Programación 2

Lenguaje Java - Conceptos claves (*2da parte*)

Profesor: Eduardo Godoy

eduardo.gl@gmail.coml

Material elaborado por Rodrigo Olivares

rodrigo.olivares@uv.cl

23 de agosto de 2017

Contenido

1. Clase String
   - Definición
   - Principales Métodos
2. Paso de parámetros
   - Referencia / Valor
3. Autoreferencia
   - Palabra reservada this
4. Atrubutos y métodos miembros
   - De Objeto
   - De Clase
5. Packages
   - Conceptos

Clase String

Definición

- En Java la clase String permite instanciar objetos que representan cadenas de caracteres NO modificables. Entonces los objetos creados a partir de la clase String, son objetos Inmutables.
- Todos los objetos que representen cadenas de caracteres en Java son instancias de la Clase String.
- La clase String se puede representan como un arreglo de caráctereres y ser tratada como tal mediante métodos implementado en ella.

Objeto inmutable: Su valor inicial persiste durante toda la ejecuciíon del programa y su estado no puede ser cambiado.

Clase String

Principales Métodos

Cuadro: Métodos.

length() :	Devuelve la longitud de la cadena
indexOf(caracter):	Devuelve la posición de la primera aparición de carácter
lastIndexOf(caracter):	Devuelve la posición de la última aparición de carácter
charAt(n) :	Devuelve el carácter que está en la posición n
substring(n1,n2) :	Devuelve la subcadena entre las posiciones n1 y n2-1
toUpperCase() :	Devuelve la cadena convertida a mayúsculas
toLowerCase() :	Devuelve la cadena convertida a minúsculas
equals(cad) :	Compara dos cadenas y devuelve true si son iguales
equals(cad)	Compara dos cadenas y devuelve true si son iguales
equalsIgnoreCase(cad)	Igual que equals pero sin considerar mayúsculas y minúsculas
compareTo(OtroString)	Devuelve 0, mayor a 0, menor a 0
compareToIgnoreCase(OtroString)	Compara sin considerar mayúsculas y minúsculas.
valueOf(N)	Convierte el valor a String.

Paso de parámetros

Referencia / Valor

- **Valor**: *el método recibe una copia del valor del argumento; el método trabaja sobre esa copia que podrá modificar su valor, sin que esto tenga incidencia al valor efectivo del argumento.*
- **Referencia**: *el método recibe la referencia del argumento; el método trabaja directamente, por lo cual podrá modificar el valor efectivo del argumento.*

Autoreferencia

Palabra reservada this

- Para hacer una referencia a la instancia actual de una clase usamos la palabra reservada this.
- Utilizaremos la palabra reservada this para referirnos al **objeto actual** o a las v**ariables de instancia de este objeto**.
- **No** utilizaremos la palabra reservada this en los métodos y atributos declarados como static.

Se puede omitir la palabra reservada this para las variables de instancia, esto depende de la existencia o no de variables con el mismo nombre en el ámbito local.

Autoreferencia

Palabra reservada this

```
public class Prueba {

        private int test = 10;

        public void printTest() {
                int test = 20;
                System.out.println("test" + test);
                System.out.println("test" + this.test);
        }

        public static void main(String[ ] args) {
                Prueba p = new Prueba();
                p.printTest();
        }
}
```

Atrubutos y métodos miembros

De Objeto

Atributos

Cada objeto, instancia de una clase, tiene su propia copia de los atributos miembro y clase pueden ser de tipos primitivos (**boolean**, **int**, **long**, **double**, ...) o referencias a objetos de otra clase.

Métodos

Los métodos son subrutinas definidas dentro de una clase. Se aplican siempre a un objeto de la clase por medio del operador punto (.) y pueden incluir argumentos explícitos que van entre paréntesis, a continuación del nombre del método.

Atrubutos y métodos miembros

De Clase

Atributos

Una clase puede tener atributos propios de la clase y no de cada objeto. A estos atributos se les llama **atributos de clase** o atributos *static*. Los atributos *static* se suelen utilizar para definir constantes comunes para todos los objetos de la clase (por ejemplo **PI** en la clase *Circulo*) o atributos que sólo tienen sentido para toda la clase (por ejemplo, un contador de objetos creados como *numCirculos* en la clase *Circulo*).

- Los atributos de clase se crean anteponiendo la palabra **static** a su declaración.
- Para llamarlas se suele utilizar el nombre de la clase (no es imprescindible), pues así, su sentido queda más claro. Por ejemplo, *Circulo.numCirculos* es un atributo de clase que cuenta el número de Círculos creados.

Atrubutos y métodos miembros

De Clase

Métodos

- Existen métodos que **no** actúan sobre objetos concretos a través del operador punto. A estos métodos se les llama **métodos de clase** o **static**.
- Un ejemplo típico de métodos *static* son los métodos mateméticos de la clase **java.lang.Math** (*sin*(), *cos*(), *exp*(), *pow*(), etc.).

- Los métodos crean anteponiendo la palabra **static** a su declaración.
- Para llamarlos se suele utilizar el nombre de la clase, en vez del nombre de un objeto de la clase (por ejemplo, **Math**.*sin*(ang), para calcular el *seno* de un ángulo).

Packages

Conceptos

Un package es una agrupación de clases.

Para que una clase pase a formar parte de un **package** llamado *packageName*, se debe agregar en ella, la sentencia:

- **package** packageName;

Los nombres de los *packages* se suelen escribir con **minúsculas**, para distinguirlos de las clases, que empiezan por **mayúscula**. El nombre de un package puede constar de varios nombres unidos por puntos (los propios packages de Java siguen esta norma, como por ejemplo *java.awt.event*).

Todas las clases que forman parte de un **package** deben estar en el mismo directorio.

Preguntas

Preguntas ?