Introduction to Julia Programming

Per Rutquist

Julia

- Currently at 0.6
- 0.7 ≈ 1.0

Why Julia?

My new favourite programming language

My previous favoirite languages

- C
- Java
- Matlab
- Python

Advantages of Julia

- Interactive
- Fast
- ...

Interactive languages

Matlab, Python...

- Immediate
- Readable
- Productive

Fast languages

Assembly, C, C++, ...

- Copy-paste
- Templating

Best of both worlds?

Python script calling fast C-routines

Best of both worlds!

Julia script calling

fast Julia-routines

Julia functions run
at the speed of C

Example

```
f(x) = 2x + 1
```

Just-in-time compilation

Julia

↓

LLVM IR

↓

Machine code

Type inference

```
julia> f(x) = 2x + 1
julia> f(1)
3

julia> f(1.0)
3.0
```

Type inference

...at compile time

```
julia> @code_typed f(1)
CodeInfo(:(begin
        return (Base.add_int)((Base.mul_int)(2,
    end))=>Int64

julia> @code_typed f(1.0)
CodeInfo(:(begin
        return (Base.add_float)((Base.mul_float)
    end))=>Float64
```

```
@code_llvm f(1)
```

```
define i64 @julia_f_60708(i64) #0 !dbg !5 {
top:
  %1 = shl i64 %0, 1
  %2 = or i64 %1, 1
  ret i64 %2
}
```

```
@code_native f(1)
```

```
        .section        __TEXT,__text,regular,pu
Filename: REPL[1]
        pushq   %rbp
        movq    %rsp, %rbp
Source line: 1
        leaq    1(%rdi,%rdi), %rax
        popq    %rbp
        retq
        nopl    (%rax,%rax)
```

```
@code_llvm f(1.0)
```

```
define double @julia_f_60729(double) #0 !dbg !5
top:
  %1 = fmul double %0, 2.000000e+00
  %2 = fadd double %1, 1.000000e+00
  ret double %2
}
```

```
@code_native f(1.0)
```

```
        .section        __TEXT,__text,regular,pu
Filename: REPL[1]
        pushq   %rbp
        movq    %rsp, %rbp
Source line: 1
        vaddsd  %xmm0, %xmm0, %xmm0
        movabsq $4808378128, %rax       ## imm =
        vaddsd  (%rax), %xmm0, %xmm0
        popq    %rbp
        retq
        nopl    (%rax,%rax)
```

Julia scripts are slow.

Write functions

Advantages of Julia

- Interactive
- Fast
- Multiple dispatch
- ...

Multiple dispatch

```
f(x,y) = 2x + y
f(x::String, y) = "Hello!"
f(x, y::String) = "Julia!"
f(x::String, y::String) = "Hello, Julia!"
```

Advantages of Julia

- Interactive
- Fast
- Multiple dispatch
- Metaprogramming

Metaprogramming

- eval, interpolation
- Macros
- Generated functions

Advantages of Julia

- Interactive
- Fast
- Multiple dispatch
- Metaprogramming
- Packages

Julia Packages

- Lots
- Good
- Written in Julia!

- Interactive
- Fast
- Multiple dispatch
- Metaprogramming
- Packages
- Lots more

Discussion!

- Questions?
- Interactive session?