
# Kotlin Strings, String Templates & Comments

Sources: http://kotlinlang.org/docs/reference/basic-syntax.html
http://petersommerhoff.com/dev/kotlin/kotlin-for-java-devs/

# Strings

- Strings are represented by the type String.
- Strings are immutable.
- Elements of a string are characters that can be accessed by the indexing operation: s[i].
- A string can be iterated over with a for-loop:

```
for (c in str) {
    println(c)
}
```

# String Literals

- Kotlin has two types of string literals:
  escaped strings that may have escaped characters in them
  and
  raw strings that can contain newlines and arbitrary text.

# String Literals

- Kotlin has two types of string literals:
  escaped strings that may have escaped characters in them
  and
  raw strings that can contain newlines and arbitrary text.

An escaped string is very much like a Java string

```
val s = "Hello, world!\n"
```

A raw string is delimited by a triple quote (“”””), contains no escaping and can contain new lines and any other characters.

```
val text = """
    |Tell me and I forget.
    |Teach me and I remember.
    |Involve me and I learn.
    |(Benjamin Franklin)
    """.trimMargin()
```

# String Literals

```
Console
<terminated> Config - Main.kt [Java Application] C:\Program Files\Java\jre1.
Tell me and I forget.
Teach me and I remember.
Involve me and I learn.
(Benjamin Franlkin)
```

You can remove leading whitespace with trimMargin()

```
val text = """
    |Tell me and I forget.
    |Teach me and I remember.
    |Involve me and I learn.
    |(Benjamin Franklin)
    """.trimMargin()
```

# String Literals

By default | is used as margin prefix, but you can choose another character and pass it as a parameter like trimMargin(">").

```
val text = """
        >  Tell me and I forget.
        >  Teach me and I remember.
        >  Involve me and I learn.
        >  (Benjamin Franlkin)
         """.trimMargin(">")

print(text)
```

*Note the impact of the two spaces we put between > and the text*

Console

```
<terminated> Config - Main.kt [Java Application] C:\Program Files\Ja
  Tell me and I forget.
  Teach me and I remember.
  Involve me and I learn.
  (Benjamin Franlkin)
```

# String Templates

- Strings may contain template expressions, i.e. pieces of code that are evaluated and whose results are concatenated into the string.
- A template expression starts with a dollar sign ($) and consists of either a simple name:

```
val i = 10
val s = "i = $i" // evaluates to "i = 10"
```

- or an arbitrary expression in curly braces:

```
val s = "abc"
val str = "$s.length is ${s.length}" // evaluates to "abc.length is 3"
```

# String Templates

Templates are supported both inside **raw strings** and inside **escaped strings**.

```
val anInt = 10
val aString = "Value of anInt is ${anInt}\n"

val text = """
        >  Tell me and I forget.
        >  Teach me and I remember.
        >  Involve me and I learn.
        >  (Benjamin Franlkin)
      """.trimMargin(">")

print(aString)
print(text)
```

Console

```
<terminated> Config - Main.kt [Java Applicati
Value of anInt is 10
  Tell me and I forget.
  Teach me and I remember.
  Involve me and I learn.
  (Benjamin Franlkin)
```

# String Templates

```
fun main(args: Array<String>) {
    var a = 1
    // simple name in template:
    val s1 = "a is $a"

    a = 2
    // arbitrary expression in template:
    val s2 = "${s1.replace("is", "was")}, but now is $a"
    println(s2)
}
```

s1

"a is 1"

```
a was 1, but now is 2
```

# Comments – single line and block comments

Just like Java and JavaScript, Kotlin supports end-of-line and block comments.

```
// This is an end-of-line comment

/* This is a block comment
   on multiple lines. */
```

Unlike Java, block comments in Kotlin can be nested.

# Comments – KDoc (equivalent to JavaDoc)

```
/**
 * A group of *members*.
 *
 * This class has no useful logic; it's just a documentation example.
 *
 * @param T the type of a member in this group.
 * @property name the name of this group.
 * @constructor Creates an empty group.
 */
class Group<T>(val name: String) {
    /**
     * Adds a [member] to this group.
     * @return the new size of the group.
     */
    fun add(member: T): Int { ... }
}
```

# Comments – KDoc

| Block tags | Currently supported KDoc block tags |
|---|---|
| @param <name> | Documents a value parameter of a function or a type parameter of a class, property or function. |
| @return | Documents the return value of a function. |
| @constructor | Documents the primary constructor of a class. |
| @receiver | Documents the receiver of an extension function. |
| @property <name> | Documents the property of a class which has the specified name. |
| @throws <class>, @exception <class> | Documents an exception which can be thrown by a method. |
| @sample <identifier> | Embeds the body of the function with the specified qualified name into the documentation for the current element, in order to show an example of how the element could be used. |
| @see <identifier> | Adds a link to the specified class or method to the See Also block of the documentation. |
| @author | Specifies the author of the element being documented. |
| @since | Specifies the version of the software in which the element being documented was introduced. |
| @suppress | Excludes the element from the generated documentation. Can be used for elements which are not part of the official API of a module but still have to be visible externally. |

For more info: http://kotlinlang.org/docs/reference/kotlin-doc.html