Gmsh (a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities)

Link: http://geuz.org/gmsh/

- 2d/3d meshing + postprocessing, unstructed meshes
- tip: tutorials
- Vhodná kombinace práce s GUI a se skriptem - edit-reload-save
- physical domains – labely podoblastí, velice užitečné, fenics by vyžduval explicitní popis hranic, zde si mohu pohodlně naklikat,
  gmsh přiřazuje labely automaticky, vhodné je ve skriptu doeditovat a rozumně přečíslovat, plus kotrola pomocí tools -> visibility
- ukázka překladu skriptu a volby max.velikosti elementu z příkazové řádky
  `gmsh -3 -clmax 0.08 pokus.geo`
- export meshe do fenicsu:
  konverzí do .xml pomocí
  příkazu: `dolfin-convert pokus.msh pokus.xml`
- načtení sítě ve fenicsu

```
mesh = Mesh("pokus.xml")
```

import labelu hranic:

```
boundaries=MeshFunction("size_t", mesh, "pokus_facet_region.xml")
```

volání Dirichl. podmínek potom standardní např.

```
u_top = Constant(0.0)
u_bottom = Constant(1.0)
bcs = [DirichletBC(V, u_bottom, boundaries, 1), DirichletBC(V, u_top, boundaries, 2)]
```

kde 1,2 jsou labely **physical** boundaries v gmsh, viz. pokus0.geo:

```
//nadefinovani geometrie
//------------------------------
cl__1 = 0.01;
//body
Point(1) = {0, 0, 0, 0.01};
Point(2) = {0.1, 0, 0, 0.01};
Point(3) = {0.1, 0.3, 0, 0.01};
Point(4) = {0, 0.3, 0, 0.01};
//cary coby dvojice bodu (orientovane)
Line(1) = {1, 2};
Line(2) = {3, 2};
Line(3) = {3, 4};
Line(4) = {4, 1};
// krivka coby mnozina orientovanych car
Line Loop(6) = {4, 1, -2, 3};
//povrch definovany vnejsi krivkou (prip. mnozinou vnitrnich krivek)
```

```
      Plane Surface(6) = {6};
      //zde definuji physical labels, ktere po konverzi mohu pouzit
jako identifikatory
      //podoblasti hranice ve fenicsu
      Bottom = 1;
      Physical Line(Bottom) = {1};
      Top = 2;
      Physical Line(Top) = {3};
      Sides = 3;
      Physical Line(Sides) = {2, 4};

      Surf = 1;
      Physical Surface(Surf) = {6};
```