Instructivo para instalar Julia

R. Ignacio Navarrete - raul.navarrete.o@usach.cl

Julia es un lenguaje de programación interpretado (*free-translate* de *interpreted languajes*) similar a MATLAB, que corre considerablemente más rápido que otros lenguajes del mismo tipo, acercándose a algunos de alto nivel como C++. Existe un *working paper* interesante que compara la *performance* de distintos lenguajes para resolver el problema del modelo neoclásico de crecimiento, caso estocástico. Les dejo la referencia.

Aruoba & Fernández-Villaverde (2014) A Comparison of Programming Languaje in Economics. *NBER Working Paper Series.*

Como spoiler, claramente lenguajes compilados como C++ o Fortran tienen un rendimiento muy superior a lenguajes interpretados, sin embargo, en esta categoría, Julia supera por bastante a MATLAB o R.

Julia es un lenguaje de código abierto, al igual que otros lenguajes como R o Python. Entonces al igual que R lo podemos correr en RStudio o Python en PyCharm; para Julia, necesitamos instalar un editor o IDE para trabajarlo cómodamente, además del lenguaje en sí mismo. En este caso, recomiendo usar Julia vía Atom con la extensión de Juno, sin embargo, existen más opciones.

Instalar Julia

Lo primero claramente es instalar el lenguaje, en este caso Julia, eligiendo su sistema operativo Windows o MacOS, y si su ordenador es de 32 o 64-bits, para el caso de "*installer*".

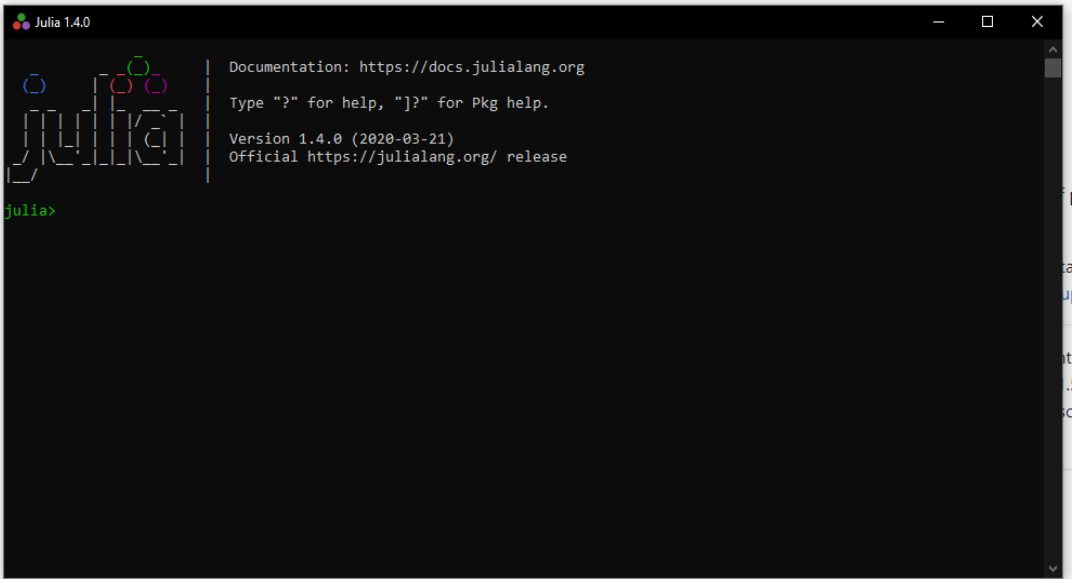
https://julialang.org/downloads/

Current stable release: v1.5.3 (Nov 9, 2020)

Checksums for this release are available in both MD5 and SHA256 formats.

Windows [help]	64-bit (installer), 64-bit (portable)	32-bit (installer), 32-bit (portable)	
macOS [help]	64-bit		
Generic Linux on x86 [help]	64-bit (GPG), 64-bit (musl)[1] (GPG)	32-bit (GPG)	
Generic Linux on ARM [help]	64-bit (AArch64) (GPG)		
Generic FreeBSD on x86 [help]	64-bit (GPG)		
Source	Tarball (GPG)	Tarball with dependencies (GPG)	GitHub

Luego de descargado el *installer*, lo ejecutan en la ubicación predeterminada o en su disco de preferencia. Para revisar que está bien instalado, una vez terminado vamos a escribir en el buscador de programas "Julia". Esto lo podemos hacer usando la "lupa" de Windows. Debiesen ver algo así.



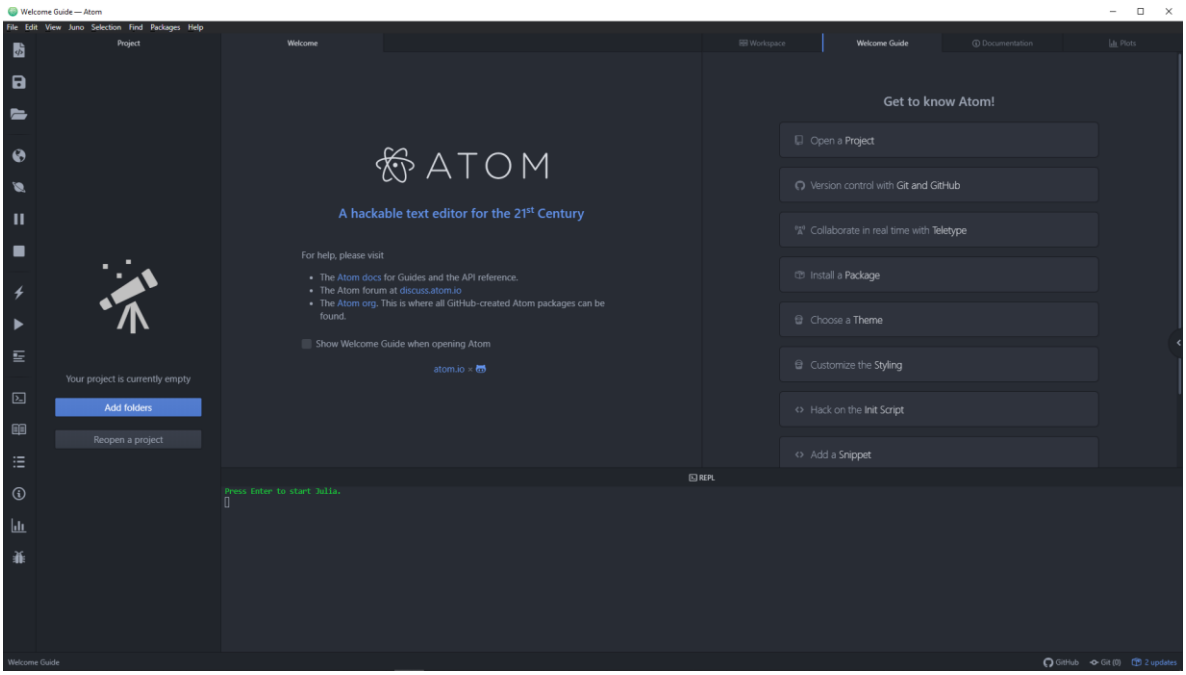
Ahora pasamos a ver el tema del editor o IDE.

Opción 1: Juno

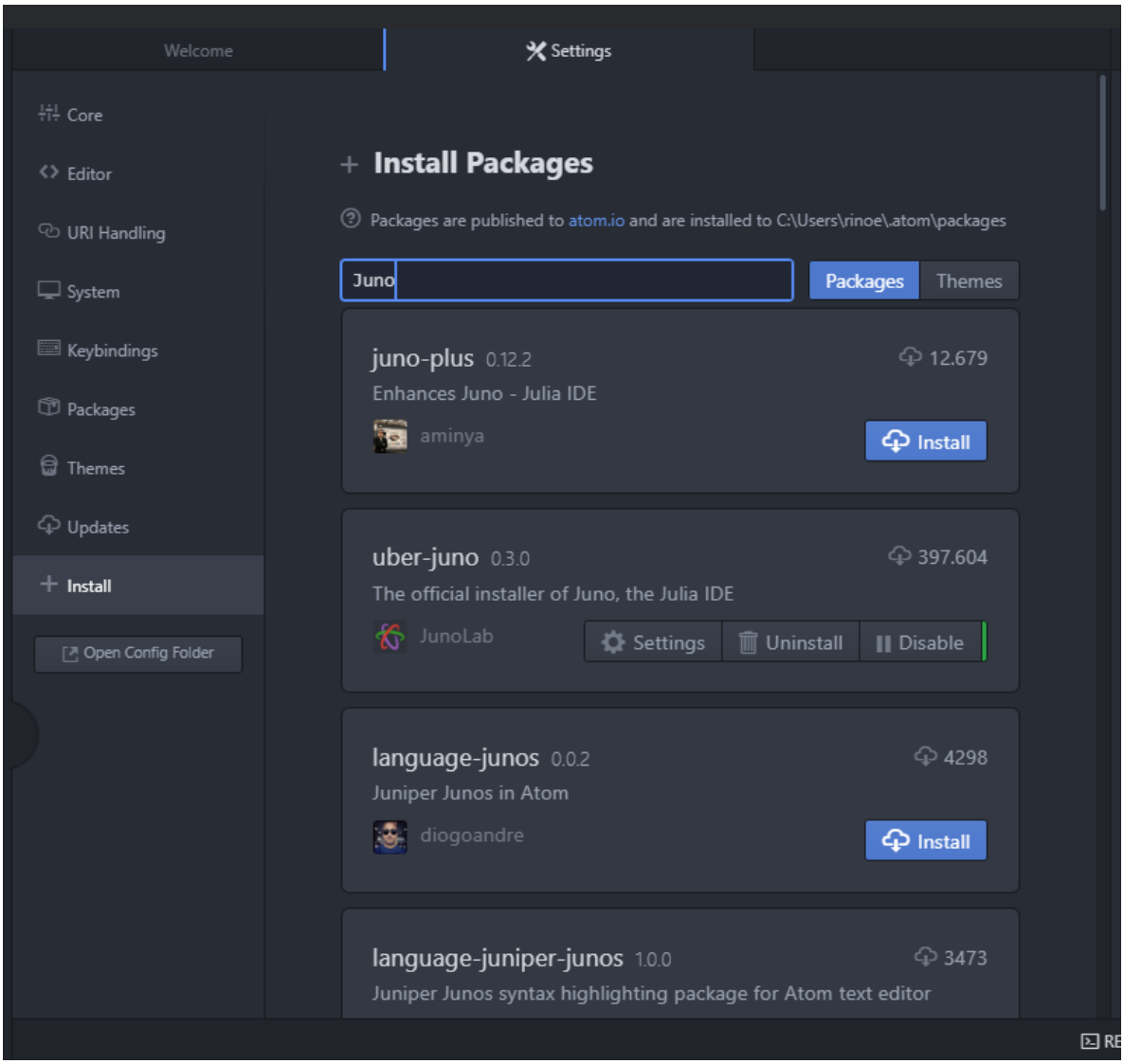
Descargamos Atom, nuevamente en la ubicación de *default* o disco de preferencia. Uno de los puntos a favor de Atom, es que además de que es un editor ameno y personalizable (Tiene varios temas, *packages*, etc); podemos conectarlo directamente a nuestra cuenta en GitHub, para tener así un espacio y flujo de trabajo ordenado, rastreable y fácilmente cooperativo.

https://atom.io/

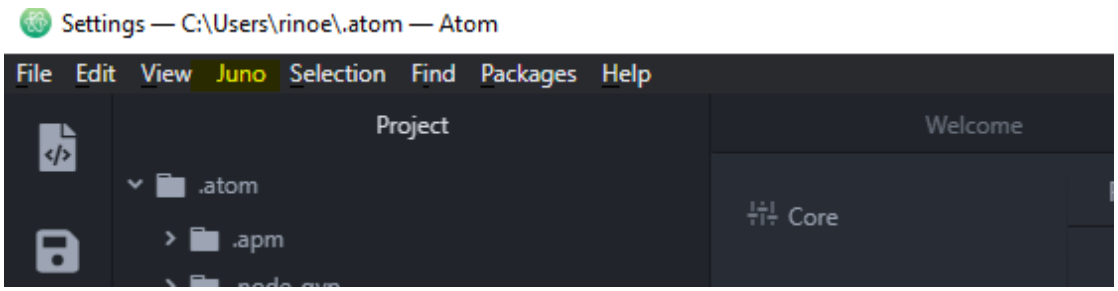
Luego, al abrir el programa debiesen ver algo así.



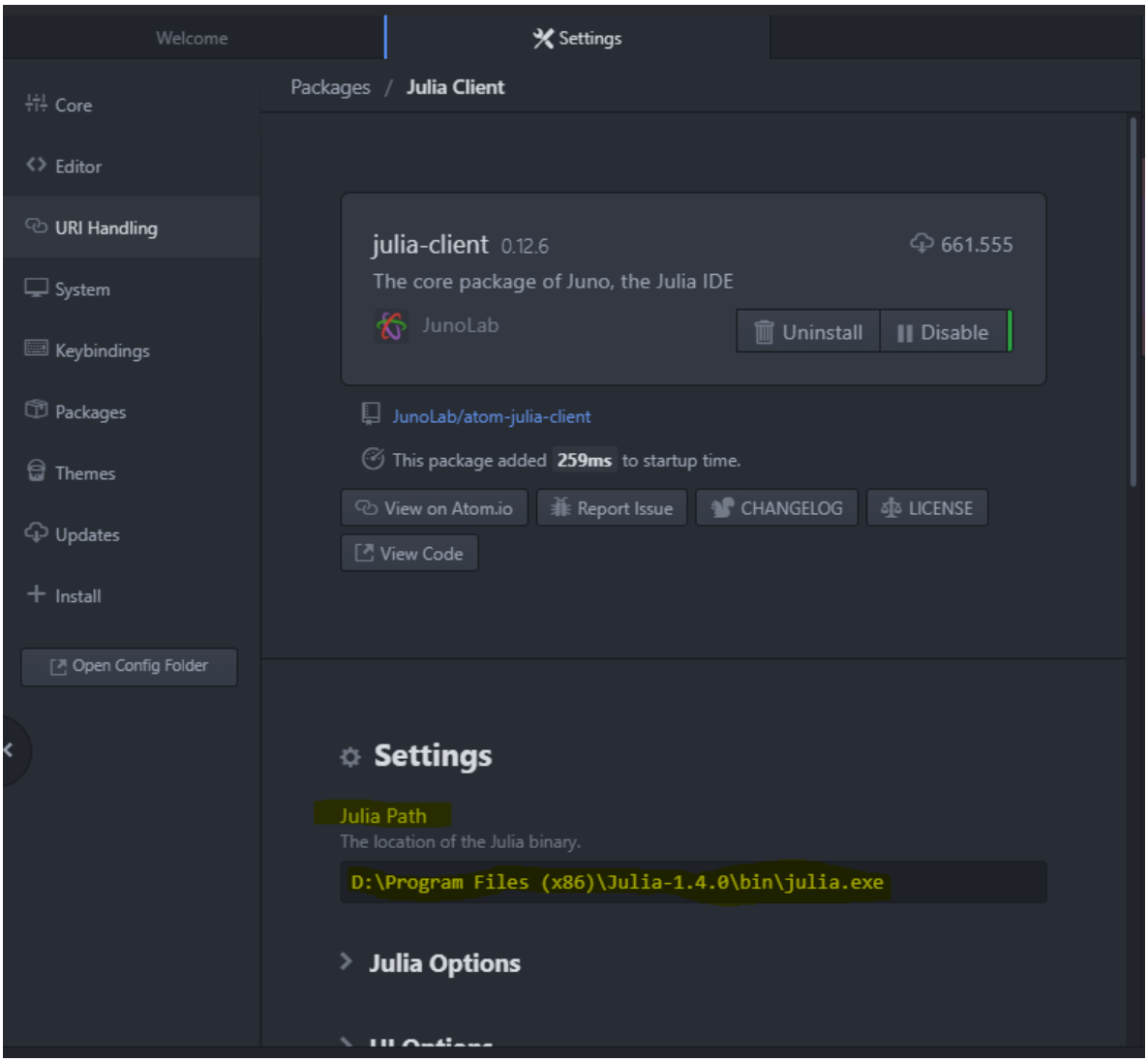
Deben ir a *Settings*, mediante el panel superior en *"File -> Settings"* o el *shortcut* "CTRL+,". Luego nos vamos a la sección de *"Install"* y en el buscador escribimos "Juno", para instalar la opción "uber-juno 0.3.0"



Ahora, lo que tenemos que hacer es especificar la ubicación donde se encuentra nuestro lenguaje, Julia. En el mismo panel superior donde se encuentra "File", debiesen observar ahora una opción que dice "Juno" o "Julia". (Destacado en amarillo)

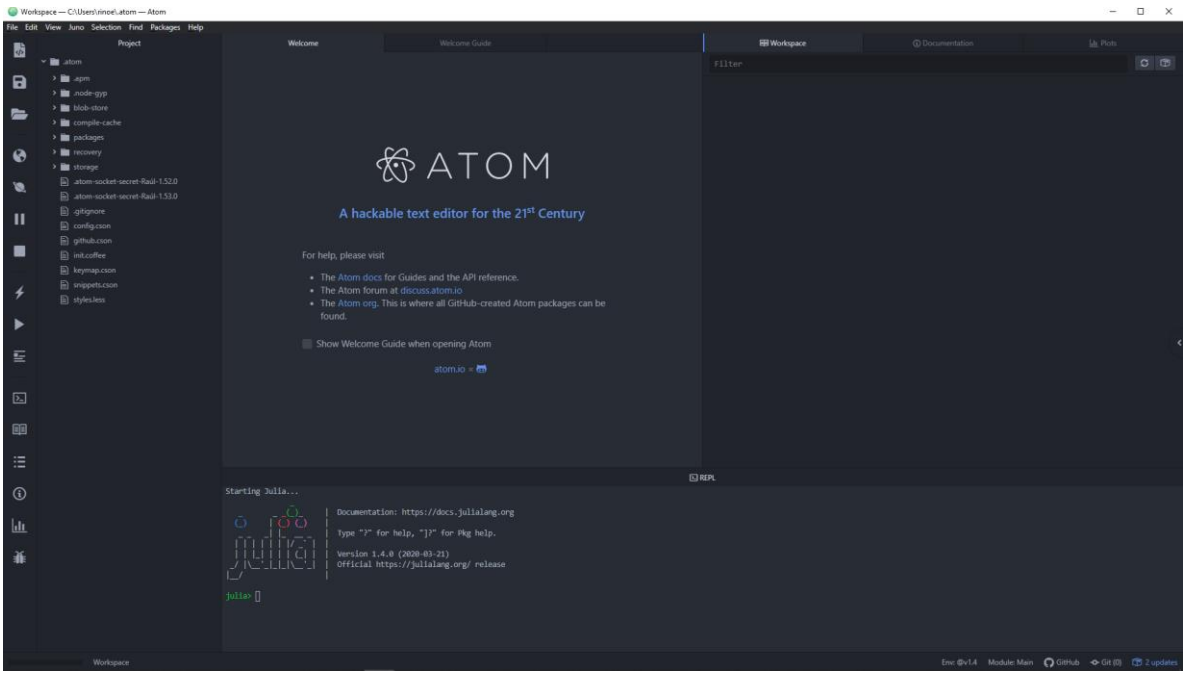


Abrimos esa sección y al final nos vamos a *"Settings"* (Entiendo que se tienen configuraciones a nivel del IDE, en este caso Atom; y configuraciones a nivel del cliente o extensión que estamos trabajando, en este caso "Uber-Juno", por eso la diferencia). Acá tienen que indicar la ubicación de donde instalaron Julia en un principio, para que el editor pueda ejecutarlo (Destacado en amarillo)



(OJO) que deben especificar la dirección del programa, no sólo de la carpeta. Por lo tanto, la carpeta del programa tendrán algo como "C:\ "Dirección donde lo instalaron"\bin", deben agregarle el programa "julia.exe"; o si no solo estarán especificando la dirección y no el programa. Entonces básicamente copian al *path*, y le agregan el nombre del programa "julia.exe".

Hecho esto, reiniciamos Atom. Con esto estamos listos para entrar a trabajar en Julia vía Juno (Atom). Cuando abran Atom, deben "iniciar Julia", seleccionando el "REPL" (abajo) que es el equivalente de Julia que teníamos al principio (primera imagen), apretando "Enter".



Para instalar algún *package*, una vez iniciado Julia, en el mismo REPL deben escribir un "]" (paréntesis cuadrado a la derecha, el que cierra]), lo que hará que cambie el "*input*" y tendrán este "(@version) pkg ". Ahí escribiendo "*add __"* pueden instalar *packages* que les sean útiles, también en la misma sección pueden escribir una suerte de *helper* mediante "?"

Para volver a la otra opción que es la que ocuparán, solamente apretan *"backspace"* o "Ctrl+C".

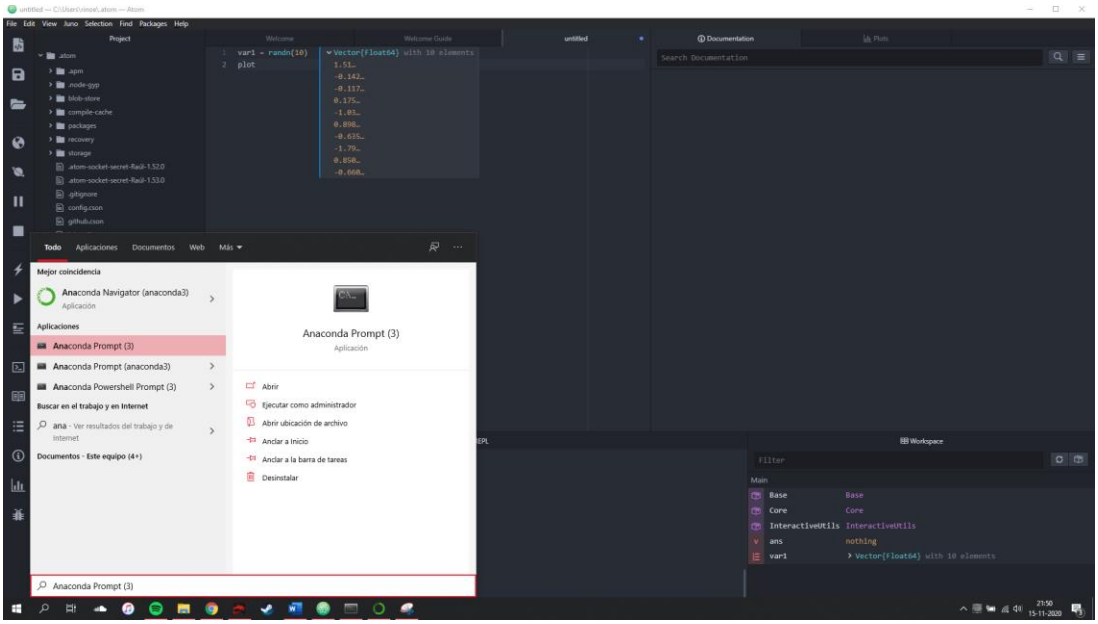
El editor es super flexible y pueden configurar la ubicación del REPL, documentación, plots, archivo de julia en cualquier lugar, incluso teniendo la opción de ver paralelamente varios archivos, a diferencia de RStudio que no tiene este opción (Al igual que lo podemos hacer en MATLAB).

Finalmente, cuando escriban un código o "*script",* guárdenlo de inmediato con la extensión ".jl" que es el dominio de Julia, para que Atom pueda reconocerlo posteriormente y aprovechar la interfaz.

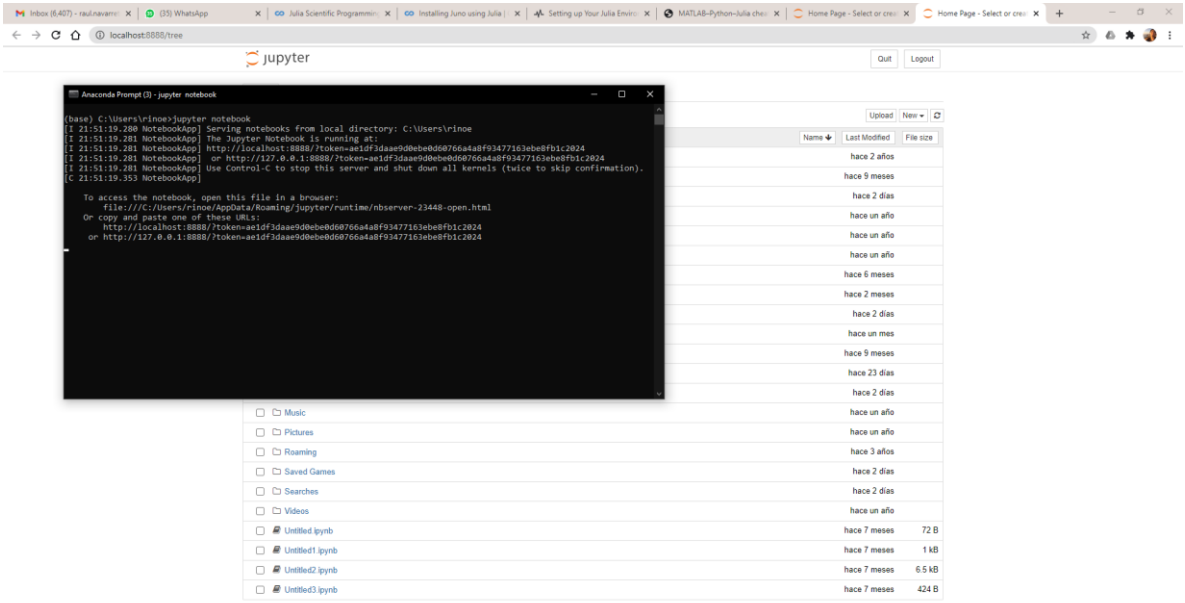
Opción 2: Jupyter

La otra opción es usar Julia vía Jupyter, que es un IDE que ocupa nuestro navegador, parecido a lo que hace Markdown al crear un archivo ".html".

Instalado Julia del punto inicial, bajamos e instalamos Anaconda (al final abajo están las descargas, https://www.anaconda.com/products/individual). Luego, probamos que la instalación esté correcta, abriendo el "cmd" del programa. En el buscador, escribimos "Anaconda", seleccionando "Anaconda prompt" (El que está seleccionado)

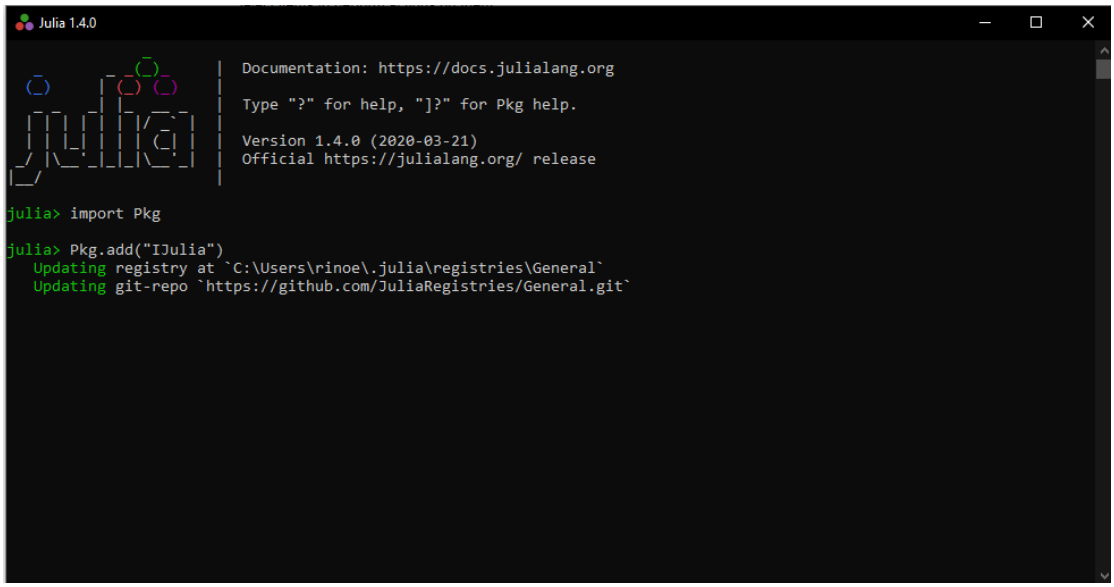


Luego, typeamos "jupyter notebook", lo que nos abrirá Jupyter en nuestro navegador determinado (Imagen).

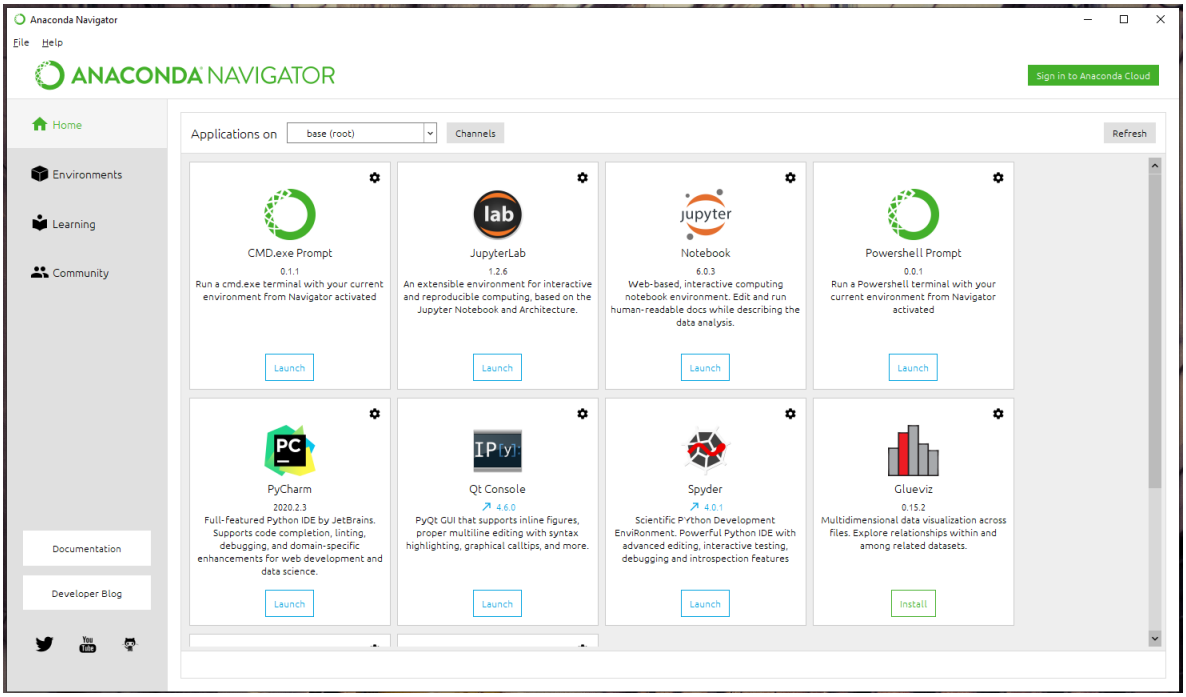


Después de chequear que efectivamente los “notebooks” estén funcionando, agregamos la extensión para Julia. (Hasta acá solo hemos instalado la funcionalidad de Jupyter, que a gusto personal es súper útil para combinar texto, código y output en un mismo “documento”).

En el “*prompt*” de Julia que vimos al principio, una vez instalado Jupyter-notebooks, escribimos “Import Pkg” y luego de ejecutarlo, escribimos “ Pkg.add(“IJulia”) “, tal cual en la imagen.

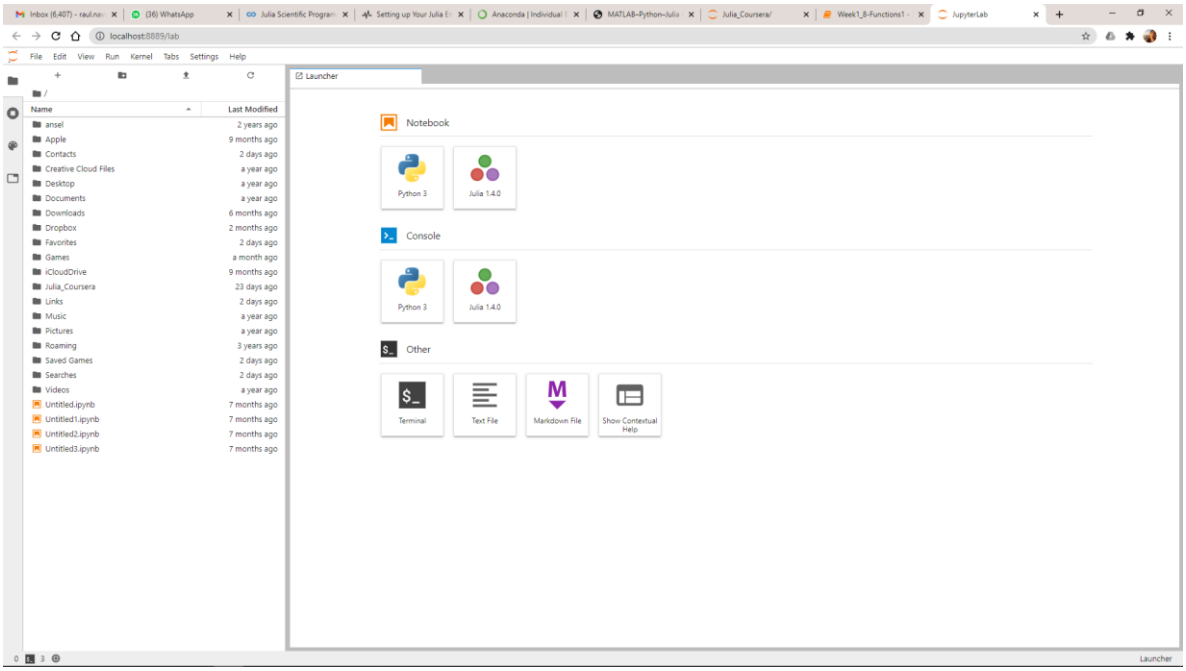


Después de ejecutar e instalado, estamos listos. Para abrir una sesión de Jupyter-notebook, podemos repetir el *step* anterior de esta opción, donde abrimos “Anaconda prompt” y escribimos “jupyter-notebook” o bien, abrimos el programa Anaconda (Imagen)

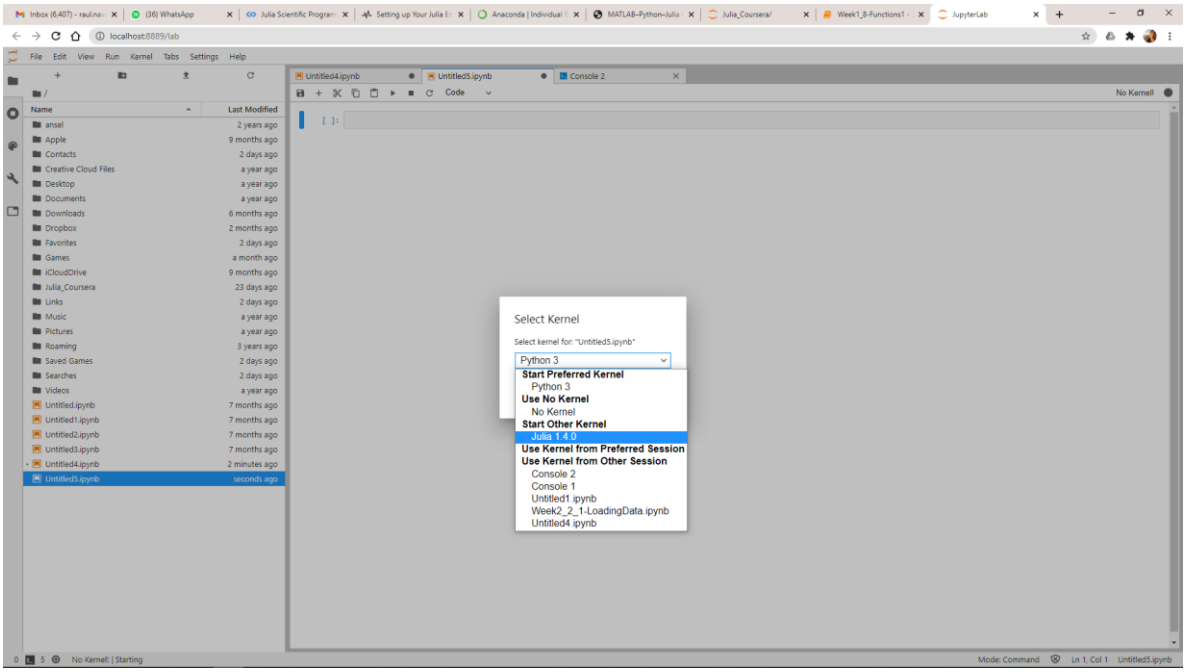


Básicamente Anaconda funciona como una plataforma para abrir distintos programas, de ahí podemos abrir Jupyter u otros programas que nos interesen (Como PyCharm). En nuestro caso, tanto Jupyter notebook como JupyterLab nos sirven (En lo personal prefiero JupyterLab, pero prefiero aún más Juno)

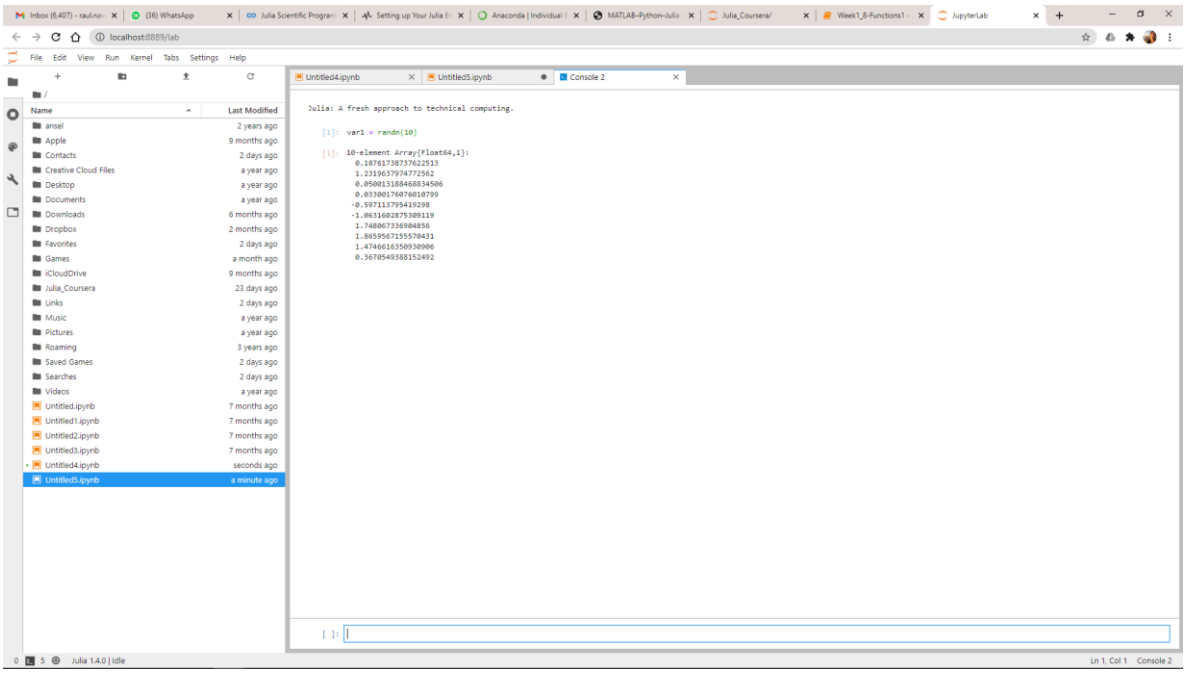
Al abrir uno (en este caso JupyterLab) vemos algo parecido a esto.



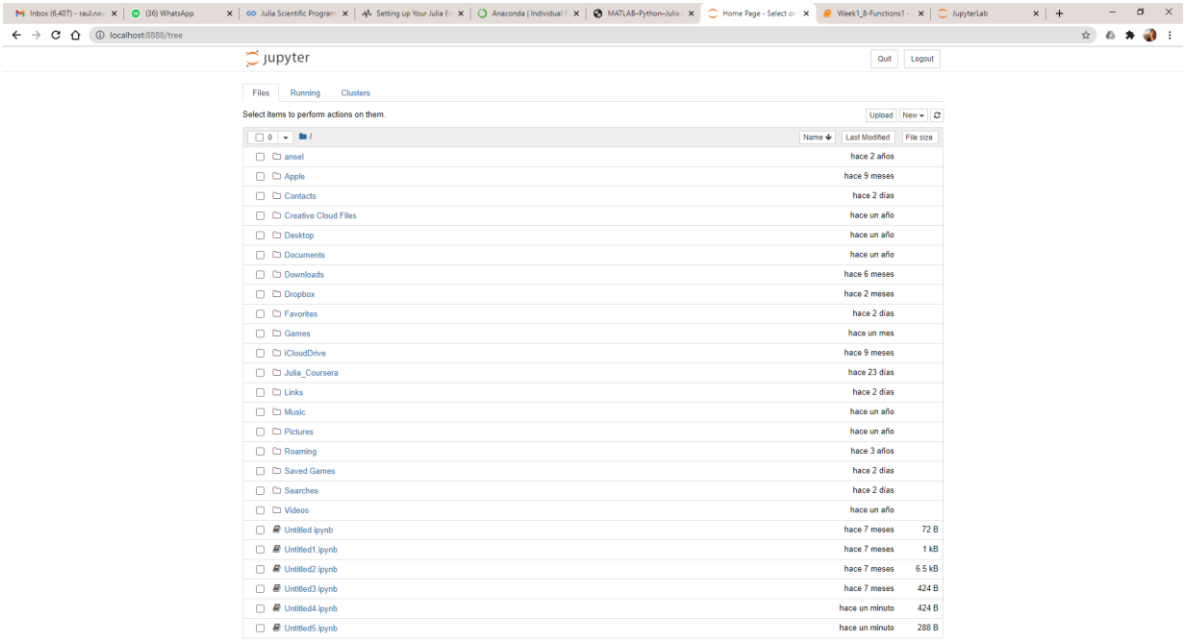
Iniciamos una sesión de Julia (Sea en consola, por ejemplo, el equivalente al REPL). O bien, un notebook mediante la otra opción. También podemos hacer esto en el panel superior mediante "*File -> New launcher"* o "*File -> New -> Console/Notebook"*.



Si está corriendo de buena forma, algo simple como esto les andará bien (Shift + Enter)



En Jupyter también podemos escoger temas, y otras cosas. Con esto ya estamos listos, en adelante pueden abrir Anaconda directamente para abrir nuevas sesiones de Jupyter, importar documentos (justo abajo del panel superior hay una flecha que cumple esta función), importar carpetas enteras con documentos que sean legibles para Jupyter, etc. Claramente, también pueden guardar códigos, los cuales se visualizan en acá.



Debo destacar el sentido pedagógico de Jupyter, sobretodo para enseñar asuntos relativos a programación, ya que tener un documento formateado (parecido a pdf) en conjunto con el código legible y modificable (como un script) y ejecutable (en el REPL), es una herramienta súper útil.

Esas son sólo dos formas de instalar Julia, seguro hay más, pero considero que son las dos mejores herramientas.

Les dejo también acá los steps para Jupyter, sacado de Coursera

Option 1: Installation of Julia and Jupyter notebook locally on your machine

As with the end of the free version of JuliaBox, you have a couple of options to set up a Julia coding environment. One of the most convenient ways is to install Julia and Jupyter notebook locally on your machine. You might consider this option if you do not have reliable internet access and want to set up a notebook environment locally on your computer for free.

Julia and Jupyter Notebook installation on **Windows OS**:

1. Download Julia from Julia Download. You can download Julia binary for 32-bit or 64-bit (.exe). Current stable version is: **v1.5.0**
2. After installing Julia using the **.exe** installer, you can open Julia REPL to test if the installation was successful.
3. Download Anaconda Distribution from Anaconda Distribution. You can download the Python **3.7** version for Windows.
4. After installing Anaconda (takes about +-7 minutes), open the **Anaconda Prompt** (similar to macOS terminal). In the prompt, type "**python --version**" to verify that Anaconda Distribution was successfully installed.
5. To test if a Jupyter Notebook instance is working, you can type "**jupyter notebook**" on the prompt.
6. Next step is to install the Julia kernel on the Jupyter Notebook. Open Julia REPL and type "**import Pkg**" followed by "**Pkg.add("IJulia")**.
7. Once installation is completed, open Anaconda prompt again and open Jupyter Notebook by typing "**jupyter notebook**".
8. Go to **new** and create a new Julia notebook instance. You can test your notebook is working by writing some Julia code such as **"println("Hello World")**. If you can execute the code, your Julia installation on the Jupyter Notebook is successful.

Julia and Jupyter Notebook installation on **macOS**:

1. Download Julia from Julia Download. Download Julia binary for macOS package i.e. **macOS 10.8+ (.dmg) 64 bit version**. Current stable release is: **v.1.5.0**
2. After installing Julia using the **.dmg** file, Open the Julia REPL and test if the installation was successful
3. Download Anaconda Distribution from Anaconda Distribution. You can download the Python **3.7** version for macOS.
4. After installing Anaconda (takes about +-7 minutes), open Mac **Terminal**. Test if the Jupyter Notebook can be launched by typing "**jupyter notebook**".
5. Next step is to install the Julia kernel on the Jupyter Notebook. Launch Julia REPL and type "**import Pkg**" followed by "**Pkg.add("IJulia")**".
6. Once IJulia is installed, open the terminal again and launch Jupyter Notebook. Go to "new" and check if you have the latest version of Julia kernel.