ToolingDocumentationCommunity

May 26, 2017

1 Tooling, Documentation, and Community

1.1 Tooling

The two most commons tools for coding in Julia are:

1. The Julia REPL (the terminal)
2. Juno, the IDE in Atom
3. Jupyter notebooks

Other solutions exist (SublimeText, Emacs, etc.). The general mantra is:

> Jupyter notebooks are for code, Juno is for data. (Mike Innes, developer of Juno)

Jupyter notebooks do not easily show code from multiple files, do not have a debugging GUI, and it doesn't have as easy access to a console, making it less friendly for developing large projects (like packages). However, Jupyter notebooks store their outputs in a convenient form, which is good for saving and sharing results. I suggest that you learn both and use the right tool for the right job.

1.1.1 Jupyter Notebooks with no Install

Go to www.juliabox.com and sign in to start using Julia on a cloud-based Jupyter notebook. (Note: you may need to use `Pkg.update()` to update your local package database ("METADATA") on first use).

To setup the workshop notebooks in JuliaBox, click on the "Sync" tab and under "Git Repositories" paste in https://github.com/UCIDataScienceInitiative/IntroToJulia for the "Git Clone URL", and click the +. Upon refreshing the homepage you should see the IntroToJulia folder appear. Click into this folder, and click into the Notebooks folder.

1.1.2 Installing Julia

Unless you're a serious hacker, I recommend installing Julia by downloading one of the binaries from http://julialang.org/downloads/. There are ready-to-use binaries for most operating systems. Just download, run the installer, and use Julia.

If you want to build from the source, the repository along with the build instructions are at https://github.com/JuliaLang/julia.

1.1.3 Installing Jupyter Notebooks

To install Jupyter notebooks (IJulia), use the following commands in the REPL:

```
In [ ]: Pkg.add("IJulia") # This will install the package
        Pkg.clone("https://github.com/UCIDataScienceInitiative/IntroToJulia") # Clo
        ### Next commands you run each time you open the REPL
        using IJulia
        notebook(detached=true,dir=Pkg.dir("IntroToJulia/Notebooks"))
```

1.1.4 Installing Juno

To install Juno, use the installation instructions from here: http://docs.junolab.org/latest/

1.2 Installing Libraries

For later in the workshop you will need some libraries installed. Suggested libraries to install are:

- Plots.jl for plotting. This will require some backends:
- PyPlot.jl
- GR.jl
- Plotly.jl
- MultivariateStats.jl
- GLM.jl

Optional libraries, depending on which projects you explore later, include:

- DifferentialEquations.jl
- LightGraphs.jl
- ForwardDiff.jl
- NLsolve.jl
- JuMP
- BenchmarkTools.jl
- ProfileView.jl
- PkgDev.jl

1.3 Documentation

The Julia documentation can be found at http://docs.julialang.org/en/latest/manual/
Documentation can be found from the REPL by using `?` in front of a command.
The Juno documentation can be found at http://docs.junolab.org/latest/

1.3.1 Good Pages to Read

The following documentation pages are good pages to read in full:

- Noteworthy Differences from Other Languages
- Performance Tips
- MATLAB, Python, Julia Syntax Comparison

1.4 Community

Julia is changing fast: there are updates every few months which bring in loads of new features, packages are updating all of the time, and the documentation, while good, is not entirely complete. The best fix for these problems is to be engaged in the Julia community.

I would highly recommend finding a good online community while learning Julia The main Julia community resources are:

- The JuliaLang Gitter - A chatroom for Julia users
- The Julia Discourse - A forum / mailing list for Julia users. Frequented by core developers.
- The JuliaLang Github Repository - The issues/PRs document everything changing in Julia
- JuliaBloggers - A blog aggregator for Julia
- r/Julia - A subreddit for Julia
- StackOverflow tag:julia-lang - Q&A site for Julia