Introduction 2: Basic Concepts in Julia

Why We Created Julia

14 February 2012, Jeff Bezanson, Stefan Karpinski, Viral B. Shah, Alan Edelman

We are greedy: we want more.

We want a language that's open source, with a liberal license. We want the speed of C with the dynamism of Ruby. We want a language that's homoiconic, with true macros like Lisp, but with obvious, familiar mathematical notation like Matlab. We want something as usable for general programming as Python, as easy for statistics as R, as natural for string processing as Perl, as powerful for linear algebra as Matlab, as good at gluing programs together as the shell. Something that is dirt simple to learn, yet keeps the most serious hackers happy. We want it interactive and we want it compiled.

(Did we mention it should be as fast as C?)

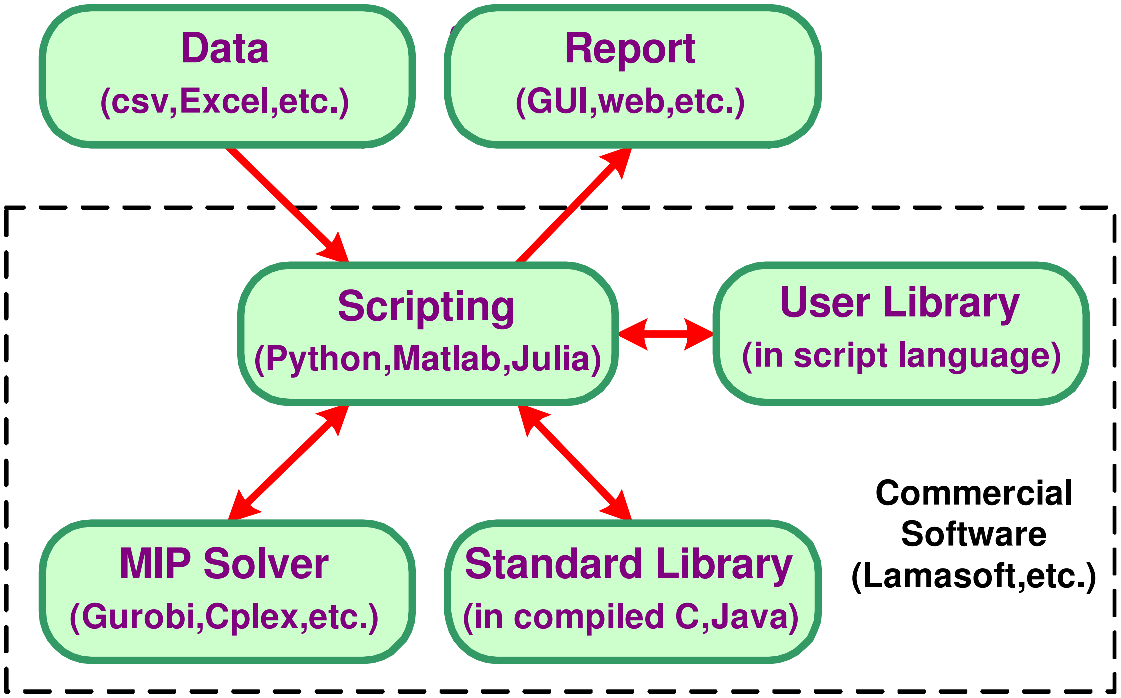
Source: julialang.org/blog/2012/02/why-we-created-julia/

Computational Tools

Data

Processing	Structured	Unstructured
Simple	Calculator	Spreadsheet
Complex	Scripting Language	Hybrid (toolbox, addon)

Logistics Software Stack



- Julia (1.8) scripting language:
  - almost as fast as C and Java (but not FORTRAN)
  - does not require a standard library complied in C/C++/Java for speed (unlike Python, Matlab, and R)
  - uses multiple dispatch to make type-specific (fast) versions of same function
  - JuMP package with algebraic language (macros) to interface with MIP solvers