Programma Corso di Programmazione Backend

Benvenuti al corso di formazione in Programmazione Backend, un viaggio formativo progettato per fornire competenze solide e pratiche nello sviluppo di applicazioni lato server. Questo corso è dedicato a chi desidera intraprendere una carriera come sviluppatore back-end o approfondire le proprie conoscenze in questo ambito, partendo da una base di utilizzo del computer.

Il backend è il cuore di ogni applicazione moderna. È qui che i dati vengono gestiti, le logiche applicative prendono vita e le richieste degli utenti trovano risposta. Durante il corso esploreremo i fondamenti della programmazione backend, i linguaggi e i framework principali, i database relazionali e non relazionali, e le pratiche di testing, culminando nella realizzazione di un progetto completo in Java Spring.

Settimana 1: Introduzione alla Programmazione e Configurazione degli Strumenti

> ### Giorno 1:
> - **Introduzione alla programmazione:** concetti fondamentali, differenza tra frontend e backend.

- **Strumenti di sviluppo:** installazione di Java, Maven, Git, e configurazione di un IDE (IntelliJ/Eclipse).

Giorno 2:

- **Git:** comandi base (clone, add, commit, push, pull).
- **Repository GitHub:** creazione di un semplice repository condiviso.
- **Risoluzione problemi:** introduzione a pseudocodice e approccio alla risoluzione di problemi.

Giorno 3:

- **Fondamenti Java:** variabili, tipi di dati, operatori, strutture di controllo.
- **Primo programma Java:** input/output da console.

Giorno 4:

- **Programmazione orientata agli oggetti:** classi, oggetti, ereditarietà, polimorfismo.
- **Esempi OOP:** applicazione pratica in Java per modellare un problema reale.

Giorno 5:

- ***Progetto OOP:*** *realizzazione di un programma con gestione di classi e metodi.*
- ***Versionamento con Git:*** *utilizzo per il progetto sviluppato.*

Settimana 2: Java e le sue Versioni

Giorno 6:

- **Fondamenti di Java:** classi, metodi, costruttori e utilizzo delle interfacce.
- **Introduzione alle Collezioni:** panoramica su ArrayList, HashMap e HashSet.
- **Esempi Pratici:** implementazione di strutture dati comuni con le collezioni di base.

Giorno 7:

- **Lambda Expressions:** sintassi di base e confronto con classi anonime.
- **Functional Interfaces:** utilizzo e implementazioni comuni come Predicate, Function e Consumer.
- **Esercizi:** creazione di espressioni lambda per semplificare codice complesso.

Giorno 8:

- **Stream API:** utilizzo di operazioni intermedie e terminali per il trattamento di dati.
- **Optional:** gestione sicura dei valori nulli per prevenire NullPointerException.
- **Progetti:** implementazione di filtri e trasformazioni con Stream API e Optional.

Giorno 9:

- **Innovazioni di Java 8:** Default Methods nelle interfacce e metodi statici.
- **Nuovi Metodi Utility:** aggiunte significative in Arrays, Collections e Objects.
- **Case Study:** come Java 8 ha migliorato la leggibilità e la manutenibilità del codice.

Giorno 10:

- ***Progetto Java:*** *utilizzo di Stream, Optional e Date API per analisi dati e gestione temporale.*
- ***Refactoring:*** *modernizzazione di codice legacy utilizzando le novità di Java 8.*

Settimana 3: SQL Language / JDBC

Giorno 11:

- **Correzione Esercitazione:** revisione approfondita dei progetti realizzati nella settimana 2.
- **Discussione sugli errori comuni:** individuazione delle problematiche riscontrate e suggerimenti per migliorare efficienza e manutenibilità del codice.
- **Refactoring e ottimizzazione del codice:** utilizzo delle best practices

apprese nelle settimane precedenti, con applicazione di Stream API, Optional e lambda expressions.

Giorno 12:

- **Database Relazionali:** concetti base di SQL, tabelle, colonne, relazioni e chiavi primarie/esterne.
- **Installazione di MySQL:** configurazione di un database MySQL e connessione con MySQL Workbench.
- **Query SQL:** SELECT, INSERT, UPDATE, DELETE con esempi pratici.

Giorno 13:

- **Introduzione a JDBC:** panoramica e vantaggi rispetto ad alternative come JPA e Hibernate.
- **Configurazione di JDBC:** installazione dei driver MySQL e connessione a un database con Java.
- **Operazioni fondamentali:** esecuzione di query SQL tramite JDBC, recupero e manipolazione dei dati.

Giorno 14 (Esercitazione):

- ***Inserimento e recupero dati:*** *creazione di un programma Java per operazioni CRUD (Create, Read, Update, Delete) su MySQL con JDBC.*
- ***Gestione degli errori SQL:*** *trattamento delle eccezioni con SQLException*

Giorno 15 (Test Intermedio):

- Esercizio teorico: test con domande a risposta aperta.
- Esercizio pratico: realizzazione di un'applicazione.
- Discussione: revisione degli errori più comuni.

Settimana 4: Revisione Codice, DAO e Sviluppo di un Gioco Testuale

Giorno 16:

- **Review del Codice:** analisi del codice scritto nelle settimane precedenti, individuazione di errori e miglioramenti.
- **Introduzione al pattern DAO:** concetti fondamentali e best practice per l'interazione con MySQL.
- **Strutturazione di un database MySQL:** progettazione di tabelle, chiavi primarie e relazioni tra dati.

Giorno 17:

- **Layer di sviluppo:** introduzione ai concetti di Service e Repository e loro ruolo nell'architettura di un'applicazione.
- **Esercitazione pratica:** avvio di un progetto di gioco testuale con connessione a un database MySQL.
- **Implementazione dei primi metodi:** creazione di classi per la gestione degli oggetti di gioco e la loro persistenza.

Giorno 18:

- **Utilizzo di Lombok:** introduzione e vantaggi della riduzione del codice boilerplate.
- **Logging in Java:** utilizzo di `java.util.logging` per la gestione dei log nell'applicazione.
- **Continua lo sviluppo del gioco testuale:** implementazione della gestione dell'inventario e interazioni tra personaggi, integrando un db MySql tramite JDBC.

Giorno 19:

- **Proseguimento dello sviluppo del gioco testuale:** aggiunta di nuove funzionalità e gestione avanzata dei dati.
- **Accenno alle metodologie Agile:** introduzione a Scrum e Kanban per la gestione dei progetti software.

Giorno 20 (Consegna Esercitazione):

- ***Finalizzazione del gioco testuale:*** *verifica delle funzionalità implementate e test delle interazioni.*
- ***Consegna del progetto:*** *ogni studente presenta il proprio lavoro e riceve feedback.*
- ***Revisione collettiva:*** *discussione sugli errori più comuni e possibili ottimizzazioni.*

Settimana 5: JPA, Hibernate e Introduzione a Spring

Giorno 21:

- **Introduzione a Spring:** concetti base di Spring Framework e Spring Boot.
- **Dependency Injection:** creazione di un'applicazione Spring con gestione dei bean.
- **Configurazione del database in Spring Boot:** utilizzo di `application.properties` e Spring Data JPA.

Giorno 22:

- **Introduzione a JPA e Hibernate:** concetti di ORM, differenze tra JDBC e JPA, vantaggi di un ORM.
- **Configurazione di Hibernate:** creazione di un progetto base con JPA, configurazione del file `persistence.xml`.
- **Gestione delle Entità:** utilizzo delle annotazioni `@Entity`, `@Table`, `@Id`, `@GeneratedValue`.

Giorno 23:

- **Relazioni tra entità:** tipi di relazioni (`@OneToOne`, `@OneToMany`, `@ManyToMany`) e gestione delle foreign key.
- **CRUD con Hibernate:** gestione delle operazioni di persistenza con `EntityManager` e `Repository`.
- **Transazioni e gestione delle eccezioni:** commit, rollback e gestione degli errori SQL.

Giorno 24:

- **JPQL (Java Persistence Query Language):** scrittura di query personalizzate.
- **Testing delle entità:** utilizzo di JUnit e TestContainers per testare l'integrazione con il database.

Giorno 25 (Esercitazione):

- ***Progetto Hibernate:*** *realizzazione di un'applicazione CRUD completa con Spring Boot e MySQL.*
- ***Test e Debugging:*** *revisione degli errori più comuni e suggerimenti per ottimizzare le query.*

Settimana 6: Approfondimento su Spring e Confronto con altri Linguaggi

Giorno 26:

- **Spring Boot e API REST:** creazione di controller REST.
- **Gestione delle richieste HTTP:** metodi GET, POST, PUT, DELETE e gestione degli status code.

Giorno 27:

- **Gestione dei dati in Spring Boot:** interazione con database Mongo.

Giorno 28:

- **Confronto con altri linguaggi backend:** panoramica su Python (FastAPI) e JavaScript (Node.js, Express).
- **Pro e Contro di Java rispetto ad altri linguaggi backend:** performance, scalabilità, community e best practice.
- **Quando scegliere Java per lo sviluppo backend?** Scenari d'uso e casi aziendali.

Giorno 29:

- **Testing in Java:** utilizzo di JUnit e Mockito per testare i servizi Spring.
- **Best practices per la scrittura di API REST:** error handling, validazione delle richieste.

Giorno 30 (Esame Finale):

- Progetto Finale: sviluppo e presentazione di un'applicazione REST con Spring Boot.

> **- Test di Valutazione: esercizi teorici e pratici su JPA, Hibernate e Spring.**
> **- Discussione e Feedback: analisi dei progetti realizzati, individuazione dei punti di forza e delle aree di miglioramento.**

← Pagina precedente

Pagina successiva →