



Curso de Julia

Episodio 1: Contexto y Sintaxis



Por: Dr. Jesús Adolfo Mejía de Dios
8 de mayo de 2023

Agenda

Este episodio tiene dos partes:

Parte I: Contexto

Parte II: Sintaxis

La primera parte explicará la motivación del uso de Julia y el contexto de su creación. Luego, en la segunda parte se hablará de la sintaxis del lenguaje.

Contexto

¿Qué hace al lenguaje Julia tan atractivo?

- ✓ Es un lenguaje de **alto rendimiento**.
- ✓ Es de **tipado dinámico** y opcional.
- ✓ **Sintaxis fácil** de aprender.
- ✓ Tiene una **extensa biblioteca** estándar (álgebra lineal, estadística, cálculo).
- ✓ Se puede **conectar a otros lenguajes** de manera casi trivial.

Características

```
julia
julia> f(x) = x^2 - x + 1
f (generic function with 1 method)

julia> f(1)
1

julia> f(1.1)
1.11

julia> f(1//2)
3//4

julia> f(1 + 2im)
-3 + 2im
```



Sintaxis fácil de aprender.

```
julia> x = rand(100_000);

julia> @time fx = f.(x);
0.000451 seconds (4 allocations: 781.328 KiB)
```

Características

- ✓ Se puede **conectar a otros lenguajes** de manera casi trivial.

Interoperando con python

```
using PyCall

# código en python
py"""
import numpy as np

def uno(x):
    return np.sin(x) ** 2 + np.cos(x) ** 2
"""

dos(x) = py"uno"(x) + py"uno"(x)
```

Características

- ✓ Se puede **conectar a otros lenguajes** de manera casi trivial.

Programa en C:
mean.c

```
double mean(int *A, int n) {  
    int i, sum=0;  
    for (i=0; i<n; ++i) sum = sum + A[i];  
    return sum / (double) n;  
}
```

Programa en Julia:
programa.jl

```
x = [10, 5, 9, 8 7]  
m = ccall(# función biblioteca  
          (:mean, "mean.so"),  
          # tipo que retorna la función  
          Float64,  
          # tipo de los argumentos  
          (Ptr{Cint},Cint),  
          # argumentos  
          x, 5  
          )  
  
println("Mean = ", m)
```

Características

$$\begin{array}{ll}\min & 12x + 20y \\ \text{s.t.} & 6x + 8y \geq 100 \\ & 7x + 12y \geq 120 \\ & x \geq 0 \\ & y \in [0, 3]\end{array}$$

<https://jump.dev/JuMP.jl>

```
using JuMP, HiGHS
```

```
model = Model(HiGHS.Optimizer)
```

```
@variable(model, x ≥ 0)
```

```
@variable(model, 0 ≤ y ≤ 3)
```

```
@constraint(model, c1, 6x + 8y ≥ 100)
```

```
@constraint(model, c2, 7x + 12y ≥ 120)
```

```
@objective(model, Min, 12x + 20y)
```

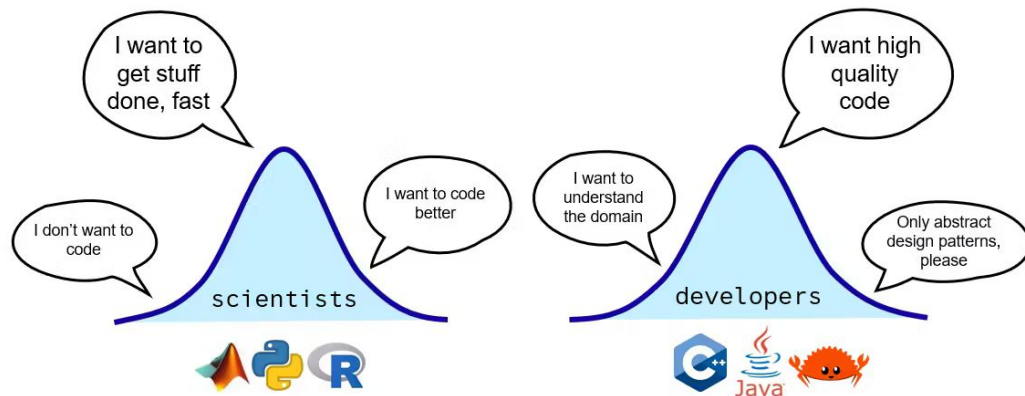
```
optimize!(model)
```

```
objective_value(model) # 204.99999999999997
```

```
value(x) # 15.0000000000000005
```

```
value(y) # 1.2499999999999996
```

Contexto

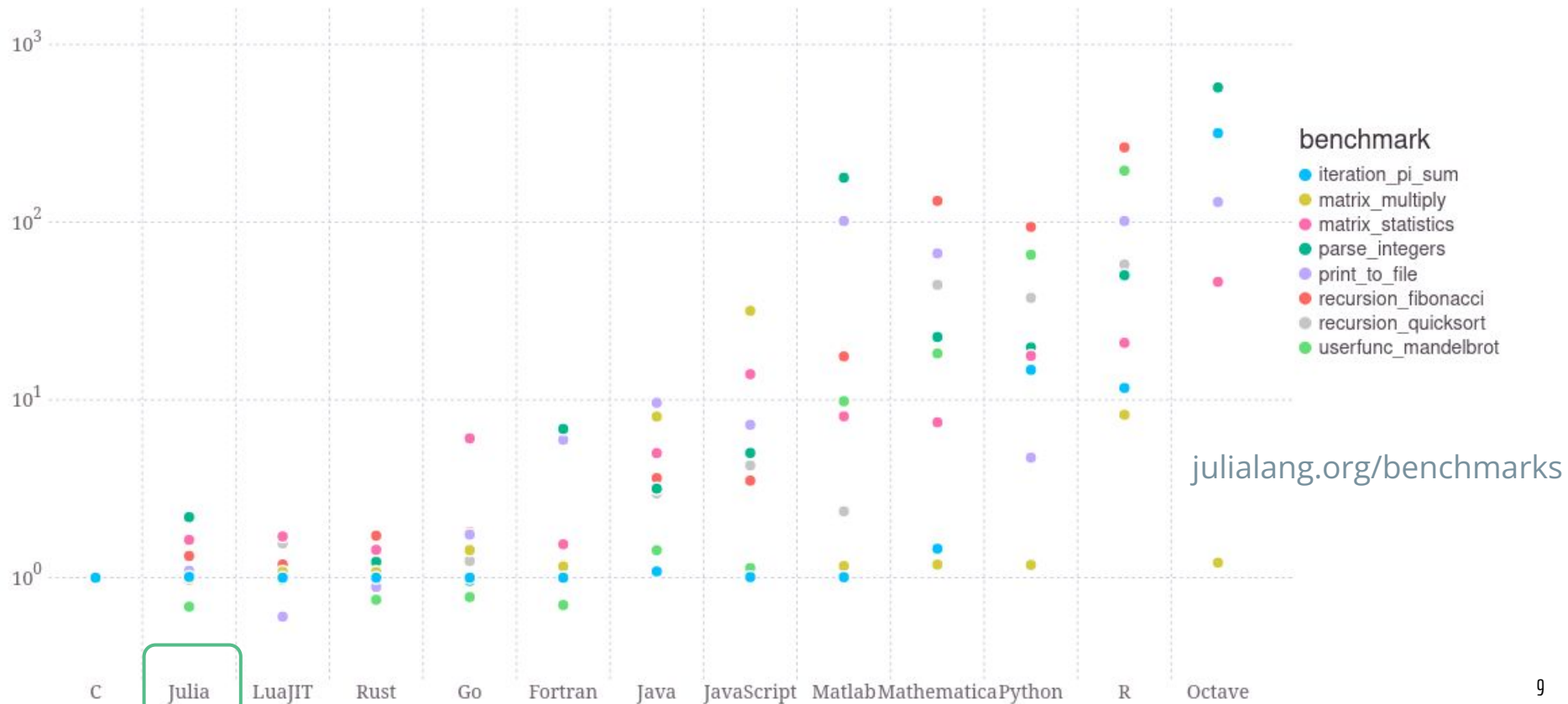


¿Para quién está dirigido?

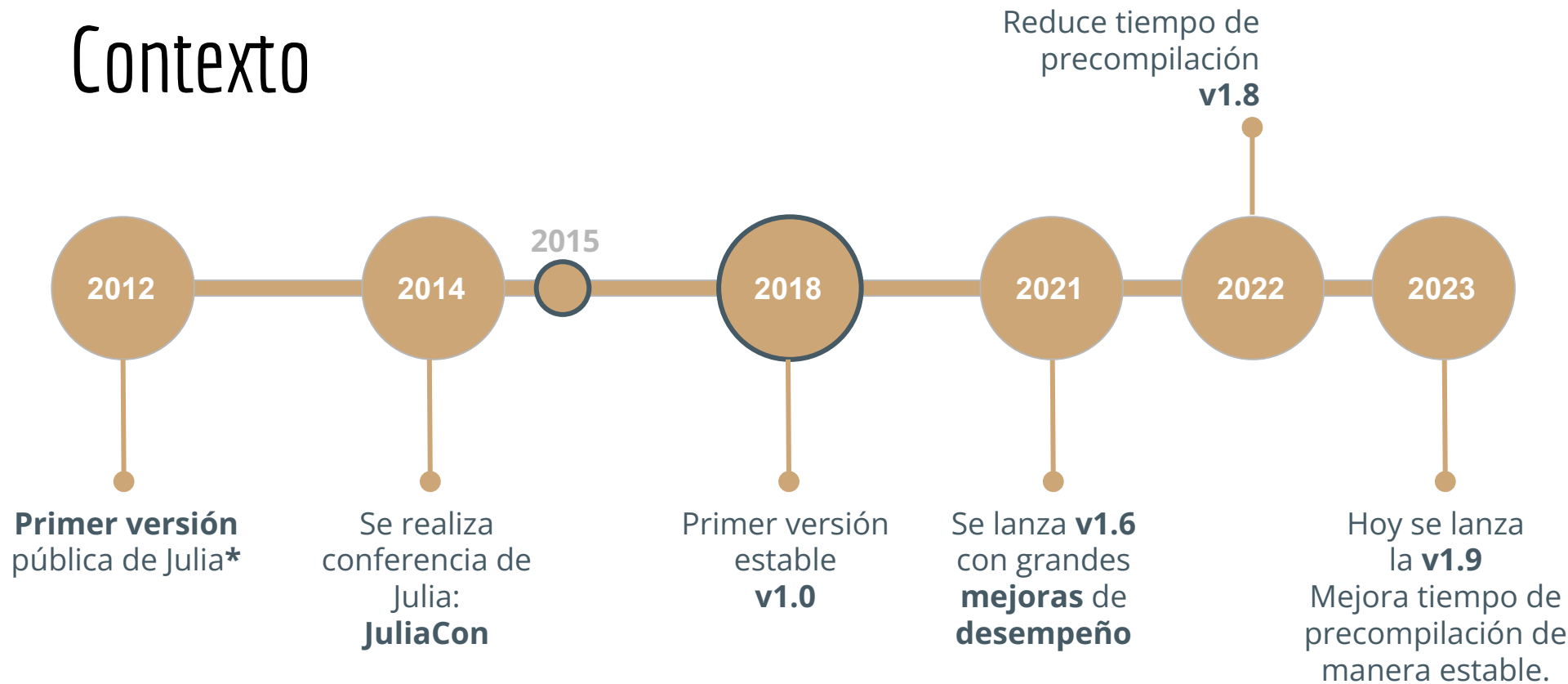
El lenguaje de programación es de **propósito general**, sin embargo la **comunidad científica** le podrá sacar mayor provecho el alto desempeño del lenguaje.



Contexto



Contexto




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

Consola

Julia tiene una **consola** (REPL) **muy poderosa**. En ella se puede **programar**, **instalar** o eliminar **paquetes**, buscar **documentación**, y para interoperar con otros lenguajes.


Modos de la consola:

- Modo **julia>** Modo por defecto.
- Modo **pkg>** Gestor de **paquetes** (se abre con **]**).
- Modo **help>** Buscar **documentación** (se abre con **?**).
- Modo **shell>** Acceder a comandos del **sistema** (se abre con **;**).



A continuación...

Parte II: Sintaxis



`cursos.ooptim.com/julia2023`