Professorship of Business Analytics & Intelligent Systems
TUM School of Management
Technical University of Munich

Julia Programming Language – a brief introduction

ÖGOR Workshop Krems

25.07.2024

Why another language?

Especially in the scientific context, there is a need for rapid prototyping (using dynamically typed languages like Python) and high performance (using compiled languages like C). Julia aims to solve this "**Two-language"** problem, using **just-in-time (JIT) compilation,** implemented by **LLVM**.

Compared to other dynamically typed languages like Python (excerpt from https://docs.julialang.org/en/v1/)

- The core language imposes very little; Julia Base and the standard library are written in Julia itself, including primitive operations like integer arithmetic.
- A rich language of types for constructing and describing objects, that can also optionally be used to make type declarations.
- The ability to define function behavior across many combinations of argument types via **multiple dispatch**.
- Automatic generation of efficient, specialized code for different argument types.
- Good performance, approaching that of statically-compiled languages like C.

How to install Julia

Via installation manager juliaup (default) - follow the instruction at https://julialang.org/downloads/

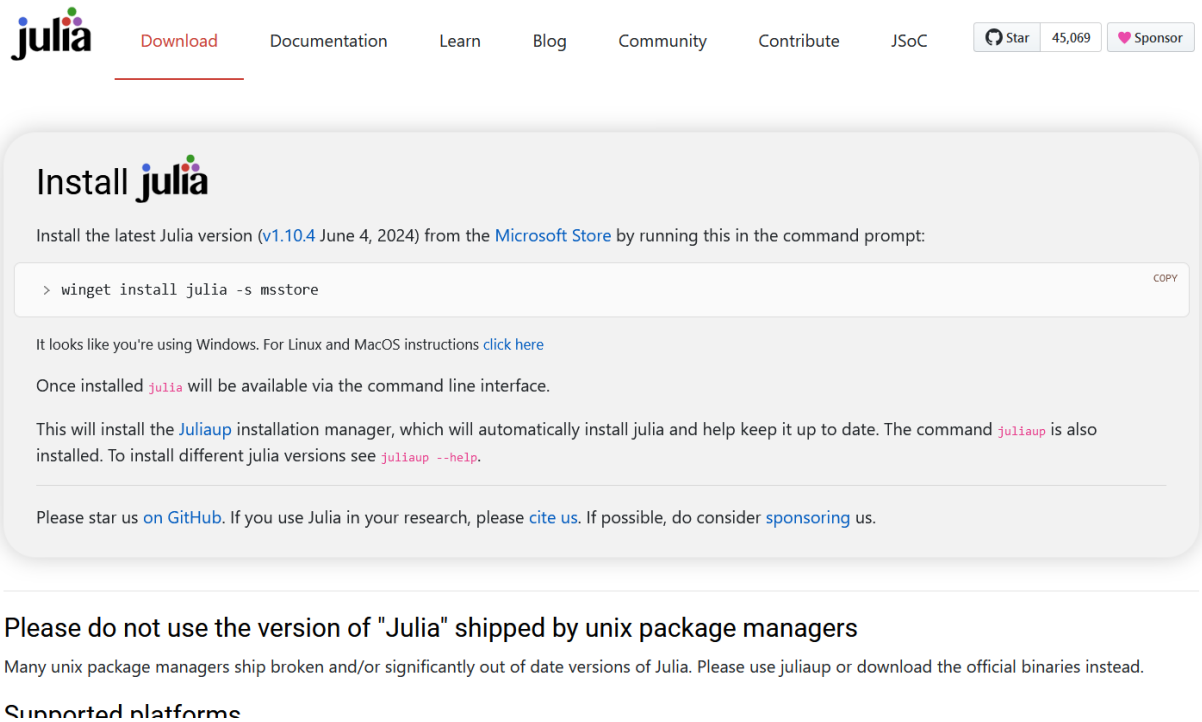
(run `juliaup update` in your terminal to fetch and install the latest version of Julia)

Editor Plugins:

- VSCode
- Vim
- Emacs

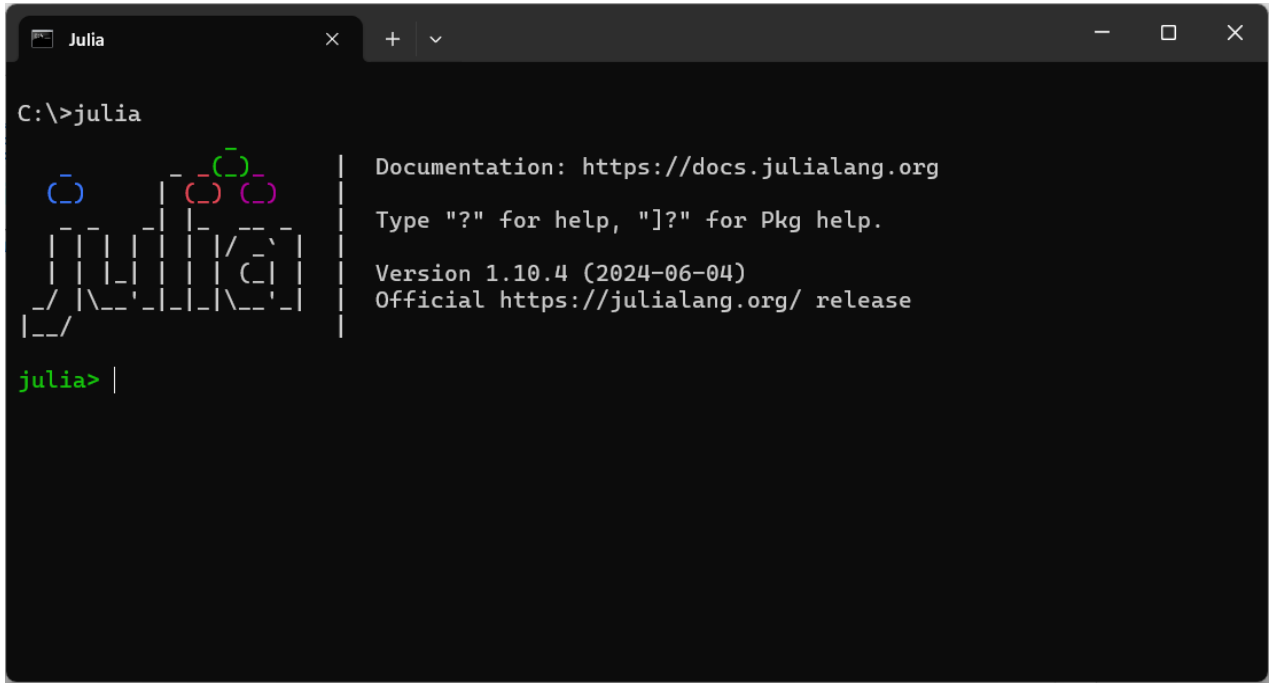
Notebooks

- Jupyter
- Pluto



Julia REPL

Start from the terminal by running `julia`



3 modes:

- REPL: julia>
  - here you can run your statements
- Pkg: (@v1.10) pkg
  - from REPL by typing `]` (closing brackets)
  - to install new packages
    e.g., `add Pluto`, install Pluto package
  - exit by pressing backspace
- Shell: shell>
  - from REPL by typing ';' (semicolon)
  - access the terminal (shell) to create files etc.
  - exit by pressing backspace

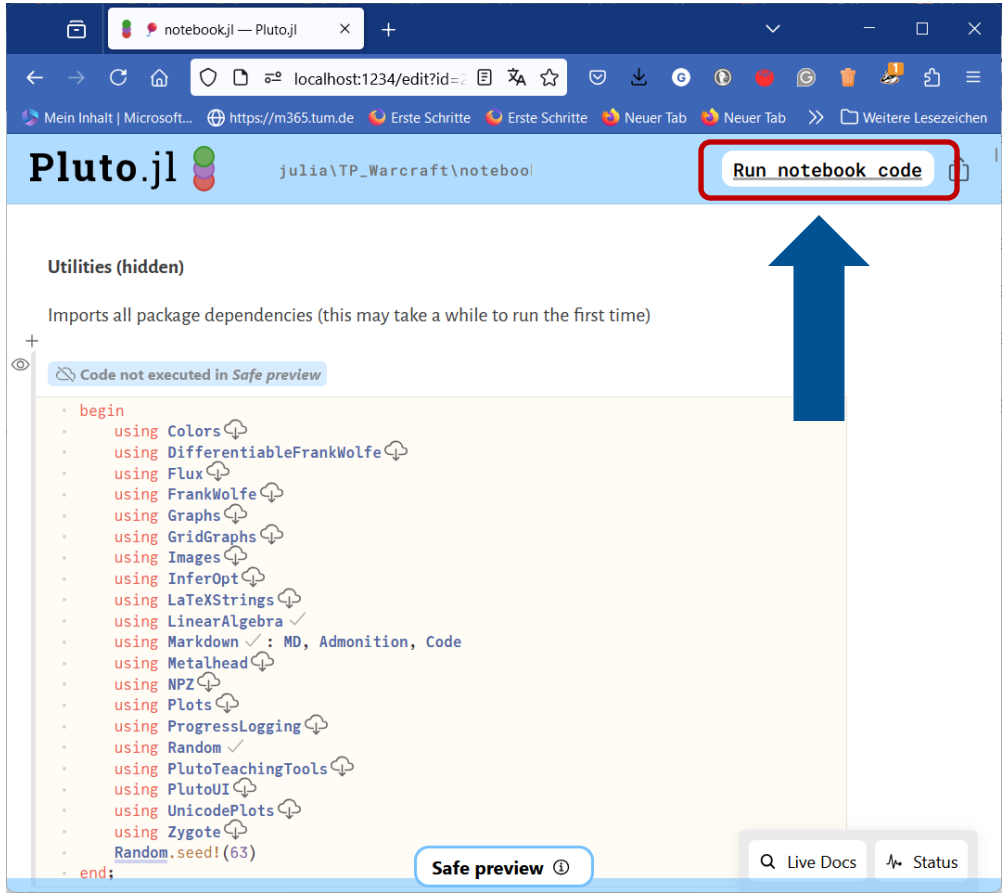
How to setup the Pluto notebook

Clone the repository from https://github.com/BatyLeo/TP_Warcraft

Follow the instruction in the Readme, i.e.,

- Download the dataset
- Open a REPL in the directory
- Run:
  - `using Pluto`
  - `Pluto.run()`
- Select `notebook.jl`

- **Click `Run notebook code`**
  - This will take some time!



In the meantime .. some Julia basics!

Useful resources:

- Website: https://julialang.org/

- Official documentation: https://docs.julialang.org/en/v1/
  - Style Guide: https://docs.julialang.org/en/v1/manual/style-guide/
  - Performance tips: https://docs.julialang.org/en/v1/manual/performance-tips/
  - Noteworthy Differences: https://docs.julialang.org/en/v1/manual/noteworthy-differences/

- Tutorials: https://juliaacademy.com/

- Youtube: https://www.youtube.com/@TheJuliaLanguage

- InferOpt Docs: https://juliadecisionfocusedlearning.github.io/InferOpt.jl/stable/