5 Things You Need To Know About Scala Compilation

Mirco Dotta
Triplequote co-founder

Scala Italy Florence,
2018-09-14

Who am I?

- Scala hacker since 2005
- Worked mostly on Scala tooling (MiMa, Eclipse Scala IDE, Scala compiler)
- Co-Founder of Triplequote
    - Make Scala compile as fast as Java thanks to **Hydra**

#1 Build time ≠ Compilation time

Build time ≠ Compilation time

- **Before** starting compilation:
  - Dependency resolution
  - Source generators
  - Formatter
  - Settings evaluation

```
[info] Updating mainSettingsProj...
[info] Updating scriptedSbtProj...
[info] Updating actionsProj...
[info] Formatting 6 Scala sources in testingProj:compile ...
[info] Done updating.
[warn] /Users/dragos/workspace/oss/sbt/internal/util-collection/src/m
l/AList.scala:
[warn] Search state exploded around line 177
[info] Done updating.
[info] Compiling 27 Scala sources to /Users/dragos/workspace/oss/sbt/
classes ...
[info] Done updating.
[info] Updating mainProj...
[info] Compiling 18 Scala sources to /Users/dragos/workspace/oss/sbt/
arget/scala-2.12/classes ...
[info] Done updating.
[info] Updating scriptedPluginProj...
[info] Updating sbtProj...
[info] Done updating.
[info] Done updating.
[info] Done compiling.
[info] Done compiling.
[info] Formatting 1 Scala source in logicProj:compile ...
[info] Formatting 6 Scala sources in runProj:compile ...
[info] Formatting 8 Scala sources in coreMacrosProj:compile ...
[info] Formatting 10 Scala sources in protocolProj:compile ...
[info] Formatting 7 Scala sources in taskProj:compile ...
[info] Formatting 13 Scala sources in completeProj:compile ...
[info] Compiling 1 Scala source to /Users/dragos/workspace/oss/sbt/in
```

Build time ≠ Compilation time

- **Before** starting compilation:
  - What do we need to compile?
  - Hash classpath, hash sources (a LOT of I/O)
  - Load analysis data

```
[info] Updating mainSettingsProj...
[info] Updating scriptedSbtProj...
[info] Updating actionsProj...
[info] Formatting 6 Scala sources in testingProj:compile ...
[info] Done updating.
[warn] /Users/dragos/workspace/oss/sbt/internal/util-collection/src/m
l/AList.scala:
[warn] Search state exploded around line 177
[info] Done updating.
[info] Compiling 27 Scala sources to /Users/dragos/workspace/oss/sbt/
classes ...
[info] Done updating.
[info] Updating mainProj...
[info] Compiling 18 Scala sources to /Users/dragos/workspace/oss/sbt/
arget/scala-2.12/classes ...
[info] Done updating.
[info] Updating scriptedPluginProj...
[info] Updating sbtProj...
[info] Done updating.
[info] Done updating.
[info] Done compiling.
[info] Done compiling.
[info] Formatting 1 Scala source in logicProj:compile ...
[info] Formatting 6 Scala sources in runProj:compile ...
[info] Formatting 8 Scala sources in coreMacrosProj:compile ...
[info] Formatting 10 Scala sources in protocolProj:compile ...
[info] Formatting 7 Scala sources in taskProj:compile ...
[info] Formatting 13 Scala sources in completeProj:compile ...
[info] Compiling 1 Scala source to /Users/dragos/workspace/oss/sbt/in
```

Build time ≠ Compilation time

- **Finally** Sbt starts compiling: Zinc and incremental compiler: **15% overhead**

```
[info] Updating mainSettingsProj...
[info] Updating scriptedSbtProj...
[info] Updating actionsProj...
[info] Formatting 6 Scala sources in testingProj:compile ...
[info] Done updating.
[warn] /Users/dragos/workspace/oss/sbt/internal/util-collection/src/m
l/AList.scala:
[warn] Search state exploded around line 177
[info] Done updating.
[info] Compiling 27 Scala sources to /Users/dragos/workspace/oss/sbt/
classes ...
[info] Done updating.
[info] Updating mainProj...
[info] Compiling 18 Scala sources to /Users/dragos/workspace/oss/sbt/
arget/scala-2.12/classes ...
[info] Done updating.
[info] Updating scriptedPluginProj...
[info] Updating sbtProj...
[info] Done updating.
[info] Done updating.
[info] Done compiling.
[info] Done compiling.
[info] Formatting 1 Scala source in logicProj:compile ...
[info] Formatting 6 Scala sources in runProj:compile ...
[info] Formatting 8 Scala sources in coreMacrosProj:compile ...
[info] Formatting 10 Scala sources in protocolProj:compile ...
[info] Formatting 7 Scala sources in taskProj:compile ...
[info] Formatting 13 Scala sources in completeProj:compile ...
[info] Compiling 1 Scala source to /Users/dragos/workspace/oss/sbt/in
```

Use an IDE to build!

#2 Type Classes +

Macro Derivation = 💣

"""| triplequote

Type Classes, Macro 💣

- Imports take precedence over companion objects
- Cost: implicit resolution
- Cost: additional code to be generated

Type Classes, Macro 💣

```scala
case class FullName(fname: String, lname: String)
case class ShippingAddress(
    billTo: FullName,
    registerTo: FullName,
    street: String)

class TestCirce extends AutoDerivation {
  val dec1: Decoder[ShippingAddress] = implicitly
  val dec2: Decoder[ShippingAddress] = implicitly
}
```

Decoders/ implicits	0	1	2
Code size after type-checking	8 KB	31 KB	54 KB

Type Classes, Macro 💣

D[ShippingAddress]

D[FullName] D[FullName] D[String]

D[String] D[String] D[String] D[String]

Avoid re-generating the same typeclass derivation multiple times!

#3 Whitebox macros are type-checked 3x!

"""| triplequote

What's the different between *whitebox* and *blackbox* macros?

Which one takes part in type-inference?

```
if (isBlackbox(expandee)) {
  val expanded1 = atPos(enclosingMacroPosition.makeTransparent)(Typed(expanded0, TypeTree(innerPt)))
  typecheck("blackbox typecheck", expanded1, outerPt)
} else {
  // whitebox expansions need to be typechecked against WildcardType first in order to avoid SI-6992 a
  // then we typecheck against innerPt, not against outerPt in order to prevent SI-8209
  val expanded1 = typecheck("whitebox typecheck #0", expanded0, WildcardType)
  val expanded2 = typecheck("whitebox typecheck #1", expanded1, innerPt)
  typecheck("whitebox typecheck #2", expanded2, outerPt)
}
}
```

Shapeless macros are whitebox

"""| tripleequote

#4 Type-checking should be around 30%

#4 Type-checking should be around 30%

- Type-checking Scala sources takes time
- but should not go above 30% of total compilation time!
    - Use **-verbose** to see times for each phase (and a lot of noise)

#4 Type-checking should be around 30%

- If more, you can bet your money on one (or more) of:
  - Macro expansion (useful flag: **-Xprint:typer**)
  - Implicit resolution (useful flag: **-Xlog-implicits**)
  - Quasiquotes, type tags (macros in disguise)

#5 The Typer/Parser Node Factor

How does the size of the AST change through phases?

#5 The Typer Multiplier

- Generally, type checking *adds* nodes
- **Mo’ nodes, mo’ time, mo’ coffee**
- What is a typical value of the TPNF? **1.2x**
    - but it can vary wildly, between 0.8 and 5x

#6 Scala compiler is single-threaded

#6 Scala compiler is single-threaded*

- It’s a batch compiler
- Careful interplay of laziness and mutability
- Laziness required for recursive types
- Mutability for performance

*** 2.12.6 has an experimental multi-threaded classfile generation phase**

#6 Scala compiler is single-threaded

- Bigger/more powerful machines won't make a dent in compilation times
- Unless you have a parallel Scala compiler!
- Hint: there is one that works today! (google Scala Hydra)

How fast is `scalac`?

Simple code (Spark)	FP-heavy (Cats)	Play Framework (LiChess)		
		Average	Views	Controllers
1’200 LoC/s	**480 LoC/s**	**1’500 LoC/s**	**2’000 LoC/s**	**500 LoC/s**

(2.12.4 numbers, with a warm compiler — except LiChess: 2.11.11)

Summary

- Compilation times are unpredictable
- A single file can account for 80% of compilation time
- Compilation speed varies greatly based on coding style
- Macro expansion can quintuple your code size