


MAGEMin framework


(Mineral Assemblage Gibbs Energy Minimization)

MAGEMin





- MPI-parallel C code
- Point-wise minimization at given P-T-X
- Metapelite (White et al., 2014)
- Metabasite (Green et al., 2016)
- Igneous (Holland et al., 2018)
- Ultramafic (Evans & Forst, 2021)
- Mantle (Stixude & Lithgow-Bertelloni, 2010)

MAGEMin_C




- Julia wrapper of the C code
- Flexible programming interface
- Database calibration
- Geodynamic coupling

MAGEMinApp



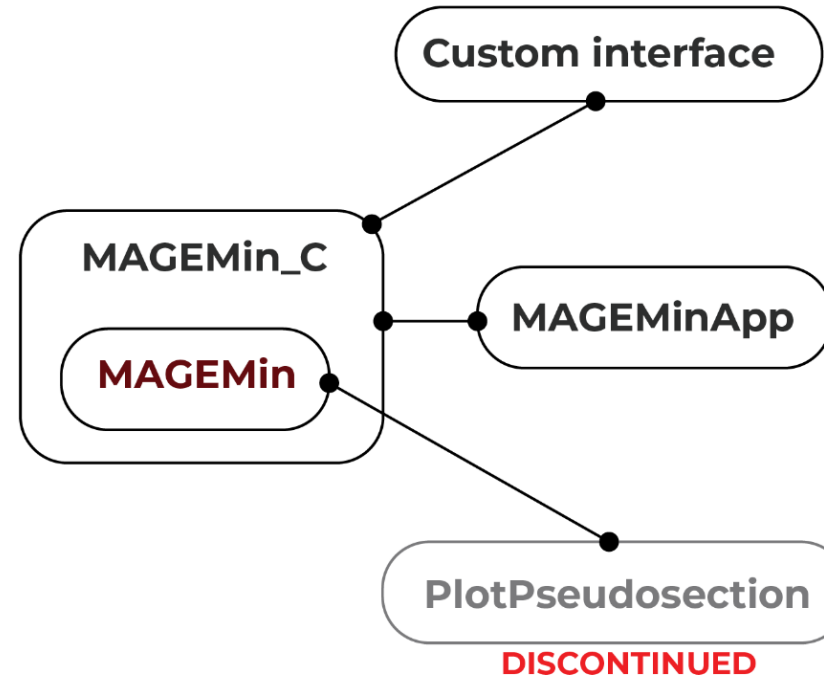
- Web browser app (graphic user interface)
- Parallel point-wise minimization
- PT, PX, TX and PT-X phase diagrams
- Auto labelling, contouring
- Fractional melting/crystallization paths

PlotPseudosection

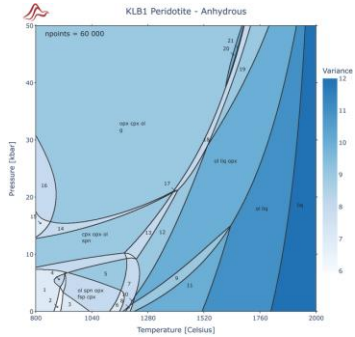


- Matlab app (Graphic user interface)
- Parallel point-wise minimization
- PT, PX, TX diagrams
- Contouring
- PT paths
- Trace element partitioning for mafic to ultramafic systems

E.G., GEODYNAMIC COUPLING



MAGEMin github: ComputationalThermodynamics



nriel@uni-mainz.de

ComputationalThermodynamics
Tools for thermodynamic computing
14 followers · Germany

Popular repositories

- MAGEMin** (Public)
The parallel Mineral Assemblage Gibbs Energy Minimization package
63 stars, 15 forks
- MAGEMin_C.jl** (Public)
Julia interface to the MAGEMin C package
10 stars, 4 forks
- MAGEMinApp.jl** (Public)
Graphical User Interface for MAGEMin, which runs in your web-browser.
5 stars, 2 forks
- SandBox**
Julia

MAGEMin_C.jl

CI passing DOI [10.5281/zenodo.11217861](https://doi.org/10.5281/zenodo.11217861)

Julia interface to the MAGEMin C package, which performs thermodynamic equilibrium calculations. See the [MAGEMin](#) page for more details on the package & how to use it.

Using the julia interface

First install julia. We recommend downloading the official binary from the [julia](#) webpage.

Next, install the `MAGEMin_C` package with:

```
julia> ]  
pkg> add MAGEMin_C
```



<https://github.com/ComputationalThermodynamics>

https://github.com/ComputationalThermodynamics/MAGEMin_C

<https://github.com/ComputationalThermodynamics/MAGEMinApp>

Required and optional software for the shortcourse



- <https://julialang.org/downloads/>
 - Allow to install and use MAGEMinApp and MAGEMin_C
- Visual Studio Code – Code editor <https://code.visualstudio.com/>
 - Allow Windows user to connect to Linux WSL and get the best performance for MAGEMin
- <https://inkscape.org/fr/>
 - Allow to open MAGEMinApp figure vector format (svg)



- **WINDOWS users**
- Recommended to install and use WSL (Windows sub-system for Linux)
- Connect to WSL and Linux Via VSCode (see next slides)
- MAGEMin performs twice faster on Linux/Mac than native Windows

Julia installation

Mac and Linux

Julia installation MAC LINUX

Install

Install the latest Julia version ([v1.10.5](#) August 27, 2024) by running this in your terminal:

```
$ curl -fsSL https://install.julialang.org | sh
```

COPY

For Windows instructions [click here](#)

Once installed `julia` will be available via the command line interface.

This will install the `Juliaup` installation manager, which will automatically install julia and help keep it up to date. The command `juliaup` is also installed. To install different julia versions see `juliaup --help`.

Please star us [on GitHub](#). If you use Julia in your research, please [cite us](#). If possible, do consider [sponsoring](#) us.

```
curl -fsSL https://install.julialang.org | sh
```

Julia installation

Windows

Julia installation WINDOWS

- Not the favoured way for performances, but faster to install and have MAGEMin up and running

Install julia

Install the latest Julia version ([v1.10.5](#) August 27, 2024) from the [Microsoft Store](#) by running this in the command prompt:

```
> winget install julia -s msstore
```

COPY

It looks like you're using Windows. For Linux and MacOS instructions [click here](#)

Once installed `julia` will be available via the command line interface.

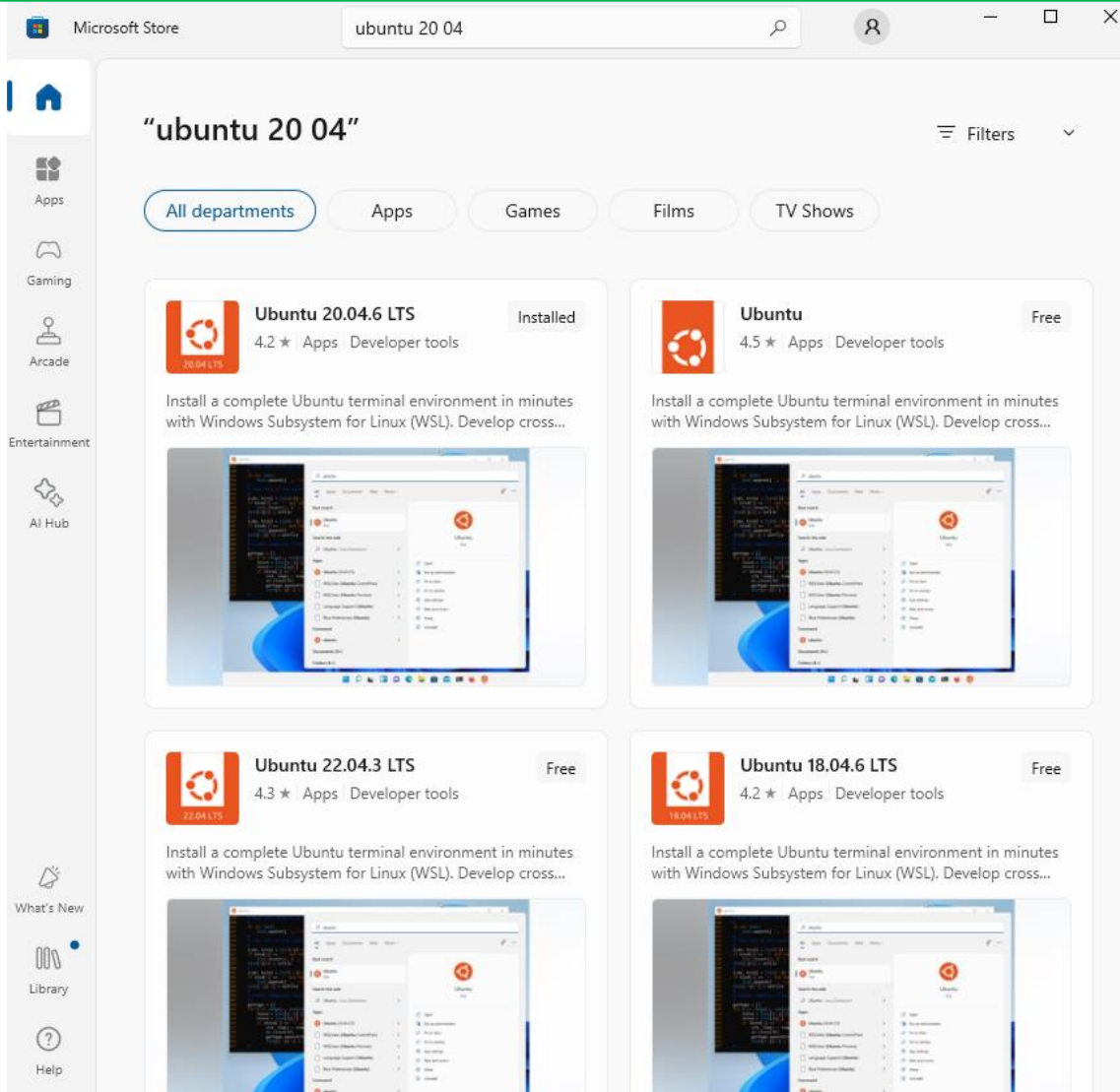
This will install the [Juliaup](#) installation manager, which will automatically install julia and help keep it up to date. The command `juliaup` is also installed. To install different julia versions see `juliaup --help`.

Please star us [on GitHub](#). If you use Julia in your research, please [cite us](#). If possible, do consider [sponsoring](#) us.

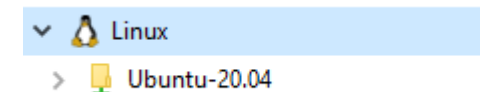
```
winget install julia -s msstore
```

MAGEMinApp installation (Windows with WSL)

- Favoured way, a bit longer to install but much faster

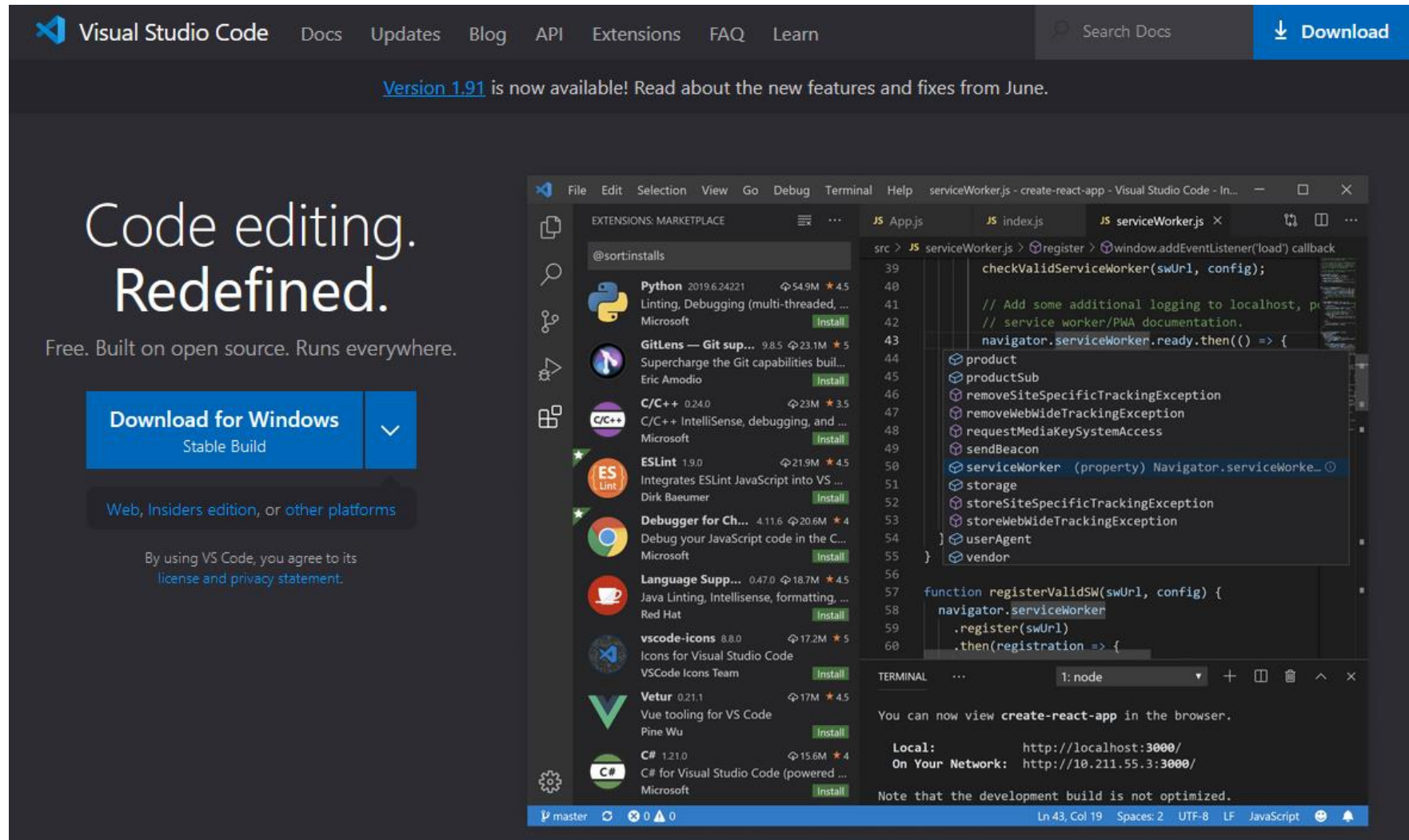


- Open Microsoft store (start-up menu)
 - Look for Ubuntu 20 04
 - Install Ubuntu 20 04
 - Restart computer
 - A terminal will open and ask for setting up a Linux username and password to your Ubuntu
-
- A new folder in the explorer should appear:



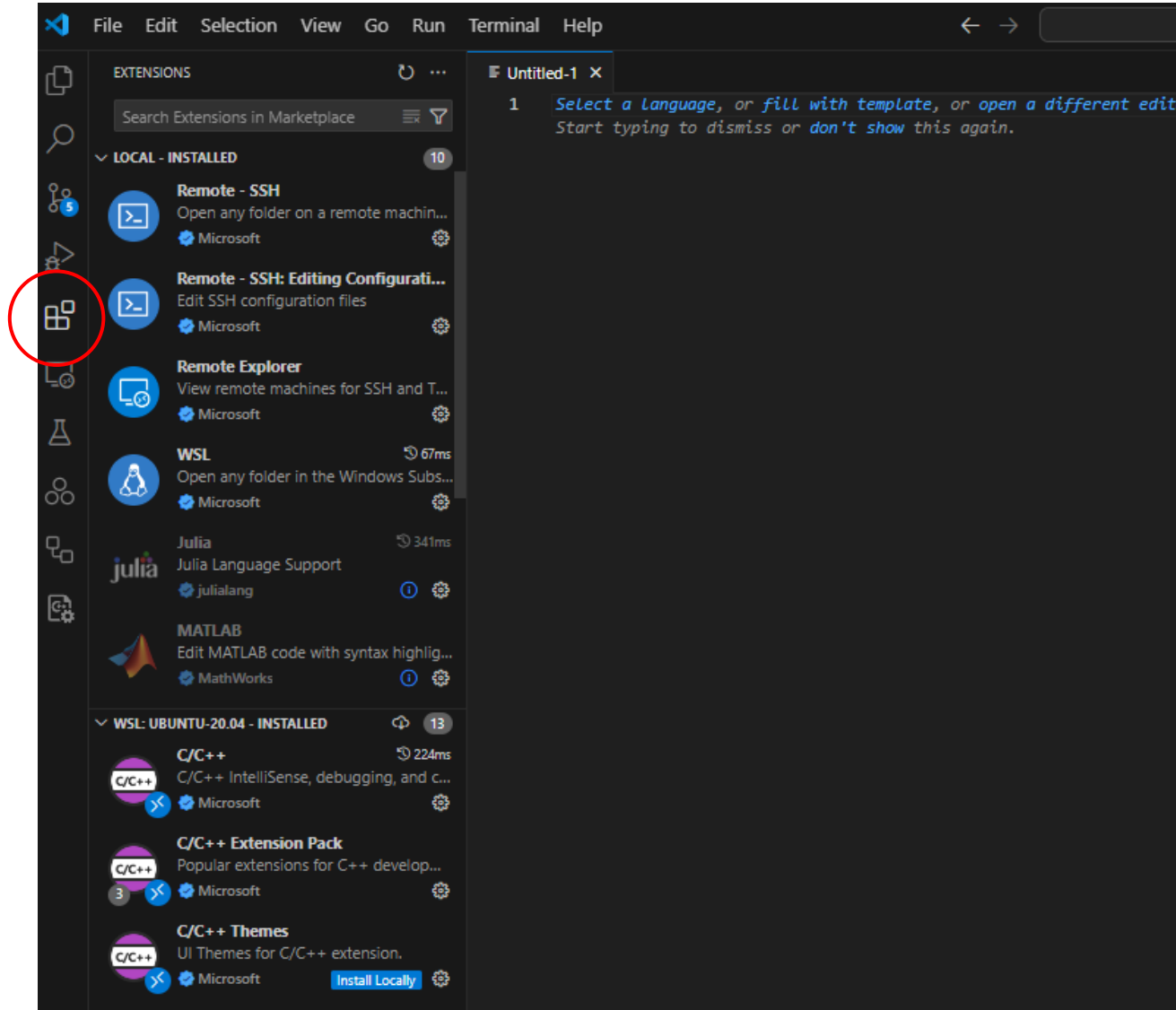
Visual Studio Code installation

- <https://code.visualstudio.com/>
- Download and install



Visual Studio Code installation

- Install WSL plugin



Visual Studio Code installation

- Install WSL plugin

The screenshot shows the Visual Studio Code interface with the Extensions Marketplace open. On the left, a list of extensions is shown, with 'WSL' by Microsoft at the top. The main panel displays the details for the 'WSL' extension (version 0.88.2). The extension is described as 'Open any folder in the Windows Subsystem for Linux (WSL) and take advantage of Visual Studio Code's full feature set.' The 'Uninstall' button is circled in red. Below the extension details, there is a section titled 'Visual Studio Code WSL' with a description of the extension's purpose and a list of frequently asked questions.

WSL v0.88.2
Microsoft microsoft.com | 27,999,641 | ★★★★★ (77)
Open any folder in the Windows Subsystem for Linux (WSL) and take advantage of Visual Studio Code's full feature set.

[Disable](#) [Uninstall](#) [Settings](#)

[DETAILS](#) [FEATURES](#) [CHANGELOG](#)

Visual Studio Code WSL

The **WSL extension** lets you use VS Code on Windows to build Linux applications that run on the [Windows Subsystem for Linux \(WSL\)](#). You get all the productivity of Windows while developing with Linux-based tools, runtimes, and utilities.

The **WSL extension** lets you use VS Code in WSL just as you would from Windows.

Why do I need the **WSL** extension?

Why WSL?

WSL lets you run a Linux environment -- including command-line tools and applications -- directly on Windows, without the overhead of a traditional virtual machine or dualboot setup. WSL especially helps web developers and those working with Bash and Linux-first tools (i.e. Ruby, Python) to use their toolchain on Windows and ensure consistency between development and production environments.

When you install a version of Linux on Windows, you're getting a full Linux environment. It's isolated from Windows- the UI is the terminal, and you can install tools, languages, and compilers into the Linux environment without modifying or disrupting your Windows installation.

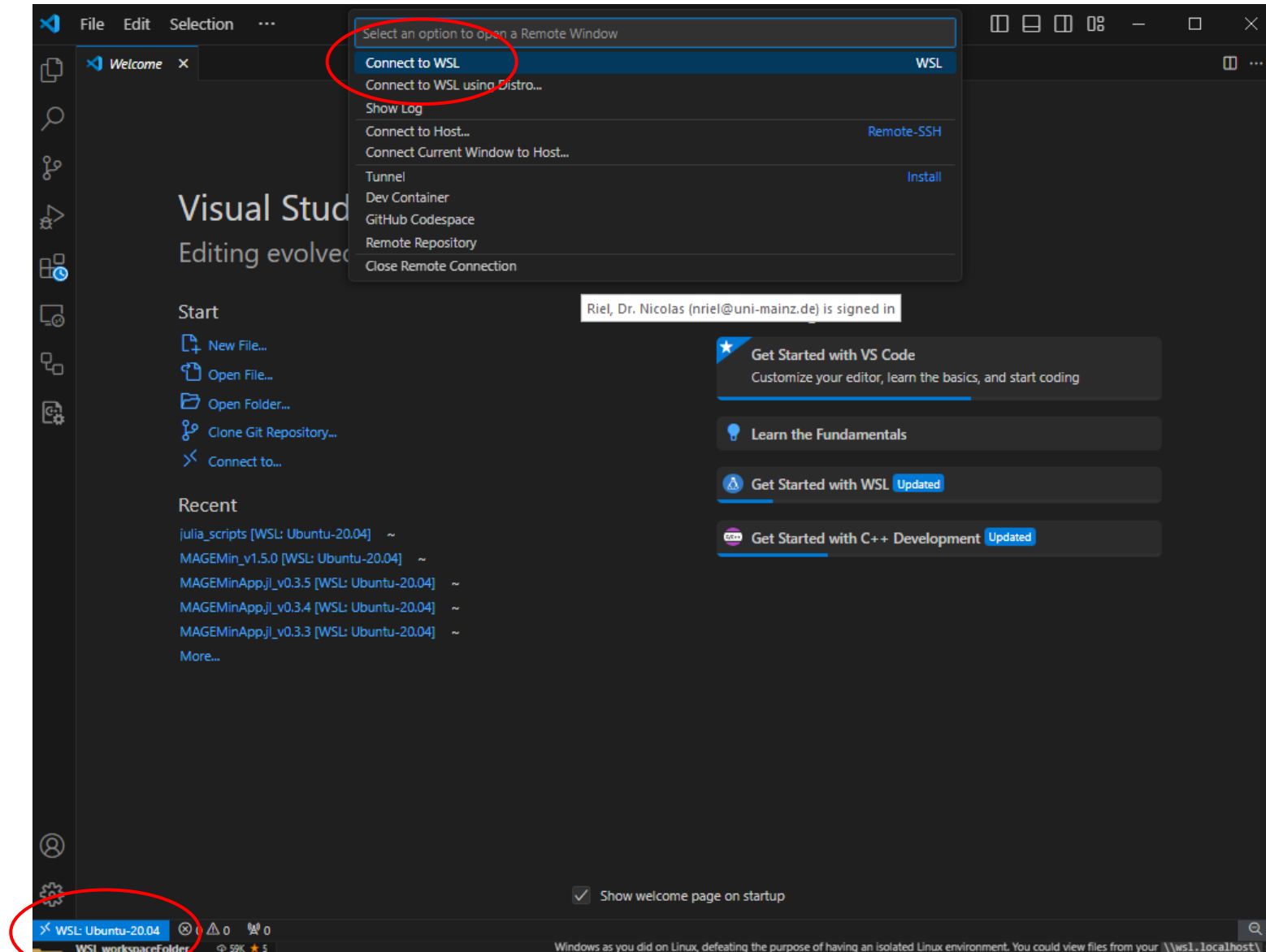
We recommend using WSL 2 as you will benefit from significant [performance advantages](#) over WSL 1.

Why the WSL extension in VS Code?

While you can edit files in Linux using Windows-based tools, you can't easily run or debug on Windows: you'd have to install all the same tools on Windows as you did on Linux, defeating the purpose of having an isolated Linux environment. You could view files from your `\\wsl.localhost` share, but you wouldn't have access to features such as autocomplete, debugging, or linting.

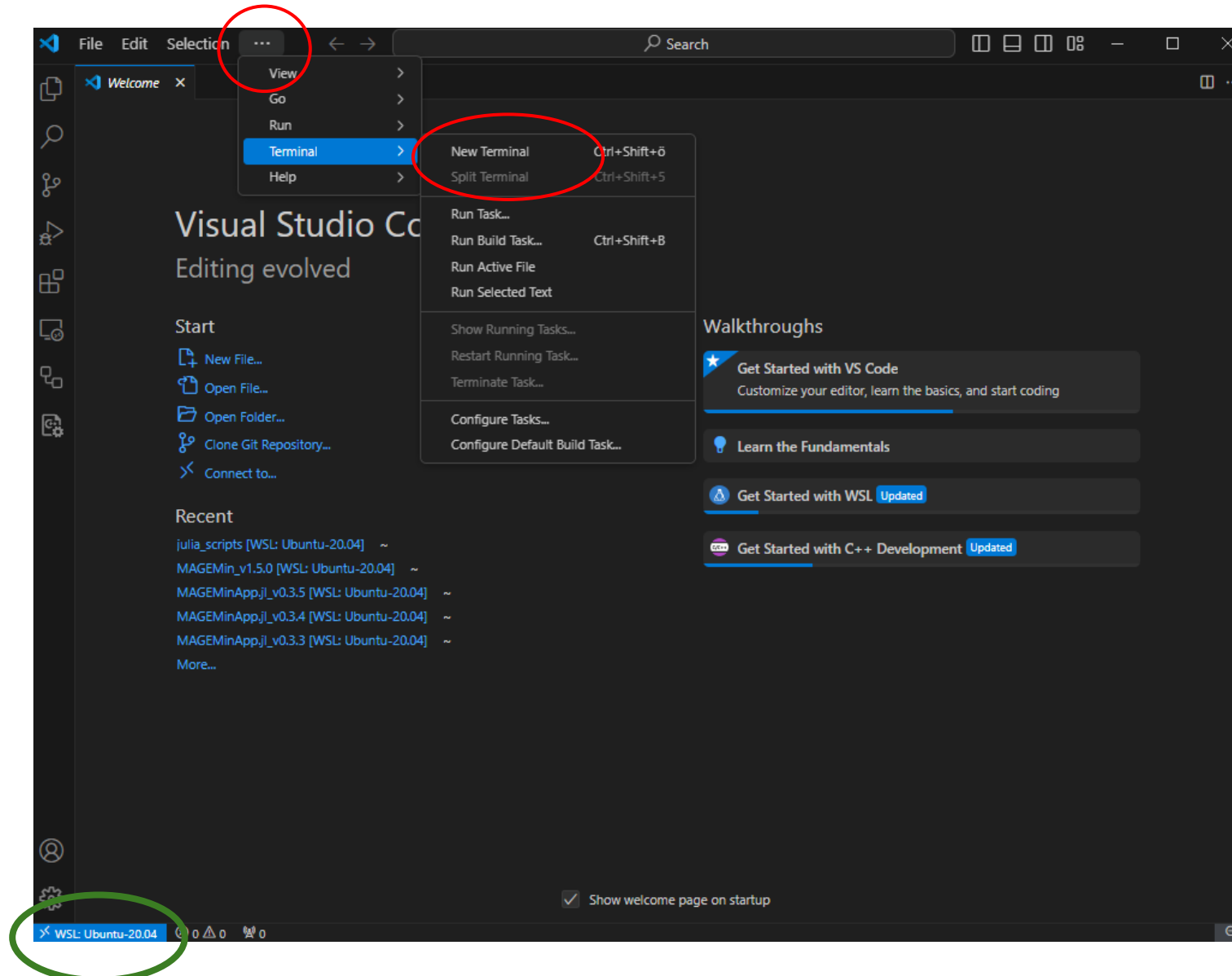
Visual Studio Code installation

- Connect vscode to WSL



Visual Studio Code installation

- Open a new terminal



Julia installation WSL (Windows)

- Get the command to download Julia:
`curl -fsSL https://install.julialang.org | sh`

<https://julialang.org/downloads/>



Download

Documentation

Learn

Blog

Community

Contribute

JSOC

Install julia

Install the latest Julia version (v1.10.4 June 4, 2024) by running this in your terminal:

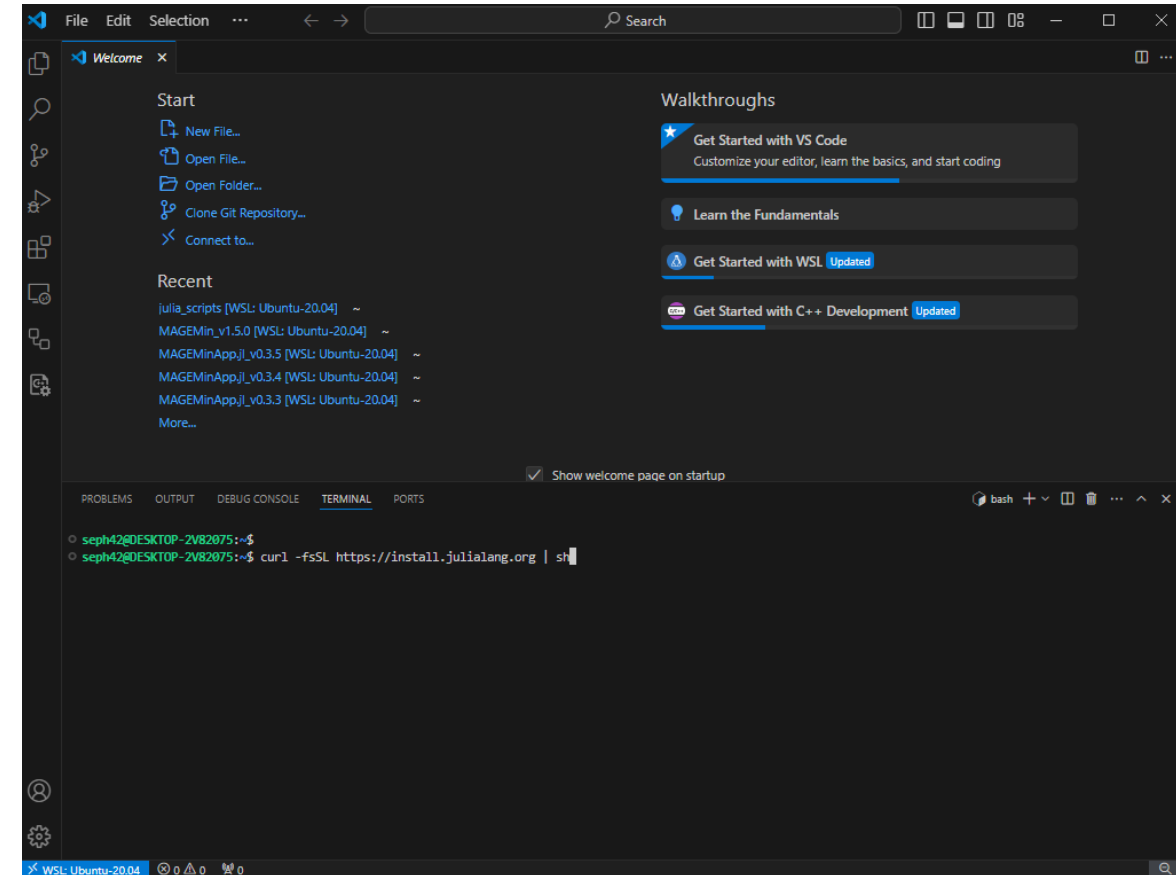
```
$ curl -fsSL https://install.julialang.org | sh
```

For Windows instructions [click here](#)

Once installed **julia** will be available via the command line interface.

This will install the **Juliaup** installation manager, which will automatically install julia and help keep it up to date. The command is installed. To install different julia versions see `juliaup --help`.

Please star us [on GitHub](#). If you use Julia in your research, please [cite us](#). If possible, do consider [sponsoring us](#).



Please do not use the version of "Julia" shipped by unix package managers

Many unix package managers ship broken and/or significantly out of date versions of Julia. Please use juliaup or download the c

- Execute the command in the terminal

MAGEMinApp & MAGEMin_C installation

Windows
Mac and Linux

Install MAGEMinApp

- Open a new terminal (vscode for those who use it, or terminal/powershell for others)
- Type:
julia -t 6 # where 6 is the number of core you want to use (depends on your machine, type
 # 'versioninfo()' to get more informations)

[illegible]

- In the terminal type ']', this will open the package manager
- Type 'add MAGEMinApp', this will download and install MAGEMinApp
- Once installed, quite the package manager by typing 'BACKSPACE' key

Launch MAGEMinApp

- In the Julia terminal, type:
‘using MAGEMinApp’
Then
‘App()’
- The following text will be displayed in the terminal:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
seph42@DESKTOP-2V82075:~$ julia -t 6

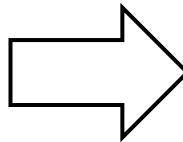
Documentation: https://docs.julialang.org
Type "?" for help, "]" for Pkg help.

Version 1.10.0 (2023-12-25)
Official https://julialang.org/ release

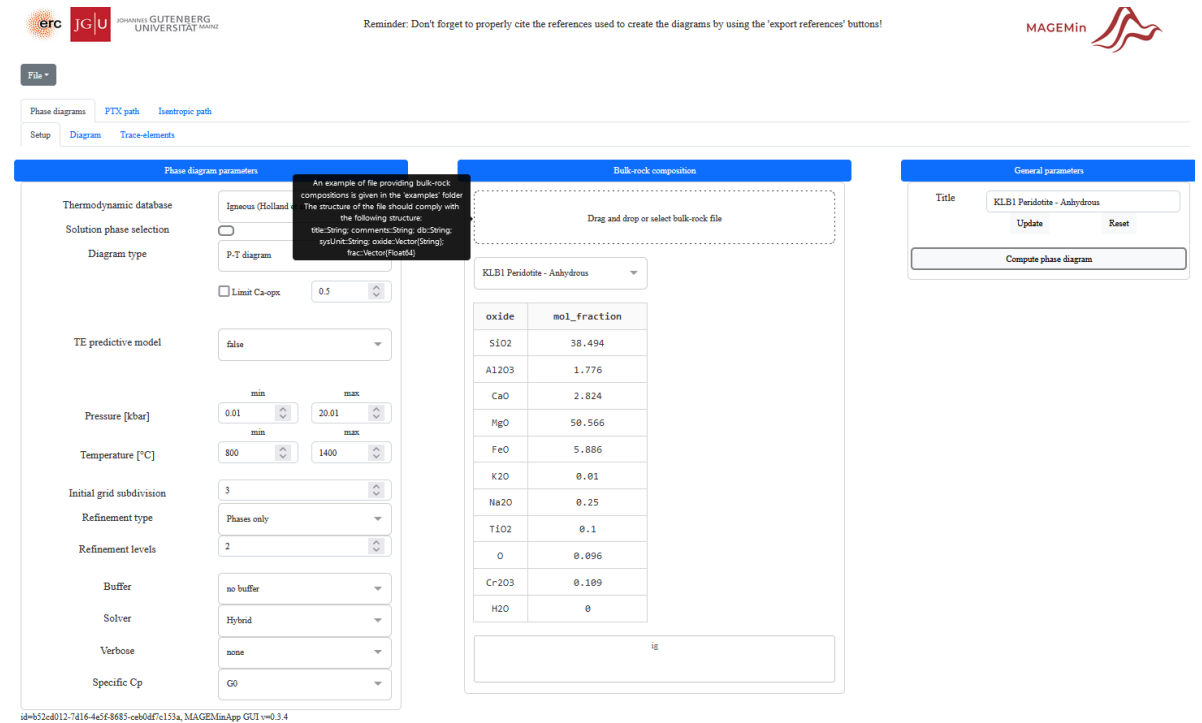
(julia) pkg> add MAGEMinApp
Resolving package versions...
No Changes to `~/julia/environments/v1.10/Project.toml`
No Changes to `~/julia/environments/v1.10/Manifest.toml`

julia> using MAGEMinApp
Using libMAGEMin.dylib from MAGEMin_jll

julia> App()
[ Info: Listening on: 127.0.0.1:8050, thread id: 1
```



- Copy and paste the address in your web-browser:



MAGEMin_C, the julia interface: installation



https://github.com/ComputationalThermodynamics/MAGEMin_C.jl

Install MAGEMin_C

```
julia> ] # opens the package
manager
pkg> add MAGEMin_C # MAGEMin_C
```

Load MAGEMin_C

```
julia> using MAGEMin_C # load MAGEMin_C
```

```
seph42@DESKTOP-2V82075:~$ julia
Documentation: https://docs.julialang.org
Type "?" for help, "]" for Pkg help.
Version 1.10.0 (2023-12-25)
Official https://julialang.org/ release

(@v1.10) pkg> add MAGEMin_C
Resolving package versions...
Updating `~/.julia/environments/v1.10/Project.toml`
[e5d170eb] + MAGEMin_C v1.5.5
No Changes to `~/.julia/environments/v1.10/Manifest.toml`

julia> using MAGEMin_C
Using libMAGEMin.dylib from MAGEMin_jll
```