

Web Academy

Fundamentos de Programação Back-end



Manoel Limeira de Lima Júnior



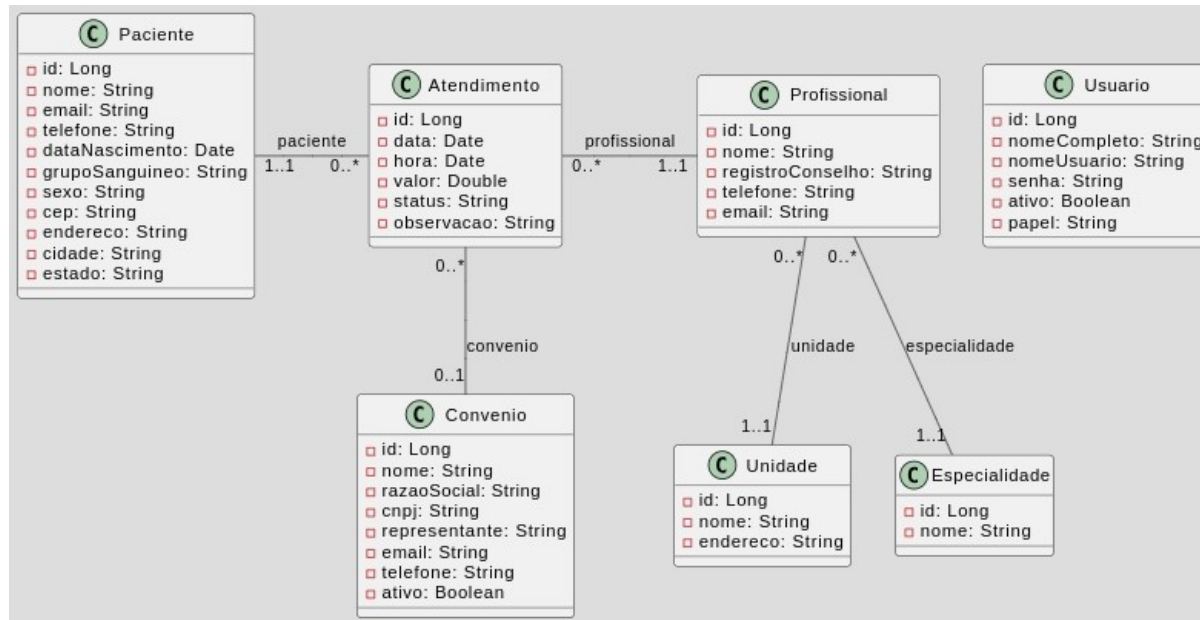
Web Academy



Apresentação

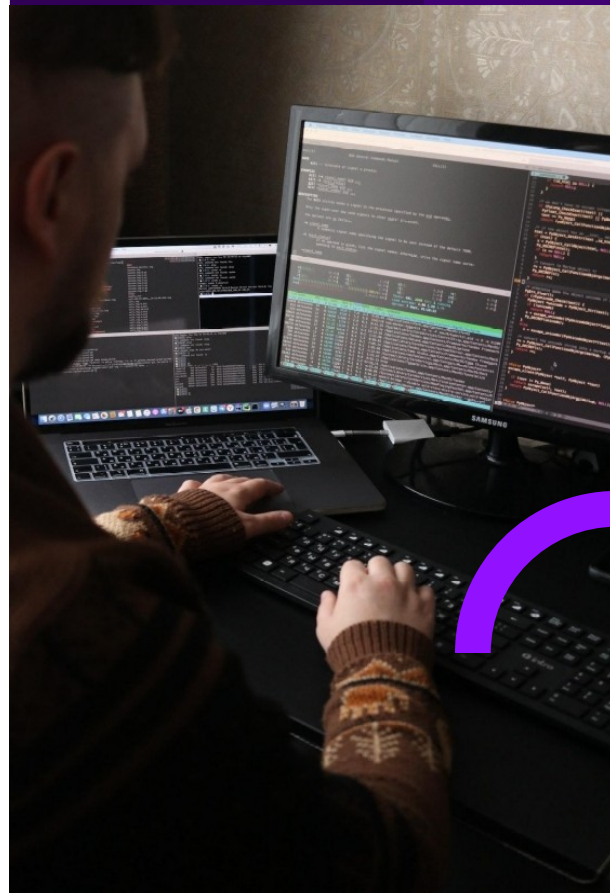
SGCM - Sistema de Gerenciamento de Consultas Médicas

- Documentação: <https://github.com/webacademyufac/s'gcmdocs>
 - Diagrama de classes



Ementa

1. Linguagens de programação **server-side**.
2. Arquitetura em **camadas**.
3. Java, Servlets e Jakarta Server Pages (**JSP**)
4. Acesso à bases de dados com **JDBC** (Java Database Connectivity).
5. Implementação de operações **CRUD** (Create, Read, Update, Delete)
6. Segurança.



Objetivos

- **Geral**
 - Capacitar o aluno na utilização de **procedimentos e técnicas básicas** de desenvolvimento de aplicações para a WEB, com ênfase nos fundamentos dos **recursos nativos da linguagem Java** aplicados ao desenvolvimento **back-end**.
- **Específicos:**
 - Compreender a estrutura de uma aplicação web construída com recursos nativos da linguagem Java;
 - Apresentar uma visão geral do funcionamento de aplicações web baseadas em Servlets e JSP;
 - Permitir ao aluno conhecer e aplicar os recursos básicos necessários para construção de aplicações web com acesso a banco de dados utilizando JDBC;
 - Demonstrar a execução de tarefas relacionadas ao processo de implantação de aplicações web.

Conteúdo programático

Introdução

Programação server-side; Java: sintaxe, modificadores de acesso, estruturas de controle, tipos básicos e arrays; Depuração de apps Java no VS Code; Arquitetura em camadas, MVC e pacotes Java;

Java e POO

Programação orientada a objetos (POO): classes e objetos; Encapsulamento, herança e polimorfismo; Sobrescrita e sobrecarga de métodos;

JDBC

Java Beans; API do JDBC; Sintaxe das principais instruções SQL usadas em operações CRUD; Execução de instruções SQL (Statements e Result Sets); SQL Joins.

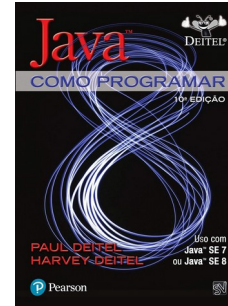
Servlets

Visão geral de Servlets; Servidores de aplicação, empacotamento e implantação web Java; Depuração de webapps Java; JSP: elementos, ações-padrão, diretivas e objetos implícitos; Segurança.

Bibliografia



Web



Java: como programar
10ª Edição - 2016
Editora Pearson
ISBN 9788543004792



Engenharia de Software Moderna
Marco Tulio Valente
<https://engsoftmoderna.info/>

Academy

Sites de referência

- **Jakarta Server Pages Specification**
 - <https://jakarta.ee/specifications/pages/3.1/jakarta-server-pages-spec-3.1.html>
- **Jakarta Servlet Specification**
 - <https://jakarta.ee/specifications/servlet/6.0/jakarta-servlet-spec-6.0.html>



Sites de conteúdo

- **Java e Orientação a Objetos (Caelum/Alura)**
 - <https://www.alura.com.br/apostila-java-orientacao-objetos>
- **Java para Desenvolvimento Web (Alura)**
 - <https://www.alura.com.br/apostila-java-web>
- **Java Tutorial (VS Code)**
 - <https://code.visualstudio.com/docs/java/java-tutorial>
- **Baeldung**
 - <https://www.baeldung.com/>

Ferramentas

- **Visual Studio Code**
 - <https://code.visualstudio.com/Download>
- **Extension Pack for Java (Extensão do VS Code)**
 - <https://marketplace.visualstudio.com/items?vscjava.vscode-java-pack>
- **Java Server Pages - JSP (Extensão do VS Code)**
 - <https://marketplace.visualstudio.com/items?pthorsson.vscode-jsp>
- **XML (Extensão do VS Code)**
 - <https://marketplace.visualstudio.com/items?redhat.vscode-xml>

Ferramentas: JDK 17

- Verificar versão do **JDK** instalada: **javac -version**
 - https://download.oracle.com/java/17/archive/jdk-17.0.6_windows-x64_bin.msi
- Criar a variável de ambiente **JAVA_HOME** configurada para o diretório de instalação do JDK. Exemplo: “C:\Program Files\Java\jdk-17”.
- Adicionar “%**JAVA_HOME**%\bin” na variável de ambiente PATH.
- Tutorial de configuração:
 - https://mkyong.com/java/how-to-set-java_home-on-windows10/

Ferramentas: Maven

- Verificar versão do **Maven** instalada: **mvn -version**
 - <https://maven.apache.org/download.cgi>
- Adicionar o diretório de instalação do Maven na variável de ambiente PATH.
 - Exemplo: “**C:\apache-maven\bin**”.
- Tutorial de configuração:
 - <https://mkyong.com/maven/how-to-install-maven-in-windows>

Ferramentas: Apache Tomcat

- Verifique se o Tomcat está instalado e funcionando:
 - Localize o aplicativo Monitor **Tomcat**
 - Acesse a URL **http://localhost:8080**, que deve exibir uma página indicando que o Tomcat está funcionando.
- Link para download:
 - <https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.7/bin/apache-tomcat-10.1.7.exe>
- Tutorial de instalação:
 - <https://github.com/webacademyufac/tutoriais/blob/main/tomcat/tomcat.md>

Ferramentas: MySQL

- Verificar se o MySQL está funcionando:
 - **mysql -u root -p**
 - Tentar acessar com senha em branco ou senha igual ao nome de usuário (root).
 - Tutorial para reiniciar a senha de root:
<https://dev.mysql.com/doc/mysql-windows-excerpt/8.0/en/resetting-permissions-windows.html>
- Link para download: <https://dev.mysql.com/downloads/file/?id=512698>
- Tutorial de instalação:
<https://github.com/webacademyufac/tutoriais/blob/main/mysql/mysql.md>
- Para criação do banco e importação de dados, a partir do diretório sql, executar os comandos:
 - **mysql -u root -p < sgcm.sql**
 - **mysql -u root -p sgcm < dados.sql**



Web Academy

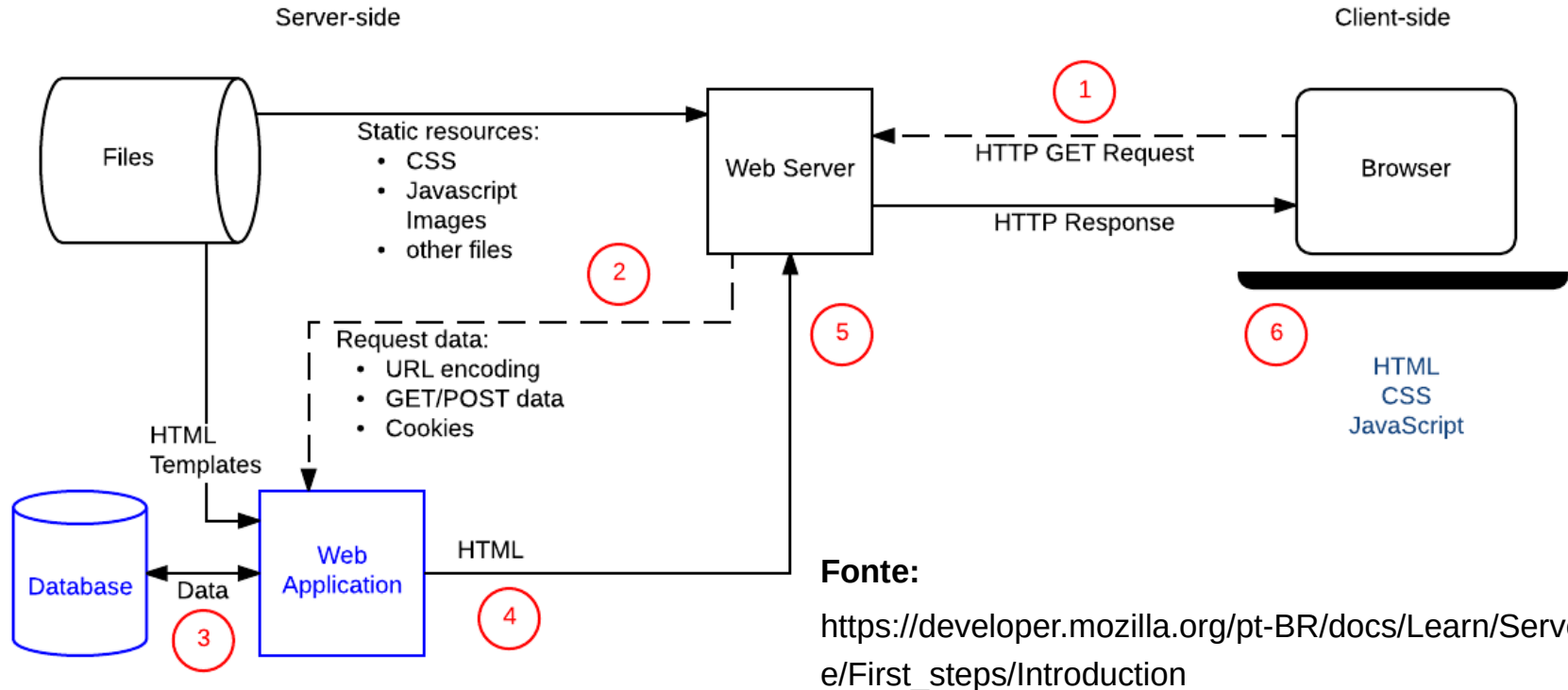


Introdução

Programação server-side

- Em **aplicações web** os navegadores (lado cliente) se comunicam com os servidores por meio do **protocolo HTTP**.
- Sempre que uma ação como a chamada de um link ou envio de formulário é realizada, uma **requisição HTTP** é feita ao servidor.
- Linguagens **client-side** estão ligadas aos aspectos visuais e comportamento da página no navegador, enquanto que linguagens **server-side** estão relacionadas a tarefas como manipular os dados que serão retornados ao cliente.
- Exemplos de linguagem server-side: Java, PHP, Python, C#, JavaScript (Node.js)

Programação server-side



Java: História

- Interessada em dispositivos eletrônicos inteligentes, a **Sun Microsystems** financiou em 1991 o projeto Green.
- Linguagem baseada em C e C++, criada por **James Gosling**, inicialmente chamada de Oak (carvalho).
- Em **1995**, no evento conhecido como SunWorl'95, a Sun apresentou o navegador HotJava e a linguagem Java. No ano seguinte, a NetScape Corp lançou a versão 2 do seu navegador (Navigator), que incorporou a funcionalidade de executar aplicações Java conhecidas como **applets**.
- Em 1996, a Sun liberou de forma gratuita para a comunidade um conjunto de ferramentas para desenvolvimento usando Java (**JDK**).



Java: Plataformas

- A Sun continuou detentora dos direitos até 2009, quando a empresa foi comprada pela Oracle (US\$ 7,4 bilhões) que continuou com a evolução da linguagem e da plataforma.
- A aquisição da Sun não gerou impacto para os desenvolvedores Java, pois a linguagem continua gratuita.
- Java Standard Edition ou **JavaSE**
 - Ambiente para o desenvolvimento de aplicações de pequeno e médio porte, além de um conjunto de APIs (Swing) e a JVM padrão.
- Java Enterprise Edition ou **JavaEE**
 - Componente baseado no desenvolvimento de aplicações empresariais multicamadas de grande porte e provê serviços adicionais, ferramentas e APIs (JPA, JSP) para simplificar a criação de aplicações complexas.





Java: ambiente de desenvolvimento

- Java entrega um ambiente para o desenvolvimento de programas composto por:
 - Uma linguagem de programação de alto nível orientada a objetos;
 - Máquina Virtual (**Java Virtual Machine** ou **JVM**), que garante independência de plataforma, pois o código executa na máquina virtual e essa pode ser portada para outras plataformas como Windows ou Linux;
 - Java Runtime Environment ou **JRE**, que agrega a máquina virtual e alguns recursos para a execução de aplicações Java; e
 - Java Development Kit ou **JDK**, que é um conjunto de utilitários que oferece suporte ao desenvolvimento de aplicações.

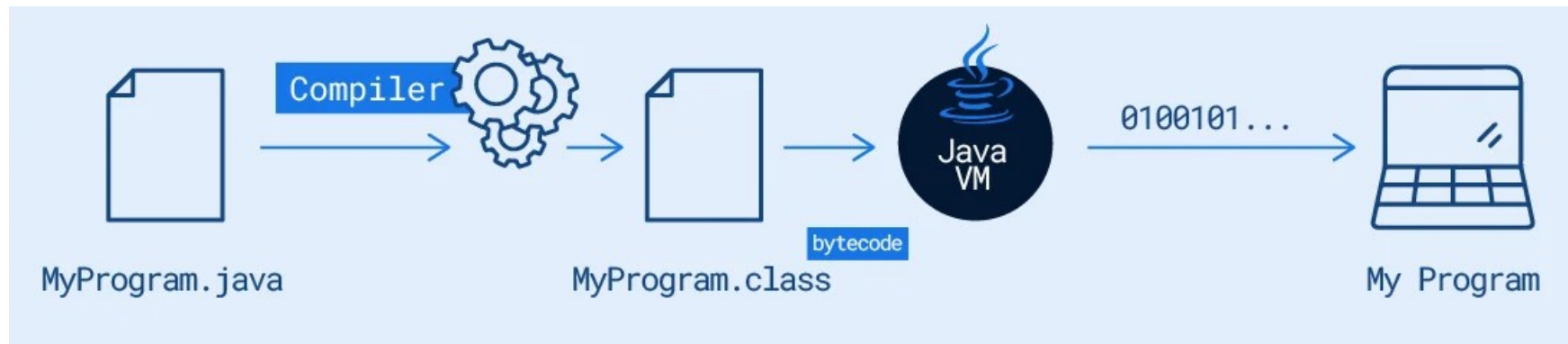
A decorative header at the top of the slide featuring a dark background with snippets of code in various colors (purple, green, red, white). The code appears to be from different programming languages, including what looks like JavaScript or JSON, and some shell-like commands.

Java: características

- Independência de plataforma (portabilidade)
- Orientação a Objetos
- Não usa ponteiros
- Multithread
- Segurança
- Recursos de rede
- Gerência automática de memória
- Sintaxe similar a C/C++

Java: programas

- Em Java, os programas são escritos em um arquivo com a extensão **.java**, que em um processo posterior serão compilados para arquivos com a extensão **.class**. Esses, por sua vez, contêm os códigos a serem executados na máquina virtual, os **bytecodes**.



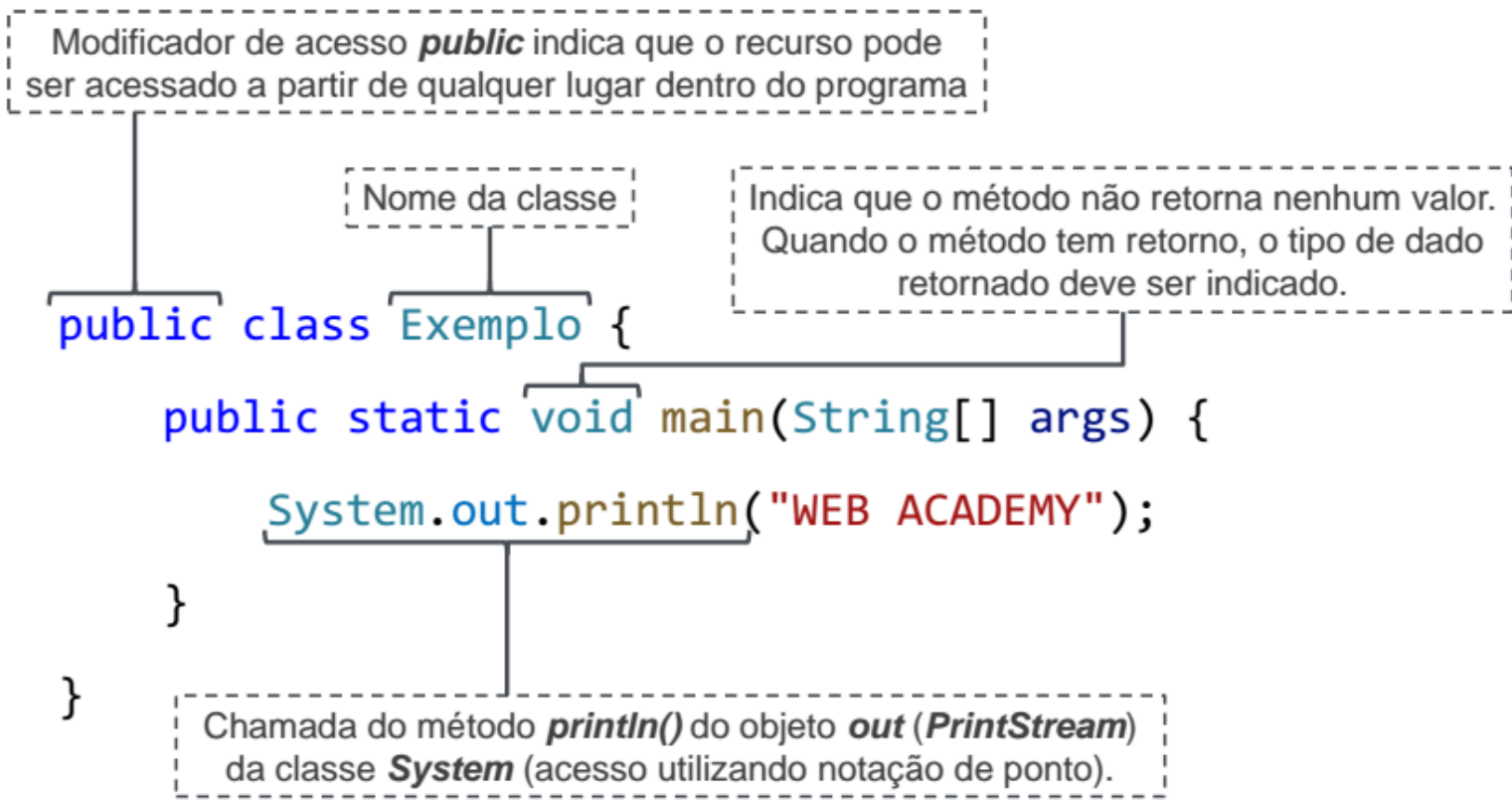
Java: exemplo

- O processo criação e execução de um aplicativo Java pode ser resumido normalmente nos seguintes passos:
 - Escrita do código-fonte (arquivo .java);
 - Compilação do programa Java em bytecodes, gerando os arquivos .class;
 - Carregamento do programa na memória pela JVM (Máquina Virtual Java);
 - Verificação de bytecode pela JVM;
 - Execução do programa pela JVM.

```
public class Exemplo {  
    public static void main(String[] args) {  
        System.out.println("WEB ACADEMY");  
    }  
}
```

```
# javac Exemplo.java  
  
# java Exemplo  
  
WEB ACADEMY
```


Java: anatomia



Java: modificadores de acesso

- **public**: permite que a classe, método ou variável seja acessado por qualquer código em **qualquer lugar**.
- **private**: permite que a classe, método ou variável seja acessado somente **dentro da própria classe** onde foi definido.
- **protected**: permite que a classe, método ou variável seja acessado **dentro da própria classe, subclasses e outras classes no mesmo pacote**.
- **default** (ou **package-private**): permite que a classe, método ou variável seja acessado somente **dentro do mesmo pacote**.

Java: tipos de dados

- Java é uma linguagem de tipagem **forte** e **estática**, portanto, requer que todas as variáveis tenham um tipo e não permite operações diretas entre tipos diferentes.
- Tipos primitivos: boolean, char, byte, short, int, long, float, double.

```
public class Exemplo {  
    public static void main(String[] args) {  
        int x = 10;  
        x = "WEB ACADEMY";  
        mensagem = "WEB ACADEMY";  
        String mensagem = "WEB ACADEMY";  
        System.out.println(mensagem);  
    }  
}
```

```
# javac Exemplo.java  
Exemplo.java:4: error: incompatible types: String  
cannot be converted to int  
        x = "WEB ACADEMY";  
          ^  
Exemplo.java:5: error: cannot find symbol  
        mensagem = "WEB ACADEMY";  
          ^  
symbol:   variable mensagem  
location: class Exemplo  
2 errors
```

Java: casting

PARA:	byte	short	char	int	long	float	double
DE:							
byte	----	<i>Impl.</i>	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
short	(byte)	----	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
char	(byte)	(short)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
int	(byte)	(short)	(char)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
long	(byte)	(short)	(char)	(int)	----	<i>Impl.</i>	<i>Impl.</i>
float	(byte)	(short)	(char)	(int)	(long)	----	<i>Impl.</i>
double	(byte)	(short)	(char)	(int)	(long)	(float)	----

Java: estruturas de controle

```
int numero = 1;
String mensagem;

// if/else
if (numero == 1) {
    mensagem = "Igual a 1";
} else {
    mensagem = "Maior ou igual 2";
}
System.out.println(mensagem);

// Operador ternário
mensagem = (numero > 3) ? "Maior que 3" : "Menor ou igual a 3";
System.out.println(mensagem);
```

Diagram illustrating the ternary operator logic:

Condição	true	false
(numero > 3)	"Maior que 3"	"Menor ou igual a 3"

Java: arrays

- Arrays são estruturas de dados que permitem armazenar e manipular coleções de elementos do mesmo tipo.
- Tipos de arrays dinâmicos: ArrayList, LinkedList, Vector, Stack, Queue, Deque.

```
// Declaração de array estático de 5 posições
int[] numeros = new int[5];

// Declaração de array dinâmico
List<Integer> numeros = new ArrayList<Integer>();

// Acessando um elemento do array estático
int numero = numeros[1];

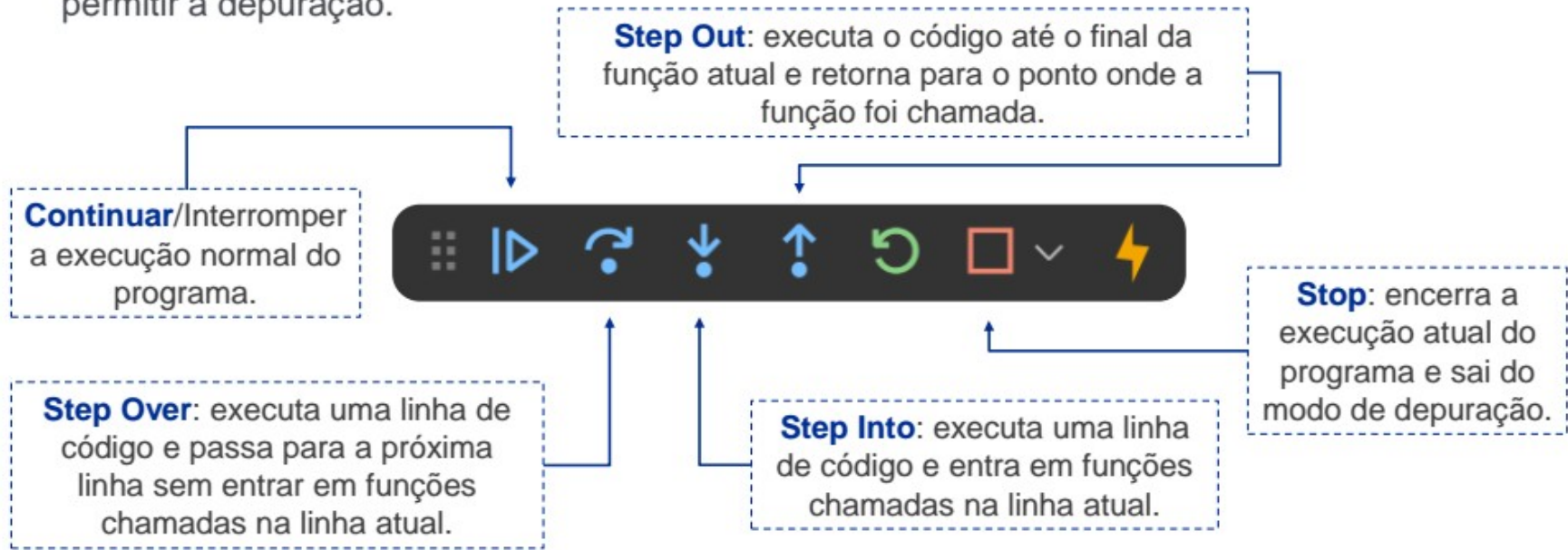
// Acessando um elemento do array dinâmico
int numero = numeros.get(1);
```

```
// Percorrendo arrays pelo índice
for (int i = 0; i < numeros.length; i++) {
    System.out.println(numeros[i]);
}

// Percorrendo arrays com loop for-each
for (int numero : numeros) {
    System.out.println(numero);
}
```

Depuração de apps Java no VS Code

- **Breakpoints:** pontos definidos no código onde a execução do programa é interrompida para permitir a depuração.





Depuração de apps Java no VS Code

- Referências úteis:
 - <https://code.visualstudio.com/docs/editor/debugging>
 - <https://code.visualstudio.com/docs/java/java-debugging>