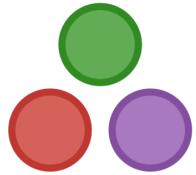


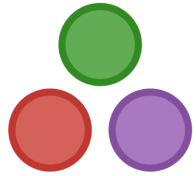
Introducción al lenguaje Julia

Francisco Gárate

Jornadas Medialab-Prado

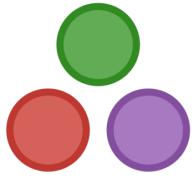
13-14, febrero 2018

¿Quién es Julia?

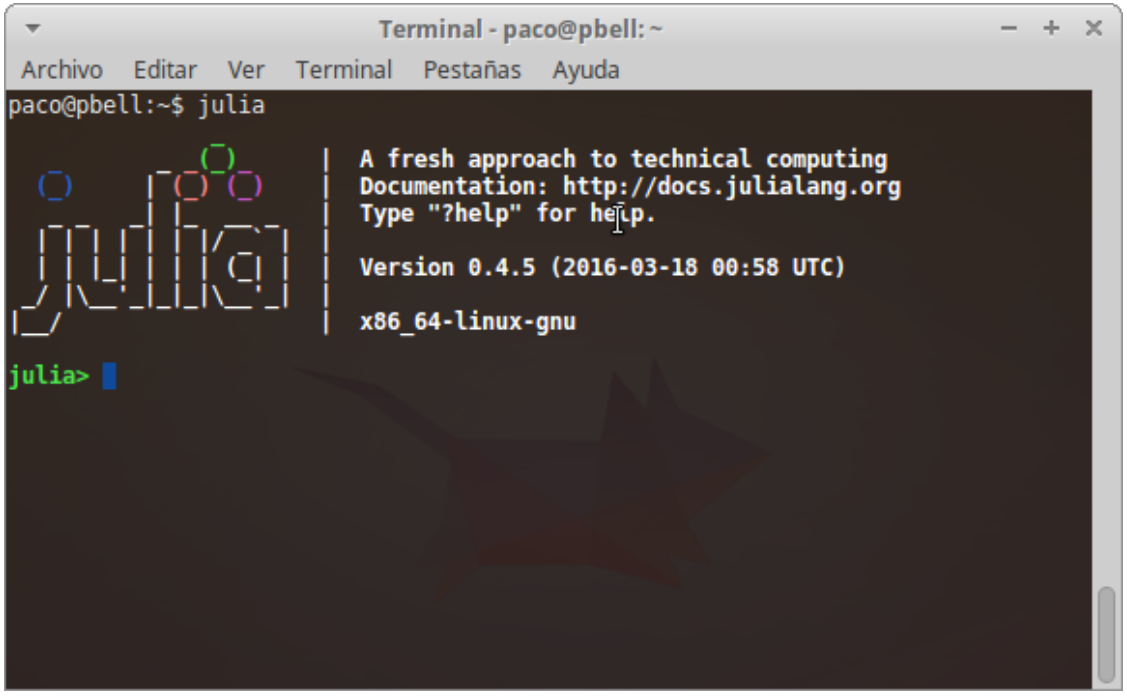


- Julia es un lenguaje de programación libre, de código abierto (licencia MIT) y con librerías amigables.
- Creado en el MIT por Jeff Bezanson, Alan Edelman, Stefan Karpinski y Viral Shah en 2009, y publicado en 2012.
- Diseñado especialmente para entornos científicos y matemáticos.
- *"Walks like Python. Runs like C"*. Se lee igual que Python o Octave, pero se comporta casi igual de rápido que C.
- Lenguaje diseñado para correr en paralelo y en la nube fácilmente.
- Documentación: `https://docs.julialang.org`

Instalar Julia

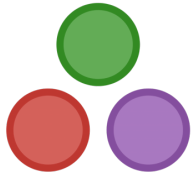


- Descargar instalador: `http://julialang.org/downloads/`
- En linux: `apt-get install julia`
- Julia incluye un terminal interactivo llamado REPL:

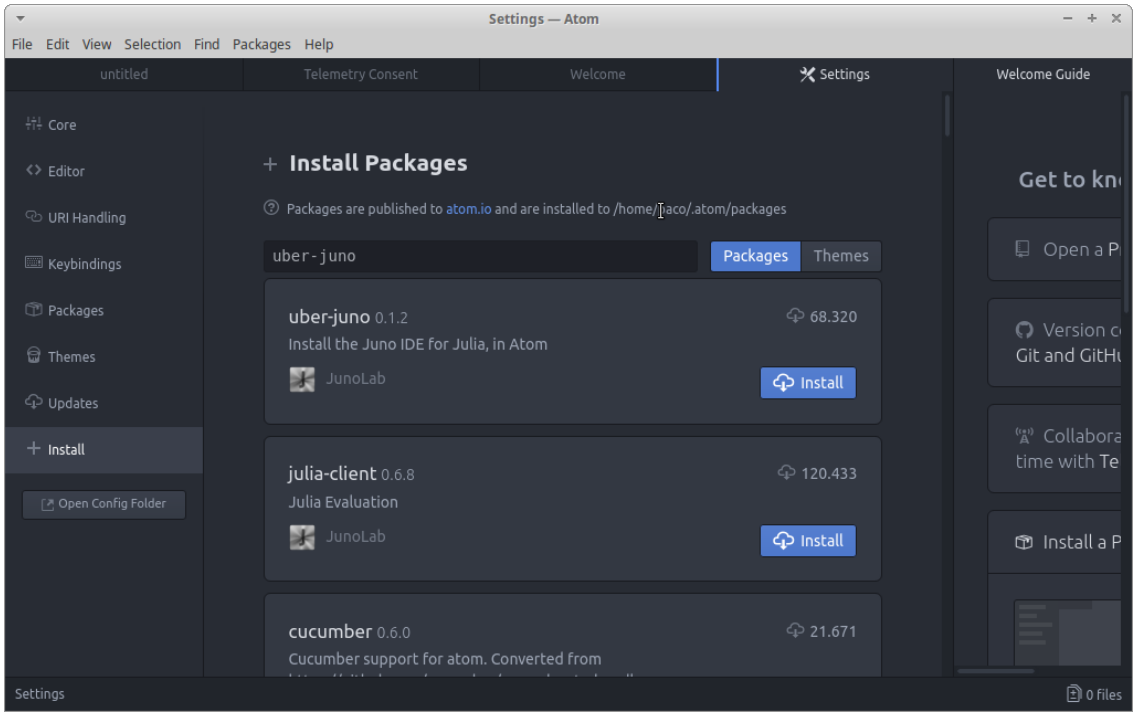


  - `;` La siguiente instrucción se ejecuta como si fuera una consola de linux
  - `?` Busca el texto a introducir en los manuales de ayuda

IDE

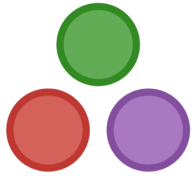


- Soporte para: Vi(m), emacs, Eclipse, Sublime Text...
- Recomendación: Atom + Juno (`http://junolab.org`)

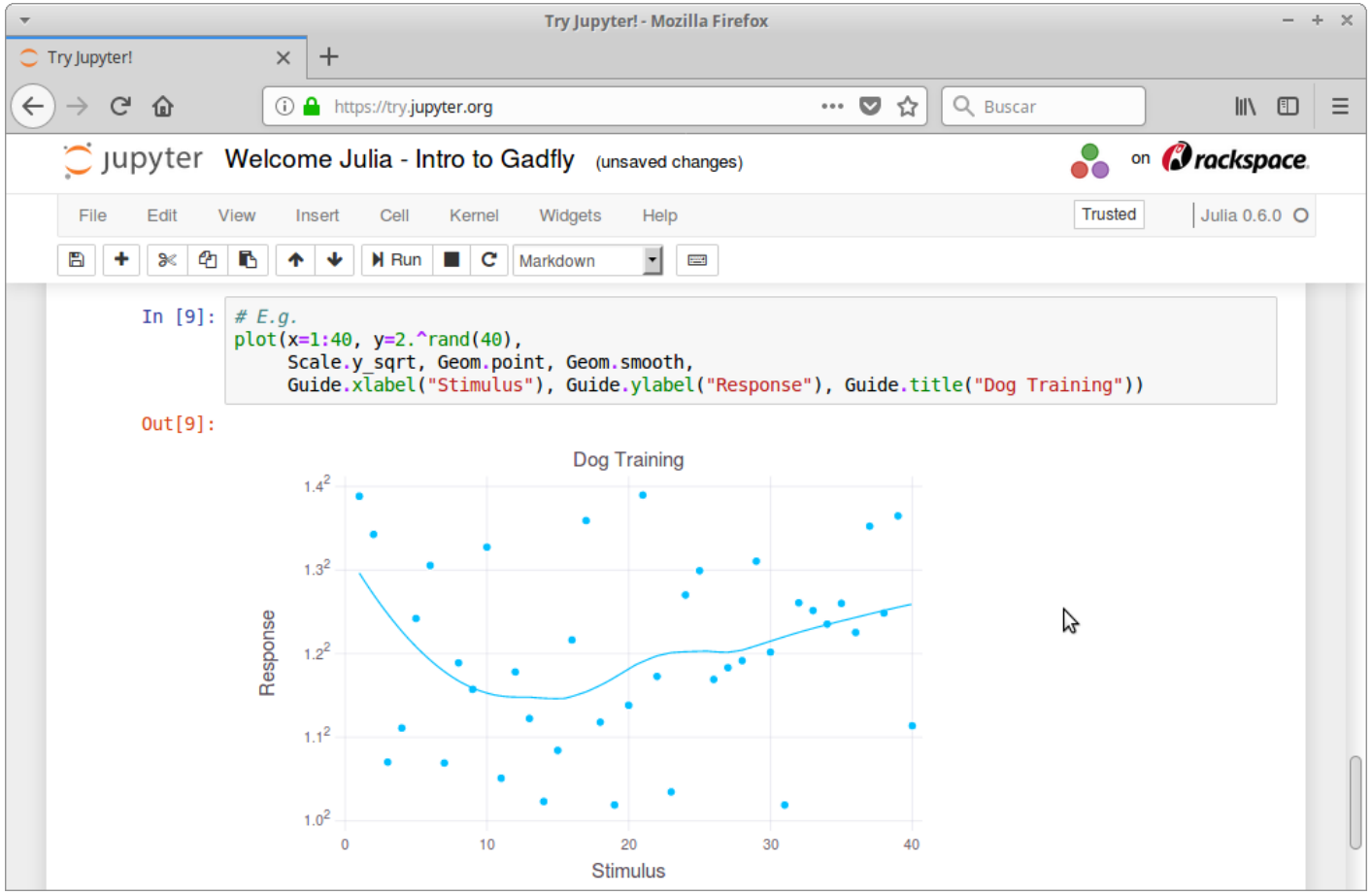


desde Atom: `Ctrl ,` o `Cmd ,`, Install, 'uber-juno'

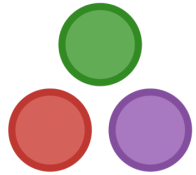
Prueba Julia (sin instalar)



- Jupyter Notebook: IJulia (similar a IPython): `https://try.jupyter.org`
- `www.juliabox.com` Incluye +100 paquetes

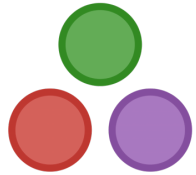


Paquetes (Packages)



- Repositorio oficial: `http://pkg.julialang.org`
- Existen actualmente 1.703 paquetes registrados y testeados: `http://pkg.julialang.org/pulse.html`
- Para instalar: `Pkg.add("Distributions")`
- Desinstalar: `Pkg.rm("Distributions")`
- Para ver el estado de tus paquetes: `Pkg.status()`

Hola Mundo



- `println("Hola Mundo")`
- Desde shell: `julia holamundo.jl`
- Operadores `+, -, *, /`

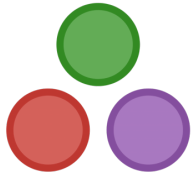
```
a = 4.0
b = 7.0
c = (b - a) * (b + a) / a
```

- Funciones matemáticas familiares:

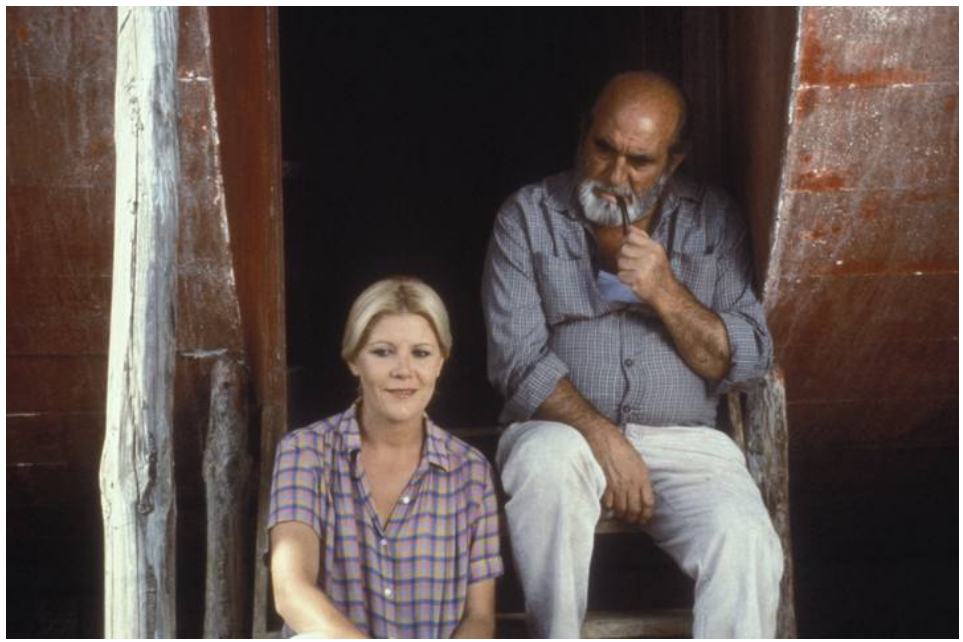
```
result = exp(-2.33) * cos(22 * 180 / pi)
```

- Funciones incluidas de serie. Por ejemplo:
  `rand()` *(genera números aleatorios entre 0 y 1)*
  `std(v)/mean(v)`
- Condicionales, listas, loops, rangos, etc.. al estilo Python

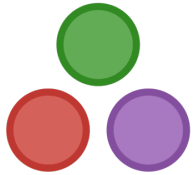
¿Por qué Julia?



Teniendo R, Matlab, Mathematica, Python...
¿De verdad hace falta otro lenguaje de programación?



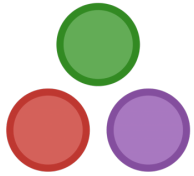
¿Por qué Julia?



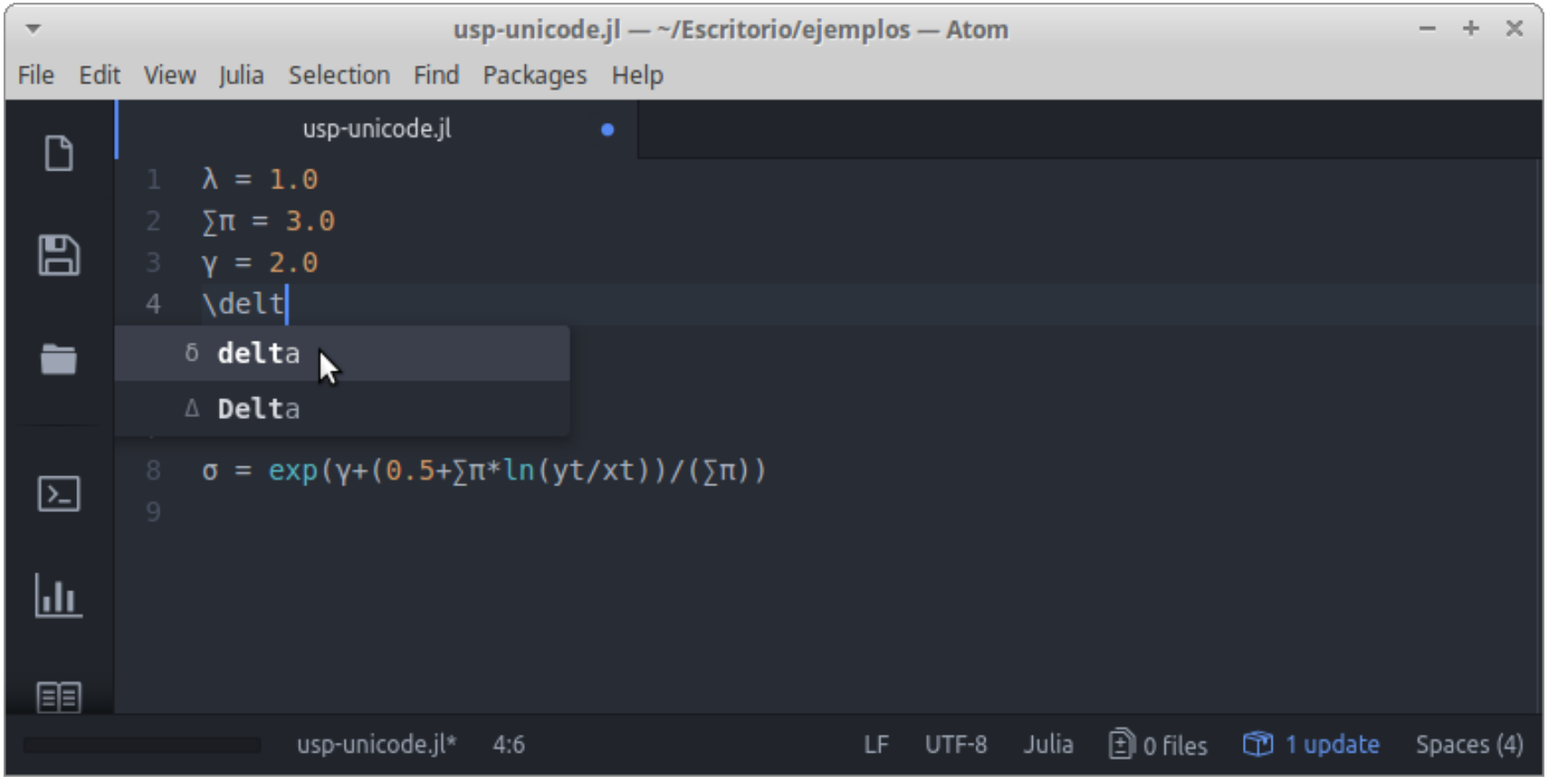
- Julia no es popular.
- No aparece en el ranking de los lenguajes más populares de GitHub ni en el Top 20 Most Popular de Stackoverflow.
- Tampoco la NASA o el CERN han liberado ninguna librería (de momento).
- ... pero eso también mola!
- Veamos las ventajas de Julia frente a otros lenguajes:

Ventajas

Soporte Unicode

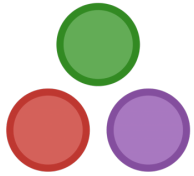


- Se introducen símbolos al igual que LaTeX



Ventajas

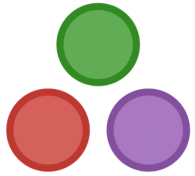
Compilador JIT

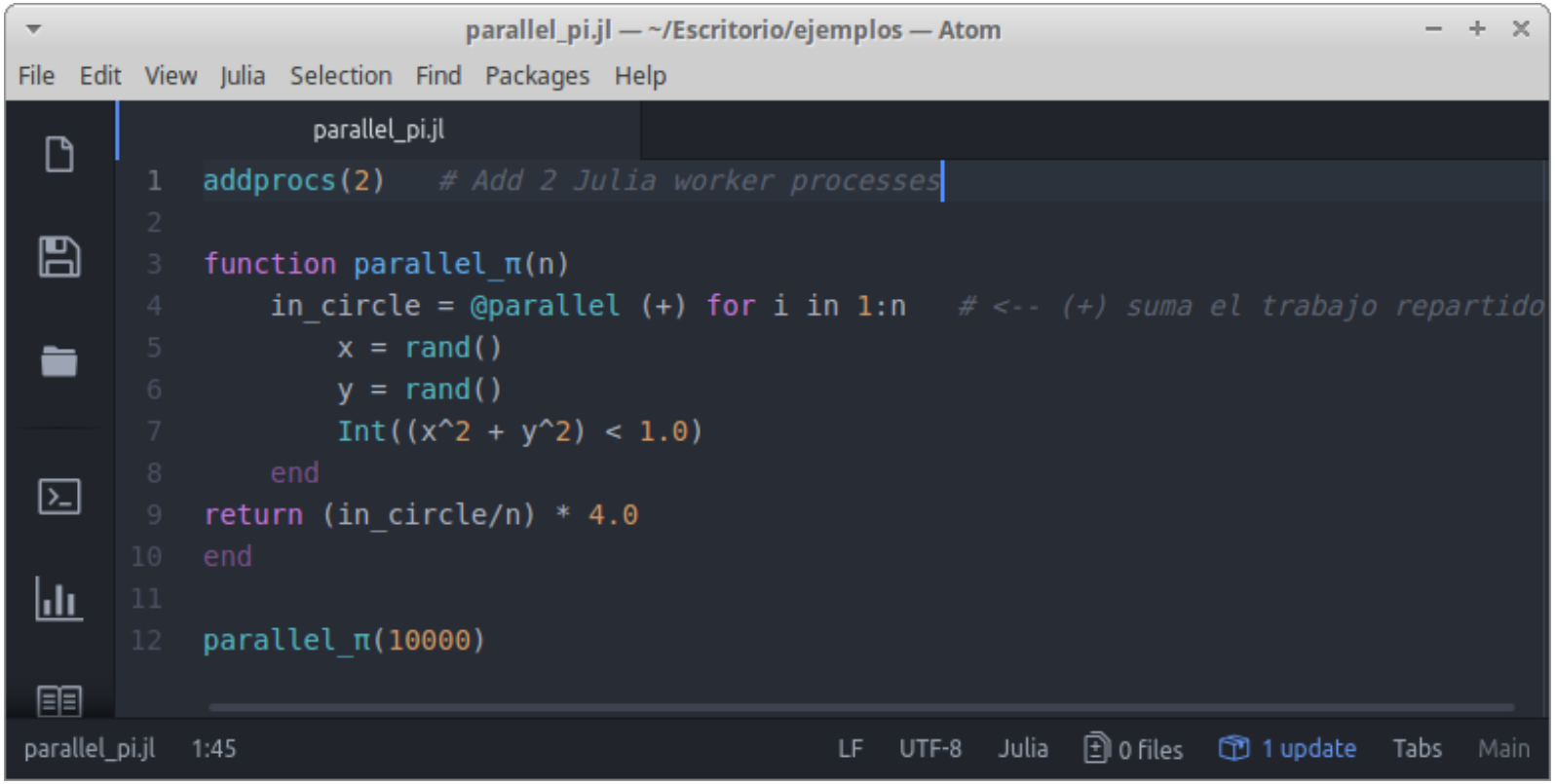


- Su compilador JIT (Just-In-Time) basado en LLVM (Máquina Virtual de Bajo Nivel) permite a Julia acercarse, e incluso igualar, a C.

Ventajas

Computación en paralelo de forma nativa

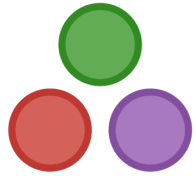




- Repositorio de modelos para programación en paralelo: `https://juliaparallel.github.io`

Ventajas

Otras razones



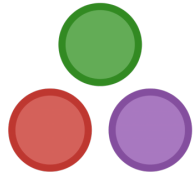
- Licencia MIT (compatible con GNU GPL)
- Robusto ecosistema de herramientas estadísticas, distribución en paralela, optimización y de visualización de datos.
- Librerías gráficas (como Gadfly) para jugar en primera división.
- Llamadas a librerías C, Fortran de forma nativa:

```
julia>ccall(:clock, Int32, ())
1180595
```

- ...y a Python (a través de PyCall.jl)
- Sintaxis familiares a Python
- En resumen: Unir facilidad de uso y alto rendimiento es posible.

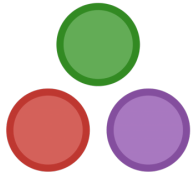
Casos reales

https://juliacomputing.com/case-studies/



- Celeste.jl: Librería que ha incrementado x225 la velocidad del análisis de imágenes (dataset de 55 TB) ayudando al National Energy Research Scientic Computing Center (NERSC) en la catalogación de objetos astronómicos.
- QuantEcon.jl: Librería desarrollada por QuantEcon, para el Banco de la Reserva Federal de NY, utilizada para la previsión y análisis monetarios (10 veces más rápido que MATLAB).
- Lora.jl y Mamba.jl: Los investigadores del cáncer del Reino Unido recurrieron a Julia para ejecutar simulaciones de crecimiento tumoral, utilizando métodos bayesianos MCMC.
- IM3 y Spark.jl: AVIVA migró a Julia su software comercial de cálculo de capital regulatorio (Solvencia II), logrando correr 100.000 simulaciones en 12 segundos (1.000 veces más rápido que IBM Algorithmics) y reduciendo un 93% el número de lineas de código.

#JuliaLangEs



- Se buscan voluntarios para:
  - Dar a conocer el lenguaje Julia
  - Traducir la documentación oficial al castellano
  - Crear/Potenciar una comunidad de usuarios de Julia en español (blog, meetup, etc...)
  - Crear o migrar librerías de R/Python a Julia
- Mas info: fgaratesantiago@gmail.com
- Repositorio de esta presentación:
  - `https://github.com/franciscogarate/Introduccion-al-lenguaje-Julia`
- ¡Muchas gracias por la atención!