ANGULAR

Ababacar Fall

ANGULAR

01 Notion de framework Front

02 Etude Comparative des Framework Front(Angular,React Js, Vue JS)

03 Choix Du Framework Angular: Etude Évolutive et Version

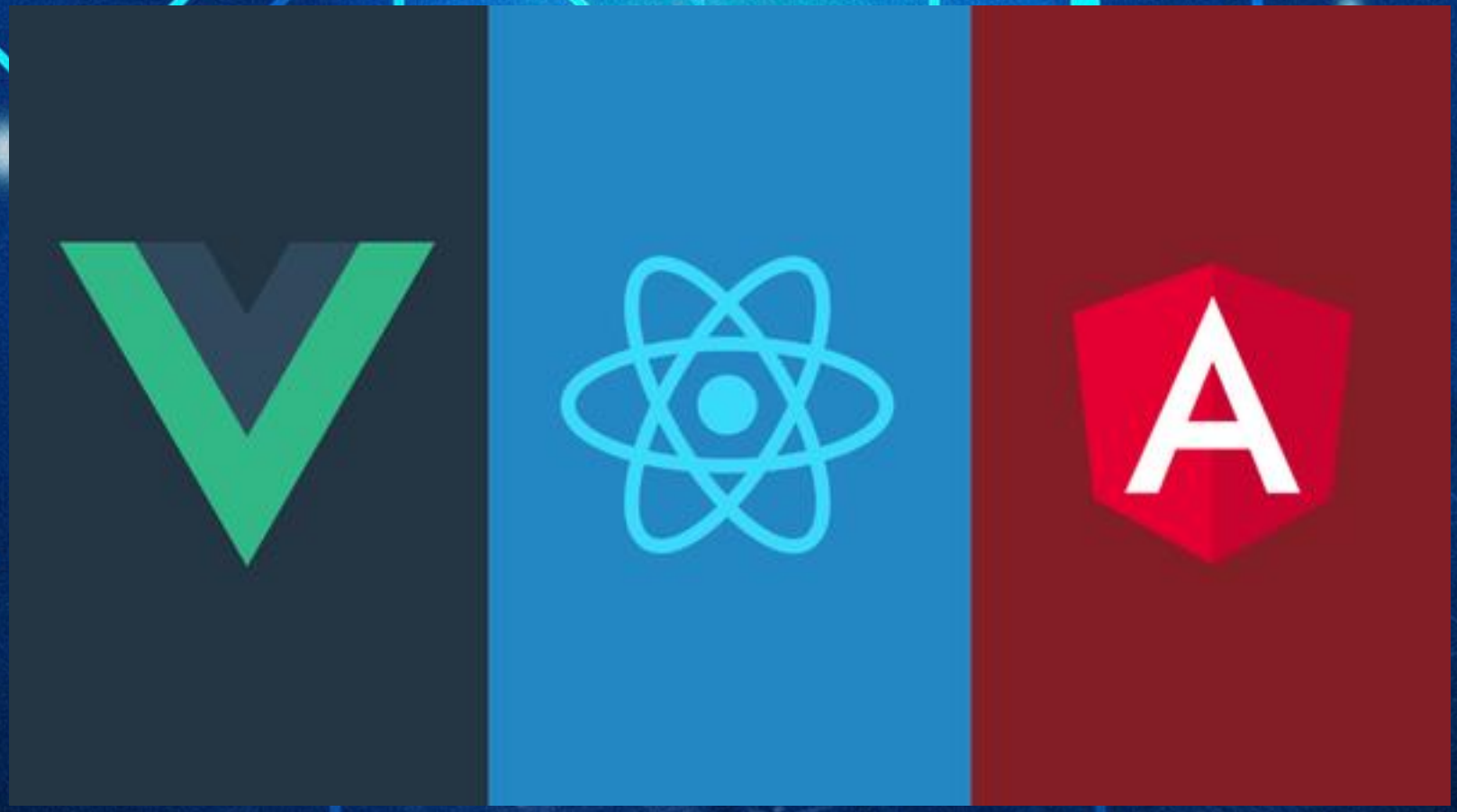
04 ES6 et TypeScript(Classes, Interfaces,Classes Anonymes et Arrow fonctions)

04 Architecture Angular

Notion de Framework Front-END

Notion de Framework Front-END

Les frameworks Front-End servent à définir ce que va voir le visiteur lorsque qu'il va arriver sur le site. Ils utilisent des langages clients tels que le langage HTML, CSS ou Javascript. En exemple de framework frontend on peut retrouver Vue.JS, React ou bien AngularJS.



Etude Comparative des Framework Front(Angular,React Js, Vue JS

Angular

ANGULAR

Angular est l'un des frameworks JS les plus performants et robustes. Ses mises à jour fréquentes (la version 6 était sortie en mai 2018) démontrent l'activité incessante de sa grande communauté, et la capacité d'adaptation d'Angular.
En plus de cette communauté extrêmement présente, la documentation officielle est détaillée et permet de mieux l'appréhender. On retrouve ainsi davantage de modèles complets qui aiguillent sur leur mise en place, et offrent un panel de solutions « prêtes à l'emploi ».
Très populaire et complet, d'autant qu'il prend en charge le CSS + HTML natif, on retrouve Angular chez de nombreuses entreprises.
Néanmoins attention, car derrière ces nombreux avantages, il existe certains inconvénients qui faut prendre en compte :

- L'apprentissage d'Angular est long et ardu, ce qui peut décourager dans un premier temps, surtout lors de premiers développement en intégrant ce type de frameworks.
- Très complet pour le développement d'applications web, il peut sembler trop complexe, dans son code notamment. Attention de ne pas se perdre face aux nombreuses solutions offertes (l'expérience étant alors la meilleure des alliées).
- Certains problèmes de migration ont pu être rencontrés lors du passage d'anciennes versions vers les nouvelles. Même si ces soucis sont voués à être résolus avec les mises à jour de version, la complexité d'Angular peut entraîner des soucis dans ces mêmes mises à jour.

React JS

React JS

React, que l'on appelle aussi ReactJS, est une bibliothèque JavaScript libre, créée par Facebook en 2013. Elle permet de concevoir des applications web via la création de composants dépendants d'un état constituant une portion HTML pour chaque changement d'état. ReactJs ne gère que l'interface d'une application web, et peut donc être utilisée avec un framework MVC tel qu'Angular.

Populaire et dans la tendance actuelle, ReactJS est utilisé par de nombreuses entreprises de type « startup » ainsi que par Netflix ou Airbnb pour ne citer qu'eux. Il existe également une communauté importante, et donc un appui pour sa progression permanente.

Étant 100% open source, React est sans cesse mis à jour et s'adapte en permanence aux contraintes de développement qui évoluent dans le temps.Sur React, la création est simplifiée grâce à sa flexibilité et son API qui permettent une utilisation facile des composants. À noter aussi que ReactJS travaille avec un DOM virtuel, permettant de gagner en performance et en rapidité.En face de ces différents avantages, on retrouve quelques inconvénients :

•Les modèles des composants sont en JSX et pas en HTML, ce qui nécessite une légère adaptation (bien que l'on reste assez proche du HTML tout de même).

•Malgré les éléments qui existent sur la sphère web pour vous aider, il y a un manque de documentation « officielle » sur ReactJS qui ne facilite pas son appréhension lorsque l'on est néophyte. À savoir que ReactJS demande de la pratique, notamment pour l'intégration dans le framework MVC (malgré la simplicité d'utilisation).

•React offre énormément de choix aux développeurs. Ce qui peut être perçu comme une qualité, est aussi un défaut puisqu'il ne tranche pas clairement et laisse au développeur le soin de décider. Pas toujours des plus confortable.

Vue JS

Vue JS

VueJS est notre troisième framework JavaScript du jour. Créé en 2013 et lancé en 2014, il est le plus facilement appréhendante des trois. L'un des gros avantages de VueJS, c'est sa simplicité au vu des choix réduits qu'il offre (contrairement à un ReactJS par exemple). Il va s'adapter en fonction de vos compétences, que vous soyez débutant ou avancé en développement front, et va permettre d'entrer dans le monde des frameworks JS le plus aisément possible. Associé à ça, vous retrouvez une documentation officielle pour vous accompagner.

Pour finir, il est important de noter qu'il est HTML et CSS compatible : vous n'aurez donc pas besoin de vous adapter à un nouveau langage (et vous gagnerez ainsi pas mal de temps).

En inconvénients de VueJS, nous pouvons citer :

•Son manque d'historique (il est assez récent) le rend moins optimisé que les autres, puisqu'il a moins vécu et a moins été confronté à des situations problématiques. On peut donc parfois rencontrer certains problèmes d'intégrations.

•Sa communauté restreinte (dû notamment au fait qu'il est le plus jeune des trois frameworks).

Choix Du Framework Angular: Etude Évolutive et Version

Choix Du Framework Angular: Etude Évolutive et Version

> Historique des versions d'Angular

Angular sort une nouvelle version majeure tous les 6 mois environ :

- Angular 2 est sorti en septembre 2016,
- Angular 4 en mars 2017,
- Angular 5 en novembre 2017,
- Angular 6 en mai 2018,
- Angular 7 en octobre 2018,
- Angular 8 en mai 2019,
- Angular 9 en février 2020,
- Angular 10 en juin 2020.

Choix Du Framework Angular: Etude Évolutive et Version

AngularJS vs Angular

Google a simplement fait **une erreur de communication**, en voulant garder le même nom qu'AngularJS (v1), alors que **le nouvel Angular est en réalité un framework différent**.

Du coup, il a d'abord fallu parler d'Angular 2 explicitement, pour bien distinguer la première version et la nouvelle. Les gens s'y sont habitués, et maintenant ne comprennent pas pourquoi nous en sommes déjà à Angular 10.

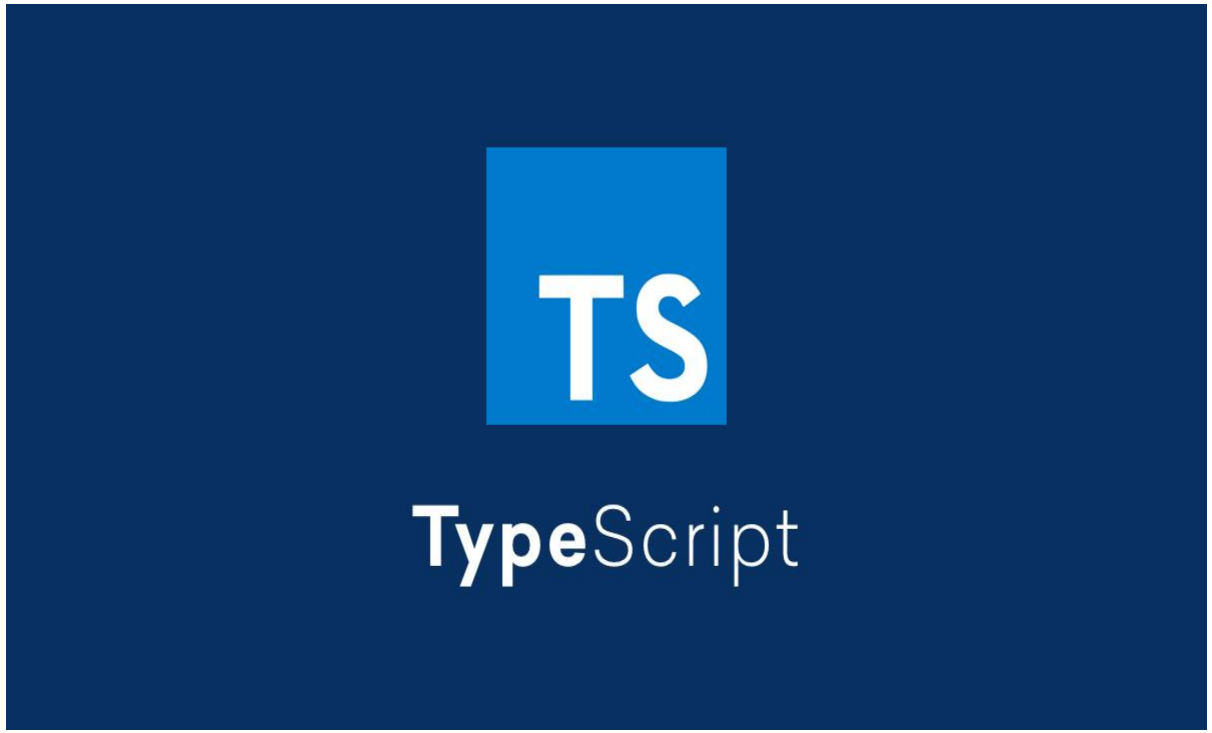
Mais c'est tout à fait normal : **c'est le fonctionnement normal du semantic versioning** que tout le monde utilise :

- Angular 4 = Angular 2, avec quelques nouveautés.
- Angular 5 = Angular 4, avec quelques nouveautés.
- Angular 6 = Angular 5, avec quelques nouveautés.
- Angular 7 = Angular 6, avec quelques nouveautés.
- Angular 8 = Angular 7, avec quelques nouveautés.
- Angular 9 = Angular 8, avec quelques nouveautés.
- Angular 10 = Angular 9, avec quelques nouveautés.

ES6 et TypeScript(Classes, Interfaces,Classes Anonymes et Arrow fonctions)

TypeScript

TypeScript est un pur langage de programmation **open source** orienté objet. Il s'agit d'un **sur** - **ensemble** de JavaScript fortement typé qui se compile en JavaScript brut. TypeScript est développé et maintenu par **Microsoft** sous la licence **Apache 2** . Il n'est pas exécuté directement sur le navigateur. Il a besoin d'un compilateur pour compiler et générer dans un fichier JavaScript. Le fichier source TypeScript a l'extension ".ts". Nous pouvons utiliser n'importe quel fichier " **.js** " valide en le renommant en fichier ".ts". TypeScript est la version ES6 de JavaScript avec quelques fonctionnalités supplémentaires.



INTERFACES

L'un des principes fondamentaux de TypeScript est que la vérification de type se concentre sur la *forme des* valeurs. Ceci est parfois appelé «typage canard» ou «sous-typage structurel». Dans TypeScript, les interfaces remplissent le rôle de nommer ces types et constituent un moyen puissant de définir des contrats dans votre code ainsi que des contrats avec du code en dehors de votre projet.

Notre première interface

Le moyen le plus simple de voir comment les interfaces fonctionnent est de commencer par un exemple simple:

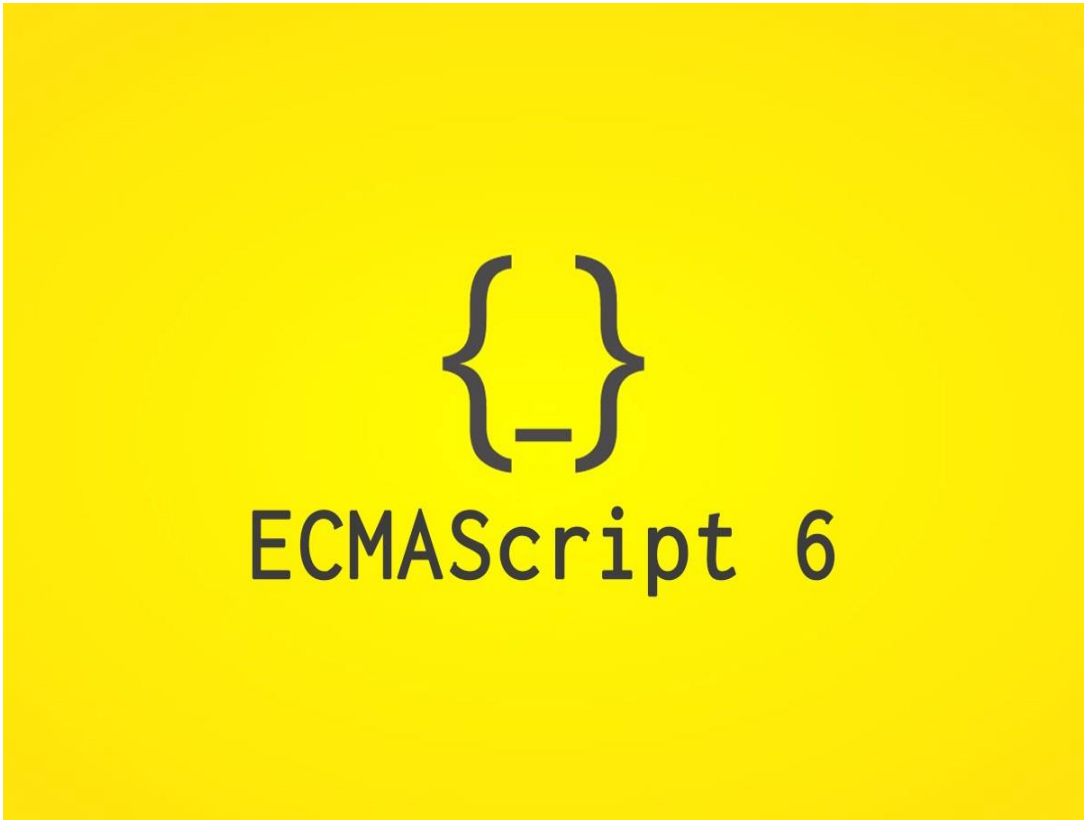
```
function printLabel(labeledObj: { label: string }) {
  console.log(labeledObj.label);
}

let myObj = { size: 10, label: "Size 10 Object" };
printLabel(myObj);
```

ES6 et TypeScript(Classes, Interfaces,Classes Anonymes et Arrow fonctions)

ES6

ECMAScript (ES) est une spécification de **langage de script** normalisée par **ECMA international** . Il a été créé pour standardiser JavaScript. Le langage de script ES contient de nombreuses implémentations, et la plus populaire est **JavaScript** . Les développeurs utilisent *ECMAScript* principalement pour **les scripts côté client** du World Wide Web (WWW).

La **sixième** édition de la norme ECMAScript est ECMAScript6 ou ES6 et renommée plus tard **ECMAScript 2015** . C'est une amélioration majeure du langage JavaScript, qui nous permet d'écrire des programmes pour des applications complexes. Il ajoute de nombreuses fonctionnalités destinées à faciliter le développement de logiciels à grande échelle. Les navigateurs Web ES6 les plus courants sont **Chrome** et **Firefox** . Un **transpilateur** convertit le code basé sur ES6 en **ES5** qui est pris en charge par de nombreux navigateurs. TypeScript est un transpilateur. Grunt, Gulp et Babel sont d'autres transpilateurs pour compiler les modules. Par conséquent, TypeScript prend en charge ES6.



Arrow Functions

Une expression de fonction fléchée (arrow function en anglais) permet d'avoir une syntaxe plus courte que les expressions de fonction et ne possède pas ses propres valeurs pour this, arguments, super, ou new.target. Les fonctions fléchées sont souvent anonymes et ne sont pas destinées à être utilisées pour déclarer des méthodes

```
const multiplyByTwo = (num) => {
    return num * 2;
}
```

Classes

Les classes JavaScript ont été introduites avec ECMAScript 2015. Elles sont un « sucre syntaxique » par rapport à l'héritage prototypal. En effet, cette syntaxe n'introduit pas un nouveau modèle d'héritage dans JavaScript ! Elle fournit uniquement une syntaxe plus simple pour créer des objets et manipuler l'héritage.

Pour utiliser une déclaration de classe simple, on utilisera le mot-clé `class`, suivi par le nom de la classe que l'on déclare (ici « Rectangle »).

```
1  class Rectangle {
2    constructor(hauteur, largeur) {
3      this.hauteur = hauteur;
4      this.largeur = largeur;
5    }
6  }
```

Classes Anonymes

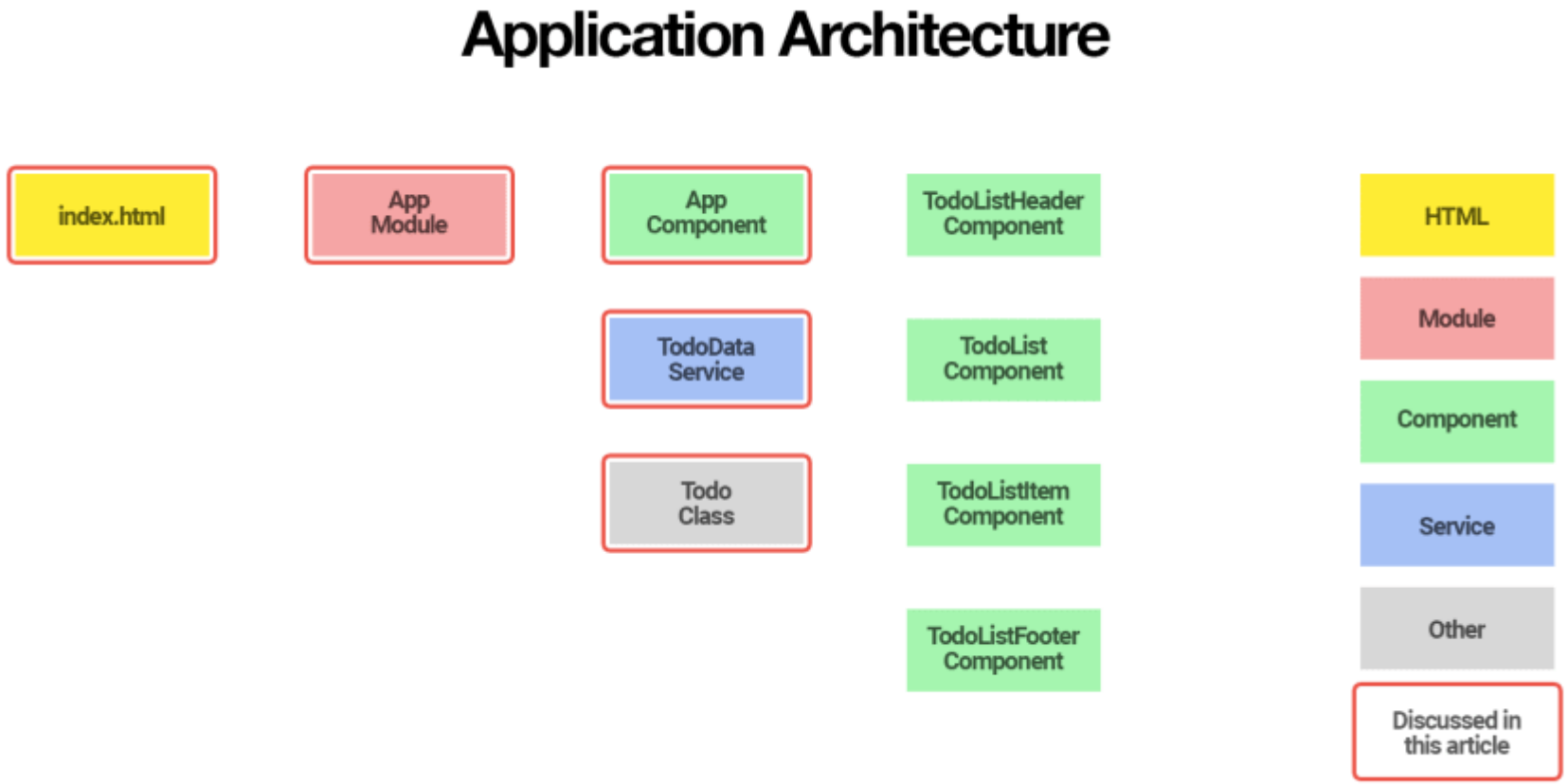
Une expression de classe est un moyen de définir une classe avec ECMASCript 2015 (ES6). Semblable aux expressions de fonctions, les expressions de classes peuvent être nommées ou anonymes. Si l'expression est nommée, le nom de la classe ne sera local que pour le corps de la fonction. Cette syntaxe n'est qu'un « sucre syntaxique » pour faciliter l'écriture du code, elle ne modifie en aucun cas le modèle d'héritage utilisé par JavaScript qui est un modèle à base de prototypes.

```
var Toto = class {
  constructor() {}
  truc() {
    return "Coucou monde !";
  }
};

var instance = new Toto();
instance.truc(); // "Coucou monde !"
Toto.name; // "Toto"
```

Architecture Angular

Architecture Angular

Architecture Angular



Architecture Angular

Architecture Angular

Angular est une plate-forme et un cadre pour la création d'applications clientes d'une seule page à l'aide de HTML et de TypeScript. Angular est écrit en TypeScript. Il implémente les fonctionnalités de base et facultatives sous la forme d'un ensemble de bibliothèques TypeScript que vous importez dans vos applications. L'architecture d'une application Angular repose sur certains concepts fondamentaux. Les blocs de construction de base sont les NgModules , qui fournissent un contexte de compilation pour les composants . Les NgModules rassemblent le code associé en ensembles fonctionnels; une application angulaire est définie par un ensemble de NgModules. Une application a toujours au moins un module racine qui permet le démarrage, et a généralement beaucoup plus de modules de fonctionnalités . Les composants définissent des vues , qui sont des ensembles d'éléments d'écran parmi lesquels Angular peut choisir et modifier en fonction de la logique et des données de votre programme. Les composants utilisent des services , qui fournissent des fonctionnalités spécifiques non directement liées aux vues. Les fournisseurs de services peuvent être injectés dans les composants en tant que dépendances , ce qui rend votre code modulaire, réutilisable et efficace. Les modules, composants et services sont des classes qui utilisent des décorateurs . Ces décorateurs marquent leur type et fournissent des métadonnées qui indiquent à Angular comment les utiliser. Les métadonnées d'une classe de composant l'associent à un modèle qui définit une vue. Un modèle combine du HTML ordinaire avec des directives Angular et un balisage de liaison qui permettent à Angular de modifier le HTML avant de le rendre pour l'affichage. Les métadonnées d'une classe de service fournissent les informations dont Angular a besoin pour les rendre disponibles aux composants via l' injection de dépendances (DI) . Les composants d'une application définissent généralement de nombreuses vues, organisées de manière hiérarchique. Angular fournit le Routerservice pour vous aider à définir des chemins de navigation parmi les vues. Le routeur offre des capacités de navigation sophistiquées dans le navigateur.


THANK YOU