Angular : introduction

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

elmouelhi.achref@gmail.com

Plan

Angular

Angular

- Framework **TypeScript** (anciennement **JavaScript**, support expérimental de **Dart**, langage de **Google**, abandonné)
- Open-source
- Présenté par **Google** en 2009
- Permettant de créer principalement des applications web
  - Front-End
  - Single page
- Et aussi mobiles (**Ionic** ou **NativeScript**)

Angular respecte l'architecture **MVVM**

- **MVVM** : **M**odel-**V**iew-**V**iew**M**odel
- **Model** : représenté généralement par une classe référencée par la couche d'accès aux données (classe ou interface **TypeScript**).
- **View**
  - contenant la disposition et l'apparence de ce qu'un utilisateur voit à l'écran,
  - recevant l'interaction avec l'utilisateur : clic, saisie, survol...
- **ViewModel**
  - remplaçant du contrôleur dans l'architecture **MVC**,
  - connecté à la vue par le **data binding**,
  - représenté dans **Angular** par un fichier `*.component.ts`.

Angular

Angular utilise ou utilisait :

- les composants web
- l'injection de dépendance
- le **DOM** Virtuel (n'est plus utilisé depuis l'intégration d'**Ivy** dans **Angular 9**)
- le change detection basé sur **Ivy** et **zone.js**.

Angular

Injection de dépendance ?

- concept connu en programmation orientée objet.
- utilisé par plusieurs frameworks Back-End (**Spring**, **Symfony**...).
- consistant à utiliser des classes sans faire de l'instanciation statique.

Angular

DOM Virtuel ?

- introduit, initialement, par **React**.
- une représentation en mémoire du **DOM** physique.
- permettant des mises à jour plus rapides et efficaces en ne modifiant que les parties nécessaires de l'interface utilisateur avant de synchroniser ces changements avec le **DOM** physique.

Angular

DOM Virtuel ?

- introduit, initialement, par **React**.
- une représentation en mémoire du **DOM** physique.
- permettant des mises à jour plus rapides et efficaces en ne modifiant que les parties nécessaires de l'interface utilisateur avant de synchroniser ces changements avec le **DOM** physique.

Remarque

- Les modifications apportées au **DOM** physique entraînent généralement des opérations coûteuses en termes de temps de traitement.
- Chaque modification peut déclencher des réorganisations et des recalculs complexes de la mise en page.

Angular

Change detection ?

- réalisé par la librairie **zone.js** (qu'on peut trouver dans `node_modules`).
- utilisé pour synchroniser la vue avec le composant.
- chaque composant dispose d'un change detector qui surveille en particulier les expressions utilisées dans l'interpolation ou le propriété binding.
- un changement de la valeur retournée par l'expression $\Rightarrow$ mise à jour de la vue.

Angular

Quelques outils utilisés par **Angular** (< 17)

- **npm** (**n**ode **p**ackage **m**anager) : le gestionnaire de paquets par défaut pour une application **JavaScript**.
- **angular-cli** **c**ommand **l**ine **i**nterface : outil proposé par **Google** pour faciliter la création et la construction d'une application **Angular** en exécutant directement des commandes.
- **webpack** : bundler **JavaScript**
  - construit le graphe de dépendances.
  - regroupe des ressources de même nature (`.js` ou `.css`...) dans un ou plusieurs bundles.
  - fonctionne avec un système de module : un fichier **JS** est un module, un fichier **CSS** est un module...
- **Ivy** : moteur de compilation et de rendu utilisé partiellement dans **Angular** 8, intégralement depuis la version 9. Il permet d'accélérer la compilation et d'avoir une meilleure lisibilité des messages d'erreur.

Angular

Quelques outils utilisés par **Angular** ($>$ 17)

- **npm** (**n**ode **p**ackage **m**anager) : le gestionnaire de paquets par défaut pour une application **JavaScript**.
- **angular-cli** **c**ommand **l**ine **i**nterface : outil proposé par **Google** pour faciliter la création et la construction d'une application **Angular** en exécutant directement des commandes.
- **application** : un nouveau builder introduit, utilisant **Vite** avec **esbuild** pour la construction et servir en mode **dev** et **Rollup** pour les bundles finaux.
- **Ivy** : moteur de compilation et de rendu utilisé partiellement dans **Angular** 8, intégralement depuis la version 9. Il permet d'accélérer la compilation et d'avoir une meilleure lisibilité des messages d'erreur.

Angular

Vite

- Outil de développement web léger et rapide
- Générateur de projet front-end
- Créé par **Evan you** (Créateur de **Vue.js**)
- Conçu, initialement, pour accélérer le processus de développement des applications **Vue.js**
- Open-source
- Prenant en charge les langages **JavaScript** et **TypeScript**
- Intégrant plusieurs frameworks et librairies : **React.js**, **Vue.js**, **Vanilla**, **Svelte**...

Angular

Relation avec esbuild Rollup

- **Vite** est un outil de build moderne pour le développement frontend.
- Il utilise deux moteurs :
  - **esbuild** pour le développement (serveur rapide, transpilation JS/TS).
  - **Rollup** pour le build de production.
- **Vite** agit comme une surcouche de haut niveau qui cache la complexité de **Rollup**.
- **Rollup** gère la génération des bundles finaux, avec un système de plugins très modulable.

Résumé

Vite = `esbuild (dev)` + `Rollup (build prod)`

Angular

Les différentes versions d'**Angular**

- **Angular** 1 (ou **AngularJS**) sorti en 2009 : utilisant **JavaScript**
- **Angular** 2 sorti en octobre 2014 : remplacement du **JavaScript** par **TypeScript**
- **Angular** 4 sorti en décembre 2016
- **Angular** 5 sorti en novembre 2017
- **Angular** 6 sorti en mai 2018
- **Angular** 7 sorti en octobre 2018
- **Angular** 8 sorti en mai 2019
- **Angular** 9 sorti en février 2020
- **Angular** 10 sorti en juin 2020
- **Angular** 11 sorti en novembre 2020
- **Angular** 12 sorti en mai 2021
- **Angular** 13 sorti en novembre 2021
- **Angular** 14 sorti en juin 2022
- **Angular** 15 sorti en novembre 2022
- **Angular** 16 sorti en mai 2023
- **Angular** 17 sorti en novembre 2023
- **Angular** 18 sorti en mai 2024
- **Angular** 19 sorti en novembre 2024
- **Angular** 20 sorti en mai 2025
- **Angular** 21 prévue pour le 17 novembre 2025

Angular

Quelques sites Web réalisés avec **Angular**

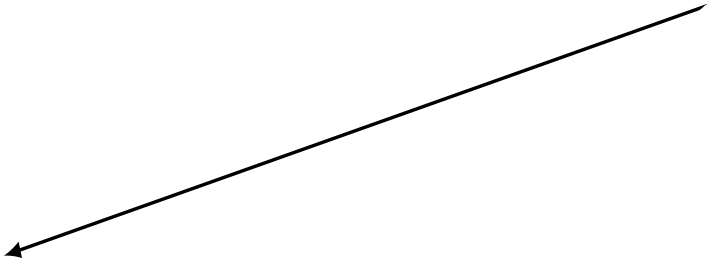
`https://www.madewithangular.com/`

Angular

4 types de fichiers

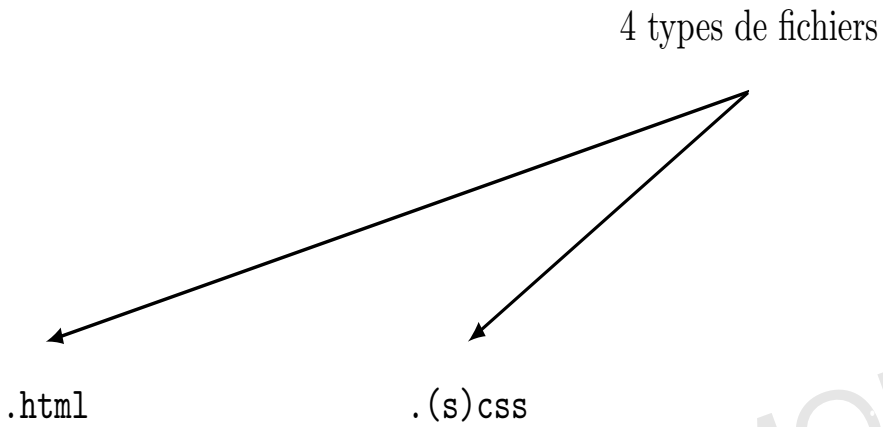
Angular

4 types de fichiers

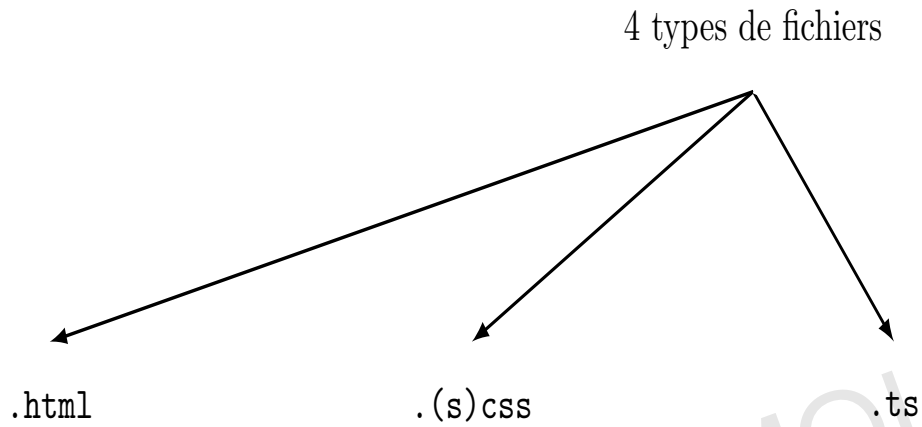


`.html`

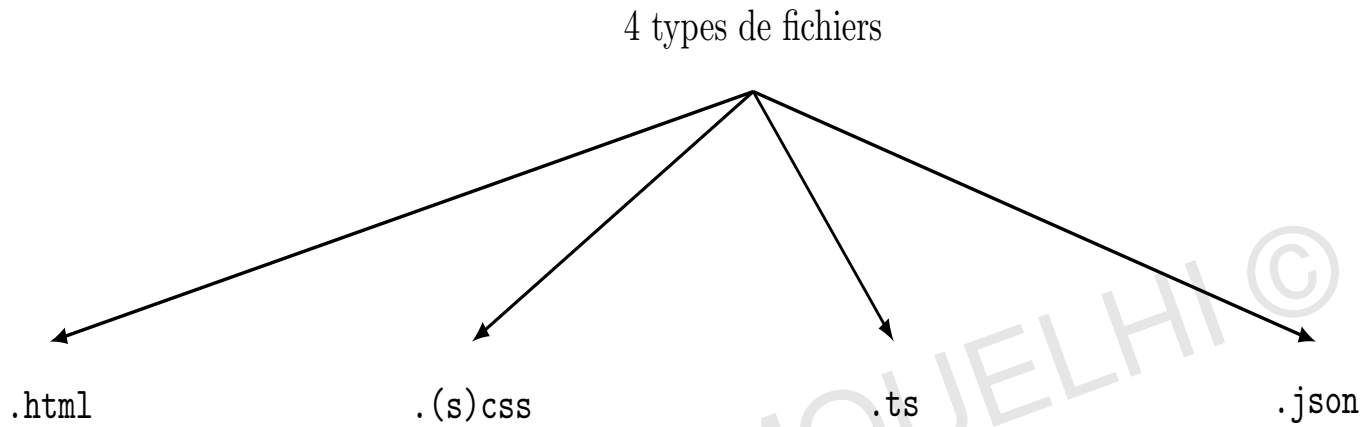
Angular



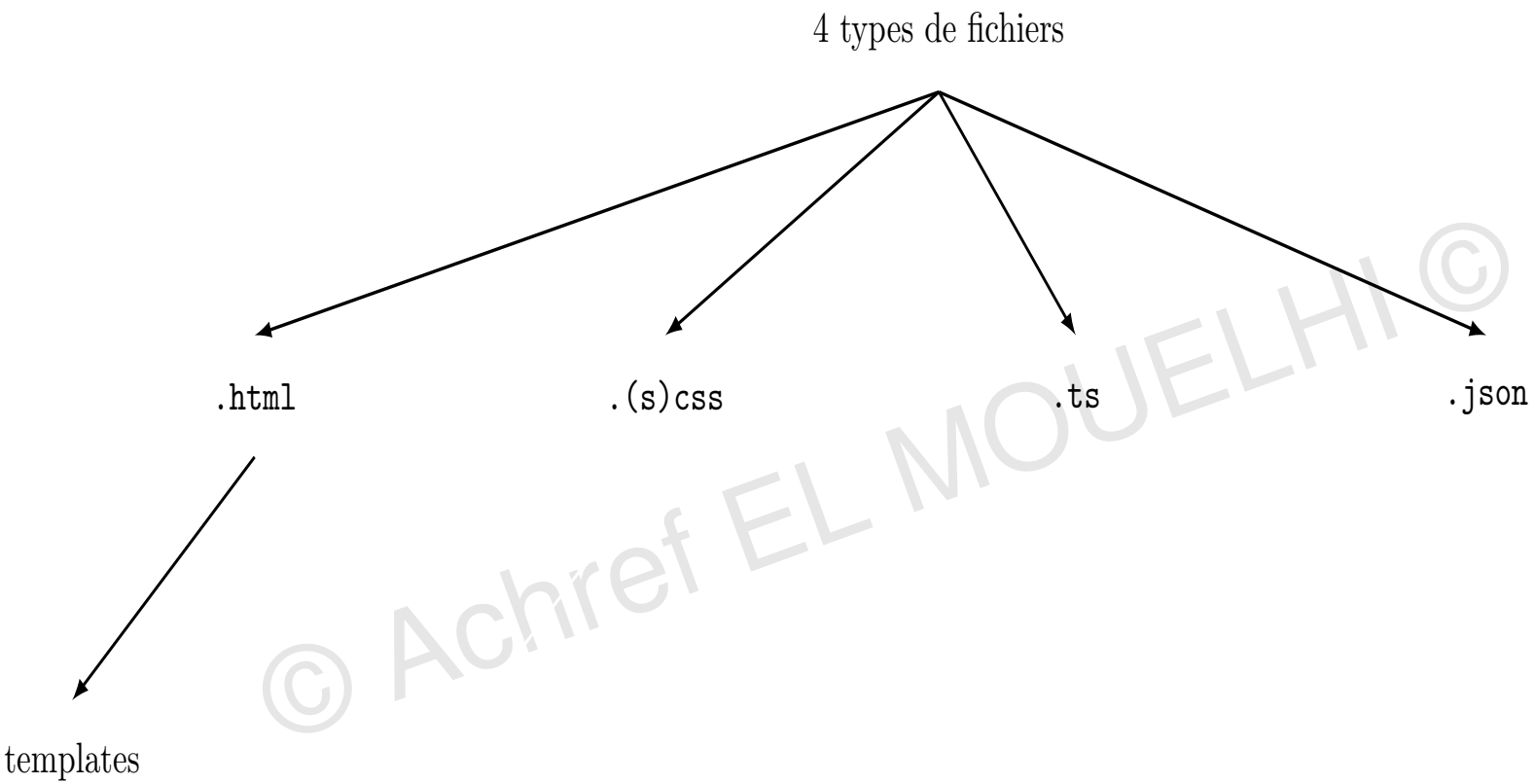
Angular



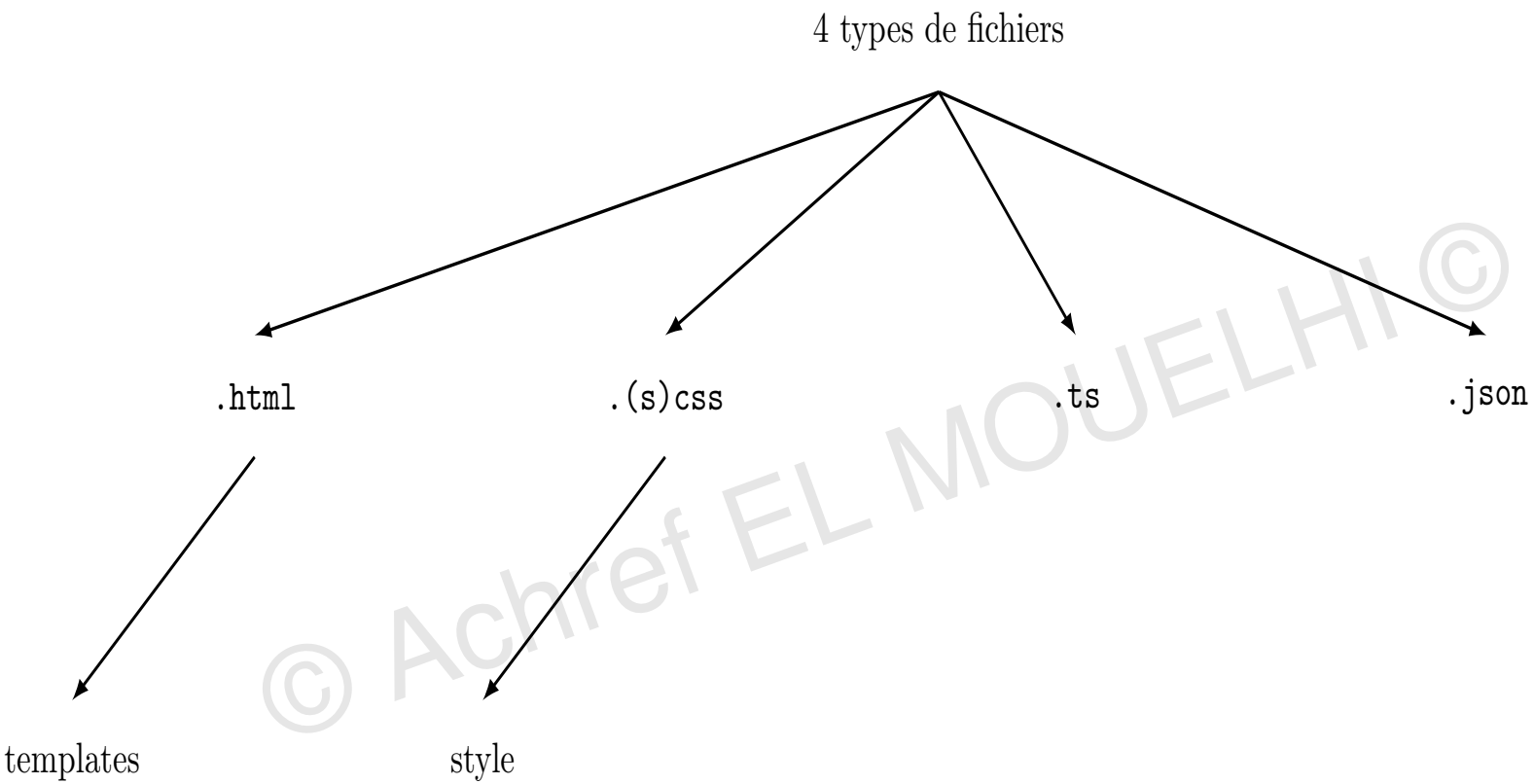
Angular



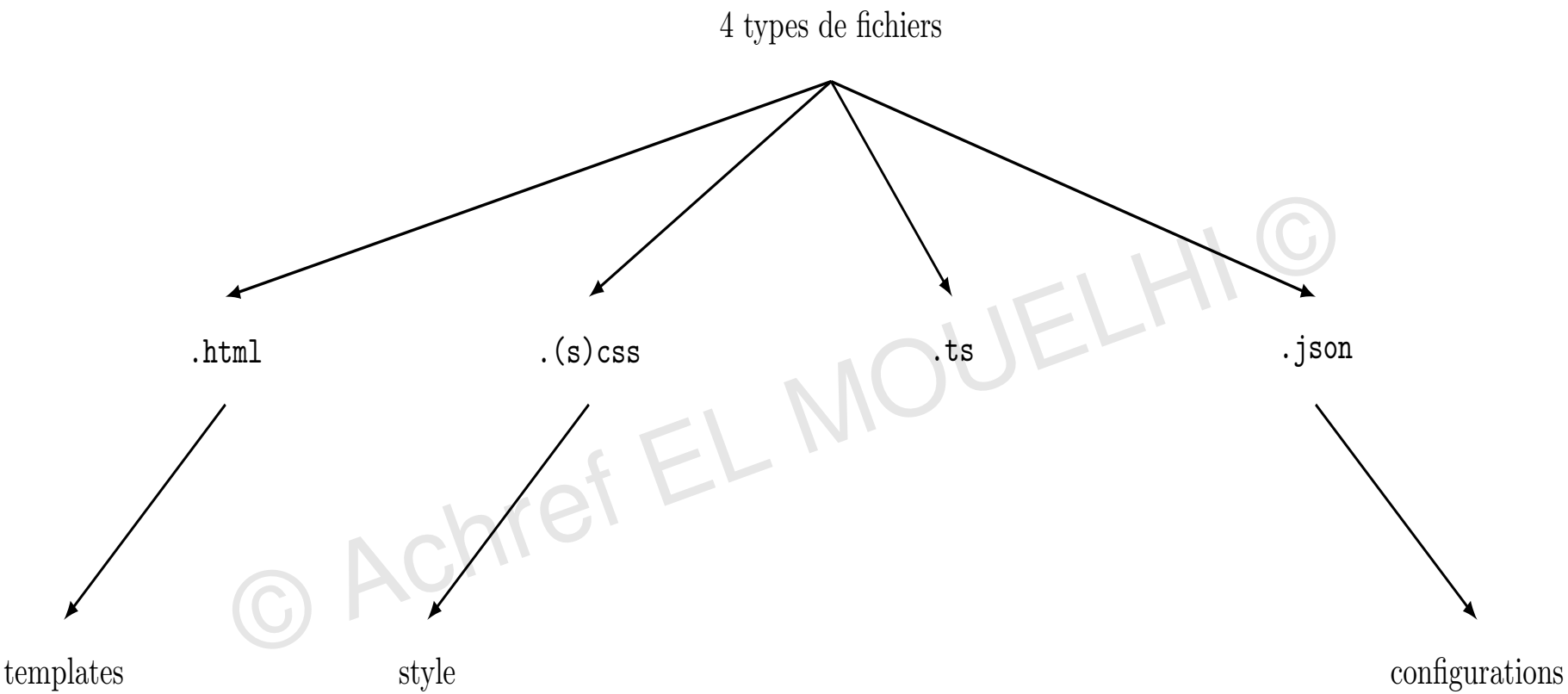
Angular



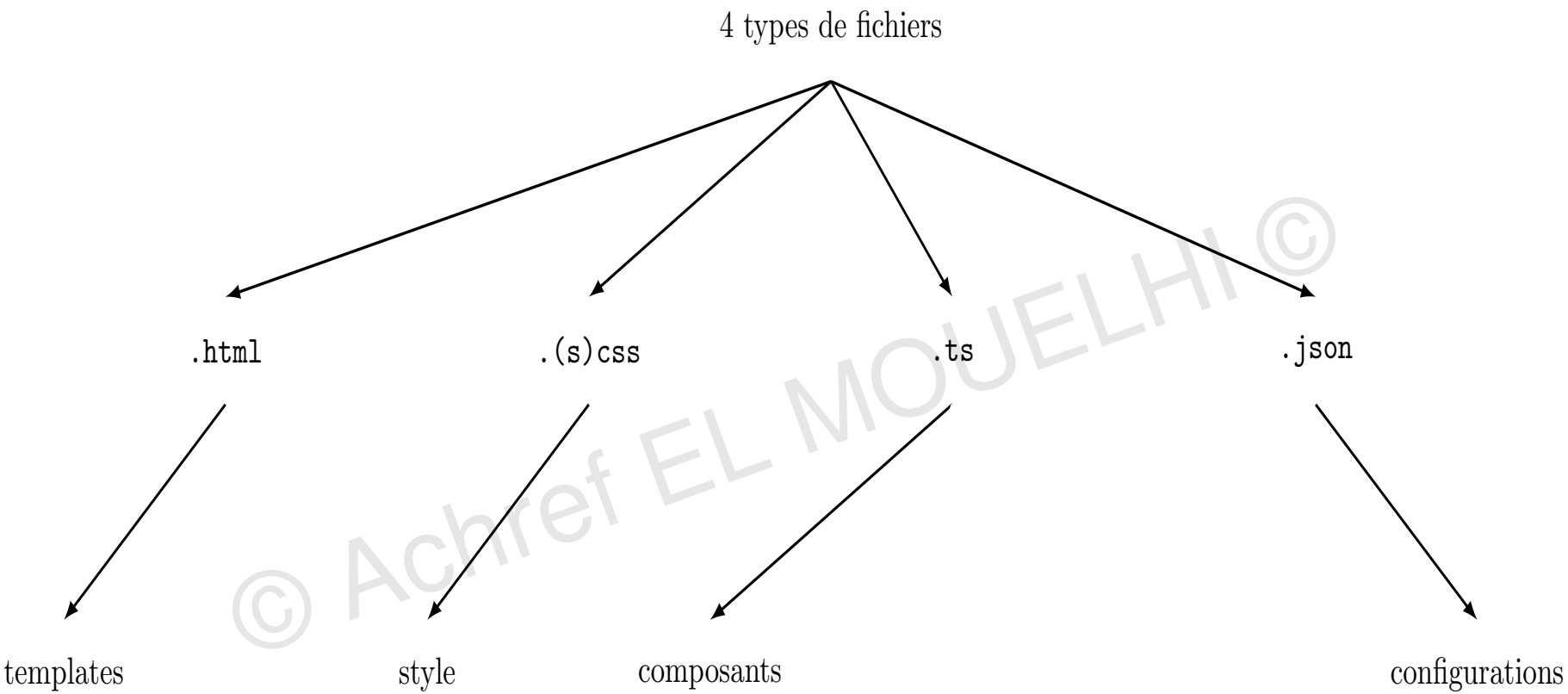
Angular



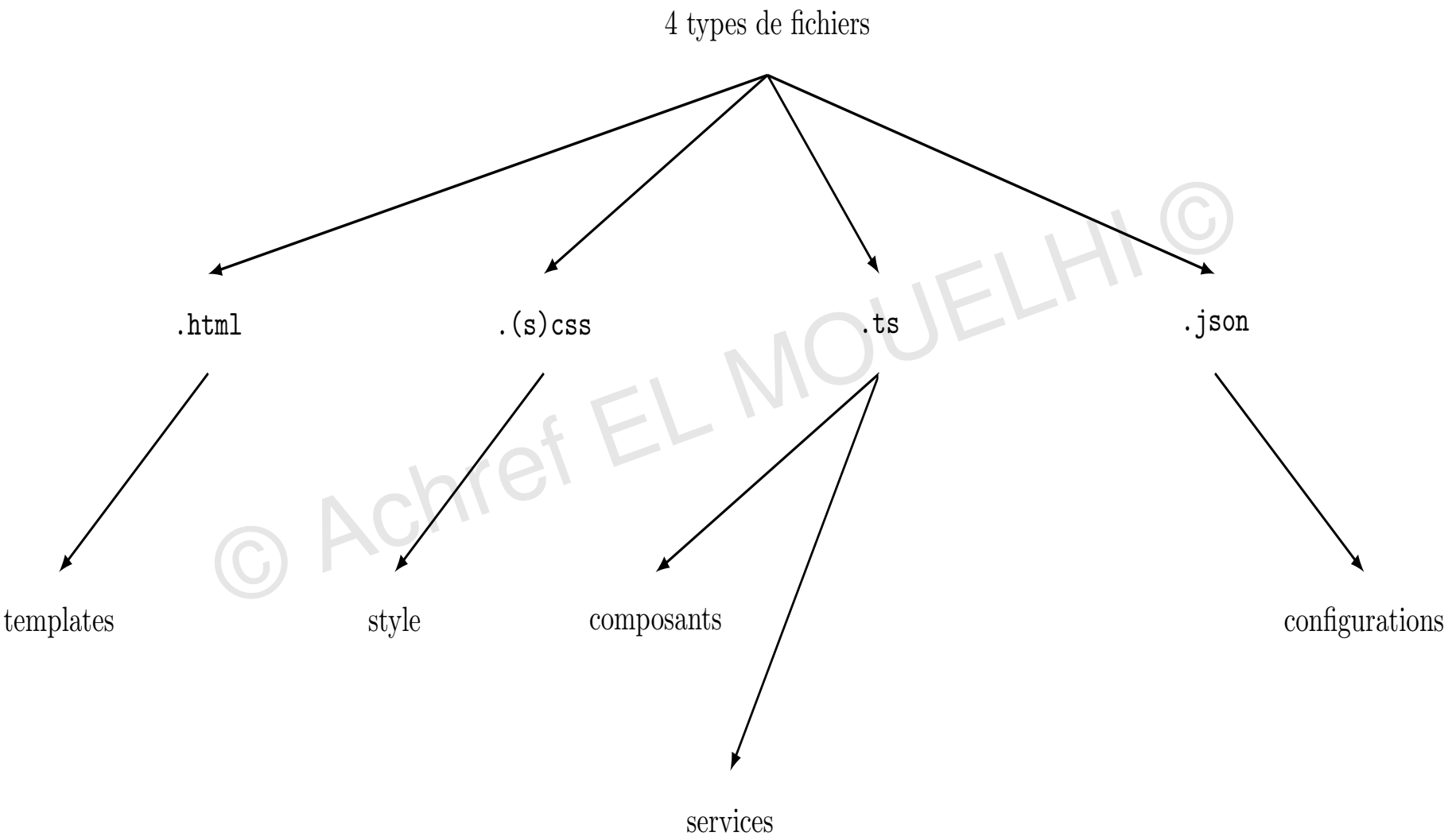
Angular



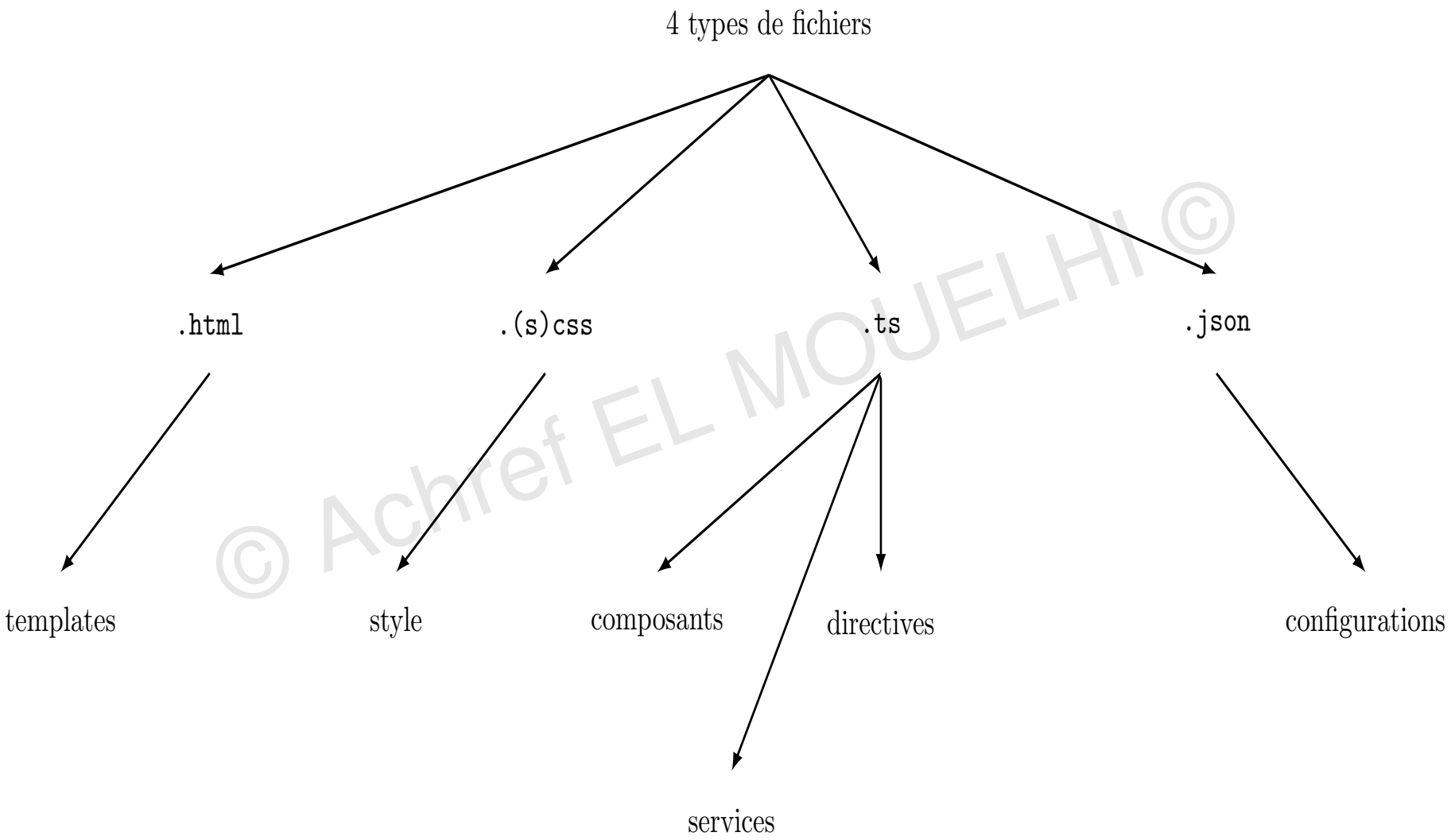
Angular



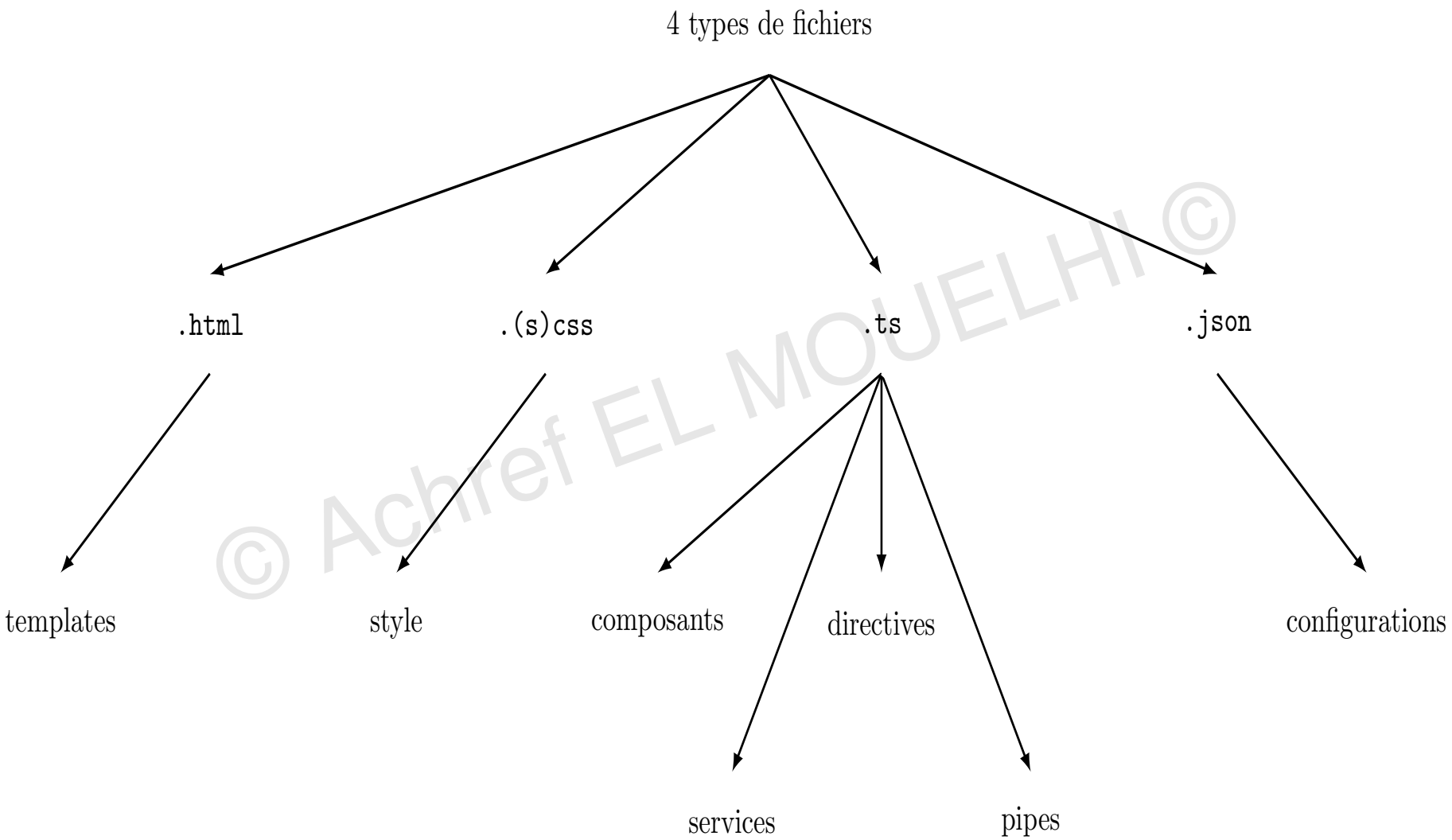
Angular



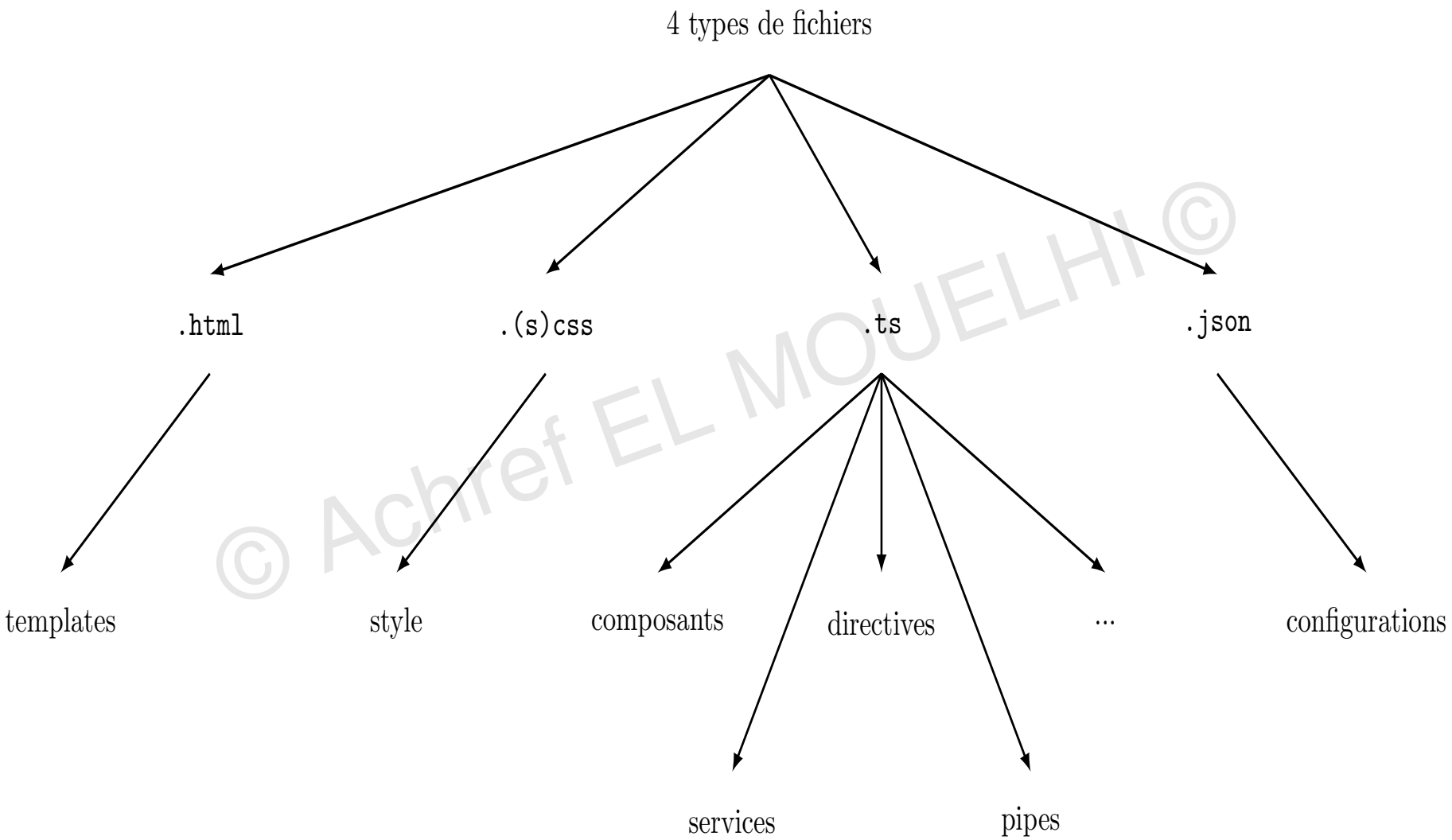
Angular



Angular



Angular



Angular

Remarques

- Le code **TypeScript** est compilé et remplacé par un code **JavaScript**.
- En revanche, **Angular** propose deux types de compilation pour les templates.

Angular

Angular : deux types de compilation

- **JIT** : **J**ust **i**n **T**ime
- **AOT** : **A**head **o**f **T**ime

Angular

JIT : Just in Time

- Chaque fichier est compilé séparément.
- Chargement lent à cause de la compilation avant chaque exécution par le navigateur.
- Compilateur d'**Angular** embarqué pour recompiler avant chaque re-exécution.
- Fichiers (bundles) générés plus volumineux à cause de l'embarquement du compilateur.
- Utilisable via les commandes `ng build` ou `ng serve`
- Plus adapté au mode développement.

AOT : Ahead of Time

- Tout le code est compilé ensemble, en insérant HTML/CSS dans les scripts.
- Chargement rapide car tout est compilé à la construction de l'application.
- Pas besoin d'embarquer le compilateur d'**Angular**.
- Fichiers (bundles) générés moins volumineux (pas de compilateur embarqué).
- Utilisable via les commandes `ng build -aot` ou `ng serve -aot`
- Plus adapté au mode production.
- Plus sécurisée, code-source original non-divulgué.

Angular

Quel IDE (Environnement de développement intégré) ?

- **Visual Studio Code** (À ne pas confondre avec Visual Studio)
- Eclipse
- ...

Angular

Quel IDE (Environnement de développement intégré) ?

- **Visual Studio Code** (À ne pas confondre avec Visual Studio)
- Eclipse
- ...

Visual Studio Code (ou VSC) , pourquoi ?

- Gratuit.
- Offrant la possibilité d'intégrer des éditeurs de texte connus (comme **Sublime Text**, **Atom**...).
- Extensible selon le langage de programmation.
- Recommandé par **Microsoft** (créateur de **TypeScript**) et **Google** pour les projets **Angular**.

Angular

Quelques raccourcis **VSC**

- Pour activer la sauvegarde automatique : aller dans `File > AutoSave`
- Pour indenter son code : `Shift` `Alt` `f`
- Pour commenter/décommenter : `Ctrl` `/` (**Windows/Linux**), `Cmd` `/` (**Mac**)
- Pour sélectionner toutes les occurrences : `Ctrl` `f2`
- Pour sélectionner l'occurrence suivante : `Ctrl` `d`
- Pour placer le curseur dans plusieurs endroits différents : `Alt`

Angular

Extension VSC pour les templates **Angular**

- **Angular Language Service**
- **Angular Snippets**

Angular

Devtools : extension **Angular** pour les navigateurs

- **Google Chrome** : `https://chrome.google.com/webstore/detail/angular-devtools/ienfalfjdbdpebioblfackkekamfmbnh`
- **Mozilla Firefox** : `https://addons.mozilla.org/en-GB/firefox/addon/angular-devtools/`

Angular

Remarque

Pour installer **Angular**, il faut télécharger et installer **Node.js** (Dernière version stable LTS)

Angular

Remarque

Pour installer **Angular**, il faut télécharger et installer **Node.js** (Dernière version stable LTS)

Pour installer Angular, exécuter la commande

```
npm install -g @angular/cli
```

Angular

Remarque

Pour installer **Angular**, il faut télécharger et installer **Node.js** (Dernière version stable LTS)

Pour installer Angular, exécuter la commande

```
npm install -g @angular/cli
```

Pour installer une version bien précise (par exemple 6.1.0)

```
npm install -g @angular/cli@6.1.0
```

Angular

Pour vérifier la version d'Angular installée, exécuter la commande

```
ng version
```

Angular

Pour vérifier la version d'Angular installée, exécuter la commande

```
ng version
```

Pour explorer la liste des commandes Angular, exécuter la commande

```
ng
```

Angular

Pour créer un nouveau projet Angular

```
ng new cours-angular
```

Angular

Pour créer un nouveau projet Angular

```
ng new cours-angular
```

Ou le raccourci

```
ng n cours-angular
```

Angular

Pour créer un nouveau projet Angular

```
ng new cours-angular
```

Ou le raccourci

```
ng n cours-angular
```


Il est recommandé de créer une nouvelle application sans installation globale

```
npx @angular/cli new projet cours-angular
```

Angular

Depuis la version 7, il faut aussi répondre aux questions suivantes

- Would you like to add Angular routing ? (**Yes**)
- Which stylesheet format would you like to use ? (**CSS**)

Angular

Depuis la version 7, il faut aussi répondre aux questions suivantes

- Would you like to add Angular routing ? (**Yes**)
- Which stylesheet format would you like to use ? (**CSS**)

Pour créer un nouveau projet Angular et éviter la première question

```
ng new cours-angular -routing
```

Angular

Depuis la version 7, il faut aussi répondre aux questions suivantes

- Would you like to add Angular routing ? (**Yes**)
- Which stylesheet format would you like to use ? (**CSS**)

Pour créer un nouveau projet Angular et éviter la première question

```
ng new cours-angular -routing
```

Pour créer un nouveau projet Angular et garder les réponses par défaut pour les deux questions

```
ng new cours-angular -defaults
```

Angular

Pour créer un projet sans les fichiers de test

```
ng n cours-angular -skip-tests
```

Angular

Pour créer un projet sans les fichiers de test

```
ng n cours-angular -skip-tests
```

ou aussi

```
ng n cours-angular -S
```

Angular

Pour créer un projet sans les fichiers `git`

```
ng n cours-angular -skip-git
```

Angular

Pour créer un projet sans les fichiers `git`

```
ng n cours-angular -skip-git
```

ou aussi

```
ng n cours-angular -g
```

Angular

Depuis la version 19

- Les projets crées sont par défaut **standalone**.
- Les modules (`NgModule`) restent supportés pour compatibilité.

Angular

Depuis la version 19

- Les projets crées sont par défaut **standalone**.
- Les modules (`NgModule`) restent supportés pour compatibilité.

Pour créer un projet avec la gestion de modules

```
ng new cours-angular -standalone=false
```

Angular

Autres options

- `-inline-template`
  - Permet d'inclure directement le template **HTML** dans le fichier **TypeScript** du composant.
  - Utile pour les composants simples où le code HTML est court.
- `-inline-style`
  - Permet d'inclure directement les styles **CSS** dans le fichier **TypeScript** du composant.
  - Réduit le nombre de fichiers créés par composant.
- `-minimal`
  - Crée un projet minimal avec uniquement les fichiers essentiels pour démarrer.
  - Supprime les fichiers et configurations non indispensables.
- `-skip-install`
  - Empêche l'exécution de la commande `npm install` après la création du projet.
  - Utile si vous souhaitez installer les dépendances plus tard ou utiliser un gestionnaire de paquets différent.

Angular

Pour lancer le projet, exécuter la commande (depuis la racine du projet)

```
ng serve
```

Angular

Pour lancer le projet, exécuter la commande (depuis la racine du projet)

```
ng serve
```

Ou le raccourci

```
ng s
```

Angular

Pour lancer le projet et ouvrir une nouvelle fenêtre dans le navigateur, exécuter la commande

```
ng s -open
```

Angular

Pour lancer le projet et ouvrir une nouvelle fenêtre dans le navigateur, exécuter la commande

```
ng s -open
```

Ou

```
ng s -o
```

Angular

Pour lancer le projet et ouvrir une nouvelle fenêtre dans le navigateur, exécuter la commande

```
ng s -open
```

Ou

```
ng s -o
```


On peut aussi lancer un projet Angular comme tout projet NodeJS, exécuter la commande

```
npm start
```

Angular

Lien vers la documentation officielle d'**Angular**

`https://angular.dev/cli`

Angular

Arborescence d'un projet **Angular**

- `node_modules` : contenant les modules **Node.js** nécessaire pour un projet **Angular**
- `public` : unique dossier accessible aux visiteurs et contenant les images, les sons...
- `src` : contenant les fichiers sources de l'application
- `package.json` : contenant l'ensemble de dépendance de l'application
- `angular.json` : contenant les données concernant la configuration du projet (l'emplacement des fichiers de démarrage...)
- `.editorconfig` : utilisé pour définir les styles de formatage de code appliqués à tous les fichiers du projet qui sont ouverts dans un éditeur supportant ce fichier de configuration (comme **VSC**)
- `tsconfig.json` : spécifie les options de compilation **TypeScript** globales pour tout le projet
- `tsconfig.app.json` : spécifie les options de compilation **TypeScript** globales pour une application particulière
- `tsconfig.spec.json` : spécifie les options de compilation **TypeScript** pour les tests unitaires (fichiers `*.spec.ts`).

Angular

`tsconfig.app.json` VS `tsconfig.json`

- Rien ne nous empêche de supprimer `tsconfig.app.json`. c'est juste un fichier de configuration supplémentaire qui permet d'ajuster la configuration en fonction de l'application.
- Utile lorsque nous avons plusieurs applications dans le même projet.
- Il est possible d'avoir
  - le dossier racine avec `tsconfig.json`
  - un sous-dossier `app-a` avec un fichier `tsconfig.app.json` dans `app-a`
  - un sous-dossier `app-b` d'une autre application avec son propre `tsconfig.app.json`

Que contient `src` ?

- `index.html` : unique fichier **HTML** d'une application **Angular**
- `styles.css` : feuille de style commune de tous les composants web de l'application
- `favicon.ico` : icône d'**Angular**
- `main.ts` : fichier qui permet le chargement du module principal (`app`)
- `app` : contient initialement les 5 fichiers du module principal
  - `app.ts` : classe associé au composant web
  - `app.html` : contenant le code **HTML** associé au composant web
  - `app.css` : contenant le code CSS associé au composant web
  - `app.spec.ts` : contenant le code de test du composant web
  - `app.routes.ts` : contenant les route
  - `app.config.ts` : fichier utilisé pour centraliser la configuration de l'application

Angular

Contenu d'`index.html`

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>CoursAngular</title>

  <!-- cette balise va permettre d'assurer le routage -->
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>

  <!-- le composant web introduit dans le module principal -->
  <app-root></app-root>
</body>
</html>
```

Contenu de `app.config.ts`

```
import { routes } from './app.routes';

export const appConfig: ApplicationConfig = {
  providers: [
    provideBrowserGlobalErrorListeners(),
    provideZoneChangeDetection({ eventCoalescing: true }),
    provideRouter(routes)
  ]
};
```

Contenu de `app.config.ts`

```
import { routes } from './app.routes';

export const appConfig: ApplicationConfig = {
  providers: [
    provideBrowserGlobalErrorListeners(),
    provideZoneChangeDetection({ eventCoalescing: true }),
    provideRouter(routes)
  ]
};
```

Explication

- `providers: [ ... ]` : permet de définir une liste de services (ou providers) que l'application utilisera. Les services mentionnés ici seront injectés dans les composants ou d'autres services en fonction des besoins de l'application.
- `provideBrowserGlobalErrorListeners` : activer la capture globale des erreurs **JavaScript** non gérées et les redirige vers le système d'erreurs **Angular** (service `ErrorHandler`)
- `provideZoneChangeDetection` : Configure le mécanisme de détection de changements basé sur `Zone.js`
- `{ eventCoalescing: true }` : permet de regrouper plusieurs événements dans un seul cycle de détection.
- `Zone.js` : la librairie qui intercepte les événements (clic, timeout, promesse...) pour notifier **Angular** et déclencher le cycle de détection de changements.
- `provideRouter(routes)` : fournit le routeur avec les routes de l'application.

Angular

Décorateur du composant

- `@Component` : pour déclarer cette classe comme composant web
- `selector` : pour définir le nom de balise correspondant à ce composant web
- `templateUrl` : pour indiquer le fichier **HTML** correspondant au composant web
- `styleUrls` : pour indiquer le(s) fichier(s) **CSS** correspondant au composant web
- `standalone` (par défaut à `true`) : fonctionnalité relativement récente d'**Angular** 14 qui permet de créer des composants sans dépendre de modules **Angular**
- `imports` : pour spécifier les composants ou autres fonctionnalités **Angular** dont ce composant a besoin.

Angular

Commençons par localiser la section `schematics` dans `cours-angular` (nom du projet) dans `angular.json` et chercher

```
"projects": {
  "cours-angular": {
    "projectType": "application",
    "schematics": {
      "@schematics/angular:application": {
        "strict": true
      },
    },
  },
},
```

Angular

Commençons par localiser la section `schematics` dans `cours-angular` (nom du projet) dans `angular.json` et chercher

```
"projects": {
  "cours-angular": {
    "projectType": "application",
    "schematics": {
      "@schematics/angular:application": {
        "strict": true
      },
    },
  },
},
```

Pour arrêter la génération des fichiers de test pour les composants (par exemple), on ajoute la section suivante

```
"projects": {
  "cours-angular": {
    "projectType": "application",
    "schematics": {
      "@schematics/angular:application": {
        "strict": true
      },
      "@schematics/angular:component": {
        "skipTests": true
      },
    },
  },
},
```

Angular

Remarque

On peut ajouter une section pour chaque élément pour lequel on veut plus générer des fichiers de test : `service`, `directive`, `class`, `pipe`...

Angular

Pour désinstaller Angular

```
npm uninstall -g @angular/cli
```

Angular

Pour désinstaller Angular

```
npm uninstall -g @angular/cli
```

Pour vider le cache (Avant la version 5 de npm)

```
npm cache clean
```

Angular

Pour désinstaller Angular

```
npm uninstall -g @angular/cli
```

Pour vider le cache (Avant la version 5 de npm)

```
npm cache clean
```

Pour vider le cache (depuis la version 5 de npm)

```
npm cache verify
```

Angular

Pour mettre à jour la version d'Angular

```
ng update @angular/cli @angular/core
```

Angular

Pour mettre à jour la version d'Angular

```
ng update @angular/cli @angular/core
```


Pour mettre à jour tous les paquets définis dans `Package.json`

```
ng update -all
```

Angular

Pour faire la migration, consulter

`https://angular.dev/update-guide`

Angular

En cas de problème avec les commandes précédentes, exécutez la commande suivante depuis PowerShell pour permettre à l'utilisateur actuel d'exécuter des scripts locaux sans restriction

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```