Java 7 SE Fundamentals

Sección 8

Métodos y sobrecarga de métodos

- Uso de modificadores
- Paso de argumentos y retorno de valores
- Creación de variables y métodos estáticos
- Sobrecarga de un método
- Crear e invocar un método

Que son los modificadores de java

- Se usan en clases, constructores, métodos, datos y bloques de nivel de clase), pero el modificador **final** también se puede usar en variables locales en un método.
- Un modificador que se puede aplicar a una clase se denomina modificador de clase.
- Un modificador que se puede aplicar a un método se denomina modificador de método.
- Un modificador que se puede aplicar a un campo de datos se denomina modificador de datos.

Modificadores existentes

- **public:** Una clase, constructor, método o campo de datos es visible para todos los programas de cualquier paquete.
- **private:** Un constructor, método o campo de datos solo es visible en esta clase.
- **potected**: Un constructor, método o campo de datos es visible en este paquete y en las subclases de esta clase en cualquier paquete.
- **static**: Define un método, un campo de datos o un bloque de inicialización estático.
- **final:** Una clase no puede heredar. Un método no se puede modificar en una subclase. Un campo de datos es una constante.

Métodos

Conceptos clave

En que consiste un método

La definición de un método consiste en el nombre de este, sus parámetros, su tipo de valor de retorno y el cuerpo.

Definición en Java

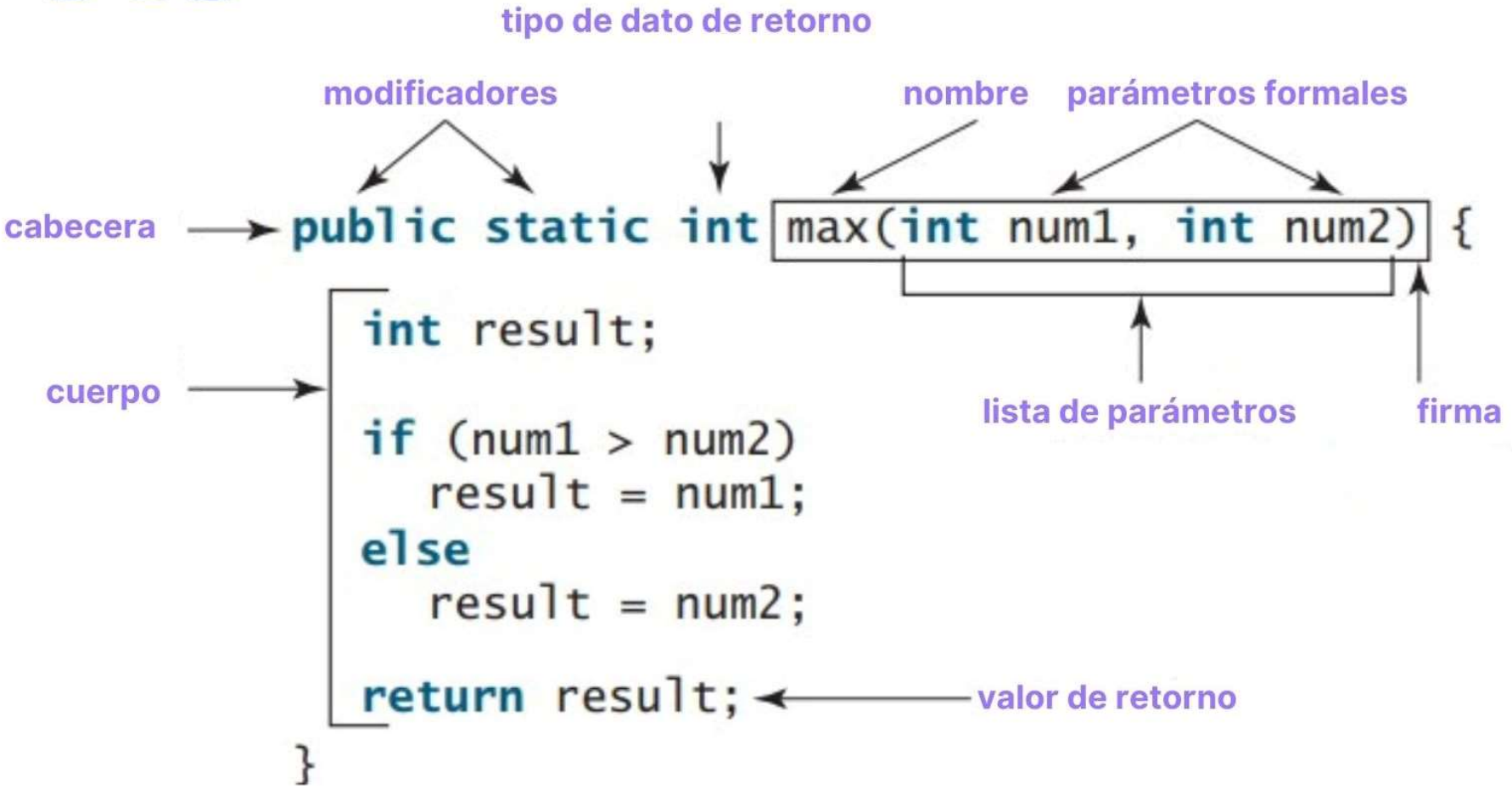
La definición de un método consta de un encabezado y de un cuerpo.

Que define el encabezado

Especifica los modificadores, el tipo de valor de retorno, el nombre del método y sus parámetros.

Si un método no devuelve ningún valor se denomina método **void**.

Definición de un Método



Invocación de un Método

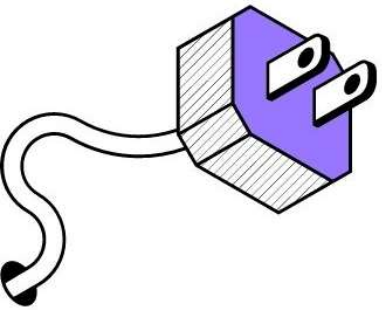
```
int z = max(x, y);
```

parámetros actuales (argumentos)

variables y métodos estáticos

- Una variable estática es compartida por todos los objetos de la clase. Un método estático no puede acceder a los miembros de instancia de la clase (sus atributos).
- Las variables estáticas almacenan valores para las variables en una ubicación de memoria común..
- Para utilizar los elementos de un paquete es necesario importar este en el módulo de código en curso, usando para ello la sentencia

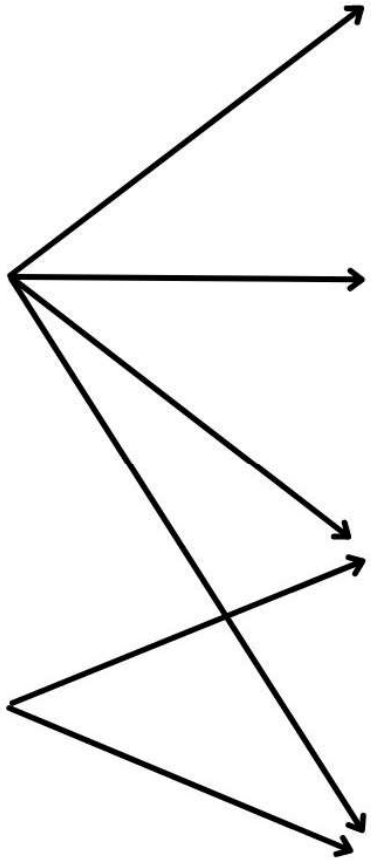
Como defino una variable o método estático



- Para el caso de una variable, basta con colocar la palabra reservada **static** en su declaración.
- Para el caso de un método, basta con colocar la palabra reservada **static** en su cabecera.

Un método de instancia

Un método estático

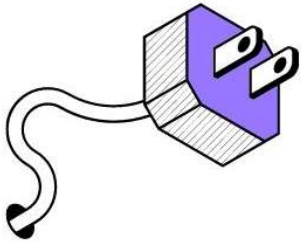


Método de instancia

Campo de datos de instancia

Método estático

Un campo de datos estático



Sobrecarga de Métodos

Conceptos clave

En que consiste la sobrecarga de un método

La sobrecarga de métodos le permite definir los métodos con el mismo nombre siempre que sus firmas sean diferentes.

```
/*Retorna el maximo de dos valores enteros*/
public static int max(int num1, int num2) {
  if (num1 > num2)
    return num1;
  else
    return num2;
}

 /*Retorna el maximo de dos valores double*/
public static double max(double num1, double num2) {
  if (num1 > num2)
    return num1;
  else
    return num2;
}

/*Retorna el maximo de tres valores double*/
public static double max(double num1, double num2, double num3) {
  return max(max(num1, num2), num3);
}
```

Java 7 SE Fundamentals

Sección 9

Uso de la encapsulación y constructores

Implementando la encapsulación

Crear un constructor

La encapsulación

Como funciona

- Hacer que los campos de datos sean privados protege los datos y hace que la clase sea fácil de mantener.
- En primer lugar, los datos pueden ser **manipulados.**
- En segundo lugar, la clase se vuelve difícil de mantener y vulnerable a errores.
- Para evitar modificaciones directas de los campos de datos, debe declarar los campos de datos como privados, utilizando el modificador privado. Esto se conoce como encapsulación de campos de datos.

Los constructores

Como funciona

- Se invoca un constructor para crear un objeto usando el operador new.
- Los constructores son un tipo especial de método.
- Un constructor debe tener el mismo nombre que la propia clase.
- Los constructores no tienen un tipo de retorno, ni siquiera **void**.
- Los constructores se invocan mediante el operador **new** cuando se crea un objeto. Los constructores juegan el papel de inicializar a los objetos.

```java
public class CircleWithPrivateDataFields {

  private double radius = 1;


  private static int numberOfObjects = 0;


  public CircleWithPrivateDataFields() {
    numberOfObjects++;
  }


  public CircleWithPrivateDataFields(double newRadius) {
    radius = newRadius;
    numberOfObjects++;
  }


  public double getRadius() {
    return radius;
  }


  public void setRadius(double newRadius) {
    radius = (newRadius >= 0) ? newRadius : 0;
  }


  public static int getNumberOfObjects() {
    return numberOfObjects;
  }


  public double getArea() {
    return radius * radius * Math.PI;
  }
}
```