OOPSLA 23 Artifact

This document provides evaluation guidelines for **Initializing Global Objects: Time and Order**.

The artifact includes an implementation of the global object initialization checking algorithm described in the paper, integrated into Dotty, the Scala 3 compiler. This document helps verify the following:

- The code snippets in the paper are either rejected or accepted by the checker as expected.
- The Scala open issues mentioned in Appendix C are fixed.
- The case study in Section 6 can be reproduced.

Getting Started

Play with Examples

We can run the global initialization checker on a test file as follows:

```
cd ./dotty
bin/scalac -Ysafe-init-global -d ./tmp tests/init-global/neg/mutable-read7
    .scala
```

Note: *The first run of the command will be slow, as it will download dependencies and build the compiler from the source code*.

The compiler is expected to produce the following warning:

```
-- Warning: tests/init-global/neg/mutable-read7.scala:7:17
   ---------------------
7 |  if (Positioned.debug) { // error
  |      ^^^^^^^^^^^^^^^
  |Reading mutable state of object Positioned during initialization of
     object Trees.
  |Reading mutable state of other static objects is forbidden as it breaks
      initialization-time irrelevance. Calling trace:
  |-> object Trees:     [ mutable-read7.scala:11 ]
  |   ^
  |-> val emptyTree = new Tree  [ mutable-read7.scala:13 ]
  |                   ^^^^^^^^
  |-> class Tree extends Positioned     [ mutable-read7.scala:12 ]
  |   ^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  |-> abstract class Positioned:        [ mutable-read7.scala:6 ]
  |   ^
  |-> if (Positioned.debug) { // error  [ mutable-read7.scala:7 ]
  |       ^^^^^^^^^^^^^^^
1 warning found
```

More test cases can be found in the directory `./dotty/tests/init-global`:

- Tests in `tests/init-global/pos` are expected to pass the check with no warnings.
- Tests in `tests/init-global/neg` are expected to be rejected by the checker with warnings.

The reviewer is invited to play with other examples.

Verify Code Snippets in the Paper

All code snippets from the paper are located in `./snippets`. The file name of a snippet has the form `2.3-1-neg.scala`, which can be read as follows:

- `2.3` means that the snippet comes from **Section 2.3** of the paper.
- `1` tells that it is the **1st** snippet in the corresponding section.
- `neg` indicates that the snippet is expected to be **rejected** by the checker with warnings.

The suffix `pos` indicates that the snippet is expected to **pass** the check with no warnings.

We can check a code snippet as follows:

```
cd ./dotty
bin/scalac -Ysafe-init-global -d ./tmp ./snippets/2.3-1-neg.scala
```

We can check all snippets with the following command:

```
./test-all.sh
```

The script will check each file and finally print the results to console as a table. The meanings of each column are as follows:

- **Snippet Name:** The name of the snippet.
- **Expected:** Will be `warning` if the test should produce initialization warnings, and `no warning` if the test should pass the check.
- **Actual:** Will be `warning` if the checker produced warnings on this test, and `no warning` if the it did not.
- **Status:** Will be `pass` if the Expected column matches the Actual column, and `fail` otherwise.

The Status column is expected to be `pass` for all rows.

Verify Checker Fixes Open Scala Issues (Appendix C)

In Appendix C of the paper, we mentioned the following Scala issues:

No.	File Name	Link
#9312	t9312.scala	https://github.com/scala/bug/issues/9312
#9115	t9115.scala	https://github.com/scala/bug/issues/9115
#9261	t9261.scala	https://github.com/scala/bug/issues/9261
#5366	t5366.scala	https://github.com/scala/bug/issues/5366
#9360	t9360.scala	https://github.com/scala/bug/issues/9360
#16152	i16152.scala	https://github.com/lampepfl/dotty/issues/16152
#9176	i9176.scala	https://github.com/lampepfl/dotty/issues/9176
#11262	i11262.scala	https://github.com/lampepfl/dotty/issues/11262

The test files can be found in the directory `./issues`. We expect all the test cases to be rejected by the checker. It can be verified by the following command:

```
for f in ./issues/*; do
    echo "$f"
    ./dotty/bin/scalac -Ysafe-init-global -d ./tmp "$f"
done
```

Reproduce the Case Study (Section 6)

To check Dotty, the Scala3 compiler, we need to first patch the compiler:

```
cd ./dotty && patch -p1 < ../dotty.patch
```

Now run the following commands:

```
sbt scala3-compiler-bootstrapped/clean
sbt scala3-compiler-bootstrapped/compile
```

The last command above is expected to produce 52 warnings. The warnings include:

- The 4 problems of initialization-time irrelevance described in Section 6.1, and
- One violation of partial ordering discussed in Section 6.2.

Note: *The initialization-time irrelevance problem between two specific objects can expose itself as several warnings if there are violations in multiple places*.

Due to code change in the Dotty repo and the improvement in implementation, the checker manages to find more violations of the two principles:

- 2 more violations of initialization-time irrelevance: `NoSymbol` / `NoDenotation` and `NameKinds` / `AvoidNameKind`.
- 2 more violations of partial ordering: `untpd` -> `Trees` -> `untpd` and `Types` -> `Names` -> `Types`.

Implementation

The implementation is integrated in Dotty, and is located mainly in the following source files:

```
./dotty/library/src/scala/annotation/init.scala
./dotty/compiler/src/dotty/tools/dotc/transform/init/Objects.scala
./dotty/compiler/src/dotty/tools/dotc/transform/init/Cache.scala
```