Scala Center updates
Q2 2020 Advisory Board meeting

Scala Center team: Julien Richard-Foy, 60%; Jamie Thompson, 100%; Ergys Dona, 100%; Adrien Piquerez, 100%; Meriam Lachkar, 100%; Sébastien Doeraene, 100%; Darja Jovanovic, 100%; VirtusLab team; Maxime Kjaer, student project; Martin Hanzel, student project

At a glance

MOOCs

@julienrf

Functional Program Design

We have improved the feedback returned by the automated graders of our new assignments in the course Functional Program Design.

Effective Programming in Scala

We are currently working on a new course, Effective Programming in Scala, which targets experienced programmers that are not familiar with Scala but who want to work in Scala.

We have drafted the 6 weeks of content and we have written the detailed content of the first two weeks, including the assignments and their automated graders.

Education: Migration to Scala 3 Workshop

@julienrf, @sjrd

We have reviewed Lunatech's course about migrating from Scala 2 to Scala 3. This hands-on course shows how to take advantage of the new features of Scala 3 in a real project.

BSP in sbt (SCP-023)

@adpi2

We have implemented partial support of the BSP protocol in sbt that has been shipped in sbt milestone 1.4.0-M1.

The workspace, sources, dependency sources and scalac options requests enable project import in both Metals and IntelliJ IDEA through BSP. Compilation requests are also supported, including diagnostic reports as well as taskStart and taskFinish notifications.

You can find more details about the usage of BSP in sbt 1.4.0-M1 in this contributors forum post.

Some of the known limitations are:

- Connection to sbt server by Metals or IntelliJ IDEA times out when the server is not yet loaded. It is recommended to first load sbt then connect to it.
- The sequence of compile diagnostics makes the error reports blink. This will be improved during the next quarter.
- sbt 1.4.0-M1 does not handle integration-test or custom configurations properly. A fix for this limitation has been merged and will be shipped in the next sbt release.

The test and run capabilities will be implemented in the forthcoming quarter. We will also provide a way to deactivate the BSP import of some subprojects or configurations which can be very handy sometimes.

Improving the Scala compiler Debugging experience (SCP-022)

@errikos

We have been working towards adding JSR-45 support into scalac. We published a roadmap for the feature on Contributors in May.

Completed so far:

- The debug info builder infrastructure within the compiler is set up.
- The debug info builder self-tests and the compiler tests are set up.

In progress:

- Emit debug info from the inliner.

We will submit a pull request within the next 2 weeks. In the meantime, you can check the progress in the branch on GitHub.

Scala 3 compiler

Front-end

@julienrf

We have improved the error messages produced by the Scala compiler in case of missing implicit (see #8611 and #8640).

We have minimized and reported issues in the type inference algorithm (#8801, #8802, #8933).

Back-end

@sjrd

We reimplemented the encoding of traits and mixin composition in the Scala 3 compiler so that it uses the same encoding as Scala 2. Dotty had initially been implemented with a new experimental encoding designed by Dmitry Petrashko. In isolation there was no issue with it, but it created a mismatch at the binary level between Scala 2 and Scala 3 libraries, causing issues when extending a Scala 2 trait from a Scala 3 class or vice versa (via the TASTy reader for Scala 2).

We are still investigating one remaining regression in the community build. This should be available by the next Dotty release (0.26.0-RC1).

Metals Support in Dotty

@bishabosha

In order to help enable Metals in the Dotty codebase itself, we now compile Dotty itself with SemanticDB, as well as all the compilation tests.

This detected a number of issues that were not handled in the SemanticDB-targeted tests, which we fixed. It also means that any changes to the language will be automatically tested against SemanticDB generation (validation will still require explicit checkfile based tests).

Dependency Management

@alexarchambault

We have started working on two sbt plugins, sbt-eviction-rule and sbt-compatibility, helping Scala developers to manage conflicting dependencies, and helping library authors to follow the recommended versioning scheme. An announcement with more details and documentation will follow soon.

Scalafix Release v1.0

@mlachkar

Scalafix, which is a refactoring and linting tool for Scala, could be a key factor in the success of Scala 3 migration. For that, we had to evolve the tool and to integrate some necessary features described in this github issue.

We started being active again in the gitter channel and in reviewing pending pull requests. We collaborated with @bjaglin, a new active maintainer, to add support for most features we planned for v1 release, and to release versions 0.9.16 and 0.9.17.

Highlights

- The project is now cross compiled with Scala 2.13, which is the first necessary step to allow usage of rules that are specific to 2.13. This feature unlocks the usage of the ExplicitResultTypes rule, which we intend to use later for the migration to Scala 3.
- Contributed by @bjaglin: there is new support for local rules that are colocated with the source code. This feature simplifies the development process for rules that will be used only inside this project.

TASTy Reader for Scala 2

@bishabosha

We opened the epic TASTy Reader pull request to scala/scala, with support for the full common subset of features that can be exposed in signatures, between Scala 3.0 and Scala 2.13. Significant effort was put into useful error reporting where incompatibilities exist between Dotty and Scala 2.13. This PR supports Dotty 0.23.0-RC1 compiled dependencies. In this there are some differences to how Dotty features are interpreted in Scala 2:

- Singleton enum values are typed as case objects before erasure

- The given modifier is identical to implicit
- Opaque type aliases are now transparent
- References to some Scala 3 features are errors
    - Union types and match types
    - Erased parameters
    - Bounds with different kinds
    - Curried type application
    - Metaprogramming primitives

In a separate branch to the PR, we updated the TASTy reader to be compatible with the latest release of Dotty: 0.25.0-RC2. In this branch, we added support of reading both def macros and macro bundles from TASTy, which were added in the new release of Dotty. Once the original PR is merged, we will submit this new branch in a new pull request.

This work is currently under review by Lukas Rytz from Lightbend.

Metals

@tgodzik, with items from @alexarchambault, @mlachkar

Recently, we have been concentrating on the Scala 3 efforts as well as general Metals and Bloop stability. We've also managed to introduce Ammonite support thanks to @alexarchambault, as well as some new features. The main releases were 0.8.4 and 0.9.0, with a new one coming very soon.

Highlights

Scala 3 support

We've recently managed to release a version of Metals that supports most of the features for Scala 3, including completions, rename, go to definition, workspace symbols, run/debug, along with some other minor ones. There is still some significant work to be done to support Scala 3 at the same level as Scala 2 currently, but this proves we are on the right track to support it once Scala 3 is released. Additionally, some work has been done on the standalone scalameta parser, which is heavily used in Scala tooling (https://github.com/scalameta/scalameta/pull/2027)

Ammonite support

Ammonite scripts (https://ammonite.io/) are a way to run Scala without having any kind of build tool. Until recently the support for ammonite files inside Metals was very limited, but thanks to @alexarchambault we now have a fully fledged Ammonite BSP server and related Metals integration. This enables diagnostics, references, completions and much more in the *.sc files. This feature is going to be added in the upcoming 0.9.1 release of Metals.

New project provider

Another new feature that will soon be released is the new project provider. This feature enables users to easily pick up any existing Scala template via a Metals-provided UI and start working in a new workspace with everything properly setup. Users can choose from a number of curated templates, ones coming from https://github.com/foundweekends/giter8/wiki/giter8-templates, or just use any existing by entering it manually. This feature is mainly aimed at beginners and is easily discoverable even in an empty Metals Visual Studio Code workspace thanks to the added Welcome view. This can also be used in any other editor via additional command, but beginners are always encouraged to start up with Visual Studio Code.

Miscellaneous

- navigating to parent method
- support for launch.json in Visual Studio Code
- new "Create new symbol" code action thanks to @gabro
- Scala 2.13.2 support
- improvements to the Bloop integration
- automatically add '+' on newline when inside a string thanks to @mlachkar
- new Code action: "Import all missing symbols"

Scala Native

@errikos, @sjrd

We have now taken over the development of Scala Native in earnest. We have been reviewing and merging pending pull requests, although there still are a lot to go through.

We finished the implementation of reflective instantiation, and used it in a rework of the testing infrastructure. This considerably simplifies the infrastructure and brings the sbt interface closer to that of the JVM and Scala.js. In particular, there is no need for source generation in a special NativeTest configuration anymore.

We also significantly simplified the build, making it easier for contributors to understand, and bringing shorter feedback loops during development.

Finally, we have been porting JUnit from Scala.js to Scala Native. The port is complete except support for Assumptions. We will submit a pull request soon. In the meantime, you can check the progress in the branch on GitHub.

Scala.js

@sjrd

We published Scala.js 0.6.33 with some last bug fixes and compile warnings. Given the quick adoption of Scala.js 1.x in the ecosystem, and after a request for comments, we decided to discontinue the support for Scala.js 0.6.x. Scala.js 0.6.33 will remain the last release in the 0.6.x

series. We will continue to publish the compiler plugin for new minor versions of Scala 2.12.x and 2.13.x as long as they work out of the box, but no longer. (We just released it for Scala 2.13.3.)

We published Scala.js 1.1.0, whose highlight is `@js.native vals` and `defs`, a very appreciated feature.

We also published sbt-jsdependencies 1.0.1 and scalajs-env-jsdom-nodejs 1.1.0 with critical bug fixes, notably for a bad interaction with React.js' development mode.

Finally, we updated the compiler plugin to support the new shapes of ASTs produced by the -Xasync transformation of Scala 2.12.12/2.13.3. This work will be released in Scala.js 1.1.1.

scalajs-bundler

@sjrd

We updated scalajs-bundler to support Scala.js 1.1.0, and future releases. The previous support was locked on using the linker of Scala.js 1.0.0. Now, we dynamically load the appropriate linker version.

This was released in sbt-scalajs-bundler 0.18.0.

Scaladex

@adpi2

Target versioning

We reworked target versioning of sbt plugins, Scala.js and Scala Native libraries for easier discovery and correct handling of binary versions:

- Milestone and release candidate versions are discarded except for Scala Native 0.4.0-M2 which is the latest Scala Native version.
- Valid Scala.js versions are 0.6 and 1.x. Scala.js 0.5 has been obsolete for 5 years and so it is discarded.
- The Scala Native valid versions are 0.3 and 0.4.0-M2. Scala Native 0.1 and 0.2 are discarded.
- The valid sbt versions are 0.13 and 1.0. Older versions are discarded.

We have also implemented support for the Dotty target versions. For the purpose of clarity, all Dotty versions are grouped under the dotty family in the home page and the search result page. However it is possible to list all the supported Dotty versions of a library in its project page.

The list of target versions is now much more concise and better reflects the reality of the Scala ecosystem.

sbt plugin imports

The existing capability to pull all the packages that are published in the sbt/sbt-plugin-releases repository had neither been used nor maintained for quite some time. We have resurrected it and improved it. In particular, it is now possible to import *linked* packages (packages published to another repository and linked to the sbt/sbt-plugin-releases). We now import more than 200 previously missing sbt plugins to Scaladex.

Performance improvements

We have initiated a rework of the persistence model for the dependencies between libraries to improve the performance on two aspects:

- the user facing performance when loading a project page
- the operational efficiency of reindexing Scaladex

The current situation is that we are forced to reindex Scaladex once in a while to compute the reverse dependencies of each release and the dependent count of each project. We are currently trying to break this constraint.

Webpack plugin for Scala.js sources (student project)

@MaximeKjaer

We published several versions of scalajs-webpack-loader up to v0.0.6. The project is now usable in simple Webpack projects. More details on its capabilities are found in the readme linked above.

This was a student project, which was successfully defended on June 18.

Compiling Java to the Scala.js IR (student project)

@arthanzel

The project scalajs-jfe (Scala.js Java Front End) now contains an almost complete implementation of a Java to Scala.js compiler. However, it is not yet usable in user projects, because we lack integration with the build tools. We believe that adding this support should be relatively easy.

This was a student project, which was successfully defended on June 18.

Scala Days 2020, Berlin edition

@darjutak

The Scala Days 2020 Belin edition was canceled due to the COVID crisis. Given the difficulties caused by the venue management, our organizing team was struggling to navigate the communication. Finally, mid-May we had all the information and we notified the community that ScalaDays Berlin 2020 is canceled (link).

We are looking into online options for later this year. Stay tuned.

Communication

@darjutak and the rest of the team

Conferences

ScalaLove

- Darja Jovanovic and Julien Richard-Foy: “The state of the Scala Center”
- Sébastien Doeraene: “Scala.js 1.0.0: what’s new, what’s better, and what’s next”
- Martin Odersky: “A Scala 3 Update”

MicroSphere

- Tomasz Godzik: “Tooling for Scala 3 - Dotty support in Metals” (slides)

Contributors threads

We created a new "Scala Center Updates" category on Contributors where we opened a dozen topics that correspond to ongoing project roadmaps. Since then, we have been using each thread as a discussion forum as well as follow up and update medium for the authors.

@scala_lang Twitter account

In May, we settled on rules for when and how to use the @scala_lang Twitter account as the Scala Center team in comparison to private accounts. We started intensifying the usage of to promote the Scala Center’s work. This increased activity brought a lot of positive feedback, both in the comments it generated in turn on the Contributors discourse and in statistics terms.

Blog posts by the Scala Center

- Import Suggestions in Scala 3
- The Scala Center stands with Black Lives Matter

Blog posts about the Scala Center

- Lunatech and our commitment to the Scala Center (by Lunatech)
- Announcing the Moving from Scala 2 to Scala 3 course (by Lunatech)
- A Bloop Tour for Metals users (by Chris Kipp, Lunatech)

Management report

@darjutak

- Project management processes put in place.
- Darja took the “Team Care” training provided by EPFL to help navigate the uncertain period during the Covid crisis which helped the team stay connected and performant, given the circumstances.
- We received a personal donation in favor of the Scala Native project. A new engineer from VirtusLab is joining the team to work on Scala Native based on that donation.
- Ergys will move on in September to start a PhD. We will have an opening for his spot soon. He will be onboarding the new engineer to the Scala Native project.
- We opened a new position in the Education department. We have some applications but we are also still searching.
- Collaborating with external contributors (companies and individuals) on community projects.
- Short-term contracts with: Alexandre Archambault and Eugene Yokota.