Métodos Vacíos

Programación de Computadoras II

Abdel G. Martínez L.

Agenda

1. Métodos Matemáticos
2. Agregando Nuevos Métodos
3. Flujo de Ejecución
4. Parámetros y Argumentos
5. Múltiples Parámetros
6. Diagrama de Pila
7. Leyendo Documentación
8. Escribiendo Documentación

Métodos Matemáticos

- La librería de Java incluye la clase Math que provee operaciones matemáticas comunes. Forma parte del paquete java.lang, por lo que no debe ser importado, sino invocado.

```
double root = Math.sqrt(17.0);
double angle = 1.5;
double height = Math.sin(angle);
```

- Las funciones trigonométricas utilizan radianes. Para convertir grados a radianes se pueden utilizar los métodos de la clase Math:

```
double radian = Math.toRadians(180.0)
double degrees = Math.toDegees(Math.PI)
```

Métodos Matemáticos

- Otro uso importante es el redondeo, donde se redondea un valor punto flotante al entero más cercano y retorna un long.

```
long x = Math.round(Math.PI  * 20.0);
```

- Recuerden siempre colocar la invocación a la clase Math. Por ejemplo, si se coloca únicamente pow(2.0, 10.0) saldrá un mensaje de error como el siguiente:

```
File: Test.java [line: 5]
Error: cannot find symbol
  symbol:    method pow(double,double)
  localtion: class Test
```

Agregando Nuevos Métodos

- Es posible definir más de un método dentro de una clase:

```
public class NuevaLinea {
   public static void nuevaLinea() {
        System.out.println();
   }
   public static void main(String args[]) {
        System.out.println(”Primera linea.”);
        nuevaLinea();
        System.out.println(”Segunda linea.”);
   }
}
```

Agregando Nuevos Métodos

- El nombre de la clase anterior es NuevaLinea. Por convención los nombres de las clases deben iniciar con mayúsculas.
- Esta clase contiene dos métodos, nuevaLinea y main.
- Java es sensitivo a las mayúsculas y minúsculas. Por lo tanto, NuevaLinea y nuevaLinea son diferentes.
- Los métodos deben iniciar con letra minúscula y seguir el formato ”camel case”, donde el nombre es parecido al patrón *esteNombre*.
- nuevaLinea y main son públicos, que significa que pueden ser invocados por otras clases.

Agregando Nuevos Métodos

- Son void, que significa que no retornan un resultado.
- Los paréntesis luego del nombre del método contienen una lista de variables, conocidas como parámetros.
- La salida del programa es la siguiente:

```
Primera linea.

Segunda linea.
```

- Los métodos pueden ser invocados más de una vez y se pueden llamar métodos dentro de otros métodos.

Flujo de Ejecución

- Usualmente los programas se tienden a leer de arriba hacia abajo, pero en el paradigma de programación de Java se tiene que seguir el flujo de ejecución del programa. Veamos este ejemplo:

```
public class NuevaLinea {
    public static void nuevaLinea() {
           System.out.println();
    }
    public static void dosLineas() {
           nuevaLinea();
           nuevaLinea();
    }
    public static void main(String[] args) {
           System.out.println(”Primera linea.”)
           dosLineas();
           System.out.println(”Segunda linea.”)
    }
}
```

Flujo de Ejecución

- El flujo de ejecucións siempre inicia con la primera sentencia del main, independientemente de dónde esté en el código fuente.
- Las sentencias se ejecutan una a la vez, en orden, hasta que se llegue a la invocación de un método.
- En lugar de continuar con la sentencia siguiente, se va a la primera línea del método invocado, se ejecutan todas sus sentencias y luego se regresa al punto exacto siguiente a la invocación.
- Recuerden que hay métodos que llaman a otros métodos, por lo que hay que seguirles la pista. Java es bueno llevando este seguimiento por lo que no requiere trabajo manual.

Parámetros y Argumentos

- Algunos métodos van a requerir argumentos, los cuales son los valores que se necesitan para invocar los métodos.
- Por ejemplo, el método pow requiere dos argumentos double mientras que el método println requiere un String.
- Cuando se utilzan los métodos, se proveen los argumentos.
- Cuando se escriben los métodos, se nombran los parámetros.
- La lista de parámetros indican cuáles argumentos se necesitan.

Parámetros y Argumentos

- En este ejemplo, el método imprimirDoble tiene un parámetro String llamado c.

```
public class ImprimirDoble {
   public static void imprimirDoble(String c) {
        System.out.println(c);
        System.out.println(c);
   }
   public static void main(String[] args) {
        imprimirDoble(”Frase a duplicar”);
   }
}
```

Parámetros y Argumentos

- Cuando se invoca el método, debemos proveer un argumento que cumpla con el tipo de dato String.
- El **pase de parámetros** es el proceso donde el argumento se asigna al parámetro del método.
- El pase puede ser un valor directo o bien a través de una variable.
- El valor que se provee como argumento debe ser del mismo tipo de datos que el parámetro.
- Algunas veces Java hace una conversión automática. Esto funciona de *int* a *double*, pero no de *int* (o *double*) a *String*.

Múltiples Parámetros

- El siguiente ejemplo es un método que acepta dos parámetros:

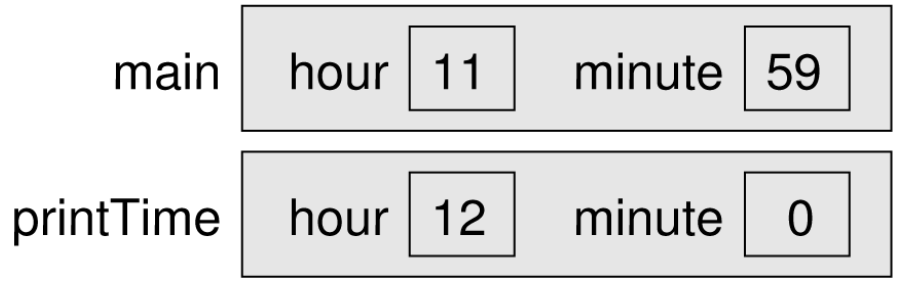
```
public static void imprimirHora(int hora, int minuto) {
   System.out.println(hora);
   System.out.println(":");
   System.out.println(minuto);
}
```

- Dos maneras de invocar este método son:

```
int hora = 11;
int minuto = 59;
imprimirHora(hora, minuto);
imprimirHora(11, 59);
```

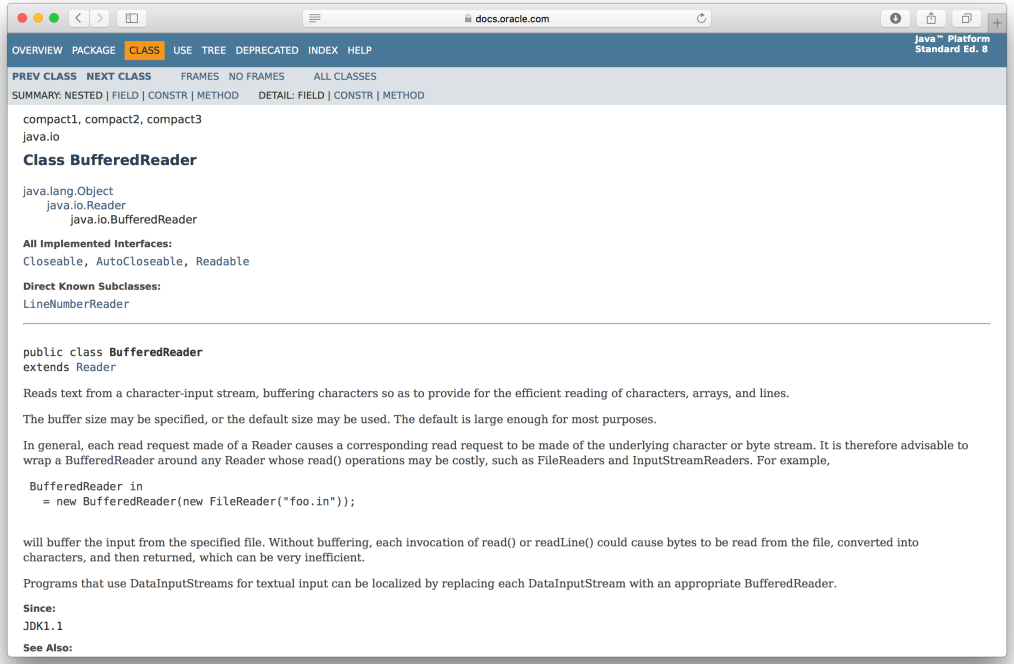
Diagramas de Pila

```
public class ImprimirHora {
    public static void imprimirHora(int hora, int minuto) {
        System.out.print(hora);
        System.out.print(":");
        System.out.println(minuto);
    }
    public static void main(String[] args) {
        int hora = 11;
        int minuto = 59;
        imprimirHora(hora+1, 0);
    }
}
```



Leyendo Documentación

- Java tiene un extensa librería de clases y métodos.
- Recomiendo siempre leer la documentación.
- Toma un poco de tiempo acostumbrarse y aprender qué partes leer y cuáles ignorar, pero la realidad es que te ayuda a reinventar la rueda.
- **https://docs.oracle.com/javase/8/docs/api/**



Escribiendo Documentación

- Java tiene la habilidad de incrustar documentación en el código.
- De esta manera se puede escribir todos los cambios hechos en el código, de una forma sencilla y consistente.
- Si uno documenta el código fuente, se puede extraer automáticamente y generar un documento HTML, usando una librería llamada **Javadoc.**
- Esta librería escanea los códigos fuente buscando el formato especial de documentación el cual inicia con /** y termina con */

Escribiendo Documentación

```
/**
 * Example program that demonstrates print vs println.
 */
public class Goodbye {
    /**
     * Prints a greeting.
     */
    public static void main(String[] args) {
        System.out.print("Goodbye, ");  // note the space
        System.out.println("cruel world");
    }
}
```

¡HASTA LA PRÓXIMA CLASE!

Tema: Condicionales y Lógica