Material descargable

egg

Teoría JAVA I

Clases de Envoltura “Wrappers”

En Java, **cada tipo de dato primitivo cuenta con una clase envolvente o "wrapper" que encapsula dicho tipo de dato primitivo en un objeto.** Estas clases wrapper son fundamentales para permitir el tratamiento de los tipos primitivos como objetos, **facilitando así el uso de métodos y la representación de valores nulos,** funcionalidades que los tipos primitivos no pueden brindar por sí solos. Además, son esenciales cuando se trabaja con colecciones de objetos que no pueden almacenar directamente tipos de datos primitivos (esto se abordará más adelante).

A continuación, presentamos las clases “wrapper” correspondientes a cada tipo de dato primitivo:

- **boolean** → Boolean
- **char** → Character
- **int** → Integer
- **double** → Double
- **byte** → Byte
- **short** → Short
- **long** → Long
- **float** → Float

Ahora, es importante revisar los métodos más relevantes proporcionados por estas clases:

Boolean

Método	Descripción
`toString()`	Devuelve un objeto `String` que representa el valor de este booleano.
`Boolean.valueOf(String s)`	Devuelve un objeto booleano dependiendo del string ingresado por parámetro. Si `s` es igual a “true” (ignorando mayúsculas y minúsculas) devuelve verdadero, sino devuelve falso.

`booleanValue()`	Devuelve el valor booleano del objeto Boolean.

Character

💡Ten presente que los caracteres tienen una representación numérica en la tabla ASCII.

Método	Descripción
`toString()`	Devuelve un objeto `String` que representa el valor de esta variable de tipo `Character`.
`compareTo(Character anotherCharacter)`	Compara numéricamente con `anotherCharacter`, devuelve 0 si son iguales, -1 si “variable” es menor numéricamente, y 1 si “variable” es mayor numéricamente.
`Character.valueOf(String s)`	Devuelve un objeto `booleano` dependiendo del string ingresado por parámetro. Si `s` es igual a “true” (ignorando mayúsculas y minúsculas) devuelve verdadero, sino devuelve falso.
`Character.getNumericValue (char ch)`	Devuelve el valor `int` que representa el carácter Unicode especificado por parámetro.
`Character.getType(char ch)`	Devuelve un valor que indica la categoría general de `ch`. Puedes ver las categorías aquí.
`Character.isLetter(char ch)`	Determina si el carácter especificado es una letra.
`Character.isWhitespace(char ch)`	Determina si el carácter especificado es un espacio en blanco.

Integer

Método	Descripción
`toString()`	Devuelve un objeto `String` que representa el valor de este `Integer`.
`compareTo(Integer anotherInteger)`	Compara numéricamente con `anotherInteger`, devuelve 0 si son iguales, -1 si `variable` es menor numéricamente, y 1 si `variable` es mayor numéricamente.
`Integer.valueOf(String s)`	Devuelve un objeto `Integer` dependiendo del string ingresado por parámetro.
`Integer.MAX_VALUE`	Una constante que contiene el valor máximo que puede tener un `int` ($2^{31}-1$).
`Integer.MIN_VALUE`	Una constante que contiene el valor mínimo que puede tener un `int` -2^{31}.

Double

Método	Descripción
`toString()`	Devuelve un objeto `String` que representa el valor de este `Double`.
`compareTo(Double anotherDouble)`	Compara numéricamente con `anotherDouble`, devuelve 0 si son iguales, -1 si `variable` es menor numéricamente, y 1 si `variable` es mayor numéricamente.
`Double.valueOf(String s)`	Devuelve un objeto `Double` dependiendo del string ingresado por parámetro.

`Double.MAX_VALUE`	Una constante que contiene el mayor valor finito positivo de tipo double, $(2-2^{-52})\times2^{1023}$.
`Double.MIN_VALUE`	Una constante que contiene el menor valor positivo distinto de cero de tipo double, 2^{-1074}.
`doubleValue()`	Devuelve el valor primitivo double correspondiente a este Double. Esto puede ser útil en situaciones donde se necesite trabajar con valores primitivos en lugar de objetos envoltorios.

💡 Siempre es recomendable utilizar la **documentación oficial** para obtener más detalles sobre el material proporcionado, su uso y su implementación.