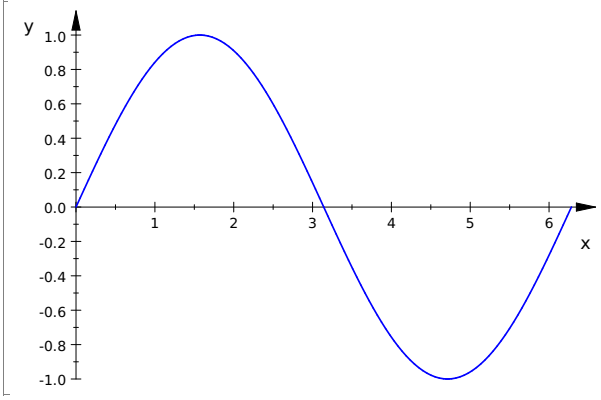
INTRODUCTION TO GRAPHICS IN MUPAD - just an introduction!

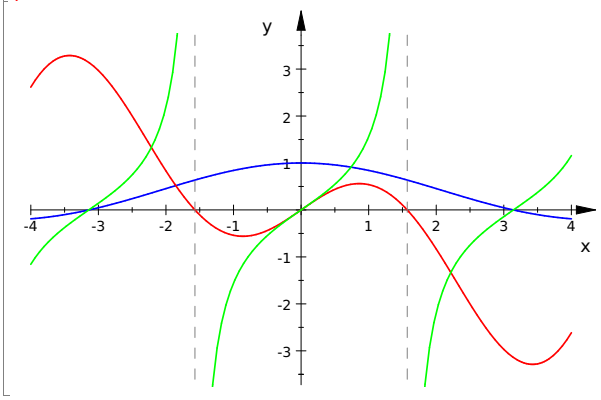
Simple 2d plot

```
plot( sin(x), x=0..2*PI  )
```



Multiple 2d plots

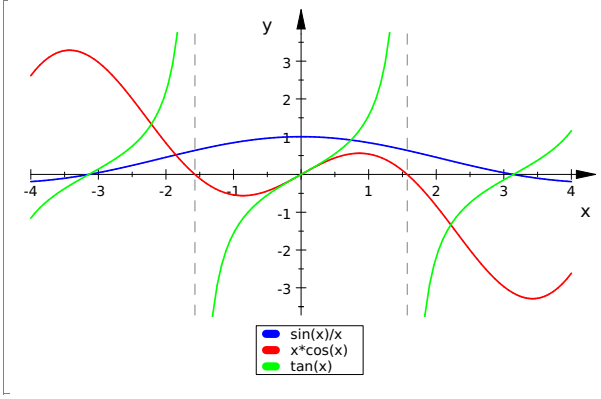
```
plot( sin(x)/x, x*cos(x), tan(x), x=-4..4)
```



Note (1) default colors, (2) truncation of y axis

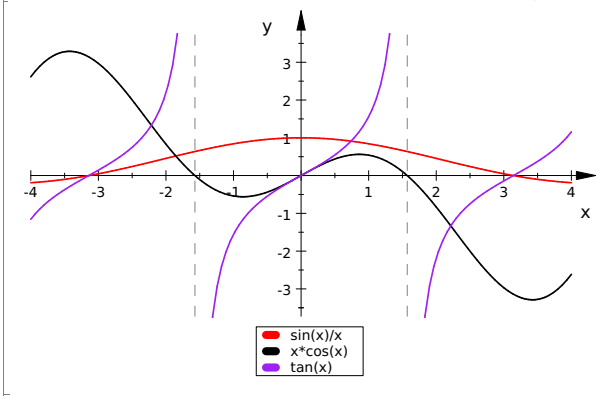
Adding legend (מקרא) - one of many plot options

```
plot( sin(x)/x, x*cos(x), tan(x), x=-4..4, LegendVisible)
```



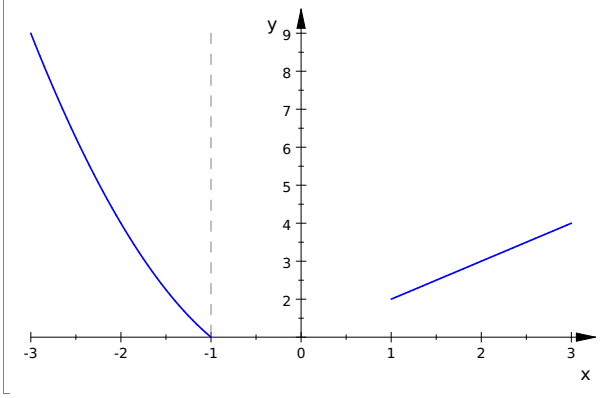
Controlling colors

```
plot( sin(x)/x, x*cos(x), tan(x), x=-4..4, LegendVisible, Colors=[RGB::Red,RGB::Black,RGB::Purple])
```

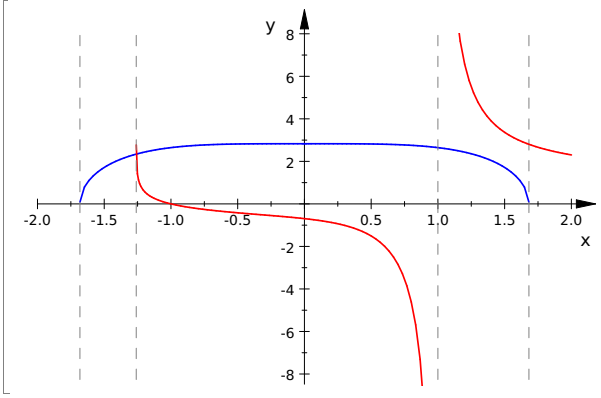


Things to be plotted do not have to be defined everywhere

```
plot( piecewise( [x<-1, x^2], [x>1, x+1] ), x=-3..3 )
```

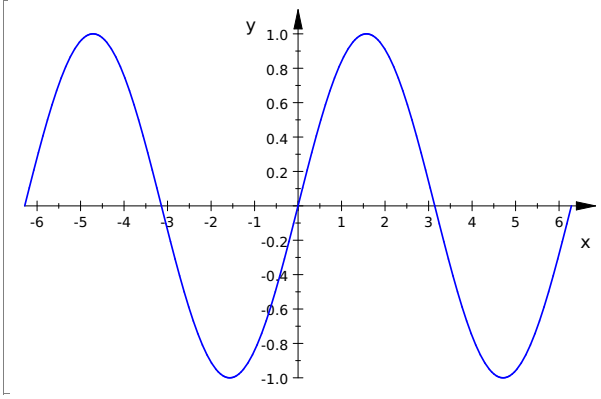



```
plot( sqrt(8 - x^4), ln(x^3 + 2)/(x - 1), x = -2 ..2 )
```



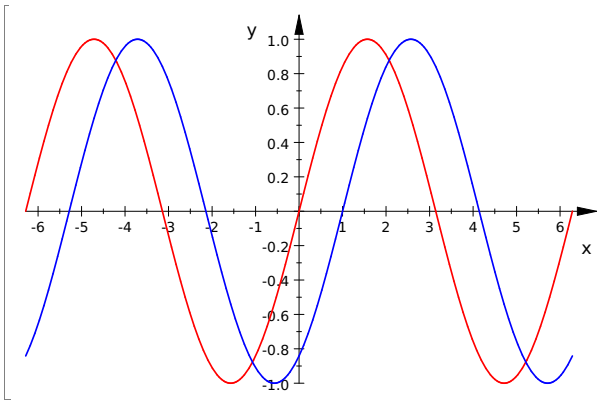
Making an animation:

```
plot( sin(a*x), x=-2*PI..2*PI, a=1..4 )
```



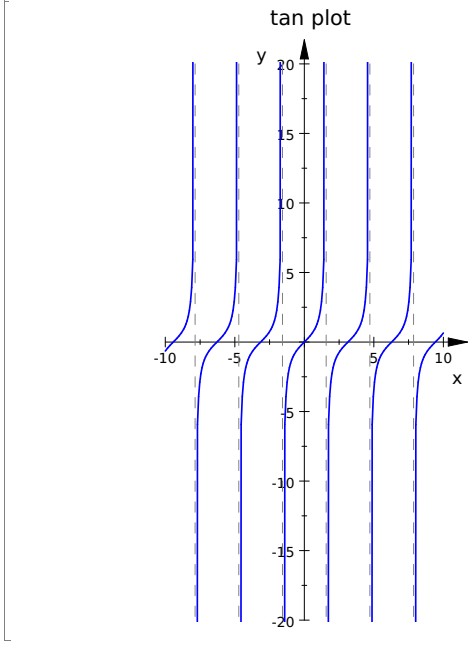
Control with the controls on the toolbar (get by clicking on the graphic)
In general left clicking on a graphic gives you tools to manipulate the graphic.
Right clicking gives you options to change size, open in a separate window, export. Play with the different options!
Back to animation. Default number of frames is 50. Change by

```
plot( sin(a*x), sin(x-a), x=-2*PI..2*PI, a=1..4, Frames=200, Colors=[RGB::Red, RGB::Blue] )
```

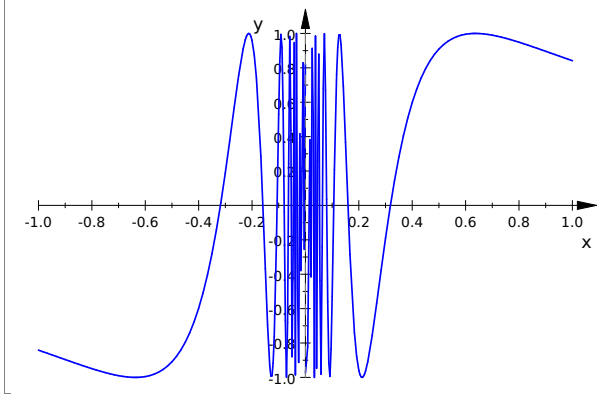


A few more useful plot attributes (there are hundreds!) :
Height, Width, Header, Scaling (=Automatic, Constrained or Unconstrained), VeiwingBoxYRange (=ymin..yman), Mesh

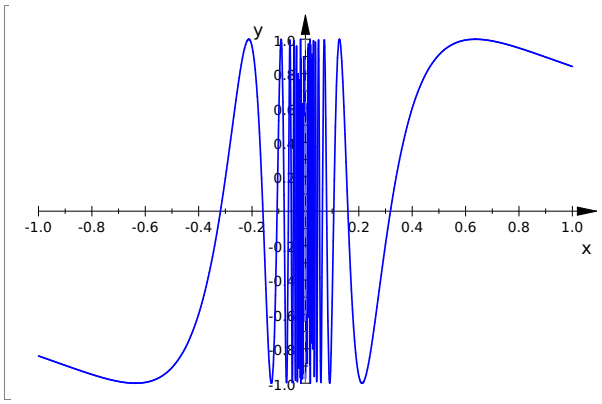
```
plot(tan(x), x=-10..10, Scaling=Constrained, ViewingBoxYRange=-20..20, Header="tan plot", Height=13*unit::cm)
```



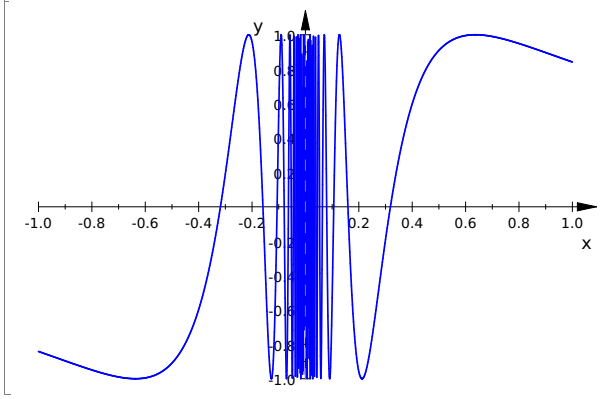

```
plot( sin(1/x), x=-1..1 )
```




```
plot( sin(1/x), x=-1..1, Mesh=500)
```

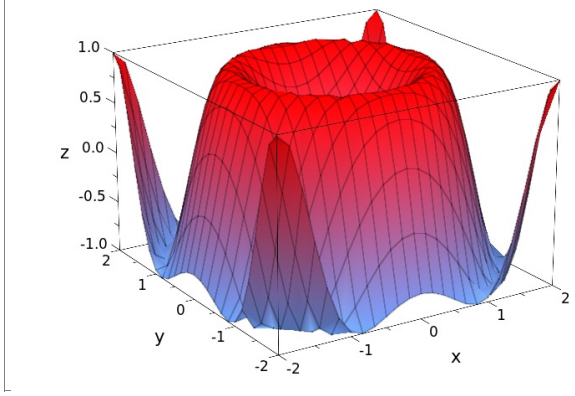



```
plot( sin(1/x), x=-1..1, Mesh=1000)
```



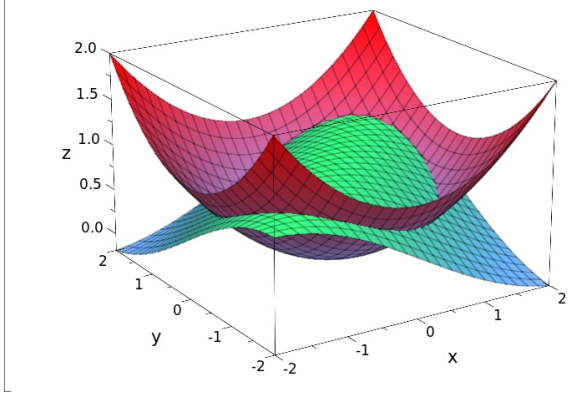
3-D plots - of graphs / surfaces

```
plot(sin(x^2 + y^2), x = -2..2, y = -2..2, #3D)
```

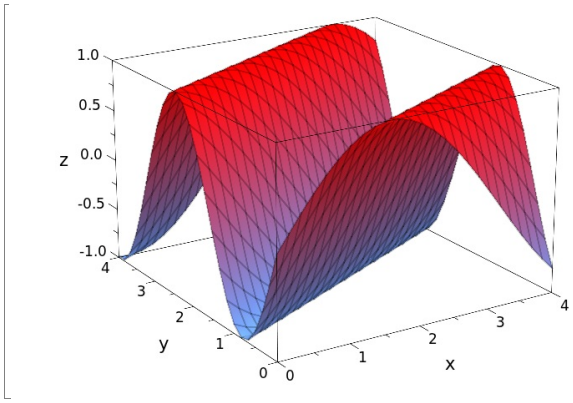


Note: coloring to help visualize, options to manipulate on toolbar (especially the auto-rotate)

```
plot((x^2 + y^2)/4, sin(x - y)/(x - y), x = -2..2, y = -2..2, #3D)
```




```
plot(sin(a*(x-2*y)), x = 0..4, y = 0..4, a = 1..2, #3D)
```



(auto-rotate while it is running!)

A little of the machinery:
the "plot" command above does 2 things - (1) for each thing to be displayed it creates a "graphical object" (consisting of all the data that needs to be displayed), and (2) it displays it/them ("rendering")
We can divide up the stages, giving us more control, and the ability to form more complex scenes.
e.g. to plot f(x)=xsin(x), add the point x=1, and the tangent at this point

```
f:=x*sin(x);  df:=diff(f,x); x0:=1;
```

$$x \sin(x)$$

$$\sin(x) + x \cos(x)$$

$$1$$

```
a:=plot::Function2d(f,x=0..2*PI)
```

plot::Function2d($x \sin(x)$, $x = 0 \,..\, 2\,\pi$)

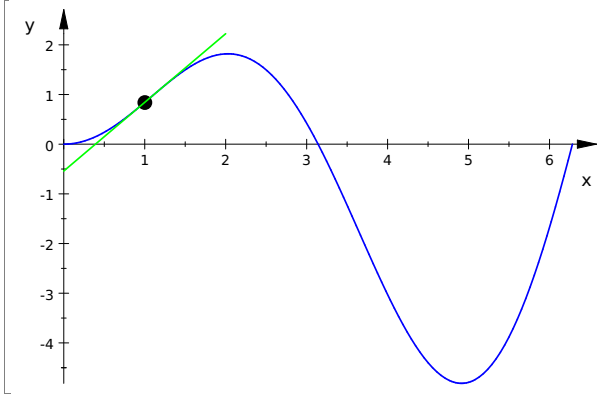
```
b:=plot::Point2d( [x0,subs(f,x=x0)], Color=RGB::Black, PointSize = 3.0*unit::mm )
```

plot::Point2d(1, sin(1), PointColor = RGB::Black, PointSize = 3.0)

```
c:=plot::Line2d( [x0-1,subs(f,x=x0)-subs(df,x=x0)],  [x0+1,subs(f,x=x0)+subs(df,x=x0)] , Color=RGB::Green)
```

plot::Line2d([0, $-\cos(1)$], [2, $\cos(1) + 2\sin(1)$])

```
plot(a,b,c)
```



You can change attributes of a particular graphical object after they have been created. For example

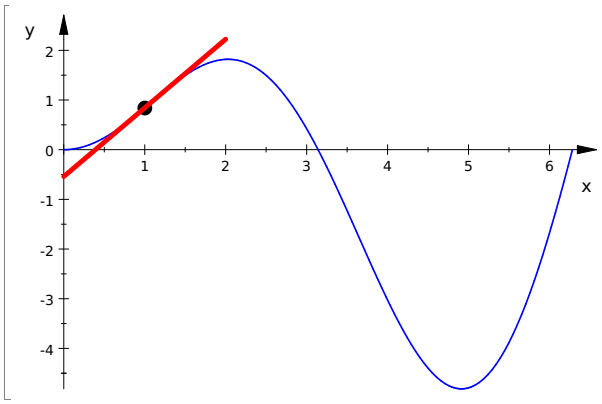
```
c::Color := RGB::Red;
```

[1.0, 0.0, 0.0]

```
c::LineWidth := 1.0*unit::mm
```

1.0 mm

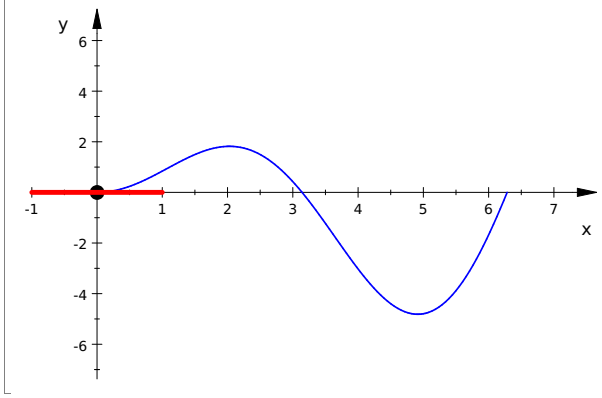
```
plot(a,b,c)
```



OK, let's animate this!!

```
delete x0;
```

```
a:=plot::Function2d(f,x=0..2*PI):
b:=plot::Point2d( [x0,subs(f,x=x0)], x0=0..2*PI, Color=RGB::Black, PointSize = 3.0*unit::mm ):
c:=plot::Line2d( [x0-1,subs(f,x=x0)-subs(df,x=x0)],  [x0+1,subs(f,x=x0)+subs(df,x=x0)] , x0=0..2*PI,
          Color=RGB::Red, LineWidth = 1.0*unit::mm):
plot(a,b,c)
```



Note: cannot mix 2d and 3d data types!

```
q:=plot::Point3d( [1,2,3] )
```

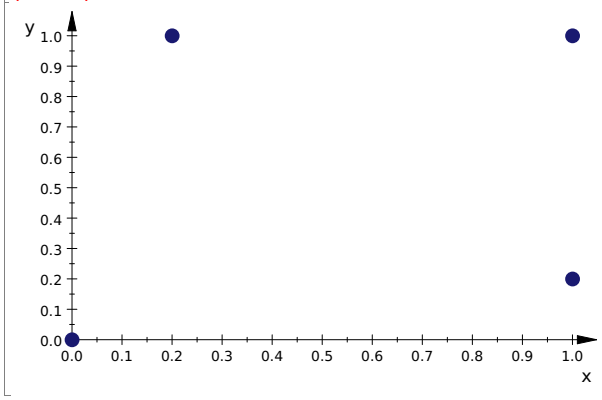
plot::Point3d(1, 2, 3)

```
plot(a,q)
Error: dimension mismatch [plot::easy[setSceneDimension]]
```

A bunch of useful primitives for plotting:

Low level: Arrow2d, Arrow3d, Box, Circle2d, Circle3d, Line2d, Line3d, Plane, Point2d, Point3d, PointList2d, PointList3d, Polygon2d, Polygon3d, Rectangle, Sphere
High level: Curve2d, Curve3d, Function2d, Function3d, Implicit2d, Implicit3d, Histogram, Piechart2d, Piechart3d, Surface
Examples:

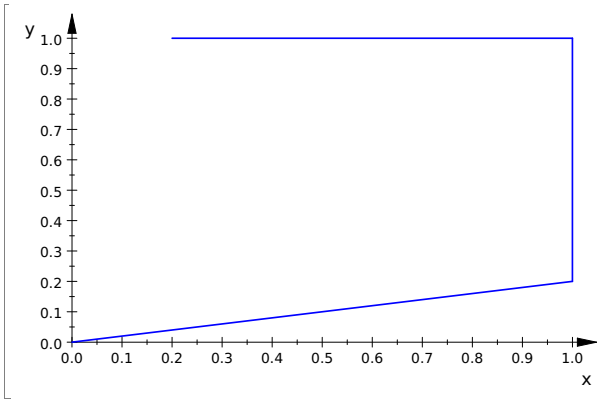
```
plot( plot::PointList2d([ [0,0], [1,0.2], [1,1], [0.2,1] ] , PointSize=3) )
```




```
p:=plot::Polygon2d([ [0,0], [1,0.2], [1,1], [0.2,1] ])
```

plot::Polygon2d([[0, 0], [1, 0.2], [1, 1], [0.2, 1]])

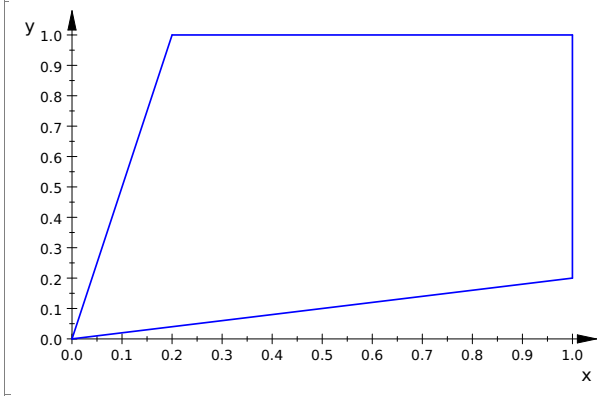
```
plot(p)
```




```
p::Closed:=TRUE
```

TRUE

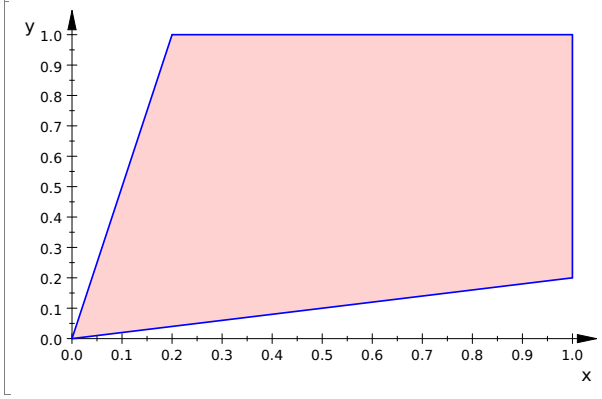
```
plot(p)
```



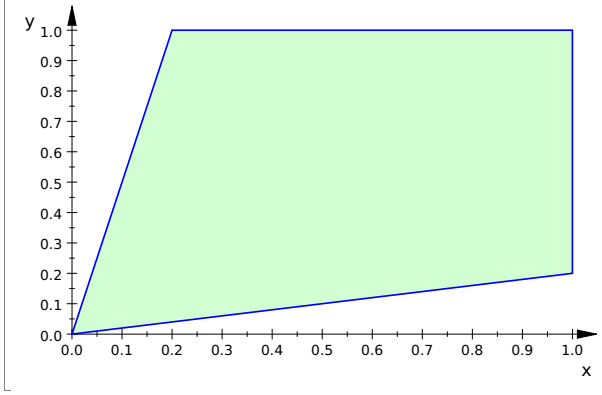

```
p::Filled:=TRUE
```

TRUE

```
plot(p)
```

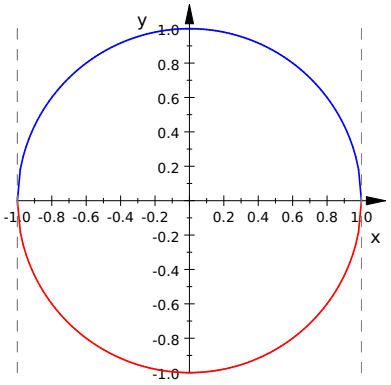



```
plot(p, FillColor=RGB::Green)
```



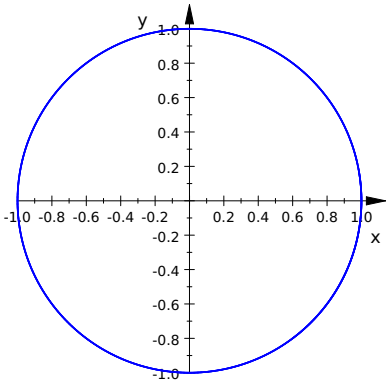
Different ways to get curves
direct definition: y=f(x)

```
plot( sqrt(1-x^2), -sqrt(1-x^2), x=-1..1, Scaling=Constrained)
```



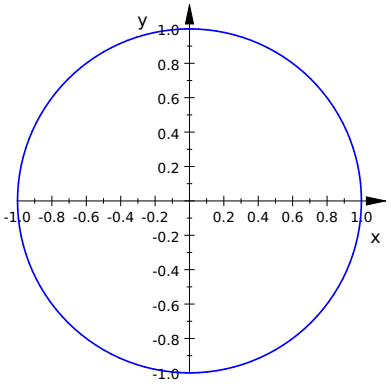
implicit definition f(x,y)=0

```
plot( plot::Implicit2d( x^2+y^2=1, x=-1..1, y=-1..1) , Scaling=Constrained)
```



parametric definition x=x(t), y=y(t)

```
plot( plot::Curve2d( [cos(t),sin(t)], t=0..2*PI ) ,Scaling=Constrained )
```

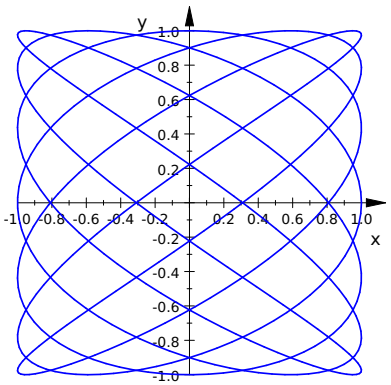


Lissajous figures: change the n and m below

```
n:=7; m:=5; plot( plot::Curve2d( [cos(n*t),sin(m*t)], t=0..2*PI, Mesh=500 ) ,Scaling=Constrained )
```

7

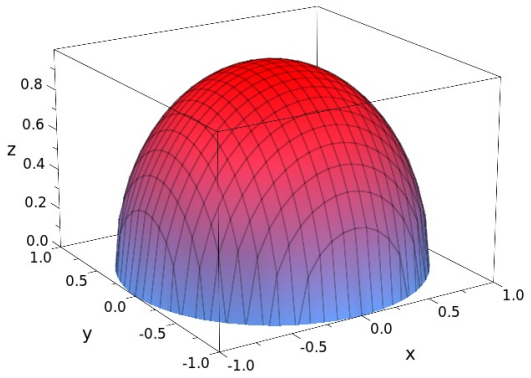
5



Different ways to get surfaces

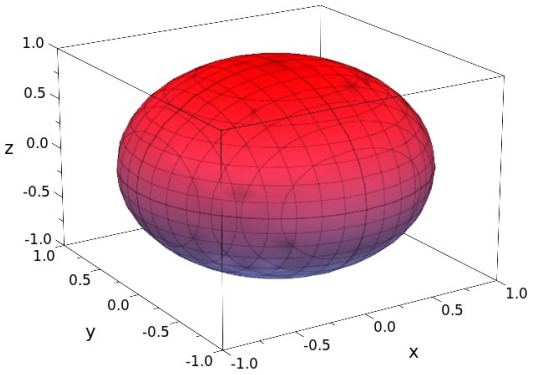
Direct definition z=f(x,y)

```
plot( plot::Function3d(sqrt(1-x^2-y^2), x=-1..1, y=-1..1) )
```



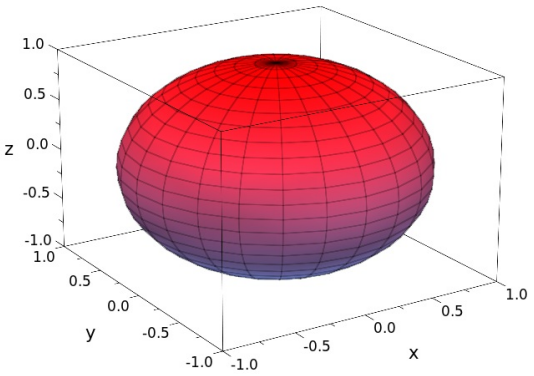
Implicit definition f(x,y,z)=0

```
plot( plot::Implicit3d( x^2+y^2+z^2=1, x=-1..1, y=-1..1, z=-1..1) )
```



Parametric definition x=x(s,t), y=y(s,t), z=z(s,t)

```
plot( plot::Surface( [sin(t)*cos(s), sin(t)*sin(s), cos(t)], t=0..PI, s=0..2*PI ))
```



A 3d curve:

```
plot( plot::Curve3d( [ cos(t), sin(t), t], t=0..50, Mesh=1000, LineWidth=0.5 ) )
```