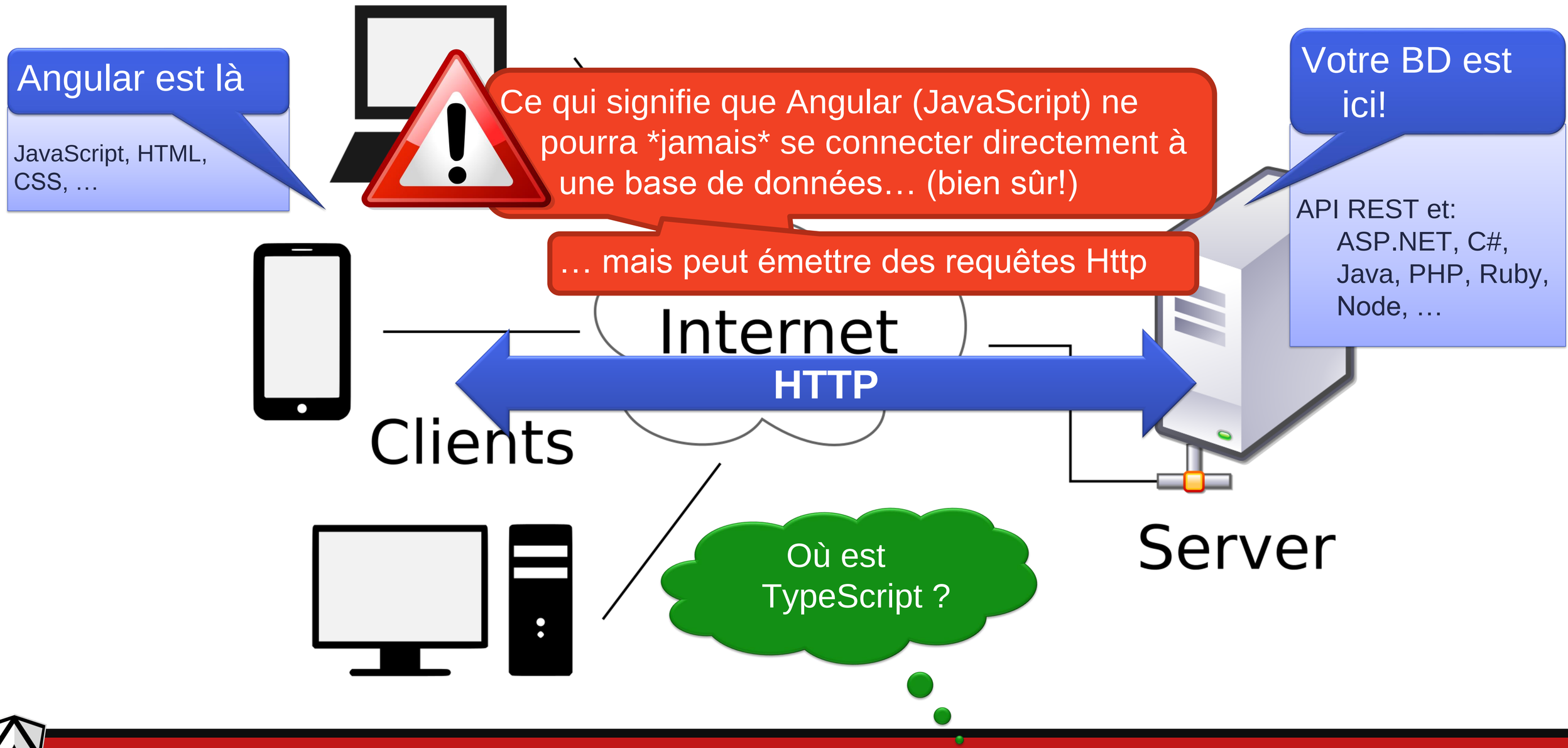
Introduction à Angular

Angular ?

- Framework JavaScript particulièrement adapté pour les applications web modernes monopage (*Single Page Application*, ou SPA)
- Compatible avec IE 9+ et autres navigateurs modernes
- Open Source, licence MIT
- v9 Février 2020

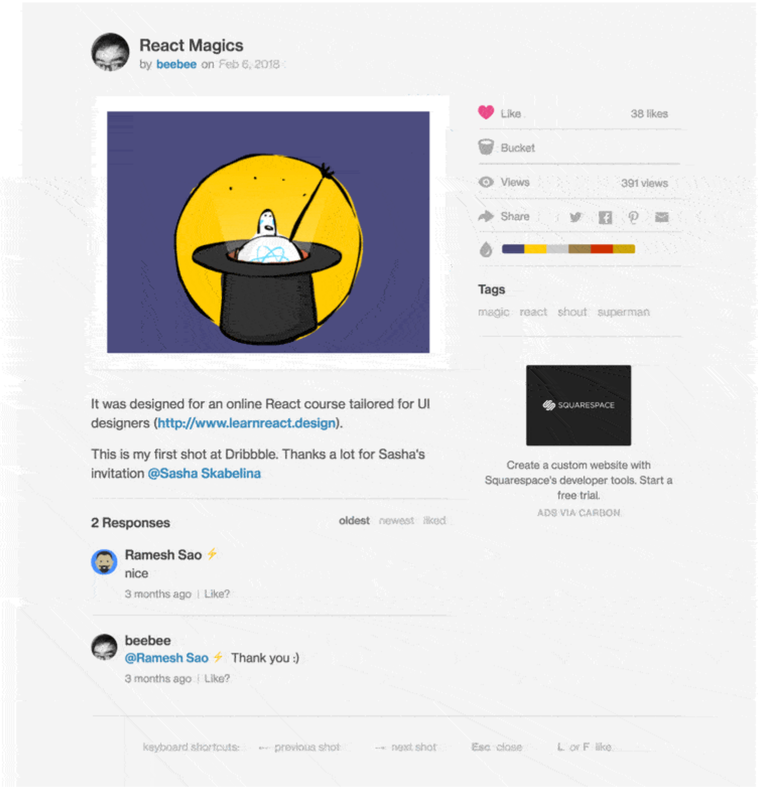
www.angular.io

Angular est l'app front end pour n'importe quel backend



Composants

Bâtir une app avec des composants

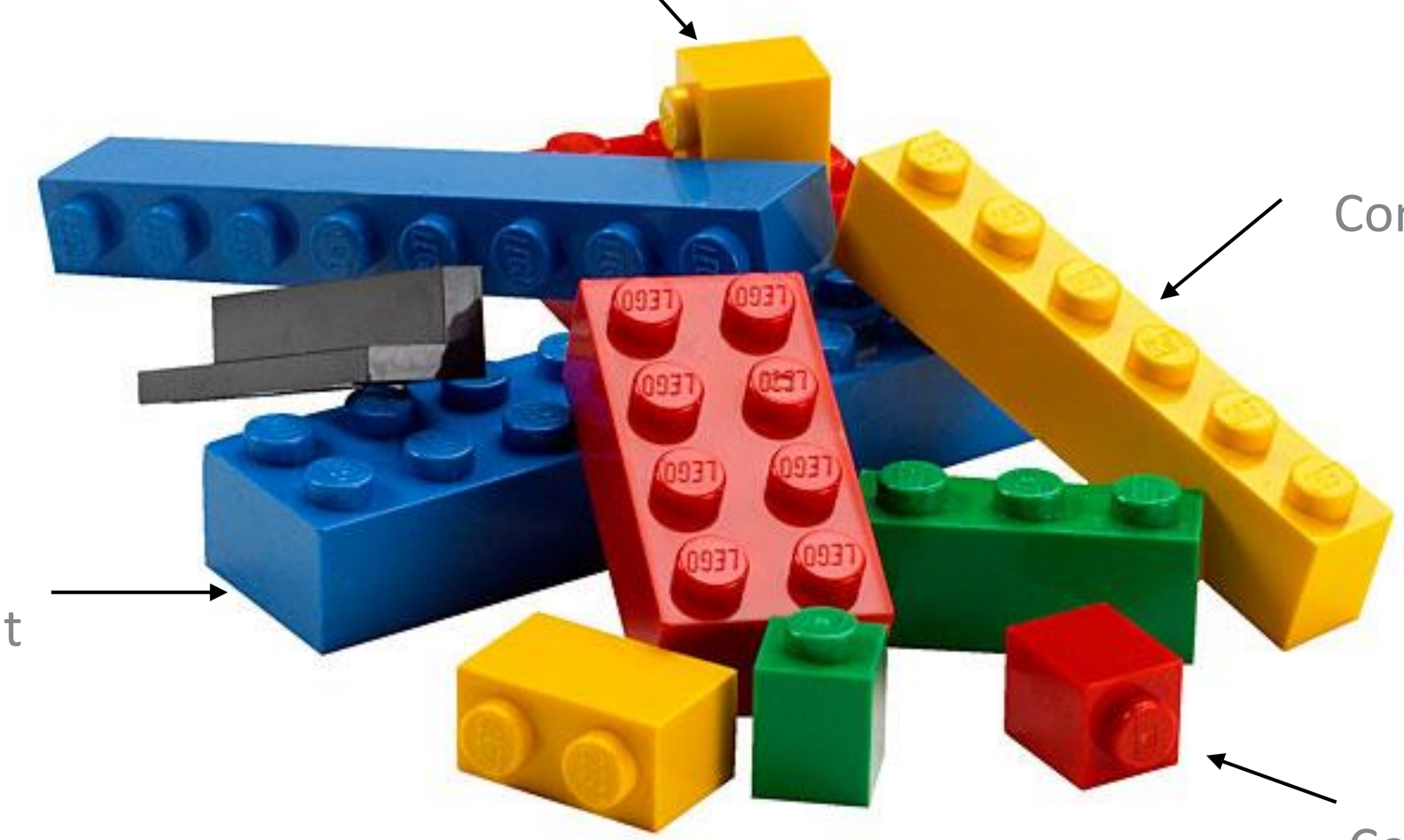


Items List
Component

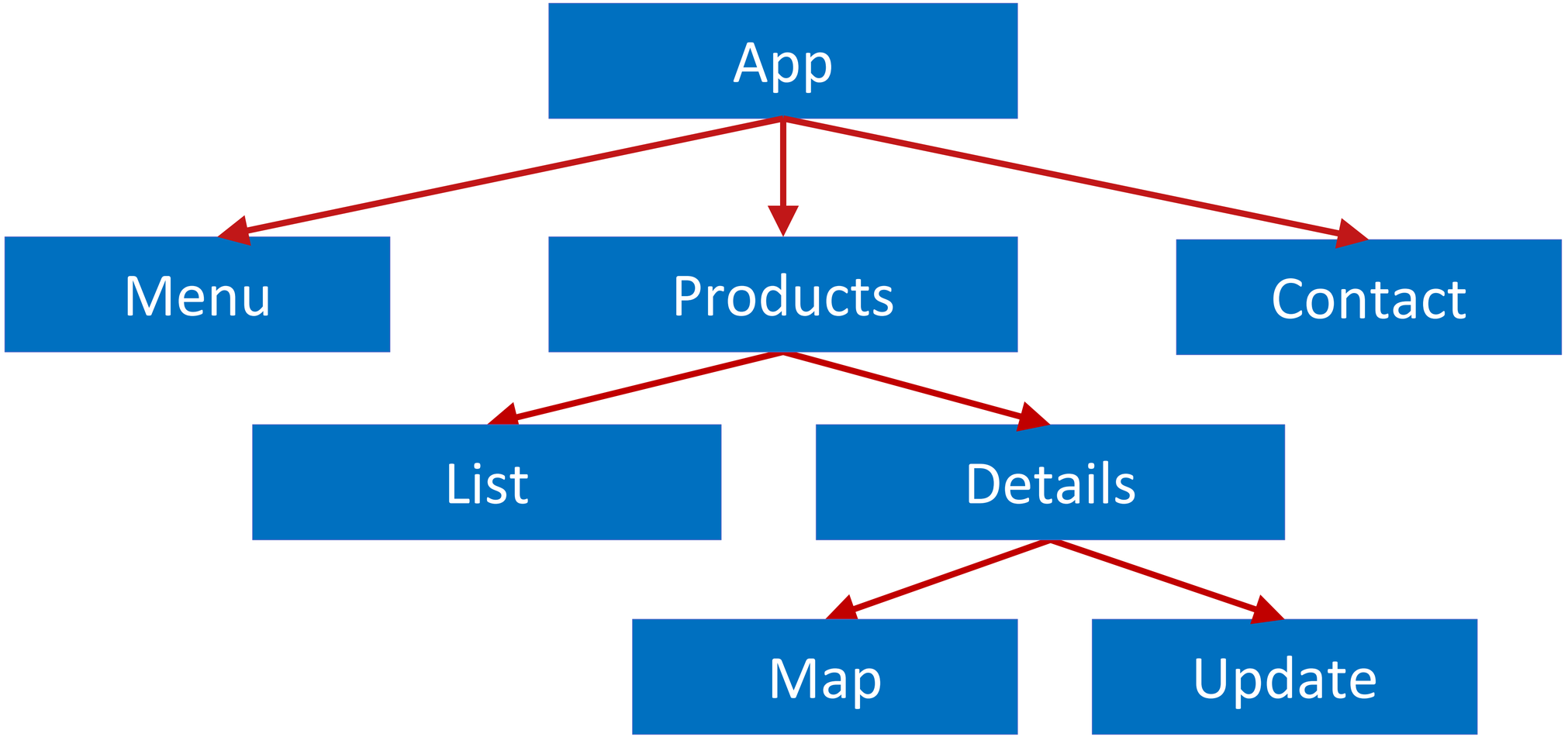
Menu
Component

Grid
Component

Header
Component



Votre app est un arbre de composants



Qu'est-ce qu'un composant?

- Un composant est un objet réutilisable

- Fait de:

Code (classe)

TypeScript!

HTML Template

- Offre un "sélecteur": **<product-list></product-list>**

place une instance du composant dans le DOM

La classe d'un composant

imports

décorateur

classe

```
import { Component } from '@angular/core';
import { DataService } from './data.service';


@Component({
  selector: 'product-detail',
  templateUrl: 'product-detail.component.html'
})

export class ProductDetailComponent {

}
```

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Syntaxe déclarative des templates

Composants et Templates

- Composants s’appuient sur des **décorateurs** pour définir des metadatas, dont le chemin vers le template HTML

```
@Component({
  selector: 'customer-info',
  templateUrl: 'customers.component.html',
})
export class CustomersComponent {
  constructor() { }
}
```

Interpolation

{{ expression }}

```
<div>
  {{ product.name }}
</div>
```

Binding de propriétés

[propriété]

```
<img [src]="imageUrl" />
<div [hi                ]="isHidden"></div>
```

Binding sur toute propriété du DOM

Aussi pour classes css!

```
<div [class.selected]="isSelected"></div>
```

Gestion des évènements

(event)

```
<button (click)="sort('price')">Tri</button>

<select (change)="setProperty()">
```

Binding bi-directionnel

[(ngModel)]

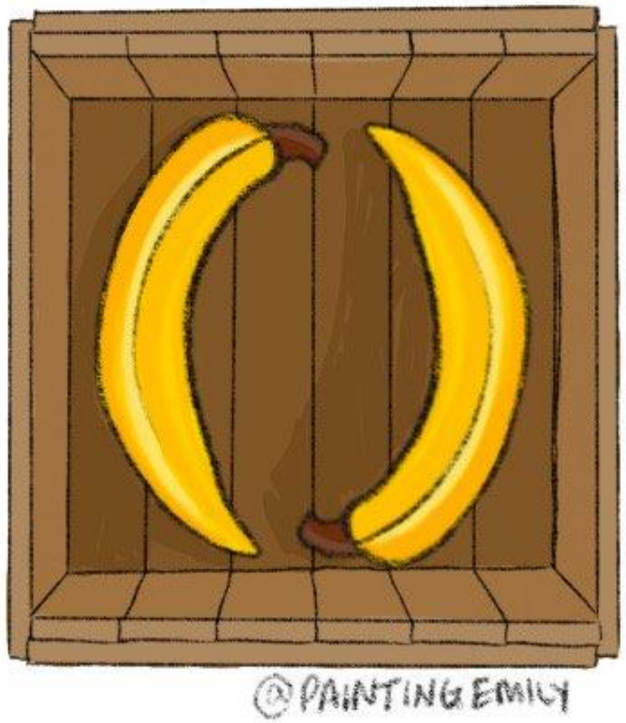
Directive built-in pour gérer la valeur de l'input et l'événement

```
<input type="text" [(ngModel)]="prod.prix" />
```

Détection des changements et mise à jour du modèle

Binding bi-directionnel

[(ngModel**)]** = “bananes dans une boîte”



Directives *built-in*

- Angular offre quelques **directives** comme ***ngFor** et ***ngIf**
- Manipulation des éléments auxquels elles sont attachées

* = *Structural directive:* qui contrôle le DOM

```
<table *ngIf="canViewProducts">
  <tr *ngFor="let product of products">
     <td>{{ product.name }}</td>
     <td>{{ product.price }}</td>
  </tr>
```

Pipes *built-in*

- Utilisation de *Pipes* pour **formater des données** dans les templates
- Plusieurs pipes built-in
  - uppercase, lowercase, slice, date, currency, async, json

```
<td>{{ customer.orderTotal | currency:'CAD':'symbol':'2.1-3' }}</td>
```

Formater comme monnaie

DÉMONSTRATION / ATELIER

Récapitulatif de syntaxe

- Interpolation: **{{** expression **}}**
- Binding de propriété: **[** attribut ou classe **]**
- Binding d'événement: **(** event **)**
- Binding bi-directionnel: **[(** ngModel **)]**
- ***ngFor**
- ***ngIf**

https://angular.io/guide/template-syntax

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Services et Injection de Dépendance

Services ?

- Les services sont simplement des classes offrant des fonctionnalités réutilisables (règles d'affaire, calculs, appels Ajax, etc.)
- Implémentés comme des singletons *par injecteur*
- Indépendant des vues et templates
- Un service peut utiliser d'autres services par **injection de dépendance**
- Un service peut être utilisé par tous les composants d'un module par **injection de dépendance**

Créer un Service

product.service.ts

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


@Injectable({
   providedIn: 'root'
})
export class ProductService {
    constructor() { }
    getProducts() {
       …
     }
}
```

Une instance du service sera stockée dans l'injecteur racine

Créer un Service

product.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Injectable({
   providedIn: 'root'
})
export class ProductService {
    constructor(private http: HttpClient) { }
    getProducts() {
      // return this.http…
     }
}
```

ProductService utilise lui même un autre service (HttpClient) injecté dans son construteur

Utiliser un Service par injection

product-list.component.ts

```
@Component({
    selector: 'product-list',
    templateUrl: 'product-list.component.html'
})
export class ProductListComponent {
    products: Product[];
    constructor(private productService: ProductService) {
      this.products = productService.getProducts();
    }
}
```

ProductService est injecté au runtime

DÉMONSTRATION / ATELIER

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Programmation Réactive avec RxJS

Programmation réactive?

- La **programmation réactive** est la programmation avec un flux de données asynchrone (collection d'événements).

- Exemple non-réactif

```
a = 1;
b = a + 1;

a = 2;

print a; ---> 2
print b; ---> 2
```

- Exemple Réactif

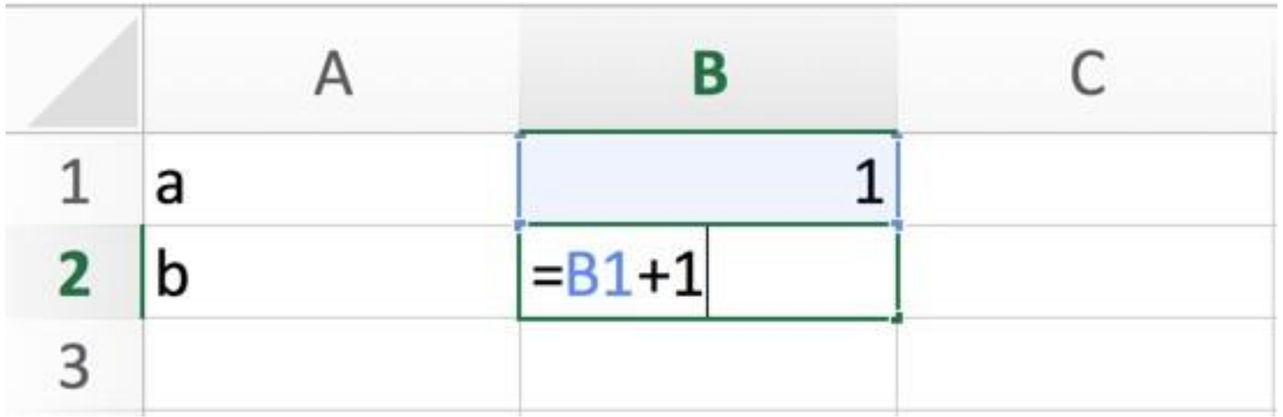
```
a = 1;
b = a + 1;

a = 2;

print a; ---> 2
print b; ---> 3
```

Programmation réactive?

- Ce n’est pas nouveau
- Vous l'avez déjà fait avant!
- Microsoft Excel est une interface de programmation réactive!



b = a + 1, dans Excel

Programmation réactive?

- **ReactiveX:** Une API pour la programmation asynchrone avec des flux observables
- **RxJS** est la version javascript
- Open Source

http://rxjs.dev

We use ReactiveX

Programmation réactive avec RxJS

- Flux de données asynchrone représenté par des **Observables**, manipulés par de riches **Opérateurs**.

RxJS = Observables + Opérateurs

Promise vs Observable

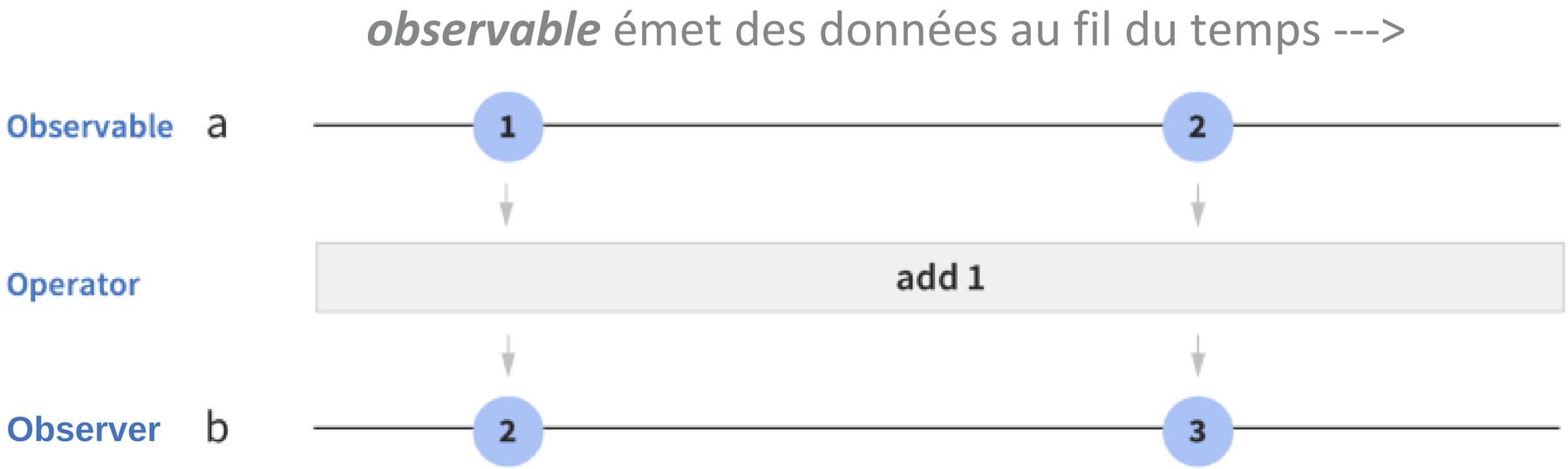
- Promise

- Observable

Aussi composition, annulation, *retry*, …

Programmation réactive avec RxJS

- Diagramme “Marbles”



observer écoute ces données avec un abonnement (***subscription***)

Syntaxe RxJS

- Exposer un **observable** dans un service

DataService

```
GetProducts() : Observable {
   return this
            .http
            .get(“http://api...”);
}
```

- **Subscribe()** depuis un composant

ProductListComponent

```
dataService
    .GetProducts()
    .subscribe(results => {
        this.products = results;
    });
```

Cet exemple est orienté client/serveur, mais les observables peuvent être utilisés avec une variété d'autres cas :
- événements (click bouton, …)
- manipulation de données, filtrage de tableaux, …
- timers
- …

RxJS

n'est

PAS

uniquement pour

HTTP!!!

Les **Observables** ne sont que des **fonctions** qui lient un ***observer*** à un ***producer***.

C'est tout.

Typiquement Composants

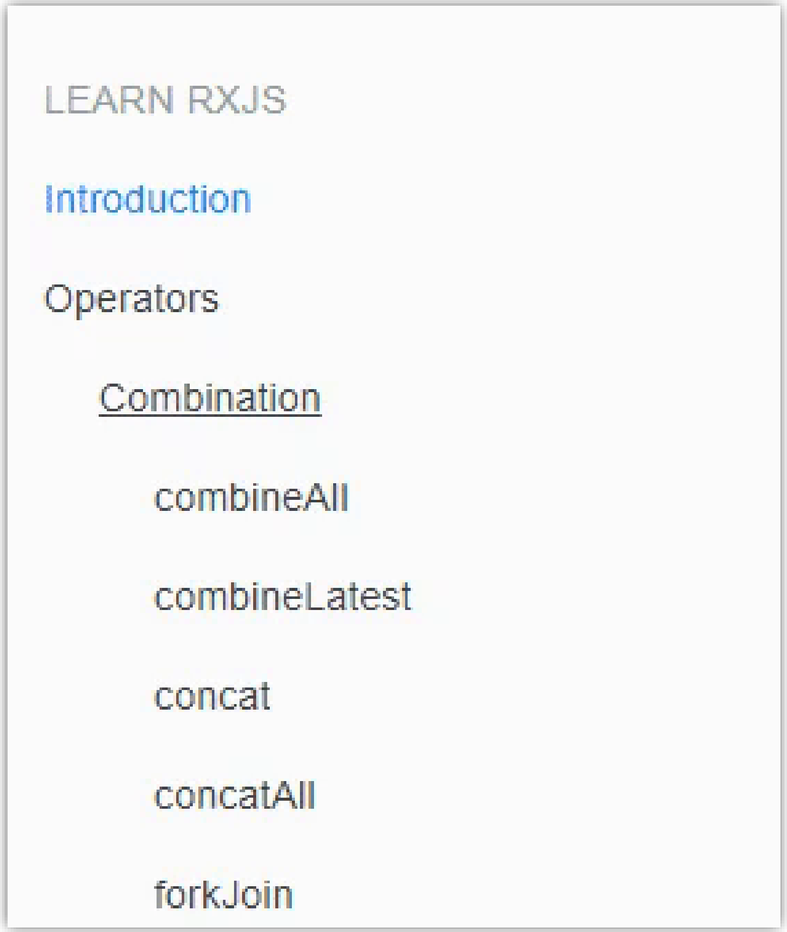
- HttpClient get
- DOM events
- Web socket
- ...

http n'est pas impliqué...

Opérateurs RxJS

- Map
- Merge
- Concat
- First, Last, Skip
- Count, Average, Min, Max
- …

http://rxmarbles.com/



```
http://rxmarbles.com/
```

DÉMONSTRATION / ATELIER

Quel opérateur utiliser?

```
https://rxjs.dev/operator-decision-tree
```

DÉMONSTRATION / ATELIER

Récap RxJS

- **RxJS** est une librarie pour la **programmation réactive** utilisant les **Observables**, pour faciliter la composition de code asynchrone ou basé sur des *callback*.
- RxJS est au **coeur de Angular**

Récap RxJS

- **Observable stream**, **Observable**, ou juste **stream** font référence à la même chose: la collecte d'éléments de données.
- Les observables sont **paresseux**, ce qui signifie qu'ils n'émettent aucun élément tant que nous n'y sommes pas abonnés (subscribe).
- Les observables continuent d'émettre des valeurs jusqu'à ce que le flux soit terminé, qu'une erreur se produise ou que nous nous désabonnions (unsubscribe).

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

HTTP

communications avec le serveur

Angular et HTTP

- Les services REST peuvent être consommés avec le **service HttpClient** (import du **HttpClientModule** depuis @angular/common/http)
- Fonctions standards **get, put, post, delete** sont supportées
- Utiliser **RxJS Observables** pour opérations **asynchrones**

Angular et HTTP

product.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable()
export class ProductService {
    constructor(private http: HttpClient) { }

    products$: Observable<Product[]> =
                this.http.get<Product[]>('http://api.products.com')
    }
}
```

Imports

Injecter le service HttpClient

Initialise un flux de produits observable

Parse la réponse JSON

$ est une convention de nommage populaire pour identifier les observables.

Angular et HTTP

product-list.component.ts

```
@Component(
    { templateUrl: 'product-list.component.html'}
)
export class ProductListComponent implements OnInit {
    public products:Product[] = [];
    constructor(private productService: ProductService) {}

    ngOnInit() {
            this.productService.products$
                .subscribe(
                    data => this.products = data,
                    error => console.log(error)
                );
    }
}
```

Injecte le service

Subscribe de l'Observable

Ou utiliser le pipe **async**!

DÉMONSTRATION / ATELIER

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Se désabonner des Observables RxJS

```
import { Component, OnInit, OnDestroy } from '@angular/core';
import { Subscription } from 'rxjs';

@Component({
  selector: 'app-product-list',
  templateUrl: './product-list.component.html'
})
export class ProductListComponent implements OnInit, OnDestroy {
  subscription: Subscription = new Subscription();

  ngOnInit() {
    this.subscription.add(
        this.productService.products$.subscribe(...)
    );
  }

  ngOnDestroy() {
    if(this.subscription) {
      this.subscription.unsubscribe();
    }
  }
}
```

Ou… juste utiliser le pipe **async**!

Le pipe async

- S’utilise avec un **Observable**
- *Unsubscribe()* automatique!

```
<ng-template #loading>
    <h2>Loading products...</h2>
</ng-template>

<ul class="products" *ngIf="products$ | async as products;else loading">
    <li *ngFor="let product of products">
        {{ product.name | uppercase }}
   </li>
</ul>
```

Modules

Module Angular

- **Permet de grouper des objets** (composants, pipes, …) afin d'obtenir une app modulaire
- **Bénéfices:**
  - Organisation du code
  - Scénarios de chargements à la demande
  - Compilation AoT (*Ahead of Time*): performances
  - Optimisation du code: *Tree Shaking*

Module Angular

app.module.ts

```
@NgModule({
  declarations: [ Component1, Directive1, Pipe1],
  imports: [ CommonModule ],
  exports: [ Component1 ],
  bootstrap: [ AppComponent ]
})
class MyAppModule {}
```

déclarer tous les objets (composants, directives, pipes) du module

tout objet des modules importés, ou il est déclaré comme *export*, est disponible

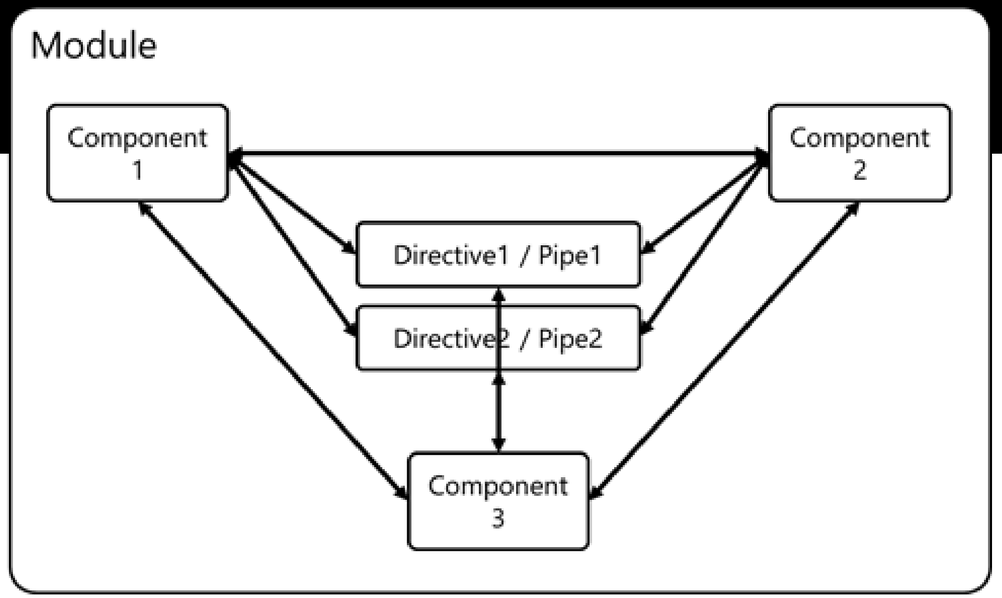
objets visibles aux autres modules

composant initial, uniquement dans le module root

Module Angular

```
@NgModule({
   declarations: [Component1, Component2, Component3, Directive1,
Directive2, Pipe1, Pipe2]
})
class MyAppModule {}
```

Dans un module tous les objets déclarés se connaissent

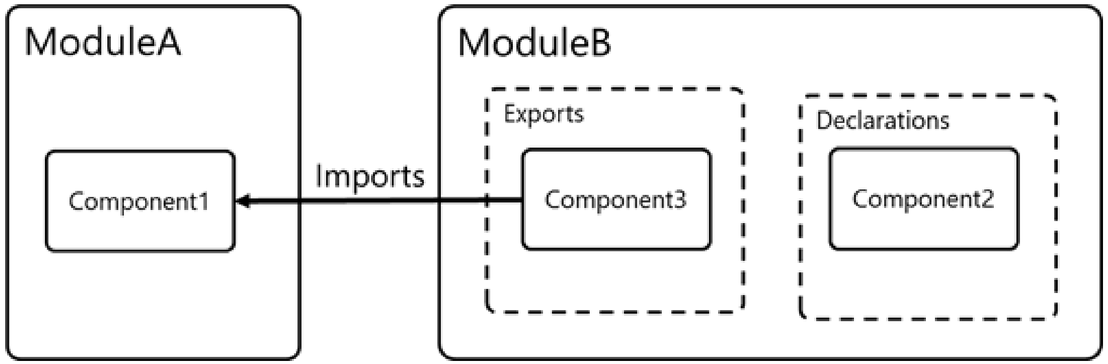


Module Angular

```
@NgModule({
   declarations: [Component1],
   imports:      [ModuleB]
})
class ModuleA {}
```

```
@NgModule({
   declarations: [Component2, Component3],
   exports:      [Component3]
})
class ModuleB {}
```

Component1 peut utiliser Component3 dans son template mais pas Component2



DÉMONSTRATION / ATELIER

Questions ?
Feedback ?

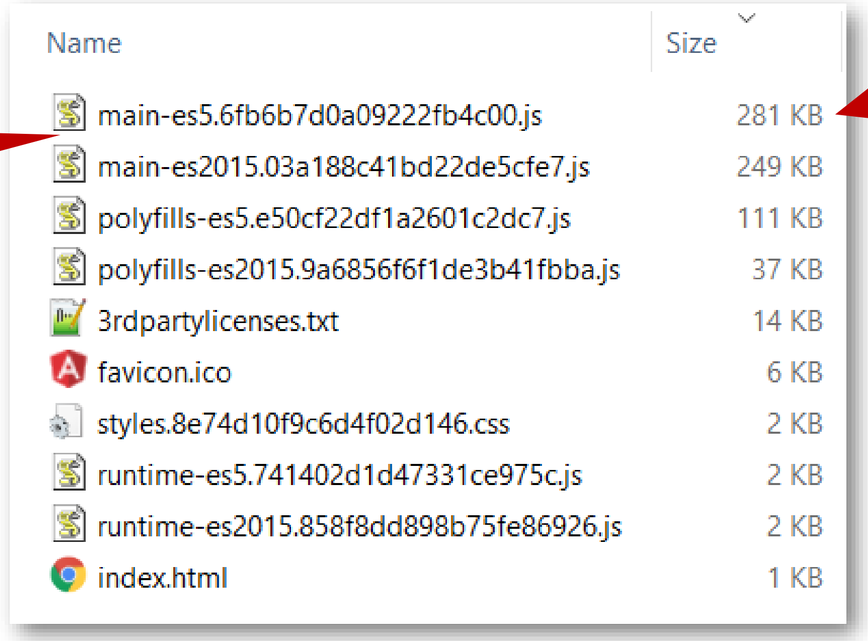
Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Déploiement

Préparer un déploiement

- Générer les *bundles* optimisés

```
> ng build --prod
```



Ids uniques pour sortir du cache

fichiers es2015: Syntaxe moderne, polyfill minimal

fichiers es5: Syntaxe ES5, polyfill complet

DÉMONSTRATION / ATELIER

CLI utilise WebPack

Pré-compilation (AoT)
- Pré-compile les templates
- Retire Ng compiler du bundle

Bundling
- Consolide tous les fichiers Js dans un seul gros package

Tree Shaking
- Retire les exports inutilisés
- Compression du code

Minification JavaScript

Gzip ← avec la compression gzip activée sur le serveur web

```
> ng build --prod
```

Bundle peut devenir jusqu'à 100x plus compact!

Support des browsers

- Fichier **browserlist:** utilisé par le build system pour ajuster le CSS et JS afin de supporter les navigateurs spécifiés.

```
> 0.5%
last 2 versions
Firefox ESR
not dead
not IE 9-11
```

Pour le support de IE 9-11, enlever 'not'

Aussi vérifier src\polyfills.ts (2 lignes à décommenter)

Puis créer une config ES5

http://angular.ac/ie

Déployer!

- Déployer avec la CLI (depuis CLI 8.3)

```
> ng add [provider]
> ng deploy
```

@angular/fire
angular-cli-ghpages
ngx-deploy-npm
@netlify-builder/deploy
@zeit/ng-deploy
@azure/ng-deploy

- Support de:
  - Firebase, GitHub pages, Netlify, AWS, Zeit, npm, et Azure

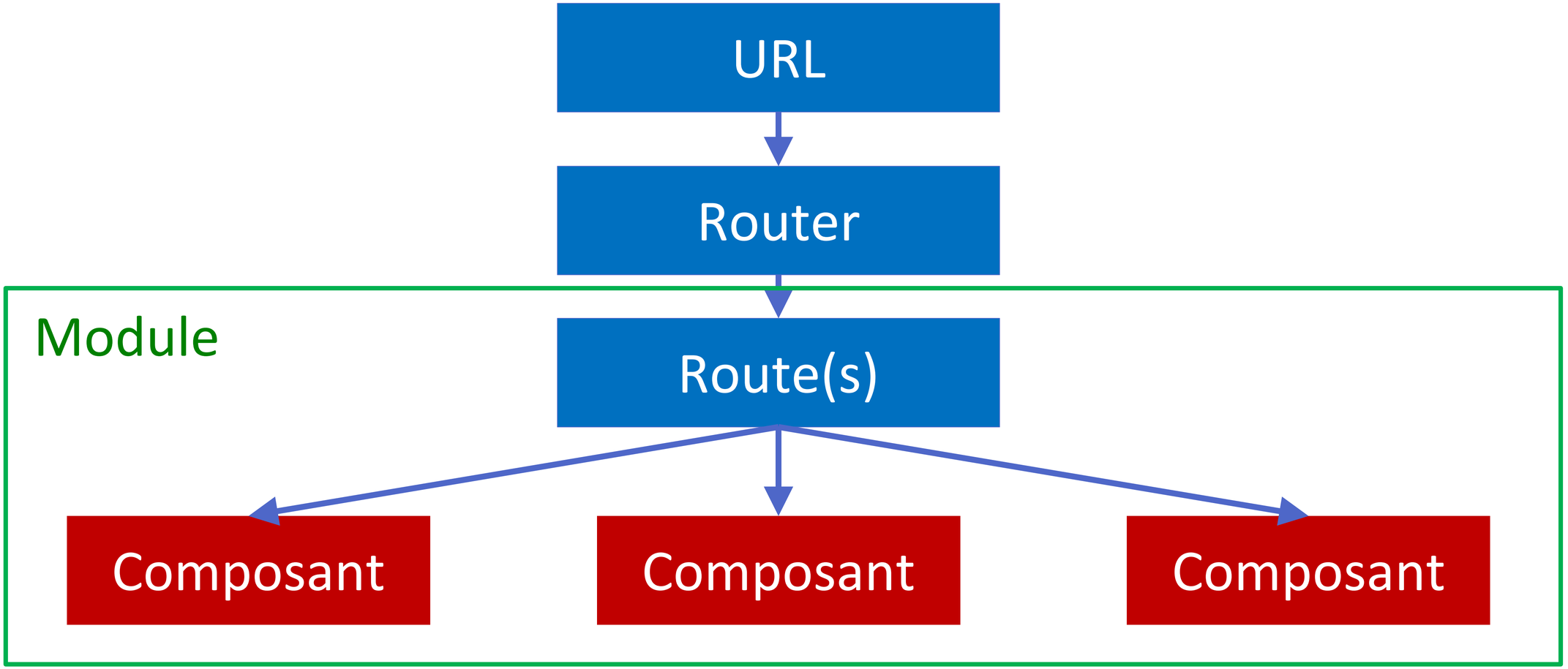
http://angular.ac/ngdeploy

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Routeur

Routeur ?



Configuration des Routes

- Définir les **routes** dans un fichier dédié

app.routing.ts

```
import { Routes, RouterModule } from '@angular/router';
import { ProductListComponent } from './product-list.component';
import { ContactComponent } from './contact.component';

const appRoutes: Routes = [
  { path: 'products', component: ProductListComponent },
  { path: 'contact', component: ContactComponent }
];

export const routing = RouterModule.forRoot(appRoutes);
```

Configuration des Routes

- Possibilité d’avoir des **routes enfants**

app.routing.ts

```
export const AppRoutes = [
  { path: 'products',
    children:[
        { path: '', component: ProductListComponent }
        { path: ':id', component: ProductDetailComponent }
    ]
  },
  { path: 'contact', component: ContactComponent }
])
```

Configuration des Routes

- Définir une **route par défaut**

app.routing.ts

```
export const AppRoutes = [
  { path: '', redirectTo:'/products', pathMatch: 'full' },
  { path: 'products',
    children:[
        { path: '', component: ProductListComponent }
        { path: ':id', component: ProductDetailComponent }
    ]
  },
  { path: 'contact', component: ContactComponent }
])
```

Router Module

- Importer les routes dans le module principal

app.module.ts

```
import { AppComponent } from './app/';
import { routing } from './app.routing';

@NgModule({
    declarations: [AppComponent],
    imports: [routing],
    bootstrap: [AppComponent]
})
export class AppModule {
}
```

Directives de Routage

- Composants chargés dans un **<router-outlet></router-outlet>**

app.component.html

```
<h1>
  {{ title }}
</h1>

<router-outlet></router-outlet>

<footer>
  Copyright Angular 2018
</footer>
```

Navigation et Routes

- Naviguer entre les Composants avec la directive **RouterLink**

Template

```
<nav>
   <a routerLink="/home">Home</a>
   <a routerLink="/products">Products</a>
   <a routerLink="/contact">Contact</a>
</nav>
<router-outlet></router-outlet>
```

- Ou par code

```
router.navigateByUrl('/products/');
```

Paramètres de Route

- Utiliser **ActivatedRoute** pour récupérer les paramètres

Component

```
import { ActivatedRoute } from '@angular/router';

constructor(private route: ActivatedRoute) { }

ngOnInit() {
   let id = this.route.snapshot.params['id'];
   if (id) {
        this.productService
            .getProductById(id)
            .subscribe(data => this.product = data);
    }
}
```

DÉMONSTRATION / ATELIER

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Lazy Loading

Lazy Loading ?

Charger à la demande ProductsModule!

app.routing.ts

```
const routes: Routes = [
 {path: '', redirectTo:'/home', pathMatch:'full’},
 {path: 'products', loadChildren: () =>
            import('./products/products.module').then(m => m.ProductsModule)},
 {path: 'home', component: HomeComponent},
 {path: 'contact', component: ContactComponent},
];
```

Faire ses routes relatives à ‘products/’

products.routing.ts

```
const routes: Routes = [
  { path: 'products', component: ProductListComponent },
  { path: 'products/:id', component: ProductDetailComponent }
];
```

products.routing.ts

```
const routes: Routes = [
  { path: '', component: ProductListComponent },
  { path: ':id', component: ProductDetailComponent }
];
```

Ne pas importer les modules chargés à la demande!

app.module.ts

```
@NgModule({
imports:[BrowserModule, HttpModule, ProductsModule],
})
export class AppModule { }
```

app.module.ts

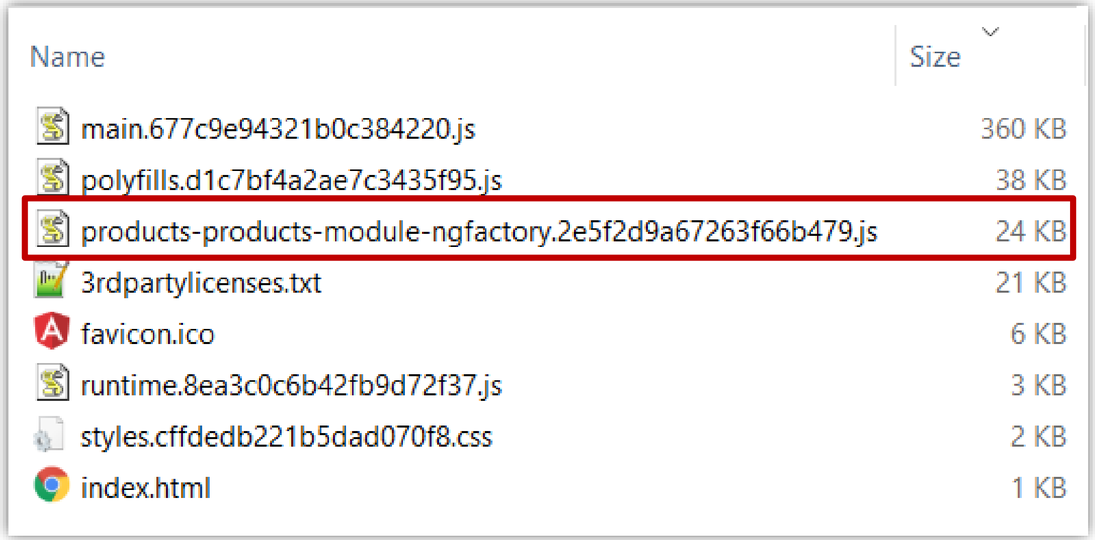
```
@NgModule({
imports:[BrowserModule, HttpModule],
})
export class AppModule { }
```

DÉMONSTRATION / ATELIER

Lazy Loading

- Nous avons un nouveau bundle!

```
> ng build --prod --named-chunks
```



Noms familiers pour vos bundles

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Formulaires et Validation

Forms Provider

- Importer le **ReactiveFormsModule**

app.module.ts

```
import { AppComponent } from './app/';
import { ReactiveFormsModule } from '@angular/forms';

@NgModule({
    declarations: [AppComponent],
    imports: [ReactiveFormsModule],
    bootstrap: [AppComponent]
})
export class AppModule {
}
```

model driven (ou reactive) form

Model Driven Form

- Définir le formulaire et ses éléments HTML

Template

```
<form>

  <label for="name">Name:</label>
  <input
    id="name"
    type="text"
    formControlName="name">
```

Model Driven Form

- Composant: Imports et Déclarations

```
import { Component } from '@angular/core';
import { FormBuilder, Validators, FormGroup } from
'@angular/forms';

@Component({
   templateUrl: 'product-insert.component.html',
    ...
})
export class ProductInsertComponent {
   insertForm: FormGroup;
   . . .
```

Model Driven Form

- Composant: Constructeur

Component

```
@Component({
   ...
})
export class ProductInsertComponent {
  constructor(
    private fb: FormBuilder,
    private productService: ProductService) { }
```

Model Driven Form

- Composant: Définir contrôles et validation

Component

```
ngOnInit() {
   this.insertForm = this.fb.group({
      'name': ['',                    <-- valeur par défaut
               [Validators.required,
                Validators.minLength(3),   <-- validateur synchrone
                Validators.maxLength(50)]
              ]                       <-- aussi possibilité de validateurs asynchrones
   });
}
```

Model Driven Form

Validation **côté client uniquement!!**

- Pour feedback immédiat et l'expérience utilisateur (mais pas pour la sécurité)
- Ne pas oublier de complémenter par une validateur côté serveur également!

Model Driven Form

- Lier le Modèle au Formulaire

Template

```
<form [formGroup]="insertForm">

. . .

</form>
```

Model Driven Form

- Fonction de *Submit*

Template

```
<form (ngSubmit)="onSubmit()"
      [formGroup]="insertForm">

. . .
</form>
```

Component

```
onSubmit() {

   this.productService.insertProduct(this.insertForm.value);

}
```

Model Driven Form

- Messages de validation et style d'erreurs

Template

```
<div *ngIf="name.touched && name.errors" class="errorMessage">

    <span *ngIf="name.errors.required">Name is required</span>
    <span *ngIf="name.errors.minlength">Min 3 chars</span>
    <span *ngIf="name.errors.maxlength">Max 50 chars</span>

</div>
```

Name:

Name is required

Ou utiliser une librairie…

https://ngx-valdemort.ninja-squad.com

Model Driven Form

- Styliser les éléments invalides avec les classes css d'Angular

css

```
.ng-touched.ng-valid {
  border: 2px solid green;
}

.ng-touched.ng-invalid {
  border: 2px solid red;
}
```

Name:

Name is required

DÉMONSTRATION / ATELIER

Questions ?
Feedback ?

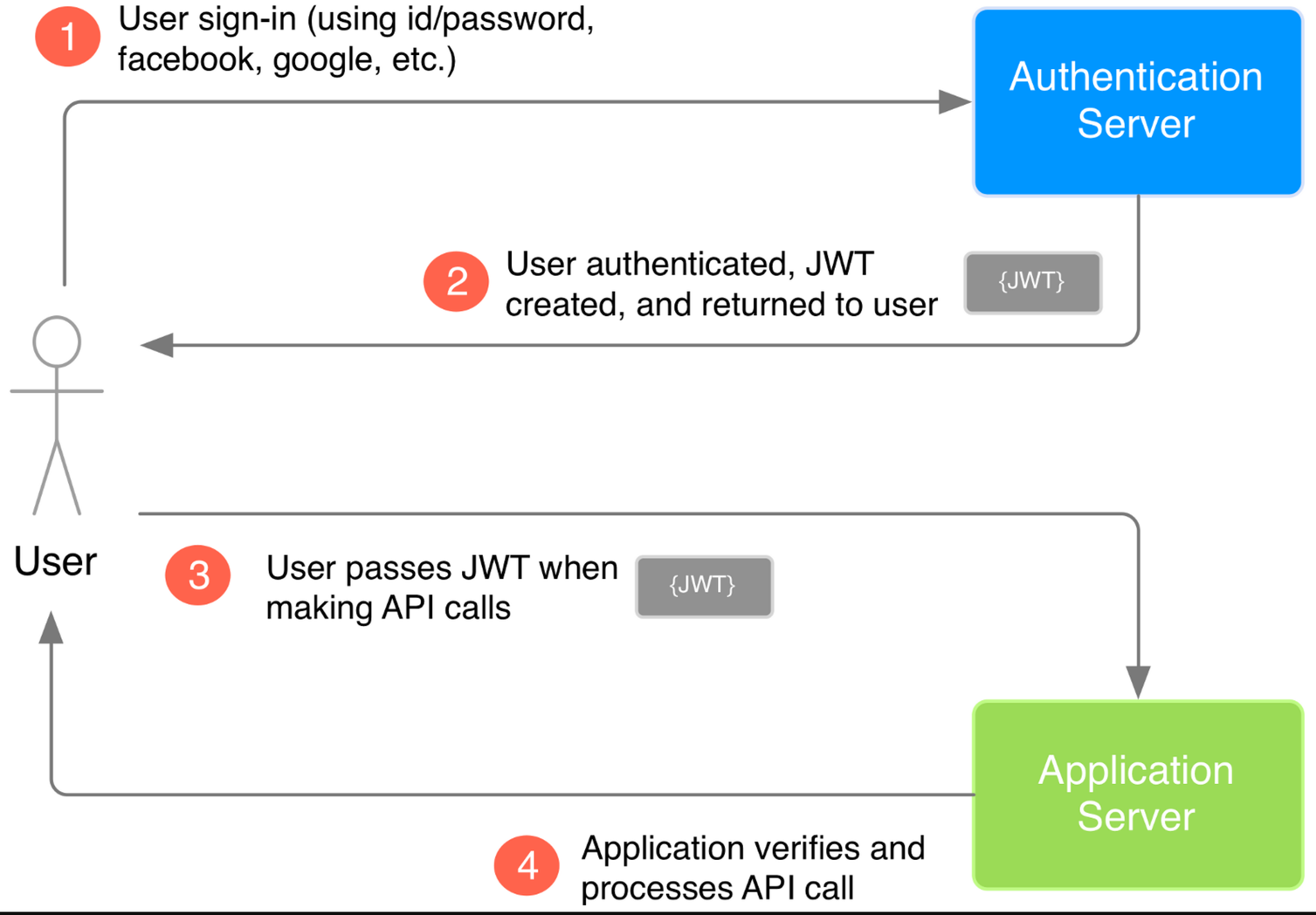
Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Authentification et sécurité

Authentification avec Angular

- Une app Angular est juste… du JavaScript
  - Du code côté client n’est **jamais** sécurisé!
- En règle générale, vos données sont stockées dans une base de données et exposées via une API REST sécurisée sur un serveur.
- **JSON Web Tokens (JWT)** est le meilleur choix pour implémenter un système d'authentification dans une app Js

Authentification basée sur JSON (JWT)



De quoi avons nous besoin?

- Composant de Login (username / password)
- Service d'Authentification (avec HttpClient)
  - *Login* (stocker le token) / *Logout*, *IsAuthenticated*
- Limiter l'accès aux utilisateurs authentifiés (Router Guards)
- Http Interceptors
  - Insertion du token de sécurité dans les en-têtes HTTP
  - Détecter les réponses non autorisées (401) et redirection vers le composant de login

DÉMONSTRATION

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Meilleures Pratiques

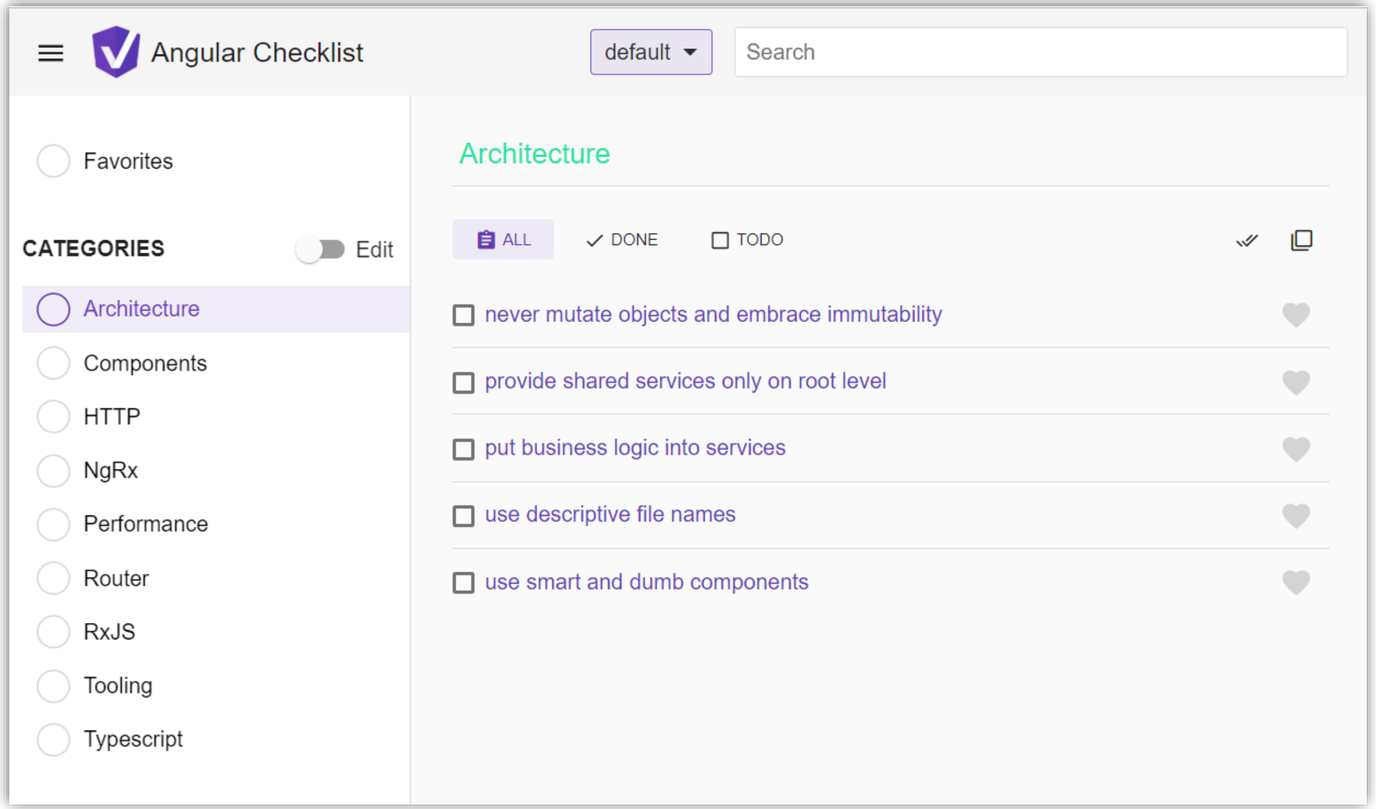
Meilleures Pratiques

Lire le *Angular Style Guide*!

- Conseils et recommandations
- Conventions de nommage, …
- Structure de l'application
- …

https://angular.io/styleguide

Angular Checklist



https://angular-checklist.io

Linting avec Codealyzer

- **Codelyzer** s'utilise par dessus TSlint pour analyser votre code
- Installé avec la CLI
- Applique les bonnes pratiques du Style Guide (personnaliser avec tslint.json)

http://codelyzer.com

Linter avec mise en forme et couleurs:

```
> ng lint --format stylish
```

Fixer les erreurs:

```
> ng lint --fix
```

DÉMONSTRATION

Questions ?
Feedback ?

Appuyez sur la barre d'espace pendant que vous parlez pour activer le son!

Ressources

- Angular Observable Data Services http://tinyurl.com/hennfnw
- Material http://tinyurl.com/z9q2wn5
- SEO http://tinyurl.com/hvvedsd
- Modules http://tinyurl.com/ha2bdyl
- Routeur http://tinyurl.com/jfyzccu
- Tests http://tinyurl.com/hbn6k3r
- Advanced Styling Guide http://tinyurl.com/jrn28pf
- Components Lifecycle http://tinyurl.com/h826r7k
- Internationalization http://tinyurl.com/h9xqjqe

RxJS

- http://rxmarbles.com
- https://www.learnrxjs.io
- http://reactivex.io/tutorials.html
- https://blog.strongbrew.io/rxjs-best-practices-in-angular/
- http://reactive.how
- https://rxviz.com
- http://reactive.how/rxjs/explorer
- https://juristr.com/blog/2018/10/journey-promises-to-rxjs/

Articles de très haute qualité!

- **https://indepth.dev/angular/**
- https://blog.nrwl.io
- https://blog.thoughtram.io
- http://blog.mgechev.com
- http://www.syntaxsuccess.com/articleList/angular
- http://blog.angular-university.io
- https://coryrylan.com
- https://blog.strongbrew.io/
- https://blog.angulartraining.com/

Certificat!

ACADÉMIE ANGULAR

Fondamentaux Angular

Ecrire votre nom ici...

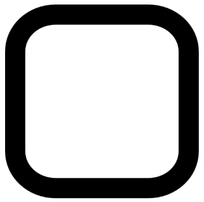
a complété(e) avec succès la formation Angular Fondamentaux

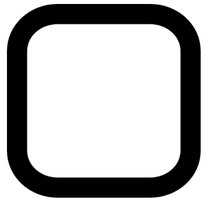
ANGULAR ACADEMY
www.academieangular.ca

INSTRUCTEUR: Laurent DUVEAU

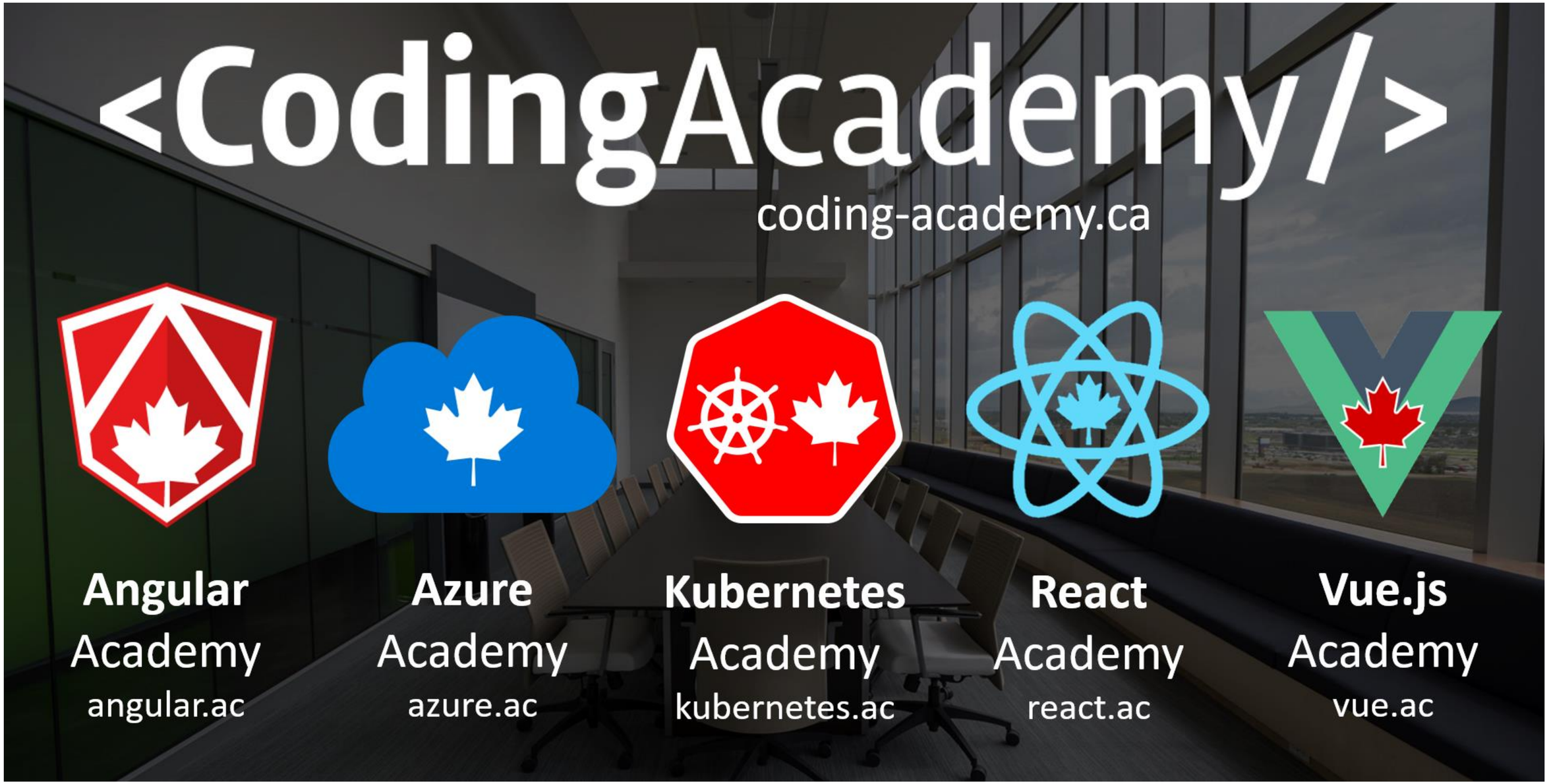
DATE:

Obtenez les badges!





Trouvez votre prochaine formation: **https://angular.ac/fr**


<CodingAcademy/>
coding-academy.ca
Angular
Academy
angular.ac
Azure
Academy
azure.ac
Kubernetes
Academy
kubernetes.ac
React
Academy
react.ac
Vue.js
Academy
vue.ac

ACADÉMIE ANGULAR

Mercí!

http://angular.**ac**/eval