Julia
Portability and Scalability

Ludwig Böss

28.07.2020

1 Packages

You can find a step-by-step guide on how to create a packe in Julia here: `https://tlienart.github.io/pub/julia/dev-pkg.html`.

1.1 Unit tests

The Julia documentation on unit tests can be found here: `https://docs.julialang.org/en/v1/stdlib/Test/`.
Feel free to use the .travis.yml file from the repo as a reference.

1.2 Documentation

You can find information here: `https://juliadocs.github.io/Documenter.jl/stable/man/guide/#Package-Guide`.
To deploy the documentation follow the description in 'Hosting Documentation'.
The travis script I supplied helps you set up the deployment via travis, you only need to generate and add the SSH keys.
Please not that the 'stable' build only happens when you add a Tag in your repo to indicate a stable release.

1.3 CompatHelper.jl

I did not cover this in the talk, but if you want to avoid having to worry about the versions of the dependencies in your package you can define a github action called 'CompatHelper'.
This will check for updates on your dependencies in a seperate branch every night. If there are new versions available it will create a pull request to your master branch.
The pull request will in turn trigger a build in Travis. If your unit tests pass you can merge the pull request and always have your dependencies up to date. More info here: `https://github.com/JuliaRegistries/CompatHelper.jl`

1.4 Registrator

If you want to register your package in the general Julia registry (think of it as making it available for install via conda in Python) you can do this simply installing the Julia registrator and commenting '@JuliaRegistrator register' under your latest commit.
Please note that you have to fulfill some guidelines and can only register versions e.g. 'v0.0.1', v'0.1.0' or 'v1.0.0'.
There is a 3 day waiting period and after that your package is merged automatically, if all checks pass.
To update a version you comment the same thing under the commit you want to update.
More info here: `https://github.com/JuliaRegistries/Registrator.jl`

1.5 TagBot

Once you added your package to the registry you may want to have something that takes care of your versioning.
'TagBot' provides this functionality by checking the version in your 'Project.toml' and if that changed and you registered an update to your package it will automatically create a Tag with the changes since your last release.
More info here: `https://github.com/marketplace/actions/julia-tagbot`

2 Parallelism

Here are some resources if you want to read up on how to write parallel code in Julia.

2.1 Share Memory

https://docs.julialang.org/en/v1/manual/parallel-computing/#man-multithreading-1
https://julialang.org/blog/2019/07/multithreading/

2.2 Distributed Memory

https://docs.julialang.org/en/v1/manual/parallel-computing/#Multi-Core-or-Distributed-Processing-1
https://docs.julialang.org/en/v1/manual/parallel-computing/#man-shared-arrays-1