



UNIVERSIDAD
COMPLUTENSE
DE MADRID



ntic
master
School

Scala

Primeros pasos

Enero de 2023



Agenda

- **REPL Scala**
- **Sintaxis de Scala**



1 REPL Scala

EI REPL



```
(cflores) → ~ sbt console
[info] welcome to sbt 1.4.9 (Private Build Java 1.8.0_312)
[info] loading global plugins from /home/charles/.sbt/1.0/plugins
[info] loading project definition from /home/charles/project
[info] set current project to charles (in build file:/home/charles/)
[info] Starting scala interpreter...
Welcome to Scala 2.12.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_312).
Type in expressions for evaluation. Or try :help.

scala> "hola,"+"mundo"
res0: String = hola,mundo
```

- **Read Eval Print Loop** es una shell interactiva parecida al intérprete de python. Para usarlo, se debe tener instalado **sbt** y **scala**.
 - *scala* desde la línea de comando
 - *sbt console* desde sbt
- Es muy útil para desarrollar haciendo evaluaciones rápidas de código.
- Compila y evalúa código de Scala inmediatamente



Comandos útiles



```
scala> :help
```

All commands can be abbreviated, e.g., :he instead of :help.

:help [command]	print this summary or command-specific help
:history [num]	show the history (optional num is commands to show)
:javap <path class>	disassemble a file or class name
:load <path>	interpret lines in a file
:paste [-raw] [path]	enter paste mode or paste a file
:quit	exit the interpreter
:replay [options]	reset the repl and replay all previous commands
:require <path>	add a jar to the classpath
:reset [options]	reset the repl to its initial state, forgetting all session entries
:save <path>	save replayable session to a file
:sh <command line>	run a shell command (result is implicitly => List[String])
:warnings	show the suppressed warnings from the most recent line which had any



Los otros REPL



- [Scastie](#): es un REPL online y que nos permite incluso tener control de configuración de los 'builds' y permite compartir el código.
- [IntelliJ IDEA](#) este IDE nos provee del Worksheet de [Scala](#) que nos permite hacer lo mismo que el REPL, pero desde una interfaz gráfica y más intuitiva que la línea de comandos. **Este será el que utilizemos durante el curso.**
- Scala IDE para Eclipse también cuenta con Worksheet de Scala gracias a un plugin y permite hacer lo mismo que en IntelliJ IDEA.



2 Sintaxis de Scala

Valores inmutables

```
scala> val message = "hello World"
message: String = hello World

scala> message = "reassignando"
<console>:12: error: reassignment to val
      message = "reassignando"
              ^
```

- Es base fundamental de Scala.
- Se declaran usando la palabra reservada: ***val***.
- Sólo permite la asignación de un valor una vez.

Inferencia de tipos

```
scala> val message = "Hello, world"
message: String = Hello, world

scala> val message: String = "Hello world"
message: String = Hello world

scala> val message: Int = "hello world"
<console>:11: error: type mismatch;
 found   : String("hello world")
 required: Int
    val message: Int = "hello world"
                        ^
```

- El compilador infiere el tipo del valor que se está asignando.
- Scala es un lenguaje con tipado estático:
 - Una vez se declara una variable de un tipo, no se puede reasignar un valor de tipo distinto.

Valores mutables



```
scala> var message = "Hello, world"
message: String = Hello, world

scala> message = "reassigned"
message: String = reassigned
```

- Se declaran usando la palabra reservada **`var`**.
- Se puede asignar un valor distinto, pero siempre del mismo tipo.



Programación orientada a expresiones



- Una sentencia (*statement*) se ejecuta y no devuelve ningún valor:
 - una asignación de valor a una variable
 - imprimir por pantalla
- Una expresión (*expression*) obtiene un valor tras evaluarse:
`val resultado = if (1 == 1) "correcto" else "raro"`
- En Scala, a diferencia de otros lenguajes, se basa en expresiones, muchas construcciones del lenguaje son expresiones, por ejemplo: *bloques de código, if o try-catch*.



Bloques de código

```
scala> val value = {  
  | val x = 1  
  | val y = 2  
  | x + y  
  | }  
value: Int = 3
```

- Los bloques de código son expresiones.
- El valor que devuelve un bloque es la última expresión que se evalúa en el bloque.
- En el ejemplo, el resultado del bloque es el valor de la expresión $x + y$
- Buena práctica: Apertura de las llaves siempre en la misma línea de la asignación.

Lenguaje ligero



```
scala> val mola =  
|   if ("Scala" startsWith "S") {  
|   |   val scala = "Scala"  
|   |   val mola = "mola"  
|   |   val mazo = "mazo"  
|   |   scala + " " + mola + " " + mazo  
|   |   } else  
|   |   "No puede ser, porque Scala != Python"  
mola: String = Scala mola mazo
```

- No se necesitan llaves para expresiones de una única línea.
- Se pueden omitir la definición de los tipos (*Inferencia de tipos*)
- No son necesarios puntos ni paréntesis para llamar a funciones.
- No es necesario `;` ni tabulaciones.
- No hace falta `return`.

