

Bitácora

25/10/2021

Git

Lizet González Pérez

Java contiene distintos tipos de datos de ellos se visualizan categorías entre primitivos/ referenciados donde se aprecian los tipos enteros, flotantes, clases, interfaces, arreglos.

Conocer el string es parte esencial puesto que las cadenas son de las más vitales en su uso, (siendo variantes)

A su vez al usar los tipos primitivos se es consciente de la capacidad y definición de cada uno. Para byte se debe declarar un rango tanto mínimo (-128) como máximo (127) (MIN VALUE/MAX VALUE).

```
byte numeroByte = 129;  
System.out.println("Valor minimo byte:" + Byte.MIN_VALUE);  
System.out.println("Valor maximo byte:" + Byte.MAX_VALUE);
```

Con ello se le obliga a convertirlo de entero a byte.

Una variable que puede sernos de gran utilidad viene siendo **numeroShort** el cual nos da un almacenamiento de 16 bytes, enfocándole como un atajo de acuerdo a los atributos min y max, utilizar los tipos primitivos nos permiten obtener ciertos valores usar uno entero (int) puede dar fallas en cuanto a la cantidad del número máximo

DE AQUI el tipo **long** (L termina) nos permite hacer ver superior la cantidad literal de ahí ahora se corrige la perdida de presión al pasar de long a entero, el tipo **long** nos ayudara a acceder al resultado de un valor más grande.

tipos primitivos enteros: byte, short, int, long

Byte: (-128 a 127)

Short: rango (-32,768 a 32,767)

Long: (8 bytes)

Para entrar a la parte de los flotantes se opera con una conversión más directa usando una F (flotante) o D (doble) al final para dar un valor máximo sin complicación siendo soportada.

Tipos flotantes (float-32) (doble-64) al definir una variable el valor es primitivo o punto flotante-doble.

Declarando si esta puede ser tipo F para darse a entender al compilar y soportar, su resultado no se altera, aquí el valor min y max se aprecian con valores negativos (E)

Si indicamos F al término de la literal se mantiene sin problema, al dar doble (D) se dará como resultado un valor infinity es decir será invalido para el valor flotante

La clase doble nos hace ver la diferencia entre min y max haciendo ver los valores mayores al compilar siendo igual al valor máximo soportado.

Un detalle al usar **var** en un numero entero y usando literal entero es iniciar un punto de ruptura en una línea que sea ejecutada, dando clic en la línea, y seleccionando **debug file** de ahí nos mostrara un tabulador que se desplegara en la sección variables

Ejecutar la línea con f8 para lazar la línea con el valor entero, al definir var se define la variable ingresada es una manera de saber sobre las variables.

Definir variable **float**, se utiliza F luego de la literal, se imprime y se mantiene el punto de ruptura, al ejecutar se observa la variable tipo doble y a su vez creando un float, resto reconoce como es cada uno

Variable **char** puede almacenar un carácter usando comilla simple y termino punto coma. Se imprime la variable y al ejecutar se ve el valor de dicho carácter, se muestra una lista **unicode** esta contiene una amplia variedad de caracteres más usados y comunes (con códigos caracteres y decimales a disposición)

Al signar **varchar** y el valor (decimal) más el carácter asociado este lo mostrara al ejecutarlo tal cual, a la vez también agregar signos del carácter

```
var varCharDecimal2 = 33;  
System.out.println("varCharDecimal2 = " + varCharDecimal);
```

Usando solo var se modifica el donde se asignara el valor, dando inferencia para encontrarse como carácter **unicode**

```
var varCharSimbolo3 = '!';  
System.out.println("varCharSimbolo3 = " + varCharSimbol
```

Tipos boolean

Almacenan true y false usando la palabra boolean **varboolean** respetando el uso de mayúsculas y minúsculas

Con **if ()** se comprueba si dicho valor es falso o verdadero, dependiendo a cual indicar se usara **else** en casos contrarios, de ahí la breve función de cómo se ira manejando.

Un ejemplo sobre edad donde se preguntara si edad es mayor / igual (>=) regresándonos el valor acertado, al compilar se mostrara que la variable es boolean

```
run:  
varBoolean = true  
La bandera es verdadera  
Eres mayor de edad
```

Si se modifica el valor (de 30 a 10) el resultado se invertirá

```
var edad = 10;  
//var esAdulto = edad >= 18;  
if( edad >= 18 ){  
    System.out.println("Eres mayor de edad");  
}  
else{  
    System.out.println("Eres menor de edad");  
}
```

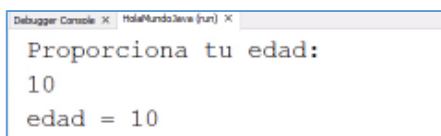
```
run:  
varBoolean = true  
La bandera es verdadera  
Eres menor de edad
```

Conversiones tipos primitivos

Pasar de string a entero y viceversa esto con `var integer.parseInt` así se pasara a entero recibiendo una cadena y cambiándola a string de tipo `int`. (Convertir string a tipo entero para realizar operaciones de tipo entero)

valorPI usando **`parsedouble`** con el valor de PI al ejecutar muestra el valor, al pasar el punto de ruptura en debug file se ejecuta paso a paso para ver la línea donde se detiene el programa verificando que la conversión a `int` se ha realizado, la variable PI ahora es **`double`** que recibe cadena.

Para cambiar variable se usa **`consola.nextLine`** se requiere de **`integer.parseInt`**, dando un texto para hacer conversión de la variable edad, ejecutando y notando que pide la edad donde se agrega, simplemente da cierta falla de acuerdo a la colocación de texto en vez del número solicitado.



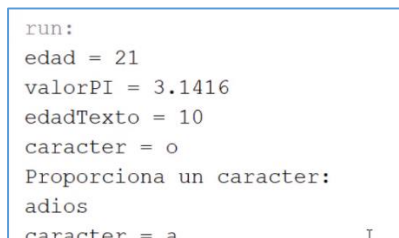
```
Debugger Console x HolaMundoJava (run) x
Proporciona tu edad:
10
edad = 10
```

De **`int`** a **`string`** usando conversión directa

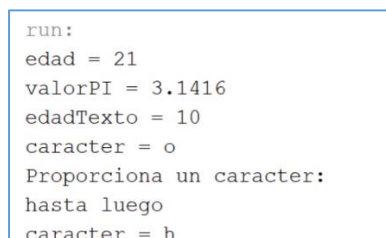
Dando un valor a la edad, en un carácter podemos colocar el método **`charAt`** dando un índice de lo que lleva la cadena, agregando índice por cada letra, al ejecutar se imprime el valor que pedimos, cambiando de variable string a char

Esto sirve al solicitar información de la consola solicitando un elemento de la consola donde se podrá reutilizar la variable **`consola.nextLine`** (uno de tantos métodos) una vez creado carácter siempre será char, la cadena podrá agregarse con `.charAt` indicando el retorno del primer carácter, se ejecuta y comprueba que el carácter se ha indicado.

Recuperando el primero indicado al colocar el valor 0



```
run:
edad = 21
valorPI = 3.1416
edadTexto = 10
caracter = o
Proporciona un caracter:
adios
caracter = a
```



```
run:
edad = 21
valorPI = 3.1416
edadTexto = 10
caracter = o
Proporciona un caracter:
hasta luego
caracter = h
```