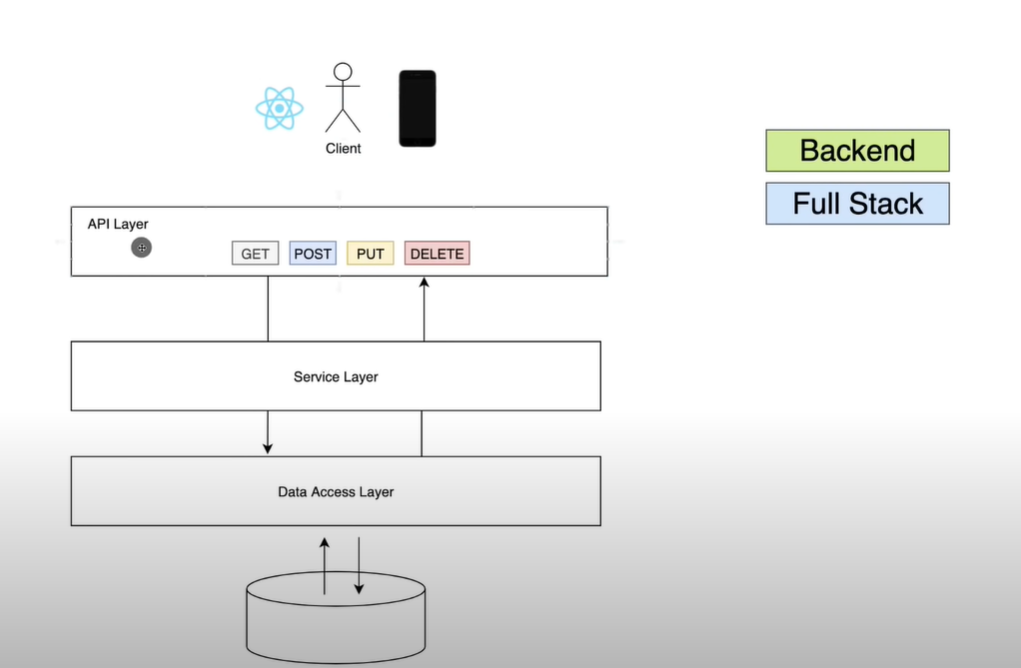
Spring boot: amazing framework that gives you everything you need to build application, logging, security, connecting to data base, configuration, …(module)



Crtl+N = generate

@RequestMapping(path = "api/v1/student")

Instead of localhost:8080 we will have localhost:8080/api/vi/student

Service layer responsible for business logics

@autowired :the framework introduced annotations-driven Dependency Injection. The main annotation of this feature is @Autowired. It **allows Spring to resolve and inject collaborating beans into our bean**.

@Component is a class-level annotation. It is used **to denote a class as a Component**. We can use @Component across the application to mark the beans as Spring's managed components.

In Spring, **the objects that form the backbone of your application and that are managed by the Spring IoC container** are called beans. A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container. Otherwise, a bean is simply one of many objects in your application.

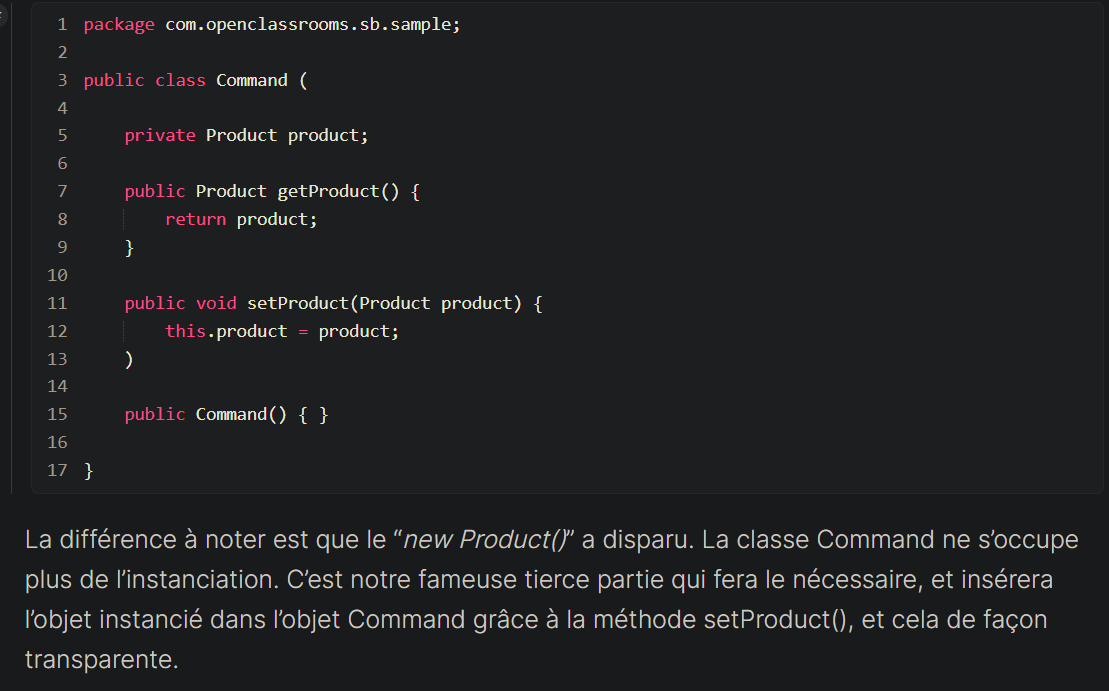
Spring IoC Container is **the core of Spring Framework**. It creates the objects, configures and assembles their dependencies, manages their entire life cycle. The Container uses Dependency Injection(DI) to manage the components that make up the application.

Il permet de créer un code évolutif grâce à une technique de programmation nommée dependency injection.

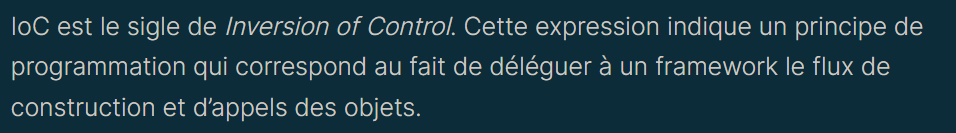
Dépendence forte:



Dependency Injection:

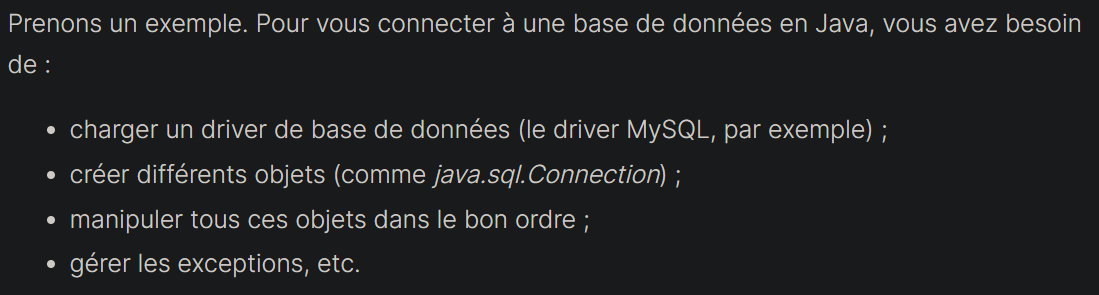


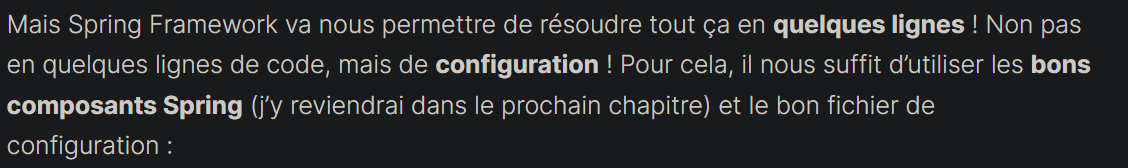
Spring Framework s’occupe de tout grâce à son loC container, ou context Spring. Il permet de créer des objets dynamiquement, et de les injecter dans d’autres objets. De plus, on pourra facilement modifier l’implémentation d’un objet, avec quasiment éero impact sur les objets qui utilisent ce dernier.

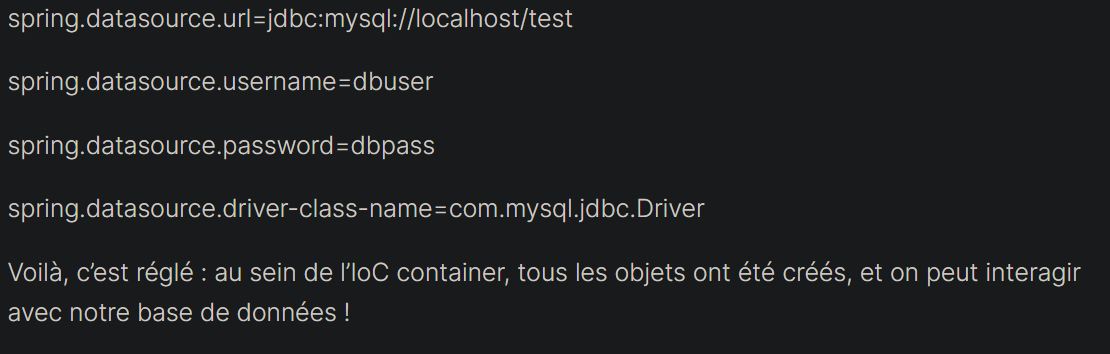


**Spring Framework** nous offer un bel avantage :

**la configuration** pour être plus performant.



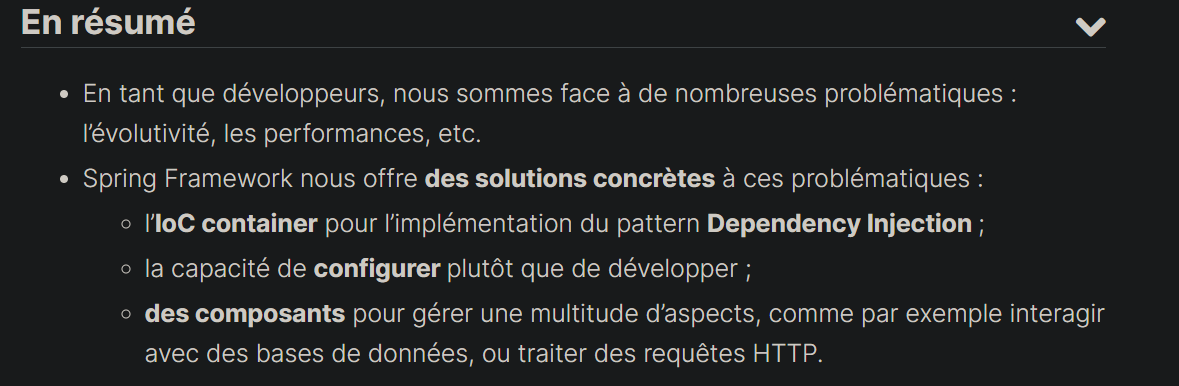


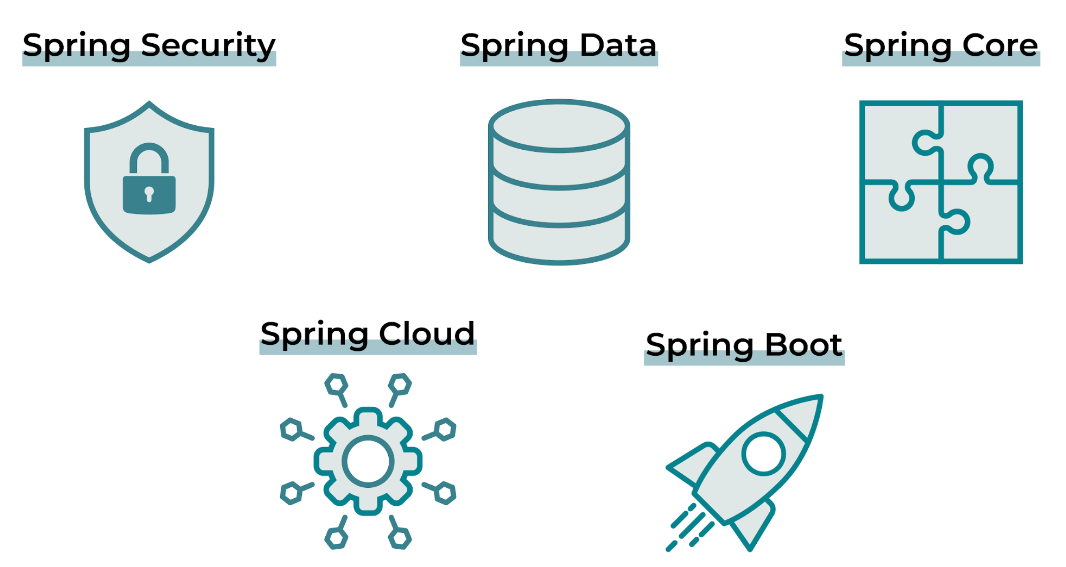


Les autres avantages du spring framework :

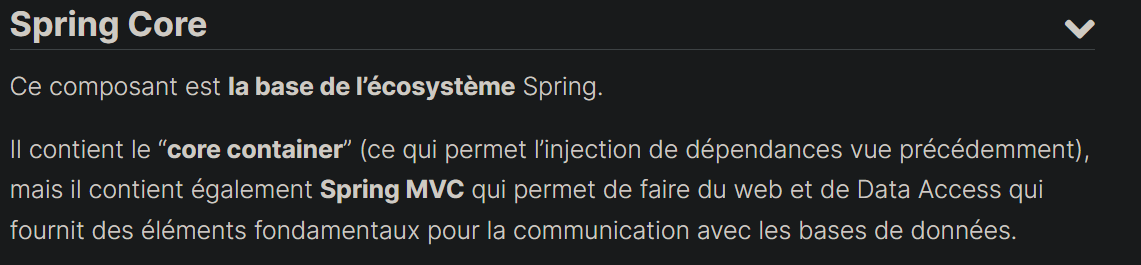
* Interagir avec une base de données.
* Traiter des requêtes HTTP et écrire des réponses HTTP.
* Exécuter des traitements par lots (batch)
* Gérer la sécurité de l’application.
* …

Il ne se limite donc pas à nous fournin l’IOC container, il répond quasiment à tous nos besoins techniques, ce qui augment aussi notre performance.



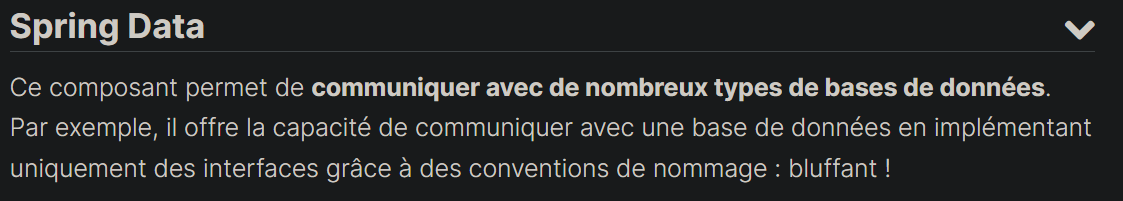


Spring core:

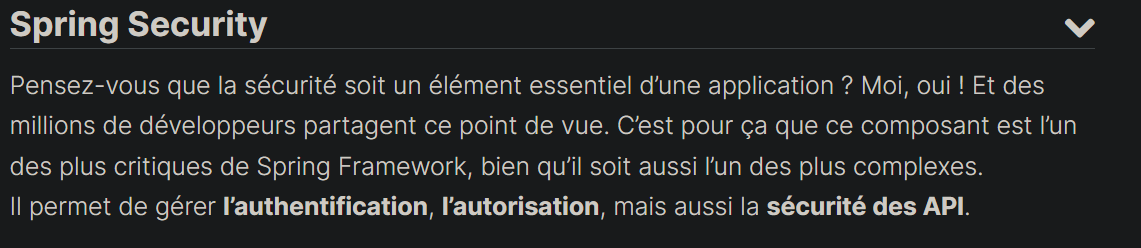


La documentation official du spring :

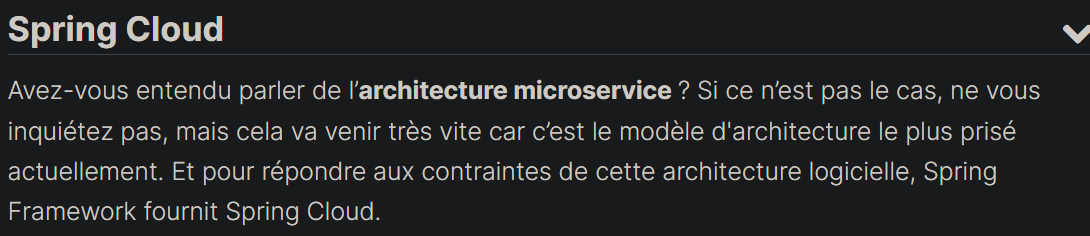
<https://spring.io/projects/spring-framework>

Spring Data

Spring Security



Spring Cloud



**Spring Boot**

C’est un composant très particulier de Spring Framework, dans la mesure où il nous permet de mettre en œuvre tous les autres. Ce cours vous montrera comment tirer profit de la puissance de Spring Boot, et de ses avantages qui sont :

* l'autoconfiguration automatique de Spring ;
* des starters de dépendances ;
* des endpoints Actuator pour fournir des données sur l’application.

En résumé:

* Spring propose de nombreux composants pour répondre aux besoins des développeurs!
* L’un des plus utiles est Spring Boot, car il permet de mettre en oeuvre les autres composants de Spring avec facilité, notamment grâce aux starters de dépendances et à l’autoconfiguration.

**Spring BOOT :**

Il est au service des autres composants, il permet d’utiliser le framework Spring avec une facilité.

Spring boot est un composant de spring framework. Spring contient de nombreux composants qui ne s’utilisent pas de façon exclusive: dans la très grande majorité des projets, il faut utilizer plusieurs composants de Spring simultanément. Donc l’intégration de plusieurs composants spring pour un même projet ajoute de la complexité. La solution est d’utiliser spring boot.

<https://openclassrooms.com/fr/courses/6900101-creez-une-application-java-avec-spring-boot/7077977-identifiez-les-avantages-de-spring-boot#/id/r-7074845>

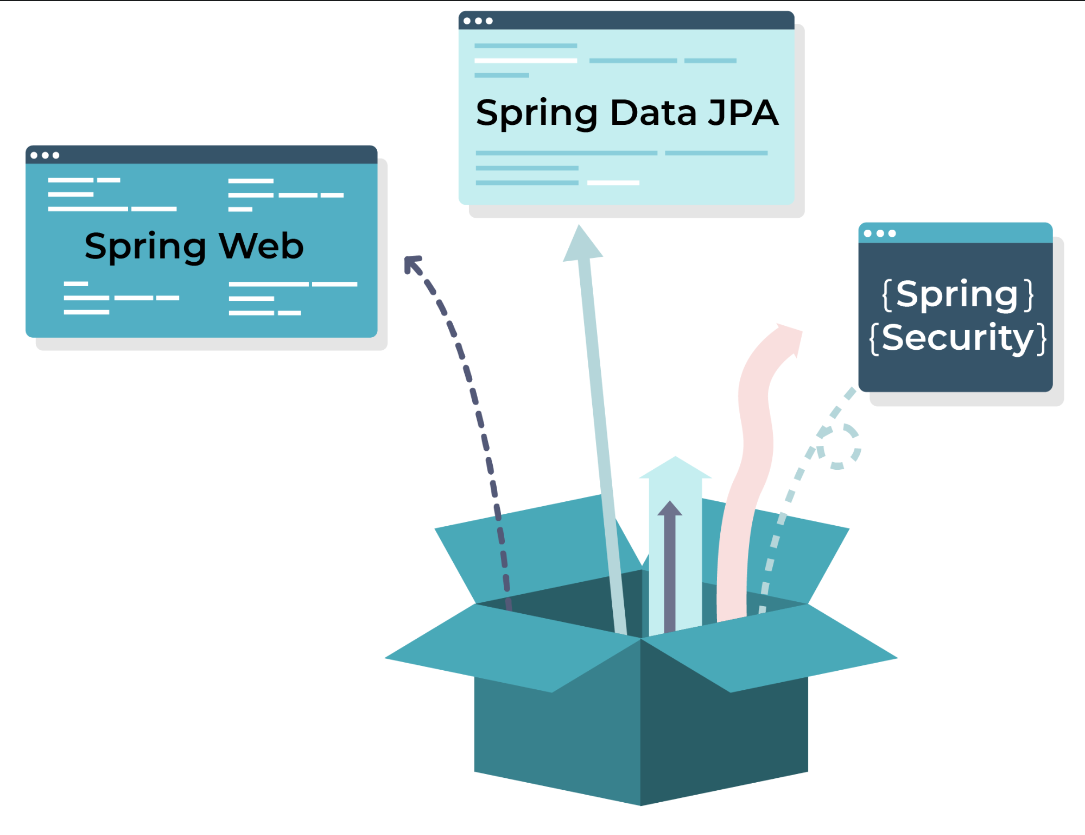
Spring boot moins de fichiers et plus d’automatisation.

En plus, côté déploiement, le projet avec Spring Boot se lance avec une simple commande, alors que le projet sans Spring Boot me demande d’installer un serveur web (comme tomcat) pour le deployer et le démarrer.

**Avantage n\*1:**

Optimisation de la gestion des dépendances.

Spring Boot nous fournit des **starters**, qui correspondent à un ensemble de dépendances homogénéisées (associations, versions). On peut les comparer à des **kits de dépendances**.



Nul besoin de définir les versions des dépendances explicitement dans le pom.xml : Maven les déduit grâce à la version de Spring Boot utilisée.

**Avantage n 2:** l’autoconfiguration.

L’avantage le plus important de spring boot. Bcp moins de configuration (concernant la gestion des servlets, la connexion à la base de données) grace aux annotation qui déclenchent automatiquement de nombreuses operations en background qiu nous sont nécessaires.

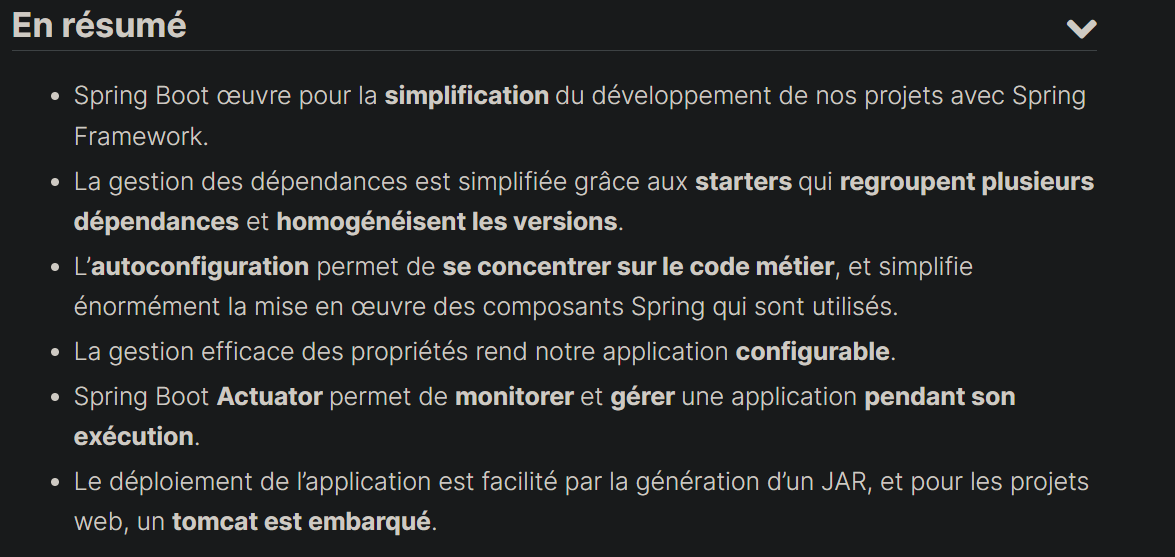
**Avantage n 3** la gestion des propriétés. Exemple c’est possible et facile à faire, de récupérer même des variables d’environnement système, et de les fournir à nos classes.

**Avantage n 4** Spring Boot Actuator correspond à une fonctionnalité de Spring Boot qui permet de monitorer et de manager notre programme pendant qu’il est en cours d’exécution. Exemple, grâce aux endpoints actuatot, on peut modifier une ppté en live, et le programme se tiendra compte sans qu’on ait besoin de le redémarrer.

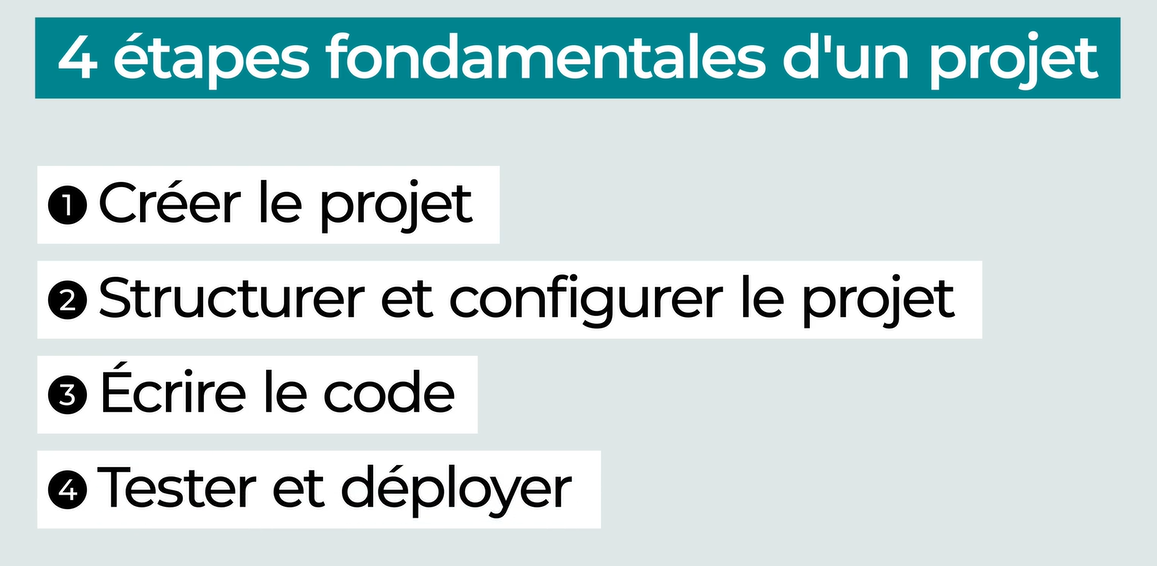
**Avantage n 5** le déploiment, spring boot produit un fichier .jar, on a besoin d’une java runtime environment pour l’exécuter. En effet, un projet spring boot contient un tomcat embarque au sein même du jar généré.

!!!! Spring boot nous offre la possibilité de générer notre projet en war et non jar, si on le souhaite.

L’injection de dépendances repose sur la délégationà une tierce partie de la gestion du flux de création des objets.



Créer un projet spring boot :



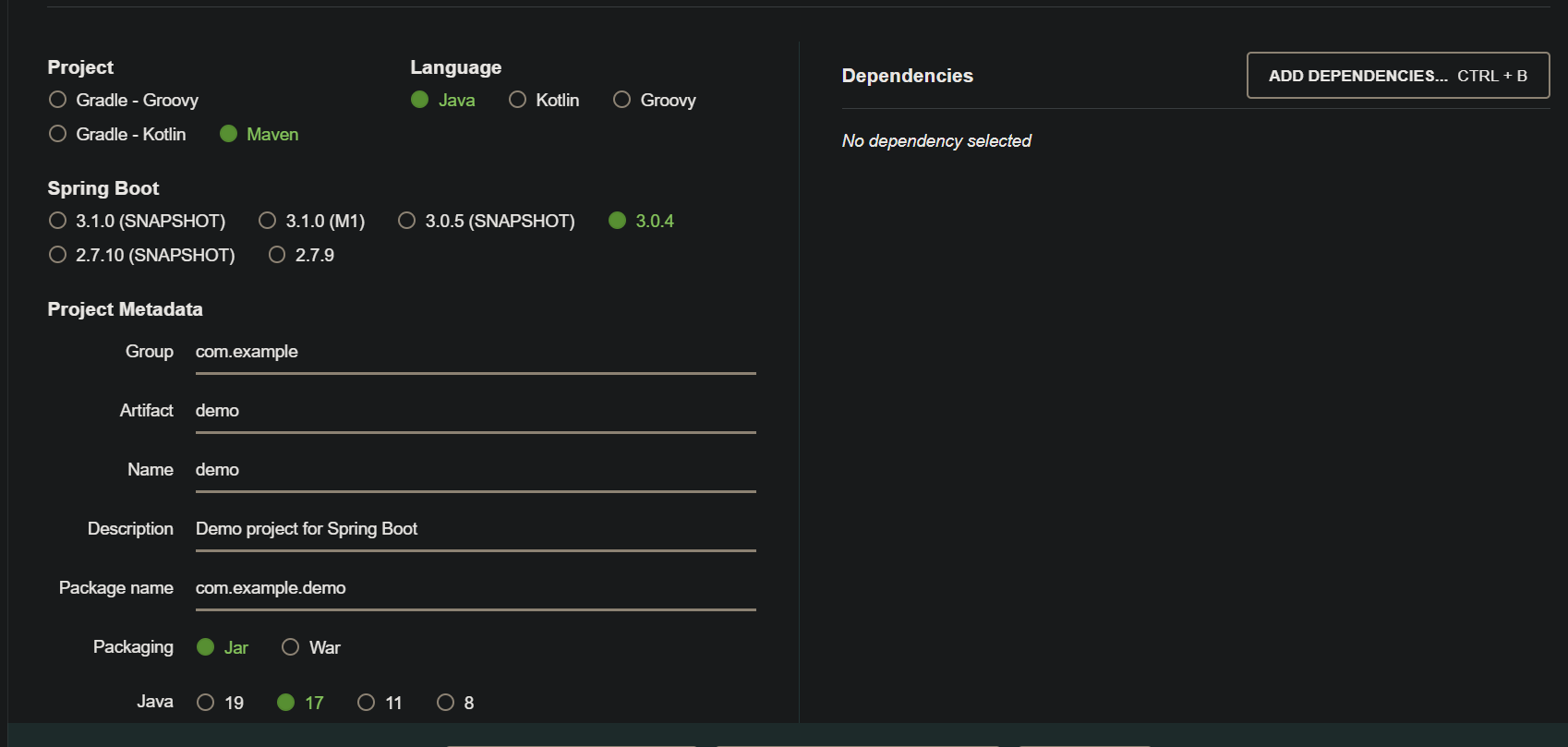
Les starters de dépendances qui sont des kits de dépendances. Exple : Par exemple, le starter **spring-boot-starter-data-jpa** va vous apporter différents JAR pour utiliser Spring et JPA, afin de communiquer avec une base de données.

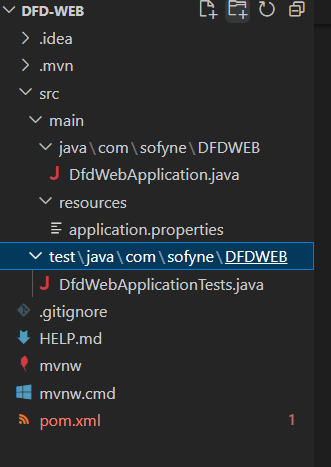
Exemple des starters :

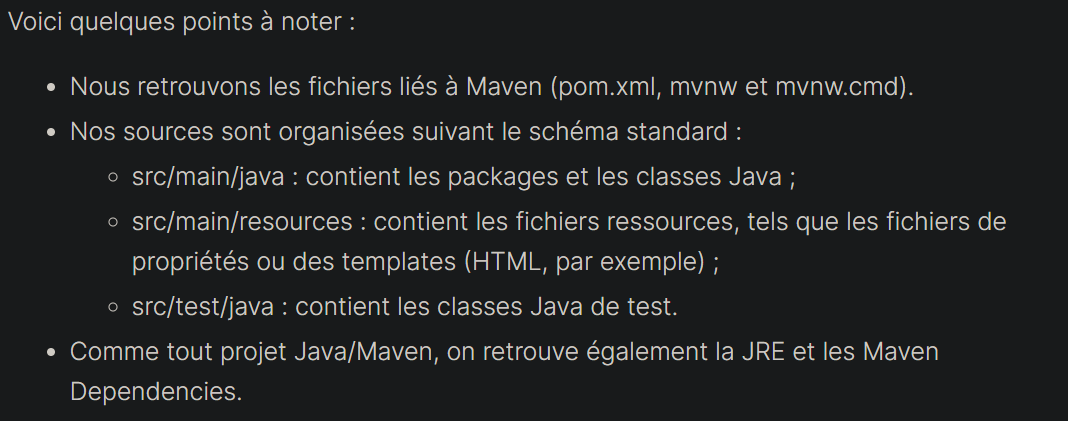
* Spring-boot-starter-core;
* Spring-boot-starter-data-jpa;
* Spring-boot-starter-security;
* Spring-boot-starter-test;
* Spring-boot-starter-web;

Créer le projet càd générer la structure minimale.

Créer le projet avec Spring initializr: start.spring.io



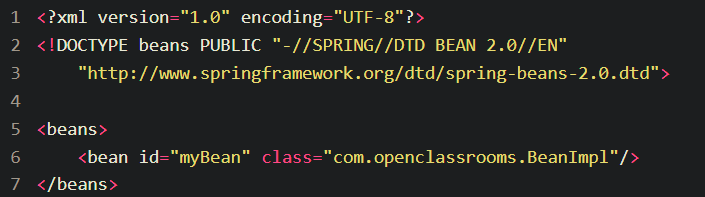




Les annotations en Springboot :

L’ioc container va instancier pour vous des classes, puis si nécessaire les injecter dans d’autres instances de classe , pour qu’une classe soit manipulée par l’Ioc container, il est necessaire de l’indiquer explicitement à Spring. Pour faire, on peut :

Utilizer les fihciers XML au sein desquels on décrit les classes que spring doit gérer, expmle :

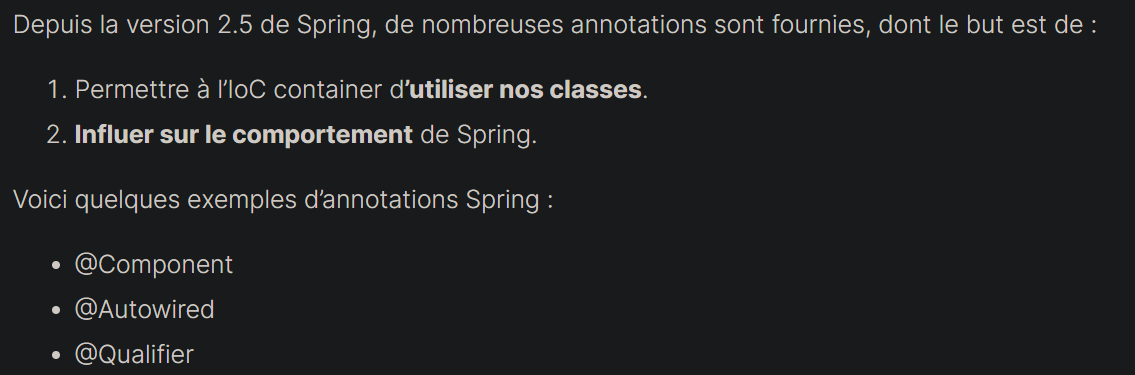


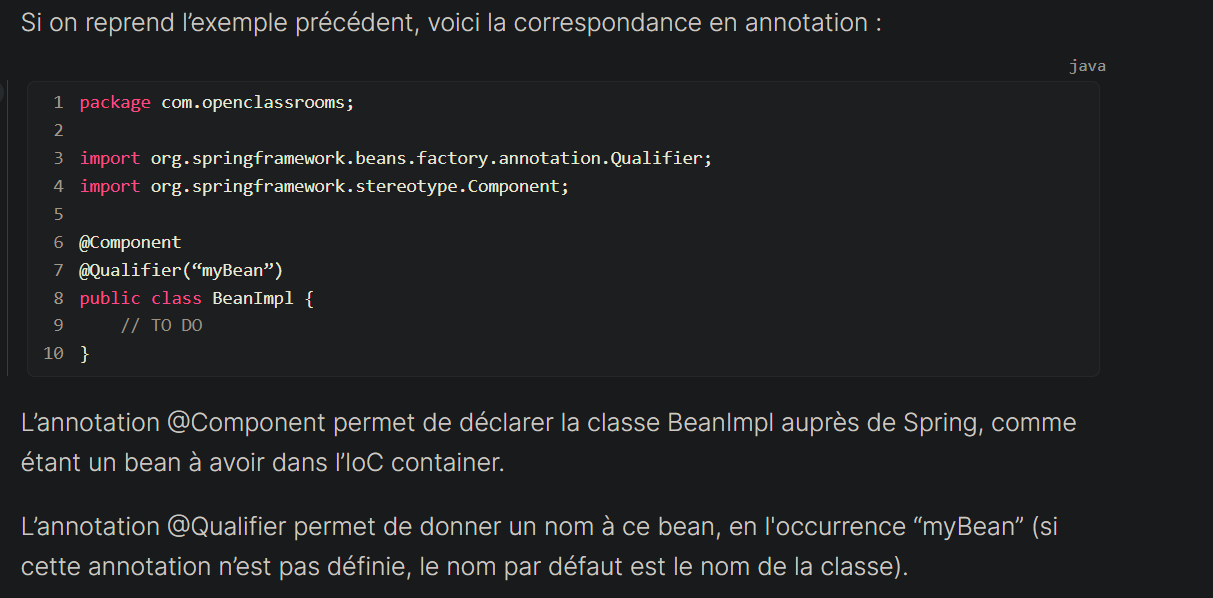
Cette solution est ancienne, et n’est plus beacoup utilisée.

La 2ème solution :

Utiliser des annotations: @[nom\_annotation]

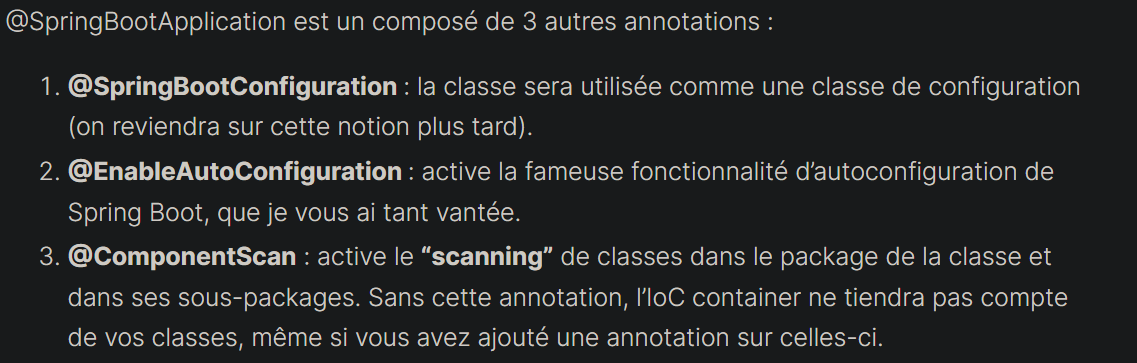
Les annotations peuvent être ajoutée à une classe, une méthode, un attribute. Elle influe sur le comportement du programme.



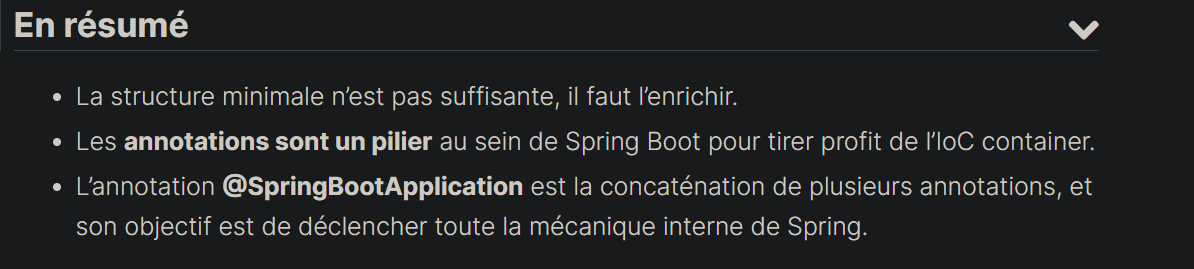


@SpringBootApplication

Elle va permettre à l’ioC container de manipuler la classe qui vient après, et à infuler sur le comportement de spring.



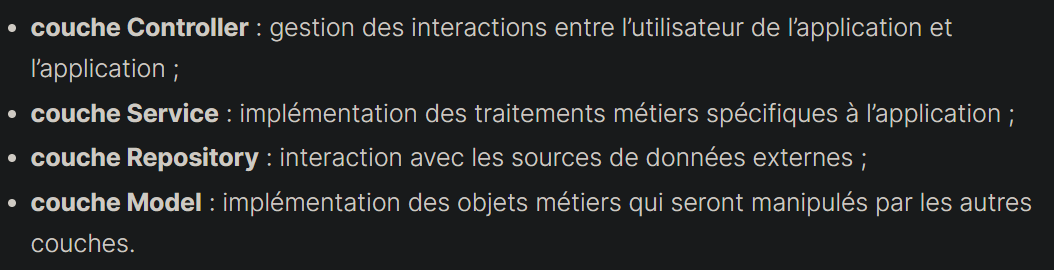
En résume, cette classe c’est ce qui déclenche toute la mécanique interne de Spring Boot et des composants Spring associés.

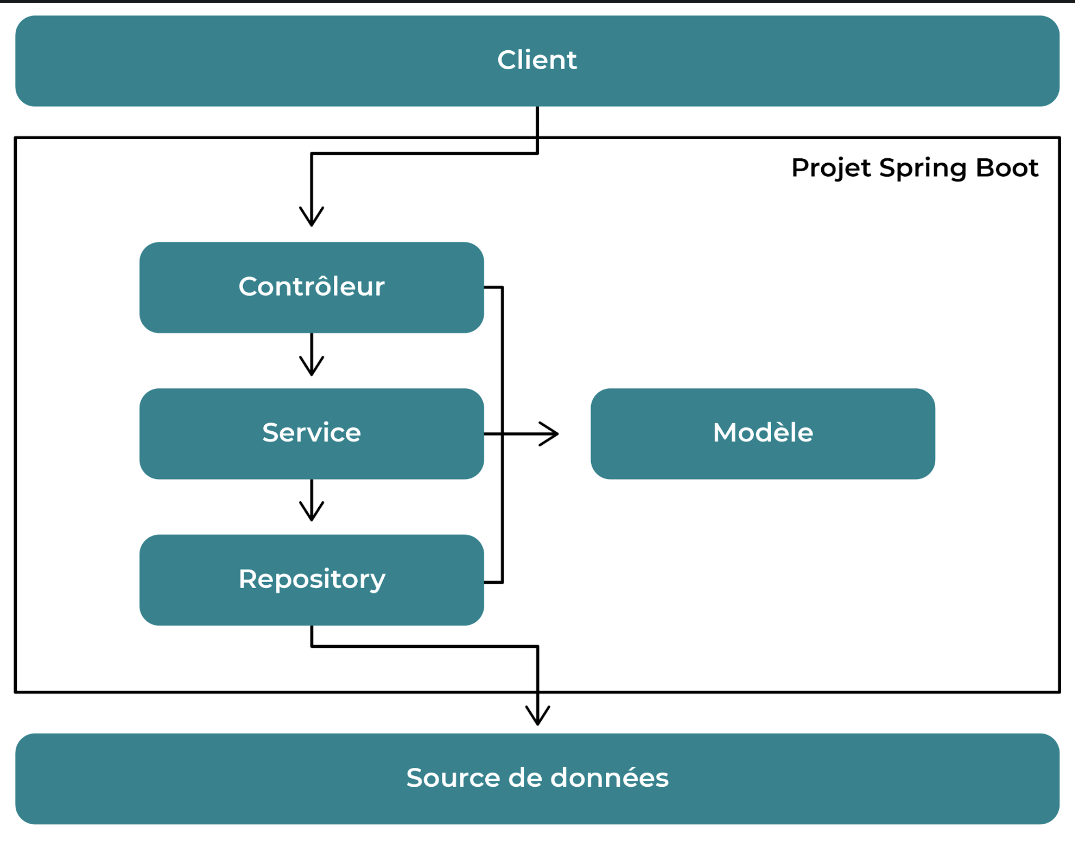


Structurer les packages et configurer le projet springboot.

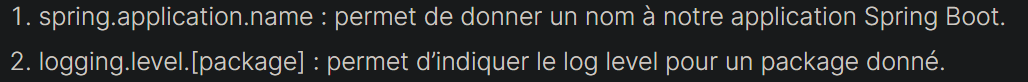
Structurer les packages: comment structure nos packages et nous on ne connait pas encore les classes qui seront produites.

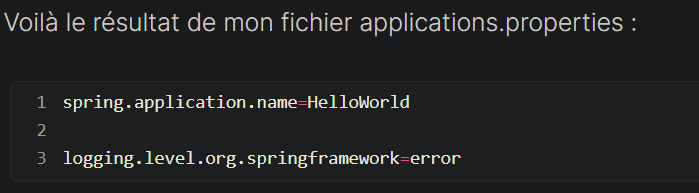
On suit des designs patterns :

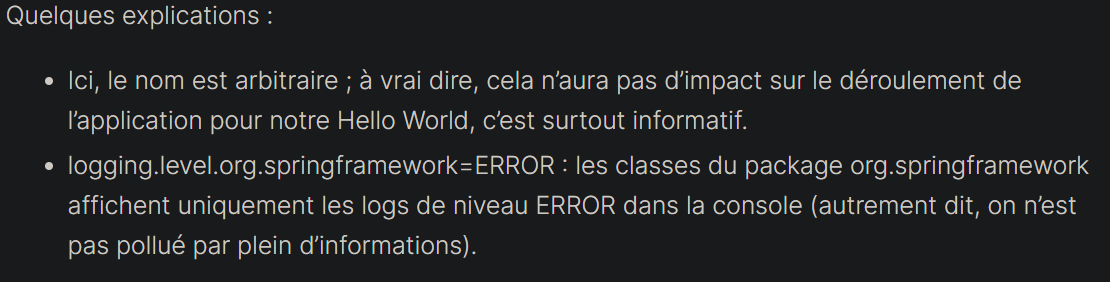


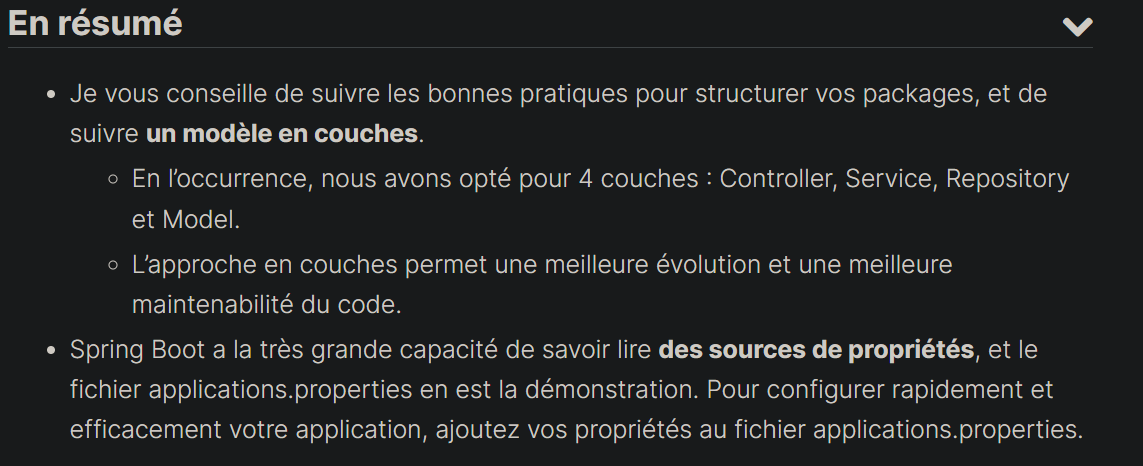


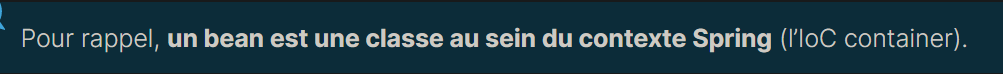
Completer le fichier application.properties :

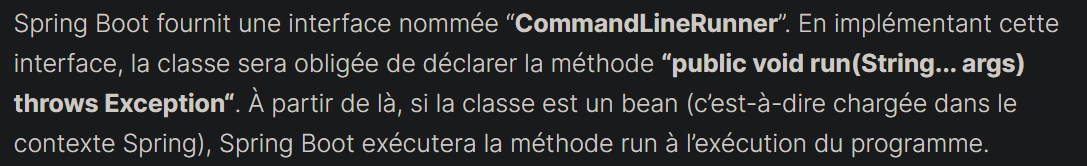


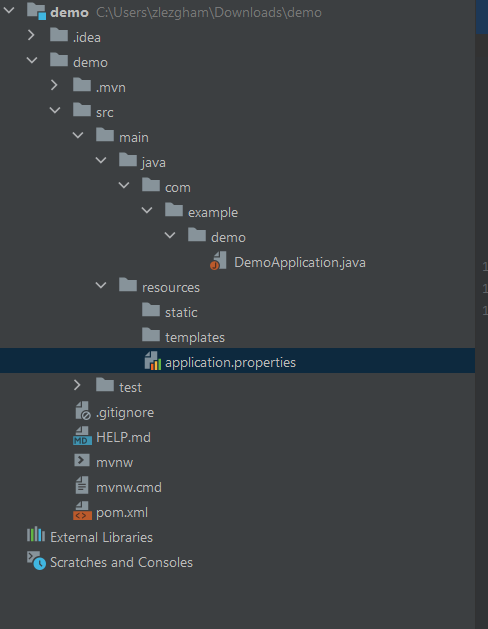












**Application.properties** is where we configure all the proprieties of our application and environment proprieties.

**Static and templates** for web dev (front end) html css js.