

JAVA para No Programadores

Módulo 2

String y qué podemos hacer

Anteriormente habíamos conocidos los *datos primitivos* en Java y también conocimos el tipo de dato *String* que nos ayuda representar una cadena de caracteres, ahora bien, los *String* no son un tipo de dato primitivo, sino un *objeto*.

Por ahora, solo diremos que los objetos son un tipo de dato más complejo que posee atributos (Variables y/o Constantes) y métodos que nos serán de gran utilidad a lo largo del desarrollo del software.



Métodos

Tipo	Método	Descripción
char	<code>charAt(int index)</code>	Devuelve el valor de carácter en el índice especificado.
String	<code>concat(String str)</code>	Concatena la cadena especificada al final de esta cadena.
boolean	<code>contains(CharSequence s)</code>	Devuelve verdadero si y solo si esta cadena contiene la secuencia especificada de valores de caracteres.
boolean	<code>endsWith(String suffix)</code>	Prueba si esta cadena termina con el sufijo especificado.
boolean	<code>equals(Object anObject)</code>	Compara esta cadena con el objeto especificado.
boolean	<code>equalsIgnoreCase(String anotherString)</code>	Compara esta cadena con otra cadena, ignorando las consideraciones de caso.
static String	<code>format(String format, Object... args)</code>	Devuelve una cadena formateada utilizando la cadena de formato y los argumentos especificados.
int	<code>indexOf(int ch)</code>	Devuelve el índice dentro de esta cadena de la primera aparición del carácter especificado.

Tipo	Método	Descripción
Boolean	isEmpty()	Devuelve verdadero si, y solo si, length () es 0
static String	join(CharSequence delimiter, CharSequence... elements)	Devuelve una nueva cadena compuesta por copias de los elementos CharSequence unidas con una copia del delimitador especificado.
int	lastIndexOf(int ch)	Devuelve el índice dentro de esta cadena de la última aparición del carácter especificado.
Int	length()	Devuelve la longitud de esta cadena.
String	replace(char oldChar, char newChar)	Devuelve una cadena resultante de reemplazar todas las apariciones de oldChar en esta cadena por newChar.
String	toLowerCase()	Convierte todos los caracteres de esta cadena a minúsculas utilizando las reglas de la configuración regional predeterminada.
String	toUpperCase()	Convierte todos los caracteres de esta cadena a mayúsculas utilizando las reglas de la configuración regional predeterminada.

Tipo	Método	Descripción
static String	valueOf(objeto o dato primitivo b)	Devuelve la representación de cadena del argumento enviado.
static String	trim()	Devuelve una cadena cuyo valor es esta cadena, con cualquier espacio en blanco inicial y final eliminado.
String	substring(int beginIndex)	Devuelve una cadena que es una subcadena de esta cadena.
char[]	substring(int beginIndex, int endIndex)	Devuelve una cadena que es una subcadena de esta cadena.
String	toLowerCase()	Convierte todos los caracteres de esta cadena a minúsculas utilizando las reglas de la configuración regional predeterminada.
String	toUpperCase()	Convierte todos los caracteres de esta cadena a mayúsculas utilizando las reglas de la configuración regional predeterminada.
static String	valueOf(objeto o dato primitivo b)	Devuelve la representación de cadena del argumento enviado.

¿Por qué usar el método equals?

En java los operadores relacionales comparan bit a bit y en los objetos (que profundizaremos más adelante) se compara no el valor, sino la posición de memoria. Por lo que si comparamos dos cadenas de texto introducidas por el teclado con el operador relación `==` esto nos devolverá *false*.

Java nos entrega un método **equals** que solventa este problema.

A == B

¡Sigamos trabajando!