Mercury

CIS-3030, Fall 2013
Vermont Technical College
Peter C. Chapin
Overview

- Mercury is a multi-paradigm language.
  - Logic
  - Functional
- Has a strong static type system.
- High degree of “purity”
  - Side effects are managed explicitly by passing around state variables.
- Well evolved module system.
- Highly efficient implementation.
Motivation

• Observation.
  – Logic languages not used to advantage.
  – Prolog not well suited to large projects.
    • Hard to write reliable code.
    • Inefficient implementations are common.
    • Logic programming not always the best approach.

• Mercury addresses these issues
  • Real type system.
  • Exceptionally efficient implementation.
  • Combines logic and functional programming approaches.
History

- Project of the University of Melbourne
  - [http://www.mercurylang.org](http://www.mercurylang.org)
  - Only one implementation.
  - Only supported on Unix (or Cygwin)
- Started in 1993
- Usable for real programs three/four years later.
- Still actively developed.
  - Released an Erlang back end that allows Mercury programs to run on the Erlang VM.
Community/Support

- Active mailing list.
  - About 5 to 15 postings per week.
  - All kinds of questions asked and answered.
  - Monitored by the developers.

- Immature tools.
  - No Mercury plug-in for Eclipse!
  - No Mercury editing mode for jEdit or Emacs!

- Minimal third party libraries.
  - Consequence of its status as a research project.
  - “Standard” library reasonably good.
Hello, Mercury!

:- module hello.
:- interface.
:- import_module io.
:- pred main(io::di, io::uo) is det.

:- implementation.

main(IOState_in, IOState_out) :-
    io.write_string("Hello, Mercury!\n", IOState_in, IOtmp),
    io.write_string("Goodbye, Mercury!\n", IOtmp, IOState_out).
Fibonacci Numbers Module

:- module fib.

:- interface.
:- import_module io.
:- pred main(io::di, io::uo) is det.

:- implementation.
:- import_module int, list, string.
:- func fib(int) = int.

% Details on next slide!
Fibonacci Numbers Details

\[
\text{fib}(N) = \\
( \text{ if } \ N \leq 2 \\
\quad \text{ then } 1 \\
\quad \text{ else } \text{fib}(N - 1) + \text{fib}(N - 2) ) .
\]

main(!IO) :-
    io.read_line_as_string(Result, !IO),
    ( Result = eof, 
      io.format("Goodbye!\n", [], !IO) 
    ;
    Result = ok(String),
    ( Result = error(ErrorCode), 
      io.format("%s\n", [s(io.error_message(ErrorCode))], !IO) 
    ).
).
Summary

• Logic Language for Software Engineering
  – More efficient than Prolog.
  – Easier to use (functional features too).
  – Strong types and real modules.

• Fairly New and Immature.
  – Minimal tool and library support.

• Enthusiastic Community

• A Language Worth Watching.
Questions?