Logo

Formation

Introduction to Spring Boot

Logo

Présenté par

el bachir

Présentation de Java

Qu'est-ce que Java ?

Java est un langage de programmation orienté objet, multiplateforme et robuste, développé par Sun Microsystems (aujourd'hui Oracle). Il est conçu pour être portable, sécurisé et performant, ce qui en fait un choix populaire pour le développement d'applications d'entreprise, mobiles (Android) et web.

- `Langage compilé et interprété (via la JVM)`
- `Syntaxe similaire à C/C++ mais simplifiée`
- `Gestion automatique de la mémoire (ramasse-miettes)`

Historique et évolutions majeures de Java

Java a été créé en 1995 par James Gosling. Voici les versions clés :

Version	Année	Principales nouveautés

Java 1.0	1996	Première version publique
Java 5	2004	Génériques, annotations, boucle for-each
Java 8	2014	Lambdas, Stream API
Java 11	2018	Version LTS majeure
Java 17	2021	Nouvelle LTS avec records, sealed classes

Pourquoi apprendre Java ?

Java reste un langage incontournable pour plusieurs raisons :

- `Écosystème immense (Spring, Android, Hadoop...)`
- `Demande forte sur le marché (emplois stables)`
- `Performance optimisée grâce à la JVM`
- `Communauté active et documentation abondante`

- `Backend de nombreuses entreprises (Airbnb, Netflix, LinkedIn)`

Les caractéristiques de Java

Principales caractéristiques techniques :

Caractéristique	Description
Orienté objet	Tout est objet (sauf types primitifs), héritage, polymorphisme
Portable	Bytecode exécuté par la JVM (Write Once, Run Anywhere)
Sécurisé	Gestion stricte de la mémoire, sandboxing
Multi-thread	Programmation concurrente intégrée
Dynamique	Chargement de classes à l'exécution, réflexion

Installation de Java et configuration

Étapes pour démarrer :

Les Bases de la Programmation en Java

Structure d'un programme Java

Un programme Java est structuré autour de classes et de méthodes. La méthode principale `main` est le point d'entrée du programme. Chaque instruction doit se trouver dans une classe.

- Une classe est définie par le mot-clé `class`.
- La méthode `main` est obligatoire pour exécuter un programme.
- Les instructions se terminent par un point-virgule (`;`).

Les variables et les types de données

Les variables sont des conteneurs pour stocker des données. Java est un langage typé statiquement, ce qui signifie que le type d'une variable doit être déclaré explicitement.

- Types primitifs : `int`, `double`, `char`, `boolean`, etc.
- Types référence : `String`, tableaux, objets.
- Déclaration : `type nomVariable = valeur;`.

Type	Exemple	Description
int	int age = 25;	Nombre entier
double	double price = 19.99;	Nombre à virgule flottante
boolean	boolean isJavaFun = true;	Valeur booléenne (true/false)
String	String name = "Alice";	Chaîne de caractères

Les opérateurs (arithmétiques, de comparaison, logiques)

Les opérateurs permettent d'effectuer des opérations sur les variables et les valeurs.

- Opérateurs arithmétiques : `+`, `-`, `*`, `/`, `%` (modulo).
- Opérateurs de comparaison : `==`, `!=`, `>`, `<`, `>=`, `<=`.

- Opérateurs logiques : `&&` (ET), `||` (OU), `!` (NON).

Les structures de contrôle (if, else, switch)

Les structures de contrôle permettent d'exécuter des blocs de code conditionnellement.

- `if` : exécute un bloc si la condition est vraie.
- `else` : exécute un bloc si la condition du `if` est fausse.
- `switch` : permet de tester plusieurs valeurs pour une variable.

Les boucles (for, while, do-while)

Les boucles permettent de répéter un bloc de code plusieurs fois.

- `for` : boucle avec un compteur.
- `while` : boucle tant qu'une condition est vraie.
- `do-while` : boucle exécutée au moins une fois, puis répétée si la condition est vraie.

Introduction aux classes et objets

Java est un langage orienté objet. Une classe est un modèle pour créer des objets, qui sont des instances de cette classe.

- Une classe contient des attributs (variables) et des méthodes (fonctions).
- Un objet est créé avec le mot-clé `new`.
- Les objets communiquent via des méthodes.

Introduction à Spring Boot

Qu'est-ce que Spring Boot ?

Spring Boot est un framework basé sur Spring, conçu pour simplifier le développement d'applications Java. Il permet de créer des applications autonomes et prêtes pour la production avec une configuration minimale. Spring Boot s'appuie sur les principes de Spring (comme l'injection de dépendances) mais ajoute des fonctionnalités pour

accélérer le développement.

- Basé sur le framework Spring
- Configuration automatique et convention over configuration
- Intègre des serveurs embarqués comme Tomcat ou Jetty
- Fournit des starters pour simplifier les dépendances

Pourquoi utiliser Spring Boot plutôt que Spring ?

Spring Boot simplifie considérablement le développement par rapport à Spring traditionnel en réduisant la configuration manuelle et en fournissant des outils intégrés.

- Élimine les fichiers XML de configuration
- Fournit des métriques et des endpoints de monitoring (Actuator)
- Facilite le déploiement avec des exécutables JAR

Critère	Spring	Spring Boot
Configuration	Manuelle (XML/JavaConfig)	Automatique
Serveur embarqué	Non inclus	Inclus (Tomcat/Jetty)
Démarrage rapide	Long (configuration requise)	Rapide (via Spring Initializr)
Dépendances	Gestion manuelle	Starters prédéfinis

Les avantages de Spring Boot

Spring Boot offre plusieurs avantages clés pour les développeurs et les entreprises.

- `Configuration automatique : Spring Boot détecte les bibliothèques présentes et configure automatiquement l'application.`
- `Serveurs embarqués : Plus besoin de déployer sur un serveur externe, Spring Boot intègre Tomcat, Jetty ou Undertow.`
- `Spring Boot Actuator : Fournit des endpoints pour surveiller et gérer l'application en production.`

- Starter POMs : Simplifie la gestion des dépendances avec des modules prédéfinis (ex: spring-boot-starter-web).
- Opinionated defaults : Choix de configurations par défaut optimisées pour accélérer le développement.

Installation et configuration de Spring Boot

Spring Boot peut être configuré rapidement grâce à des outils comme Spring Initializr.

- Spring Initializr (https://start.spring.io/) : Outil en ligne pour générer un projet Spring Boot avec les dépendances souhaitées.
- IDE support : IntelliJ IDEA, Eclipse et VS Code offrent une intégration native avec Spring Boot.
- Maven/Gradle : Spring Boot est compatible avec les deux systèmes de build.

Création d'une Première Application Spring Boot

Structure d'un projet Spring Boot

Un projet Spring Boot suit une structure standard qui facilite l'organisation et la maintenance du code. Voici les principaux répertoires et fichiers :

- `src/main/java : Contient le code source Java de l'application.`
- `src/main/resources : Contient les fichiers de configuration (application.properties ou application.yml), les templates et les fichiers statiques.`
- `src/test/java : Contient les tests unitaires et d'intégration.`
- `pom.xml (ou build.gradle) : Fichier de configuration pour Maven (ou Gradle) qui définit les dépendances et les plugins.`

Répertoire/Fichier	Description
src/main/java	Code source principal de l'application.
src/main/resources/application.properties	Fichier de configuration des propriétés de l'application.
pom.xml	Fichier de configuration Maven pour les dépendances.

Annotation @SpringBootApplication

L'annotation @SpringBootApplication est une méta-annotation qui combine trois annotations essentielles :

- `@Configuration : Indique que la classe peut être utilisée pour définir des beans Spring.`
- `@EnableAutoConfiguration : Active la configuration automatique de Spring Boot.`
- `@ComponentScan : Active la recherche de composants (comme @Controller, @Service, @Repository) dans le package courant et ses sous-packages.`

Création d'un contrôleur REST basique

Un contrôleur REST dans Spring Boot est une classe annotée avec @RestController qui gère les requêtes HTTP.

Voici les étapes pour créer un contrôleur basique :

- `Annoter la classe avec @RestController.`

- Définir des méthodes avec des annotations comme @GetMapping, @PostMapping, etc., pour mapper les requêtes HTTP.
- Retourner des données (souvent au format JSON) directement depuis la méthode.

Exécution de l'application et test avec un navigateur ou Postman

Pour exécuter une application Spring Boot, vous pouvez utiliser :

- L'IDE : Exécutez la méthode main de la classe annotée avec @SpringBootApplication.
- En ligne de commande : Utilisez la commande mvn spring-boot:run (pour Maven) ou gradle bootRun (pour Gradle).
- Une fois l'application démarrée, testez les endpoints avec un navigateur ou Postman.

Méthode	Commande/Description
Via IDE	Exécutez la méthode main de la classe principale.
Via Maven	mvn spring-boot:run

Via Gradle	gradle bootRun
Test	Ouvrez http://localhost:8080/hello dans un navigateur ou Postman.

Les Concepts Clés de Spring Boot

Injection de dépendances (@Autowired, @Component, @Service, @Repository)

L'injection de dépendances est un mécanisme fondamental de Spring Boot qui permet de gérer les dépendances entre les composants de manière automatique. Cela favorise la modularité et la testabilité du code.

- `@Component: Marque une classe comme un composant Spring, permettant sa détection automatique lors du scan de classe.`
- `@Service: Spécialisation de @Component pour les classes de service métier.`
- `@Repository: Spécialisation de @Component pour les classes d'accès aux données (DAO).`
- `@Autowired: Injecte automatiquement une instance d'un composant dans un autre.`

Spring MVC et la gestion des requêtes HTTP

Spring MVC est un framework basé sur le modèle MVC (Modèle-Vue-Contrôleur) qui simplifie la création d'applications web en gérant les requêtes HTTP.

- `@RestController: Combine @Controller et @ResponseBody pour les API REST.`
- `@RequestMapping: Mappe les requêtes HTTP vers des méthodes de contrôleur.`
- `@GetMapping, @PostMapping, etc.: Annotations spécifiques pour les méthodes HTTP.`

Annotation	Description
@RestController	Définit un contrôleur pour les API REST.
@RequestMapping	Mappe une URL vers une méthode.
@GetMapping	Mappe une requête GET.

Configuration avec application.properties/application.yml

Spring Boot permet de configurer l'application via des fichiers de propriétés (application.properties) ou YAML (application.yml). Ces fichiers centralisent les paramètres tels que les ports, les URLs de base de données, etc.

- `application.properties: Format clé-valeur (par exemple, server.port=8080).`
- `application.yml: Format YAML plus lisible et structuré.`

Propriété	Exemple	Description
server.port	8080	Définit le port du serveur.
spring.datasource.url	jdbc:h2:mem:testdb	URL de la base de données.

Introduction à la base de données embarquée (H2)

H2 est une base de données relationnelle embarquée, légère et en mémoire, souvent utilisée pour le développement et les tests avec Spring Boot.

- `Mode mémoire: Les données sont perdues après l'arrêt de l'application.`

- `Console H2: Interface web pour interagir avec la base de données (activée via spring.h2.console.enabled=true).`

Propriété	Valeur par défaut	Description
spring.h2.console.enabled	false	Active la console H2.
spring.h2.console.path	/h2-console	Chemin d'accès à la console.

Conclusion

Spring Boot simplifie le développement Java

Spring Boot est un framework qui permet de simplifier la configuration et le déploiement d'applications Java. Il offre des fonctionnalités clés comme l'autoconfiguration, les starters et les outils embarqués, réduisant ainsi la complexité du développement.

- Autoconfiguration : Spring Boot configure automatiquement les dépendances et les paramètres en fonction des bibliothèques détectées dans le projet.
- Starters : Des dépendances prédéfinies pour des cas d'usage courants (ex : spring-boot-starter-web pour les applications web).
- Outils embarqués : Serveurs comme Tomcat ou Jetty intégrés par défaut, éliminant la nécessité de déploiement externe.

Prêt à explorer les fonctionnalités avancées

Après cette introduction, vous pouvez approfondir vos connaissances avec des fonctionnalités plus avancées de Spring Boot pour créer des applications robustes et évolutives.

- Spring Data JPA : Simplifie l'accès aux bases de données avec des repositories prédéfinis.
- Spring Security : Gère l'authentification et l'autorisation de manière sécurisée.
- Spring Cloud : Pour le développement d'applications microservices.
- Actuator : Surveillance et gestion de l'application en production.

Fonctionnalité	Utilité
Spring Data JPA	Accès aux bases de données avec peu de code
Spring Security	Sécurisation des applications
Spring Cloud	Développement de microservices
Actuator	Surveillance des métriques et santé de l'application

Créer vos propres applications

Avec les bases acquises, vous pouvez maintenant commencer à développer vos propres applications Spring Boot.

Voici quelques étapes pour démarrer :

- `Utilisez Spring Initializr (https://start.spring.io/) pour générer un projet avec les dépendances nécessaires.`
- `Explorez la documentation officielle de Spring Boot pour des guides détaillés.`
- `Expérimentez avec des tutoriels pratiques pour consolider vos connaissances.`