Scala Center updates
Q3 2020 Advisory Board meeting

Scala Center team: Julien Richard-Foy, 60%; Jamie Thompson, 100%; Ergys Dona, 100% until August 31st; Vincenzo Bazzucchi, 100% since September 1st; Fengyun Liu, 100% since October 1st; Adrien Piquerez, 100%; Meriam Lachkar, 100%; Sébastien Doeraene, 100%; Darja Jovanovic, 100%; VirtusLab team: Tomasz Godzik, 100%; Wojciech Mazur, 100%.

External contractors, part time: Eric Loots, Noel Welch, Alvin Alexander, Eugene Yokota.

At a glance

MOOCs

@julienrf @vincenzobaz

Reactive Programming on Coursera

We ported the course Reactive Programming to the Coursera platform, it was originally published on EdX.

The purpose of this project is manifold:

- Attract more students
- Study the impact of the platform on course popularity and revenue
- Analyse the impact of different communication approaches on course popularity and revenue. The course will be published without an immediate announcement, so that we can study how Coursera helps users to discover the course. We will then try announcements on different communication media, which could be Twitter or scala-lang so that we can tune our strategy to have the best results for future courses.

The port also gave us the motivation to improve our continuous integration platform which is going to help manage the great contributions that we are receiving from LAMP and from external actors.

For example, LAMP contributed a new error and grade reporting system and @noelwelsh proposed and implemented a new student assignment.

Effective Scala course

The course content is almost complete, we are currently working on the very last assignment. We will start beta testing in November. We plan to release it early in 2021, just after Scala 3.0.0 is released.

Lecture recording is scheduled for December, at EPFL's MOOC factory.

The course is 6 weeks long, it targets professional developers with no prior experience in Scala. It covers the following topics: language syntax, definitions, expressions, domain modeling, collections in scala-library, loops, tooling, object oriented programming, functional programming, testing, implicits, error handling, and concurrency.

Existing Courses

We have updated the existing courses (already available on Coursera) to work with Scala 3.

BSP in sbt (SCP-023)

@adpi2

The support of the Build Server Protocol (BSP) in sbt has been released in sbt 1.4.0

All the major BSP endpoints are supported:

- the server lifetime endpoints
- the workspace build targets request
- the sources and resources requests
- the dependency sources request
- the compile request

- the test and run requests

We added a workspace reload request to the BSP specification to provide the client with the ability to reload the build configuration.

We have submitted a BSP support in sbt 1.4 blog post to the scala-lang website, that announces and describes the support of BSP in sbt. It should be published in the forthcoming days.

The support of BSP in sbt is being adopted by the community. IntelliJ Scala plugin is already able to import an sbt project with BSP. Metals contributors are actively working on a better integration of sbt in Metals using BSP.

Improving the Scala compiler Debugging experience (SCP-022)

@errikos

We completed the implementation of JSR-45 in the scalac compiler, and submitted a pull request to scala/scala. The pull request underwent several back-and-forth, and is unfortunately not merged yet. The discussion is now in the hands of the compiler team and JetBrains.

Scala 3 compiler

Scala 3 Enum Stabilisation

@bishabosha

We improved and finalized the API for enum definitions in Scala 3.

Highlights

- Enum definitions may now override `toString` on singleton enum values.
- Enum definitions now correctly support the Java serialization framework: cached values will be reused when deserializing singleton values.
- The values array for enumerations is now enabled for all enumerations, even those with a type parameter.
- The `values` array and `valueOf` lookup method will now only be enabled if an enum only has singleton cases.
- By default, class cases of enum definitions will copy the variance of type parameters from the parent enum.
- In an enum definition, class cases will now generate companion objects where `apply` and `copy` now return the most specific subtype, rather than the parent enum type.
- Enum definitions are now supported in Scala.js.

- Enum definitions now declare a public fromOrdinal method on the companion object for use by serialization frameworks.

Scala.js Support in Scala 3

@sjrd

We have implemented virtually all of the support for Scala.js in Scala 3. A preview with the portable subset and basic JavaScript interoperability was released in Dotty 0.27.0-RC1. Support for non-native JS classes will be released as part of Scala 3.0.0-M1, the next release of Dotty. The only missing features are JS exports, which we will implement for the next milestone.

We have submitted a blog post for scala-lang.org about this implementation, which should be published in the coming days.

SemanticDB support in Dotty

@bishabosha

We improved support for extension methods and the @main annotation when producing semanticDB in Scala 3.

Scala 3 Migration Guide

@adpi2 @mlachkar

We are providing the community with knowledge about the migration to Scala 3 in the Scala 3 Migration Guide website. At the time of writing, it covers the following topics:

Compatibility Reference

The compatibility reference page describes the compatibility between Scala 2.13 and Scala 3.0 at the language level, at compile-time and at run-time. It is illustrated with various diagrams of dependencies between Scala artifacts.

Scala 3 migration and syntax rewrites

With the help of Eric Loots from Lunatech, we have documented the Scala migration mode and the syntax rewrite capability of the Scala 3 compiler. The migration rewrite rules are listed in the table of incompatibilities, described below.

Table of incompatibilities

The table of incompatibility contains a list of all known source incompatibilities between Scala 2.13 and Scala 3.0. One can refer to it to find out if the incompatibility is a Scala 2.13 deprecation or feature warning, if it has a Scalafix rewrite or a Scala 3 migration rewrite.

Scala 2 macros migration status

This page contains a large, yet incomplete, list of Scala 2.13 macro libraries that requires a re-implementation in Scala 3. Migrating macro libraries is an important step towards the adoption of Scala 3.

Macros migration tutorial

The macro migration tutorial explains how to make a Scala 2 macro library available in Scala 3. It describes the cross-building technique but also the very new macro-mixing technique.

Scala-migrat3

@mlachkar @adpi2

Scala-migrat3 is a new project targeted at easing, if not automating, the migration of a codebase to Scala 3. It is focused on addressing changes in the type inference algorithm between the two compilers. The tool automatically inserts explicit type annotations and/or inferred terms (such as implicit parameters) in the codebase to make it compatible with Scala 3, without changing its meaning. We target projects that aim to cross-compile in Scala 2.13 and 3.0.

The technical solution that we are implementing was described in a Contributors post.

In order to implement this project, we evolved the scalafix API to allow retrieving Patches and applying them selectively (released in v0.9.20).

We developed a new scalafix rule to add explicit type annotations for every type inferred by the compiler, and we are currently working on a second rule to add the implicit parameters and implicit conversions that are inferred.

We also developed an algorithm that finds the minimum set of those annotations that are required to make a project compile in Scala 3.

At present, the tool can automatically fix 7 among the 9 incompatibilities documented in the migration guide.

Dependency Management

@julienrf @alexarchambault

We have implemented two sbt plugins aiming at simplifying dependency management.

The first one, sbt-version-policy, targets library authors. It ensures that library releases follow the recommended versioning scheme.

The second one, sbt-eviction-rules, targets application developers. It suppresses false eviction warnings. Note that sbt 1.4+ supports the same feature out of the box, so this plugin is relevant only for the transition period (until all library authors publish their versioning scheme).

We believe that these two plugins will simplify managing transitive dependencies by providing accurate eviction warnings.

TASTy Reader for Scala 2

@bishabosha

We merged in the Scala 2 compiler the support for reading dependencies compiled with the forwards compatible subset of Scala 3 (enabled by the TASTy Reader feature). Support is ongoing as further incompatibilities are discovered. We have also written a blog post (pending PR to be released in the coming days) and pages in the Scala 3 migration guide to explain how users will be able to take advantage of forward compatibility when migrating projects.

Metals

@tgodzik, @mlachkar

We released Metals v0.9.4 with a number of improvements and new features.

Worksheets for Scala 3

During the last few months we have been working on providing users with a worksheet experience on par with the one currently available for Scala 2. After initial implementation and after some more improvements both in Metals and in a related Mdoc project, we have managed to bring the new feature to the users.

More details on how to use them can be found in the documentation.

sbt files support

Previously, users would not be able to use most of Metals' features inside of sbt build definition files, which hindered their ability to work with them. Now, it is possible to use the most important features. Full information about what is supported can be found in the documentation.

Organize imports

During the last few months, we have been working on making it possible to use Scalafix's organize imports rule available as a default in the Metals language server. This will enable everyone to sort their imports inside Scala files according to their preferences. The feature is not

yet released, but was recently merged and is being tested to make sure we provide a worthwhile experience to the users.

Show implicit terms and inferred types

We added the ability to display implicit parameters and inferred types for functions, which was one of the most anticipated features. This will be available in most editors, however it will be best supported in Visual Studio Code, where the additional information will be shown inside the code itself. The work will be part of the next release.

Scalameta

@tgodzik

We have been working on the Scalameta parser, which is a basis for several tools used in the Scala community (for example scalafmt), to support Scala 3. We have managed to add support for most of the Scala 3 new features, with only 2 still missing, and managed to parse almost the entire Dotty codebase except for 4 files. We are now planning on adding the last remaining parts and releasing the new version to be used in both Metals and scalafmt.

Scala Native

@errikos, @WojciechMazur

We have added support for Scala 2.12, while Scala 2.13 support is currently under pending review. We're finishing resolving blockers and high priority bugs, as well as introducing binary breaking changes before the 0.4.0 release.

Scala 2.12 support

We managed to speed up our work on releasing support for the newer Scala versions starting with Scala 2.12. To support its changes, we add support for default methods in the NIR format.

Improved native interoperability

We are making it easier to pass functions to C libraries by replacing the implementation of `CFuncPtr` SAM traits with implicit conversion methods taking arbitrary Scala `FunctionN`. The changes also fix a fundamental unsoundness in native interoperability.

Miscellaneous

- We included source code positions in the NIR format, allowing us to enhance debugging support in the future.
- We fixed the `CString` interpolator to deal with escapes in triple quoted strings
- We added support for interop with C dynamic functions, passed as `void*`

Scala.js

@sjrd

We published Scala.js 1.2.0 and, later, Scala.js 1.3.0., The major highlight of the latter release is support for module splitting, which was implemented by Tobias Schlatter.

We added support for a number of JDK APIs, including all the higher-order methods of `java.util` collections, and many Unicode-related functions of `java.lang.String` and `java.lang.Character`.

scalajs-bundler

@sjrd

We updated scalajs-bundler to support Scala.js 1.3.0 and future releases. The support for module splitting required breaking an internal API that scalajs-bundler relies on.

This was released in sbt-scalajs-bundler 0.20.0.

Communication

@darjutak

The Scala Center team has been increasingly more proactive in the communication department since May 2020. We intend to continue the frequency and quality going further, this is just to highlight some of our activities:

Blog posts:

- Scala 3 - A community powered release
- BSP Support in sbt 1.4

Active Contributors threads in these categories, about 25, and growing:

- Scala Center Updates
- Scala 3 Release Projects

In the last quarter there was also a major increase in communication between Scala teams across the board, and contributors, the product of this internal communication is visible in the overall acceptance and excitement throughout the community.

We have another 7 blogs lined up until the end of 2020, and are working on some video material in a form of tutorials, code examples and more, that will go along well with the Scala 3 release candidate in December this year.

Management

@darjutak

Internal trainings, teams sprees, and the first team retreat

Team building was the main internal focus of the management in the last year. After saying goodbye to 3 of our dear colleagues in the second half of 2019, we have employed 6 new people since the beginning of 2020, one of which left in the meantime, and 3 out of which in the Q3.

This is why onboarding, project management, and team building were crucial for successfully delivering on our promises. We used the budget usually reserved for conference expenses towards team retreat and various team trainings. We also used the knowledge we have as individuals and started the “team sprees” once a week in the afternoon to share, discuss, teach, and learn from each other. This resulted in a stronger cohesion of the team, several team initiatives, and overall high team spirit in a short period of time and in the context of COVID-19 constraints.

Team movements in Q3

Hires

July 2020

Wojciech Mazur (VirtusLab), 100% Scala Native;

September 2020

Vincenzo Bazzucchi (full Scala Center employee), 80% Education, 20% other;
Katja Goltsova (student engineer contract) 15h/week, Scala 3 related projects;

October

Fengyun Liu (full Scala Center employee), 80% Scala 3 compiler stability and performance; 20% research.

Alumni:

As of September 2020

Ergys Dona, who decided to apply for a Phd program at EPFL.

Contractors:

Alvin Alexander (2-3 days per week), Scala 3 documentation;

Noel Welsh (2-3 days per week), “Effective Scala” course;

Eric Loots (2 days per week), Migration guide, “Effective Scala” course, documentation.

Strengthening connections with EPFL

In the last quarter, we have started strengthening our connection with different departments and centers at EPFL. We believe there is a vast potential in collaborating closer with the Vice Presidency of Innovation who are specialised (amongst other) in helping centers identify, approach, and negotiate with new, for example, Advisory Board leads, as well as participate in various internal conferences with other centers to share our experience, offer and get support. We enrolled in a couple of Working Groups and already opened the doors to 4 potential Advisory Board members - more on this in the next quarter, as these things take time and patience.

Donation

Signed and received the first private donation from a donor who explicitly asked to stay anonymous. With this donation and some additional funds from our side we were able to employ an engineer for a year to work on Scala Native (Wojciech Mazur), the project that had great success and community response since March onwards.