

Softveri za rad sa konačnim elementima

Softveri za rad sa konačnim elementima

Koristićemo Firedrake za implementaciju metode konačnih elemenata. Da bismo kompletan proces rešavanja problema i visuelizacije mogli realizovati koristićemo sledeće softvere:

- **Python** - podrazumevani programski jezik
- **VS Code** - editor za pisanje koda (nije neophodan jer može sve u terminalu, ali je udobniji rad u njemu)
- **Firedrake** - za metod konačnih elemenata
- **Gmsh** - alat za meširanje, tj. generisanje mreže koja nije jednostavna (Firedrake ima ugrađene samo proste geometrije segment, kvadrat, kvadar)
- **ParaView** - alat za vizuelizaciju rezultata (Firedrake može samo 2D grafike).

Instalacija na Linux-u

Instalacija za Linux

Podrazumevam da radimo na Linux-u - distribucija Ubuntu, pa uputstvo za instalaciju navodim za njega. Python je sastavni deo svake distribucije Linux-a, pa ga ne moramo posebno instalirati. Otvoriti komandni prozor (Ctrl+Alt+T) i redom kopirati sledeće komande:

1. Ažuriranje sistema

```
sudo apt update && sudo apt upgrade -y
```

2. Instalacija VS Coda

```
sudo snap install code --classic  
code --install-extension ms-python.python  
code --install-extension ms-toolsai.jupyter  
code --install-extension ms-python.vscode-pylance
```

3. Priprema za Firedrake (sistemske paketi + PETSc)

```
# napravi radni direktorijum (po želji)  
mkdir -p ~/firedrake-work && cd ~/firedrake-work
```

```
# preuzmi firedrake-configure  
curl -O  
https://raw.githubusercontent.com/firedrakeproject/firedrake/main/scripts/firedrake-configure
```

4. Sistemske pakete (Ubuntu)

```
sudo apt install $(python3 firedrake-configure --show-system-packages) -y
```

5. Instalacija PETSc (verzija tačno koju Firedrake traži)

```
# kloniraj odgovarajuću verziju PETSc-a  
git clone --branch $(python3 firedrake-configure --show-petsc-version)  
https://gitlab.com/petsc/petsc.git  
cd petsc
```

```
# konfiguriši sa opcijama koje izbacuje firedrake-configure  
python3 ../_firedrake-configure --show-petsc-configure-options | xargs -L1 ./configure
```

```
# kompajliraj (ume da potraje)  
make PETSC_DIR=$PWD PETSC_ARCH=arch-firedrake-default all
```

```
# opcionalno: kratka provera  
make check
```

```
# vrati se u parent dir  
cd ..
```

6. Virtuelno okruženje + instalacija Firedrake

```
#instalacija python3-venv
sudo apt update
sudo apt install python3-venv -y

# pravljenje i aktivacija virtuelnog okruzenja
python3 -m venv venv-firedrake
source venv-firedrake/bin/activate

#Postavi promenljive okruženja (PETSc i ostalo)
export $(python3 firedrake-configure --show-env)

#instaliraj Firedrake (traje...)
pip install --no-binary h5py 'firedrake[check]'
```

7. Testiranje instalacije - ako sve prođe u redu je Firedrake-check

8. Instalacija Gmsh

```
sudo apt update
sudo apt install -y gmsh
gmsh -v
pip install gmsh
```

9. Instalacija ParaView

Otvoriti na stranici Download ParaView i preuzmi [Linux x86_64](#) arhivu. Sačuvaj u ~/Downloads. Raspakuj i pokreni:

```
mkdir -p ~/opt
cd ~/Downloads
# zameni naziv fajla stvarnim imenom preuzetog fajla (npr.
ParaView-6.0.0-MPI-Linux-x86_64.tar.xz)
tar -xf ParaView-6.0.0*.tar.* -C ~/opt
cd ~/opt/ParaView-6.0.0*/bin

# pokretanje
QT_QPA_PLATFORM=xcb ~/opt/ParaView-6.0.0*/bin/paraview
```

(Opciono) Napraviti alias da se zove samo „paraview“

Dodaj u `~/.bashrc`:

```
echo "alias paraview='QT_QPA_PLATFORM=xcb  
~/opt/ParaView-6.0.0*/bin/paraview'" >> ~/.bashrc
```

```
source ~/.bashrc
```

Sada se pokreće samo sa

```
paraview
```

Ovim je instalacija svih softvera gotova. Preći na test primer.

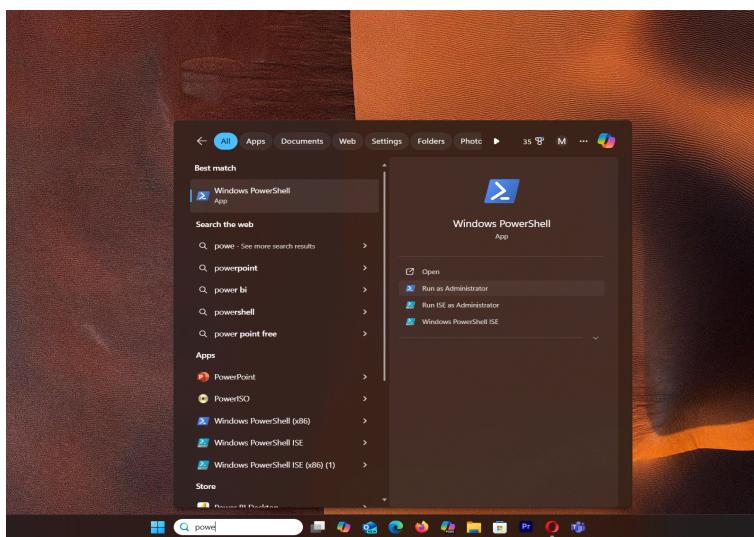
Instalacija na Windows-u

Firedrake biblioteku na Windows operativnom sistemu možemo koristiti preko WSL-a (Windows Subsystem for Linux). Ovaj postupak je zvanično podržan od strane Firedrake tima i obezbeđuje stabilno i pouzdano razvojno okruženje.

Važna napomena: Ne koristiti Anacondu / conda okruženja! Firedrake ne radi pravilno u conda okruženjima zbog konflikata sa MPI, PETSc-om i drugim C/C++ bibliotekama koje Firedrake kompajlira prilikom instalacije.

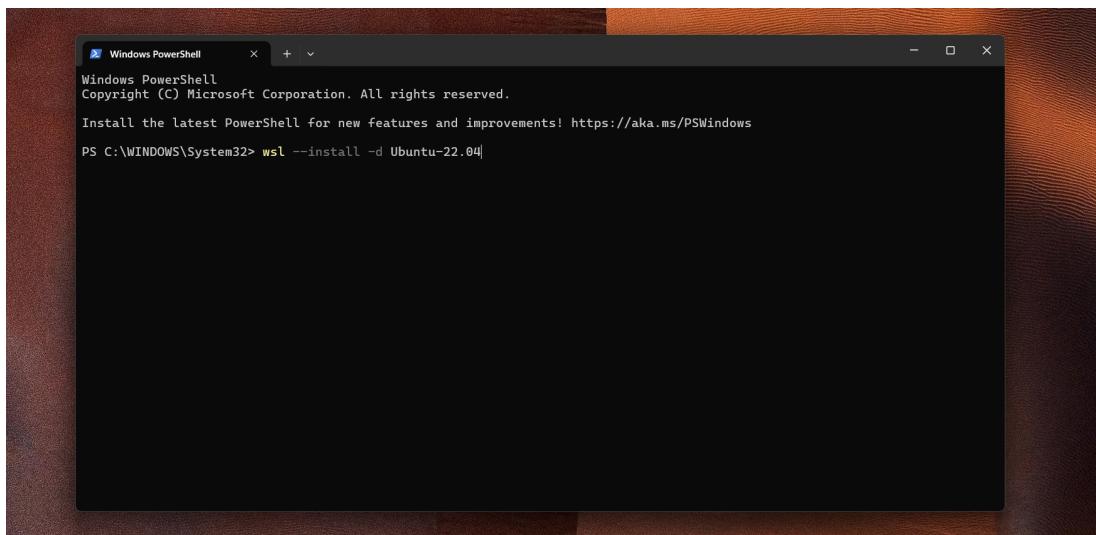
1. Instalacija WSL-a (Windows Subsystem for Linux)

Otvoriti Windows Power Shell kao administrator.

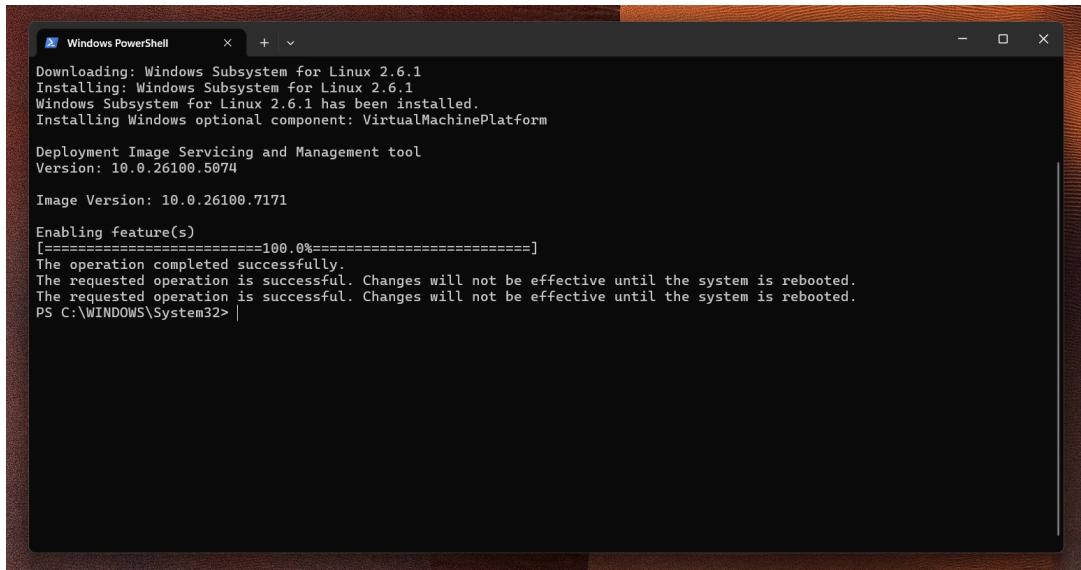


U Power Shell okruženju pokrenuti komandu:

```
wsl --install -d Ubuntu-22.04
```



Dozvoliti aplikaciji da izvrši instalaciju. Ako je instalacija prošla dobro, pojaviće se sledeće poruke u Power Shell-u:



```
Windows PowerShell

Downloading: Windows Subsystem for Linux 2.6.1
Installing: Windows Subsystem for Linux 2.6.1
Windows Subsystem for Linux 2.6.1 has been installed.
Installing Windows optional component: VirtualMachinePlatform

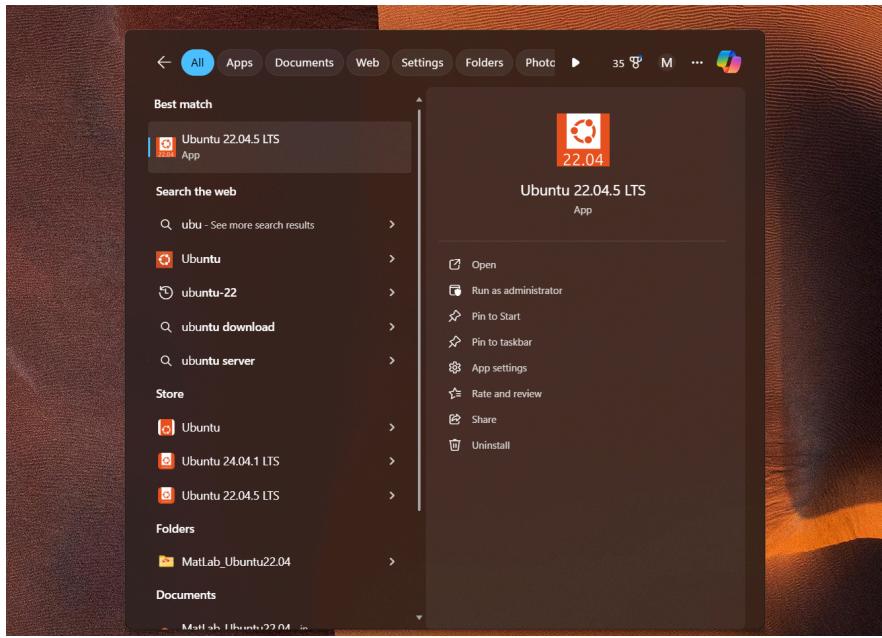
Deployment Image Servicing and Management tool
Version: 10.0.26100.5074

Image Version: 10.0.26100.7171

Enabling feature(s)
[=====100.0%=====]
The operation completed successfully.
The requested operation is successful. Changes will not be effective until the system is rebooted.
The requested operation is successful. Changes will not be effective until the system is rebooted.
PS C:\WINDOWS\System32> |
```

Restartovati računar.

Nakon restartovanja, Ubuntu je dostupan preko Start-a. U slučaju da ga nema, ponoviti prethodni postupak instalacije.



Prilikom prvog pokretanja Ubuntu-a, potrebno je uneti korisničko ime i lozinku, npr:

Username: **firedrake_ubuntu**

Lozinka: **stagod**

Lozinka će biti potrebna pri svakom pozivu sude komande. Ovim smo završili postavku Ubuntu-a. Možemo preći na instalaciju Firedrake-a.

2. Instalacija Firedrake-a

Otvorimo Ubuntu.

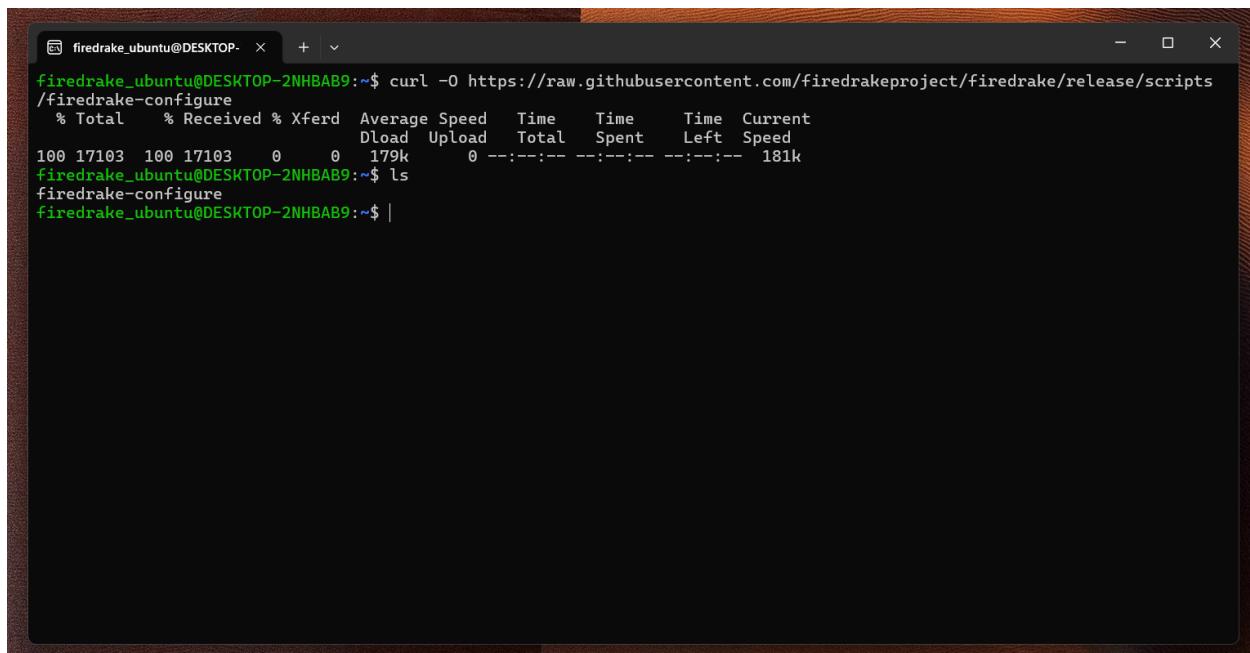
Da bi proces instalacije pojednostavio, tim Firedrake-a je obezbedio pomoćnu skriptu pod nazivom **firedrake-configure**. Ova skripta se preuzme komandom:

```
curl -O  
https://raw.githubusercontent.com/firedrakeproject/firedrake/release/scripts/firedrake-configure
```

Napomena 1 : **firedrake-configure ne instalira Firedrake**. Ona je samo pomoćna skripta koja generiše konfiguracione opcije koje Firedrake zahteva u različitim koracima tokom instalacije.

Napomena 2: ako se neka od navedenih komandi za instalaciju Firedrake-a ne izvršava ili prijavljuje grešku, onda komande kopirati direktno sa Firedrake-ovog sajta:

<https://www.firedrakeproject.org/install.html#installing-firedrake-using-pip>



A screenshot of a terminal window titled "firedrake_ubuntu@DESKTOP-2NHBAB9:~\$". The window shows the command "curl -O https://raw.githubusercontent.com/firedrakeproject/firedrake/release/scripts/firedrake-configure" being run. The output of the curl command shows a progress bar and statistics: Total: 100, Received: 17103, Xferd: 17103, Average Speed: 179k, Time: 0, Time: --:--:--, Time: --:--:--, Current Speed: 181k. After the download completes, the user runs "ls" to list the files in the current directory, which shows "firedrake-configure".

Ako se **firedrake-configure** nalazi u trenutnom folderu (možemo proveriti komandom **ls**), ažurirajmo sistem i instalirajmo potrebne alate pomoću komande:

```
sudo apt install $(python3 firedrake-configure --show-system-packages)
```

Ovde instalirani paketi predstavljaju kombinaciju sistemskih zavisnosti, kao što su C kompjajler, BLAS i MPI, i „spoljnih paketa“ koje koristi PETSc, poput MUMPS-a i HDF5-a.

Da bi Firedrake radio kako treba, neophodno je instalirati **PETSc** sa **tačno određenim skupom spoljnih paketa**. **PETSc** (Portable, Extensible Toolkit for Scientific Computation) je velika i veoma optimizovana biblioteka za **numeričko rešavanje parcijalnih diferencijalnih jednačina**, posebno na:

- velikim mrežama (high-resolution mesh),
- paralelnim računarima,
- klasterima,
- HPC (High-Performance Computing) sistemima.

PETSc je napisan u C-u, ali ima interfejs za Python (preko petsc4py), C++ i Fortran.

Da bismo instalirali PETSc, potrebno je uraditi sledeće korake:

1. Klonirati PETSc repozitorijum, čekirajući odgovarajuću verziju (u firedrake-configure su info o verziji):

```
git clone --branch $(python3 firedrake-configure --show-petsc-version)
https://gitlab.com/petsc/petsc.git
cd petsc
```

2. Pokrenuti PETSc configure, prosleđujući mu parametre koje generiše **firedrake-configure**:

```
python3 ../firedrake-configure --show-petsc-configure-options | xargs
-L1 ./configure
```

3. Iskompajlirati PETSc pokretanjem **make** komande:

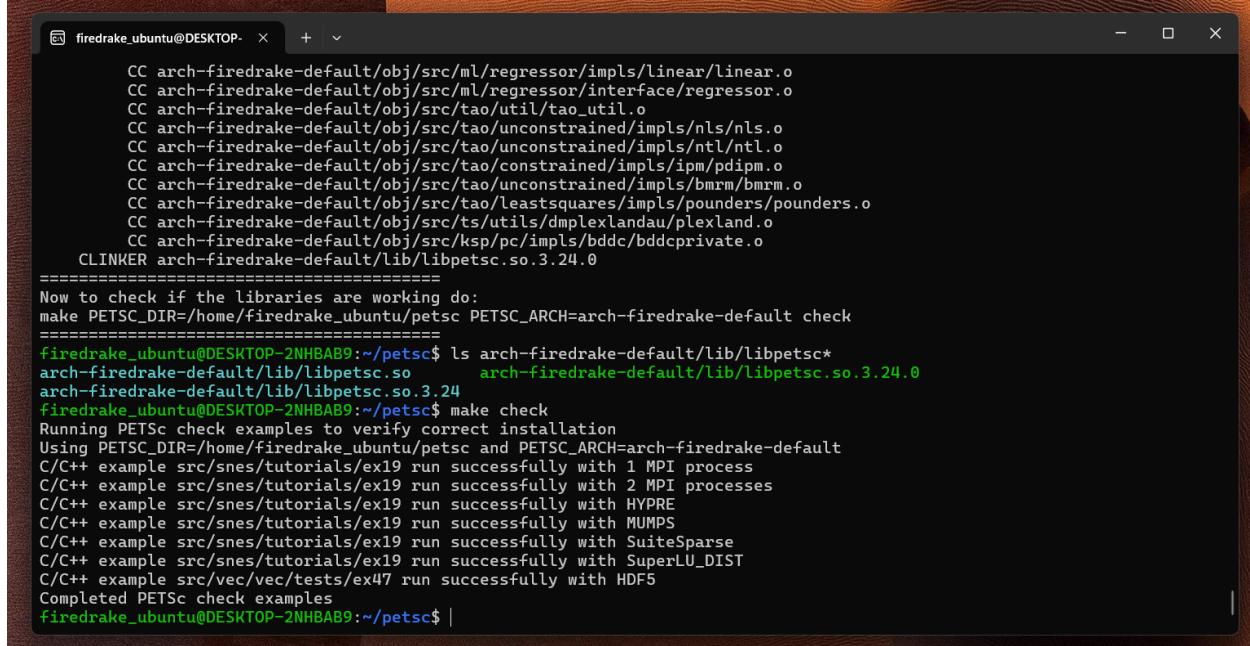
```
make PETSC_DIR=/path/to/petsc PETSC_ARCH=arch-firedrake-default all
```

4. Testiramo instalaciju:

```
make check
```

Pokrenuće se nekoliko primera (ex19, ex47...) sa različitim solverima (HYPRE, MUMPS, SuiteSparse...). Ako neki externi paket nije baš savršeno uklopljen, neki test može da padne. To **ne znači automatski** da

PETSc ne radi. Samo ako se pojave poruke tipa Error, to ne smemo da ignorišemo. Tada se koraci 1,2 i 3 moraju ponoviti.



```
firedrake_ubuntu@DESKTOP-2NHBAB9:~/petsc$ CC arch=firedrake-default/obj/src/ml/regressor/impls/linear/linear.o
CC arch=firedrake-default/obj/src/ml/regressor/interface/regressor.o
CC arch=firedrake-default/obj/src/tao/util/tao_util.o
CC arch=firedrake-default/obj/src/tao/unconstrained/impls/nls/nls.o
CC arch=firedrake-default/obj/src/tao/unconstrained/impls/ntl/ntl.o
CC arch=firedrake-default/obj/src/tao/constrained/impls/ipm/pdipm.o
CC arch=firedrake-default/obj/src/tao/unconstrained/impls/bmrm/bmrm.o
CC arch=firedrake-default/obj/src/tao/leastsquares/impls/pounders/pounders.o
CC arch=firedrake-default/obj/src/ts/utils/dmplexlandau/plexlandau.o
CC arch=firedrake-default/obj/src/ksp/pc/impls/bddc/bddcprivate.o
CLINKER arch=firedrake-default/lib/libpetsc.so.3.24.0
=====
Now to check if the libraries are working do:
make PETSC_DIR=/home/firedrake_ubuntu/petsc PETSC_ARCH=arch=firedrake-default check
=====
firedrake_ubuntu@DESKTOP-2NHBAB9:~/petsc$ ls arch=firedrake-default/lib/libpetsc*
arch=firedrake-default/lib/libpetsc.so      arch=firedrake-default/lib/libpetsc.so.3.24.0
arch=firedrake-default/lib/libpetsc.so.3.24
firedrake_ubuntu@DESKTOP-2NHBAB9:~/petsc$ make check
Running PETSc check examples to verify correct installation
Using PETSC_DIR=/home/firedrake_ubuntu/petsc and PETSC_ARCH=arch=firedrake-default
C/C++ example src/snes/tutorials/ex19 run successfully with 1 MPI process
C/C++ example src/snes/tutorials/ex19 run successfully with 2 MPI processes
C/C++ example src/snes/tutorials/ex19 run successfully with HYPRE
C/C++ example src/snes/tutorials/ex19 run successfully with MUMPS
C/C++ example src/snes/tutorials/ex19 run successfully with SuiteSparse
C/C++ example src/snes/tutorials/ex19 run successfully with SuperLU_DIST
C/C++ example src/vec/vec/tests/ex47 run successfully with HDF5
Completed PETSc check examples
firedrake_ubuntu@DESKTOP-2NHBAB9:~/petsc$ |
```

Ako je sve prošlo dobro, vratimo se u folder iznad sa

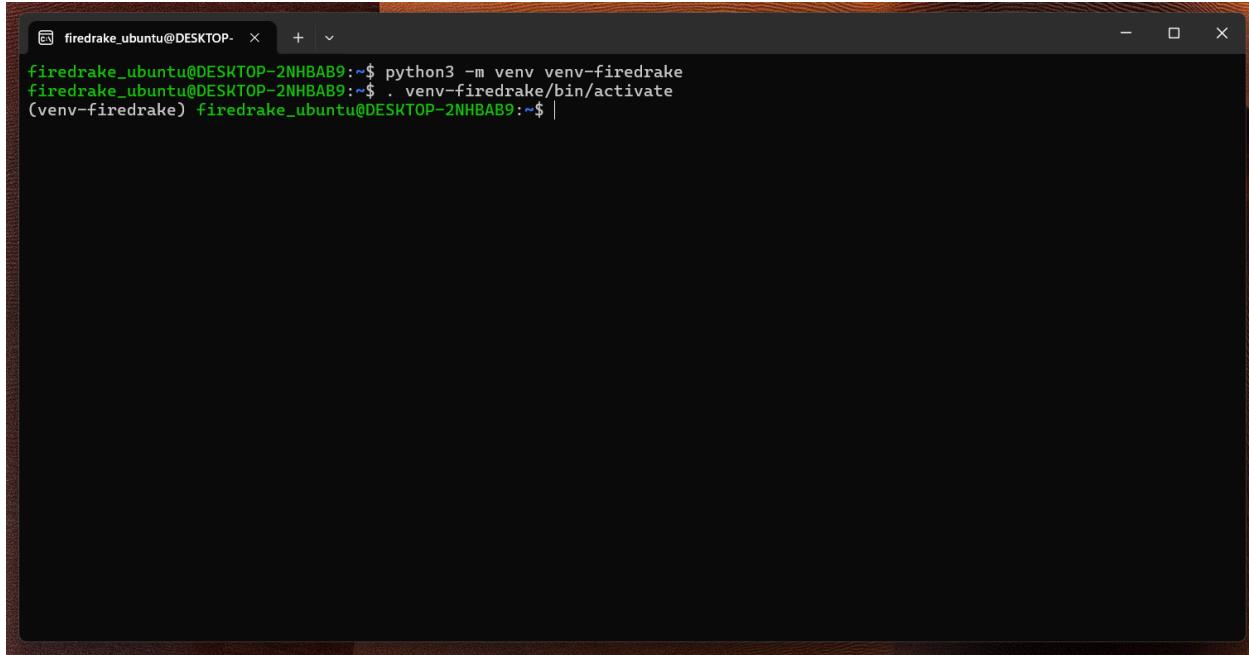
```
cd ..
```

Konačno prelazimo na instalaciju Firedrake-a.

1. Prvo napravimo virtuelno okruženje

```
python3 -m venv venv-firedrake
i aktivirajmo ga
. venv-firedrake/bin/activate
```

Ovo nije obavezno, ali se veoma preporučuje — na taj način čuvate svoj glavni Python urednim i ne mešate ga sa paketima potrebnim za Firedrake.



```
firedrake_ubuntu@DESKTOP-2NHBAB9:~$ python3 -m venv venv-firedrake
firedrake_ubuntu@DESKTOP-2NHBAB9:~$ . venv-firedrake/bin/activate
(venv-firedrake) firedrake_ubuntu@DESKTOP-2NHBAB9:~$ |
```

2. Očistimo pip keš:

```
pip cache purge
```

I ovo je opciono, ali se snažno preporučuje, jer neki keširani pip paketi mogu biti linkovani sa starim ili nedostajućim bibliotekama i zato mogu da pokvare instalaciju.

3. Podesite neophodne promenljive okruženja

Ovo se može uraditi pomoću `firedrake-configure`:

```
export $(python3 firedrake-configure --show-env)
```

Ova komanda će, između ostalog, postaviti sledeće promenljive:

```
CC=mpicc
CXX=mpicxx
PETSC_DIR=/path/to/petsc
PETSC_ARCH=arch-firedrake-{default,complex}
HDF5_MPI=ON
```

Ova komanda radi samo ako smo u pravom početnom direktorijumu.

Podrazumeva se da je PETSc kloniran u poddirektorijum trenutnog direktorijuma

(odnosno `<trenutni_dir>/petsc`). Ako ste tačno pratili uputstva do ovog trenutka, to bi već trebalo da bude slučaj.

4. Instalirajte Firedrake:

```
pip install --no-binary h5py 'firedrake[check, vtk]'
```

Iako nije strogo potrebno da instalirate Firedrake-ove opcione zavisnosti pomoću [\[check\]](#), preporučuje se zato što vam omogućava da proverite da je instalacija bila uspešna (videti ispod). Dodajemo i opciju vtk, kako bismo imali VTK (Visualization Toolkit) podršku.

Instalacija može potrajati 10–60 minuta u zavisnosti od brzine računara.

Firedrake je sada instaliran i spremam za korišćenje!

Provera instalacije

Preporučujemo da nakon instalacije pokrenete nekoliko jednostavnih testova kako biste proverili da je Firedrake potpuno funkcionalan.

Da biste to uradili, posle instalacije pokrenite:

firedrake-check

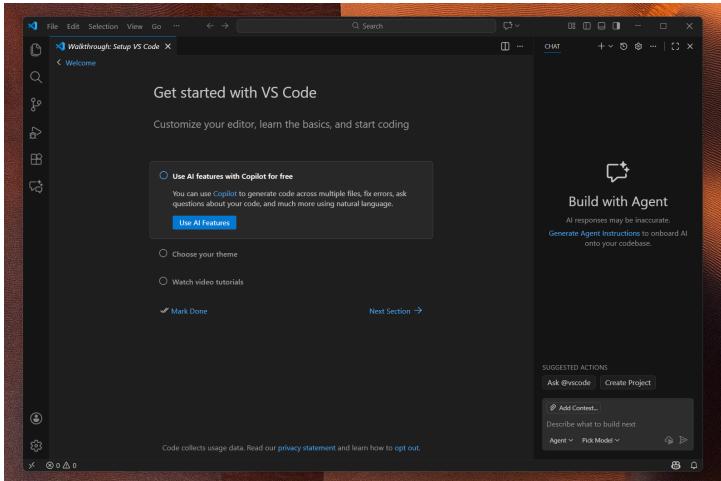
Ova komanda pokreće nekoliko jedinica testova koji proveravaju veliki deo funkcionalnosti biblioteke. Testovi bi trebalo da traju minut ili manje.

```
firedrake ubuntu@DESKTOP-  +  -  x
Requirement already satisfied: MarkupSafe>=0.9.2 in ./venv-firedrake/lib/python3.10/site-packages (from mako->loopy>2024.1->firedrake[check,vtk]) (3.0.3)
Collecting six>=1.5
  Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
Building wheels for collected packages: firedrake
  Building wheel for firedrake (pyproject.toml) ... done
    Created wheel for firedrake: filename=firedrake-2025.10.2-cp310-cp310-linux_x86_64.whl size=5188339 sha256=28abc146d9658ff0eb075c95fab5b5e1a00fdf4a5b8f26cccc6095a45db0d5048
      Stored in directory: /home/firedrake_ubuntu/.cache/pip/wheels/ce/dd/00/6c3f71d45d804e4d97cf3def29fb3e9e6b2d2272b0a4dd9fa0
Successfully built firedrake
Installing collected packages: six, pyparsing, pillow, kiwisolver, fonttools, cycler, contourpy, python-dateutil, matplotlib, vtk, firedrake
Successfully installed contourpy-1.3.2 cycler-0.12.1 firedrake-2025.10.2 fonttools-4.60.1 kiwisolver-1.4.9 matplotlib-3.10.7 pillow-12.0.0 pyparsing-3.2.5 python-dateutil-2.9.0.post0 six-1.17.0 vtk-9.5.2
(venv-firedrake) firedrake_ubuntu@DESKTOP-2NHBAB9:~$ firedrake-check
  Running serial smoke tests
.....
[100%]
32 passed, 3 deselected in 7.70s
  Tests passed
  Running parallel smoke tests (nprocs=3)
.....
[100%]
0%
2 passed in 1.45s
[100%]
2 passed in 1.45s
[100%]
2 passed in 1.45s
  Tests passed
(venv-firedrake) firedrake_ubuntu@DESKTOP-2NHBAB9:~$ |
```

Instaliranje editora - VS Code

Za editor se preporučuje Visual Studio Code. Lako se instalira - otići na web stranicu <https://code.visualstudio.com/docs/?dv=win64user>

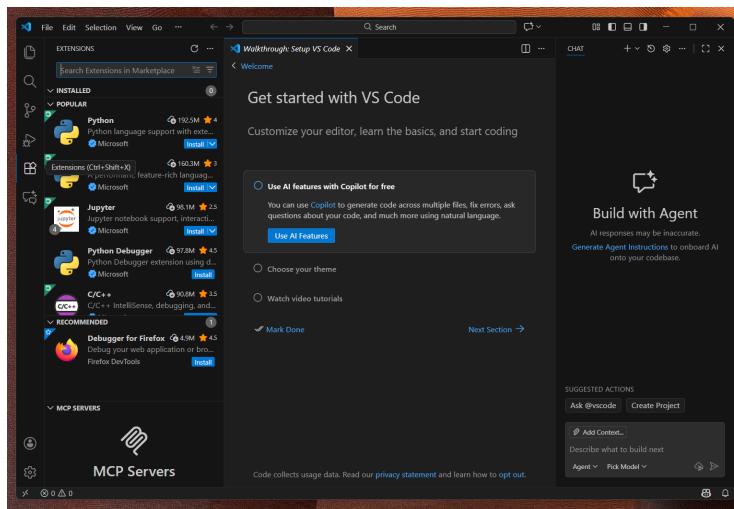
Skinuti exe fajl i instalirati. Ovako izgleda prvom pokretanju:



Dodaj ekstenzije u VS Code

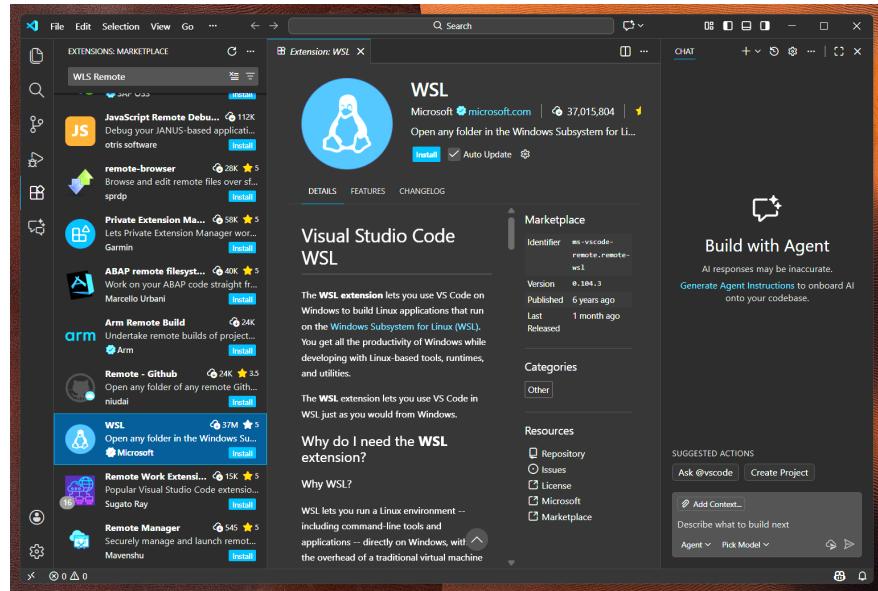
Pokreni VS Code u Windowsu, pa:

1. Sa leve strane klikni na ikonicu **Extensions** (četvrtasti blokić).

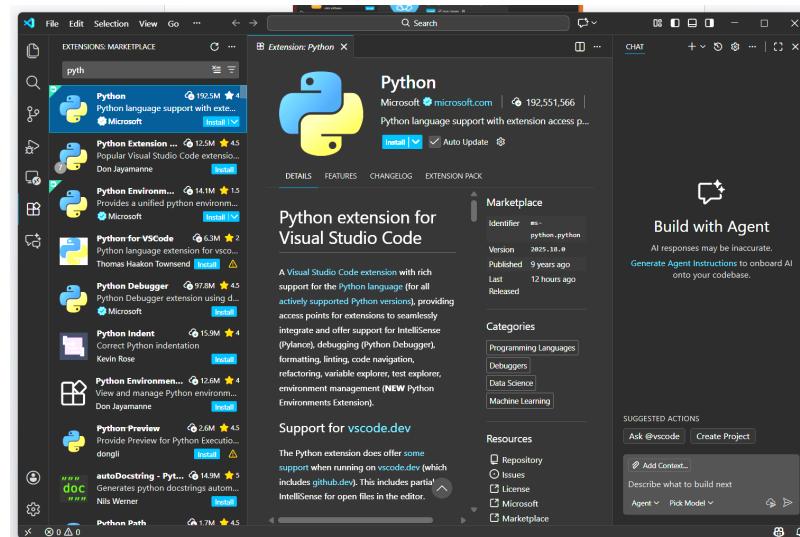


2. Instalirati:

- **WSL (Remote – WSL)**



- **Python (Microsoft-ova ekstenzija)**



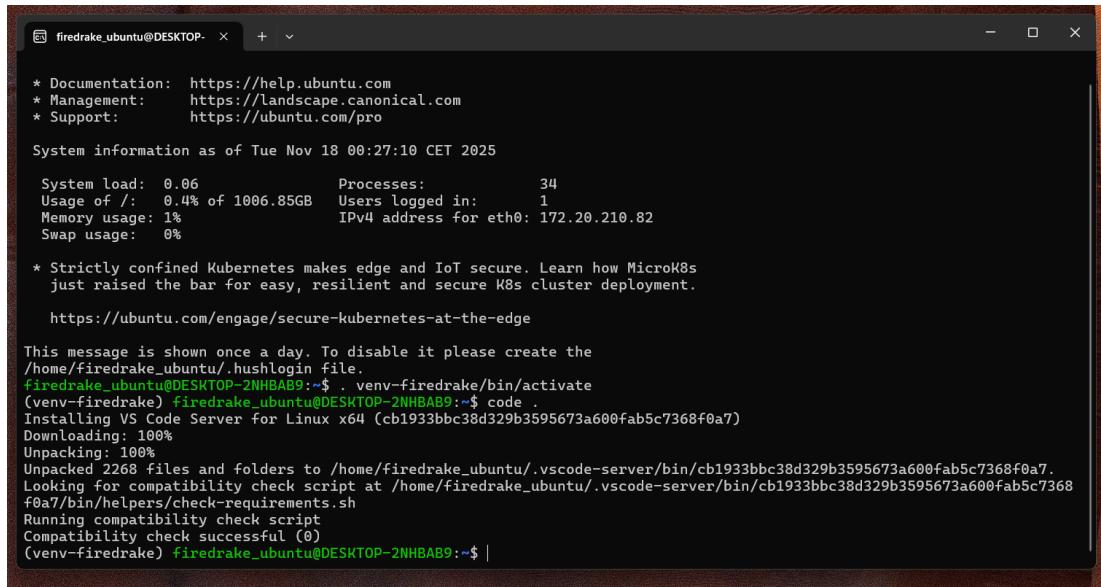
Povezivanje VS Code-a sa Ubuntu WSL-om

Najlakši način je **iz Ubuntu terminala**.

U Ubuntu terminalu:

```
cd ~
```

code .



```
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/pro

System information as of Tue Nov 18 00:27:10 CET 2025

System load: 0.06      Processes: 34
Usage of /: 0.4% of 1006.85GB  Users logged in: 1
Memory usage: 1%      IPv4 address for eth0: 172.20.210.82
Swap usage: 0%

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s just raised the bar for easy, resilient and secure K8s cluster deployment.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

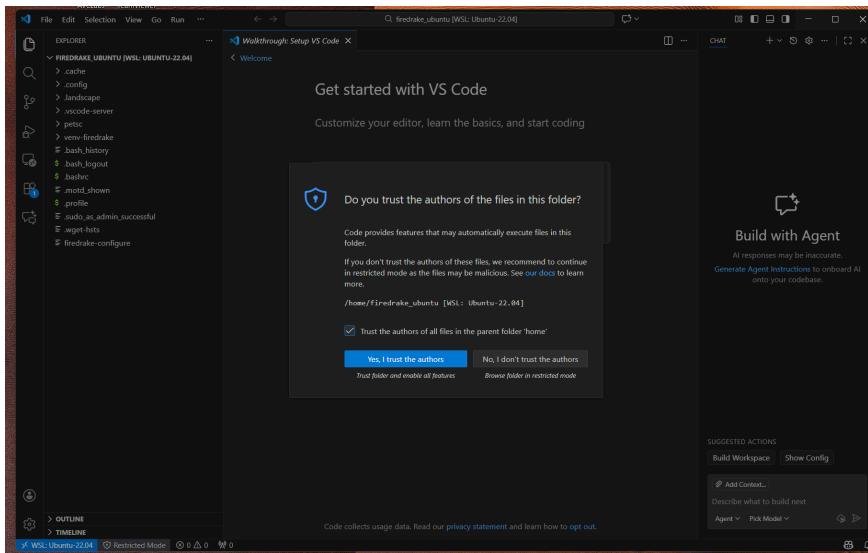
This message is shown once a day. To disable it please create the /home/firedrake_ubuntu/.hushlogin file.

firedrake_ubuntu@DESKTOP-2NHBAB9:~$ . venv-firedrake/bin/activate
(venv-firedrake) firedrake_ubuntu@DESKTOP-2NHBAB9:~$ code .
Installing VS Code Server for Linux x64 (cb1933bbc38d329b3595673a600fab5c7368f0a7)
Downloading: 100%
Unpacking: 100%
Unpacked 2268 files and folders to /home/firedrake_ubuntu/.vscode-server/bin/cb1933bbc38d329b3595673a600fab5c7368f0a7.
Looking for compatibility check script at /home/firedrake_ubuntu/.vscode-server/bin/cb1933bbc38d329b3595673a600fab5c7368f0a7/bin/helpers/check-requirements.sh
Running compatibility check script
Compatibility check successful (0)
(venv-firedrake) firedrake_ubuntu@DESKTOP-2NHBAB9:~$ |
```

Ako je sve OK:

- otvorice se VS Code prozor
- u donjem levom uglu ce pisati “**WSL: Ubuntu-22.04**”

To nači da VS Code radi “unutra” WSL-a, koristi Linux, a ne Windows Python.



Povezivanje VS Code-a sa Firedrake okruženjem

U VS Code-u:

1. Otvori **Command Palette**:

Ctrl + Shift + P

2. Ukucati: **Python: Select Interpreter**

Ako ga nema, instalirati extenziju za Python ponovo.

3. Izabratи interpreter iz svog virtuelnog okruženja, nešto kao:

/home/.../venv-firedrake/bin/python

Ako ga ne vidi:

- klikni na “Enter interpreter path”
 - pronađi ručno **venv-firedrake/bin/python**.
-

Instaliranje Gmsh-a

1. Ažurirati pakete (uvek dobro uraditi pre instalacije)

U WSL terminalu:

```
sudo apt update  
sudo apt upgrade -y
```

2. Instalirati Gmsh sa zvaničnog Ubuntu repozitorijuma

Najjednostavnije:

```
sudo apt install gmsh
```

Prihvati (y) ako postavi neko pitanje o instalaciji.

Proveriti instalaciju:

```
gmsh --version
```

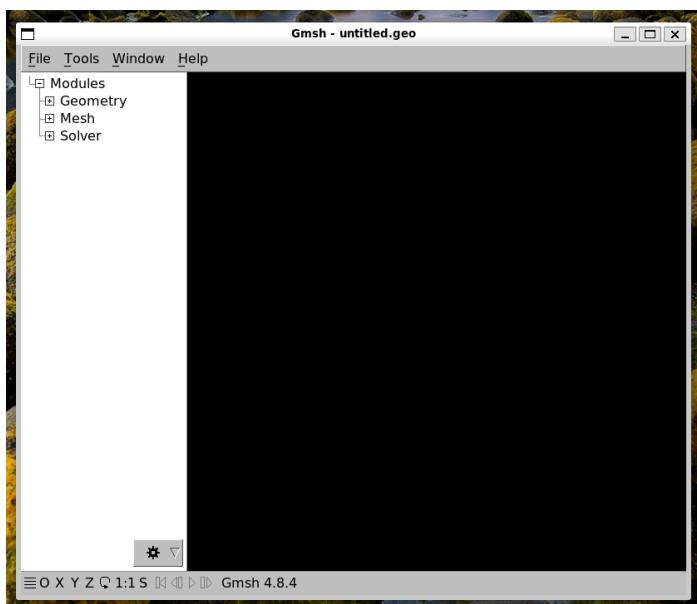
Meni ispisuje 4.8.4.

3. Pokretanje Gmsh-A

Komanda:

```
gmsh
```

Ako se otvorí grafički prozor — sve radi.



Instaliranje ParaView-a

1. Preuzmi ParaView za Windows

Ići na zvaničnu stranicu:

 <https://www.paraview.org/download/>

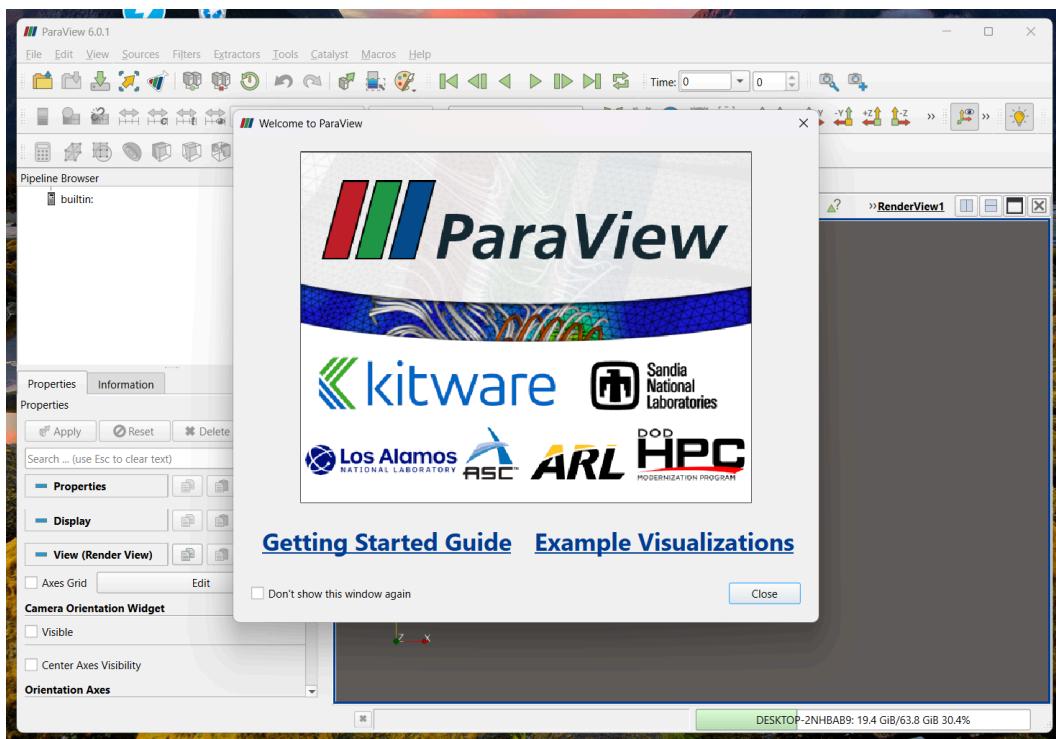
Preuzmi najnoviji fajl (ekstenzija `.msi`).

2. Instalacija

Pokrenuti instalaciju `.msi` → “Next → Next → Install”.

Posle instalacije, pokrenuti:

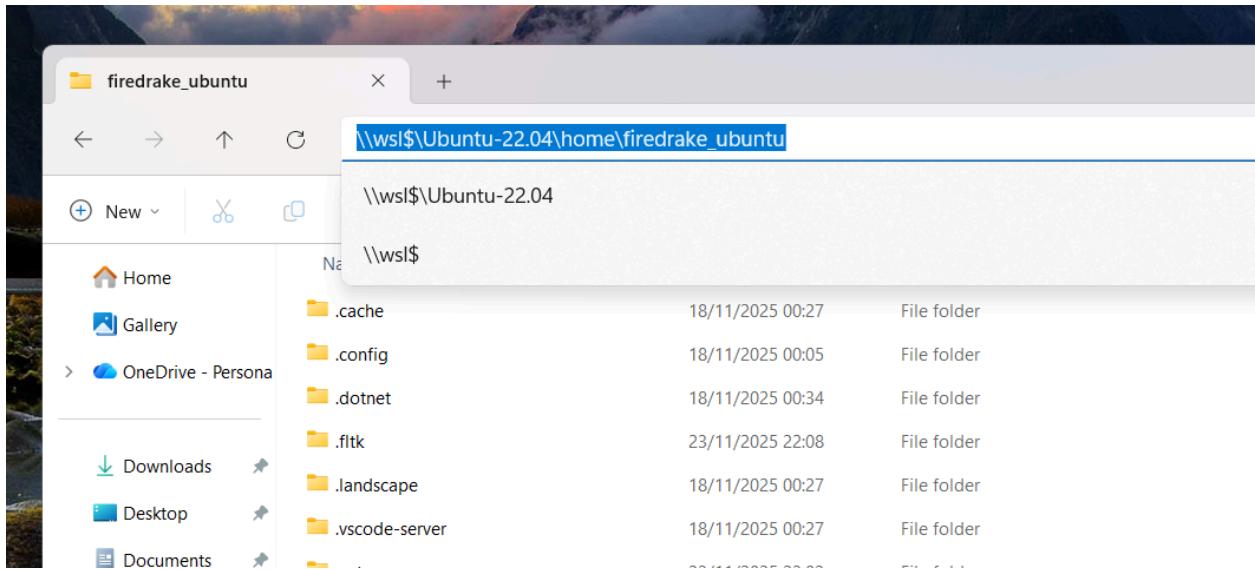
ParaView (x64)



3. Kako iz Windows ParaView-a otvoriti fajlove koji se nalaze u WSL-u

WSL fajlovi se nalaze u virtuelnom Linux sistemu, ali Windows ih vidi preko mrežnog puta:

`\wsl$\Ubuntu-22.04\home\firedrake_ubuntu`



Dakle, da otvorиш Firedrake rezultate:

1. U ParaView:

File → Open

2. U path bar upiši:

\\wsl\$\\Ubuntu-22.04\\home\\firedrake_ubuntu\\

U polje Name upisati ima fajla.

ParaView odmah može da učita:

- .**pvd**
- .**vtu**
- .**vts**
- .**vtr**
- .**xmf**
- .**pvtu**

- .pvt
-

4. Kako da se krećeš do WSL foldera bez kucanja

1. U Windows Explorer-u idi na:

`\wsl$\Ubuntu-22.04\home`

2. Desni klik na svoj folder (npr. `firedrake_ubuntu`) → **Pin to Quick Access**

Sada ćeš u ParaView-u lako moći da ga biraš preko „Quick Access“.

Testiranje

Test primer

Sledeće komande se unose u Ubuntu.

Napravićemo domen kružnicu izdeljenu na četiri dela.

1. Kreiramo fajl **circle4.geo** u kojem ćemo definisati geometriju domena:
code **circle4.geo**

Otvoriće se VS Code. Kopirati u njega sledeće linije i sačuvati:

```
// circle of radius R, split into 4 boundary segments with physical tags 1..4
R = 1.0;
h = 0.07; // target mesh size near boundary

Point(0) = {0, 0, 0};      // center
Point(1) = { R, 0, 0, h};
Point(2) = { 0, R, 0, h};
Point(3) = {-R, 0, 0, h};
Point(4) = { 0,-R, 0, h};

// four circular arcs (start, center, end)
Circle(1) = {1, 0, 2}; // tag 1/4 boundary quarter
Circle(2) = {2, 0, 3};
Circle(3) = {3, 0, 4};
Circle(4) = {4, 0, 1};

// close loop and make surface
Curve Loop(10) = {1,2,3,4};
Plane Surface(20) = {10};

// physical groups: facets (for BCs) & surface (for subdomain)
Physical Curve(1) = {1}; // right→top
Physical Curve(2) = {2}; // top→left
Physical Curve(3) = {3}; // left→bottom
Physical Curve(4) = {4}; // bottom→right
Physical Surface(100) = {20};

// a gentle size field toward interior (optional)
Field[1] = MathEval;
Field[1].F = "0.05 + 0.12*sqrt(x*x+y*y)";
Background Field = 1;
```

2. Vratimo se u terminal i napravimo mrežu koristeći sledeće linije:

```
gmsh -2 circle4.geo -o circle4.msh -format msh2
```

3. Napraviti fajl solve_poisson.py i kopirati sledeći kod za rešavanje Poasonove jednačine u Firedraku:

```
from firedrake import *

# učitaj mrežu iz Gmsh-a (physical curves 1..4 postaju facet markeri)
mesh = Mesh("circle4.msh")

# proveri oznake granica (opcionalno)
print("Facet markers:", sorted(mesh.exterior_facets.unique_markers))

V = FunctionSpace(mesh, "CG", 1)
u = Function(V, name="u")
v = TestFunction(V)

x, y = SpatialCoordinate(mesh)
f = Constant(1.0)

# slabi oblik Poasona:  $-\Delta u = f$ , sa prirodnim (0 Nojman) gde nema Dirihle
a = dot(grad(u), grad(v))*dx
L = f*v*dx

# različiti Dirihleovi na segmentima:
# segment 1: u = 0
# segment 3: u = 1
# (segmenti 2 i 4 su Nojman = 0)
bcs = [DirichletBC(V, 0.0, 1),
        DirichletBC(V, 1.0, 3)]

solve(a -L == 0, u, bcs=bcs)

File("u.pvd").write(u)
print("Done. Cells:", mesh.num_cells())
```

4. U terminalu pokrenuti komande:

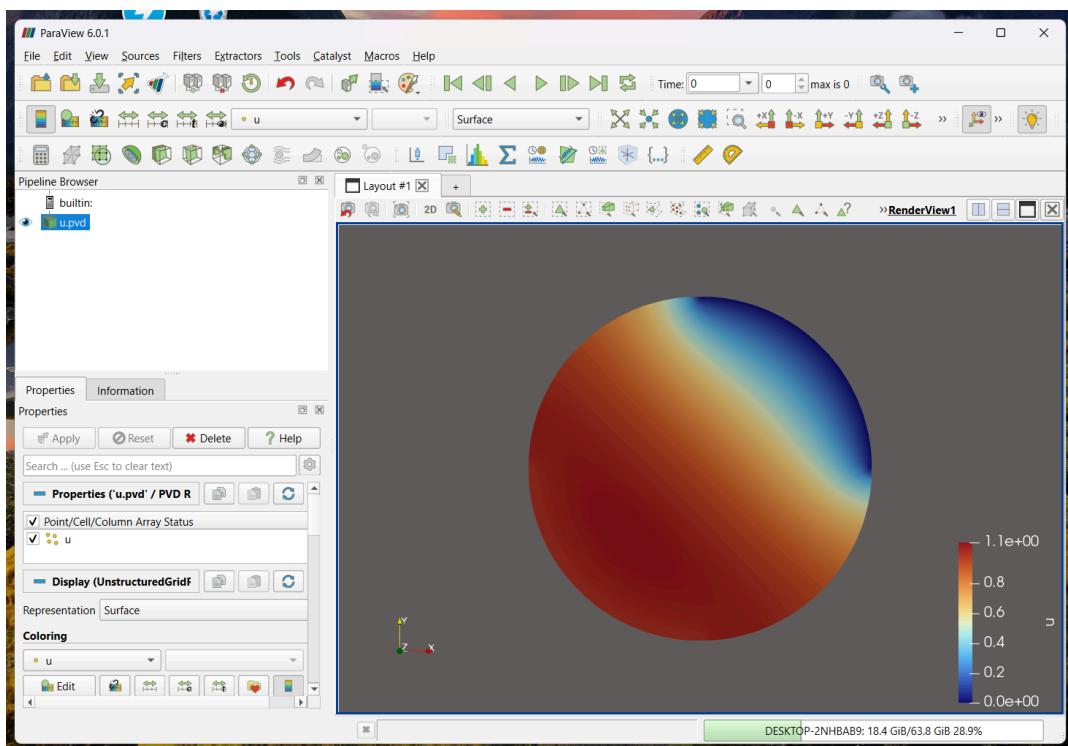
```
# ako želimo serijski
source venv-firedrake/bin/activate
python solve_poisson.py
```

```
# ako želimo paralelno (2 procesa)
source venv-firedrake/bin/activate
mpirun --oversubscribe -n 2 python solve_poisson.py
# očekujemo kreiranje fajla u.pvd
```

5. Visuelizacija:

paraview

→ File → Open → izaberi u.pvd → Apply.



Ponovna aktivacija okruženja

Kada izademo iz terminala, okruženje možemo aktivirati na sledeći način:

```
cd ~/firedrake-work
source venv-firedrake/bin/activate
```

A da se deaktivira, jednostavno komanda
`deactivate`