PREPARANDO AMBIENTE DE DESENVOLVIMENTO

**JAVA FULL STACK**

Em tradução literal, Full Stack quer dizer Pilha Cheia.

Até pouco tempo atrás as empresas classificavam os profissionais de tecnologia em:

* Programador Front – End.
* Programador Back – End.
* Analista de Banco de Dados.
* Analista de Infraestrutura (DevOps).

Com o tempo as empresas perceberam uma enorme dificuldade em gerenciar estes times individualmente com a proposta de construção de produtos como um Sorftware. O maior dilema para as empresas com times distribuídos por habilidades, é o fato da dificuldade de compreensão dos requisitos e a proposta de funcionalidade do projeto.

É por esta razão que hoje o mercado de TI tem exigido e recrutado profissionais em desenvolvimento que sejam capazes de dominar as quatros habilidades citadas acima, sendo denominado como um Desenvolvedor Full Stack.

**Back – End**

Back-end representa a parte por trás de toda interação de usuário em um Sorftware, seja um site, um sistema desktop, uma aplicação web e até mesmo os aplicativos.

Também denominado como SERVER SIDER (do lado do SERVIDOR), a camada de back-end é responsável por disponibilizar os recursos abaixo:

* Regra de negócio
* Acesso a dados
* Segurança
* Integração via APIs

Para nos tornarmos um programador back-end, precisamos conhecer uma ou algumas das linguagens de programação e banco de dados disponíveis e utilizados pela empresa.

**LINGUAGEM DE PROGRAMAÇÃO**

* JavaScript Java C#
* PHP Kotlin Scala
* Perl C++ Python
* Ruby

Conheça mais sobre as características de cada linguagem pelo link ***ewally***

**Front-End**

Front-end está diretamente relacionado ao Desenvolvimento dos recursos disponíveis para que haja a interação dos usuários em nosso software ou aplicativo. Todo os componentes gráficos e comportamentos que acontecem devido ao uso da aplicação é desenvolvida por um profissional denominado de **Dev Front End.**

Um profissional Dev Front End deve pensar em requisitos de suma importância para o desenvolvimento de qualquer aplicação que necessite de interação com o usuário.

* Aparência
* Usabilidade
* Objetivo
* Segurança

Diferentemente da camada de back-end, o font-end é composto por uma linguagem de programação a java script e tecnologias complementares que são conhecidas como : Hypetxto, gerador de estilo, Bibliotecas ou Framework nos quais vamos listar abaixo:

|  |  |
| --- | --- |
| TECNOLOGIA | DEFINIÇÃO |
| HTML | Linguagem de marcação de hypertexto |
| CSS | Linguagem de estilização de conteúdo HTML |
| BOOTSTRAP | Biblioteca de estilo CSS pré-definido e criado pelo Twitter |
| JQUERY | Biblioteca JavaScript muito utilizada antes da era das aplicações SPA |
| NODEJS\* | Plataforma capaz de executar códigos JavaScript fora de um navegador web |
| VUEJS | Framework JavaScript de código-aberto, focado no desenvolvimento de interfaces de usuário e aplicativos de páginas únicas |
| REACT | O React é uma biblioteca JavaScript de código aberto com foco em criar interfaces de usuário em páginas web. É mantido pelo facebook, Instagram, outras empresas e uma comunidade de desenvolvedores individuais. |
| ANGULAR | Angular é uma plataforma de aplicações web de código-fonte aberto e front-end baseado em TypeScript liderado pela Equipe Angular do Google e por uma comunidade de indivíduos e corporações. |
| TYPESCRIPT | TypeScript é uma linguagem de programação de código aberto desenvolvida pela Microsoft. É um superconjunto sintático escrito de JavaScript e adiciona tipagem estática opcional à linguagem. |

**DESENVOLVEDOR WEB FULL STACK**

Um desenvolvedor web full Stack é uma pessoa que pode desenvolver software cliente e servidor. Além de dominar HTML e CSS, ele também sabe:

* Programa em um navegador (como usar JavaScript, Jquery, angular ou vue)
* Programar um servidor (como usar PHP, Java, Python ou Node)
* Programar um banco de dados (como usar SQL, SQLite ou MongoDB)

Par se tornar um excelente Desenvolvedor Web Full Strack você jamais deverá esquecer que as empresas também buscam e valorizam profissionais especialistas capazes de dominar com excelência algum ou alguns dos requisitos essenciais no desenvolvimento de softwares.

**JAVA**

Hoje java é mantida pela Oracle inc. e já está na sua versão 18 do JDK.

**Orientada a Objetos**

Em programação orientada a objetos, um objeto é uma abstração do mundo real composto por um identificador (nome), atributos(propriedades) e métodos (comportamentos) definidos em uma classe.

Em java, uma classe é descrita em um arquivo com a extensão (.java). Abaixo iremos explorar algumas expressões utilizadas na programação a objetos e uma ilustração de estrutura de uma classe Student e a criação de objetos a partir desta classe.

* **Instancia** (new): É o ato de criar um objeto a partir de estrutura definida em uma classe.
* **Comportamentos** (behavior): Também conhecido como ações ou métodos, é toda parte comportamental que um objeto dispõe.
* **Características** (states): Também conhecido como atributos ou propriedades, é toda informação que representa o estado do objeto.
* **Identificador** (identity): Proposito existencial aos objetos que serão criados.
* **Classe** (class): A estrutura e ou representação que direciona a criação dos objetos de mesmo tipo.

Primeiro criamos a estrutura correspondente para assim podermos cria-los com a as características e possibilidade de realização de ações (comportamentos) como se fosse no mundo real.

1 //Criando a classe Student

2

3

4 public class Student {

6 String name;

7 int age;

8 Color color;

9 Sex sex;

10

11 void eating (Food food){

12 //Nosso código

13 }

14 void drinking (Eat eat){

15 //Nosso código

16 }

17 void running (){

18 //Nosso código

19 }

20 }

1 //Criando objetos a partir da classe Student

2

3 public class School {

4 public static void main(String[] rags) throws Exception {

5 Student student1 = new Stundent();

6 student1 . name = “John”;

7 student1 . age = 12;

8 student1 . color = Color.FAIR;

9 student1 . sex = Sex.MALE;

10 Student student2 = new Stundent();

11 student2 . name = “Sophia”;

12 student2 . age = 10;

13 student2 . color = Color.FAIR;

14 student2 . sex = Sex.MALE;

15 }

16 }

**TECNOLOGIAS**

* Java JDK
* Eclipse
* Hibernate
* Spring Farmework
* Postegres
* Swagger
* Github
* Mavem
* REST
* Heroku

**ARQUITETURA**

O projeto Banckline será desenvolvida na arquitetura do Spring Framework + Spingboot estruturado no padrão MVC para prover uma API Rest para movimentações financeiras simples.

**SPRING FRAMEWORK**

Framework open source desenvolvido para a plataforma Java baseado nos padrões de projetos invenção de controle e injeção de dependência.

Sua estrutura é composta por módulos afins de reduzir a complexidade no desenvolvimento de aplicações simples ou corporativa.

**Spring Framework Runtime**

**Data Access/Integration** **WEB (MVC/Remoting)**

Instrumentation

Aspects

AOP

CORE CONTAINER

Beans

Core

Context

Expression language

TEST

**Spring versus JavaEE**

Usar o Spring era um caminho mais simples e mais fácil de evoluir. Aí chegou a versão 5 do JavaEE e a discussão voltou a ficar um pouco mais quente.

O Spring Framework é estruturado em dois pilares:

**INVERSION OF CONTROL OU LOC**

Trata-se do redirecionamento do fluxo de execução de um código retirando parcialmente o controle sobre ele e delegando-o para um container. O principal propósito é minimizar o acoplamento do código.

**INJEÇÃO DE DEPÊNDECIA**

É um padrão de desenvolvimento com a finalidade de manter baixo o nível de acoplamento entre módulos de um sistema.

Para desfrutar dos recursos oferecidos pelo Spring framework, precisamos compreender três conceitos essenciais.

* BEANS

Objetivo que é instanciado (Criado), montado e gerenciado por um container através do principio da inversão de controle.

* SCOPES

Controle da existência de nossos objetivos Componentes da aplicação em uma esfera de utilização: Singleton, Prototype, Reques, Session, global

* Autowired

Um recurso em forma de anotação(indicação) onde deverá ocorrer uma injeção automática de dependência.

**SPRINGBOOT**

Enquanto o Spring framework é baseado no padrão de injeção de dependência, o springboot na configuração automática.

**Starters**

Os starters são dependências que agrupam outras dependências com um propósito em comum. Dessa forma, somente uma configuração é realizada no seu gerenciador de dependências.

Starters essenciais

**Spring-boot-starter\***

* **data-jpa** : Integração ao banco de dados via JPA – Hibernate.
* **data-mongodb** : Interação com banco de dados MongoDB
* **web**: Inclusão do container Tomcat para aplicação REST
* **web-services**: Webservices baseado na arquitetura SOAP
* **batch**: Implementação de JOBs de processos.
* **test**: Disponibilização de recursos para teste unitários como JUnit
* **openfeign**: Client HTTP baseado em interfaces
* **actuator**: Gerenciamento de monitoramento da aplicação.

INSTALANDO OS PROGRAMAS PARA O AMBIENTE DE DESENVOLVIMENTO

Instalado fora das IDEs.

1. Git download
2. GitHub desktop
3. PostgreSQL