Julia User Group Meeting

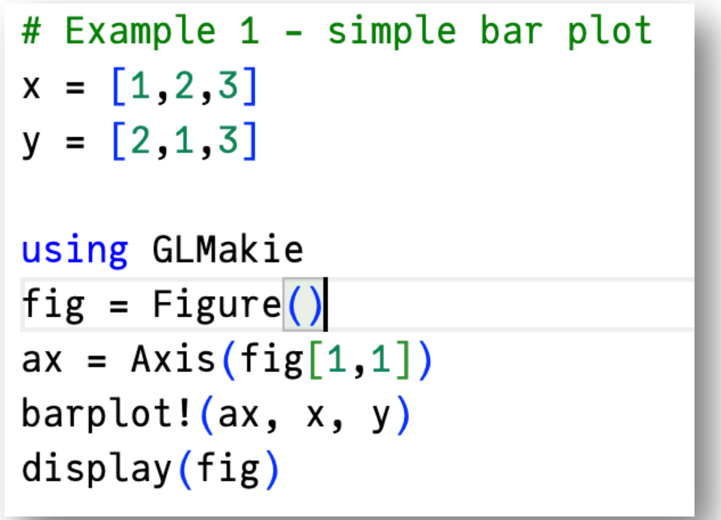
GUIs with GLMakie

Boris Kaus

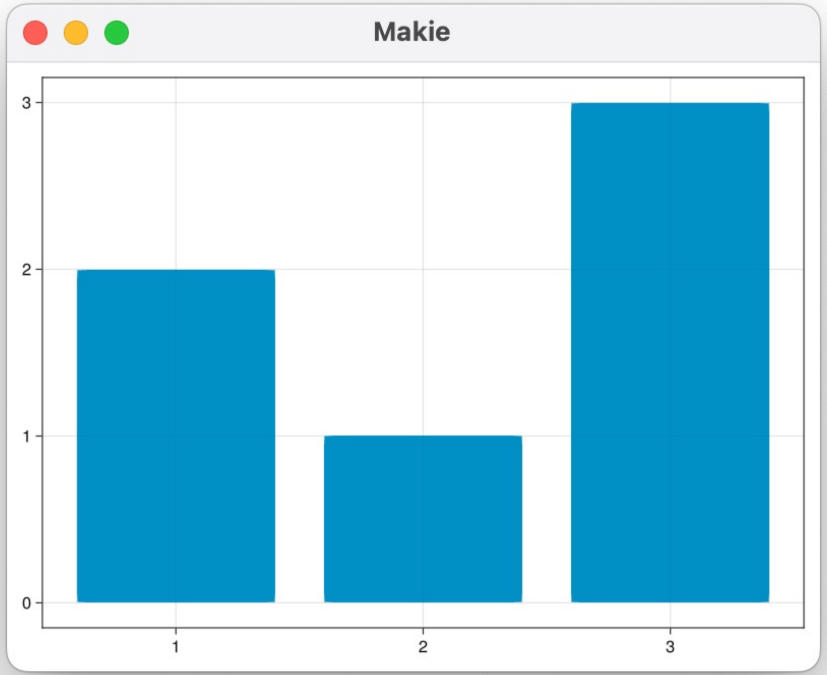
Graphical User Interfaces using GLMakie

First GUI

- Let’s start with a very simple plot:

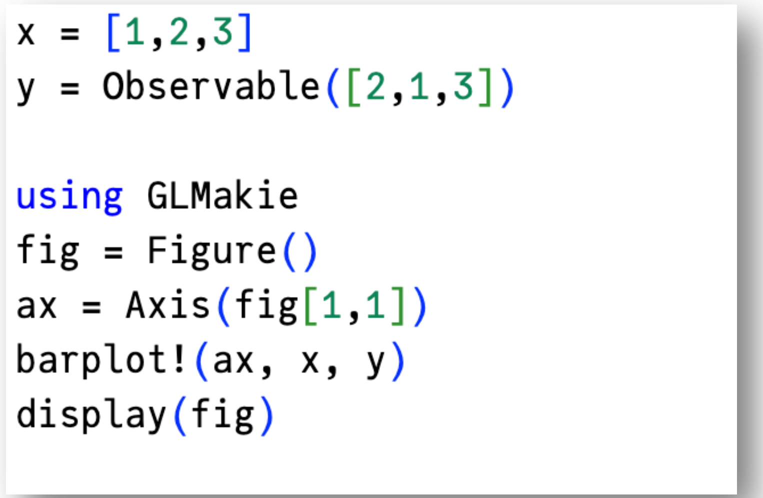


```julia
# Example 1 - simple bar plot
x = [1,2,3]
y = [2,1,3]

using GLMakie
fig = Figure()
ax = Axis(fig[1,1])
barplot!(ax, x, y)
display(fig)
```

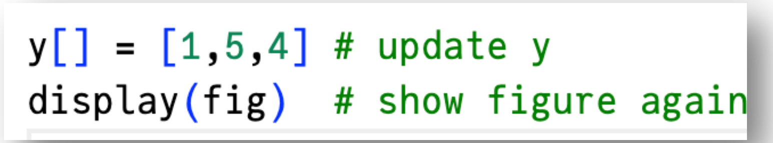


Observable

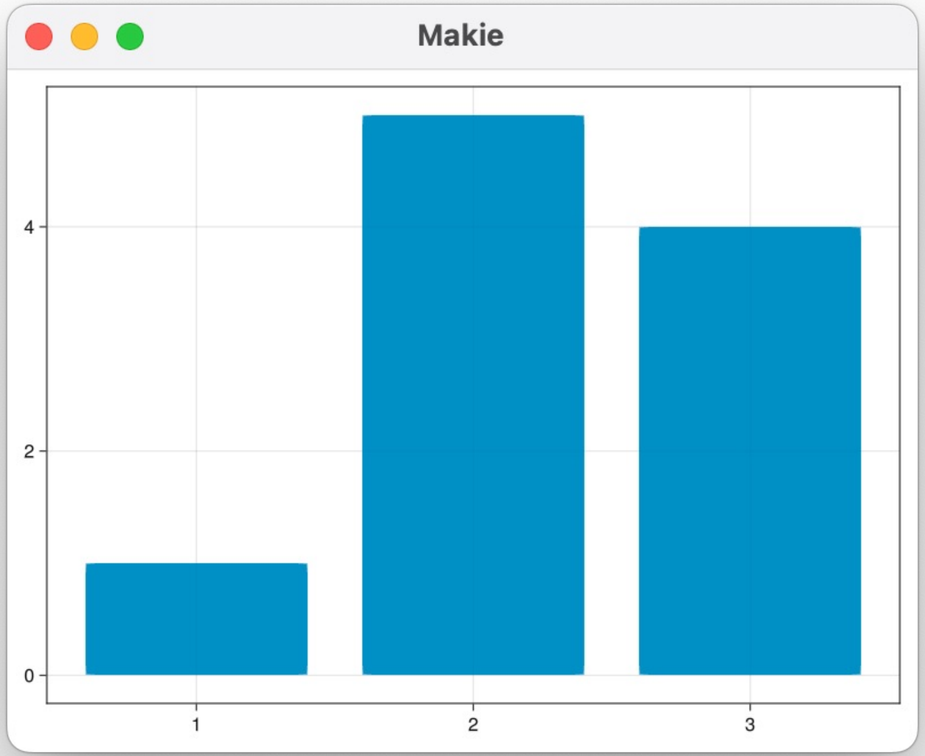
- If you make a variable an Observable, it will update the plot once this variable is changed



```julia
x = [1,2,3]
y = Observable([2,1,3])

using GLMakie
fig = Figure()
ax = Axis(fig[1,1])
barplot!(ax, x, y)
display(fig)
```



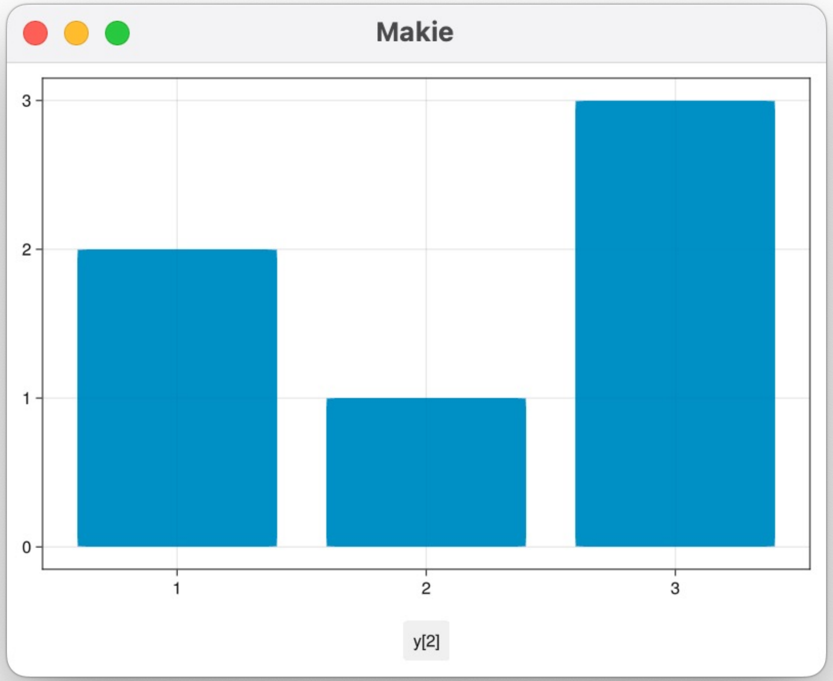
```julia
y[] = [1,5,4] # update y
display(fig)  # show figure again
```



Let’s add a Button

- You can adjust the location by making fig[1,1:3] larger:

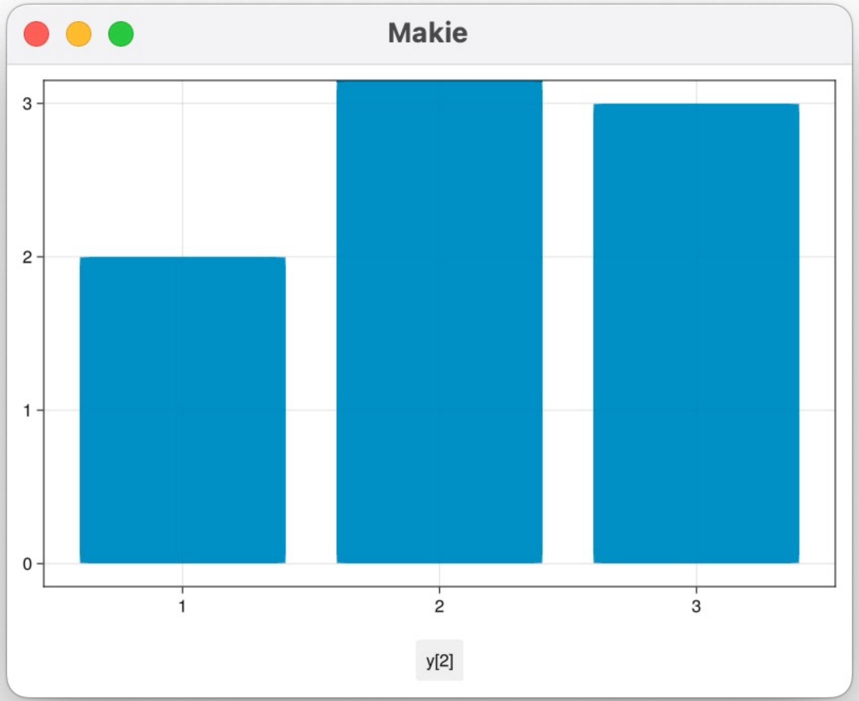
```
x = [1,2,3]
y = Observable([2,1,3])

using GLMakie
fig = Figure()
ax  = Axis(fig[1,1:3])
but = Button(fig[2,2], label="y[2]")
barplot!(ax, x, y)

display(fig)
```



Let’s add some action to the Button

- Use the on(…) do n construct:

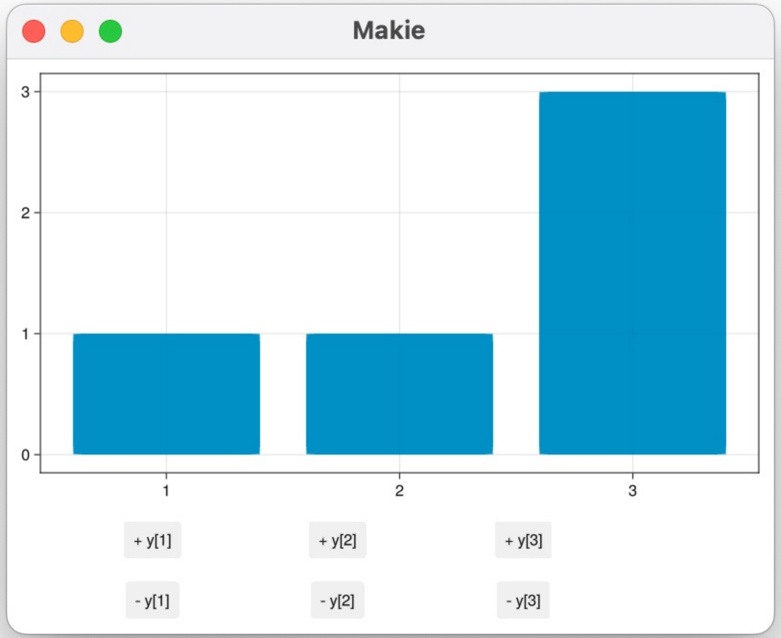
```julia
x = [1,2,3]
y = Observable([2,1,3])

using GLMakie
fig = Figure()
ax  = Axis(fig[1,1:3])
but = Button(fig[2,2], label="y[2]")
barplot!(ax, x, y)

on(but.clicks) do n
    @show n
    y[][2] +=1     # update y[2]
    notify(y)      # update plot
end
display(fig)
```



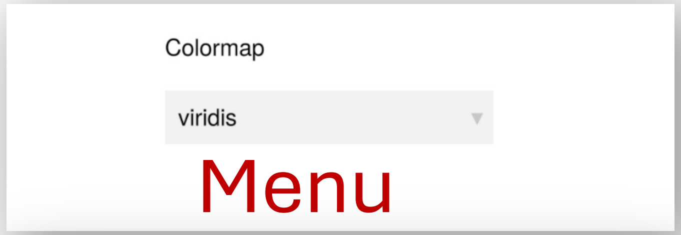
```
julia> n = 1
n = 2
n = 3
```

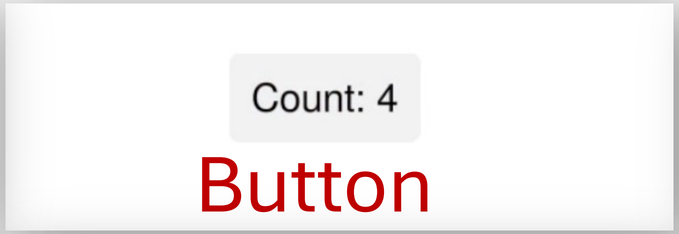
Exercise 1

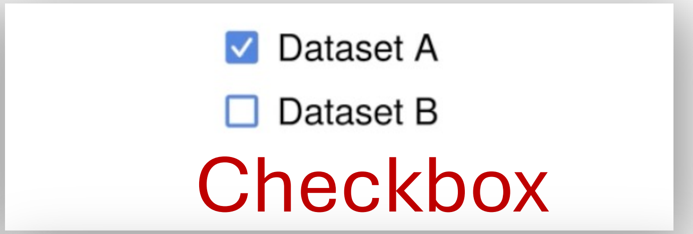
1. Add 2 additional buttons with which you can increase y[1] and y[3]
2. Add another row of buttons with which you can decrease the values by 1

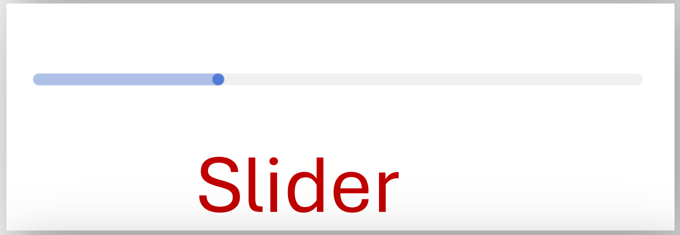


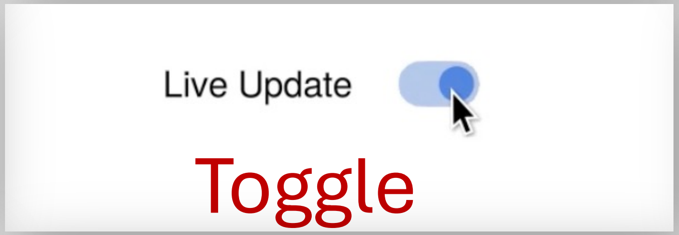
Different GUI items

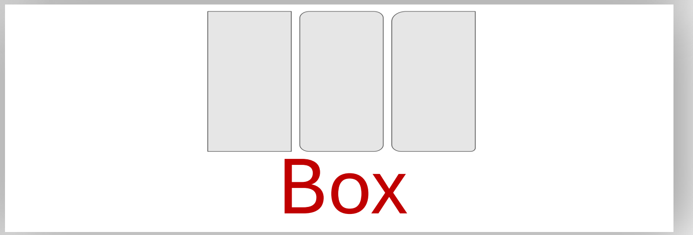


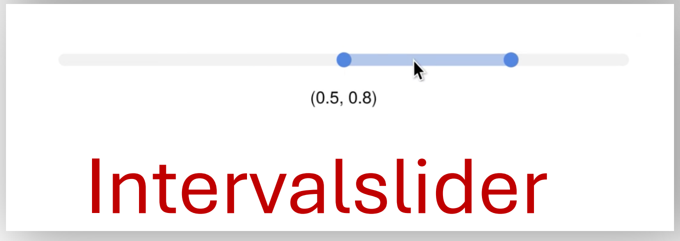


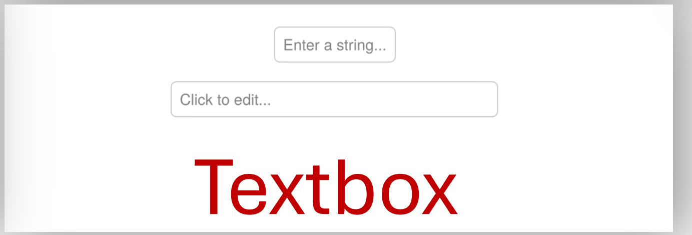












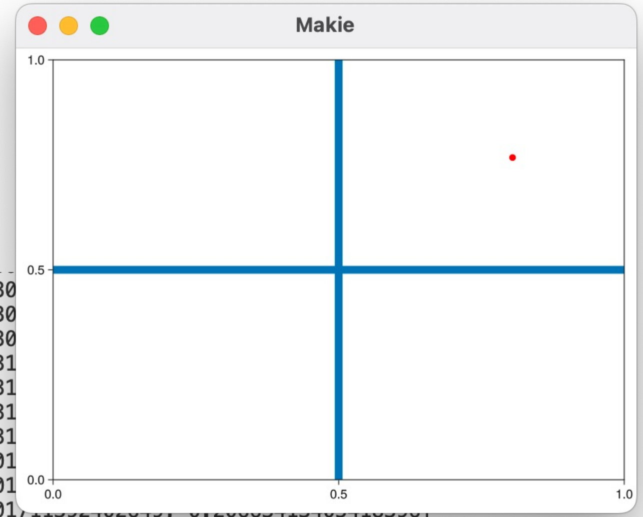
For help see the “blocks” part of the documentation:
https://docs.makie.org/stable/reference/blocks/button

Exercise 2 – more sophisticated GUI

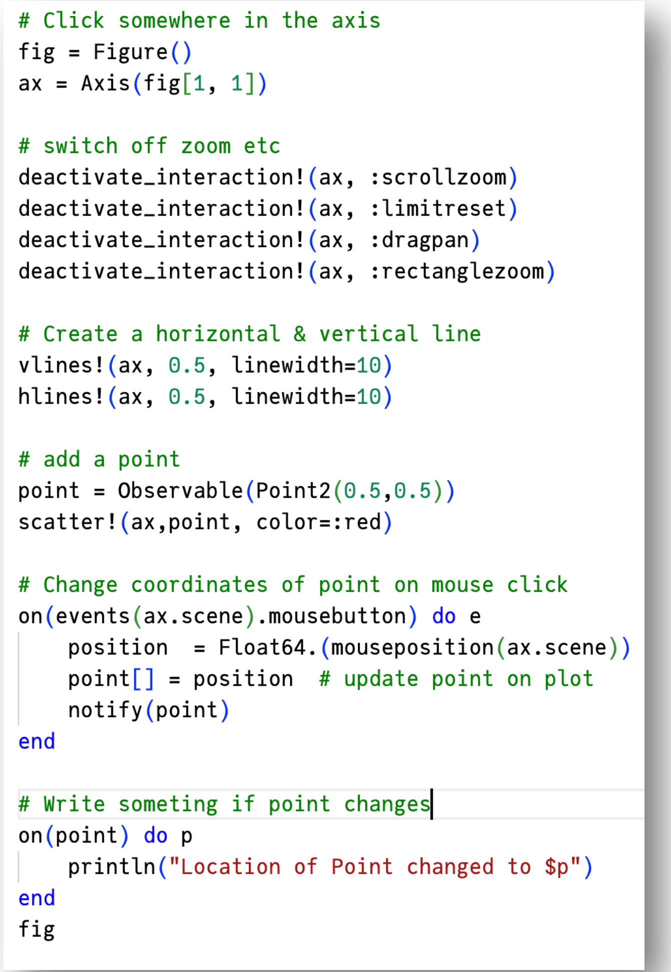
- Create a GUI that plots either one of the two functions:
  - $f(x) = a*x^3 + b*x^2$
  - $f(x) = a*x^2 + b*x$
  - Over the interval from x=-2:.01:2
- Features of the GUI:
  - Dropdown menu to select the function
  - Dropdown menu to select the color of the line (:red, :blue, :green)
  - Textboxes to specify a and b

Feedback after clicking on an axis

- You can add feedback when you click on an axis
- Example:
  - Use your mouse to select a point
  - Use this to select a river in a catchment area



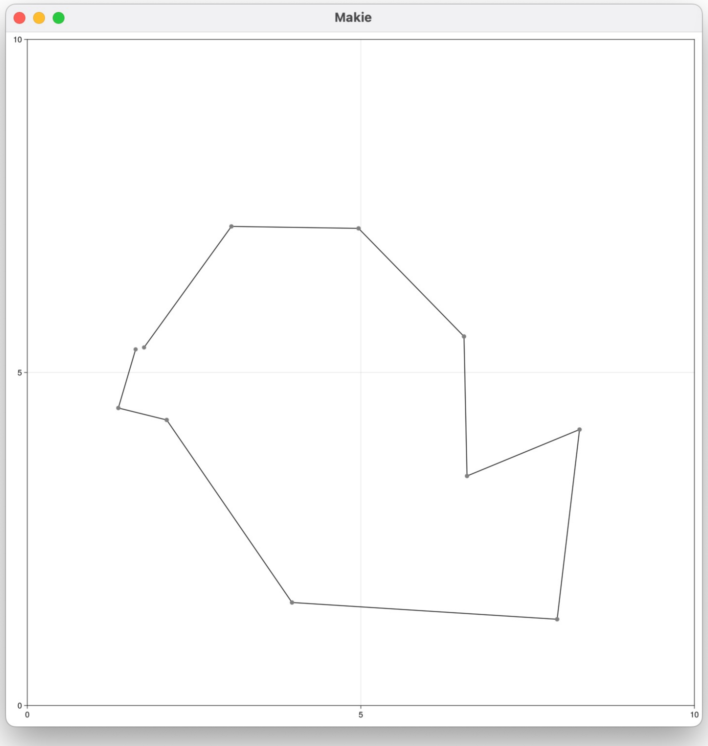
```
Location of Point changed to [0.8830
Location of Point changed to [0.8830
Location of Point changed to [0.8830
Location of Point changed to [0.1031
Location of Point changed to [0.1031
Location of Point changed to [0.1031
Location of Point changed to [0.1031
Location of Point changed to [0.8601
Location of Point changed to [0.8601
Location of Point changed to [0.8601
Location of Point changed to [0.8601711392402649, 0.20663413405418396]
```



```
# Click somewhere in the axis
fig = Figure()
ax = Axis(fig[1, 1])

# switch off zoom etc
deactivate_interaction!(ax, :scrollzoom)
deactivate_interaction!(ax, :limitreset)
deactivate_interaction!(ax, :dragpan)
deactivate_interaction!(ax, :rectanglezoom)

# Create a horizontal & vertical line
vlines!(ax, 0.5, linewidth=10)
hlines!(ax, 0.5, linewidth=10)

# add a point
point = Observable(Point2(0.5,0.5))
scatter!(ax,point, color=:red)

# Change coordinates of point on mouse click
on(events(ax.scene).mousebutton) do e
    position  = Float64.(mouseposition(ax.scene))
    point[] = position  # update point on plot
    notify(point)
end

# Write someting if point changes
on(point) do p
    println("Location of Point changed to $p")
end
fig
```

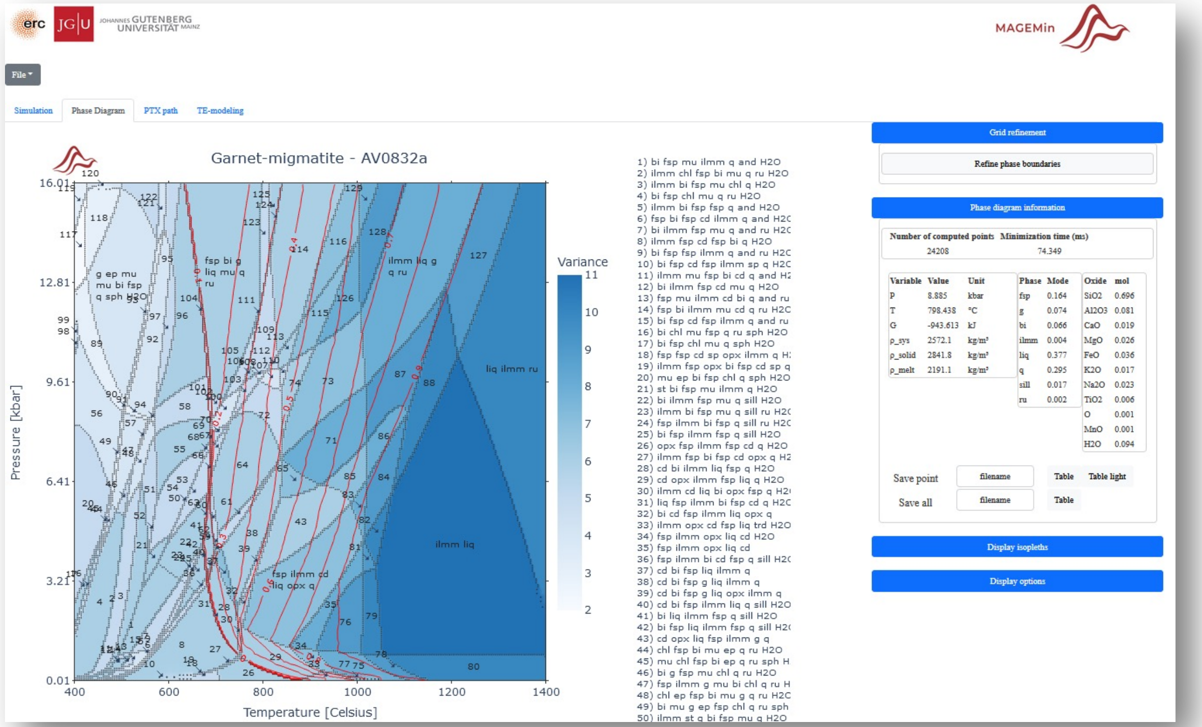
Draw a curve interactively

```julia
# Draw a line with mouse clicks
fig = Figure()
ax = Axis(fig[1, 1])
for interact in keys(ax.interactions)
    deactivate_interaction!(ax, interact)
end
points = Observable(Point2f[])
linesegments!(ax, points, color = :black)
scatter!(ax, points, color = :gray)

on(events(ax.scene).mousebutton) do event
    if event.button == Mouse.left
        if event.action == Mouse.press || event.action == Mouse.release
            mp = mouseposition(ax.scene)
            push!(points[], mp)
            notify(points)
        end
    end
end
fig
```



Other packages – Dash.jl

- Run GUI’s directly in the browser
- More complicated to develop
- e.g., MAGEMinApp.jl or InteractiveGeodynamics.jl



```
1) bi fsp mu ilmm q and H2O
2) ilmm chl fsp bi mu q ru H2O
3) ilmm bi fsp mu chl q H2O
4) bi fsp chl mu q ru H2O
5) ilmm bi fsp fsp q and H2O
6) fsp bi fsp cd ilmm q and H2C
7) bi ilmm fsp mu q and ru H2C
8) ilmm fsp cd fsp bi q H2O
9) bi fsp fsp ilmm q and ru H2C
10) bi fsp cd fsp ilmm sp q H2C
11) ilmm mu fsp bi cd q and H2
12) bi ilmm fsp cd mu q H2O
13) fsp mu ilmm cd bi q and ru
14) fsp bi ilmm mu cd q ru H2C
15) bi fsp cd fsp ilmm q and ru
16) bi chl mu fsp q ru sph H2O
17) bi fsp chl mu q sph H2O
18) fsp fsp cd sp opx ilmm q H
19) ilmm fsp opx bi fsp cd sp q
20) mu ep bi fsp chl q sph H2O
21) st bi fsp mu ilmm q H2O
22) bi ilmm fsp mu q sill H2O
23) ilmm bi fsp mu q sill ru H2C
24) fsp ilmm bi fsp q sill ru H2C
25) bi fsp ilmm fsp q sill H2O
26) opx fsp ilmm fsp cd q H2O
27) ilmm fsp bi fsp cd opx q H2
28) cd bi ilmm liq fsp q H2O
29) cd opx ilmm fsp liq q H2O
30) ilmm cd liq bi opx fsp q H2
31) liq fsp ilmm bi fsp cd q H2C
32) bi cd fsp ilmm liq opx q
33) ilmm opx cd fsp liq trd H2O
34) fsp ilmm opx liq cd H2O
35) fsp ilmm opx liq cd
36) fsp ilmm bi cd fsp q sill H2C
37) cd bi fsp liq ilmm q
38) cd bi fsp g liq ilmm q
39) cd bi fsp g liq opx ilmm q
40) cd bi fsp ilmm liq q sill H2O
41) bi liq ilmm fsp q sill H2O
42) bi fsp liq ilmm fsp q sill H2C
43) cd opx liq fsp ilmm g q
44) chl fsp bi mu ep q ru H2O
45) mu chl fsp bi ep q ru sph H
46) bi g fsp mu chl q ru H2O
47) fsp ilmm g mu bi chl q ru H
48) chl ep fsp bi mu g q ru H2C
49) bi mu g ep fsp chl q ru sph
50) ilmm st g bi fsp mu q H2O
```

Number of computed points	Minimization time (ms)
24208	74.349

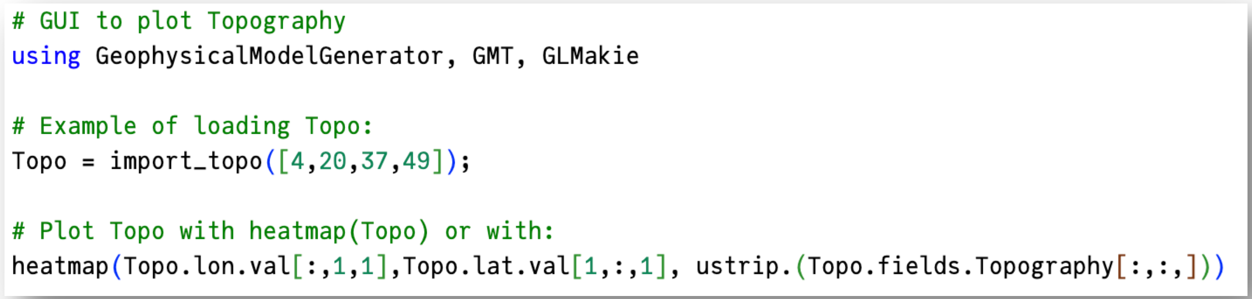
Variable	Value	Unit
P	8.885	kbar
T	798.438	°C
G	-943.613	kJ
ρ_sys	2572.1	kg/m³
ρ_solid	2841.8	kg/m³
ρ_melt	2191.1	kg/m³

Phase	Mode
fsp	0.164
g	0.074
bi	0.066
ilmm	0.004
liq	0.377
q	0.295
sill	0.017
ru	0.002

Oxide	mol
SiO2	0.696
Al2O3	0.081
CaO	0.019
MgO	0.026
FeO	0.036
K2O	0.017
Na2O	0.023
TiO2	0.006
O	0.001
MnO	0.001
H2O	0.094

Exercise 3 - topography

- Create a GUI that uses the GeophysicalModelGenerator package to plot a topography from a certain region area
- Specify lower-left and upper-right longitude-latitude corners



```julia
# GUI to plot Topography
using GeophysicalModelGenerator, GMT, GLMakie

# Example of loading Topo:
Topo = import_topo([4,20,37,49]);

# Plot Topo with heatmap(Topo) or with:
heatmap(Topo.lon.val[:,1,1],Topo.lat.val[1,:,1], ustrip.(Topo.fields.Topography[:,:,]))
```

Summary

- You made your first GUIs