RASCAL

www.rascal-mpl.org

Figure Cheat Sheet

Font and Text Properties	
font(str fontName)	Named font
fontSize(int s)	Font size
fontColor(Color c)	Font color
fontColor(str name)	Font color
textAngle(num a)	Text angle

Wedge Properties	
fromAngle(num a)	Begin angle
toAngle(num a)	End angle
innerRadius(num r)	Inner radius

Other Properties	
id(str name)	Figure name
doi(int n)	Degree-of-interest

Property (FProp)

Size Properties	
width(num w)	Horizontal (H) size
height(num h)	Vertical (V) size
size(num s), size(num w, num h)	H and V size
gap(num g) gap(num w, num h)	H and V gap
hgap(num h) vgap(num v)	H or V gap

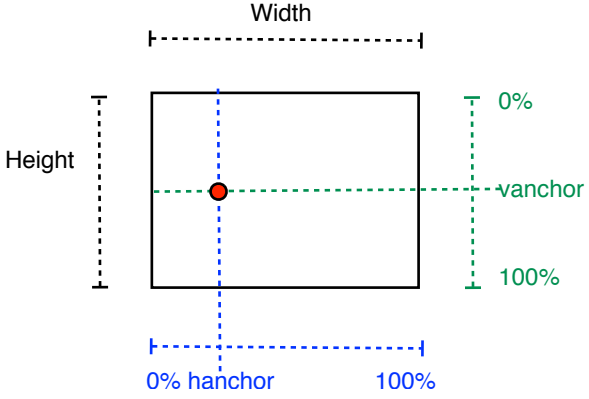
Alignment Properties	
anchor(num h, num v)	Anchor point
hanchor(num h)	H anchor point
vanchor(num v)	V anchor point

Line and Border Properties	
lineWidth(num w)	Line width
lineColor(Color c)	Line color
lineColor(str name)	Line color
fillColor(Color c)	Fill color
fillColor(str name)	Fill color

Shape Properties	
shapeConnected()	Connect vertices
shapeClosed()	Close shape
shapeCurved()	Interpolate vertices

Color	
Color gray(int gray, real alpha)	Gray value (0-255) with transparency (0-100%)
Color gray(real perc)	Gray value (0-100%)
Color gray(real perc, real alpha)	Gray value with transparency
Color color(str colorName)	Named color (from SVG standard) http://www.w3.org/TR/SVG/types.html#ColorKeywords
Color color(str colorName, real alpha)	Named color with transparency
Color rgb(int r, int g, int b)	Color based on red (r), green (g) and blue (b)
Color rgb(int r, int g, int b, real alpha)	RGB color with transparency
list[Color] interpolateColor(Color from, Color to, real perc)	Interpolate between two colors (in RGB space)
list[Color] colorSteps(Color from, Color to, int steps)	List of interpolated colors
Color(num) colorScale(list[num] values, Color from, Color to)	Color scale from list of numbers
list[str] colorNames()	List of color names
Color palette(int n)	Palette of 12 colors

Bounding Box Model



Aliases
alias Color = int;
alias Figures = list[Figure]
alias Edges = list[Edge]
alias Nodes = Figures

Primitives	
text(str s, FProp p ...)	Text string
outline(map[int,Color] cols, FProp p ...)	File outline

Containers	
box(FProp p ...)	box
box(Figure f, FProp p ...)	box with inner figure
ellipse(FProp p ...)	ellipse
ellipse(Figure f, FProp p ...)	ellipse with inner figure
wedge(FProp p ...)	wedge
wedge(Figure f, FProp p ...)	wedge with inner figure
space(FProp p ...)	invisible box
space(Figure f, FProp p ...)	invisible box with inner figure

Figure (Figure)

Composition	
hcat(Figures fs, FProp p ...)	Horizontal concatenation
vcat(Figures fs, FProp p ...)	Vertical concatenation
hvcat(Figures fs, FProp p ...)	Horizontal/vertical composition
overlay(Figure fs, FProp p ...)	Overlay of figures
grid(Figures fs, FProp p ...)	Fixed grid layout
pack(Figures fs, FProp p ...)	Packed layout
graph(Nodes n, Edges E, FProp p ...)	Graph layout
tree(Nodes n, Edges e, FProp p ...)	Tree layout
treemap(Nodes n, Edges e, FProp p ...)	Treemap layout

Transformation	
rotate(real a, Figure f)	Rotate
scale(real perc, Figure f)	Scale x
scale(real xperc,real yperc, Figure f)	Scale

Vertex	
vertex(num x, num y)	(x,y) point
vertex(num x, num y, Figure f)	(x,y) point with figure

Edge	
edge(str from, str to)	edge between named figures
edge(str from, str to, FProp p ...)	edge with properties

Example

```
module Example
import vis::Figure; import vis::Render;
// Display a red box of 100x200
public void main(){
  Figure b = box(size(100,200), fillColor("red"));
  render(b);
}
```

ChartProperties (CProp)		
areaPlot()	linePlot()	vertical()
chartSize(num w, num h)	ring(num h)	xLabel(str txt)
curvePlot()	stackedBars()	yLabel(str txt)
horizontal()	subTitle(str txt)	

Chart

Chart
Figure barChart(str title, map[str,int] facts, CProp cp...)
Figure barChart(str title, NamedNumbers facts, CProp cp...)
Figure barChart(str title, list[str] categories, NamedNumberSeries facts, CProp cp...)
Figure pieChart(str title, map[str,int] facts, CProp cp...)
Figure xyChart(str title, NamedPairSeries facts, CProp cp...)

Aliases
alias NamedNumbers = list[tuple[str name, num val]];
alias NamedPairSeries = list[tuple[str name, list[tuple[num xval, num yval]] values]];
alias NamedNumberSeries = list[tuple[str name, list[num] values]];