

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

ORACLE

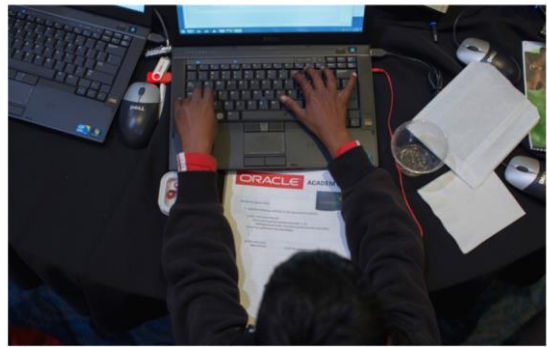
Academy

Java Fundamentals

7-3:

Modificador estático y clases anidadas

ORACLE
Academy



Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

Objetivos

- Esta lección abarca los siguientes temas:
 - Creación de variables estáticas
 - Uso de variables estáticas
 - Creación de métodos estáticos
 - Uso de métodos estáticos
 - Creación de clases estáticas
 - Uso de clases estáticas



Modificador estático

- Mediante el uso de variables de instancia, cada instancia de una clase creada con la palabra clave `new` crea una copia de todas las variables de instancia en esa clase
- Por ejemplo en la clase `Empleado` que figura más adelante, se crea una copia exclusiva del apellido y primer nombre para cada nuevo objeto `Empleado` que se cree en una clase de controlador

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

ORACLE
Academy

JF 7-3
Modificador estático y clases anidadas

Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

4

Es una práctica habitual hacer que los campos aparezcan como privados para evitar que se pueda acceder a ellos directamente. Proporcionaríamos métodos de acceso o mutadores para acceder a estos campos si fuese necesario.

Palabra clave estático

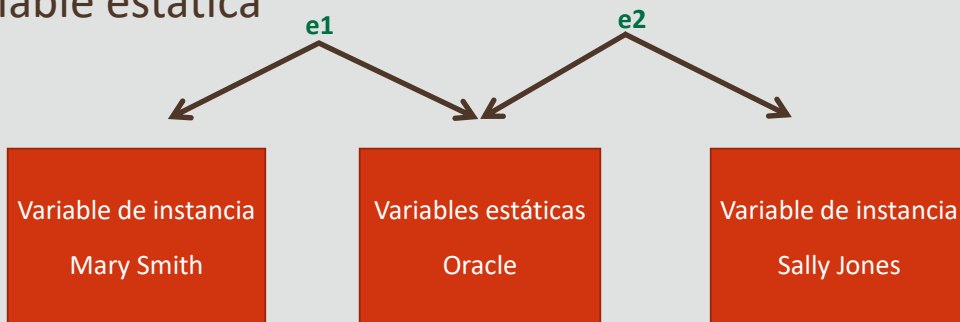
- Estático es una palabra clave en Java que modifica la asociación de un elemento con una clase
- El contenido de una clase que se identifica como estático se comparte en todas las instancias de la clase
- Esto significa que todas las instancias de esa clase comparten una copia de los elementos estáticos y cada una posee su propias copias exclusivas de elementos de instancia o elementos no estáticos

Agregar el modificador estático a un campo/variable se conoce normalmente como variable de clase.

El valor solo se almacena una vez y pueden acceder a él todas las instancias de clase.

Ejemplo de estáticos

- Considere la inicialización de un String estático con el valor “Oracle” denominado myCompany, que representa a la empresa del empleador
- Cada instancia de Empleado tendrá todavía sus variables de instancia exclusivas, pero compartirá la variable estática



Así pues, si se cambia el texto en el campo de nombre de e1, afectaría a esa instancia únicamente. Si hemos actualizado e1 para myCompany, puesto que todas las demás instancias de Employee hacen referencia al mismo valor, también verían este cambio.

Variables estáticas

- Variables estáticas
 - Conocidas también como variables de clases
 - Se declaran con la palabra clave estática
 - Tienen solamente una copia en la memoria, a diferencia de las variables de instancia, que conservan una copia por instancia
 - Se comparten mediante instancias de objetos
 - Conservan el mismo valor para todas las instancias de clase

Los valores estáticos son útiles para tareas como el mantenimiento del recuento de objetos que se crean, cuando solo necesita una copia de las constantes o valores comunes útiles que se utilizarán en varios objetos, etc.

Variables estáticas

- Acceso público para variables estáticas:
 - Si el acceso es público, pueden ser modificadas directamente por otras clases
 - Considere convertir a la variable en constante mediante el uso de la palabra clave final, a fin de evitar modificaciones
 - Ejemplo:

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

A pesar de que hay una palabra clave Java (const), no se utiliza. Se utiliza final para las constantes en Java.

Programación de prácticas y variables estáticas

- En programación se recomienda inicializar a las variables estáticas con valores, en lugar de confiar en los valores predeterminados nulo y 0
- Los valores inicialmente asignados se pueden cambiar en la medida en que la clase se encuentre activa en la memoria de JVM
- La recolección de elementos no utilizados los elimina de la memoria y los valores iniciales asignados aparecen nuevamente la próxima vez que la utilice

Declaración de una variable estática

- Para declarar una variable estática, incluya la palabra clave estática, como se indica a continuación
 - Puede ser pública, protegida, determinada o privada
 - Deberá tener valores asignados, pero automáticamente se le asignan valores nulos para las instancias de clase: un String vacío o 0 para números primitivos
 - Deberán actuar como constantes con la palabra clave final cuando usen un acceso público, protegido o determinado

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

La palabra clave final no tiene que utilizarse con las variables estáticas. Esto simplemente obliga a que la variable no se pueda cambiar una vez que se establece un valor.

Cambios a una variable estática

- Las variables estáticas que no sean finales se pueden leer o se les pueden asignar nuevos valores mediante el uso de la palabra clave `this` en los métodos de instancia
 - Los cambios realizados por métodos de instancia se cambian para todas las instancias
 - Un cambio a una variable estática puede indicar que la clase deberá estar limitada a un solo objeto
 - Se conoce como el patrón Singleton

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

El patrón Singleton fuerza la idea de que solo puede haber una instancia de una clase. En la primera llamada a esta clase, se crea una nueva instancia. Todas las llamadas futuras no crearán nuevas instancias, pero devolverán la que se creó.

Ejemplo de variables estáticas

- Crear una clase denominada Tortuga que contenga una variable denominada alimento
 - Esta variable es estática dado que todas nuestras tortugas comen el mismo alimento
 - La clase Tortuga incluirá otra variable más, denominada edad
 - Dado que cada tortuga tiene una edad diferente, se recomienda convertir a esta variable en una variable de instancia privada en lugar de una variable estática

```
public class Turtle {  
    public static String food = "Turtle Food";  
    private int age;  
  
    public Turtle(int age) {  
        this.age = age;  
    } //end constructor  
} //end class Turtle
```



Acceso a las variables estáticas

- Las variables de instancia requieren que exista una instancia para una clase antes de permitir el acceso

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

- Usted puede acceder a las variables estáticas sin crear una instancia para una clase
- En un método main, esta sentencia imprimiría la variable alimento sin ninguna referencia a la instancia

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

De hecho, las buenas prácticas de programación indican que debe utilizar siempre el nombre de la clase para acceder a variables estáticas y no el nombre de una instancia.

Notación para acceder a variables estáticas

- En general, se accede a las variables estáticas mediante la notación:

```
ClassName.variableName;
```

Modificador y métodos estáticos

- Los métodos estáticos o de clase existen una sola vez y se comparten entre todas las instancias de clase
 - Pueden ser utilizados por otros métodos de clase o métodos de instancia según su modificador de acceso
 - No pueden acceder a las variables no estáticas o de instancia
 - Los métodos estáticos pueden acceder únicamente a las variables estáticas
 - No pueden acceder a los métodos no estáticos o de instancia
 - Los métodos estáticos pueden acceder únicamente a otros métodos estáticos
 - Se pueden redefinir en subclases
 - Pueden ser públicos, protegidos, predeterminados o privados

Modificador y métodos estáticos

- La invocación de un método de instancia es diferente con relación al método de clase (estático)
- Por ejemplo
 - Primero debe crear una instancia y luego usar una notación de puntos para invocar a un método de instancia; mientras que, el nombre de la clase, la notación de puntos y el nombre del método estático invocan a un método estático

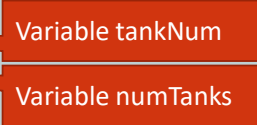
Modificador y métodos estáticos

- El método estático incluye un contenedor para construir una instancia de una clase
- Cuando la clase tiene un constructor de acceso privado, el método estático es uno de los dos enfoques utilizados para crear una instancia de esa clase

Ejemplo de la clase Tortuga

- La clase Tortuga tiene una variable estática que identifica a la cantidad de tanques que tenemos disponibles (numTanks) y una variable de instancia (tankNum) que nos indica en qué tanque se encuentra la tortuga

```
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);  
    } //end constructor  
    public void swim() { //implementation }  
    public int getAge() { return age; }  
    public int getTankOfResidence() { return tankNum; }  
    public static String fishTank() { return "I have " + numTanks  
                                     + " fish tanks.";  
    } //end method fishTank  
} //end class Nesting
```



Métodos estáticos en el ejemplo de la clase Tortuga

- Revisar los métodos en la clase Tortuga



```
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);  
    } //end constructor  
    public void swim(){//implementation}  
    public int getAge(){return age;}  
    public int getTankOfResidence(){return tankNum;}  
    public static String fishTank() {return "I have " + numTanks + " fish tanks.";}  
    } //end method fishTank  
} //end class Turtle
```

swim() es un método de instancia
Aunque cada tortuga puede nadar, pueden hacerlo de diferente manera según su edad

fishTank() es un método estático
y accede a una variable (numTanks)

getAge() y getTankOfResidence() son métodos de instancia no estáticos porque acceden a variables no estáticas
Los métodos estáticos no pueden acceder a los elementos no estáticos

Creación de instancias de clases utilizando métodos estáticos

- Los métodos estáticos también se utilizan para crear instancias de clases cuando el acceso del constructor de la clase es privado y el método es parte de la misma clase
- Esto es posible porque se accede públicamente al método estático con acceso privado a la clase

```
...  
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 y clases estáticos

- Las clases estáticas o anidadas:
 - Pueden existir como clases anidadas
 - No pueden existir como clases independientes



Una clase anidada es una clase creada dentro de otra clase



ORACLE
Academy

JF 7-3
Modificador estático y clases anidadas

Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

21

Hay algunas otras formas de anidar clases en Java que no están cubiertas en este curso.

Clases anidadas estáticas

- Las clases estáticas
 - se implementan dentro de otras clases y las otras clases se denominan clases contenedoras
 - Pueden extender el comportamiento de la clase contenedora
 - Se pueden sobrecargar como los constructores comunes

Clases anidadas estáticas

- La clase anidada estática también proporciona los medios para crear una instancia en una clase contenedora cuando su constructor está configurado con acceso privado
- Esta es la segunda manera en que se puede crear una instancia de una clase que posee un calificador de acceso restringido o privado para sus constructores de clase

Ejemplo de clases anidadas estáticas

```
public class Space {  
    //Space class variables  
  
    public static class Planet{  
        //planet class variables and constructors  
        public Planet() {  
            ...implementation...  
        } //end constructor  
  
        public Planet(String, int size){  
            ...implementation...  
        } //end constructor  
    } //end class Planet  
  
    //more space class implementation  
} //end class Space
```


Terminología

- Los términos clave usados en esta lección son los siguientes:
 - Método de clases
 - Variable de clases
 - Clase interior
 - Nested class
 - Modificador estático
 - Método estático
 - Clase anidada estática
 - Variable estática

Resumen

- En esta lección, habrá aprendido a:
 - Crear variables estáticas
 - Usar variables estáticas
 - Crear métodos estáticos
 - Usar métodos estáticos
 - Crear clases estáticas
 - Usar clases estáticas



ORACLE
Academy

JF 7-3
Modificador estático y clases anidadas

Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

26

The Oracle Academy logo 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 a thin black border, with dark gray horizontal bars at the top and bottom.

ORACLE

Academy