Strings en Kotlin

Las cadenas (Strings) se utilizan para almacenar texto.

Una cadena contiene una colección de caracteres entre comillas dobles:

Ejemplo
```
var greeting = "Hello"
```

Sin embargo, al igual que con otros tipos de datos, puedes especificar el tipo si insistes:

Ejemplo
```
var greeting: String = "Hello"
```

Nota: Si desea crear un String sin asignar el valor (y asignar el valor más tarde), debe especificar el tipo al declarar la variable:

Ejemplo
Esto funciona bien:
```
var name: String
name = "John"
println(name)
```

Ejemplo
Esto generará un error:
```
var name
name = "John"
println(name)
```

Acceder a una cadena

Para acceder a los caracteres (elementos) de una cadena, debe consultar el número de índice entre corchetes.

Los índices de cadenas comienzan con 0. En el siguiente ejemplo, accedemos al primer y tercer elemento en txt:

Ejemplo
```
var txt = "Hello World"
println(txt[0]) // first element (H)
println(txt[2]) // third element (l)
```

[0] es el primer elemento. [1] es el segundo elemento, [2] es el tercer elemento, etc.

Longitud de la cadena

Una cadena en Kotlin es un objeto que contiene propiedades y funciones que pueden realizar ciertas operaciones en cadenas, escribiendo un carácter de punto (.) después de la variable de cadena específica. Por ejemplo, la longitud de una cadena se puede encontrar con la length propiedad:

Strings en Kotlin

Ejemplo

```
var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
println("The length of the txt string is: " + txt.length)
```

Funciones de cadena

Hay muchas funciones de cadena disponibles, por ejemplo toUpperCase()y toLowerCase():

Ejemplo

```
var txt = "Hello World"
println(txt.toUpperCase())   // Outputs "HELLO WORLD"
println(txt.toLowerCase())   // Outputs "hello world"
```

Comparando cadenas

La función compara dos cadenas y devuelve 0 si ambas son iguales:compareTo(string)

Ejemplo

```
var txt1 = "Hello World"
var txt2 = "Hello World"
println(txt1.compareTo(txt2))  // Outputs 0 (they are equal)
```

Encontrar una cadena en una cadena

La indexOf()función devuelve el índice (la posición) de la primera aparición de un texto específico en una cadena (incluidos los espacios en blanco):

Ejemplo

```
var txt = "Please locate where 'locate' occurs!"
println(txt.indexOf("locate"))  // Outputs 7
```

Recuerda que Kotlin cuenta las posiciones desde cero.
0 es la primera posición en una cadena, 1 es la segunda, 2 es la tercera...

Comillas dentro de una cadena

Para utilizar comillas dentro de una cadena, utilice comillas simples ('):

Ejemplo

```
var txt1 = "It's alright"
var txt2 = "That's great"
```

Strings en Kotlin

Concatenación de cadenas

El + operador se puede utilizar entre cadenas para sumarlas y crear una nueva cadena. Esto se llama concatenación:

Ejemplo
```
var firstName = "John"
var lastName = "Doe"
println(firstName + " " + lastName)
```

Tenga en cuenta que hemos agregado un texto vacío (" ") para crear un espacio entre el nombre y el apellido al imprimir.

También puedes usar la función plus() para concatenar dos cadenas:

Ejemplo
```
var firstName = "John "
var lastName = "Doe"
println(firstName.plus(lastName))
```

Plantillas de cadenas/interpolación

En lugar de concatenación, también puedes usar "plantillas de cadena", que es una manera fácil de agregar variables y expresiones dentro de una cadena.

Simplemente consulte la variable con el $símbolo:

Ejemplo
```
var firstName = "John"
var lastName = "Doe"
println("My name is $firstName $lastName")
```

Las "Plantillas de cadenas" son una característica popular de Kotlin, ya que reduce la cantidad de código. Por ejemplo, no es necesario especificar un espacio en blanco entre nombre y apellido, como hicimos en el ejemplo de concatenación.