Introducing the Fortran-lang community

Ivan Pribec   Vincent Magnin   Asdrubal Lozada-Blanco   Ondřej Čertik
Jeremie Vandenplas   Milan Curcic   Rohit Goswami   Sebastian Ehlert
Ioannis Nikiteas   Gabriele Bellomia   Carl Burkert   Emanuele Pagone
Brad Richardson   Amir Shahmoradi

deRSE23 - 21 Feb 2023

Overview

Why Fortran?

- 60+ years of high-level, high-performance computing

Fortran-lang community

- State of the Fortran ecosystem
- A new online community for Fortran users

The future is bright

- Fortran-lang open source software
- stdlib and fpm

Fortran

- Developed in 1954-1957 at IBM; team led by John Backus.
- The name stands for *FOR*mula *TRAN*slation.
- The goal was to ease translation of mathematical formulae to machine code for scientists and engineers.
- First high-level, cross-platform programming language; highly influential.
- Many scientific apps and libraries developed in Fortran.
- Base language still developed, current standard ISO/IEC 1539-1:2018.

Fortran in science and popular culture

Why Fortran? (1)

Fortran Package Manager - fpm

LFortran Compiler

- Multiple backends (LLVM, x86, C, Julia, ...)
- Recent progress
    - MINPACK compilation (`scipy.optimize.least_squares(..., method=lm)`): https://fortran-lang.discourse.group/t/lets-get-lfortran-to-compile-minpack/4550
    - Generics prototype: https://fortran-lang.discourse.group/t/feedback-for-generics-prototype/5231

Fortran support for Visual Studio Code

Community Engagement

- FortranCon
- Google Summer of Code
  - 2021: 5 students
  - 2022: 5 students
  - 2023: Application pending (announced tommorow!)
- Sovereign Tech Fund (https://sovereigntechfund.de/fortran.html)

stdlib has roughly doubled in size in the past year

Modules one year ago vs today

ascii	bitsets	error
io	kinds	linalg
logger	math	optval
quadrature	sorting	specialfunctions
stats	stats_distribution_PRNG	stringlist_type
strings	string_type	system

18 modules, 7 derived types, 119 procedures

Demo: stdlib_logger

ex_logger.f90

```
use stdlib_logger, only: global_logger
implicit none
call global_logger%add_log_file('log.txt')
call global_logger%log_debug('I am invisible')
call global_logger%log_information('Something informative')
call global_logger%log_error('Oopsie daisy')
end
```

log.txt

```
2021-09-13 23:31:30.346: INFO: Something informative
2021-09-13 23:31:30.346: ERROR: Oopsie daisy
```

Demo: stdlib_bitsets

ex_bitsets.f90

```fortran
use stdlib_bitsets
implicit none

integer :: i; type(bitset_64) :: b1, b2

call b1%from_string('001100') ! S6B001110
b2 = [(.true., i=1,6)]        ! S6B111111

call xor(b1, b2)              ! S6B110001, S6B111111

call b1%set(2, 4)             ! S6B111111 -- N.B. 0-based index
print *, b1 == b2             ! T
end
```

Demo: stdlib_sorting

ex_sorting.f90

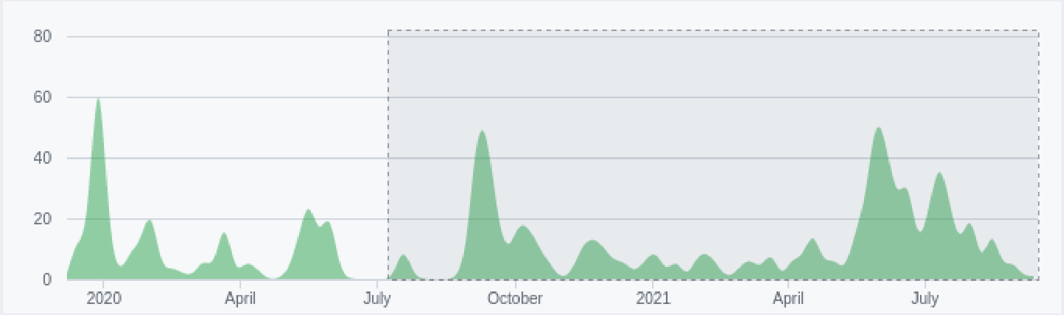
```
use stdlib_sorting, only: sort_index
use stdlib_kinds, only: int64
implicit none

integer :: digits(6) = [3,1,4,1,5,9]
character :: chars(6) = ['a','b','c','d','e','f']
integer(int64) :: index(6)
call sort_index(digits, index)
print '(6i1)', digits       ! 113459
print '(6i1)', index        ! 241356
print '(6a1)', chars(index) ! bdacef
end
```

New contributors have been key to stdlib's growth

- From 16 committors to 34, including our 2 GSoC students
- Over 100 new Issues: bugs, workflow improvements, feature proposals
- From 52 to 97 contributors (commits, discussion, reviews)

Commits to stdlib since FortranCon 2020



stdlib is now easier to install

- Dependencies
    - Fortran compiler (supporting at least F2008)
    - CMake (or just make)
    - fypp preprocessor (python script)
- Install each separately or use conda package manager
- Exports both CMake package files & pkg-config files
- New support for fpm-based workflow

It is now trivial for fpm packages to depend on stdlib

fpm.toml

```
...
[dependencies]
stdlib.git = "https://github.com/fortran-lang/stdlib"
stdlib.branch = "stdlib-fpm"
...
```

It just works!

Cross-platform support monitored with GitHub’s CI workflow

Platforms tested on every pull request

GNU	9,10,11	Ubuntu 20.04	x86_64
GNU	9,10,11	macOS 10.15	x86_64
GNU (MSYS)	10	Windows Server 2019	x86_64
GNU (MinGW)	10	Windows Server 2019	x86_64, i686
Intel classic	2021.1	Ubuntu 20.04	x86_64
Intel classic	2021.1	macOS 10.15	x86_64

- If your compiler supports F2008/F2018, stdlib should compile
- Some require minor workarounds (NAG, some older GNU versions)

Room for improvement

- Fill out numerical capabilities
  - “Simple” functions are often not so simple (e.g., *cbrt*)
  - Difficult to find reviewers with domain knowledge (see: Probability Distributions)
  - What to put in stdlib versus create fpm package?
- Improve consistency of documentation
  - Lots of variability in style & level of detail
  - To be addressed with standardized tempates

Outlook: Next 12 months

- Probability distributions: Uniform, normal, exponential, gamma, and beta
- Generic linked list
- Generic map type
- Hash functions
- Improved OS and file system facilities
- Selection algorithms
- Portability across platforms
- All new intrinsics planned for Fortran 202X
- Improved stdlib test suite

Summary

- stdlib aims to be a de facto standard library of general-purpose and numerical facilities for Fortran
- Roughly doubled in size in the past year, both in terms of modules and contributors
- New modules include bitsets, logging, math utilities, sorting, special functions, RNG, and string handling
- Infrastructure and packaging improvements have made stdlib easier to install and use