


Godot: All the Benefits of Implicit and Explicit Futures (Artifact)

Kiko Fernandez-Reyes 

Uppsala University, Sweden
kiko.fernandez@it.uu.se

Dave Clarke 

Storytel, Stockholm, Sweden

Ludovic Henrio 

Univ Lyon, EnsL, UCBL, CNRS, Inria, LIP, France
ludovic.henrio@ens-lyon.fr

Einar Broch Johnsen 

University of Oslo, Norway
einarj@ifi.uio.no

Tobias Wrigstad 

Uppsala University, Sweden
tobias.wrigstad@it.uu.se

Abstract

This artifact contains an implementation of data-flow futures in terms of control-flow futures, in the Scala language. In the implementation, we show microbenchmarks that solve the three identified problems from the paper:

1. The Type Proliferation Problem,
2. The Fulfilment Observation Problem, and
3. The Future Proliferation Problem

There are also detailed instructions on design decisions that differ from the formal semantics and restrictions on the limits of how much can be encoded in the Scala language. We provide examples, e.g., creation of a proxy service using data-flow futures, as well as tests that exercise different parts of the type system.

2012 ACM Subject Classification Software and its engineering → Concurrency control; Software and its engineering → Concurrent programming languages; Software and its engineering → Concurrent programming structures

Keywords and phrases Futures, Concurrency, Type Systems, Formal Semantics

Digital Object Identifier 10.4230/DARTS.5.2.1

Acknowledgements We thank the reviewers of the artifact for their helpful comments.

Related Article Kiko Fernandez-Reyes, Dave Clarke, Ludovic Henrio, Einar Broch Johnsen, and Tobias Wrigstad, “Godot: All the Benefits of Implicit and Explicit Futures”, in 33rd European Conference on Object-Oriented Programming (ECOOP 2019), LIPIcs, Vol. 134, pp. 2:1–2:28, 2019.

<https://dx.doi.org/10.4230/LIPIcs.ECOOP.2019.2>

Related Conference 33rd European Conference on Object-Oriented Programming (ECOOP 2019), July 15–19, 2019, London, United Kingdom

1 Scope

This artifact shows an implementation of the formal semantics of data-flow futures, in terms of a library in the Scala programming language. It also show how far one can get encoding data-flow futures in terms of control-flow futures, and pinpoints the places that require modification of the compiler (or an advanced macro system).



© Kiko Fernandez-Reyes, Dave Clarke, Ludovic Henrio, Einar Broch Johnsen, and Tobias Wrigstad; licensed under Creative Commons Attribution 3.0 Germany (CC BY 3.0 DE)

Dagstuhl Artifacts Series, Vol. 5, Issue 2, Artifact No. 1, pp. 1:1–1:2



DAGSTUHL ARTIFACTS SERIES Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1:2 Godot: All the Benefits of Implicit and Explicit Futures (Artifact)

2 Content

The artifact package includes:

- Documentation in files `README.html`, and `README.pdf`.
- The Scala implementation of data-flow futures with tests and examples, in the Scala language (in folder `godot`).
- An `assets` folder used by the `README.html` file.
- A virtual machine `GodotArtefact.ova` that, under `/home/vagrant/Desktop/Godot-Artifact/`, contains the same files as mentioned above.¹

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS).

4 Tested platforms

The artifact disk image works on any platform running Oracle VirtualBox version 6.0.4 with 5 GiB of free disk space and 2 GiB of free RAM. The artifact also works on any machine that has the Scala compiler installed.

5 License

This artifact is provided under the MIT license.

6 MD5 sum of the artifact

d90b8cda99ad792ac9e97f65184087e9

7 Size of the artifact

2.94 GiB

¹ This has been done for ease of reading, so that a reader does not need to change between host and target machine when reading instructions.