Isabelle/PIDE as IDE for ML

Makarius Wenzel
August 2016

http://sketis.net

Isabelle is usually positioned as environment for interactive and automated theorem proving, but its Prover IDE (PIDE) may be used for regular program development as well. Standard ML is particularly important here, since it is the bootstrap language of Isabelle/ML (i.e. SML with many add-ons) and Isabelle/Pure (i.e. the logical framework). The ML IDE functionality of Isabelle + Poly/ML is manifold:

Continuous feedback from static analysis and semantic evaluation is already available for years, e.g. Isabelle2014 (August 2014). It is a corollary of how PIDE interaction works, and of the integration of the Poly/ML compiler into that framework.

Source files are statically checked and semantically evaluated while the user is editing. The annotated sources contain markup about inferred types, references to defining positions of items etc.

Source-level debugging within the IDE is new in Poly/ML 5.6, which is bundled with Isabelle2016 (February 2016).

The Prover IDE provides the Debugger dockable to connect to running ML threads, inspect the stack frame with local ML bindings, and evaluate ML expressions in a particular run-time context. See also chapter 5 of the Isabelle/jEdit manual, as distributed with Isabelle2016.

IDE support for the Isabelle/Pure bootstrap process is new technology for the the next release (presumably published towards the end of 2016). A recent repository version is Isabelle/ec095a532a2b; it is temporarily available from http://www4.in.tum.de/~wenzelm/test/Isabelle_15-Aug-2016.

The `ROOT.ML` file acts like a quasi-theory in the context of theory `ML_Bootstrap`: this allows continuous checking of all loaded ML files. The theory file is presented with a modified header to import Pure from the running Isabelle instance. Results from changed versions of each stage are *not* propagated to the next stage, and isolated from the actual Isabelle/Pure that runs the IDE itself. The sequential dependencies of the above files are only observed for batch build.

It is also possible to modify standalone SML projects, to edit the sources freely in the ML IDE. For example, MetiTarski https://bitbucket.org/lcpaulson/metitarski can participate after some trivial changes of its `ROOT.ML` file.