

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by two horizontal dark gray bars, one at the top and one at the bottom.

ORACLE

Academy

Java Fundamentals

7-3

Modificador estático e classes

ORACLE
Academy



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Objetivos

- Esta aula abrange os seguintes tópicos:
 - Criar variáveis estáticas
 - Usar variáveis estáticas
 - Criar métodos estáticos
 - Usar métodos estáticos
 - Criar classes estáticas
 - Usar classes estáticas



Modificador Estático

- Usando variáveis de instâncias, cada instância de uma classe criada com a palavra-chave new cria uma cópia de todas as variáveis de instâncias nessa classe
- Por exemplo, na classe Employee abaixo, uma cópia exclusiva de lastname e firstname é criada para cada novo objeto Employee que é criado em uma classe Driver

```
public class Employee{  
    private String lastname;  
    private String firstname;  
    ...more code  
} //fim da classe Employee  
//create two Employees in a main method:  
Employee e1 = new Employee("Smith", "Mary");  
Employee e2 = new Employee("Jones", "Sally");
```

É prática comum ter os campos como privados para protegê-los do acesso direto. Se necessário, forneceríamos métodos acessor e modificador para acesso a esses campos.

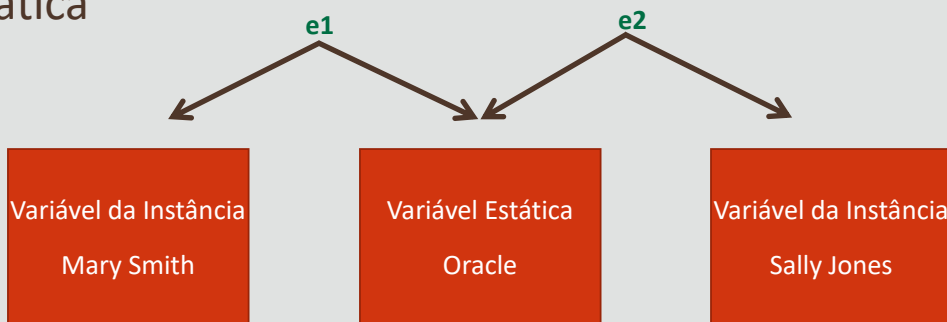
Palavra-chave Static

- Static é uma palavra-chave em Java que modifica a associação de um item com uma classe
- O conteúdo de uma classe que é identificado como estático é compartilhado entre todas as instâncias da classe
- Isso significa que todas as instâncias da classe compartilham uma cópia dos itens estáticos, e cada uma tem suas próprias cópias de itens da instância ou de itens não estáticos

Quando adicionamos o modificador estático a um campo/variável, isso normalmente é denominado de variável de classe. Onde o valor é armazenado apenas uma vez e é acessível por todas as instâncias de classe.

Exemplo de Static

- Considere inicializar uma String estática com o valor “Oracle” chamado myCompany que representa a empresa do empregador
- Cada instância de Employee ainda teria suas próprias variáveis de instâncias, mas compartilharia a variável estática



Então, a alteração do texto no mesmo campo de e1 afetaria apenas essa instância. Se atualizássemos e1 para myCompany todas as outras instâncias de Funcionário fariam referência ao mesmo valor, depois, elas também veriam essa mudança.

Variáveis Estáticas

- Variáveis estáticas
 - Também são chamadas de variáveis de classes
 - São declaradas com a palavra-chave static
 - Têm somente uma cópia na memória, diferentemente das variáveis de instâncias, que contêm uma cópia por instância
 - São compartilhadas por instâncias de objetos
 - Contêm o mesmo valor para todas as instâncias de classes

Os valores estáticos são úteis para tarefas como manter a contagem de quantos objetos são criados quando você só precisa de uma cópia de constantes úteis ou de valores comuns que será usada em vários objetos etc.

Variáveis Estáticas

- Acesso público para variáveis estáticas:
 - Se forem públicas, elas poderão ser modificadas diretamente por outras classes
 - Torne a variável uma constante usando a palavra-chave final para evitar modificações
 - Exemplo:

```
public static final int MODEL_NUM = 883;
```

Embora haja uma constante de palavra-chave Java, ela não é usada. final é usado para constantes em Java.

Práticas de Programação e Variáveis Estáticas

- A boa prática de programação inicializa variáveis estáticas com valores, em vez de confiar nos valores padrão nulo e 0
- Os valores inicialmente atribuídos podem ser alterados, contanto que a classe esteja ativa na memória da JVM
- A coleta de lixo remove-a da memória, e os valores iniciais atribuídos retornam na próxima utilização

Declarando uma Variável Estática

- Para declarar uma variável estática, inclua a palavra-chave `static`, conforme mostrado abaixo
 - Pode ser pública, protegida, padrão ou privada
 - Deve ter valores atribuídos, mas são atribuídos automaticamente valores nulos para instâncias de classes: uma string vazia ou 0 para números primitivos
 - Devem agir como constantes com a palavra-chave `final` quando usam um acesso público, protegido ou padrão

```
public class Nesting {  
    // Declare public static variable.  
    public static final int MODEL_NUM = 883;  
    ...  
} //fim da classe Nesting
```

A palavra-chave `final` não precisa ser usada com variáveis estáticas. Isso simplesmente impõe a condição de que a variável não pode ser alterada até que um valor tenha sido definido.

Alterações em uma Variável Estática

- As variáveis estáticas que não são finais podem ser lidas ou receber novos valores usando a palavra-chave opcional `this` em métodos de instâncias
 - As alterações feitas por métodos de instâncias são feitas em todas as instâncias
 - Uma alteração em uma variável estática pode indicar que a classe deve ser limitada a somente um objeto
 - Isso é conhecido como o padrão Singleton

```
private static String myCompany = "Oracle";  
public void setMyCompany(String s) {  
    this.mycompany = s;  
} //fim do método myCompany  
...
```

O padrão Singleton força a ideia de que só pode haver uma instância de uma classe. Na primeira chamada a essa classe, uma nova instância é criada. As futuras chamadas não criam novas instâncias, mas retornam a que foi criada.

Exemplo de Variável Estática

- Crie uma classe chamada Turtle que contenha uma variável chamada comida
 - Essa variável é estática, pois todas as nossas tartarugas comem a mesma comida
 - A classe Turtle terá mais uma variável chamada idade
 - Como cada tartaruga tem uma idade diferente, é melhor que esta variável seja uma variável de instância privada em vez de estática

```
public class Turtle {  
    public static String food = "Turtle Food";  
    private int age;  
  
    public Turtle(int age) {  
        this.age = age;  
    } //fim construtor  
} //fim da classe Turtle
```



Acessando Variáveis Estáticas

- As variáveis de instância requerem que haja uma instância da classe antes que o acesso seja possível

```
public class Turtle {  
    public static String food = "Turtle Feed";  
    private int age;  
    ...  
} //fim da classe Turtle
```

- Você pode acessar variáveis estáticas sem criar uma instância da classe
- Em um método principal, essa instrução imprimiria a variável "food" sem nenhuma referência de instância

```
System.out.println("I feed " + Turtle.food  
                    + " to all of my turtles!");
```

De fato, as práticas recomendadas de programação dizem que você sempre deve usar o nome da classe para acessar variáveis estáticas não o nome da instância.

Notação para Acessar Variáveis Estáticas

- Geralmente, as variáveis estáticas são acessadas pela notação:

```
ClassName.variableName;
```

Modificador Estático e Métodos

- Os métodos estáticos ou de classe existem uma vez e são compartilhados por todas as instâncias de classes
 - Podem ser usados por outros métodos de classe ou de instâncias com base nos seus modificadores de acesso
 - Não podem acessar variáveis não estáticas ou de instâncias
 - Os métodos estáticos só podem acessar variáveis estáticas
 - Não podem acessar métodos não estáticos ou de instâncias
 - Os métodos estáticos só podem acessar outros métodos estáticos
 - Podem ser redefinidos em subclasses
 - Podem ser públicos, protegidos, padrão ou privados

Modificador Estático e Métodos

- Há uma diferença entre chamar um método de instância e um método de classe (estático)
- Por exemplo
 - Você deve primeiro criar uma instância e usar uma notação de pontos para chamar um método de instância; no entanto, o nome da classe, uma notação de pontos e o nome do método estático chama um método estático

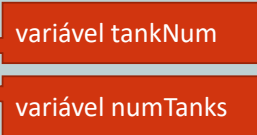
Modificador Estático e Métodos

- O método estático fornece um mecanismo para construir uma instância de uma classe
- Quando a classe tem um construtor de acesso privado, um método estático é uma de duas abordagens para criar uma instância da classe

Exemplo de Classe Turtle

- A classe Turtle tem uma variável estática que identifica o número de aquários que temos disponíveis (numTanks) e uma variável de instância (tankNum) que indica em qual aquário a Turtle está

```
public class Turtle {  
    public static String food = "Turtle Feed";  
    public int age;  
    public int tankNum;   
    public static int numTanks = 3;  
    public Turtle(int age) {  
        this.age = age;  
        tankNum = (int) ((Math.random() * numTanks) + 1);  
    } //fim construtor  
    public void swim() { //implementation  
    public int getAge() { return age; }  
    public int getTankOfResidence() { return tankNum; }  
    public static String fishTank() { return "I have " + numTanks  
                                         + " fish tanks."; }  
    } //fim do método fishTank  
} //fim da classe Nesting
```



Exemplo de Método Estático na Classe Turtle

- Analise os métodos na classe Turtle



```
public class Turtle {  
    public static String food = "Turtle Feed";  
    public int age;  
    public int tankNum;  
    public static int numTanks = 3;  
    public Turtle(int age){  
        this.age = age;  
        tankNum = (int)((Math.random()*numTanks)+1);  
    } //fim construtor  
    public void swim(){//implementation}  
    public int getAge(){return age;}  
    public int getTankOfResidence(){return tankNum;}  
    public static String fishTank() {return "I have " + numTanks + " fish tanks.";}  
    }//fim do método fishTank  
}//fim da classeTurtle
```

swim() é um método de instância. Embora cada tartaruga saiba nadar, elas podem nadar de maneira diferente dependendo da idade.

fishTank() é um método estático e acessa uma variável estática (numTanks).

getAge() e getTankOfResidence() são métodos de instância não estáticos porque acessam variáveis não estáticas. Os métodos estáticos não podem acessar itens não estáticos.

Criando Instâncias de Classe Usando Métodos Estáticos

- Outro uso de métodos estáticos é para criar instâncias de classe quando o acesso do construtor de classes é privado, e o método faz parte da mesma classe
- Isso é possível porque o método estático é publicamente acessível com acesso privado à classe

```
...  
private Nesting() {...implementation...}  
...  
public static Nesting getInstance() {  
    Nesting nesting = new Nesting();  
    return nesting; }  
...  
// Instantiate a private class with a method.  
Nesting n1 = Nesting.getInstance();  
...
```

Modificador Estático e Classes

- Classes estáticas ou aninhadas:
 - Podem existir como classes aninhadas
 - Não podem existir como classes independentes



Uma classe aninhada é uma classe criada dentro de outra classe.



Há mais algumas maneiras de aninhar classes em Java que não são abordadas neste curso.

Classes Aninhadas Estáticas

- Classes Aninhadas Estáticas

- Classes estáticas são implementadas dentro de outras classes, e as outras classes são conhecidas como classes de container
- Podem estender o comportamento da classe de container
- Podem ser sobrecarregadas como construtores comuns

Classes Aninhadas Estáticas

- A classe aninhada estática também fornece os meios para instanciar uma classe contida quando o construtor é configurado com acesso privado
- Esta é a segunda maneira de instanciar uma classe que tem um qualificador de acesso restrito ou privado para os construtores da classe

Exemplo de Classes Aninhadas Estáticas

```
public class Space {  
    //Space class variables  
  
    public static class Planet{  
        //planet class variables and constructors  
        public Planet() {  
            ...implementation...  
        } //fim construtor  
  
        public Planet(String, int size){  
            ...implementation...  
        } //fim construtor  
    } //fim da classe Planet  
  
    //more space class implementation  
} //fim da classe Space
```


Terminologia

- Os principais termos usados nesta aula foram:
 - Métodos de classe
 - Variável de classe
 - Classe interna
 - Classe aninhada
 - Modificador estático
 - Método estático
 - Classe aninhada estática
 - Variável estática

Resumo

- Nesta aula, você deverá ter aprendido a:
 - Criar variáveis estáticas
 - Usar variáveis estáticas
 - Criar métodos estáticos
 - Usar métodos estáticos
 - Criar classes estáticas
 - Usar classes estáticas



The Oracle Academy logo is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font, with the letters slightly spaced out. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by two horizontal dark gray bars, one at the top and one at the bottom.

ORACLE

Academy