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 **JavaScript** (ou **TypeScript** ou **Dart** : langage de **Google**)
- 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 :

- les composants web
- l'injection de dépendance
- le **DOM** Virtuel
- le change detection

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**

- **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

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 prévu en mai 2024

Angular

Quelques sites Web réalisés avec **Angular**

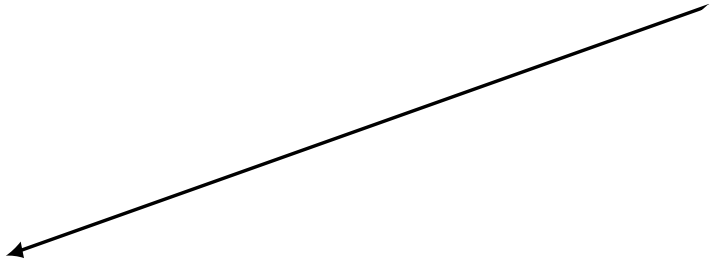
`https://www.madewithangular.com/`

Angular

4 types de fichiers

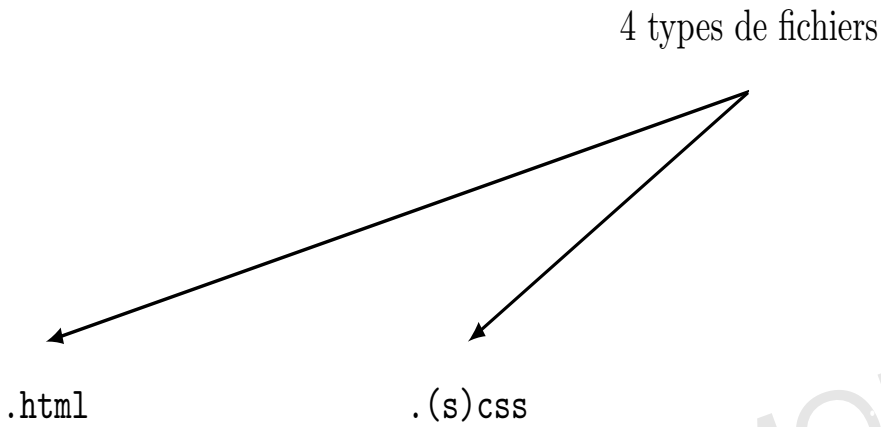
Angular

4 types de fichiers

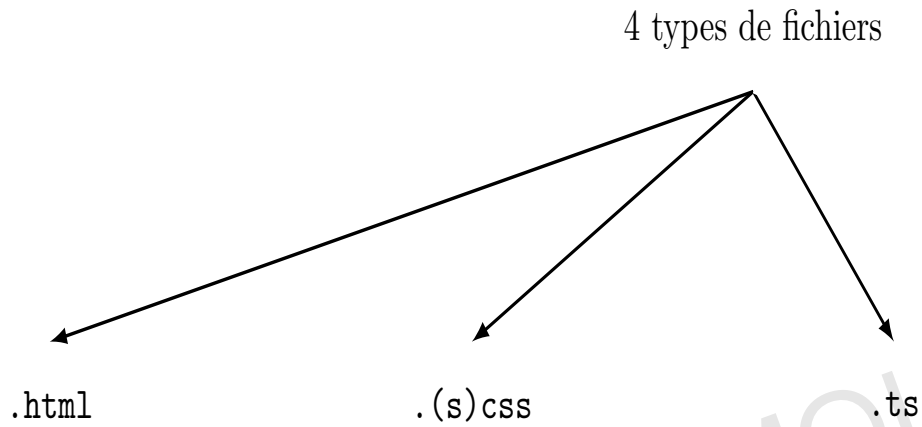


`.html`

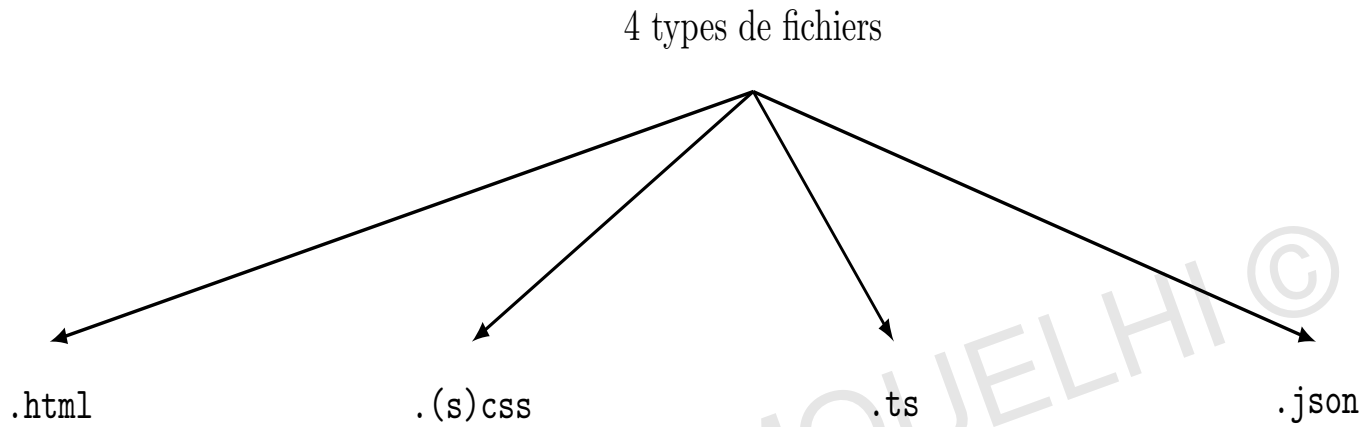
Angular



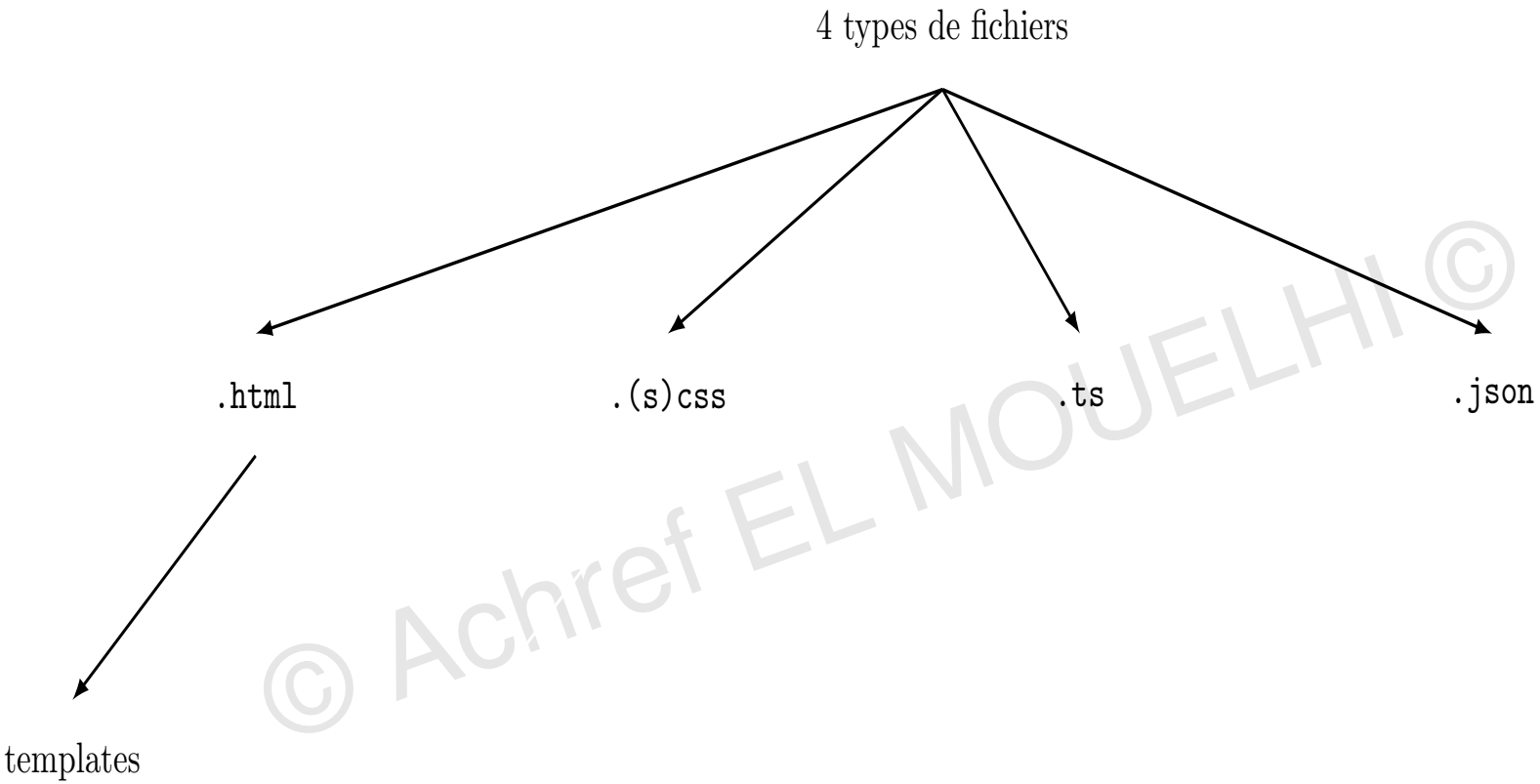
Angular



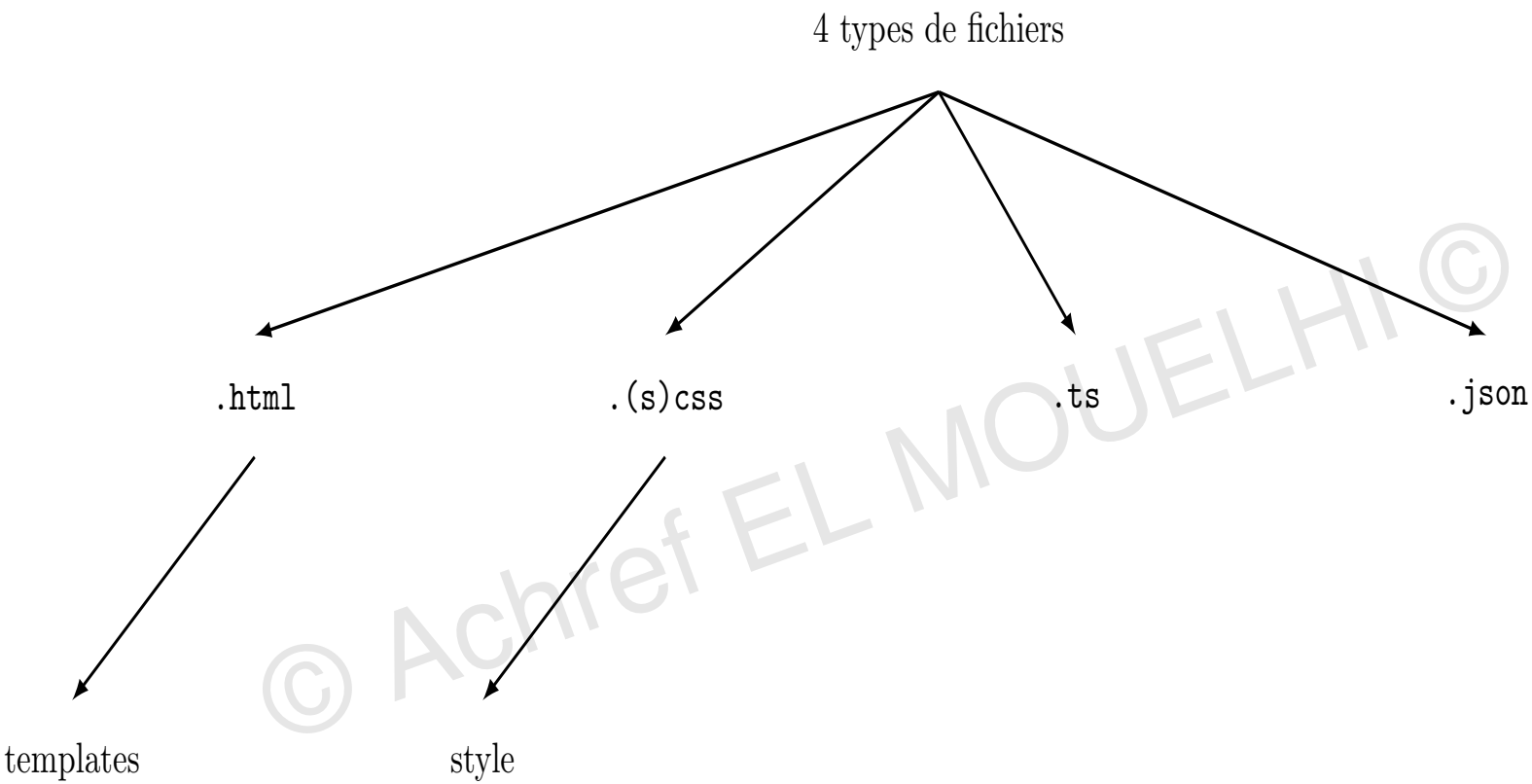
Angular



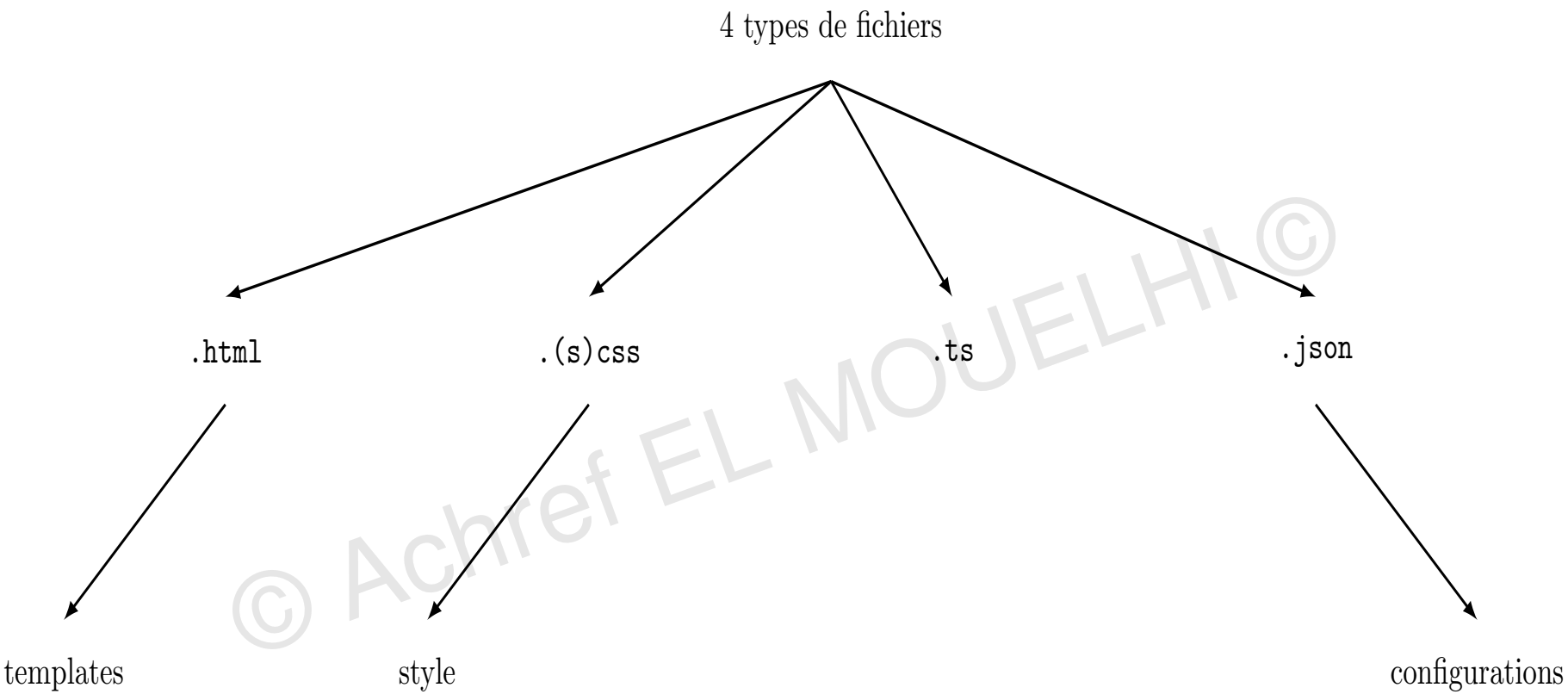
Angular



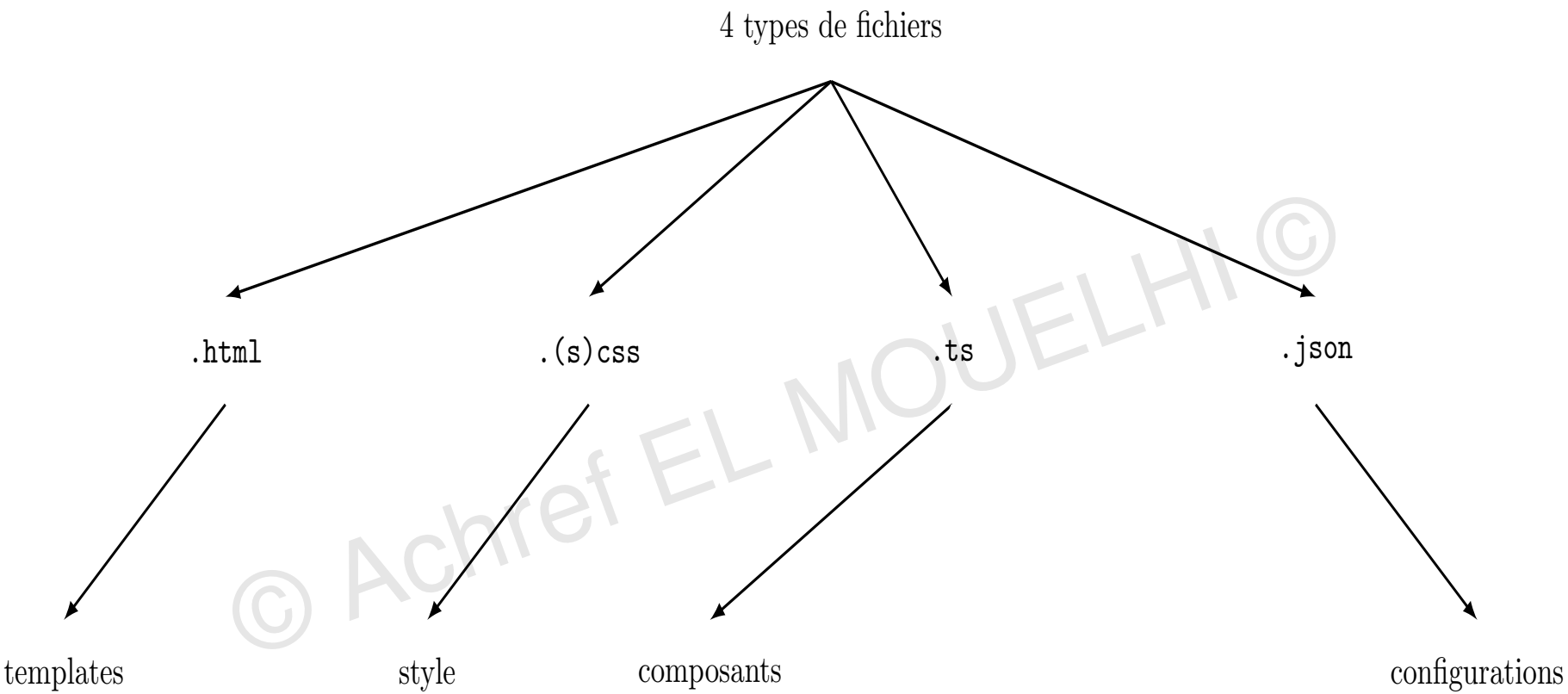
Angular



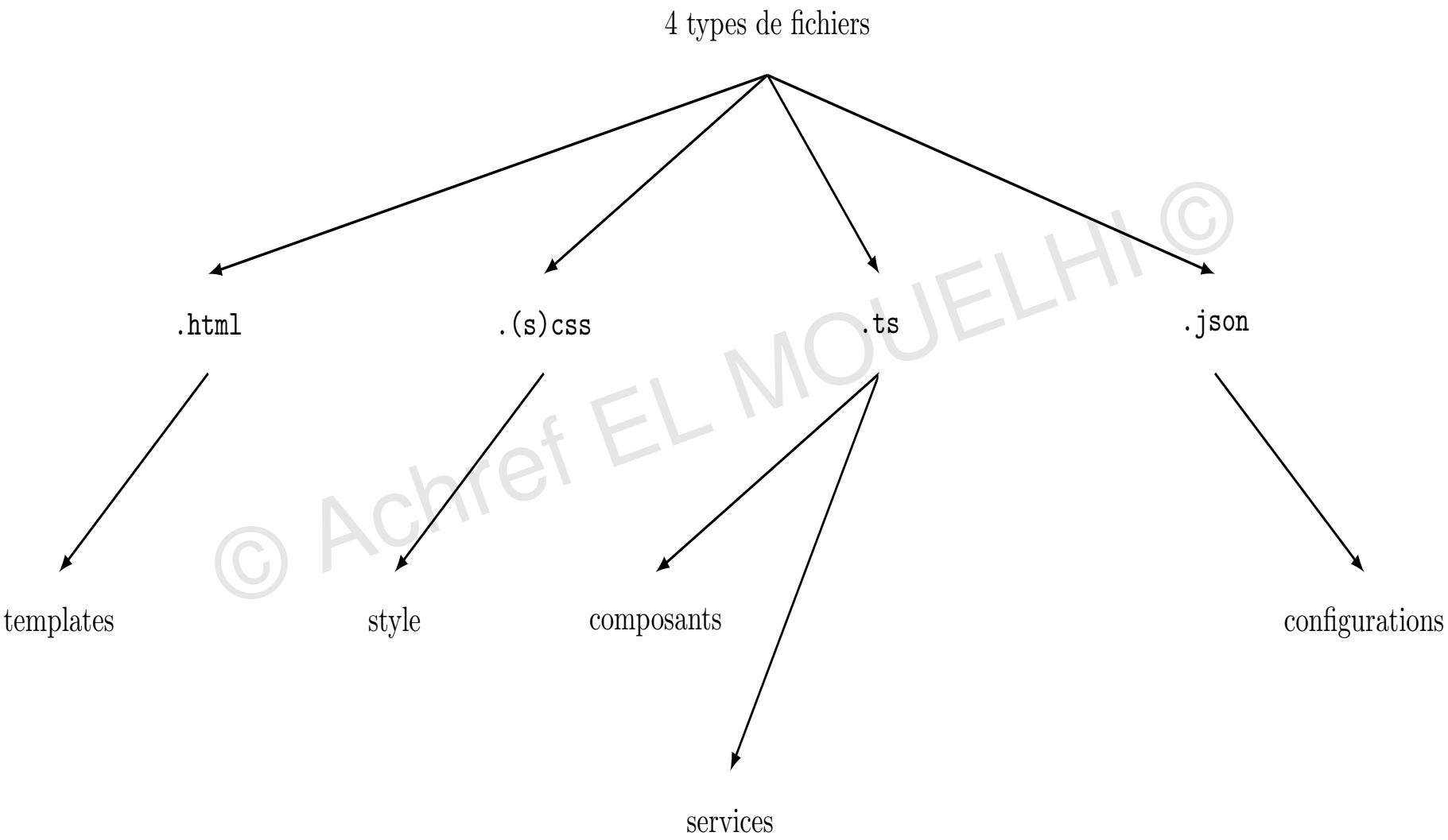
Angular



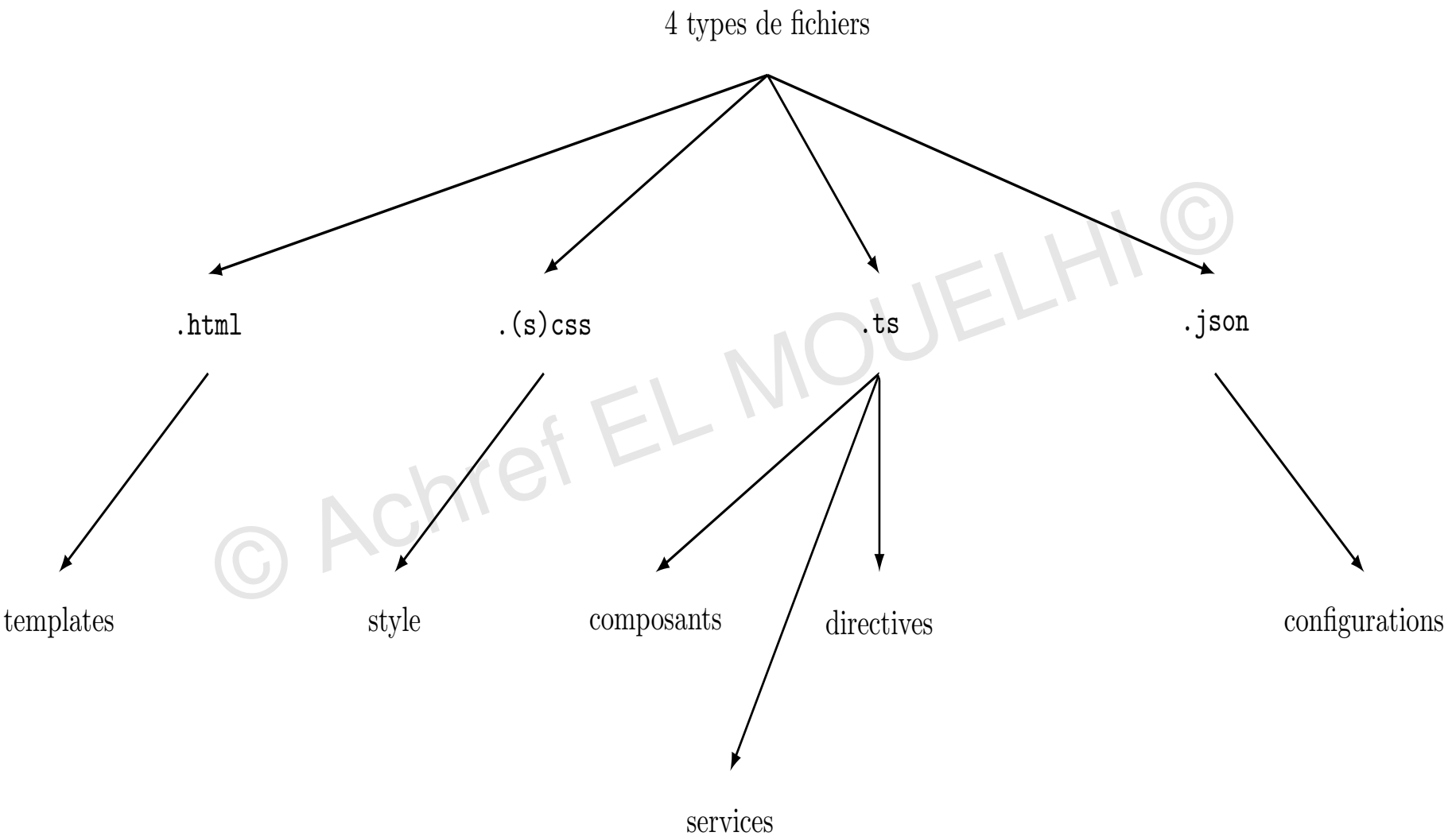
Angular



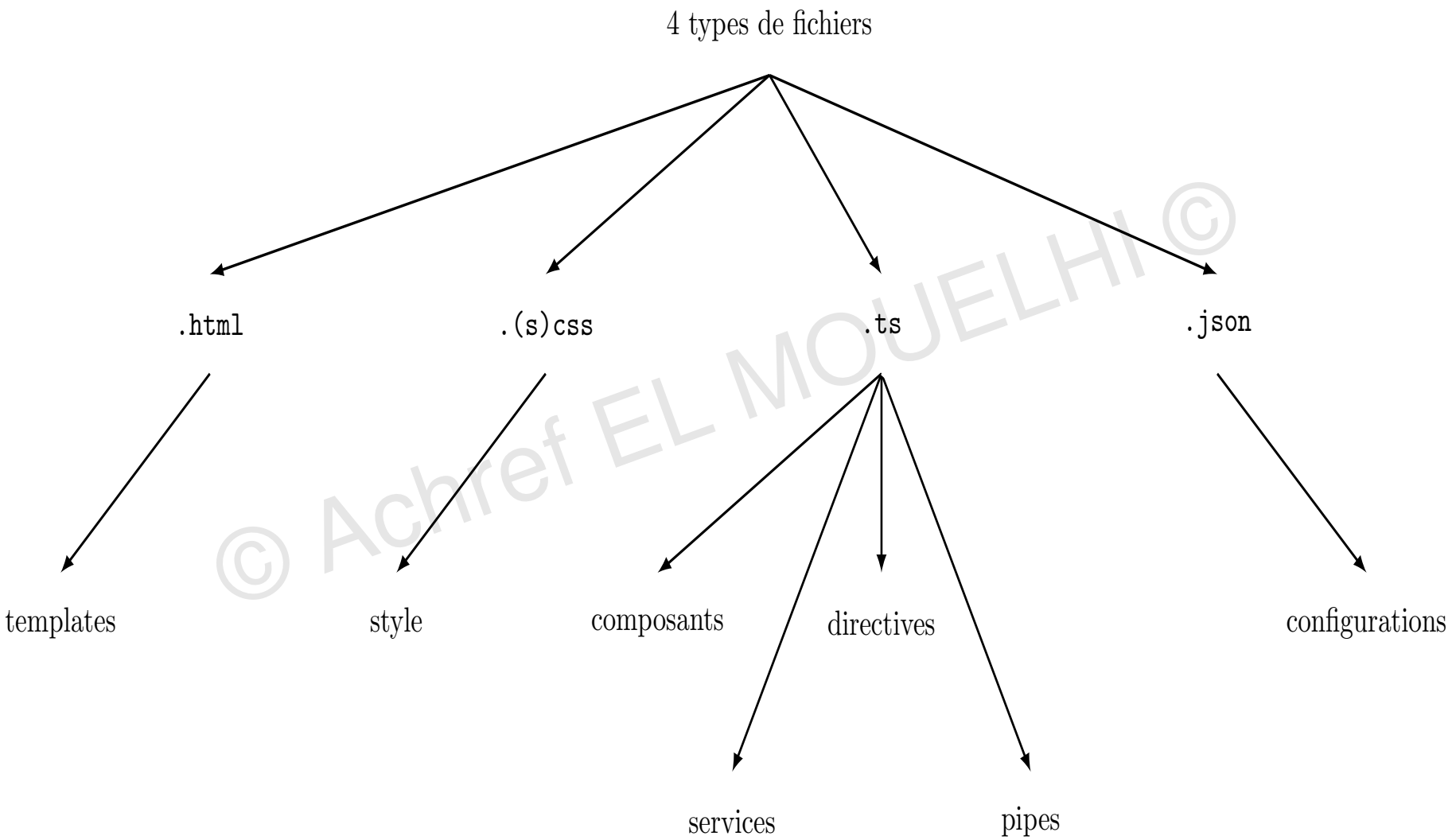
Angular



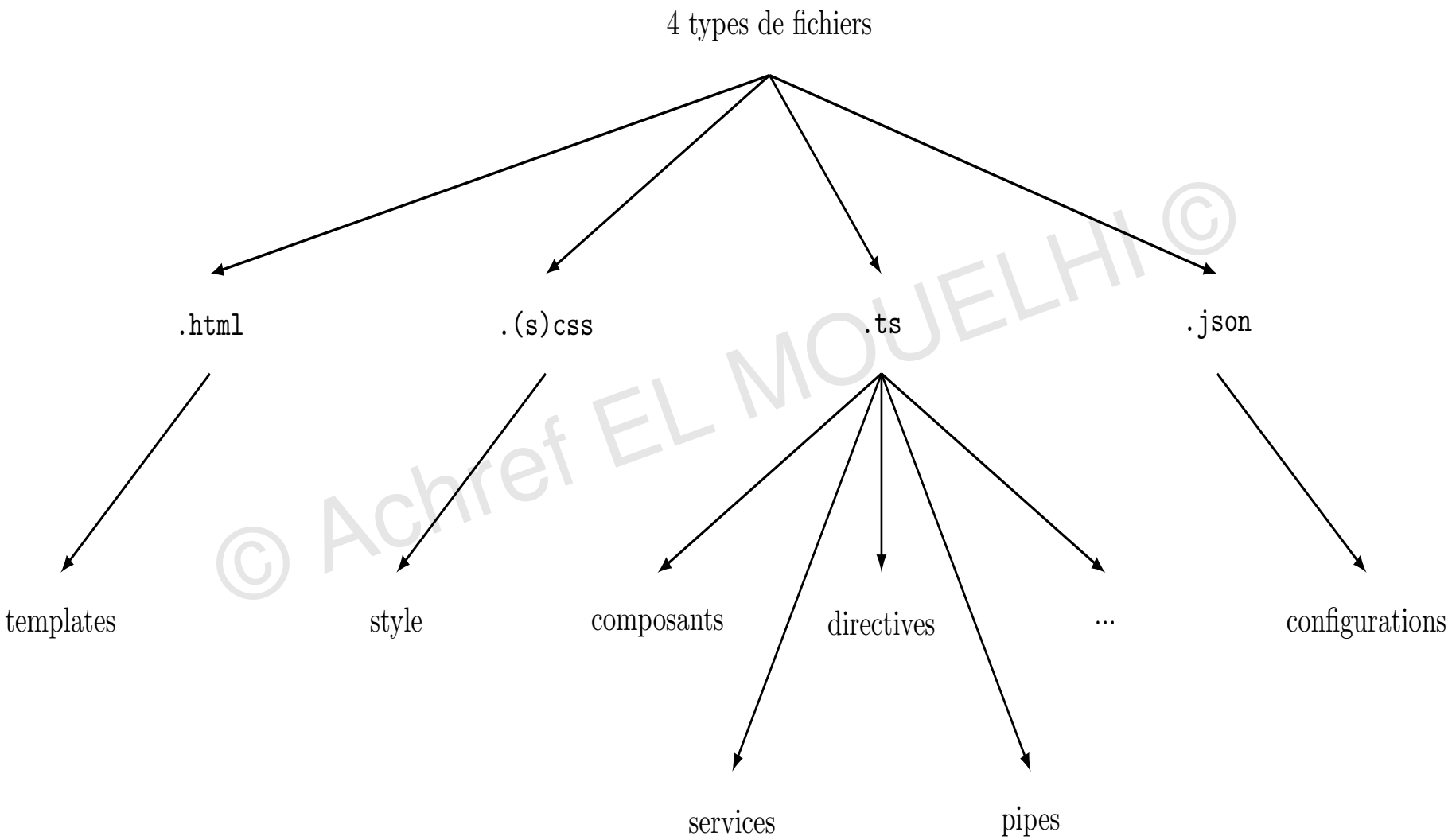
Angular



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

Remarque

Pour installer **Angular**, il faut télécharger et installer **NodeJS** (Dernière version stable LTS)

Angular

Remarque

Pour installer **Angular**, il faut télécharger et installer **NodeJS** (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 **NodeJS** (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

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**) pour les projets **Angular**.

Angular

Quelques raccourcis **VSC**

- Pour activer la sauvegarde automatique : aller dans `File > AutoSave`
- Pour indenter son code : `Alt` `Shift` `f`
- Pour commenter/décommenter : `Ctrl` `:`
- 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

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 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 --skipTests
```

Angular

Pour créer un projet sans les fichiers de test

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

ou aussi

```
ng n cours-angular --skipTests
```

ou encore

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

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 `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 --skipGit
```

Angular

Depuis Angular 16, il est possible de créer un projet sans modules (tout est défini dans les composants)

```
ng new cours-angular --standalone
```

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

Arborescence d'un projet **Angular**

- `node_modules` : contenant les modules **Node.js** nécessaire pour un projet **Angular**
- `src` : contenant les fichiers sources de l'application
- `package.json` : contenant l'ensemble de dépendance de l'application
- `.angular-cli.json` (ou `angular.json` depuis la version 6) : 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` ?

- `assets` : unique dossier accessible aux visiteurs et contenant les images, les sons...
- `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.module.ts` : classe correspondante au module principal
  - `app.component.ts` : classe associé au composant web
  - `app.component.html` : contenant le code **HTML** associé au composant web
  - `app.component.css` : contenant le code CSS associé au composant web
  - `app.component.spec.ts` : contenant le code de test du composant web

Angular

Contenu d'`index.html`

```
<!doctype html>
<html lang="en">
<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.module.ts`

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

Contenu de `app.module.ts`

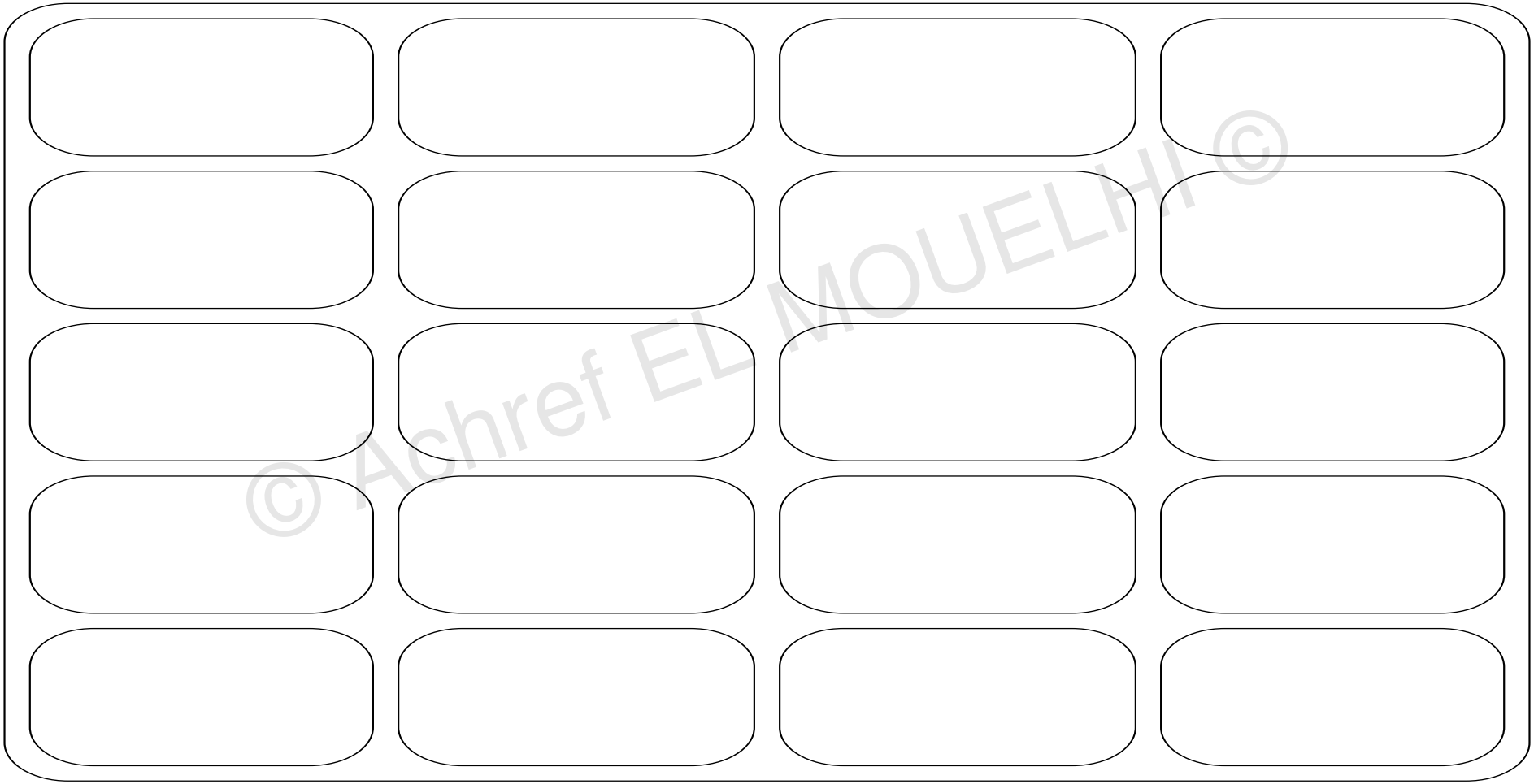
```
@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Explication

- `@NgModule` : pour déclarer cette classe comme module
- `declarations` : dans cette section, on déclare les composants de ce module
- `imports` : dans cette section, on déclare les modules nécessaires pour le module
- `providers` : dans cette section, on déclare les services qui seront utilisés dans le module
- `bootstrap` : dans cette section, on déclare le composant principal de ce module

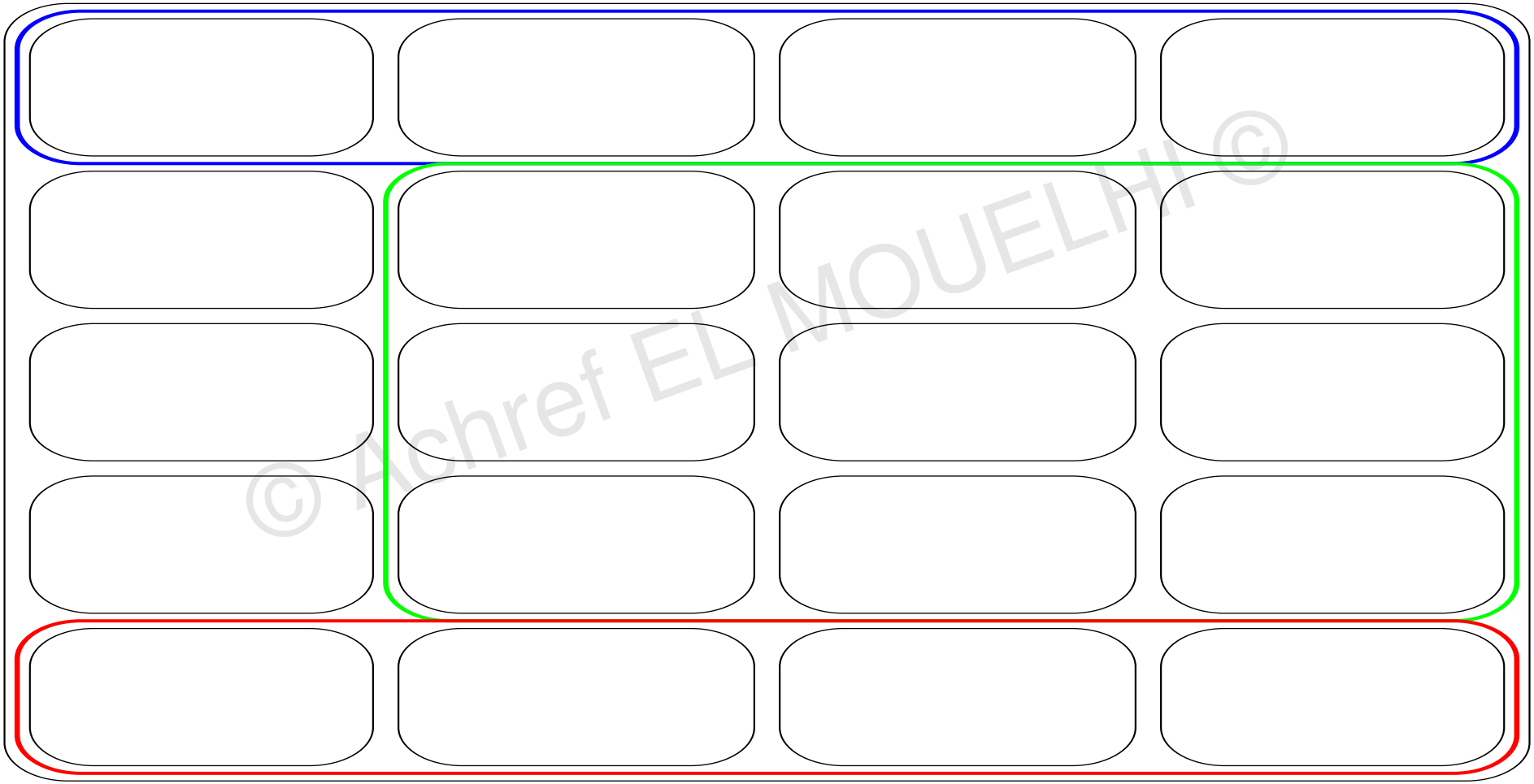
Angular

Angular : les composants



Angular

Angular : les modules



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

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 @angular/cli @angular/core --all=true
```

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
```