JuliaEO Workshop 2024:

Introduction to Julia: Creating Packages for Earth Observational Work

Presented By: Nathanael Wong
material at https://github.com/natgeo-wong/JuliaEO2024_Nat

1

Notes

- This presentation (especially the parts using Terminal) are geared towards ***Linux*** and ***macOS*** users
  - Some familiarity with ***Terminal*** is required (don't worry, I'll go through step by step)
  - For ***Windows*** users, I recommend using the ***Windows Terminal***

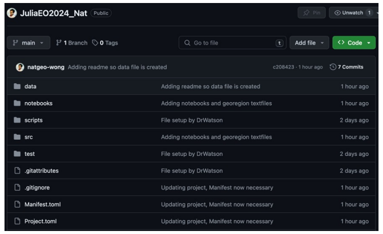
- It is also good to learn how to fiddle around and thus understand the concepts of ***package environment*** especially when developing packages

2

Notes

- Go to https://github.com/natgeo-wong/JuliaEO2024_Nat

- Clone the repository there



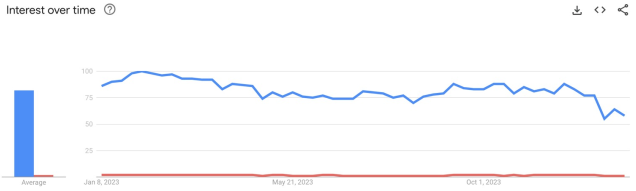
natgeo-wong Adding readme so data file is created c208423 · 1 hour ago 7 Commits

data	Adding readme so data file is created	1 hour ago
notebooks	Adding notebooks and georegion textfiles	1 hour ago
scripts	File setup by DrWatson	2 days ago
src	Adding notebooks and georegion textfiles	1 hour ago
test	File setup by DrWatson	2 days ago
.gitattributes	File setup by DrWatson	2 days ago
.gitignore	Updating project, Manifest now necessary	1 hour ago
Manifest.toml	Updating project, Manifest now necessary	1 hour ago
Project.toml	Updating project, Manifest now necessary	1 hour ago

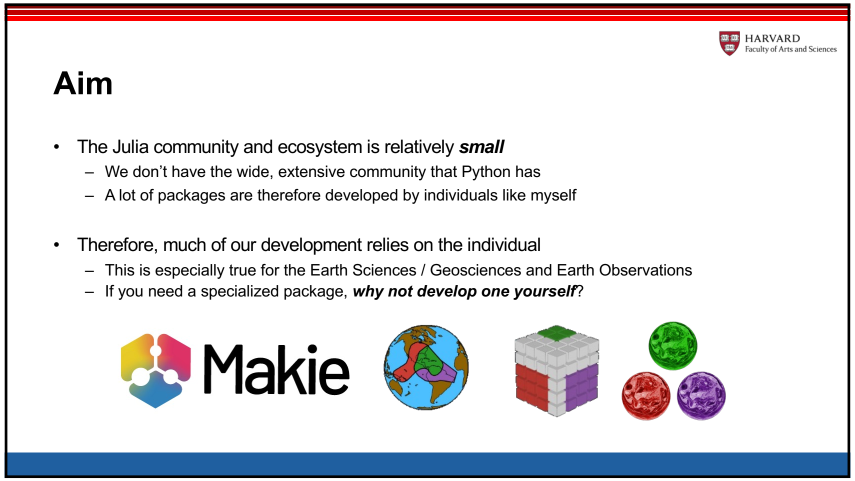
3

Aim

- The Julia community and ecosystem is relatively ***small***
  - We don't have the wide, extensive community that Python has
  - A lot of packages are therefore developed by individuals like myself



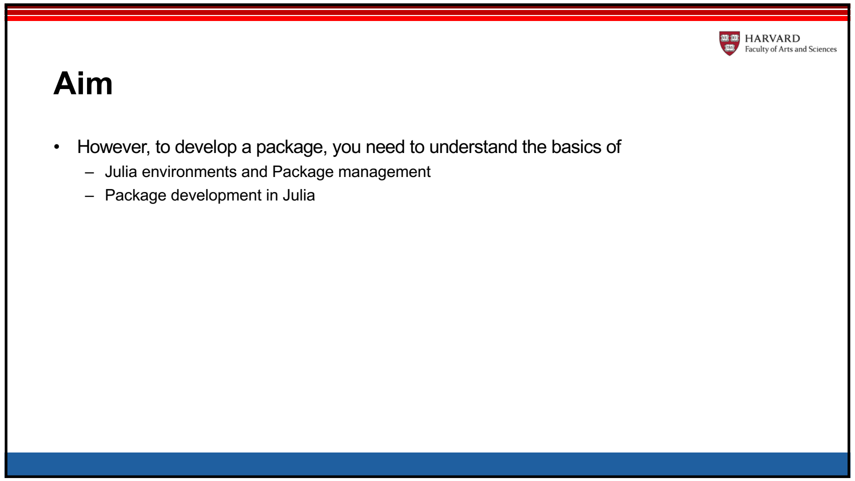
4



Aim

- The Julia community and ecosystem is relatively ***small***
  - We don't have the wide, extensive community that Python has
  - A lot of packages are therefore developed by individuals like myself
- Therefore, much of our development relies on the individual
  - This is especially true for the Earth Sciences / Geosciences and Earth Observations
  - If you need a specialized package, ***why not develop one yourself***?

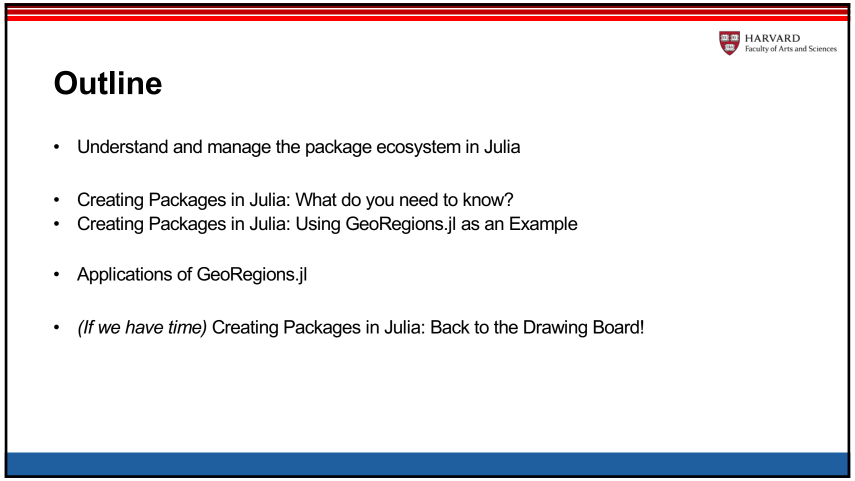
5



Aim

- However, to develop a package, you need to understand the basics of
  - Julia environments and Package management
  - Package development in Julia

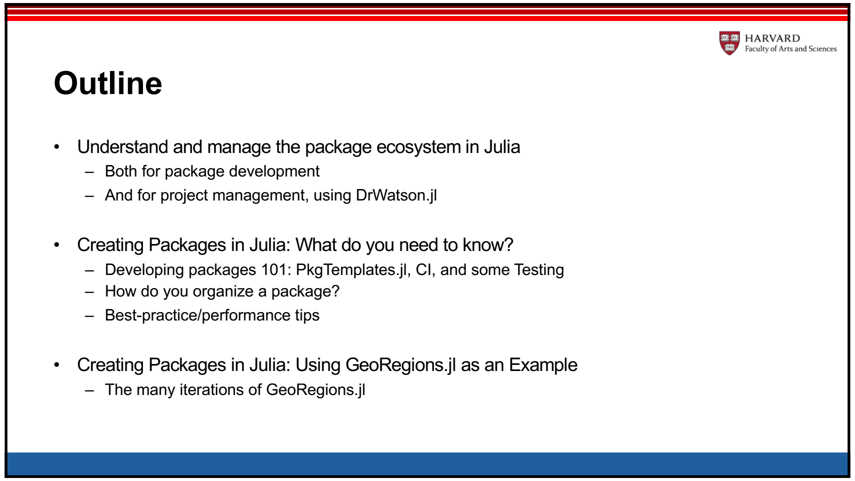
6



Outline

- Understand and manage the package ecosystem in Julia
- Creating Packages in Julia: What do you need to know?
- Creating Packages in Julia: Using GeoRegions.jl as an Example
- Applications of GeoRegions.jl
- *(If we have time)* Creating Packages in Julia: Back to the Drawing Board!

7



Outline

- Understand and manage the package ecosystem in Julia
  - Both for package development
  - And for project management, using DrWatson.jl
- Creating Packages in Julia: What do you need to know?
  - Developing packages 101: PkgTemplates.jl, CI, and some Testing
  - How do you organize a package?
  - Best-practice/performance tips
- Creating Packages in Julia: Using GeoRegions.jl as an Example
  - The many iterations of GeoRegions.jl

8

Outline

- Applications of GeoRegions.jl
  - The basic functionality of GeoRegions.jl: Define, select and extract data
  - Using GeoRegions.jl in other Packages: What do you need to know?

9

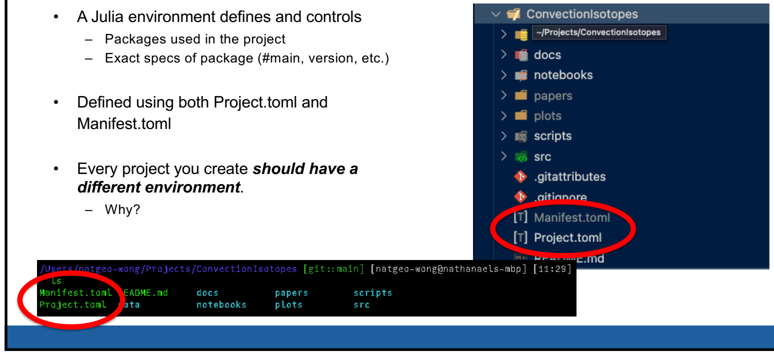
how do I shot web? (i.e., how to install and uninstall stuff)

THE JULIA PACKAGE ECOSYSTEM

10

What is an Environment?

- A Julia environment defines and controls
  - Packages used in the project
  - Exact specs of package (#main, version, etc.)
- Defined using both Project.toml and Manifest.toml
- Every project you create ***should have a different environment***.
  - Why?

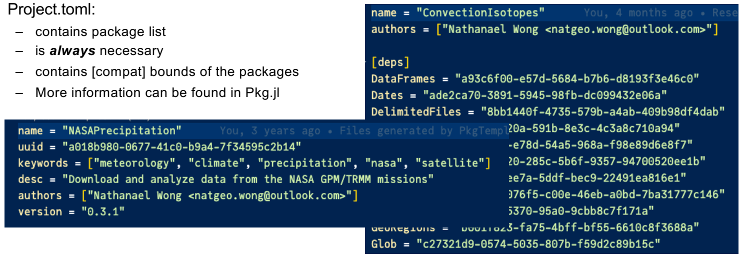


```
/Users/natgeo-wong/Projects/ConvectionIsotopes [git::main] [natgeo-wong@nathanaels-mbp] [11:29]
 ls
Manifest.toml README.md     docs          papers        scripts
Project.toml  data          notebooks     plots         src
```

11

What is an Environment?

- Project.toml:
  - contains package list
  - is ***always*** necessary
  - contains [compat] bounds of the packages
  - More information can be found in Pkg.jl



```
name = "ConvectionIsotopes"       You, 4 months ago • Rese
authors = ["Nathanael Wong <natgeo.wong@outlook.com>"]

[deps]
DataFrames = "a93c6f00-e57d-5684-b7b6-d8193f3e46c0"
Dates = "ade2ca70-3891-5945-98fb-dc099432e06a"
DelimitedFiles = "8bb1440f-4735-579b-a4ab-409b98df4dab"
                 20a-591b-8e3c-4c3a8c710a94"
                 -e78d-54a5-968a-f98e89d6e8f7"
                 20-285c-5b6f-9357-94700520ee1b"
                 ee7a-5ddf-bec9-22491ea816e1"
                 076f5-c00e-46eb-a0bd-7ba31777c146"
                 5370-95a0-9cbb8c7f171a"
GeoRegions = "b001f823-fa75-4bff-bf55-6610c8f3688a"
Glob = "c27321d9-0574-5035-807b-f59d2c89b15c"
```

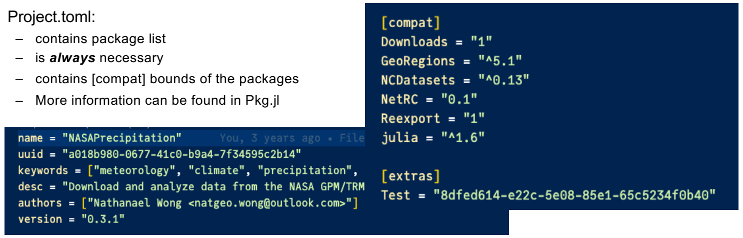
```
name = "NASAPrecipitation"       You, 3 years ago • Files generated by PkgTempl
uuid = "a018b980-0677-41c0-b9a4-7f34595c2b14"
keywords = ["meteorology", "climate", "precipitation", "nasa", "satellite"]
desc = "Download and analyze data from the NASA GPM/TRMM missions"
authors = ["Nathanael Wong <natgeo.wong@outlook.com>"]
version = "0.3.1"
```

12

HARVARD
Faculty of Arts and Sciences

What is an Environment?

- Project.toml:
  - contains package list
  - is ***always*** necessary
  - contains [compat] bounds of the packages
  - More information can be found in Pkg.jl



```
name = "NASAPrecipitation"        You, 3 years ago • File
uuid = "a018b980-0677-41c0-b9a4-7f34595c2b14"
keywords = ["meteorology", "climate", "precipitation",
desc = "Download and analyze data from the NASA GPM/TRM
authors = ["Nathanael Wong <natgeo.wong@outlook.com>"]
version = "0.3.1"
```

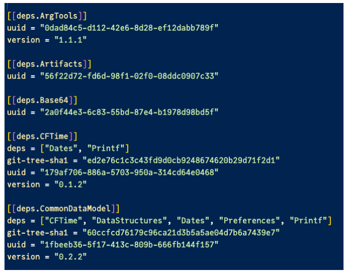
```
[compat]
Downloads = "1"
GeoRegions = "^5.1"
NCDatasets = "^0.13"
NetRC = "0.1"
Reexport = "1"
julia = "^1.6"

[extras]
Test = "8dfed614-e22c-5e08-85e1-65c5234f0b40"
```

13

HARVARD
Faculty of Arts and Sciences

What is an Environment?

- Project.toml:
  - contains package list
  - is ***always*** necessary
  - contains [compat] bounds of the packages
  - More information can be found in Pkg.jl
- Manifest.toml:
  - contains package ***and dependency*** information
  - is not necessary upon startup, will ***be created*** upon project/environment ***initialization***
  - is necessary ***an exact environment duplicate*** is required (e.g., for reproducibility purposes)



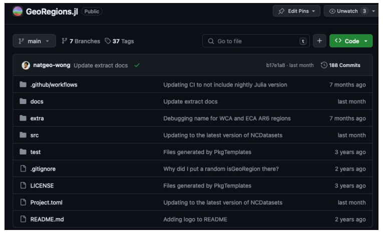
```
[[deps.ArgTools]]
uuid = "0dad84c5-d112-42e6-8d28-ef12dabb789f"
version = "1.1.1"

[[deps.Artifacts]]
uuid = "56f22d72-fd6d-98f1-02f0-08ddc0907c33"

[[deps.Base64]]
uuid = "2a0f44e3-6c83-55bd-87e4-b1978d98bd5f"

[[deps.CFTime]]
deps = ["Dates", "Printf"]
git-tree-sha1 = "ed2e76c1c3c43fd9d0cb9248674620b29d71f2d1"
uuid = "179af706-886a-5703-950a-314cd64e0468"
version = "0.1.2"

[[deps.CommonDataModel]]
deps = ["CFTime", "DataStructures", "Dates", "Preferences", "Printf"]
git-tree-sha1 = "60ccfcd76179c96ca21d3b5a5ae04d7b6a7439e7"
uuid = "1fbeeb36-5f17-413c-809b-666fb144f157"
version = "0.2.2"
```

14

HARVARD
Faculty of Arts and Sciences

What is an Environment?

- You must always have a Project.toml
- A Manifest.toml can be created on-the-spot through ***precompilation*** of the environment
  - GitHub has no Manifest.toml



natgeo-wong Update extract docs ✓ b17e1a8 · last month 188 Commits

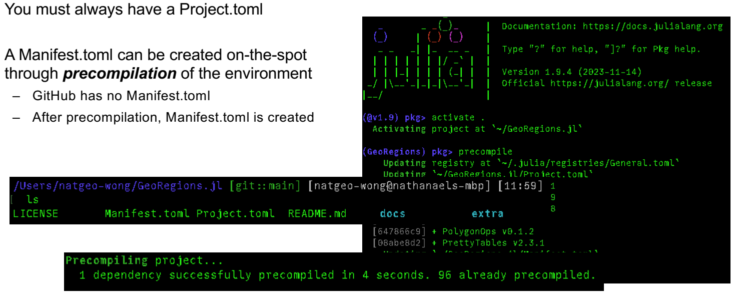
.github/workflows	Updating CI to not include nightly Julia version	7 months ago
docs	Update extract docs	last month
extra	Debugging name for WCA and ECA AR6 regions	7 months ago
src	Updating to the latest version of NCDatasets	last month
test	Files generated by PkgTemplates	3 years ago
.gitignore	Why did I put a random isGeoRegion there?	2 years ago
LICENSE	Files generated by PkgTemplates	3 years ago
Project.toml	Updating to the latest version of NCDatasets	last month
README.md	Adding logo to README	2 years ago

15

HARVARD
Faculty of Arts and Sciences

What is an Environment?

- You must always have a Project.toml
- A Manifest.toml can be created on-the-spot through ***precompilation*** of the environment
  - GitHub has no Manifest.toml
  - After precompilation, Manifest.toml is created



```
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.9.4 (2023-11-14)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

(@v1.9) pkg> activate .
  Activating project at `~/GeoRegions.jl`

(GeoRegions) pkg> precompile
    Updating registry at `~/.julia/registries/General.toml`
    Updating `~/GeoRegions.jl/Project.toml`
```

```
/Users/natgeo-wong/GeoRegions.jl [git::main] [natgeo-wong@nathanaels-mbp] [11:59] 198
[  ls
LICENSE       Manifest.toml Project.toml  README.md     docs          extra
```

```
  [647866c9] + PolygonOps v0.1.2
  [08abe8d2] + PrettyTables v2.3.1
```

```
Precompiling project...
  1 dependency successfully precompiled in 4 seconds. 96 already precompiled.
```

16

Activity: Creating an Environment

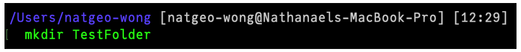
- Now you try!

17

Activity: Creating an Environment

- Now you try!

1. Create a folder



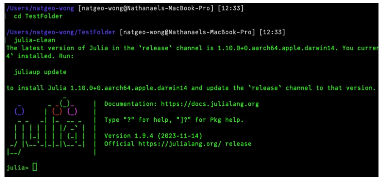
```
/Users/natgeo-wong [natgeo-wong@Nathanaels-MacBook-Pro] [12:29]
[  mkdir TestFolder
```

18

Activity: Creating an Environment

- Now you try!

1. Create a folder
2. Run Julia in this folder (*command: julia*)



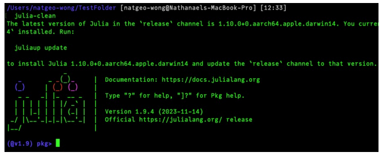
```
/Users/natgeo-wong [natgeo-wong@Nathanaels-MacBook-Pro] [12:33]
[  cd TestFolder

/Users/natgeo-wong/TestFolder [natgeo-wong@Nathanaels-MacBook-Pro] [12:33]
[  julia-clean
The latest version of Julia in the `release` channel is 1.10.0+0.aarch64.apple.darwin14. You curren
4` installed. Run:

  juliaup update

to install Julia 1.10.0+0.aarch64.apple.darwin14 and update the `release` channel to that version.
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.9.4 (2023-11-14)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> 
```

19

Activity: Creating an Environment

- Now you try!

1. Create a folder
2. Run Julia in this folder (*command: julia*)
3. Enter the package console (*command:]*)



```
/Users/natgeo-wong/TestFolder [natgeo-wong@Nathanaels-MacBook-Pro] [12:33]
[  julia-clean
The latest version of Julia in the `release` channel is 1.10.0+0.aarch64.apple.darwin14. You curre
4` installed. Run:

  juliaup update

to install Julia 1.10.0+0.aarch64.apple.darwin14 and update the `release` channel to that version.
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.9.4 (2023-11-14)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

(@v1.9) pkg> 
```

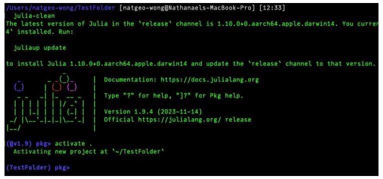
20

HARVARD
Faculty of Arts and Sciences

Activity: Creating an Environment

- Now you try!

1. Create a folder
2. Run Julia in this folder (command: *julia*)
3. Enter the package console (command: *]*)
4. Activate the current environment (command: *activate .*)



```
/Users/natgeo-wong/TestFolder [natgeo-wong@Nathanaels-MacBook-Pro] [12:33]
  julia-clean
The latest version of Julia in the `release` channel is 1.10.0+0.aarch64.apple.darwin14. You curren
4` installed. Run:

  juliaup update

to install Julia 1.10.0+0.aarch64.apple.darwin14 and update the `release` channel to that version.
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.9.4 (2023-11-14)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

(@v1.9) pkg> activate .
  Activating new project at `~/TestFolder`

(TestFolder) pkg>
```

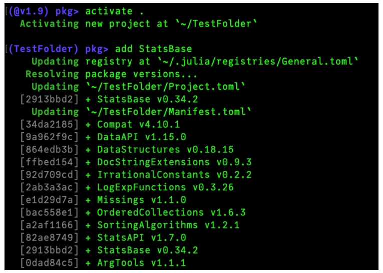
21

HARVARD
Faculty of Arts and Sciences

Activity: Creating an Environment

- Now you try!

1. Create a folder
2. Run Julia in this folder (command: *julia*)
3. Enter the package console (command: *]*)
4. Activate the current environment (command: *activate .*)
5. Add a random package (e.g., StatsBase)



```
(@v1.9) pkg> activate .
  Activating new project at `~/TestFolder`

(TestFolder) pkg> add StatsBase
    Updating registry at `~/.julia/registries/General.toml`
   Resolving package versions...
    Updating `~/TestFolder/Project.toml`
  [2913bbd2] + StatsBase v0.34.2
    Updating `~/TestFolder/Manifest.toml`
  [34da2185] + Compat v4.10.1
  [9a962f9c] + DataAPI v1.15.0
  [864edb3b] + DataStructures v0.18.15
  [ffbed154] + DocStringExtensions v0.9.3
  [92d709cd] + IrrationalConstants v0.2.2
  [2ab3a3ac] + LogExpFunctions v0.3.26
  [e1d29d7a] + Missings v1.1.0
  [bac558e1] + OrderedCollections v1.6.3
  [a2af1166] + SortingAlgorithms v1.2.1
  [82ae8749] + StatsAPI v1.7.0
  [2913bbd2] + StatsBase v0.34.2
  [0dad84c5] + ArgTools v1.1.1
```

22

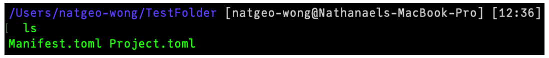
HARVARD
Faculty of Arts and Sciences

Activity: Creating an Environment

- Now you try!

1. Create a folder
2. Run Julia in this folder (command: *julia*)
3. Enter the package console (command: *]*)
4. Activate the current environment (command: *activate .*)
5. Add a random package (e.g., StatsBase)

- Your previously empty folder should now have ***both*** Project.toml and Manifest.toml



```
/Users/natgeo-wong/TestFolder [natgeo-wong@Nathanaels-MacBook-Pro] [12:36]
  ls
Manifest.toml Project.toml
```

23

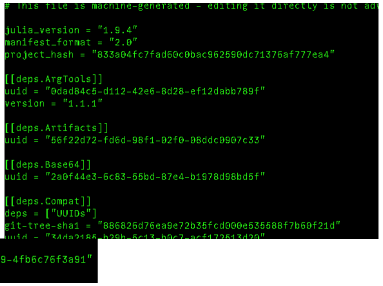
HARVARD
Faculty of Arts and Sciences

Activity: Creating an Environment

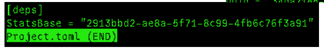
- Now you try!

1. Create a folder
2. Run Julia in this folder (command: *julia*)
3. Enter the package console (command: *]*)
4. Activate the current environment (command: *activate .*)
5. Add a random package (e.g., StatsBase)

- Your previously empty folder should now have ***both*** Project.toml and Manifest.toml



```
# This file is machine-generated - editing it directly is not adv

julia_version = "1.9.4"
manifest_format = "2.0"
project_hash = "833a04fc7fad60c0bac962590dc71376af777ea4"

[[deps.ArgTools]]
uuid = "0dad84c5-d112-42e6-8d28-ef12dabb789f"
version = "1.1.1"

[[deps.Artifacts]]
uuid = "56f22d72-fd6d-98f1-02f0-08ddc0907c33"

[[deps.Base64]]
uuid = "2a0f44e3-6c83-55bd-87e4-b1978d98bd5f"

[[deps.Compat]]
deps = ["UUIDs"]
git-tree-sha1 = "886826d76ea9e72b35fcd000e535588f7b60f21d"
uuid = "34da2185-b29b-5c13-b0c7-acf172513d20"
```



```
[deps]
StatsBase = "2913bbd2-ae8a-5f71-8c99-4fb6c76f3a91"
Project.toml (END)
```

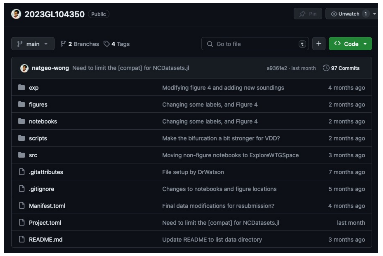
24

HARVARD
Faculty of Arts and Sciences

Activity: Precompiling an Environment

- Now you try!

1. Go to https://github.com/natgeo-wong/2023GL104350 (this is my recent paper)
2. Clone the repository



2023GL104350 Public

natgeo-wong Need to limit the [compat] for NCDatasets.jl a9361e2 · last month 97 Commits

exp — Modifying figure 4 and adding new soundings — 4 months ago
figures — Changing some labels, and Figure 4 — 2 months ago
notebooks — Changing some labels, and Figure 4 — 2 months ago
scripts — Make the bifurcation a bit stronger for VDD? — 2 months ago
src — Moving non-figure notebooks to ExploreWTGSpace — 3 months ago
.gitattributes — File setup by DrWatson — 7 months ago
.gitignore — Changes to notebooks and figure locations — 5 months ago
Manifest.toml — Final data modifications for resubmission? — 4 months ago
Project.toml — Need to limit the [compat] for NCDatasets.jl — last month
README.md — Update README to list data directory — 3 months ago

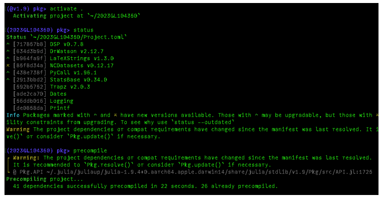
25

HARVARD
Faculty of Arts and Sciences

Activity: Precompiling an Environment

- Now you try!

1. Go to https://github.com/natgeo-wong/2023GL104350 (this is my recent paper)
2. Clone the repository
3. Open Julia and activate the environment



```
(@v1.9) pkg> activate .
  Activating project at `~/2023GL104350`

(2023GL104350) pkg> status
Status `~/2023GL104350/Project.toml`
^ [717857b8] DSP v0.7.8
^ [634d3b9d] DrWatson v2.12.7
^ [b964fa9f] LaTeXStrings v1.3.0
⌅ [85f8d34a] NCDatasets v0.12.17
^ [438e738f] PyCall v1.96.1
^ [2913bbd2] StatsBase v0.34.0
  [592b5752] Trapz v2.0.3
  [ade2ca70] Dates
  [56ddb016] Logging
  [de0858da] Printf
Info Packages marked with ^ and ⌅ have new versions available. Those with ^ may be upgradable, but those with ⌅
ility constraints from upgrading. To see why use `status --outdated`
Warning The project dependencies or compat requirements have changed since the manifest was last resolved. It i
ve()` or consider `Pkg.update()` if necessary.

(2023GL104350) pkg> precompile
┌ Warning: The project dependencies or compat requirements have changed since the manifest was last resolved.
│ It is recommended to `Pkg.resolve()` or consider `Pkg.update()` if necessary.
└ @ Pkg.API ~/.julia/juliaup/julia-1.9.4+0.aarch64.apple.darwin14/share/julia/stdlib/v1.9/Pkg/src/API.jl:1725
Precompiling project...
  41 dependencies successfully precompiled in 22 seconds. 26 already precompiled.
```

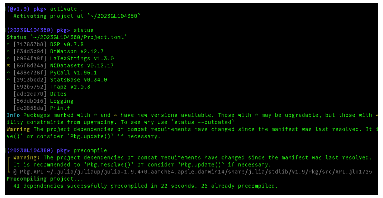
26

HARVARD
Faculty of Arts and Sciences

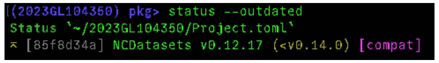
Activity: Precompiling an Environment

- Now you try!

1. Go to https://github.com/natgeo-wong/2023GL104350 (this is my recent paper)
2. Clone the repository
3. Open Julia and activate the environment
4. Precompile the environment



```
(@v1.9) pkg> activate .
  Activating project at `~/2023GL104350`

(2023GL104350) pkg> status
Status `~/2023GL104350/Project.toml`
^ [717857b8] DSP v0.7.8
^ [634d3b9d] DrWatson v2.12.7
^ [b964fa9f] LaTeXStrings v1.3.0
⌅ [85f8d34a] NCDatasets v0.12.17
^ [438e738f] PyCall v1.96.1
^ [2913bbd2] StatsBase v0.34.0
  [592b5752] Trapz v2.0.3
  [ade2ca70] Dates
  [56ddb016] Logging
  [de0858da] Printf
Info Packages marked with ^ and ⌅ have new versions available. Those with ^ may be upgradable, but those with ⌅
ility constraints from upgrading. To see why use `status --outdated`
Warning The project dependencies or compat requirements have changed since the manifest was last resolved. It i
ve()` or consider `Pkg.update()` if necessary.

(2023GL104350) pkg> precompile
┌ Warning: The project dependencies or compat requirements have changed since the manifest was last resolved.
│ It is recommended to `Pkg.resolve()` or consider `Pkg.update()` if necessary.
└ @ Pkg.API ~/.julia/juliaup/julia-1.9.4+0.aarch64.apple.darwin14/share/julia/stdlib/v1.9/Pkg/src/API.jl:1725
Precompiling project...
  41 dependencies successfully precompiled in 22 seconds. 26 already precompiled.
```



```
(2023GL104350) pkg> status --outdated
Status `~/2023GL104350/Project.toml`
⌅ [85f8d34a] NCDatasets v0.12.17 (<v0.14.0) [compat]
```

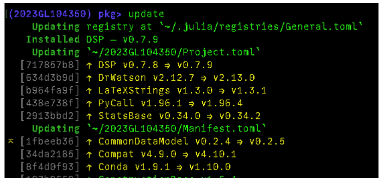
27

HARVARD
Faculty of Arts and Sciences

Activity: Precompiling an Environment

- Now you try!

1. Go to https://github.com/natgeo-wong/2023GL104350 (this is my recent paper)
2. Clone the repository
3. Open Julia and activate the environment
4. Precompile the environment
5. Update the environment



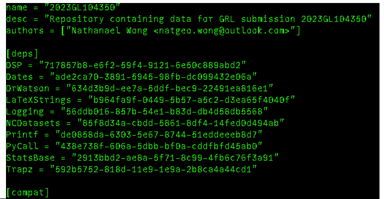
```
(2023GL104350) pkg> update
    Updating registry at `~/.julia/registries/General.toml`
   Installed DSP ─ v0.7.9
    Updating `~/2023GL104350/Project.toml`
  [717857b8] ↑ DSP v0.7.8 ⇒ v0.7.9
  [634d3b9d] ↑ DrWatson v2.12.7 ⇒ v2.13.0
  [b964fa9f] ↑ LaTeXStrings v1.3.0 ⇒ v1.3.1
  [438e738f] ↑ PyCall v1.96.1 ⇒ v1.96.4
  [2913bbd2] ↑ StatsBase v0.34.0 ⇒ v0.34.2
    Updating `~/2023GL104350/Manifest.toml`
⌅ [1fbeeb36] ↑ CommonDataModel v0.2.4 ⇒ v0.2.5
  [34da2185] ↑ Compat v4.9.0 ⇒ v4.10.1
  [8f4d0f93] ↑ Conda v1.9.1 ⇒ v1.10.0
```

28

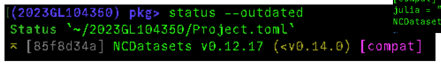
Activity: Precompiling an Environment

- Now you try!

1. Go to https://github.com/natgeo-wong/2023GL104350 (this is my recent paper)
2. Clone the repository
3. Open Julia and activate the environment
4. Precompile the environment
5. Update the environment



```
name = "2023GL104350"
desc = "Repository containing data for GRL submission 2023GL104350"
authors = ["Nathanael Wong <natgeo.wong@outlook.com>"]

[deps]
DSP = "717857b8-e6f2-59f4-9121-6e50c889abd2"
Dates = "ade2ca70-3891-5945-98fb-dc099432e06a"
DrWatson = "634d3b9d-ee7a-5ddf-bec9-22491ea816e1"
LaTeXStrings = "b964fa9f-0449-5b57-a5c2-d3ea65f4040f"
Logging = "56ddb016-857b-54e1-b83d-db4d58db5568"
NCDatasets = "85f8d34a-cbdd-5861-8df4-14fed0d494ab"
Printf = "de0858da-6303-5e67-8744-51eddeeeb8d7"
PyCall = "438e738f-606a-5dbb-bf0a-cddfbfd45ab0"
StatsBase = "2913bbd2-ae8a-5f71-8c99-4fb6c76f3a91"
Trapz = "592b5752-818d-11e9-1e9a-2b8ca4a44cd1"

[compat]
julia = "1"
NCDatasets = "0.12.7"
```



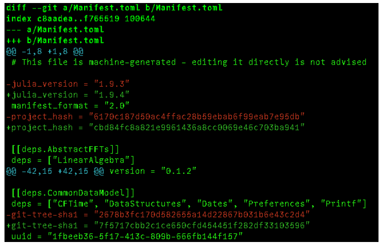
```
(2023GL104350) pkg> status --outdated
Status `~/2023GL104350/Project.toml`
⌅ [85f8d34a] NCDatasets v0.12.17 (<v0.14.0) [compat]
```

29

Activity: Precompiling an Environment

- Now you try!

1. Go to https://github.com/natgeo-wong/2023GL104350 (this is my recent paper)
2. Clone the repository
3. Open Julia and activate the environment
4. Precompile the environment
5. Update the environment
6. Compare the difference in Manifest using git



```
diff --git a/Manifest.toml b/Manifest.toml
index c8aadea..f765519 100644
--- a/Manifest.toml
+++ b/Manifest.toml
@@ -1,8 +1,8 @@
 # This file is machine-generated - editing it directly is not advised

-julia_version = "1.9.3"
+julia_version = "1.9.4"
 manifest_format = "2.0"
-project_hash = "6170c187d50ac4ffac28b59ebab6f99eab7e95db"
+project_hash = "cbd84fc8a821e9961436a8cc0069e46c703ba941"

 [[deps.AbstractFFTs]]
 deps = ["LinearAlgebra"]
@@ -42,15 +42,15 @@ version = "0.1.2"

 [[deps.CommonDataModel]]
 deps = ["CFTime", "DataStructures", "Dates", "Preferences", "Printf"]
-git-tree-sha1 = "2678b3fc170d582655a14d22867b031b6e43c2d4"
+git-tree-sha1 = "7f5717cbb2c1ce650cfd454451f282df33103596"
 uuid = "1fbeeb36-5f17-413c-809b-666fb144f157"
```

30

Managing Environments in Package Creation

- PkgTemplates.jl (https://github.com/JuliaCI/PkgTemplates.jl)
  - Can be used to easily create new Julia packages (we'll get to this later)
  - Will setup a Package Environment by creating both Project.toml and Manifest.toml
  - By default, setups Git to track only the Project.toml, not the Manifest.toml

31

Managing Environments in Package Creation

- PkgTemplates.jl (https://github.com/JuliaCI/PkgTemplates.jl)
  - Can be used to easily create new Julia packages (we'll get to this later)
  - Will setup a Package Environment by creating both Project.toml and Manifest.toml
  - By default, setups Git to track only the Project.toml, not the Manifest.toml
- You generally don't want to track the Manifest.toml of a Package
  - Can anyone tell me why?

32

HARVARD
Faculty of Arts and Sciences

Managing Environments in Package Creation

- PkgTemplates.jl (https://github.com/JuliaCI/PkgTemplates.jl)
  - Can be used to easily create new Julia packages (we'll get to this later)
  - Will setup a Package Environment by creating both Project.toml and Manifest.toml
  - By default, setups Git to track only the Project.toml, not the Manifest.toml

- You generally don't want to track the Manifest.toml of a Package
  - Can anyone tell me why?
  - A:
    - [compat] requirements are ***already set in Project.toml***
    - You want to allow for flexibility in ***dependencies***, as these dependencies are likely also used by other packages in the environment
    - Manifest.toml will change very rapidly and will ***vary from device to device***, not logical to track changes

33

HARVARD
Faculty of Arts and Sciences

Managing Environments for Personal Projects

- DrWatson.jl (https://github.com/JuliaDynamics/DrWatson.jl)
  - More for creating your own self-contained projects (more on this later!)
  - Will setup a Project Environment by creating both Project.toml and Manifest.toml
  - By default, tracks ***both*** the Project.toml and Manifest.toml to ***ensure project reproducibility***

34

HARVARD
Faculty of Arts and Sciences

Managing Environments for Personal Projects

- DrWatson.jl (https://github.com/JuliaDynamics/DrWatson.jl)
  - More for creating your own self-contained projects (more on this later!)
  - Will setup a Project Environment by creating both Project.toml and Manifest.toml
  - By default, tracks ***both*** the Project.toml and Manifest.toml to ***ensure project reproducibility***

- The scope of this lecture mostly focuses around Package creation, not project creation, but I though it would be good to distinguish the two

- Can anyone give me an example of when it is good to track/commit the Manifest.toml?

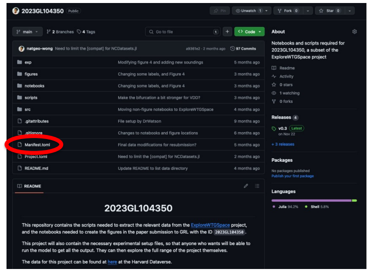
35

HARVARD
Faculty of Arts and Sciences

Managing Environments for Personal Projects

- Paper reproducibility!
- Example given just now (this is an actual repository for my paper)

- This project folder was created using DrWatson.jl



natgeo-wong Need to limit the [compat] for NCDatasets.jl a9361e2 · 2 months ago 97 Commits

exp	Modifying figure 4 and adding new soundings	5 months ago
figures	Changing some labels, and Figure 4	3 months ago
notebooks	Changing some labels, and Figure 4	3 months ago
scripts	Make the bifurcation a bit stronger for VDD?	3 months ago
src	Moving non-figure notebooks to ExploreWTGSpace	4 months ago
.gitattributes	File setup by DrWatson	9 months ago
.gitignore	Changes to notebooks and figure locations	6 months ago
Manifest.toml	Final data modifications for resubmission?	5 months ago
Project.toml	Need to limit the [compat] for NCDatasets.jl	2 months ago
README.md	Update README to list data directory	4 months ago

About

Notebooks and scripts required for 2023GL104350, a subset of the ExploreWTGSpace project

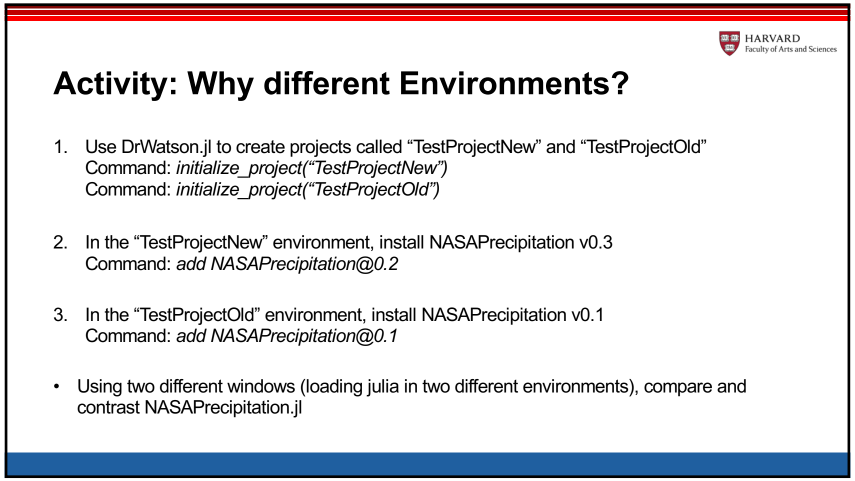
2023GL104350

This repository contains the scripts needed to extract the relevant data from the ExploreWTGSpace project, and the notebooks needed to create the figures in the paper submission to GRL with the ID `2023GL104350`.

This project will also contain the necessary experimental setup files, so that anyone who wants will be able to run the model to get all the output. They can then explore the full range of the project themselves.

The data for this project can be found at here at the Harvard Dataverse.

36



Activity: Why different Environments?

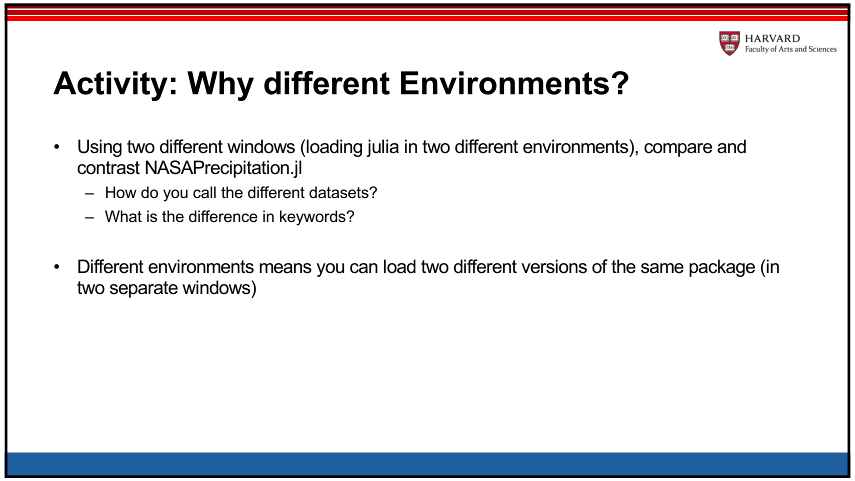
1. Use DrWatson.jl to create projects called “TestProjectNew” and “TestProjectOld”
   Command: *initialize_project(“TestProjectNew”)*
   Command: *initialize_project(“TestProjectOld”)*

2. In the “TestProjectNew” environment, install NASAPrecipitation v0.3
   Command: *add NASAPrecipitation@0.2*

3. In the “TestProjectOld” environment, install NASAPrecipitation v0.1
   Command: *add NASAPrecipitation@0.1*

- Using two different windows (loading julia in two different environments), compare and contrast NASAPrecipitation.jl

37

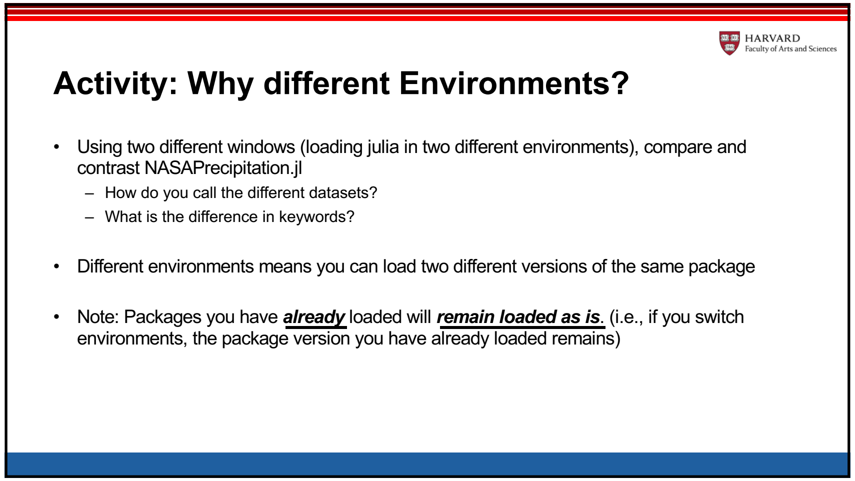


Activity: Why different Environments?

- Using two different windows (loading julia in two different environments), compare and contrast NASAPrecipitation.jl
  - How do you call the different datasets?
  - What is the difference in keywords?

- Different environments means you can load two different versions of the same package (in two separate windows)

38



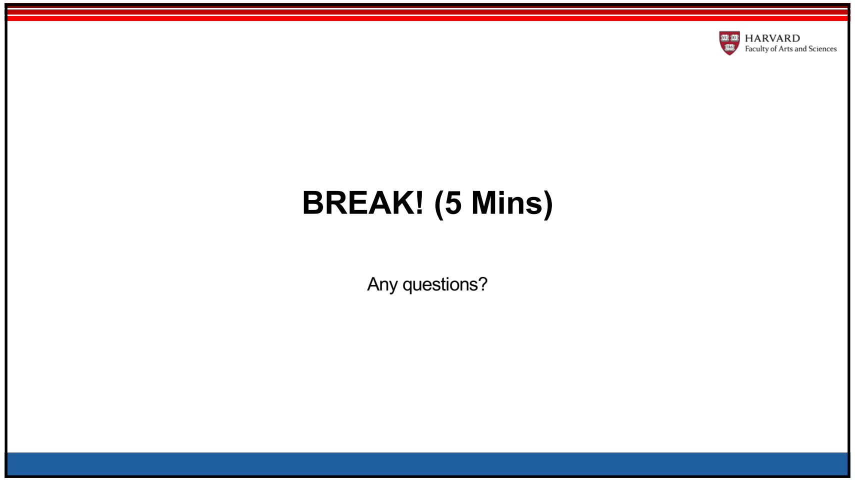
Activity: Why different Environments?

- Using two different windows (loading julia in two different environments), compare and contrast NASAPrecipitation.jl
  - How do you call the different datasets?
  - What is the difference in keywords?

- Different environments means you can load two different versions of the same package

- Note: Packages you have ***already*** loaded will ***remain loaded as is.*** (i.e., if you switch environments, the package version you have already loaded remains)

39



BREAK! (5 Mins)

Any questions?

40

HARVARD
Faculty of Arts and Sciences

An introduction to PkgTemplates.jl and other things you need to know

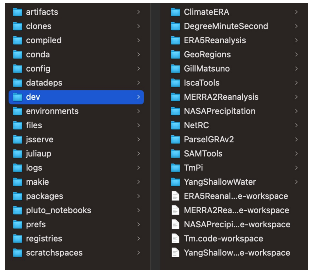
CREATING PACKAGES IN JULIA: AN INTRODUCTION

41

HARVARD
Faculty of Arts and Sciences

Package Creation with PkgTemplates.jl

- I usually use PkgTemplates.jl to create and develop new Julia packages
- PkgTemplates.jl will automatically create new packages inside ~/.julia/dev
  - This is the ***dev*** folder, where all packages you develop are stored
  - Note: You can also ***develop*** preexisting packages, just do: *]dev PkgName*



artifacts
clones
compiled
conda
config
datadeps
dev
environments
files
jsserve
juliaup
logs
makie
packages
pluto_notebooks
prefs
registries
scratchspaces

ClimateERA
DegreeMinuteSecond
ERA5Reanalysis
GeoRegions
GillMatsuno
IscaTools
MERRA2Reanalysis
NASAPrecipitation
NetRC
ParselGRAv2
SAMTools
TmPi
YangShallowWater
ERA5Reanal...e-workspace
MERRA2Rea...e-workspace
NASAPrecipi...e-workspace
Tm.code-workspace
YangShallow...e-workspace

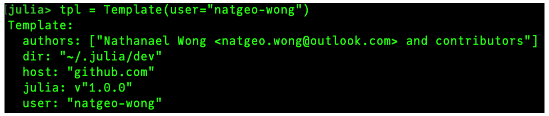
42

HARVARD
Faculty of Arts and Sciences

Activity: Your First Package!

- Your turn! Let's try developing your first package!

1. *using PkgTemplates*
2. *tpl = Template()*
3. *tpl("MyFirstPackage")*



```
julia> tpl = Template(user="natgeo-wong")
Template:
  authors: ["Nathanael Wong <natgeo.wong@outlook.com> and contributors"]
  dir: "~/.julia/dev"
  host: "github.com"
  julia: v"1.0.0"
  user: "natgeo-wong"
```

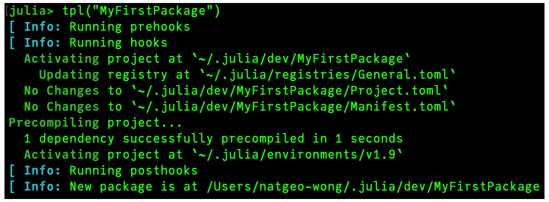
43

HARVARD
Faculty of Arts and Sciences

Activity: Your First Package!

- Your turn! Let's try developing your first package!

1. *using PkgTemplates*
2. *tpl = Template()*
3. *tpl("MyFirstPackage")*



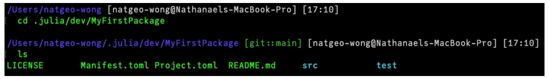
```
julia> tpl("MyFirstPackage")
[ Info: Running prehooks
[ Info: Running hooks
  Activating project at `~/.julia/dev/MyFirstPackage`
    Updating registry at `~/.julia/registries/General.toml`
  No Changes to `~/.julia/dev/MyFirstPackage/Project.toml`
  No Changes to `~/.julia/dev/MyFirstPackage/Manifest.toml`
Precompiling project...
  1 dependency successfully precompiled in 1 seconds
  Activating project at `~/.julia/environments/v1.9`
[ Info: Running posthooks
[ Info: New package is at /Users/natgeo-wong/.julia/dev/MyFirstPackage
```

44

Activity: Your First Package!

- Your turn! Let's try developing your first package!

1. *using PkgTemplates*
2. *tpl = Template()*
3. *tpl("MyFirstPackage")*



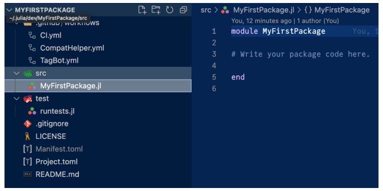
```
/Users/natgeo-wong [natgeo-wong@Nathanaels-MacBook-Pro] [17:10]
  cd .julia/dev/MyFirstPackage

/Users/natgeo-wong/.julia/dev/MyFirstPackage [git::main] [natgeo-wong@Nathanaels-MacBook-Pro] [17:10]
  ls
LICENSE       Manifest.toml Project.toml  README.md     src           test
```

45

Activity: Your First Package!

- Your turn! Let's try developing your first package!

1. *using PkgTemplates*
2. *tpl = Template()*
3. *tpl("MyFirstPackage")*



```
MYFIRSTPACKAGE
~/.julia/dev/MyFirstPackage/src
  .github / workflows
    CI.yml
    CompatHelper.yml
    TagBot.yml
  src
    MyFirstPackage.jl
  test
    runtests.jl
  .gitignore
  LICENSE
  Manifest.toml
  Project.toml
  README.md
```

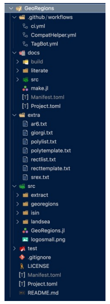
src > MyFirstPackage.jl > { } MyFirstPackage

You, 12 minutes ago | 1 author (You)

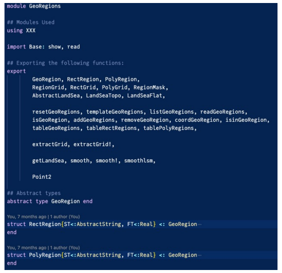
```julia
module MyFirstPackage

# Write your package code here.

end
```

46

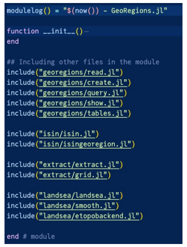
What does a fully-formed Package look like?



```
GeoRegions
  .github / workflows
    ci.yml
    CompatHelper.yml
    TagBot.yml
  docs
    build
    literate
    src
    make.jl
    Manifest.toml
    Project.toml
  extra
    ar6.txt
    giorgi.txt
    polylist.txt
    polytemplate.txt
    rectlist.txt
    recttemplate.txt
    srex.txt
  src
    extract
    georegions
    isin
    landsea
    GeoRegions.jl
    logosmall.png
  test
  .gitignore
  LICENSE
  Manifest.toml
  Project.toml
  README.md
```



```julia
module GeoRegions

## Modules Used
using XXX

import Base: show, read

## Exporting the following functions:
export
        GeoRegion, RectRegion, PolyRegion,
        RegionGrid, RectGrid, PolyGrid, RegionMask,
        AbstractLandSea, LandSeaTopo, LandSeaFlat,

        resetGeoRegions, templateGeoRegions, listGeoRegions, readGeoRegions,
        isGeoRegion, addGeoRegions, removeGeoRegion, coordGeoRegion, isinGeoRegion,
        tableGeoRegions, tableRectRegions, tablePolyRegions,

        extractGrid, extractGrid!,

        getLandSea, smooth, smooth!, smoothlsm,

        Point2

## Abstract types
abstract type GeoRegion end

You, 7 months ago | 1 author (You)
struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion…
end

You, 7 months ago | 1 author (You)
struct PolyRegion{ST<:AbstractString, FT<:Real} <: GeoRegion…
end
```



```julia
modulelog() = "$(now()) - GeoRegions.jl"

function __init__()…
end

## Including other files in the module
include("georegions/read.jl")
include("georegions/create.jl")
include("georegions/query.jl")
include("georegions/show.jl")
include("georegions/tables.jl")

include("isin/isin.jl")
include("isin/isingeoregion.jl")

include("extract/extract.jl")
include("extract/grid.jl")

include("landsea/landsea.jl")
include("landsea/smooth.jl")
include("landsea/etopobackend.jl")

end # module
```

47

Designing a Package

- What are you trying to accomplish when you are designing a package?
- Let's go around the room. What would you create a package for in Earth Observation?
  - Streamline workflows (e.g. downloading datasets)
  - Documentation of personal projects

48

Designing a Package

- What are you trying to accomplish when you are designing a package?
- Let's go around the room. What would you create a package for in Earth Observation?
  - Data retrieval (from online servers, data repositories, etc.)
  - Data analysis (timeseries analysis, temporal/spatial smoothing, daily/monthly means)
  - Plotting and visualization of data

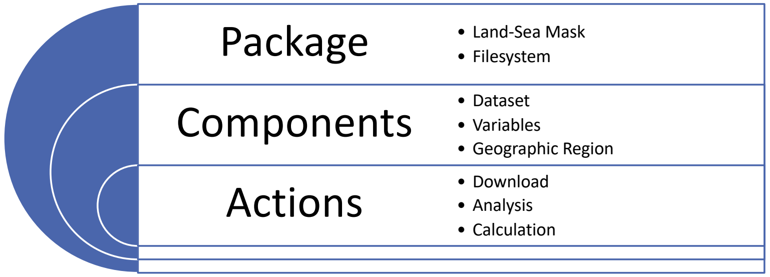
49

Designing a Package

- What are you trying to accomplish when you are designing a package?
- Let's go around the room. What would you create a package for in Earth Observation?
  - Data retrieval (from online servers, data repositories, etc.)
  - Data analysis (timeseries analysis, temporal/spatial smoothing, daily/monthly means)
  - Plotting and visualization of data
- What are these? They are ***actions*** that you would perform on a ***dataset***
  - A package must ***first*** focus on ***defining*** these datasets and their ***components***
  - How would you organize a package?

50

Designing a Package



Package
- Land-Sea Mask
- Filesystem

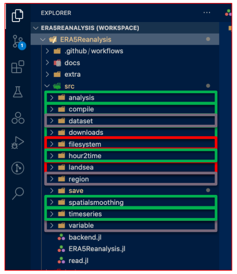
Components
- Dataset
- Variables
- Geographic Region

Actions
- Download
- Analysis
- Calculation

51

Designing a Package

- See filesystem structure on the right here
  - Red = package filesystem
  - Purple = package components
  - Green = actionables / analysis
- You also have miscellaneous backend items
  - Date2String functions
  - Error checks
  - Nan-means
  - ***Real2Int*** functions



```
ERA5REANALYSIS (WORKSPACE)
  ERA5Reanalysis
    .github / workflows
    docs
    extra
    src
      analysis
      compile
      dataset
      downloads
      filesystem
      hour2time
      landsea
      region
      save
      spatialsmoothing
      timeseries
      variable
      backend.jl
      ERA5Reanalysis.jl
      read.jl
```

52

HARVARD Faculty of Arts and Sciences

Designing a Package

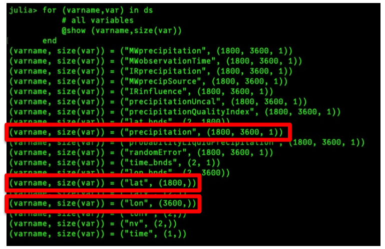
- What does this mean?
  - You need to be able to know what data is available/provided
  - What is the package going to do for ***you***?
  - You may not want to retrieve everything (e.g., is all the information relevant to you?)

53

HARVARD Faculty of Arts and Sciences

Designing a Package

- What does this mean?
  - You need to be able to know what data is available/provided
  - What is the package going to do for ***you***?
  - You may not want to retrieve everything (e.g., is all the information relevant to you?)
- E.g., the Global Precipitation Mission provides a lot of extra data
  - e.g., IR Precipitation (Infrared Radar)
  - Do you need these data? Or just the total precipitation values?



```
julia> for (varname,var) in ds
           # all variables
           @show (varname,size(var))
       end
(varname, size(var)) = ("MWprecipitation", (1800, 3600, 1))
(varname, size(var)) = ("MWobservationTime", (1800, 3600, 1))
(varname, size(var)) = ("IRprecipitation", (1800, 3600, 1))
(varname, size(var)) = ("MWprecipSource", (1800, 3600, 1))
(varname, size(var)) = ("IRinfluence", (1800, 3600, 1))
(varname, size(var)) = ("precipitationUncal", (1800, 3600, 1))
(varname, size(var)) = ("precipitationQualityIndex", (1800, 3600, 1))
(varname, size(var)) = ("lat_bnds", (2, 1800))
(varname, size(var)) = ("precipitation", (1800, 3600, 1))
(varname, size(var)) = ("probabilityLiquidPrecipitation", (1800, 3600, 1))
(varname, size(var)) = ("randomError", (1800, 3600, 1))
(varname, size(var)) = ("time_bnds", (2, 1))
(varname, size(var)) = ("lon_bnds", (2, 3600))
(varname, size(var)) = ("lat", (1800,))
(varname, size(var)) = ("latv", (2,))
(varname, size(var)) = ("lon", (3600,))
(varname, size(var)) = ("lonv", (2,))
(varname, size(var)) = ("nv", (2,))
(varname, size(var)) = ("time", (1,))
```

54

HARVARD Faculty of Arts and Sciences

Multiple Datasets in a Package

- Packages may handle multiple datasets

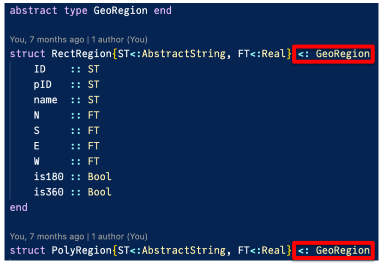
GPM_3IMERGDE.06/	2024-01-02T10:10:56GMT	-
GPM_3IMERGDF.06/	2021-06-01T14:11:24GMT	-
GPM_3IMERGDF.07/	2023-08-26T13:01:05GMT	-
GPM_3IMERGDL.06/	2024-01-02T14:46:48GMT	-
GPM_3IMERGHH.06/	2023-04-19T19:38:27GMT	-
GPM_3IMERGHH.07/	2023-08-25T22:15:24GMT	-
GPM_3IMERGHHE.06/	2024-01-01T08:06:07GMT	-
GPM_3IMERGHHL.06/	2024-01-01T19:00:30GMT	-
GPM_3IMERGM.06/	2021-05-06T16:44:26GMT	-
GPM_3IMERGM.07/	2023-08-25T22:16:34GMT	-

55

HARVARD Faculty of Arts and Sciences

Multiple Datasets in a Package

- Packages may handle multiple datasets
  - Each of these datasets may have different properties
  - Each of these datasets may have different performable actions
- How do we handle multiple datasets in a single package?
- ***Types*** + multiple dispatch ***methods***!

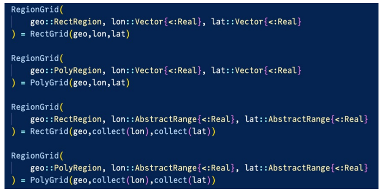


```
abstract type GeoRegion end

You, 7 months ago | 1 author (You)
struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
    ID    :: ST
    pID   :: ST
    name  :: ST
    N     :: FT
    S     :: FT
    E     :: FT
    W     :: FT
    is180 :: Bool
    is360 :: Bool
end

You, 7 months ago | 1 author (You)
struct PolyRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
```

56

Multiple Datasets in a Package

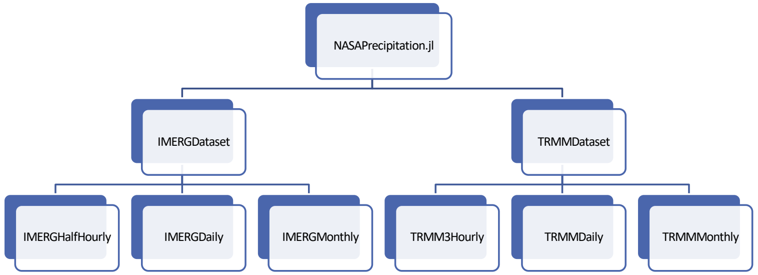
- Packages may handle multiple datasets
  - Each of these datasets may have different properties
  - Each of these datasets may have different performable actions
- How do we handle multiple datasets in a single package?
- ***Types*** + multiple dispatch ***methods***!



```
RegionGrid(
    geo::RectRegion, lon::Vector{<:Real}, lat::Vector{<:Real}
) = RectGrid(geo,lon,lat)

RegionGrid(
    geo::PolyRegion, lon::Vector{<:Real}, lat::Vector{<:Real}
) = PolyGrid(geo,lon,lat)

RegionGrid(
    geo::RectRegion, lon::AbstractRange{<:Real}, lat::AbstractRange{<:Real}
) = RectGrid(geo,collect(lon),collect(lat))

RegionGrid(
    geo::PolyRegion, lon::AbstractRange{<:Real}, lat::AbstractRange{<:Real}
) = PolyGrid(geo,collect(lon),collect(lat))
```

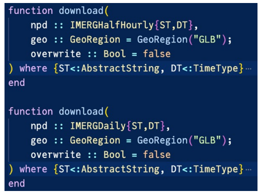
57

Multiple Datasets in a Package

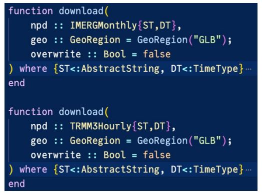


58

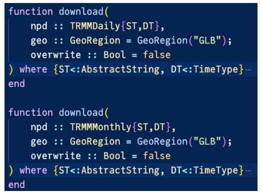
Multiple Datasets in a Package



```
function download(
    npd :: IMERGHalfHourly{ST,DT},
    geo :: GeoRegion = GeoRegion("GLB");
    overwrite :: Bool = false
) where {ST<:AbstractString, DT<:TimeType}⋯
end

function download(
    npd :: IMERGDaily{ST,DT},
    geo :: GeoRegion = GeoRegion("GLB");
    overwrite :: Bool = false
) where {ST<:AbstractString, DT<:TimeType}⋯
end
```



```
function download(
    npd :: IMERGMonthly{ST,DT},
    geo :: GeoRegion = GeoRegion("GLB");
    overwrite :: Bool = false
) where {ST<:AbstractString, DT<:TimeType}⋯
end

function download(
    npd :: TRMM3Hourly{ST,DT},
    geo :: GeoRegion = GeoRegion("GLB");
    overwrite :: Bool = false
) where {ST<:AbstractString, DT<:TimeType}⋯
end
```

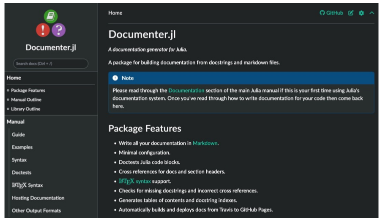


```
function download(
    npd :: TRMMDaily{ST,DT},
    geo :: GeoRegion = GeoRegion("GLB");
    overwrite :: Bool = false
) where {ST<:AbstractString, DT<:TimeType}⋯
end

function download(
    npd :: TRMMMonthly{ST,DT},
    geo :: GeoRegion = GeoRegion("GLB");
    overwrite :: Bool = false
) where {ST<:AbstractString, DT<:TimeType}⋯
end
```

59

Understanding your Package

- You (and your collaborators) are the designer of the package
- You know your package best. ***Does everyone else?***
  - A good Julia package is not only comprehensive and well-organized, but it must also ***be easy for people to understand***
- ***Documentation***! (made using Documenter.jl)
  - PkgTemplates.jl will also set this up



Documenter.jl

A documentation generator for Julia.

A package for building documentation from docstrings and markdown files.

Note

Please read through the Documentation section of the main Julia manual if this is your first time using Julia's documentation system. Once you've read through how to write documentation for your code then come back here.

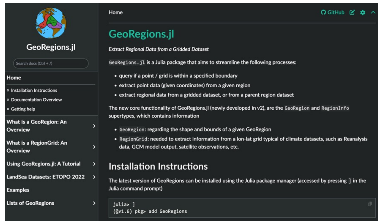
Package Features

- Write all your documentation in Markdown.
- Minimal configuration.
- Doctests Julia code blocks.
- Cross references for docs and section headers.
- LaTeX syntax support.
- Checks for missing docstrings and incorrect cross references.
- Generates tables of contents and docstring indexes.
- Automatically builds and deploys docs from Travis to GitHub Pages.

60

Understanding your Package

- You (and your collaborators) are the designer of the package
- You know your package best. ***Does everyone else?***
  - A good Julia package is not only comprehensive and well-organized, but it must also ***be easy for people to understand***
- ***Documentation***! (made using Documenter.jl)
  - PkgTemplates.jl will also set this up


GeoRegions.jl
Search docs (Ctrl + /)
Home
Installation Instructions
Documentation Overview
Getting help
What is a GeoRegion: An Overview
What is a RegionGrid: An Overview
Using GeoRegions.jl: A Tutorial
LandSea Datasets: ETOPO 2022
Examples
Lists of GeoRegions
Home
GitHub

GeoRegions.jl

Extract Regional Data from a Gridded Dataset

`GeoRegions.jl` is a Julia package that aims to streamline the following processes:

- query if a point / grid is within a specified boundary
- extract point data (given coordinates) from a given region
- extract regional data from a gridded dataset, or from a parent region dataset

The new core functionality of GeoRegions.jl (newly developed in v2), are the `GeoRegion` and `RegionInfo` supertypes, which contains information

- `GeoRegion`: regarding the shape and bounds of a given GeoRegion
- `RegionGrid`: needed to extract information from a lon-lat grid typical of climate datasets, such as Reanalysis data, GCM model output, satellite observations, etc.

Installation Instructions

The latest version of GeoRegions can be installed using the Julia package manager (accessed by pressing `]` in the Julia command prompt)

```
julia> ]
(@v1.6) pkg> add GeoRegions
```

61

TL,DR

- ***PkgTemplates.jl*** is a good tool for package developers in Julia
- Designing a package requires familiarity with the relevant datasets and the variables
- If you want to design a package, I recommend sketching out
  - What your package is supposed to do
  - What datasets and variables are in your package, how are they accessed?
- Multiple datasets can be handled using Julia ***types*** and ***multiple dispatch***
- Documentation is also necessary if you want other people to use your package!
  - Read through Documenter.jl for further details (I won't be able to cover this in this lecture)

62

BREAK! (5 Mins)

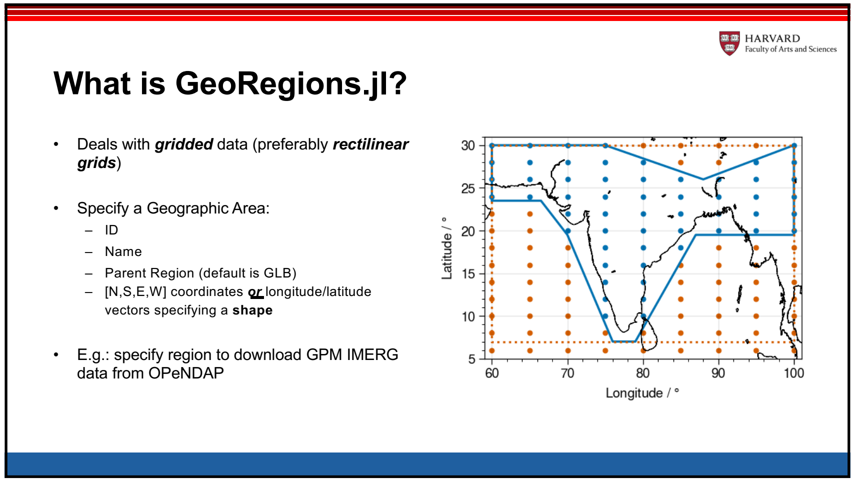
Any questions?

63

The logic behind the development of GeoRegions.jl

CREATING PACKAGES IN JULIA: GEOREGIONS.JL AS AN EXAMPLE

64

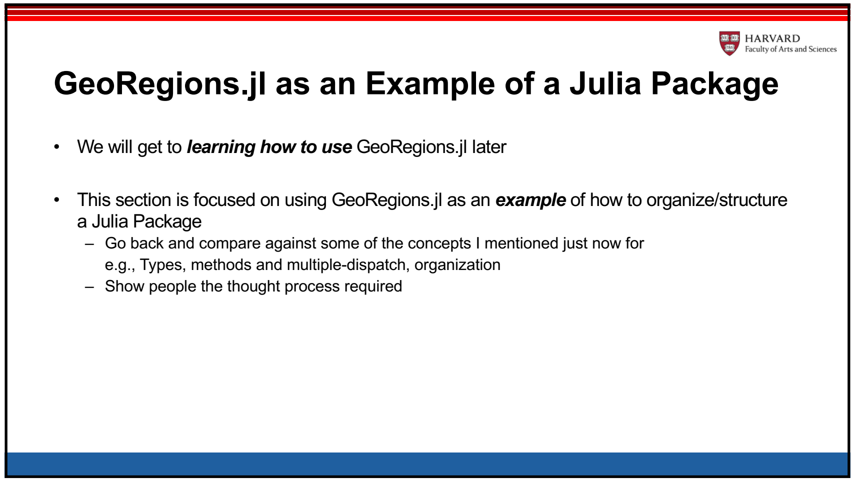


What is GeoRegions.jl?

- Deals with ***gridded*** data (preferably ***rectilinear grids***)
- Specify a Geographic Area:
  - ID
  - Name
  - Parent Region (default is GLB)
  - [N,S,E,W] coordinates **or** longitude/latitude vectors specifying a **shape**
- E.g.: specify region to download GPM IMERG data from OPeNDAP

Latitude / °
30
25
20
15
10
5
60
70
80
90
100
Longitude / °

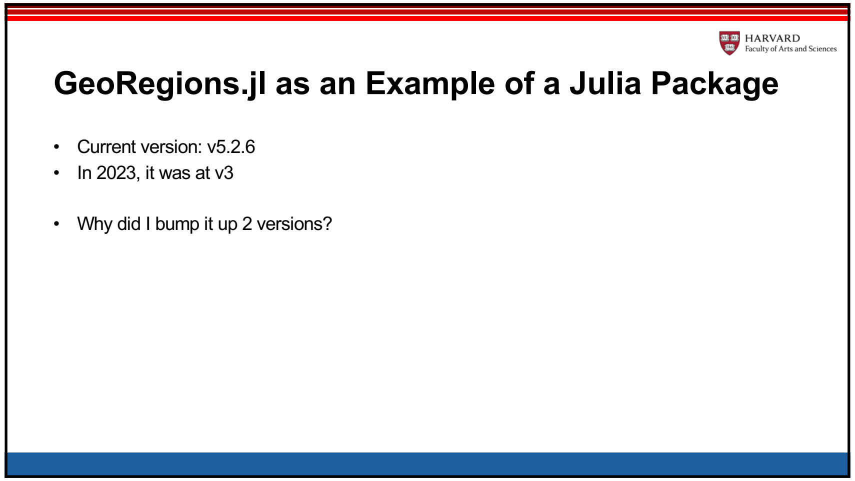
65



GeoRegions.jl as an Example of a Julia Package

- We will get to ***learning how to use*** GeoRegions.jl later
- This section is focused on using GeoRegions.jl as an ***example*** of how to organize/structure a Julia Package
  - Go back and compare against some of the concepts I mentioned just now for e.g., Types, methods and multiple-dispatch, organization
  - Show people the thought process required

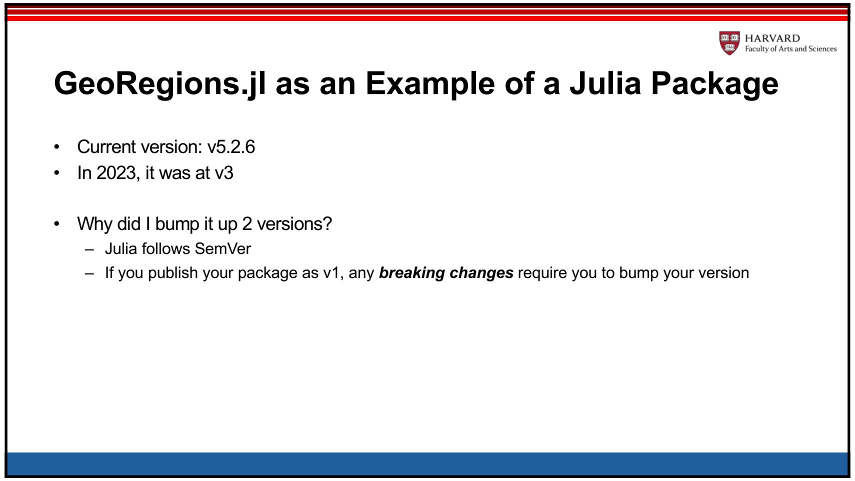
66



GeoRegions.jl as an Example of a Julia Package

- Current version: v5.2.6
- In 2023, it was at v3
- Why did I bump it up 2 versions?

67



GeoRegions.jl as an Example of a Julia Package

- Current version: v5.2.6
- In 2023, it was at v3
- Why did I bump it up 2 versions?
  - Julia follows SemVer
  - If you publish your package as v1, any ***breaking changes*** require you to bump your version

68

HARVARD
Faculty of Arts and Sciences

GeoRegions.jl as an Example of a Julia Package

- Current version: v5.2.6
- In 2023, it was at v3

- Why did I bump it up 2 versions?
  - Julia follows SemVer
  - If you publish your package as v1, any ***breaking changes*** require you to bump your version
  - It is better to wait and ensure that your package is ***stable*** before bumping to v1
    (Note that most of the packages introduced during this workshop are ***v0.X***)

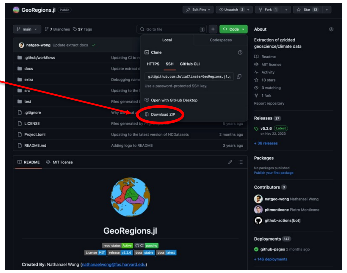
69

HARVARD
Faculty of Arts and Sciences

GeoRegions.jl as an Example of a Julia Package

- Go to https://github.com/JuliaClimate/GeoRegions.jl and download/clone the repository

- We will explore the package together



natgeo-wong Update extract docs

.github/workflows	Updating CI to n	
docs	Update extract d	
extra	Debugging name	
src	Updating to the l	
test	Files generated b	
.gitignore	Why di	
LICENSE	Files generated by	5 years ago
Project.toml	Updating to the latest version of NCDatasets	2 months ago
README.md	Adding logo to README	3 years ago

git@github.com:JuliaClimate/GeoRegions.jl.g

Use a password-protected SSH key.

Extraction of gridded geoscience/climate data

on Nov 22, 2023

No packages published

natgeo-wong Nathanael Wong
pitmonticone Pietro Monticone
github-actions[bot]

github-pages 2 months ago

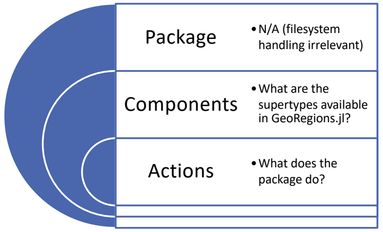
GeoRegions.jl

Created By: Nathanael Wong (nathanaelwong@fas.harvard.edu)

70

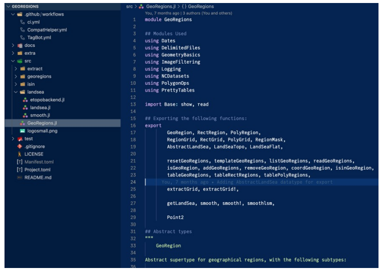
HARVARD
Faculty of Arts and Sciences

GeoRegions.jl as an Example of a Julia Package


Package
- N/A (filesystem handling irrelevant)

Components
- What are the supertypes available in GeoRegions.jl?

Actions
- What does the package do?

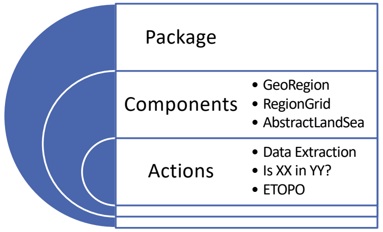


```julia
module GeoRegions

## Modules Used
using Dates
using DelimitedFiles
using GeometryBasics
using ImageFiltering
using Logging
using NCDatasets
using PolygonOps
using PrettyTables

import Base: show, read

## Exporting the following functions:
export
        GeoRegion, RectRegion, PolyRegion,
        RegionGrid, RectGrid, PolyGrid, RegionMask,
        AbstractLandSea, LandSeaTopo, LandSeaFlat,

        resetGeoRegions, templateGeoRegions, listGeoRegions, readGeoRegions,
        isGeoRegion, addGeoRegions, removeGeoRegion, coordGeoRegion, isinGeoRegion,
        tableGeoRegions, tableRectRegions, tablePolyRegions,
      You, 7 months ago • Adding AbstractLandSea datatype for export
        extractGrid, extractGrid!,

        getLandSea, smooth, smooth!, smoothlsm,

        Point2

## Abstract types
"""
    GeoRegion

Abstract supertype for geographical regions, with the following subtypes:
```

71

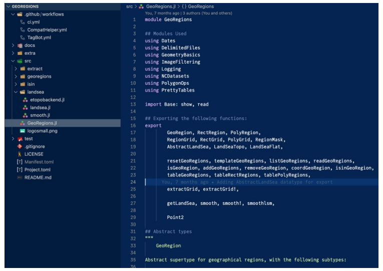
HARVARD
Faculty of Arts and Sciences

GeoRegions.jl as an Example of a Julia Package


Package

Components
- GeoRegion
- RegionGrid
- AbstractLandSea

Actions
- Data Extraction
- Is XX in YY?
- ETOPO


GEOREGIONS
src > GeoRegions.jl > { } GeoRegions
You, 7 months ago | 3 authors (You and others)

module GeoRegions

Modules Used
using Dates
using DelimitedFiles
using GeometryBasics
using ImageFiltering
using Logging
using NCDatasets
using PolygonOps
using PrettyTables

import Base: show, read

Exporting the following functions:
export
        GeoRegion, RectRegion, PolyRegion,
        RegionGrid, RectGrid, PolyGrid, RegionMask,
        AbstractLandSea, LandSeaTopo, LandSeaFlat,

        resetGeoRegions, templateGeoRegions, listGeoRegions, readGeoRegions,
        isGeoRegion, addGeoRegions, removeGeoRegion, coordGeoRegion, isinGeoRegion,
        tableGeoRegions, tableRectRegions, tablePolyRegions,
      You, 7 months ago • Adding AbstractLandSea datatype for export
        extractGrid, extractGrid!,

        getLandSea, smooth, smooth!, smoothlsm,

        Point2

Abstract types
"""
    GeoRegion

Abstract supertype for geographical regions, with the following subtypes:

72

GeoRegions.jl as an Example of a Julia Package

- The master file of PackageName.jl is always found in src/PackageName.jl
- This is the starting point of every package

- In src/GeoRegions.jl, note the order of what I am doing:

73

GeoRegions.jl as an Example of a Julia Package

- The master file of PackageName.jl is always found in src/PackageName.jl
- This is the starting point of every package

- In src/GeoRegions.jl, note the order of what I am doing:

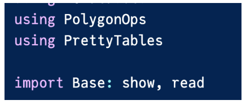
  1. ***Load*** the ***package dependencies*** (information contained in Project.toml)
  2. ***Export*** the ***package functions and Types*** defined in GeoRegions.jl
  3. ***Define*** the most important ***Types*** in your Package (preferably at least the abstract types)
  4. ***Define*** your ***functions*** (and ***include*** other files with functions)

- You can shift the order of (2/Export) around with the others

74

GeoRegions.jl as an Example of a Julia Package

1. Load the ***package dependencies*** (information contained in Project.toml)
   - using is okay for most cases
   - import is used when you want to ***reexport*** a function



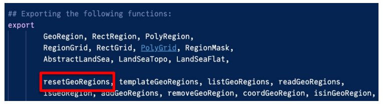
```
using PolygonOps
using PrettyTables

import Base: show, read
```

75

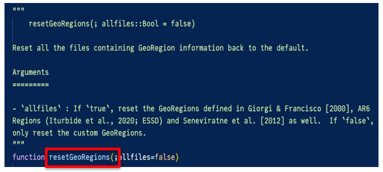
GeoRegions.jl as an Example of a Julia Package

1. Load the ***package dependencies*** (information contained in Project.toml)

2. Export the ***package functions and Types*** defined in GeoRegions.jl
   - Order doesn't matter
   - I usually export the ***Types*** on top, then the functions below
   - Exporting Types is important if you are ***creating parent packages***



```
## Exporting the following functions:
export
        GeoRegion, RectRegion, PolyRegion,
        RegionGrid, RectGrid, PolyGrid, RegionMask,
        AbstractLandSea, LandSeaTopo, LandSeaFlat,

        resetGeoRegions, templateGeoRegions, listGeoRegions, readGeoRegions,
        isGeoRegion, addGeoRegions, removeGeoRegion, coordGeoRegion, isinGeoRegion,
```

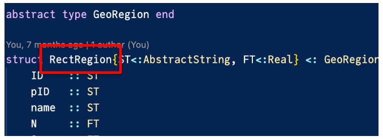


```
"""
    resetGeoRegions(; allfiles::Bool = false)

Reset all the files containing GeoRegion information back to the default.

Arguments
=========

- `allfiles` : If `true`, reset the GeoRegions defined in Giorgi & Francisco [2000], AR6
Regions (Iturbide et al., 2020; ESSD) and Seneviratne et al. [2012] as well.  If `false`,
only reset the custom GeoRegions.
"""
function resetGeoRegions(;allfiles=false)
```

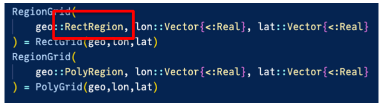
76

GeoRegions.jl as an Example of a Julia Package

1. Load the ***package dependencies*** (information contained in Project.toml)
2. Export the ***package functions and Types*** defined in GeoRegions.jl
3. Define the most important Types in your Package (preferably at least the abstract types)
   – A Type ***must be defined first*** before it can be used in a function



```
abstract type GeoRegion end

You, 7 months ago | 1 author (You)
struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
    ID    :: ST
    pID   :: ST
    name  :: ST
    N     :: FT
```

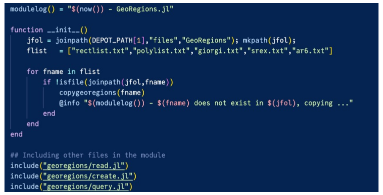


```
RegionGrid(
    geo::RectRegion, lon::Vector{<:Real}, lat::Vector{<:Real}
) = RectGrid(geo,lon,lat)
RegionGrid(
    geo::PolyRegion, lon::Vector{<:Real}, lat::Vector{<:Real}
) = PolyGrid(geo,lon,lat)
```

77

GeoRegions.jl as an Example of a Julia Package

1. Load the ***package dependencies*** (information contained in Project.toml)
2. Export the ***package functions and Types*** defined in GeoRegions.jl
3. Define the most important Types in your Package (preferably at least the abstract types)
4. Define your functions (and ***include*** other files with functions)
   – Order of function definition doesn't matter
   – Reminder: Relevant Types ***must be defined first***

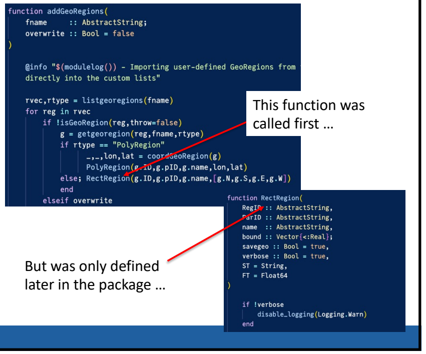


```
modulelog() = "$(now()) - GeoRegions.jl"

function __init__()
    jfol = joinpath(DEPOT_PATH[1],"files","GeoRegions"); mkpath(jfol);
    flist   = ["rectlist.txt","polylist.txt","giorgi.txt","srex.txt","ar6.txt"]

    for fname in flist
        if !isfile(joinpath(jfol,fname))
            copygeoregions(fname)
            @info "$(modulelog()) - $(fname) does not exist in $(jfol), copying ..."
        end
    end
end

## Including other files in the module
include("georegions/read.jl")
include("georegions/create.jl")
include("georegions/query.jl")
```

78

GeoRegions.jl as an Example of a Julia Package

1. Load the ***package dependencies*** (information contained in Project.toml)
2. Export the ***package functions and Types*** defined in GeoRegions.jl
3. Define the most important Types in your Package (preferably at least the abstract types)
4. Define your functions (and ***include*** other files with functions)
   – Order of function definition doesn't matter
   – Reminder: Relevant Types ***must be defined first***



```
function addGeoRegions(
    fname     :: AbstractString;
    overwrite :: Bool = false
)

    @info "$(modulelog()) - Importing user-defined GeoRegions from 
    directly into the custom lists"

    rvec,rtype = listgeoregions(fname)
    for reg in rvec
        if !isGeoRegion(reg,throw=false)
            g = getgeoregion(reg,fname,rtype)
            if rtype == "PolyRegion"
                _,_,lon,lat = coordGeoRegion(g)
                PolyRegion(g.ID,g.pID,g.name,lon,lat)
            else; RectRegion(g.ID,g.pID,g.name,[g.N,g.S,g.E,g.W])
            end
        elseif overwrite
```

```
function RectRegion(
    RegID :: AbstractString,
    ParID :: AbstractString,
    name  :: AbstractString,
    bound :: Vector{<:Real};
    savegeo :: Bool = true,
    verbose :: Bool = true,
    ST = String,
    FT = Float64
)

    if !verbose
        disable_logging(Logging.Warn)
    end
```

This function was called first …

But was only defined later in the package …

79

GeoRegions.jl as an Example of a Julia Package

- I learned all these practices by trial and error, committed to v1 too early
  – This is why the version is large (at v5 currently)

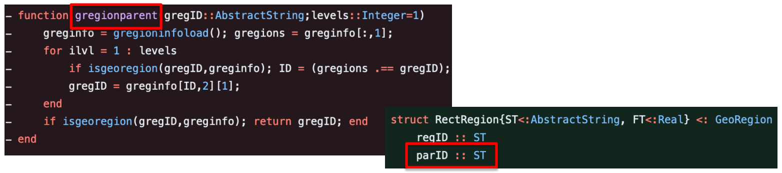
- *v1 → v2: shifted to using Types instead of Dictionaries*
- v2 → v3: breaking changes for dataset downloading (would require people to redownload datasets all over again)
- *v3 → v4: changed field names for **RegionGrid** Types (would break parent packages)*
- *v4 → v5: changed field names for **GeoRegion** Types (would break parent packages)*

80

GeoRegions.jl as an Example of a Julia Package

- I learned all these practices by trial and error, committed to v1 too early
  - This is why the version is large (at v5 currently)

- v1 → v2: shifted to using Types instead of calling Strings



```julia
- function gregionparent(gregID::AbstractString;levels::Integer=1)
-     greginfo = gregioninfoload(); gregions = greginfo[:,1];
-     for ilvl = 1 : levels
-         if isgeoregion(gregID,greginfo); ID = (gregions .== gregID);
-         gregID = greginfo[ID,2][1];
-     end
-     if isgeoregion(gregID,greginfo); return gregID; end
- end
```

```julia
struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
    regID :: ST
    parID :: ST
```

81

GeoRegions.jl as an Example of a Julia Package

- I learned all these practices by trial and error, committed to v1 too early
  - This is why the version is large (at v5 currently)

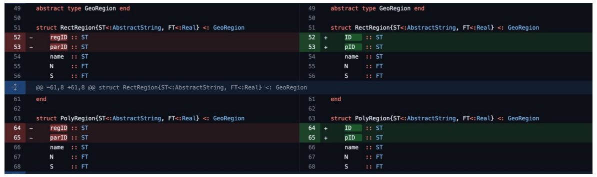
- v1 → v2: shifted to using Types instead of calling Strings
  - GeoRegion information was loaded and ***stored***
  - No need to keep calling functions again and again

82

GeoRegions.jl as an Example of a Julia Package

- I learned all these practices by trial and error, committed to v1 too early
  - This is why the version is large (at v5 currently)

- v3 → v4: changed field names for **RegionGrid** Types (would break parent packages)
- ***v4 → v5: changed field names for GeoRegion Types (would break parent packages)***



```diff
  abstract type GeoRegion end

  struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
-     regID :: ST
-     parID :: ST
      name  :: ST
      N     :: FT
      S     :: FT
@@ -61,8 +61,8 @@ struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
  end

  struct PolyRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
-     regID :: ST
-     parID :: ST
      name  :: ST
      N     :: FT
      S     :: FT
```

```diff
  abstract type GeoRegion end

  struct RectRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
+     ID    :: ST
+     pID   :: ST
      name  :: ST
      N     :: FT
      S     :: FT

  end

  struct PolyRegion{ST<:AbstractString, FT<:Real} <: GeoRegion
+     ID    :: ST
+     pID   :: ST
      name  :: ST
      N     :: FT
      S     :: FT
```

83

GeoRegions.jl as an Example of a Julia Package

- I learned all these practices by trial and error, committed to v1 too early
  - This is why the version is large (at v5 currently)

- v3 → v4: changed field names for **RegionGrid** Types (would break parent packages)
- v4 → v5: changed field names for **GeoRegion** Types (would break parent packages)

- Why did I do this?

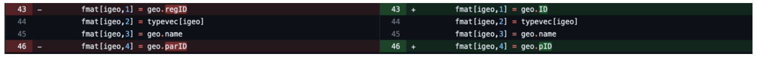
84

GeoRegions.jl as an Example of a Julia Package

- I learned all these practices by trial and error, committed to v1 too early
  - This is why the version is large (at v5 currently)

- v3 → v4: changed field names for **RegionGrid** Types (would break parent packages)
- v4 → v5: changed field names for **GeoRegion** Types (would break parent packages)

- Why did I do this?
  - A: e.g. I felt that using “ID” was more intuitive than “regID”



```
-         fmat[igeo,1] = geo.regID
          fmat[igeo,2] = typevec[igeo]
          fmat[igeo,3] = geo.name
-         fmat[igeo,4] = geo.parID
```

```
+         fmat[igeo,1] = geo.ID
          fmat[igeo,2] = typevec[igeo]
          fmat[igeo,3] = geo.name
+         fmat[igeo,4] = geo.pID
```

85

GeoRegions.jl as an Example of a Julia Package

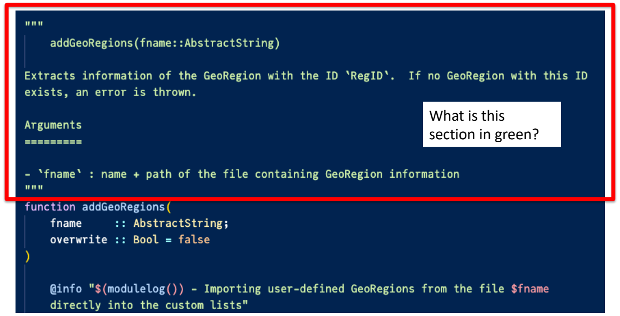
- I learned all these practices by trial and error, committed to v1 too early
  - This is why the version is large (at v5 currently)

- v3 → v4: changed field names for **RegionGrid** Types (would break parent packages)
- v4 → v5: changed field names for **GeoRegion** Types (would break parent packages)

- Why did I do this?
  - A: e.g. I felt that using “ID” was more intuitive than “regID”
  - ***Simpler is always better***

86

GeoRegions.jl as an Example of a Julia Package



```
"""
    addGeoRegions(fname::AbstractString)

Extracts information of the GeoRegion with the ID `RegID`.  If no GeoRegion with this ID
exists, an error is thrown.

Arguments
=========

- `fname` : name + path of the file containing GeoRegion information
"""
function addGeoRegions(
    fname     :: AbstractString;
    overwrite :: Bool = false
)

    @info "$(modulelog()) - Importing user-defined GeoRegions from the file $fname
    directly into the custom lists"
```

What is this section in green?

87

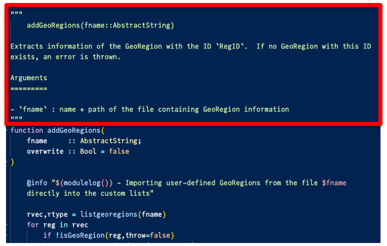
GeoRegions.jl as an Example of a Julia Package

- It is important to provide documentation for your functions

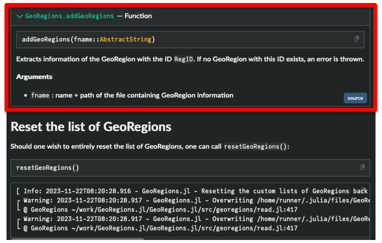
- Good documentation also provides examples and use-cases, not only API functionality

88

GeoRegions.jl as an Example of a Julia Package



```
"""
    addGeoRegions(fname::AbstractString)

Extracts information of the GeoRegion with the ID `RegID`.  If no GeoRegion with this ID
exists, an error is thrown.

Arguments
=========

- `fname` : name + path of the file containing GeoRegion information
"""
function addGeoRegions(
    fname     :: AbstractString;
    overwrite :: Bool = false
)

    @info "$(modulelog()) - Importing user-defined GeoRegions from the file $fname
    directly into the custom lists"

    rvec,rtype = listgeoregions(fname)
    for reg in rvec
        if !isGeoRegion(reg,throw=false)
```



GeoRegions.addGeoRegions — Function

```
addGeoRegions(fname::AbstractString)
```

Extracts information of the GeoRegion with the ID `RegID`. If no GeoRegion with this ID exists, an error is thrown.

Arguments

- `fname` : name + path of the file containing GeoRegion information

source

Reset the list of GeoRegions

Should one wish to entirely reset the list of GeoRegions, one can call `resetGeoRegions()`:

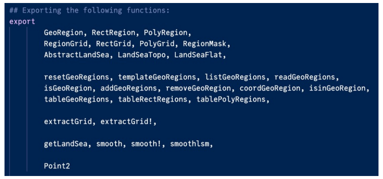
```
resetGeoRegions()
```

```
[ Info: 2023-11-22T08:20:28.916 - GeoRegions.jl - Resetting the custom lists of GeoRegions back
┌ Warning: 2023-11-22T08:20:28.917 - GeoRegions.jl - Overwriting /home/runner/.julia/files/GeoRe
└ @ GeoRegions ~/work/GeoRegions.jl/GeoRegions.jl/src/georegions/read.jl:417
┌ Warning: 2023-11-22T08:20:28.917 - GeoRegions.jl - Overwriting /home/runner/.julia/files/GeoRe
└ @ GeoRegions ~/work/GeoRegions.jl/GeoRegions.jl/src/georegions/read.jl:417
```

89

GeoRegions.jl as an Example of a Julia Package

- You can export functions, and more importantly, ***types***, for use in other packages
- e.g.,
  - GeoRegions.jl exports the LandSeaFlat type

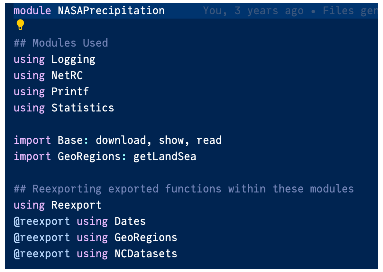


```
## Exporting the following functions:
export
        GeoRegion, RectRegion, PolyRegion,
        RegionGrid, RectGrid, PolyGrid, RegionMask,
        AbstractLandSea, LandSeaTopo, LandSeaFlat,

        resetGeoRegions, templateGeoRegions, listGeoRegions, readGeoRegions,
        isGeoRegion, addGeoRegions, removeGeoRegion, coordGeoRegion, isinGeoRegion,
        tableGeoRegions, tableRectRegions, tablePolyRegions,

        extractGrid, extractGrid!,

        getLandSea, smooth, smooth!, smoothlsm,

        Point2
```

90

GeoRegions.jl as an Example of a Julia Package

- You can export functions, and more importantly, ***types***, for use in other packages
- e.g.,
  - GeoRegions.jl exports the LandSeaFlat type
  - NASAPrecipitation uses this type

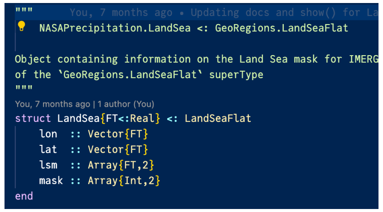


```
module NASAPrecipitation        You, 3 years ago • Files gen

## Modules Used
using Logging
using NetRC
using Printf
using Statistics

import Base: download, show, read
import GeoRegions: getLandSea

## Reexporting exported functions within these modules
using Reexport
@reexport using Dates
@reexport using GeoRegions
@reexport using NCDatasets
```

91

GeoRegions.jl as an Example of a Julia Package

- You can export functions, and more importantly, ***types***, for use in other packages
- e.g.,
  - GeoRegions.jl exports the LandSeaFlat type
  - NASAPrecipitation.jl creates a ***subtype*** of this GeoRegions type



```
"""        You, 7 months ago • Updating docs and show() for La
    NASAPrecipitation.LandSea <: GeoRegions.LandSeaFlat

Object containing information on the Land Sea mask for IMERG
of the `GeoRegions.LandSeaFlat` superType
"""
You, 7 months ago | 1 author (You)
struct LandSea{FT<:Real} <: LandSeaFlat
    lon  :: Vector{FT}
    lat  :: Vector{FT}
    lsm  :: Array{FT,2}
    mask :: Array{Int,2}
end
```

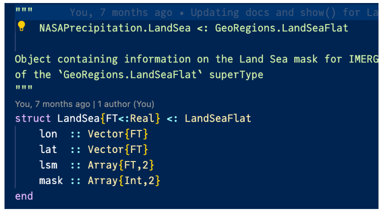
92

GeoRegions.jl as an Example of a Julia Package

- You can export functions, and more importantly, ***types***, for use in other packages

- e.g.,
  - GeoRegions.jl exports the LandSeaFlat type
  - NASAPrecipitation.jl creates a ***subtype*** of this GeoRegions type

- This is how you



```
"""      You, 7 months ago • Updating docs and show() for La
    NASAPrecipitation.LandSea <: GeoRegions.LandSeaFlat

Object containing information on the Land Sea mask for IMERG
of the `GeoRegions.LandSeaFlat` superType
"""
You, 7 months ago | 1 author (You)
struct LandSea{FT<:Real} <: LandSeaFlat
    lon  :: Vector{FT}
    lat  :: Vector{FT}
    lsm  :: Array{FT,2}
    mask :: Array{Int,2}
end
```

93

BREAK! (5 Mins)

Any questions?

94

How do I use GeoRegions.jl in Earth Observation work?

USING GEOREGIONS.JL

95

Breakdown of Today's Tutorial

- Using GeoRegions.jl
  - Defining your own GeoRegion
  - Properties of a GeoRegion

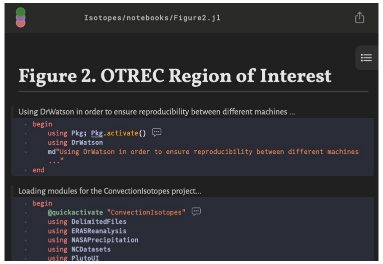
- How do you use GeoRegions?
  - Data Extraction for a particular region (defined by a GeoRegion)
  - Is a point/region within a GeoRegion of interest?
  - Land-Sea Mask Datasets (retrieving and manipulating ETOPO data)

- Using GeoRegions.jl in other packages

96

Breakdown of Today’s Tutorial

- For this part, we will be using Pluto.jl notebooks! (https://plutojl.org)
  - Good for step-by-step story format presentation
  - Interactive and in-real-time
- I like using Pluto.jl notebooks to do data visualization and presentation



Isotopes/notebooks/Figure2.jl

Figure 2. OTREC Region of Interest

Using DrWatson in order to ensure reproducibility between different machines ...

```
begin
    using Pkg; Pkg.activate()
    using DrWatson
    md"Using DrWatson in order to ensure reproducibility between different machines
    ..."
end
```

Loading modules for the ConvectionIsotopes project...

```
begin
    @quickactivate "ConvectionIsotopes"
    using DelimitedFiles
    using ERA5Reanalysis
    using NASAPrecipitation
    using NCDatasets
    using PlutoUI
```

97

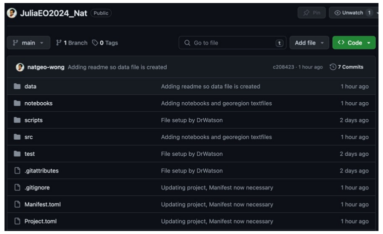
Breakdown of Today’s Tutorial

- Today’s notebooks are ***not*** a exhaustive tutorial of the functionalities of GeoRegions.jl
  - For a more comprehensive breakdown, it is always best to refer to the documentation
- Today’s introduction to GeoRegions.jl is more to reinforce concepts taught earlier, i.e.,
  - how do you want to ***design*** a package and what is it supposed to ***do***?
  - using Types and multiple dispatch
  - exporting package functionality for future use (e.g., how GeoRegions.jl Types can be exported for use in parent packages such as NASAPrecipitation.jl)

98

Breakdown of Today’s Tutorial

- Go to https://github.com/natgeo-wong/JuliaEO2024_Nat
- Clone the repository there



natgeo-wong Adding readme so data file is created c208423 · 1 hour ago 7 Commits

data	Adding readme so data file is created	1 hour ago
notebooks	Adding notebooks and georegion textfiles	1 hour ago
scripts	File setup by DrWatson	2 days ago
src	Adding notebooks and georegion textfiles	1 hour ago
test	File setup by DrWatson	2 days ago
.gitattributes	File setup by DrWatson	2 days ago
.gitignore	Updating project, Manifest now necessary	1 hour ago
Manifest.toml	Updating project, Manifest now necessary	1 hour ago
Project.toml	Updating project, Manifest now necessary	1 hour ago

99

Opening the notebooks

- First, you need to setup the environment?
- Recall, what do you need to do?
- ***Precompile the environment!***

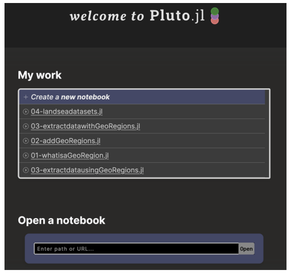
100

HARVARD
Faculty of Arts and Sciences

Opening the notebooks

- Go into the ***notebooks*** folder

1. Run Julia
2. Loading the Pluto.jl package (*using Pluto*)
3. Open a Pluto notebook session (*Pluto.run()*)



welcome to **Pluto**.jl

My work

- + *Create a **new** notebook*
- 04-landseadatasets.jl
- 03-extractdatawithGeoRegions.jl
- 02-addGeoRegions.jl
- 01-whatisaGeoRegion.jl
- 03-extractdatausingGeoRegions.jl

Open a notebook

101

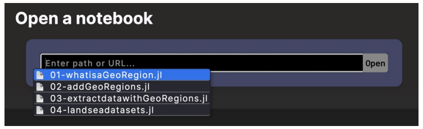
HARVARD
Faculty of Arts and Sciences

Opening the notebooks

- Go into the ***notebooks*** folder

1. Run Julia
2. Loading the Pluto.jl package (*using Pluto*)
3. Open a Pluto notebook session (*Pluto.run()*)

- From there, you can open Pluto notebooks



Open a notebook

- 01-whatisaGeoRegion.jl
- 02-addGeoRegions.jl
- 03-extractdatawithGeoRegions.jl
- 04-landseadatasets.jl

102

HARVARD
Faculty of Arts and Sciences

Opening the notebooks

- Go into the ***notebooks*** folder

1. Run Julia
2. Loading the Pluto.jl package (*using Pluto*)
3. Open a Pluto notebook session (*Pluto.run()*)

- From there, you can open Pluto notebooks
  - We will go in order from 01 to 04

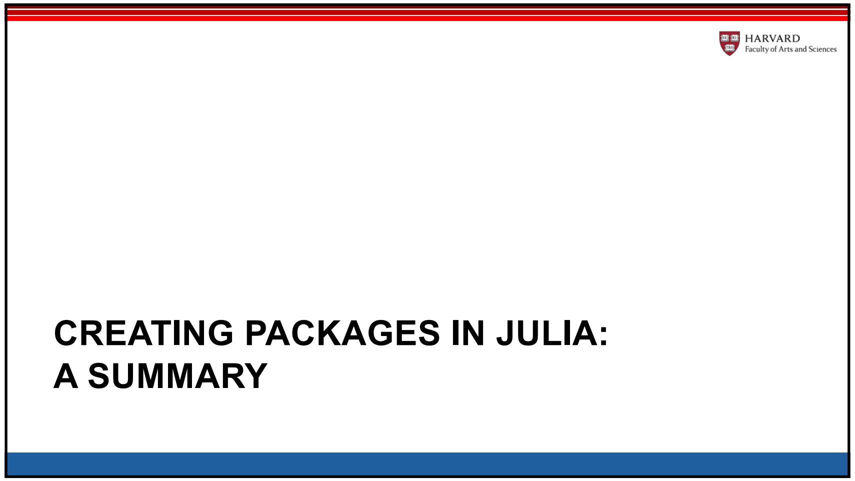
103

HARVARD
Faculty of Arts and Sciences

Notebooks Time!

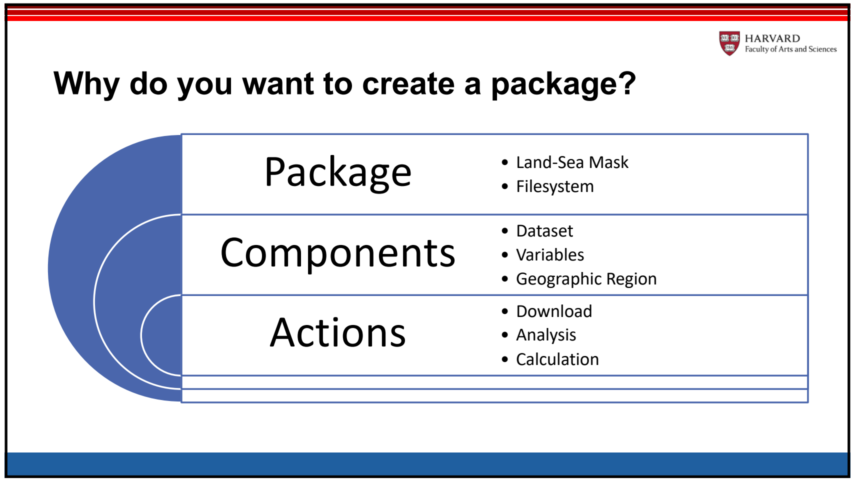
We'll be using notebooks for this part of the lecture

104



CREATING PACKAGES IN JULIA: A SUMMARY

105



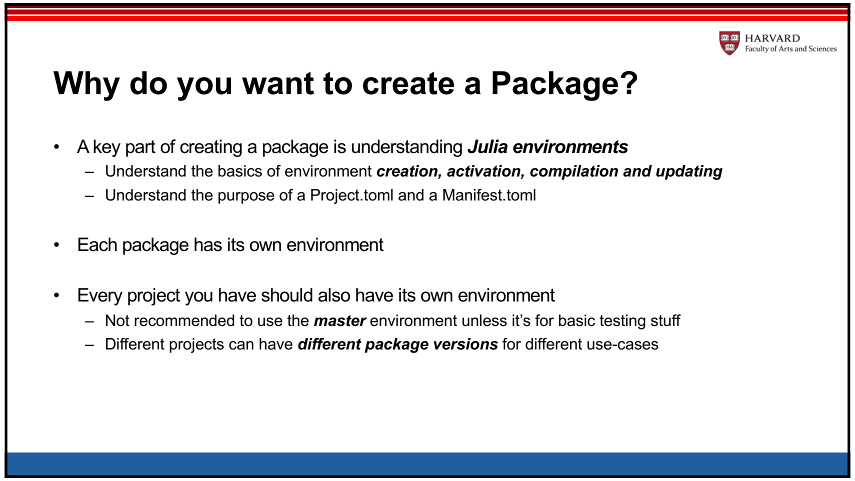
Why do you want to create a package?

Package
- Land-Sea Mask
- Filesystem

Components
- Dataset
- Variables
- Geographic Region

Actions
- Download
- Analysis
- Calculation

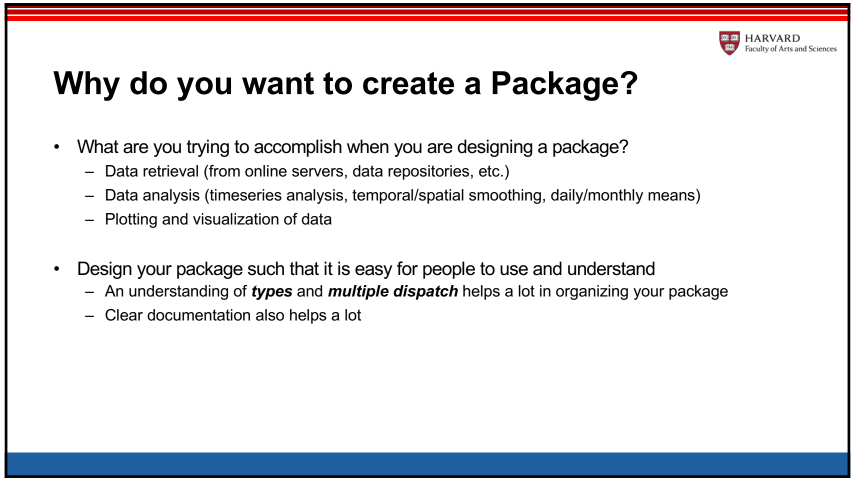
106



Why do you want to create a Package?

- A key part of creating a package is understanding ***Julia environments***
  - Understand the basics of environment ***creation, activation, compilation and updating***
  - Understand the purpose of a Project.toml and a Manifest.toml
- Each package has its own environment
- Every project you have should also have its own environment
  - Not recommended to use the ***master*** environment unless it's for basic testing stuff
  - Different projects can have ***different package versions*** for different use-cases

107



Why do you want to create a Package?

- What are you trying to accomplish when you are designing a package?
  - Data retrieval (from online servers, data repositories, etc.)
  - Data analysis (timeseries analysis, temporal/spatial smoothing, daily/monthly means)
  - Plotting and visualization of data
- Design your package such that it is easy for people to use and understand
  - An understanding of ***types*** and ***multiple dispatch*** helps a lot in organizing your package
  - Clear documentation also helps a lot

108

HARVARD
Faculty of Arts and Sciences

The End

Thanks for listening!

109