Aula 4 – Métodos I e II

Prof. Me. Rodrigo Brito Battilana

Roteiro

- Introdução
- Estruturas de controle
- A instrução de seleção única if
- A instrução de seleção dupla if . . . Else
- A instrução de Seleção Switch
- A instrução de repetição while
- A Instrução de repetição for

Roteiro

- Métodos especiais
  - Métodos get e set
  - Métodos staticos
  - Campos static
- Classe Math
- Métodos com múltiplos parâmetros
- Referências

Introdução

- A experiência mostrou que a melhor maneira de desenvolver e manter um programa grande é construí-lo a partir de pequenos e simples pedaços, ou **módulos**.

- Essa técnica é chamada **dividir para conquistar**.

Introdução

- Os métodos são declarados dentro de classes. Em geral, as classes são agrupadas em pacotes para que possam ser importadas e reutilizadas.

- Os métodos permitem modularizar um programa separando suas tarefas em unidades autocontidas.

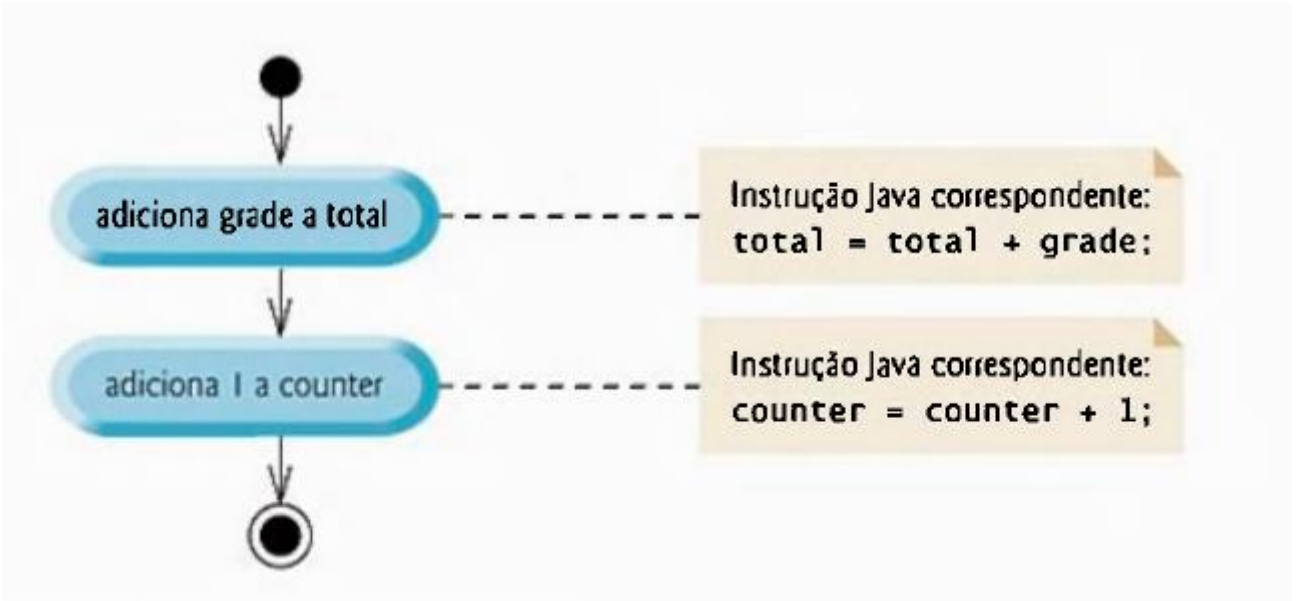
Introdução

- As instruções em um método são escritas somente uma vez e permanecem ocultas de outros métodos.

- Usar os métodos existentes como blocos de construção para criar novos programas é uma forma de **reutilização de software** que permite evitar repetição de código dentro de um programa.

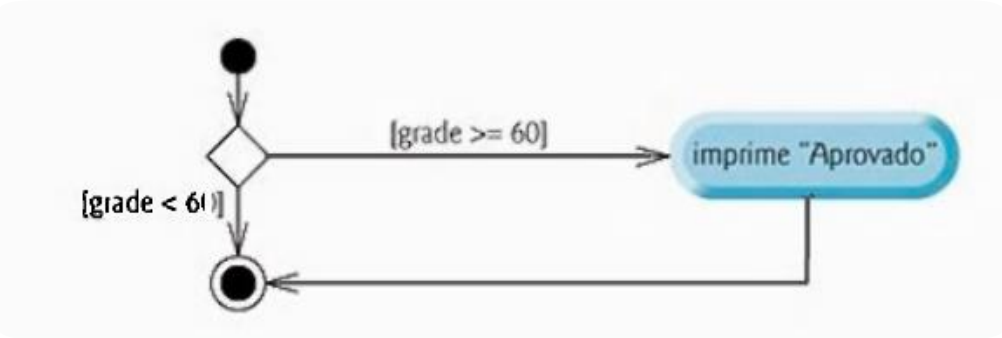
Estruturas de controle

- Estruturas de controle são utilizadas para controlar o fluxo de dados.



A instrução de seleção única if

- Estrutura do if permite que o programa faça um desvio no fluxo de execução dada uma determinada condição.



Ex:

```
if ( studentGrade >= 6.0 )
System.out.println(
"Passed" );
```

A instrução de seleção dupla if . . . Else

- Estrutura do if permite que o programa faça um desvio no fluxo de execução dada uma determinada condição.

Ex:

```
i f ( studentGrade >= 6.0 )
        S y s t e m . o u t . p r i n t l n ( "Passed" );
Else
S y s t e m . o u t . p r i n t l n ( “You shall not pass!!!" );
```

A instrução de Seleção Switch

- A estrutura de seleção do switch case permite que uma ou mais ações sejam executadas quando uma opção for escolhida.

Ex:

```
Switch(op){
  case 1:
            ação;
            break;
  case 2:
          ação 2;
            break;
Default:
            Ação default;
}
```

A instrução de repetição while

- A instrução de repetição do while é utilizada para repetir uma ou mais ações dependendo da condição.

Ex:

```
While(condição){
        ação;
}
```

A Instrução de repetição for

- A instrução de repetição do for é utilizada para repetir uma ou mais ações uma quantidade de vezes definida.

Ex:

```
For(int i=0;i < condição; i++){
   ação
}
```

Módulos de programa em Java

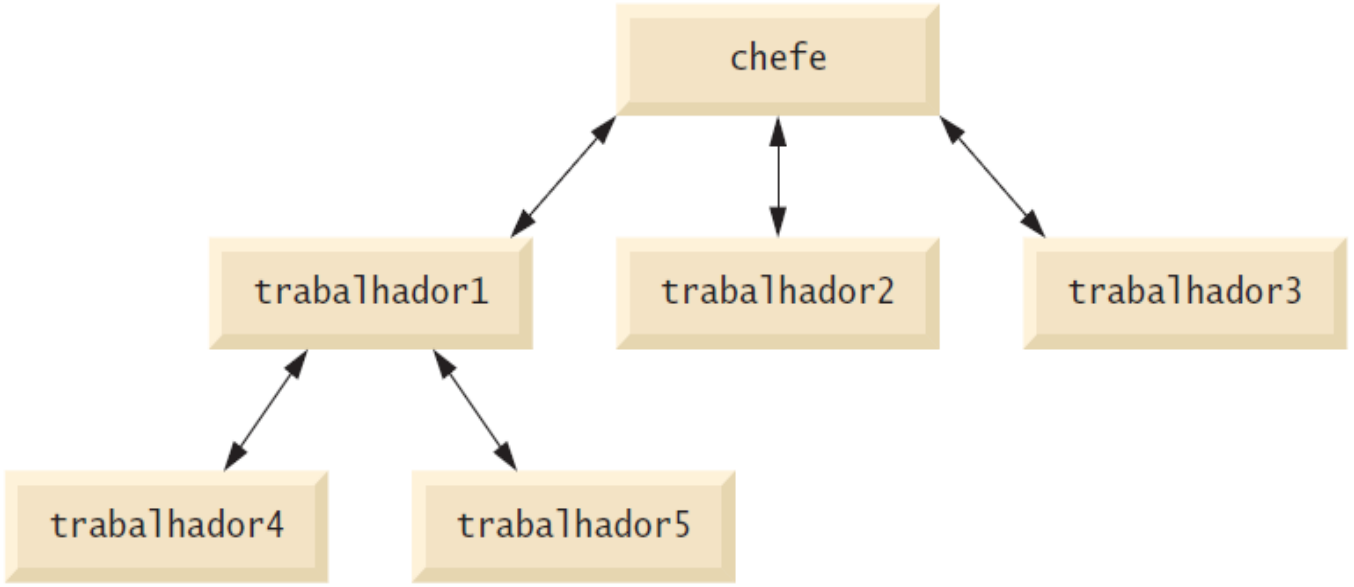


Figura 6.1 | Relacionamento hierárquico de método trabalhador/método chefe.

Métodos especiais

- Métodos get

Os métodos get são utilizados para recuperar um valor de um método ou variável que esteja com o modificador de acesso private.

Ex:

```
Tipo getNomeDoMetodo(){
   return tipo;
}
```

Métodos especiais

- Métodos set

Os métodos set são utilizados para alterar um valor de um variável que esteja com o modificador de acesso private.

Ex:

```
Tipo setNomeDoMetodo(tipo variavel){
   variávellocal = variável;
}
```

Métodos `static`, campos `static`

- Uma classe pode conter métodos `static` para realizar tarefas comuns que não exigem um objeto da classe.

- Quaisquer dados que um método `static` poderia requerer para realizar suas tarefas podem ser enviados ao método como argumentos em uma chamada de método.

Métodos `static`, campos `static`

- Um método `static` é chamado especificando o nome da classe em que o método é declarado seguido por um ponto (.) e pelo nome do método, como em:

NomeDaClasse.nomeDoMétodo(*argumentos*)

classe `Math`

- A classe `Math` fornece os métodos `static` para realizar cálculos matemáticos comuns.

- A constante `Math.PI` (3,141592653589793) é a relação entre a circunferência de um círculo e seu diâmetro.

classe `Math`

- A constante `Math.E` (2,718281828459045) é o valor base para logaritmos naturais (calculados com o método `static Math log`).

- `Math.PI` e `Math.E` são declaradas com os modificadores `public`, `final` e `static`. Torná-los `public` permite que você use esses campos nas suas próprias classes.

classe `Math`

- Um campo declarado com a palavra-chave `final` é constante — seu valor não pode ser alterado depois de ele ser inicializado.

- Tanto `PI` como `E` são declarados `final` porque seus valores nunca mudam.

- Tornar esses campos `static` permite que eles sejam acessados pelo nome da classe `Math` e um ponto (`.`) separador, como ocorre com os métodos da classe `Math`.

Campos `static`

- Todos os objetos de uma classe compartilham uma cópia dos campos `static` da classe. As variáveis de classe e as variáveis de instância representam os campos de uma classe.

- Quando você executa a Java Virtual Machine (JVM) com o comando `java`, a JVM carrega a classe especificada e utiliza esse nome de classe para invocar o método `main`.

Campos `static`

- É possível especificar **argumentos de linha de comando** adicionais que a JVM passará para o seu aplicativo.

- Você pode colocar um método `main` em cada classe que declara – somente o método `main` na classe que você utiliza para executar o aplicativo será chamado pelo comando `java`.

Declarando métodos com múltiplos parâmetros

- Quando um método é chamado, o programa faz uma cópia dos valores de argumento do método e os atribui aos parâmetros correspondentes do método.

- Quando o controle do programa retorna ao ponto em que método foi chamado, os parâmetros do método são removidos da memória.

Declarando métodos com múltiplos parâmetros

- Um método pode retornar no máximo um valor, mas o valor retornado poderia ser uma referência a um objeto que contém muitos valores.

- Variáveis devem ser declaradas como campos de uma classe somente se forem utilizadas em mais de um método da classe ou se o programa deve salvar seus valores entre chamadas aos métodos da classe.

Declarando métodos com múltiplos parâmetros

- Se um método tiver mais de um parâmetro, os parâmetros serão especificados como uma lista separada por vírgulas.

- Deve haver um argumento na chamada de método para cada parâmetro na declaração do método.

Declarando métodos com múltiplos parâmetros

- Todos os objetos em Java têm um método especial chamado `toString` que retorna uma representação `String` do conteúdo do objeto.

- Quando um objeto é concatenado com uma String, a JVM chama implicitamente o método `toString` do objeto a fim de obter a representação `String` do objeto.

Declarando métodos com múltiplos parâmetros

- Pode-se dividir grandes literais `String` em várias `Strings` menores e colocá-las em múltiplas linhas de código para melhorar a legibilidade, depois remontar as `Strings` utilizando concatenação.

Referências

- Deitei, Harvey M. **Java C o m o Programar**, 8ªed tradução Edson Furmankiewicz ; São Paulo, Editora: Pearson Prentice Hall, 2010.