Introdução ao método main()

Nas aulas de Programação Orientada a Objetos, estudamos sobre o que são Classes, Objetos e Métodos. Lógico que, embora observamos um pouco de código, ainda não nos aprofundamos, apenas realizamos um “reconhecimento do gramado”.

O nosso objetivo nas aulas é aprender em detalhes sobre a linguagem e o que ela pode oferecer. Por isso, fique atento ao conteúdo teórico, pois essas informações podem te ajudar futuramente.

Antes de tudo, vamos começar a nossa sequência de conteúdo falando do método main().

1. Main()

Este método é o ponto de entrada para a execução de um programa Java. A tradução da palavra “main” para o português é “principal”, então sabemos que este método é importante para a aplicação, porque ele serve como um ponto de entrada para o programa.

Este método pode conter o código para executar ou chamar outros métodos, e em aplicações mais robustas, esse método pode conter a sua própria classe, que em muitos casos, recebe o nome do método.

1.1 Sintaxe

Veja abaixo como a sintaxe do método main é simples:

Public static void main(String[] args){

// Código

}

Mas o que significa todos esses nomes, como void, Public, static, e outros nomes? Vamos entender o que eles significam.

1.2 Public

Quando estudamos Programação Orientada a Objetos, estudamos sobre os modificadores de acesso. Mas o que são os modificadores de acesso?

São palavras-chaves que define a visibilidade, ou o acesso de uma classe. Essas palavras são: public, private, protected.

No método main em nosso exemplo, o modificador definido foi o public. Caso o modificador não seja definido, o programa executará um erro porque, quando o método main não é público, o JRE não o encontrará.

1.3 Static

Quando o programa Java é iniciado, não há objetos presente na classe. Usar static no método main possibilita que a JVM possa carregar a classe na memória chamar o método main sem criar uma instancia da classe.

Executar a aplicação sem a definição do static fará o programa executar um erro porque a main não é static.

1.4 Void

Os métodos em Java precisam fornecer um tipo de retorno, e definir o método main com void indica que o método main não retorna nenhum valor.

Caso tenha uma definição de retorno, o programa executa um erro porque o Java não espera um valor de retorno quando o tipo de retorno é void.

1.5 Main()

Quando a aplicação Java inicia, ela sempre procura o método main. Ao compilar a aplicação, o programa executará um erro, onde o JRE não encontra o método main na classe.

1.6 String[] args

O método main aceita um único argumento array do tipo String, essa forma, permite que argumentos de linha de comando sejam passados ao programa.

String[] indica que args é um parâmetro do tipo array de strings, indicando que pode conter um conjunto de strings, onde cada elemento da matriz é uma string.

Args é um parâmetro que pode ser qualquer nome em Java. Este nome é normalmente utilizado, mas é possível usar qualquer outro nome valido.

Referências

public static void main(String[] args) - Método principal Java.

Acesso: 25/05/2024

<https://www.digitalocean.com/community/tutorials/public-static-void-main-string-args-java-main-method>

Modificadores de Acesso em Java.

Acesso: 25/05/2024

<https://www.devmedia.com.br/modificadores-de-acesso-em-java/18822>

O que é de fato o "public static main(String[] args)".

Acesso: 25/05/2024

<https://www.dio.me/articles/o-que-e-de-fato-o-public-static-mainstring-args#:~:text=public%20%C3%A9%20um%20modificador%20de,outro%20elemento%20dentro%20do%20c%C3%B3digo>