What's new in xmCHART 4.0

At a glance

- New graphic engine rewritten from scratch for better performance and image quality.
- Fast hardware accelerated Direct2D rendering on Windows OS.
- Enhanced external functions.
- New chart types: heat maps, tree maps, vector plots.
- Barcodes: QRCode, PDF417, EAN, etc.
- Shading, glow and shadow effects for all charts and graphic elements.
- Enhanced and improved symbol collection.
- Support of round and angled bar ends.
- New alternative line smoothing algorithm.
- Import of Base64 encoded images directly from FileMaker.
- Import and export of high resolution images.
- New and improved export to SVG.
- Improved support for non-western fonts.
- Full Unicode support.
- All code is now 64-bit clean.
- Ready for FileMaker 17, FileMaker 17 Server, FileMaker 17 Go and WebDirect.
- Optimized for Apple's Retina and Windows HiDPI displays.
- Numerous bug fixes.

Requirements:
- FileMaker Pro 12 or higher.
- xmCHART 4.0.10 requires FileMaker Pro 15 or higher.
- macOS 10.7 (Lion) or higher, Windows 7 or higher.
- Windows Server 2008 R2 with Platform Update, Windows Server 2012 or higher.

Table Of Contents

New Chart Types

xmCHART 4 comes with three new chart types: Heat maps, tree maps and vector plots.

Heat Maps

A total of 3 functions are available to set up heat maps.

HeatMap(appearance;numOfRows;numOfCols)
HeatMapOptions(cellGap;cellRounding;cellScaleFactor;doArrangeColumnByColumn)
FillColorScale(seriesIndex;colorScaleID;shadingFlag;numOfColorTones)

HeatMap(appearance;numOfRows;numOfCols)

Arguments	*Type*	*Range*	*Default*	*Notes*
appearance	int	0..127	0	*Constants:* shadow, label
numOfRows	int	1..100000	*automatic*	
numOfCols	int	1..100000	*automatic*	

For example:

```
HeatMap(label+shadow)  /* Heap map with cell labels and shadows.     */
HeatMap(;10)           /* Heap map with 10 rows.                     */
HeatMap(shadow;50;100) /* Heap map with 50 x 100 cells with shadows. */
```

HeatMapOptions(cellGap;cellRounding;cellScaleFactor;doArrangeColumnByColumn)

Arguments	*Type*	*Range*	*Default*	*Notes*
cellGap	num	0..100	0	*Absolute in px or in % of plot area.*
cellRounding	num	0..1	0	*square 0..1 oval*
cellScaleFactor	num	0..1	1	
doArrangeColumnByColumn	int	0..1	off	*Constants: off (0), on (1)*

For example:

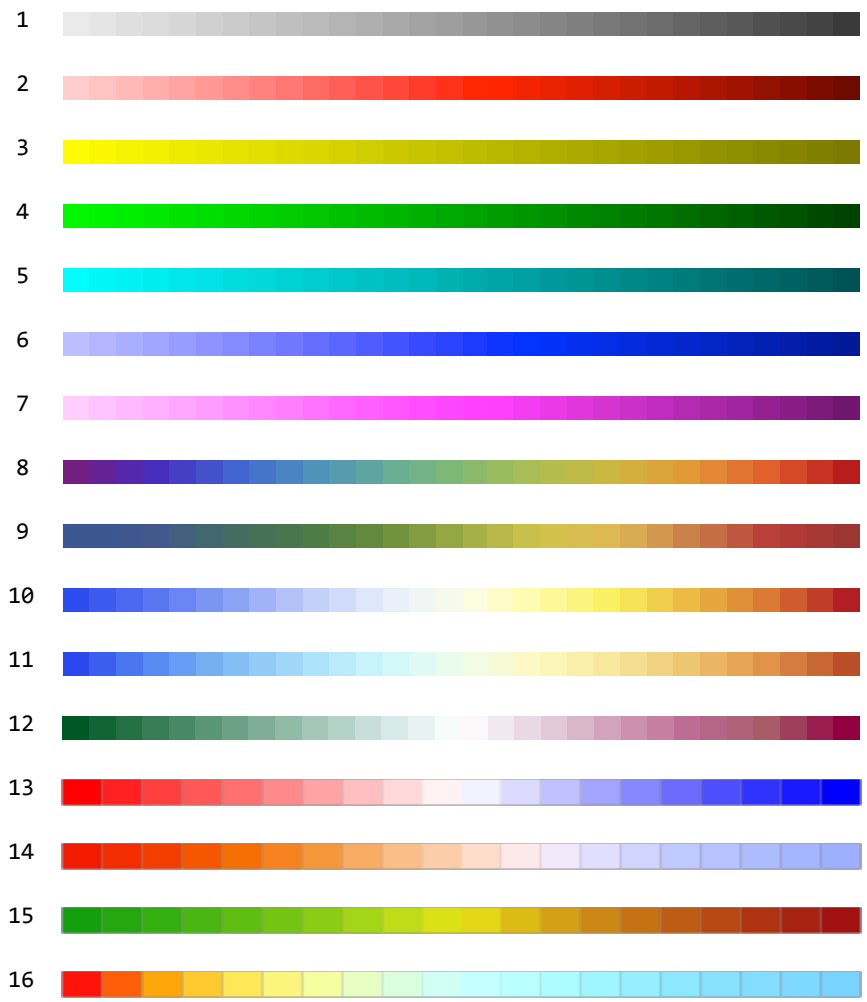
```
HeatMapOptions(2)     /* 2 pixel wide gap.                                          */
HeatMapOptions(0.5%)  /* Gap size is 0.5% of the diagonal of the plot area.         */
HeatMapOptions(;0)    /* Rect cells.                                                */
HeatMapOptions(;0.5)  /* Round rect cells.                                          */
HeatMapOptions(;1)    /* Oval shaped cells.                                         */
HeatMapOptions(;;1)   /* All cells have the same size.                              */
HeatMapOptions(;;0.5) /* The cell with the lowest value has half the size of the max. cell. */
```


FillColorScale(seriesIndex;colorScaleID;shadingFlag;numOfColorTones)

Arguments	*Type*	*Range*	*Default*	*Notes*
seriesIndex	int	0..10000	all	*Constants: all (0) (all series)*
colorScaleID	int	1..20	1	*See table: Color Scales*
shadingFlag	int	-1..0	solid	*Constants: solid (0), shaded (-1)*
numOfColorTones	int	0..100	0	*0...automatic*

For example:

```
FillColorScale(;2)        /* Red color scale.                          */
FillColorScale(;2;;5)     /* Red color scale, reduced to 5 color tones. */
FillColorScale(;2;shaded) /* Red color scale, shaded.                  */
FillColorScale(;8;-1)     /* Rainbow color scale, shaded.              */
```



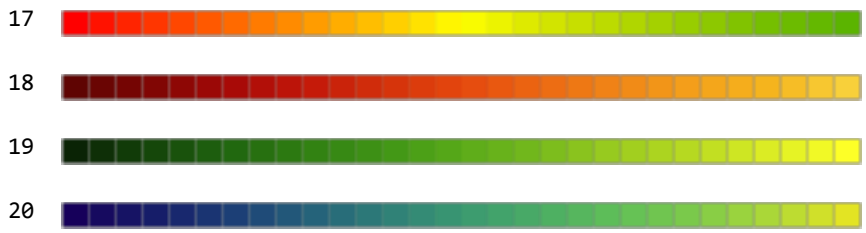
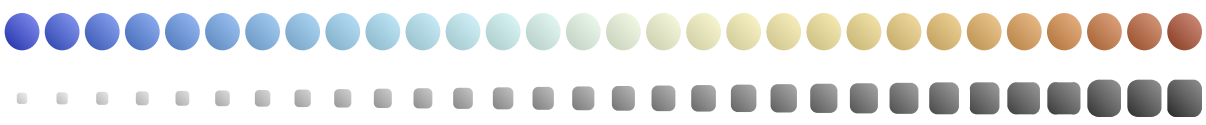


Table: Color Scales

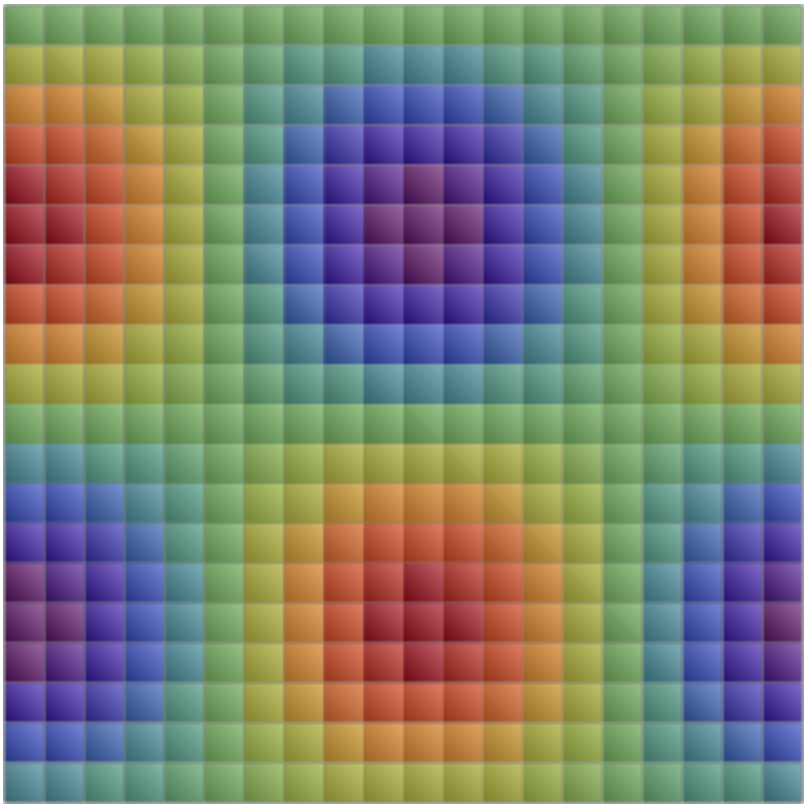
Examples:

```
OpenDrawing(450;15)
  OpenChart(0;0;450;15;on)
    ChartData(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
              16 17 18 19 20 21 22 23 24 25 26 27 28 29 30)
    HeatMap(;1) /* 1 row. */
    FillColorScale(1;8;shaded)
    BorderStyle(1;none)
  CloseChart()
CloseDrawing()
```

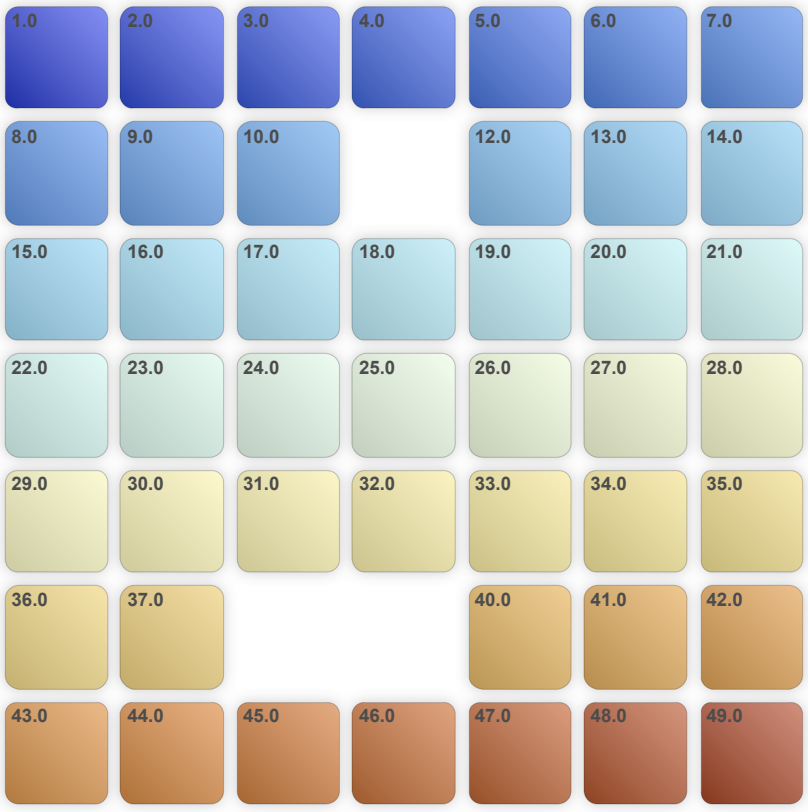
```
OpenDrawing(450;40)
  OpenChart(0;0;450;15;on)
    ChartData(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
              16 17 18 19 20 21 22 23 24 25 26 27 28 29 30)
    HeatMap(;1)         /* 1 row. */
    HeatmapOptions(2;1) /* 2 pixel wide gaps and oval shaped cells. */
    FillColorScale(1;11;shaded)
    BorderStyle(1;none)
  CloseChart()
  OpenChart(0;25;450;15;on)
    ChartData(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
              16 17 18 19 20 21 22 23 24 25 26 27 28 29 30)
    HeatMap(;1)               /* 1 row. */
    HeatmapOptions(2;0.5;0.3) /* 2 pixel wide gaps and scaled cells. */
    FillColorScale(1;1;shaded)
    BorderStyle(1;none)
  CloseChart()
CloseDrawing()
```



```
OpenDrawing(300;300)
  OpenChart(0;0;300;300;on)
    ChartData(10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 13 13 13 12 11 10
9 8 8 7 7 7 8 8 9 10 11 12 13 13 16 16 15 13 12 10 8 7 5 4 4 4 5 7 8 10 12 13 15 16 18 18 17
15 13 10 8 5 3 2 2 2 3 5 8 10 13 15 17 18 20 19 18 16 13 10 7 4 2 1 0 1 2 4 7 10 13 16 18 19
20 20 18 16 13 10 7 4 2 0 0 0 2 4 7 10 13 16 18 20 20 19 18 16 13 10 7 4 2 1 0 1 2 4 7 10 13
16 18 19 18 18 17 15 13 10 8 5 3 2 2 2 3 5 8 10 13 15 17 18 16 16 15 13 12 10 8 7 5 4 4 4 5
7 8 10 12 13 15 16 13 13 13 12 11 10 9 8 8 7 7 7 8 8 9 10 11 12 13 13 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 7 7 8 8 9 10 11 12 13 13 13 13 13 12 11 10 9 8 8 7 4
4 5 7 8 10 12 13 15 16 16 16 15 13 12 10 8 7 5 4 2 2 3 5 8 10 13 15 17 18 18 18 17 15 13 10
8 5 3 2 0 1 2 4 7 10 13 16 18 19 20 19 18 16 13 10 7 4 2 1 0 0 2 4 7 10 13 16 18 20 20 20 18
16 13 10 7 4 2 0 0 1 2 4 7 10 13 16 18 19 20 19 18 16 13 10 7 4 2 1 2 2 3 5 8 10 13 15 17 18
18 18 17 15 13 10 8 5 3 2 4 4 5 7 8 10 12 13 15 16 16 16 15 13 12 10 8 7 5 4 7 7 8 8 9 10 11
12 13 13 13 13 13 12 11 10 9 8 8 7)
    HeatMap()
    FillColorScale(1;8;shaded)
    BorderStyle(1;none)
  CloseChart()
CloseDrawing()
```



```
OpenDrawing(320;320)
  OpenChart(10;10;300;300;on)
    ChartData(1 2 3 4 5 6 7 8 9 10 null 12 13 14 15 16 17 18 19 20
              21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37
              null null 40 41 42 43 44 45 46 47 48 49)
    HeatMap(label+shadow)
    HeatMapOptions(5;0.25;1)
    FillColorScale(1;11;shaded)
    BorderStyle(1;;0.25;0 0 0 60)
    LabelTexts(1;"|f1|")
    LabelStyle(1;Arial;7;bold;60 60 60)
    LabelOptions(1;topLeft)
    ShadowStyle(1;0 0 5;gray)
  CloseChart()
CloseDrawing()
```



Tree Maps

Two functions are available for creating tree maps:

TreeMap(appearance)
TreeMapOptions(subdomainGap)

TreeMap(appearance)

Arguments	*Type*	*Range*	*Default*	*Notes*
appearance	int	0..127	0	*Constants:* shadow, label

The 1st argument appearance makes it possible to add shadow (appearance=shadow) and labels (appearance=label) to the map. Appearance options can be combined by using a plus sign "+".

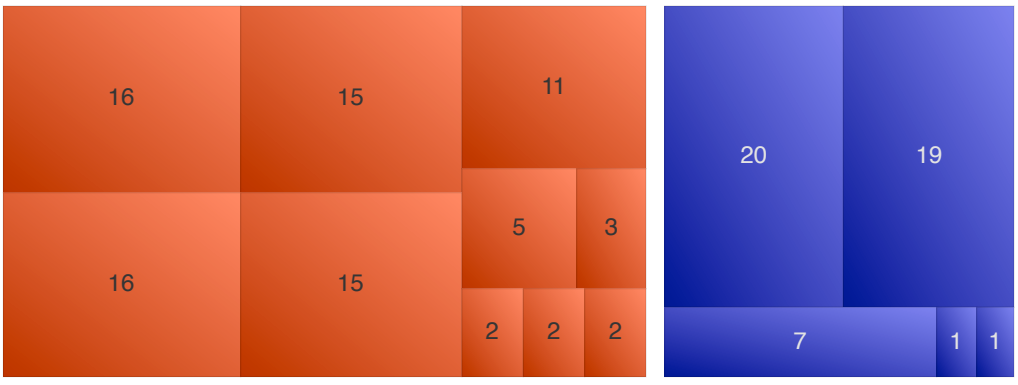
TreeMapOptions(subdomainGap)

Arguments	*Type*	*Range*	*Default*	*Notes*
subdomainGap	num	0..100	0	*Absolute in px or in % of plot area.*

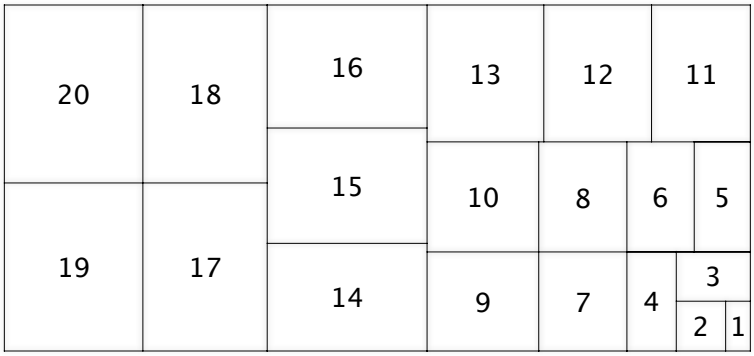
By using the 1st argument subdomainGap the space between data series (subdomains) can be controlled. The subdomain gap can be entered as percentage of the diagonal of the plot area or absolute in pixels.

Examples:

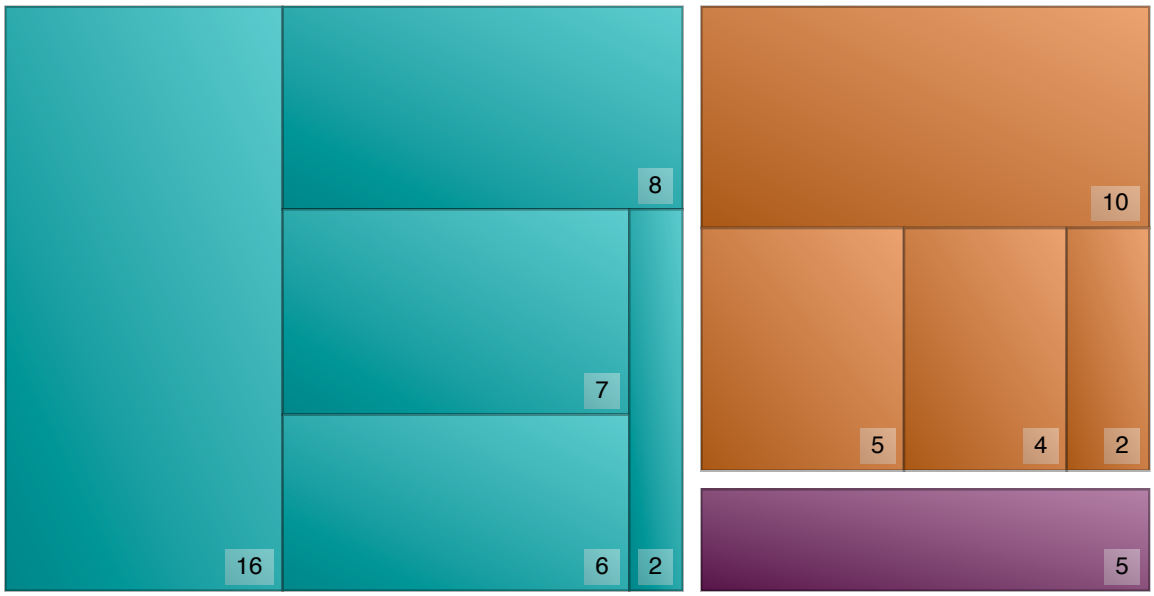
```
OpenDrawing(400;160)
  ChartData(15 16 2 16 15 5 3 11 2 2 ; 20 19 1 1 7)
  TreeMap(label)
  TreeMapOptions(1.5%)
  FillColorScheme(classic;shaded)
  BorderStyle(all;;0.5;0 0 0 30)
  LabelStyle(1;;;;40 40 40)
  LabelStyle(2;;;;220 220 220)
CloseDrawing()
```



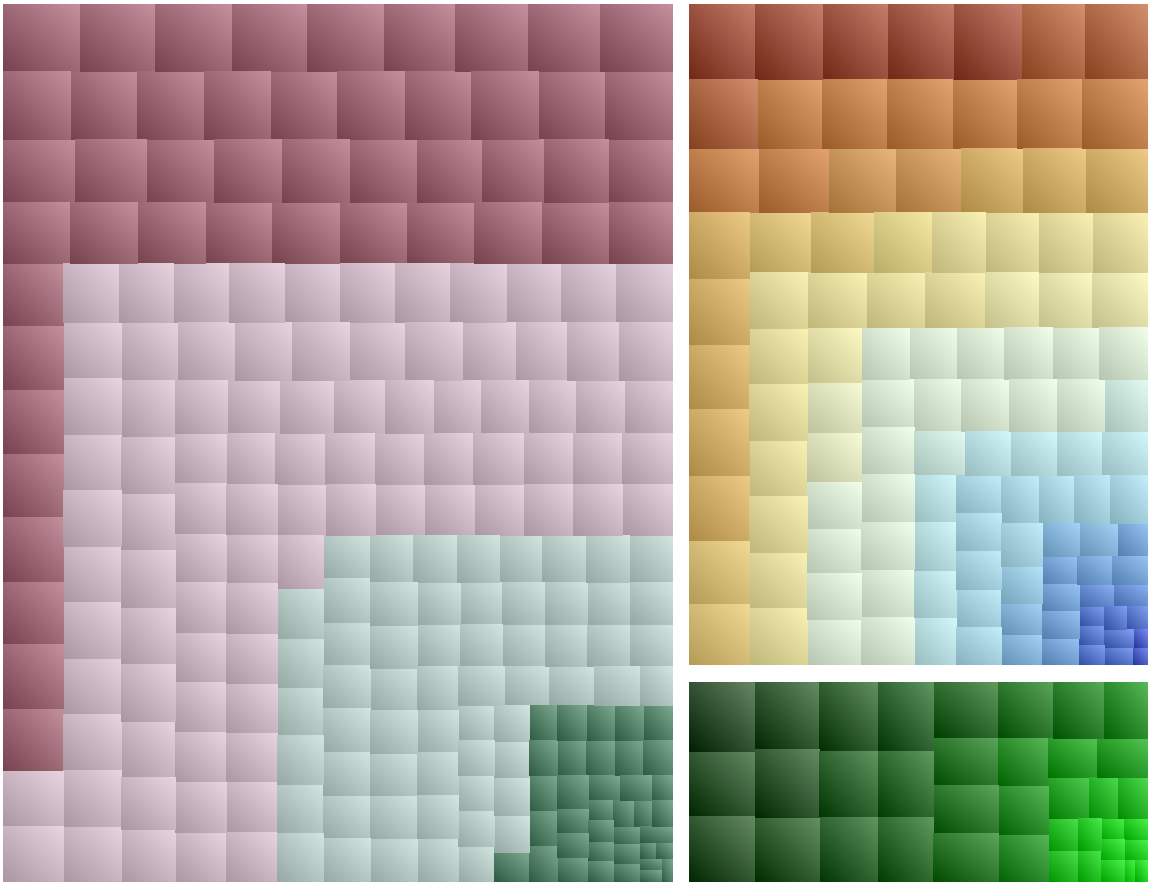
```
OpenDrawing(300;150)
  ChartData(10 1 13 5 12 3 9 18 15 14 4 20 19 17 11 6 16 7 8 2)
  TreeMap(label+shadow)
  FillStyle(all;none)
  BorderStyle(all;;0.25)
  ShadowStyle(all;0 0 2;black)
CloseDrawing()
```




```
OpenDrawing(450;240)
  ChartData(10 4 5 2; 6 16 7 8 2;5)
  TreeMap(label)
  TreeMapOptions(1.25%)
  BorderStyle(all;;1.0;0 0 0 80)
  FillColorScheme(2;shaded)
  LabelBackground(0;255 255 255 100;shaded;0)
  LabelOptions(all;bottomRight;-1;-1)
CloseDrawing()
```



```
/* Tree map with color scale. The function FillColorScale() is discussed in detail in
   combination with heat maps. */

OpenDrawing(450;350)
  ChartData(10 10 10 10 10 10 10 10 10 13 13 13 12 11 10 9 8 8 7 7 7 8 8 9 10 11 12 13 13 16
16 15 13 12 10 8 7 5 4 4 4 5 7 8 10 12 13 15 16 18 18 17 15 13 10 8 5 3 2 2 2 3 5 8 10 13 15
17 18 20 19 18 16 13 10 7 4 2 1 0 1 2 4 7 10 13 16 18 19 20 20 18 16 13 10 7 4 2 0 6 14 0 2
4 7 10 13 16 18 20 20 19 18 16 13 10 7;
4 2 1 0 1 2 4 7 10 13 16 18 19 18 18 17 15 13 10 8 5 3 2 2 2 3 5 8 10 13 15 17 18 16 16 15
13 12 10;
8 7 5 4 4 4 5 7 8 10 12 13 15 16 13 13 13 12 11 10 9 8 8 7 7 7 8 8 9 10 11 12 13 13 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 7 7 8 8 9 10 11 12 13 13 13 13 13 12 11
10 9 8 8 7 4 4 5 7 8 10 12 13 15 16 16 16 15 13 12 10 8 7 5 4 2 2 3 5 8 10 13 15 17 18 18 18
17 15 13 10 8 5 3 2 0 1 2 4 7 10 1 3 16 18 19 20 19 18 16 13 10 7 4 2 1 0 0 2 4 7 10 13 16
18 20 20 20 18 16 13 10 7 4 2 0 0 1 2 4 7 10 13 16 18 19 20 19 18 16 13 10 7 4 2 1 2 2 3 5 8
10 13 15 17 18 18 18 17 15 13 10 8 5 3 2 4 4 5 7 8 10 12 13 15 16 16 16 15 13 12 10 8 7 5 4
7 7 8 8 9 10 11 12 13 13 13 13 13 12 11 10 9 8 8 7)
  TreeMap()
  TreeMapOptions(1%)
  FillColorScale(1;11;shaded)
  FillColorScale(2;4;shaded)
  FillColorScale(3;12;shaded;;4) /* 4 color tones */
  BorderStyle(all;none)
CloseDrawing()
```



Vector Plots

Vector plots can be drawn both as one and two-dimensional. Three functions are available for creating vector plots.

VectorPlot(appearance;doShiftIntervals)
VectorPlot2D(appearance)
VectorPlotOptions(arrowheadType;maxArrowLength;arrowheadSize;arrowAnchorOffset)

VectorPlot(appearance;doShiftIntervals)

Arguments	*Type*	*Range*	*Default*	*Notes*
appearance	int	0..127	0	*Constants:* shadow, label, symbol, horizontal
doShiftIntervals	int	0..1	off	*Constants: off (0), on (1)*

The VectorPlot() function serves to draw one-dimensional vector plots. The 1st data series in the ChartData() function defines the positions, the 2nd data series the vector lengths and the 3rd data series the vector directions of the 1st series, the 4th, 5th and 6th data series in ChartData() the positions, lengths and directions of the 2nd series, and so on. Direction angles are to be entered within a range of ±360 degrees, positive angles run clockwise. 0 degrees is horizontal (3 o'clock).
The 1st argument appearance makes it possible to rotate the plot 90 degrees (appearance=horizontal) and to add symbols (appearance=symbol), shadow (appearance=shadow) and labels (appearance=label) to the vectors. All options can be combined by using a plus sign "+". For one-dimensional vector plots it is possible to move all vectors half an interval width so that they do not lie on the interval borders but rather in the middle of the intervals. This can be done by activating the 2nd argument doShiftIntervals=on.

VectorPlot2D(appearance)

Arguments	*Type*	*Range*	*Default*	*Notes*
appearance	int	0..127	0	*Constants:* shadow, label, symbol

The VectorPlot2D() function makes it possible to draw two-dimensional vector charts. The 1st data series in the ChartData() function defines the x-values, the 2nd data series the y-values, the 3rd data series the vector lengths and the 4th data series the vector directions of the first series, the 5th, 6th, 7th and 8th data series in ChartData() the x-values, y-values, lengths and directions of the 2nd series, and so on.
As for one-dimensional vector charts, symbols (appearance=symbol), shadow (appearance=shadow) and labels (appearance=label) can be added to the vectors by using the argument appearance. The options can be combined by using a plus sign "+". Rotating (appearance=horizontal) is not supported for two-dimensional plots as this is simply possible by switching the data series in ChartData().

VectorPlotOptions(arrowheadType;maxArrowLength;arrowheadSize;arrowAnchorOffset)

Arguments	*Type*	*Range*	*Default*	*Notes*
arrowheadType	int[]	0..4	3	*See table: Arrowhead Types*
maxArrowLength	num	0..1000	10%	*Absolute in px or in % of plot area.*
arrowheadSize	num[]	0..1000	25%	*Absolute in px or in % of arrow length.*
arrowAnchorOffset	num	0..1	0	*tail 0..1 head*

As the 1st argument arrowheadType, it is possible to define different arrowhead types — see table *Arrowhead Types*. By entering two type values you can put arrowheads on both ends. By using the 2nd argument maxArrowLength, the maximum arrow length can be controlled. If no value is defined, the maximum default arrow length equals to 10% of the diagonal of the plot area. In addition, the size of the arrowhead can be controlled with the 3rd argument arrowheadSize. The default arrowhead size is 25% of the arrow length. Both, arrow length and arrowhead size can be entered as percentage of the diagonal of the plot area or absolute in pixels. The 4th argument arrowAnchorOffset can be used to control the anchor point of the arrow. The default anchor point is set to the arrow tail (arrowAnchorOffset=0).

Arrowhead	*Constant*	*Examples*
	0	VectorPlotOptions(0)
	1	VectorPlotOptions(1)
	2	VectorPlotOptions(2)
	3	VectorPlotOptions(3) // Default.
	4	VectorPlotOptions(4)

Table: Arrowhead Types

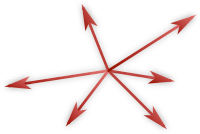
For example:

```
VectorPlotOptions(0)       /* No arrowheads.                                          */
VectorPlotOptions(3)       /* Default arrowheads.                                     */
VectorPlotOptions(1 1)     /* Vertical bars on both ends.                             */
VectorPlotOptions(3 3)     /* Arrowheads on both ends.                                */
VectorPlotOptions(0 3)     /* Reversed arrows.                                        */
VectorPlotOptions(;100)    /* Max. arrow length 100 px.                               */
VectorPlotOptions(;30%)    /* Max. arrow length 30% of the diagonal of the plot area. */
VectorPlotOptions(;;5)     /* Arrowhead size 5 px.                                    */
VectorPlotOptions(;;15%)   /* Arrowhead size 15% of the arrow length.                 */
VectorPlotOptions(;;;0.5)  /* Anchor point is set to the middle of the arrow.         */
VectorPlotOptions(;;;1.0)  /* Anchor point is set to the end of the arrow.            */
```

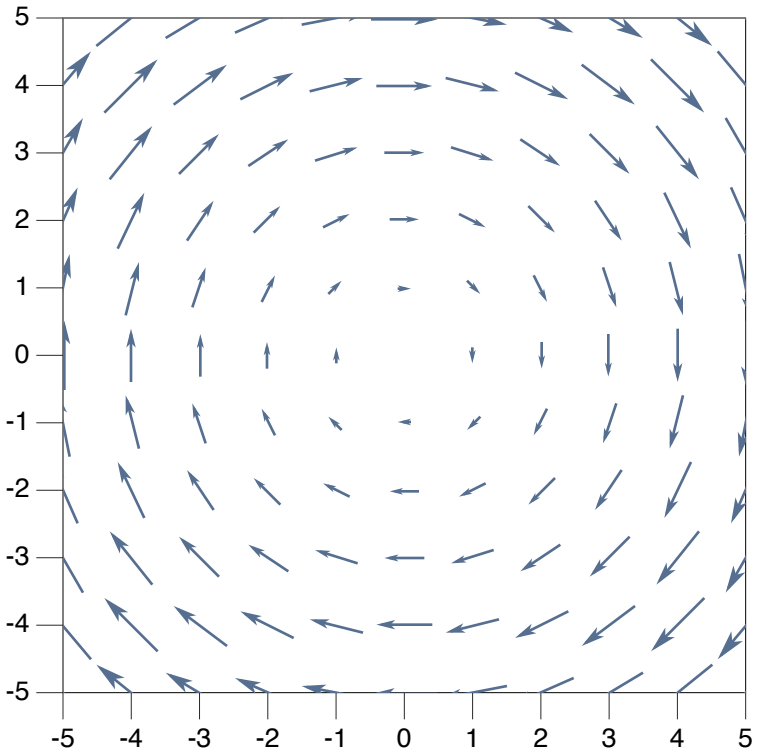

Examples:

```
OpenDrawing(450;22)
  OpenChart(0;6;150;11;on)
    ChartData(1;1;0)
    VectorPlot(symbol)
    VectorPlotOptions(;50%) /* Anchor: arrow tail (default) */
    LineStyle(1;;0.5;90 120 90)
    SymbolStyle(1;bullet;8;1;90 120 90)
    AxisOptions(all;none)
    GridLocation(all;none)
  CloseChart()
  OpenChart(130;6;220;11;on)
    ChartData(1;1;0)
    VectorPlot(symbol)
    VectorPlotOptions(3 3;40%;;0.5) /* Anchor: mid of arrow */
    LineStyle(1;;1;150 120 90)
    SymbolStyle(1;cross;10;1;150 120 90)
    AxisOptions(all;none)
    GridLocation(all;none)
  CloseChart()
  OpenChart(300;6;200;11;on)
    ChartData(1;1;0)
    VectorPlot(symbol)
    VectorPlotOptions(;40%;;1.0) /* Anchor: arrowhead */
    LineStyle(1;;2;86 111 145)
    SymbolStyle(1;vBar;10;1;86 111 145)
    AxisOptions(all;none)
    GridLocation(all;none)
  CloseChart()
CloseDrawing()
```

```
OpenDrawing(80;80)
  OpenChart(0;0;80;80;on)
    ChartDataOptions(xyxy)
    ChartData(10 50 1 10;10 50 1.2 45;10 50 1 127;10 50 1.5 170;10 50 1 -110;10 50 1.5 -32)
    VectorPlot2D(shadow)
    VectorPlotOptions(4;35%;10)
    LineStyle(1;;1;200 0 0;shaded)
    ShadowStyle(1;0 0 2)
    AxisOptions(all;none)
    GridLocation(all;none)
  CloseChart()
CloseDrawing()
```



```
OpenDrawing(300;300)
  ChartData(-5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -5 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -3 -3 -3 -3 -3
-3 -3 -3 -3 -3 -3 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4
4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 ;
-5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2
-1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3
4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4 -3 -2 -1 0 1 2 3 4 5 -5 -4
-3 -2 -1 0 1 2 3 4 5 ;
7.1 6.4 5.8 5.4 5.1 5 5.1 5.4 5.8 6.4 7.1 6.4 5.7 5 4.5 4.1 4 4.1 4.5 5 5.7 6.4 5.8 5 4.2
3.6 3.2 3 3.2 3.6 4.2 5 5.8 5.4 4.5 3.6 2.8 2.2 2 2.2 2.8 3.6 4.5 5.4 5.1 4.1 3.2 2.2 1.4 1
1.4 2.2 3.2 4.1 5.1 5 4 3 2 1 0 1 2 3 4 5 5.1 4.1 3.2 2.2 1.4 1 1.4 2.2 3.2 4.1 5.1 5.4 4.5
3.6 2.8 2.2 2 2.2 2.8 3.6 4.5 5.4 5.8 5 4.2 3.6 3.2 3 3.2 3.6 4.2 5 5.8 6.4 5.7 5 4.5 4.1 4
4.1 4.5 5 5.7 6.4 7.1 6.4 5.8 5.4 5.1 5 5.1 5.4 5.8 6.4 7.1 ;
-135 -129 -121 -112 -101 -90 -79 -68 -59 -51 -45 -141 -135 -127 -117 -104 -90 -76 -63 -53
-45 -39 -149 -143 -135 -124 -108 -90 -72 -56 -45 -37 -31 -158 -153 -146 -135 -117 -90 -63
-45 -34 -27 -22 -169 -166 -162 -153 -135 -90 -45 -27 -18 -14 -11 180 180 180 180 180 0 0 0 0
0 0 169 166 162 153 135 90 45 27 18 14 11 158 153 146 135 117 90 63 45 34 27 22 149 143 135
124 108 90 72 56 45 37 31 141 135 127 117 104 90 76 63 53 45 39 135 129 121 112 101 90 79 68
59 51 45)
  VectorPlot2D()
  VectorPlotOptions(4 0;;35%;0.5)
  LineStyle(1;;1;86 111 145)
  AxisLine(all;0)
  AxisMajorTicks(all;10;0.25;40 40 40;;out)
  MajorGridLineWidths(all;all;0)
  GridFrame(all;0.25)
CloseDrawing()
```



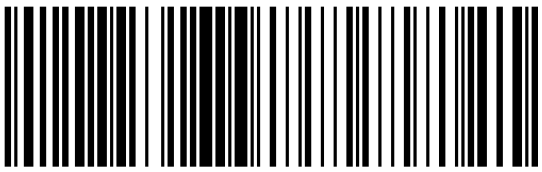
Barcodes

The following 8 barcode functions have been added in xmCHART 4:

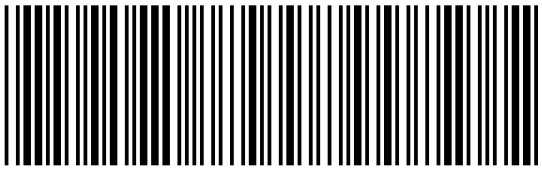
```
Barcode128(left;top;width;height;text;color)
Barcode39(left;top;width;height;text;color;checksumFlag)
Barcode93(left;top;width;height;text;color)
BarcodeCodabar(left;top;width;height;text;color)
BarcodeEAN(left;top;width;height;text;color)
BarcodeI25(left;top;width;height;text;color) /* Interleaved 2 of 5 barcode. */
BarcodePDF417(left;top;width;height;text;color)
BarcodeQR(left;top;width;height;text;color;errorCorrection)
```

Examples:

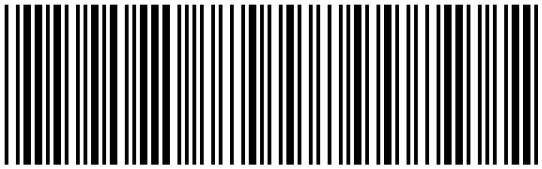
```
OpenDrawing(240;100;print)
  Barcode128(20;20;200;60;"0123456789ABCDEF")
CloseDrawing()
```



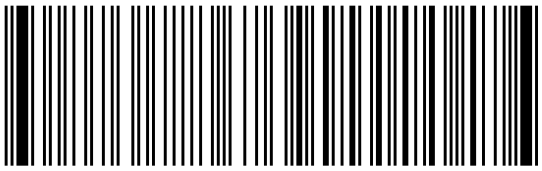
```
OpenDrawing(240;100;print)
  Barcode39(20;20;200;60;"abc123")
CloseDrawing()
```



```
OpenDrawing(240;100;print)
  /* checksumFlag: 0...off (default) */
  /*               1...on            */
  Barcode39(20;20;200;60;"abc123";black;on) /* Code 39 with checksum. */
CloseDrawing()
```



```
OpenDrawing(240;100;print)
  Barcode93(20;20;200;60;"0123456789ABCDEF")
CloseDrawing()
```



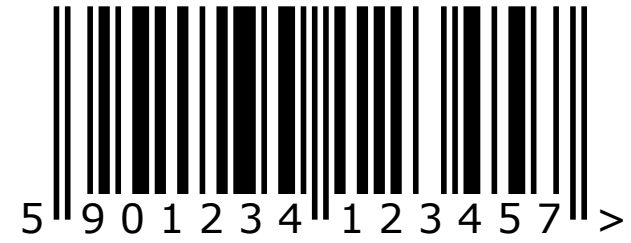
```
OpenDrawing(240;100;print)
  BarcodeCodabar(20;20;200;60;"31117013206375")
  AddText(120;95;"3 1117 01320 6375";Verdana;15;plain;black;center)
CloseDrawing()
```

The BarcodeEAN() function accepts code with and without checksum digit. For example:

```
BarcodeEAN(10;10;200;80;"01234565")      /* EAN-8 with checksum digit (5).                */
BarcodeEAN(10;10;200;80;"0123456")       /* EAN-8 checksum digit automatically generated.  */
BarcodeEAN(10;10;200;80;"3652241713774") /* EAN-13 with checksum digit (4).               */
BarcodeEAN(10;10;200;80;"365224171377")  /* EAN-13 checksum digit automatically generated. */
```

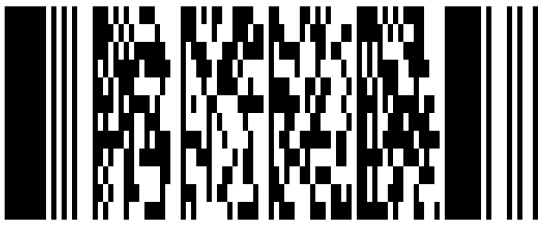
For more on calculating the checksum digit, see for example:
https://en.wikipedia.org/wiki/International_Article_Number_(EAN)

```
OpenDrawing(240;100;print)
  BarcodeEAN(20;10;200;80;"5901234123457")
  AddText(7;95;"5";Verdana;15)
  AddRect(26;80;90;20;white)
  AddText(30;95;"9 0 1 2 3 4";Verdana;15)
  AddRect(123;80;90;20;white)
  AddText(127;95;"1 2 3 4 5 7";Verdana;15)
  AddText(223;95;">";Verdana;15)
CloseDrawing()
```




```
OpenDrawing(240;100;print)
  BarcodeI25(20;20;200;60;"0123456789") /* Interleaved 2 of 5 barcode. */
  AddText(120;95;"0 1 2 3 4 5 6 7 8 9";Verdana;16;bold;black;center)
CloseDrawing()
```

```
OpenDrawing(240;100;print)
  BarcodePDF417(20;10;200;80;"http://www.xmchart.com")
CloseDrawing()
```



```
OpenDrawing(400;140;print)
  BarcodePDF417(20;10;360;120;
"Source: https://en.wikipedia.org/wiki/PDF417
PDF417 is a stacked linear barcode symbol format used in a variety of applications,
primarily transport, identification cards, and inventory management. PDF stands for Portable
Data File. The 417 signifies that each pattern in the code consists of 4 bars and spaces,
and that each pattern is 17 units long. The PDF417 symbology was invented by Dr. Ynjiun P.
Wang at Symbol Technologies in 1991. (Wang 1993) It is represented by ISO standard 15438.
PDF417 is one of the formats (along with Data Matrix) that can be used to print postage
accepted by the United States Postal Service. PDF417 is also selected by the airline
industry's Bar Coded Boarding Pass standard (BCBP) as the 2D bar code symbolism for paper
boarding passes. PDF417 is the standard selected by the Department of Homeland Security as
the machine readable zone technology for RealID compliant driver licenses and state issued
identification cards. It is also used by FedEx on package labels.")
CloseDrawing()
```

```
OpenDrawing(120;120;print)
  BarcodeQR(10;10;100;100;"What's new in xmCHART 4.0";mediumVioletRed)
CloseDrawing()
```

```
OpenDrawing(120;120;print)
  BarcodeQR(10;10;100;100;”http://www.xmchart.com”;black;2) /* Error correction = 2 (Mid) */
CloseDrawing()
```

```
/* Note: Some scanners will fail reading inverted (white on black) QR codes. */
OpenDrawing(120;120;print)
  OpenView(5;5;110;110)
    BarcodeQR(5;5;100;100;"こんにちは世界 (Hello World)";white)
    Background(black)
  CloseView()
CloseDrawing()
```

Using QR code for vCards

For more information about vCard file format, see for example:
https://en.wikipedia.org/wiki/VCard
http://tools.ietf.org/search/rfc6350

Example:

```
OpenDrawing(220;220;print)
  BarcodeQR(10;10;200;200;"
BEGIN:VCARD
VERSION:2.1
N:Gump;Forrest
FN:Forrest Gump
ORG:Bubba Gump Shrimp Co.
TITLE:Shrimp Man
PHOTO;GIF:http://www.example.com/dir_photos/my_photo.gif
TEL;WORK;VOICE:(111) 555-1212
TEL;HOME;VOICE:(404) 555-1212
ADR;WORK:;;100 Waters Edge;Baytown;LA;30314;United States of America
LABEL;WORK;ENCODING=QUOTED-PRINTABLE:100 Waters Edge=0D=0ABaytown, LA 30314=0D=0AUnited
States of America
ADR;HOME:;;42 Plantation St.;Baytown;LA;30314;United States of America
LABEL;HOME;ENCODING=QUOTED-PRINTABLE:42 Plantation St.=0D=0ABaytown, LA 30314=0D=0AUnited
States of America
EMAIL;PREF;INTERNET:forrestgump@example.com
REV:20080424T195243Z
END:VCARD")
CloseDrawing()
```

Shading, Glow and Shadow Effects

xmCHART 4 supports shading, glow and shadow effects for all charts and graphic elements.

Shading

The argument `pattern` — part of numerous xmCHART functions — has been enhanced in xmCHART 4 by two constants: `solid` (0) and `shaded` (-1). The argument name `pattern` has been renamed in xmCHART 4 to `colorVariant` to better reflect the newly added options.

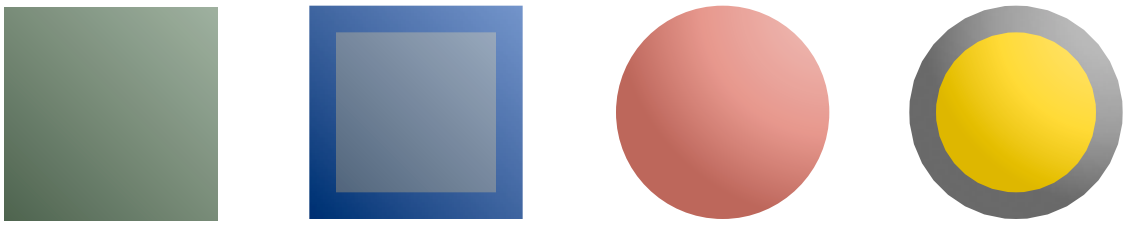
Constant	*Name*	*Notes*
-1	shaded	New in xmCHART 4
0	solid	New in xmCHART 4 (default)
1	transparent	Same as in xmCHART 3
2..64		Same as in xmCHART 3 (black&white patterns)
65..128		Same as in xmCHART 3 (color patterns)

Table: Color Variants

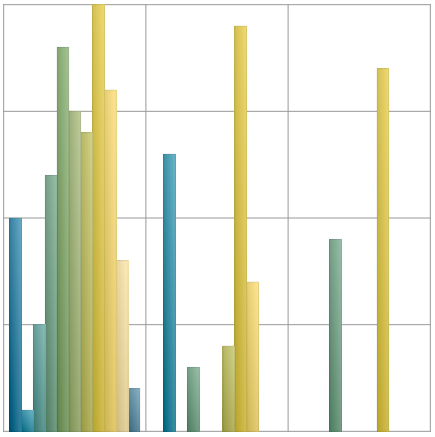
Note:
Shading is ignored in combination with color gradients.

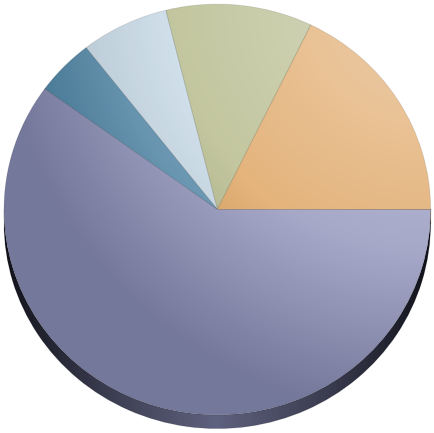
Examples:

```
OpenDrawing(450;100)
  AddRect( 20;10;80;80;90 120 90;shaded)
  AddRect(135;10;80;80;86 111 145;shaded;10;0 51 153;shaded)
  AddOval(250;10;80;80;215 101 90;shaded)
  AddOval(360;10;80;80;darkYellow;shaded;10;110 110 110;shaded)
CloseDrawing()
```



```
OpenDrawing(450;180)
  OpenChart(35;10;160;160;on)
    ChartData(10; 1 13; 5; 12 3 9; 18; 15; 14 4; 20 19 17; 16 7; 8; 2)
    BarChart()
    BorderStyle(all;;0.25;0 0 0 25)
    FillColorScheme(20;shaded)
    AxisOptions(all;none) /* Hide axes. */
    MajorGridLineWidths(all;all;0.25)
  CloseChart()
  OpenChart(250;10;160;160;on)
    ChartData(2 3 5 8 27)
    PieChart(;6.5;;306)
    BorderStyle(all;;0.25;0 0 0 25)
    FillColorScheme(1;shaded)
  CloseChart()
CloseDrawing()
```





Glow and Shadow Effects

The argument `shadowOffset` — part of the `ShadowStyle()` function and background functions — can now have up to three values, i.e. the shadow offsets in x and y-direction and the blur radius. By varying these values subtle shadow and glow effects can be achieved. Furthermore, glow and shadow effects can be added to all texts and graphic elements. Examples:

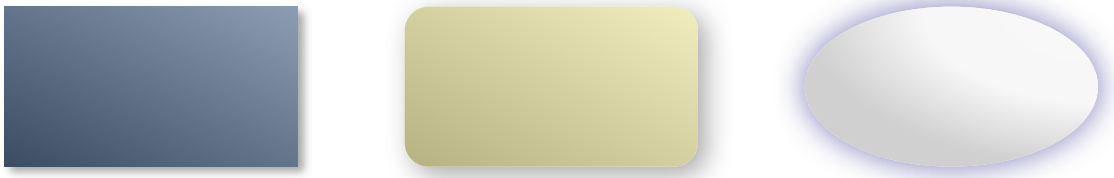
```
ShadowStyle(all;3 3 3;136 136 136 170) /* Default settings. */
ShadowStyle(all;2 2 4;gray)
LabelBackground(1;white;;0;;;0 0 10;red) /* Set offsets 0 0 to achieve a glow effect. */
AxisLabelStyle(x;Arial;15;bold;40 40 40;;0;;;;0 0 5;yellow)
AddLine(10;10;100;100;2.5;darkRed;;0 0 3)
```

Notes:

- To achieve a glowing effect set the offsets in x and y-direction to 0.
- The argument `shadowPattern` available in previous versions is ignored in xmCHART 4.
- The argument name `shadowOffset` has been renamed in xmCHART 4 to `shadowEffect`.
  For example:
  `xmCHART 3: ShadowStyle(seriesIndex;shadowOffset;shadowColor;shadowPattern)`
  `xmCHART 4: ShadowStyle(seriesIndex;shadowEffect;shadowColor)`

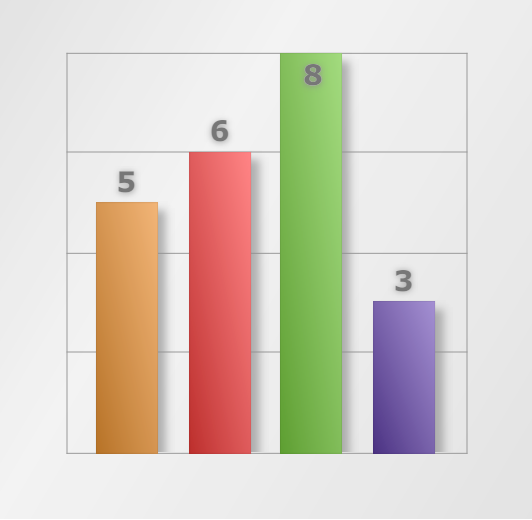
Examples:

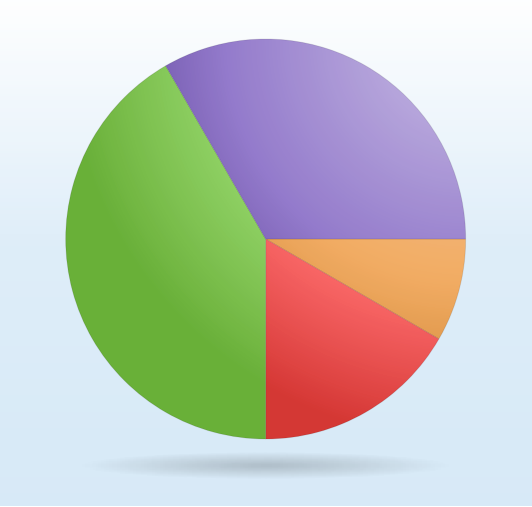
```
OpenDrawing(450;100)
  AddRect( 20;20;110;60;86 111 145;shaded;0;;;2 2)
  AddRoundRect(170;20;110;60;18;18;paleGoldenRod;shaded;0;;;3 3 10)
  AddOval(320;20;110;60;whiteSmoke;shaded;0;;;0 0 12;darkBlue) /* Glow effect. */
CloseDrawing()
```



```
OpenDrawing(400;205)
  OpenView(0;0;160;210)
    AddPath(M 140 190             /* Move to: xPt yPt */
            L 140  20             /* Line to: xPt yPt */
            L  70  20             /* Line to: xPt yPt */
            A  20  70 50 50 0 0 0 /* Arc to: xEndPt yEndPt xRadius yRadius rotation flags */
            L  20 190             /* Line to: xPt yPt */
            Z                     /* Close (outer path) */
            M  40 170 L 60 170 L 60 150 L 40 150 Z                /* Punch out square. */
            M  70 160 A 90 160 10 10 0 0 0 A 70 160 10 10 0 0 0 z /* Punch out circle. */
            M 100 150 L 120 150 L 110 170 Z;                      /* Punch out triangle. */
            greenYellow;shaded;   /* Fill:   color ; colorVariant         */
            2.5;slateBlue;shaded; /* Border: width ; color ; colorVariant */
            2 2 5)                /* Shadow: xOffset yOffset blurRadius   */
  CloseView()
  AddText(170;170;"こんにちは世界";ヒラギノ角ゴ Pro W3;18.5;3;slateBlue;;;;;;;1 1 2;gray)
  AddText(170;190;"Text with shadow effect";Times New Roman;19;3;slateBlue;;;;;;;1 1 2;gray)
CloseDrawing()
```

```
OpenDrawing(450;200)
  OpenView(10;0;200;195)
    OpenChart(25;20;150;150;on)
      ChartData(5;6;8;3)
      BarChart(shadow+label;;50)
      BorderStyle(all;;0.25;0 0 0 20)
      FillStyle(1; 232 133  41;shaded) /* Orange */
      FillStyle(2; 254  42  50;shaded) /* Red */
      FillStyle(3; 108 200  52;shaded) /* Green */
      FillStyle(4;  89  50 172;shaded) /* Indigo */
      AxisOptions(all;none)            /* Hide axes. */
      MajorGridLineWidths(all;all;0.25)
      LabelStyle(all;Verdana;10.5;bold;100 100 100;center;;;;;0 0 2)
    CloseChart()
    Background(1 30 1
               0.0  220 220 220 255
               0.35 240 240 240 255
               1.0  220 220 220 255;;0)
  CloseView()

  OpenView(250;0;200;195)
    OpenChart(25;20;150;150;on)
      ChartData(1 2 5 4)
      PieChart(;;;90)
      BorderStyle(all;;0.25;0 0 0 25)
      FillStyle(1; 232 133  41;shaded) /* Orange */
      FillStyle(2; 254  42  50;shaded) /* Red */
      FillStyle(3; 108 200  52;shaded) /* Green */
      FillStyle(4;  89  50 172;shaded) /* Indigo */
    CloseChart()
    AddOval(30;175;140;10;2 0.5 0.5
                          0 0 0 0 50
                          1 0 0 0 0) /* Shadow */
    Background(1 90 0
               0.0 255 255 255 255
               0.5 214 232 246 255
               1.0 206 228 245 255;;0)
  CloseView()
CloseDrawing()
```





Symbols

A total of 40 new symbols — constants 19 to 58 — are available in xmCHART 4. Furthermore, an optional background color (attributes `backgroundColor` and `backgroundColorVariant`) and shadow (attributes `shadowEffect` and `shadowColor`) have been added to all symbol functions.

```
AddSymbol(x;y;type;size;stroke;color;colorVariant;backgroundColor;backgroundColorVariant;
          shadowEffect;shadowColor)
SymbolStyle(seriesIndex;type;size;stroke;color;colorVariant;backgroundColor;
          backgroundColorVariant)
DropLineReferencePoint(seriesIndex;xCenter;yCenter;symbolType;symbolSize;lineStroke;
          symbolColor;symbolColorVariant;symbolBackgroundColor;symbolBackgroundColorVariant;
          shadowEffect;shadowColor)
```

Constant		*Name*	*Example*
	0	none	SymbolStyle(1;none)
	1	cross	AddSymbol(10;10;cross;12;2;40 40 40)
	2	bullet	AddSymbol(10;10;bullet;12;1;40 40 40)
	3	square	AddSymbol(10;10;square;12;1;40 40 40)
	4	diamond	AddSymbol(10;10;diamond;12;1;40 40 40)
	5	downTriangle	AddSymbol(10;10;downTriangle;12;1;40 40 40)
	6	upTriangle	AddSymbol(10;10;upTriangle;12;1;40 40 40)
	7	plus	AddSymbol(10;10;plus;12;2;40 40 40)
	8	circle	AddSymbol(10;10;circle;12;2;40 40 40)
	9	hollowSquare	AddSymbol(10;10;hollowSquare;12;2;40 40 40)
	10	hollowDiamond	AddSymbol(10;10;hollowDiamond;12;2;40 40 40)
	11	hollowDownTriangle	AddSymbol(10;10;hollowDownTriangle;12;2;40 40 40)
	12	hollowUpTriangle	AddSymbol(10;10;hollowUpTriangle;12;2;40 40 40)
	13	hBar	AddSymbol(10;10;hBar;12;2;40 40 40)
	14	vBar	AddSymbol(10;10;vBar;12;2;40 40 40)
	15	leftBar	AddSymbol(10;10;leftBar;12;2;40 40 40)
	16	rightBar	AddSymbol(10;10;rightBar;12;2;40 40 40)
	17	topBar	AddSymbol(10;10;topBar;12;2;40 40 40)
	18	bottomBar	AddSymbol(10;10;bottomBar;12;2;40 40 40)

Constant		Name	Example
	19	star3	AddSymbol(10;10;star3;12;1;40 40 40)
	20	star4	AddSymbol(10;10;star4;12;1;40 40 40)
	21	star5	AddSymbol(10;10;star5;12;1;40 40 40)
	22	star6	AddSymbol(10;10;star6;12;1;40 40 40)
	23	star7	AddSymbol(10;10;star7;12;1;40 40 40)
	24	star8	AddSymbol(10;10;star8;12;1;40 40 40)
	25	spokes3	AddSymbol(10;10;spokes3;12;1.5;40 40 40)
	26	spokes4	AddSymbol(10;10;spokes4;12;1.5;40 40 40)
	27	spokes5	AddSymbol(10;10;spokes5;12;1.5;40 40 40)
	28	spokes6	AddSymbol(10;10;spokes6;12;1.5;40 40 40)
	29	spokes7	AddSymbol(10;10;spokes7;12;1.5;40 40 40)
	30	spokes8	AddSymbol(10;10;spokes8;12;1.5;40 40 40)
	31	downTriangleHalfLeft	AddSymbol(10;10;downTriangleHalfLeft;12;1;40 40 40)
	32	downTriangleHalfRight	AddSymbol(10;10;downTriangleHalfRight;12;1;40 40 40)
	33	upTriangleHalfLeft	AddSymbol(10;10;upTriangleHalfLeft;12;1;40 40 40)
	34	upTriangleHalfRight	AddSymbol(10;10;upTriangleHalfRight;12;1;40 40 40)
	35	circleHalfTop	AddSymbol(10;10;circleHalfTop;12;1;40 40 40)
	36	circleHalfRight	AddSymbol(10;10;circleHalfRight;12;1;40 40 40)
	37	circleHalfBottom	AddSymbol(10;10;circleHalfBottom;12;1;40 40 40)
	38	circleHalfLeft	AddSymbol(10;10;circleHalfLeft;12;1;40 40 40)
	39	circleHalfTopRight	AddSymbol(10;10;circleHalfTopRight;12;1;40 40 40)
	40	circleHalfBottomRight	AddSymbol(10;10;circleHalfBottomRight;12;1;40 40 40)
	41	circleHalfBottomLeft	AddSymbol(10;10;circleHalfBottomLeft;12;1;40 40 40)
	42	circleHalfTopLeft	AddSymbol(10;10;circleHalfTopLeft;12;1;40 40 40)
	43	squareHalfTop	AddSymbol(10;10;squareHalfTop;12;1;40 40 40)
	44	squareHalfRight	AddSymbol(10;10;squareHalfRight;12;1;40 40 40)
	45	squareHalfBottom	AddSymbol(10;10;squareHalfBottom;12;1;40 40 40)
	46	squareHalfLeft	AddSymbol(10;10;squareHalfLeft;12;1;40 40 40)
	47	squareHalfTopRight	AddSymbol(10;10;squareHalfTopRight;12;1;40 40 40)

Constant	Name	Example
48	squareHalfBottomRight	AddSymbol(10;10;squareHalfBottomRight;12;1;40 40 40)
49	squareHalfBottomLeft	AddSymbol(10;10;squareHalfBottomLeft;12;1;40 40 40)
50	squareHalfTopLeft	AddSymbol(10;10;squareHalfTopLeft;12;1;40 40 40)
51	diamondHalfTop	AddSymbol(10;10;diamondHalfTop;12;1;40 40 40)
52	diamondHalfRight	AddSymbol(10;10;diamondHalfRight;12;1;40 40 40)
53	diamondHalfBottom	AddSymbol(10;10;diamondHalfBottom;12;1;40 40 40)
54	diamondHalfLeft	AddSymbol(10;10;diamondHalfLeft;12;1;40 40 40)
55	diamondHalfTopRight	AddSymbol(10;10;diamondHalfTopRight;12;1;40 40 40)
56	diamondHalfBottomRight	AddSymbol(10;10;diamondHalfBottomRight;12;1;40 40 40)
57	diamondHalfBottomLeft	AddSymbol(10;10;diamondHalfBottomLeft;12;1;40 40 40)
58	diamondHalfTopLeft	AddSymbol(10;10;diamondHalfTopLeft;12;1;40 40 40)

Table: Symbols

Examples:

```
OpenDrawing(450;60)
  AddSymbol( 25;30;spokes4;16;3.5;153 89 91;solid;;;0 0 2)
  AddSymbol( 75;30;upTriangleHalfRight;16;2;110 110 110;shaded;;;0 0 2)
  AddSymbol(125;30;squareHalfBottom;16;2;150 120 90;shaded;;;1 1 2)
  AddSymbol(175;30;diamondHalfRight;16;2;90 120 90;shaded;;;1 1 2)
  AddSymbol(225;30;star3;16;2;86 111 145;solid;;;0 0 2)
  AddSymbol(275;30;bullet;16;1;215 101 90;shaded;;;0 0 5;brown) /* Glow effect */
  AddSymbol(325;30;circleHalfTopRight;16;1.5;110 110 110;solid;;;0 0 2)

  /* Combined symbol */
  AddSymbol(375;30;circle;16;2.0;90 120 90;solid;;;0 0 2)
  AddSymbol(375;30;spokes3;16;1.5;90 120 90;solid;;;0 0 2)

  /* Combined symbol */
  AddSymbol(425;30;circleHalfTopRight;16;2;86 111 145;shaded)
  AddSymbol(425;30;circleHalfBottomRight;16;2;86 111 145;shaded)
CloseDrawing()
```

Graphics Primitives

To all graphics primitive functions (except `AddPicture()`) two shadow arguments — `shadowEffect` and `shadowColor` — have been added. For more details see section Shading. Furthermore, all "fill" functions, i.e. `AddRect()`, `AddRoundRect()`, `AddOval()`, `AddSlice()`, `AddPolygon()` and `AddSmoothPolygon()` have been enhanced with 3 border arguments: `bordersStroke`, `borderColor` and `borderColorVariant`. This makes it possible to define the fill, border and shadow effect in one single function.

Function	*Arguments*
AddArc	left;top;width;height;startAngle;arcAngle;lineStroke;lineColor; lineColorVariant;shadowEffect;shadowColor
AddArrow	xStart;yStart;xEnd;yEnd;lineStroke;lineColor;lineColorVariant; headLocation;headLength;headWidth;headInset;hasHollowHead;shadowEffect; shadowColor
AddEllipse	left;top;width;height;lineStroke;lineColor;lineColorVariant;shadowEffect; shadowColor
AddFrame	left;top;width;height;frameStroke;frameColor;frameColorVariant; shadowEffect;shadowColor
AddLine	xStart;yStart;xEnd;yEnd;lineStroke;lineColor;lineColorVariant; shadowEffect;shadowColor
AddOval	left;top;width;height;fillColor;fillColorVariant;borderStroke; borderColor;borderColorVariant;shadowEffect;shadowColor
AddPath	pathData;fillColor;fillColorVariant;borderStroke;borderColor; borderColorVariant;shadowEffect;shadowColor
AddPolygon	scanDirection;listOfCoords;fillColor;fillColorVariant;borderStroke; borderColor;borderColorVariant;shadowEffect;shadowColor
AddPolyline	scanDirection;listOfCoords;lineStroke;lineColor;lineColorVariant; shadowEffect;shadowColor
AddRect	left;top;width;height;fillColor;fillColorVariant;borderStroke; borderColor;borderColorVariant;shadowEffect;shadowColor
AddRoundFrame	left;top;width;height;xRadius;yRadius;frameStroke;frameColor; frameColorVariant;shadowEffect;shadowColor
AddRoundRect	left;top;width;height;xRadius;yRadius;fillColor;fillColorVariant; borderStroke;borderColor;borderColorVariant;shadowEffect;shadowColor
AddSlice	left;top;width;height;startAngle;arcAngle;innerRadius;fillColor; fillColorVariant;borderStroke;borderColor;borderColorVariant; shadowEffect;shadowColor
AddSmoothPolygon	scanDirection;listOfCoords;fillColor;fillColorVariant;borderStroke; borderColor;borderColorVariant;shadowEffect;shadowColor;smoothingMethod

Function	*Arguments*
AddSmoothPolyline	scanDirection;listOfCoords;lineStroke;lineColor;lineColorVariant; shadowEffect;shadowColor;smoothingMethod
AddSymbol	xPosition;yPosition;symbolType;symbolSize;lineStroke;symbolColor; symbolColorVariant;symbolBackgroundColor;symbolBackgroundColorVariant; shadowEffect;shadowColor
AddText	xPosition;yPosition;text;font;size;style;color;hAlignment;vAlignment; orientation;maxWidth;maxHeight;ellipsisPosition;shadowEffect;shadowColor

Table: Graphics Primitives

Enhanced `AddPath()` function

In addition to the path command IDs 1 to 10, easy to remember path command letters have been added in xmCHART 4, see table *Path Commands*. The command letters are case sensitive — use uppercase letters for absolute coordinate values, lowercase letters for coordinates relative to the current point.

Command	*ID*	*Absolute Command Letter*	*Relative Command Letter*
Close path	1	Z	z *(same as Z)*
Move To	2	M	m
Line To	3	L	l
Quadratic Bézier To	4	Q	q
Cubic Bézier To	5	C	c
Elliptical arc To	6	A	a
Horizontal line to	7	H	h
Vertical line to	8	V	v
Shorthand quad. Bézier	9	T	t
Shorthand cubic Bézier	10	S	s

Table: Path Commands

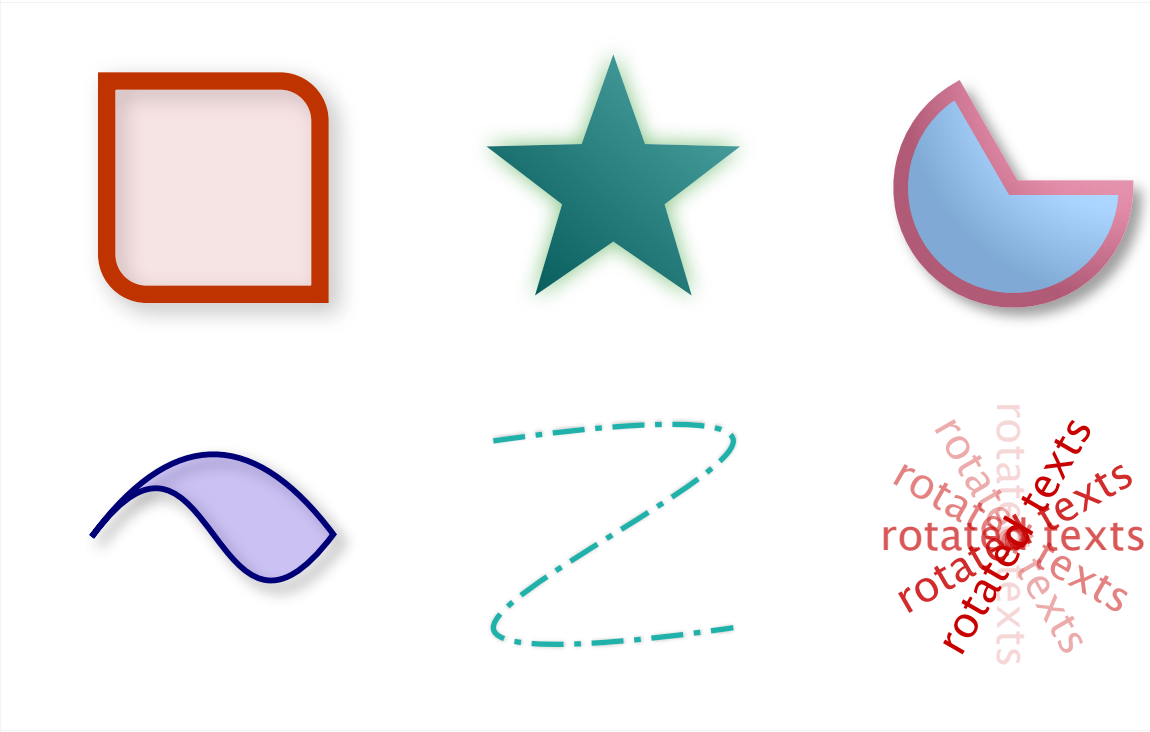
Examples;

```
// All 5 functions give the same result.
AddPath(2 100 100 3 150 100 3 100 120 1)
AddPath(M 100 100 L 150 100 L 100 120 Z)
AddPath(M 100 100 l  50   0 l -50  20 Z)
AddPath(M 100 100 l  50   0 l -50  20 z)
AddPath(m 100 100 L 150 100 l -50  20 z)
```

See also the path example in section Shading.

Examples:

```
OpenDrawing(450;260)
  AddPath(M  35  25              /* Move to:  35  25 */
          l   0  65              /* Line to:  35  90 */
          a  15  15 15 15 0 0 0  /* Arc to:   50 105 */
          l  65   0              /* Line to: 115 105 */
          l   0 -65              /* Line to: 115  40 */
          a -15 -15 15 15 0 0 0  /* Arc to:  100  25 */
          z;                     /* Close            */
          255 155 155 50;solid;  /* Fill             */
          6.5;darkRed;solid;     /* Border           */
          3 3 8;gray)            /* Shadow effect    */
  AddSymbol(225;65;star5;100;0;teal;shaded;;;0 0 8;green)
  AddSlice(330;20;90;90;90;240;0.0;lightBlue;shaded;5.5;paleVioletRed;shaded;3 3 5)
  AddPath(M  30 195                /* Move to:              30 195 */
          q  45 -60  90   0        /* Quadratic Bézier to: 120 195 */
          c -45  60 -45 -60 -90 0; /* Cubic Bézier to:      30 195 */
          50 0 255 50;solid;       /* Fill                         */
          2.2;darkBlue;solid;      /* Border                       */
          3 3 5)                   /* Shadow effect                */
  AddSmoothPolygon(xyxy;180 160
                        270 160
                        180 230
                        270 230;none;;2 12 4 2.5 4;lightSeaGreen;solid;0 0 1;gray;4)
  AddText(375;195;"rotated texts";;16;;200 0 0 255;center;center;-60)
  AddText(375;195;"rotated texts";;16;;200 0 0 212;center;center;-30)
  AddText(375;195;"rotated texts";;16;;200 0 0 169;center;center;  0)
  AddText(375;195;"rotated texts";;16;;200 0 0 126;center;center; 30)
  AddText(375;195;"rotated texts";;16;;200 0 0  83;center;center; 60)
  AddText(375;195;"rotated texts";;16;;200 0 0  40;center;center; 90)
CloseDrawing()
```



Round and Angled Bar Ends

The new style function `BarStyle()` makes it possible to define different bar end shapes for bar and Gantt charts. Round and angled bar ends are not supported in combination with 3D bars.

BarStyle(seriesIndex;startCapData;endCapData)

Arguments	*Type*	*Range*	*Default*	*Notes*
seriesIndex	int	0..10000	all	*Constants: all (0) (all series)*
startCapData	int[]	0..7	0	*See table: Bar Ends*
endCapData	int[]	0..7	0	*See table: Bar Ends*

The arguments `startCapData` and `endCapData` can have up to four values, i.e. the cap type, see table *Bar Ends*, an optional size value and an also optional flag (0 or 1) to indicate whether the size value is absolute in pixels or as percentage of the bar width (default). The bar caps 4 (peak) and 7 (notch) have an optional 4th argument `alignment` with a value between 0 and 1.
For example:

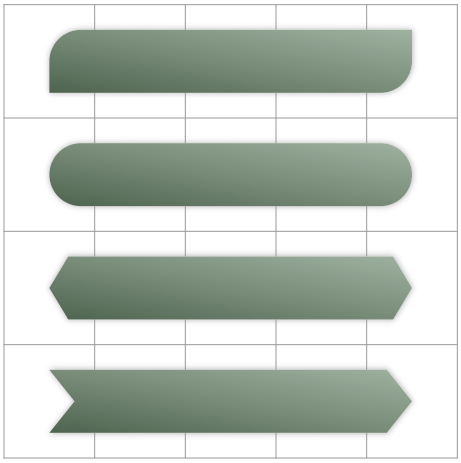
```
BarStyle(1;0;0)           /* Straight bar ends. (default)                        */
BarStyle(1;1;1)           /* Bar with rounded corners. Radius: 25% of bar width. */
BarStyle(1;1 25;1 25)     /* Same as previous example.                           */
BarStyle(1;1 25 1;1 25 1) /* Same as previous example.                           */
BarStyle(1;1 50;1 50)     /* Round bar ends. Radius: 50% of bar width.           */
BarStyle(1;1 15 0;1 15 0) /* Bar with rounded corners. Radius: 15 pixels.        */
BarStyle(1;0;4)           /* Bar with peaked end.                                */
BarStyle(1;0;5)           /* Bar with skewed end.                                */
BarStyle(1;4;7)           /* Bar with a notched and a peaked end.                */
```

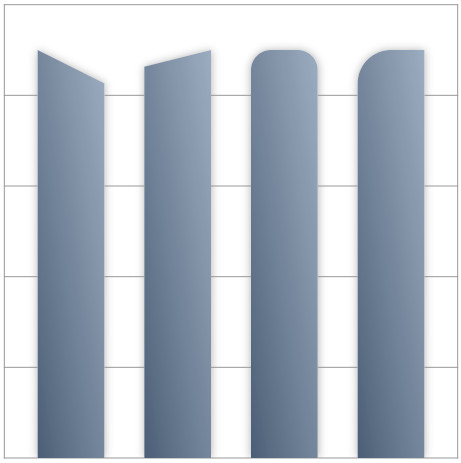
Cap	*Type*	*Example*
	0	`BarStyle(all;0;0)  /* Straight (default) */`
	1	`BarStyle(all;1 50;1 50) /* Rounded  */`
	2	`BarStyle(all;2;2)  /* Rounded right  */`
	3	`BarStyle(all;3 50;3 50) /* Rounded left  */`
	4	`BarStyle(all;0;4 10)  /* Peaked  */`
	5	`BarStyle(all;0;5)  /* Skewed right  */`
	6	`BarStyle(all;0;6 15)  /* Skewed left  */`
	7	`BarStyle(all;0;7 25)  /* Notched  */`

Table: Bar Ends

Examples:

```
OpenDrawing(450;200)
  OpenView(0;0;200;200)
    OpenChart(25;15;170;170;on)
      ChartData(0.1 0.9; 0.1 0.9; 0.1 0.9; 0.1 0.9)
      GanttChart(shadow;80)
      FillStyle(all;90 120 90;shaded)
      BorderStyle(all;none)
      BarStyle(1;3 50;2 50) /* Start cap: rounded left, End cap: rounded right */
      BarStyle(2;1 50;1 50) /* Start cap: rounded,      End cap: rounded       */
      BarStyle(3;4 30;4 30) /* Start cap: peaked,       End cap: peaked        */
      BarStyle(4;7 40;4 40) /* Start cap: notched,      End cap: peaked        */
      ShadowStyle(all;0 0 2)
      ScalingOptions(y;on)
      AxisOptions(all;none) /* hide axes */
      MajorGridLineWidths(all;all;0.25)
    CloseChart()
  CloseView()
  OpenView(250;0;200;200)
    OpenChart(5;15;170;170;on)
      ChartData( 0.9; 0.9; 0.9; 0.9)
      BarChart(shadow;100;60)
      FillStyle(all;86 111 145;shaded)
      BorderStyle(all;0;2.0;150 150 145)
      BarStyle(1;0;5 50) /* Start cap: straight, End cap: skewed right */
      BarStyle(2;0;6 25) /* Start cap: straight, End cap: skewed left  */
      BarStyle(3;0;1 30) /* Start cap: straight, End cap: rounded      */
      BarStyle(4;0;3 50) /* Start cap: straight, End cap: rounded left */
      ShadowStyle(all;0 0 3)
      AxisOptions(all;none) /* hide axes */
      MajorGridLineWidths(all;all;0.25)
    CloseChart()
  CloseView()
CloseDrawing()
```





Alternative Line Smoothing Algorithm

A new line smoothing algorithm called pchip has been added in xmCHART 4. pchip stands for "Piecewise Cubic Hermite Interpolating Polynomial" and has no overshoots and less oscillation if the data are not smooth. On the other hand, the smoothing algorithm used in smooth produces a more accurate result if the data consists of values of a smooth function. In math terminology, smooth (4) returns a spline with curvature continuity, i.e. continuous derivatives of first and second order, pchip (5) has tangent continuity, i.e. only the first-order derivative is continuous.

Constant	*Name*	*Example*
0	none	BorderStyle(all;none) /* Hide border line. */
1	jump	LineStyle(1;jump;0.25;red) /* Thin red line segments. */
2	step	LineStyle(1;step;1 1 1;red) /* Dotted red stair-step line. */
3	poly	LineStyle(1;poly;0.5 6 2;red) /* Dashed red polygonal curve. */
4	smooth	LineStyle(1;smooth;1 6 2 1 2) /* Dot-dashed black smooth spline. */
5	pchip	LineStyle(1;pchip;2.5;red) /* Thick red pchip spline. */

Table: Line Shapes

Optionally, a so-called tension factor can be added to the smooth and pchip constant number. The tension factor in the interval (0,2) can be interpreted as the "length" of the tangent. The default tension is 1, a tension of 0 yields to a polygonal shape. For example:

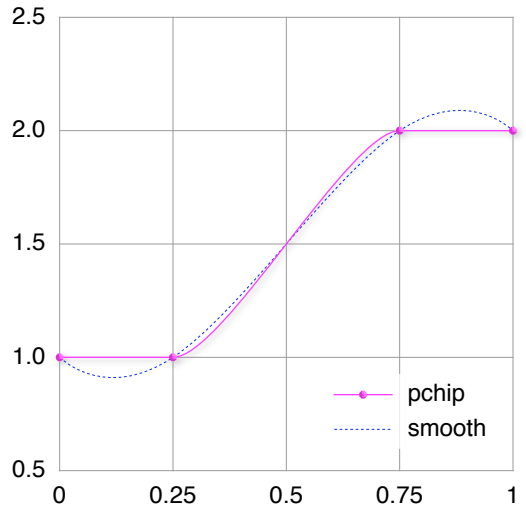
```
LineStyle(1;4 1)     /* Equivalent to LineStyle(1;smooth) */
LineStyle(1;4 0)     /* Equivalent to LineStyle(1;poly)   */
LineStyle(1;4 0.8)
LineStyle(1;5 1)     /* Equivalent to LineStyle(1;pchip)  */
LineStyle(1;5 0)     /* Equivalent to LineStyle(1;poly)   */
LineStyle(1;5 1.2)
```

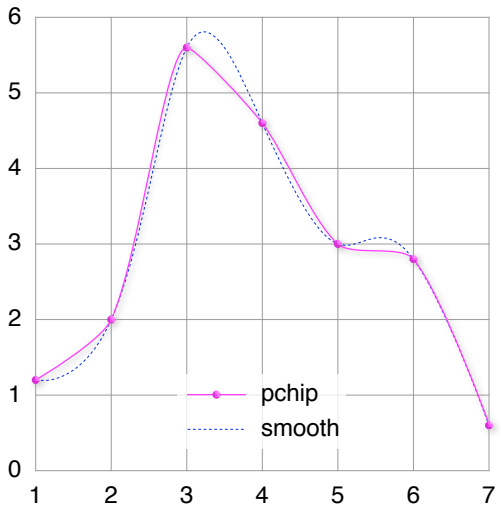
Line smoothing is available for the functions LineStyle(), BorderStyle() and MovingAverage-LineStyle() and the graphics primitives AddSmoothPolygon() and AddSmoothPolyline().

Examples:

```
OpenDrawing(450;200)
  OpenView(10;0;210;200)
    OpenChart(25;15;170;170;on)
      ChartData(0.0 0.25 0.75 1.0 ; 1.0 1.0 2.0 2.0 ;
                0.0 0.25 0.75 1.0 ; 1.0 1.0 2.0 2.0 )
      LineChart2D(symbol+shadow)
      LineStyle(1;5 1.5;0.5;magenta) /* pchip (5) with tension factor 1.5 */
      LineStyle(2;smooth;0.25 1 1;blue)
      SymbolStyle(1;bullet;2;1;magenta;shaded)
```

```
      SymbolStyle(2;bullet;2;1;none)
      ShadowStyle(1;1 1 2)
      ShadowStyle(2;0)
      Scaling(x;linear;0;1;4)
      Scaling(y;linear;0.5;2.5;4) AxisLine(all;0)
      AxisMajorTicks(all;0)
      AxisMajorTickLabelTexts(x;"|u|")
      AxisMajorTickLabelOptions(x;out;0;3)
      AxisMajorTickLabelOptions(y;out;-3;0)
      MajorGridLineWidths(all;all;0.25)
      LegendTexts("pchip";"smooth")
      LegendBackground(255 255 255 200;;0)
      LegendOptions(bottomRight;on)
    CloseChart()
  CloseView()
  OpenView(235;0;200;200)
    OpenChart(25;15;170;170;on)
      ChartData( 1.2 2 5.6 4.6 3 2.8 0.6 ;
                 1.2 2 5.6 4.6 3 2.8 0.6 )
      LineChart(symbol+shadow)
      LineStyle(1;pchip;0.5;magenta)
      LineStyle(2;smooth;0.25 1 1;blue)
      SymbolStyle(1;bullet;2;1;magenta;shaded)
      SymbolStyle(2;bullet;2;1;none)
      ShadowStyle(1;1 1 2)
      ShadowStyle(2;0)
      Scaling(y;linear;0;6;6)
      AxisLine(all;0)
      AxisMajorTicks(all;0)
      AxisMajorTickLabelOptions(x;out;0;3)
      AxisMajorTickLabelOptions(y;out;-3;0)
      MajorGridLineWidths(all;all;0.25)
      LegendTexts("pchip";"smooth")
      LegendBackground(255 255 255 200;;0)
      LegendOptions(bottomCenter;on)
    CloseChart()
  CloseView()
CloseDrawing()
```





Import of Base64 Encoded Images

xmCHART 4 offers a fast new method for importing images from FileMaker. The argument `sourceType`, part of the functions `AddPicture()`, `PictureStyle()`, `BackgroundPict()` and `ChartBackgroundPict()` has been enhanced by the constant `stream`, see table *Input Sources*. The PNG image data must be Base64 encoded and may be optionally prefixed with a data URI header: `""`. For example:

```
BackgroundPict(stream;"iVBORw0KGgoAAAANSUhEUgAAAEAAAABABAMA...")
BackgroundPict(stream;"...")
```

Constant	*Name*	*Notes*
1	clipboard	*Not supported in IWP/CWP, WebDirect and FileMaker Go. Deprecated.*
2	resource	*Set of built-in gradients. Deprecated.*
3	file	*Not recommended in a server environment. Instead, use* `stream`.
4	stream	*Base64 encoded PNG image.*

Table: Input Sources

Examples:

```
OpenDrawing(24;24)
  /* Add high resolution image. */
  /* The original image size is 64x64 pixels. */
  AddPicture(0;0;24;24;stream;"
MzMyZmZlmZmYzMzMAAAD///8QLgCRAAAAB3RSTlP///////8AGksDRgAAAAlwSFlzAAALEgAACxIB0t1+/
AAAABx0RVh0U29mdHdhcmUAQWRvYmUgRmlyZXdvcmtzIENTM5jWRgMAAAFISURBVEiJldRPcoIwFAZwVxygI+7bUQ7QD
pxAWQvhsS4Iuf8RnBpC3t+mzQrhN5Lv48HB4/V2hetnQU4d8I8z/
KzhZIEjgBQIPCCuTgf1DqDSQPoDgF4DFwTgJMGEr0MrwYOAXoKSAJAgZHAfZ5ZjB6GiItbVcjDtp8PRyMGc0tWkzAheI
Ya0XR241+FKcipgUcGcuwXf5P3fMbNF+Vuo+v3LqpqMAxqIvz/
NlVx3EmQHZiagkyA7k1vObTUKyI09yTl4BSxqSuPV6zSAY4wawDEaFdRaCAxQDK+CFMPpIPsJSjFGHaQYjQH2p2F8Bvc
Y6ElQEGM4C8Sh6izwvYHWArHsygRbjMIEq9wjBaHs3gbszZcglN38Ai68aA5KXjQHi9gjAzMvmgPPixagZkULUJKJVsD
CihZg5nvkYOJ75MDfqgwoC3biCUwU+G00jdkYAAAAAElFTkSuQmCC")
CloseDrawing()
```

```
OpenDrawing(220;200)
  ChartData(5 4 3 2 1)
  BarChart(shadow)
  PictureStyle(all;stream;"
iVBORw0KGgoAAAANSUhEUgAAAB4AAAAmCAMAAADkx9tQAAAAA3NCSVQICAjb4U/
gAAABhlBMVEWYzTWXzDWXyzWWyjSWyzSVyjSVyTSUyDSTxzOSxjOTxjOSxTOSxDORxDORwzOQwzKQwjKPwTKOwDKPwDK
OvzKNvzGNvjGMvTGLvDGMvDGKuzCKujCJuTCJuDCItzCIuDCHty+Hti+GtS+DsS6Bri2ArCx9qix9qCt8qCt8pit8pyt
6pSt1nSl0nSh0nChzmyhznChymyhymihymShxmChwmChwlyhvlydskyZrkSZojSRniyRhgiJigyJhgyJfgCFffyFdfSB
aeh9Ydx9UcR1TcR1RbhxRbRxQbBxLZRtMZhpLZRpJYxpIYRlFXRhDWxhCWRc8UhU7UBU6TxQ6ThQ3ShM1RxIyRBIxQhE
vPxAuPhAtPRAsPA8oNg4nNQ0oNQ4mMw0lMg0kMQ0kMA0iLgwgKwseKAodJwocJgocJQoaIwkaIgkZIgkZIQkYIQgYIAg
WHQgWHQcTGgcTGQcTGQYPFQUOEwUNEgUMEAQKDQMHCQIHCgIFBwIFBgIEBQEDBQECAgEAAACUJQm7AAAACXBIWXMAAAs
SAAALEgHS3X78AAAAHHRFWHRTb2Z0d2FyZQBBZG9iZSBGaXJld29ya3MgQ1MzmNZGAwAAAWBJREFUOI2lzFVTglEUheF
tByq22N3d3d3Y3TqiiNiBgeufe5B9OHu4+G58Ltc7s4j+JcISRVqiaKHu8MMzJodoijJqv6EsiiWKYozNQMWnXUwUa1z
/ZdSIieKMvb/qd4hJ5np/IC/
EyRwvNJ7+PEzJIZ4SLVFCSNX4ypoy3WOmBLKxjCNojxV6tFESG4SxocckSmYTwFswvuFMj8mUwubhdwfzJbx6TCE7c+L
pJpj38axHO6WydZxzXoVPj6mUxraxxXkOyNVrKB/AyXkUKA/ldHaBGc69QL1eKZPdYYBzgw/
teqUs9opulX0qN96jX6+UzT5RqXLgoOkYI3qlHOZy5XmAXZVbllxDeg1lRWVnIIuJHIbKkyq3iiksD7+H5TzjFuh7ANr
ERPmGys1XQLuYwnL1SVguMFQu2wE6xESFhhdfhesqi4mKDS9eipeBTjFRkeGFu2gW6BITlViiUku/
wwG2Nx7am24AAAAASUVORK5CYII=";1.0)
  BorderStyle(all;;0.25;0 0 0 50)
  ShadowStyle(all;2 2 3)
  AxisLine(all;0)       /* Hide axis lines. */
  AxisMajorTicks(all;0) /* Hide axis ticks. */
  AxisMajorTickLabelTexts(x;"A";"B";"C";"D";"E")
  MajorGridLineWidths(all;all;0.25)
  GridLocation(all;front)
  ChartBackground(xy;1 30 1
                     0.0  220 220 220 255
                     0.35 240 240 240 255
                     1.0  220 220 220 255;;0)
CloseDrawing()
```

Import and Export of High Resolution Images

Import High Resolution Bitmap Images

xmCHART 4 supports the import of high resolution bitmap images. This proves useful, for example, for printing diagrams with background images. In order to achieve a high printing quality, make sure that the original size of the image is at least 2x larger than the display size of the image. For example, if the display size is 100x100 pixels, the original size of the imported image should be at least 200x200 pixels, see first example in section: Import of Base64 Encoded Images.

Examples:

```
OpenDrawing(400;300;print)
  ...
  BackgroundPict(file;"background.png") /* Image size for example: 1200x900. */
  ...
CloseDrawing()

OpenView(0;0;350;250)
  ...
  BackgroundPict(stream;<Base64 encoded PNG image>) /* Image size for example: 700x500. */
  ...
CloseView()

OpenChart(10;50;400;400;on)
  ...
  ChartBackgroundPict(xy;stream;<Base64 encoded PNG image>) /* Image size e.g.: 800x800. */
  ...
CloseChart()

AddPicture(10;10;100;100;file;"image.png") /* Image size for example: 200x200. */
AddPicture(10;10;;;file;"image.png")       /* Image shown in original size.    */
AddPicture(10;10;100;100;clipboard)        /* Image size for example: 200x200. */

PictureStyle(1;stream;<Base64 encoded PNG image>;25.0) /* For bar charts, the image width
                                                          should be at least 2x larger than
                                                          the width of the bars. */
```

Notes:

• Importing images from a file in a server environment is not recommended. Instead, import images directly from FileMaker by using a Base64 encoded PNG image string. Details can be found in section: Import of Base64 Encoded Images.
• The clipboard is not supported in IWP/CWP, WebDirect and FileMaker Go.

Export High Resolution Bitmap Images

The created drawing can be exported as a high resolution bitmap image to a file, copied to the clipboard or stored in a FileMaker container field. xmCHART 4 supports image resolution factors from 0.25 (very coarse and pixelated) to 4.0 (high resolution).

Output high resolution bitmap image to FileMaker:

The image resolution can be controlled with the 3rd argument `resolution` in the function `OpenDrawing(width;height;resolution)`. If there is no 3rd argument, the drawing will be created as a PNG bitmap with a resolution optimized for screen display. The following `resolution` options are available:

Constant	*Name*	*Notes*
-1	print	PNG bitmap, resolution optimized for printing (300 dpi).
0	screen	PNG bitmap, resolution optimized for screen display.
0.25..4		PNG bitmap, adjustable resolution from coarse (0.25) to fine (4.0).

Table: Output Resolutions

For example:

```
OpenDrawing(800;600)        /* PNG bitmap, optimized for screen display, e.g.: 96 dpi  */
OpenDrawing(800;600;screen) /* Same as OpenDrawing(800;600)                            */
OpenDrawing(800;600;0)      /* Same as OpenDrawing(800;600)                            */
OpenDrawing(800;600;print)  /* PNG bitmap, optimized for printing (300 dpi)            */
OpenDrawing(800;600;-1)     /* Same as OpenDrawing(800;600;print)                      */
```

A high resolution bitmap, for example `print` (300 dpi), is recommended for FileMaker script commands such as *Save/Send Records As PDF...* or *Export Field Contents...*

Export high resolution bitmap image to a file:

To all five bitmap output functions a new argument `resolution` has been added in xmCHART 4. The argument `resolution` can vary from 0.25 (very coarse and pixelated) to 4.0 (high resolution). The 3rd argument `creatorType` available in various `SaveAs...()` functions is ignored in xmCHART 4 and is only for compatibility with previous xmCHART versions.

SaveAsBMPFile(fileName;fileFlag;creatorType;resolution)
SaveAsGIF(fileName;fileFlag;creatorType;resolution)
SaveAsJPGFile(fileName;fileFlag;creatorType;compression;resolution)
SaveAsPNGFile(fileName;fileFlag;creatorType;resolution)
SaveAsTIFFile(fileName;fileFlag;creatorType;resolution)

For example:

```
SaveAsPNGFile("drawing.png";replace;;2.0) /* Export image 2x larger than original size.   */
SaveAsJPGFile("graph.jpg";;;;max;3.5)     /* Export image 3.5x larger than original size. */
SaveAsTIFFile("chart.tif")                /* Export image identical to original size.     */
```

Unicode Support

xmCHART 4 is entirely Unicode-based. Texts may contain any Unicode characters and all text processing is done in Unicode. For more information on Unicode characters, see for example:

http://www.unicode.org/charts/
http://unicode-table.com/en/sections/basic-latin/

Examples:

```
OpenDrawing(200;210)
  AddText(10; 20;"Hello World (English)";Arial;12;plain;40 40 40)
  AddText(10; 40;"مرحبا العالم (Arabic)";Tahoma;12;plain;40 40 40)
  AddText(10; 60;"你好世界 (Chinese)";黑體-繁;12;plain;40 40 40)
  AddText(10; 80;"Γεια σας κόσμο (Greek)";Arial;12;plain;40 40 40)
  AddText(10;100;"안녕하세요 세계 (Hangul)";애플명조;12;plain;40 40 40)
  AddText(10;120;"שלום עולם (Hebrew)";Microsoft Sans Serif;12;plain;40 40 40)
  AddText(10;140;"नमस्ते दुनिया (Hindi)";;12;plain;40 40 40)
  AddText(10;160;"こんにちは世界 (Japanese)";ヒラギノ角ゴ Pro W3;12;plain;40 40 40)
  AddText(10;180;"Привет мир (Russian)";Verdana;12;plain;40 40 40)
  AddText(10;200;"สวัสดีชาวโลก (Thai)";Tahoma;12;plain;40 40 40)
CloseDrawing()
```

Hello World (English)
مرحبا العالم (Arabic)
你好世界 (Chinese)
Γεια σας κόσμο (Greek)
안녕하세요 세계 (Hangul)
שלום עולם (Hebrew)
नमस्ते दुनिया (Hindi)
こんにちは世界 (Japanese)
Привет мир (Russian)
สวัสดีชาวโลก (Thai)

Unicode escape sequences

A Unicode escape sequence represents the single Unicode character formed by the hexadecimal number following the \u or \U character. The lowercase \u escape sequence is followed by 4 hex digits, the uppercase \U escape sequence is similar, but expects 8 hex digits, not 4.

Examples:

```
OpenDrawing(450;50)
  AddText(15;40;
  "⇆ \u270D \u27F2 \u266C \u2030 \u2718 \u2714 \u2109 \u267B"
  ;Arial;40;plain;86 111 145)
CloseDrawing()
```

```
OpenDrawing(450;80)
  AddText(10;60;
 "罘\U0002629B \U0001F50E \U0001F512 \U0001F4CE \U0001F4BB \U0001F464 \U0001F4AC \U0001F557"
  ;Arial;40;bold;86 111 145)
CloseDrawing()
```

Output in macOS

Output in Windows

Note:

xmCHART also supports the following "standard" escape sequences:

```
\n (Newline)
\r (Carriage return)
\t (Horizontal tab)
\\ (Backslash)
\" (Double quote)
```

Miscellaneous

- The attributes colorVariant and opacity have been added to all color scheme functions:

```
BorderColorScheme(schemeIndex;colorVariant;opacity)
FillColorScheme(schemeIndex;colorVariant;opacity)
LineColorScheme(schemeIndex;colorVariant;opacity)
SymbolColorScheme(schemeIndex;colorVariant;opacity)
```

  Please note, for all color scheme functions the attribute colorVariant is limited to solid (0) and shaded (-1). For example:

```
FillColorScheme(classic;shaded) // Okay.
FillColorScheme(1;solid)        // Okay.
FillColorScheme(1;38)           // Error.
```

- New color constant none added. For example:

```
// All 3 functions are equivalent.
FillStyle(2;none)
FillStyle(2;0 0 0 0)
FillStyle(2;;transparent)
```

- In xmCHART 4 invisible data series are removed from the legend. This proves useful for more advanced diagrams.

- xmCHART 4 supports non-integer font sizes, for example 9.5 or 16.25. Furthermore, the limit for the maximum font size has been increased from 127 to 1000 points.

- The category and series gap limits for histograms, bar and Gantt charts have been increased from 1000 to 10000 (in % of the bar width).

- The default value for the argument fileFlag, part of all SaveAs...() functions has been changed from addCounter in xmCHART 3 to replace in xmCHART 4. For example:

```
SaveAsPNGFile("g1.png") // In xmCHART 4 same as: SaveAsPNGFile("g1.png";replace)
SaveAsPNGFile("g1.png") // In xmCHART 3 same as: SaveAsPNGFile("g1.png";addCounter)
```

- The default JPEG image quality in SaveAsJPGFile() has been increased from normal to high. JPEG image quality constants: 1...min, 2...low, 3...normal, 4...high, 5...max

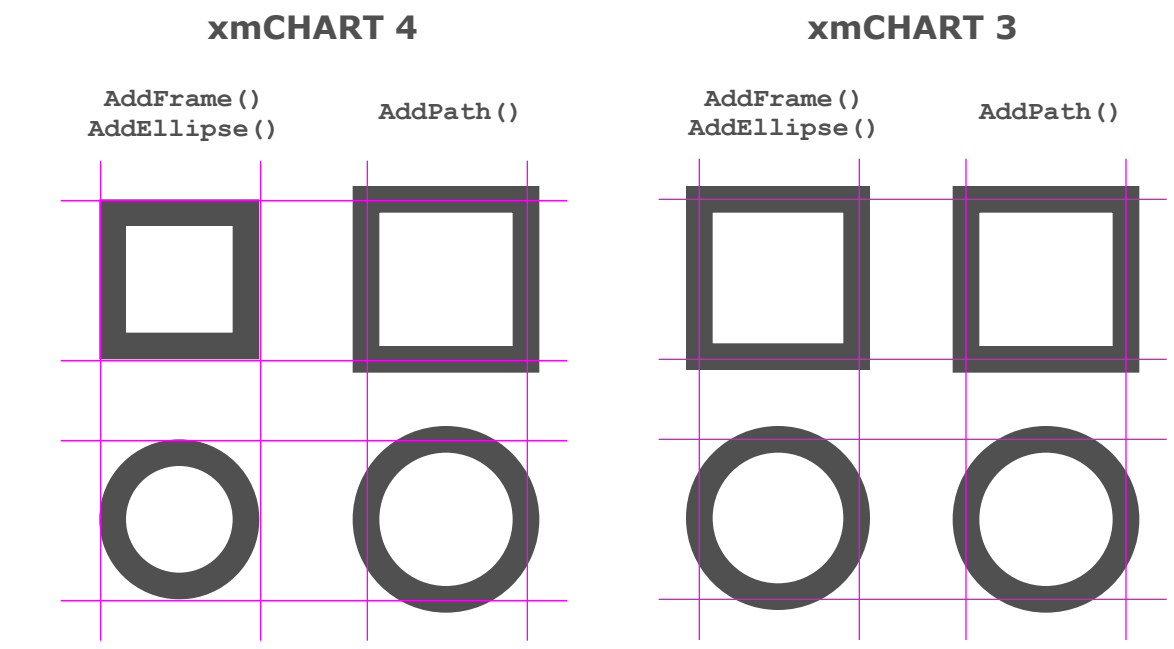
- New default shadow settings in xmCHART 4:

```
shadowEffect: 3.0 3.0 3.0     // offsetX offsetY blur
shadowColor:  136 136 136 170 // red green blue alpha (gray translucent)
```

- By default the error alert sound is turned off in xmCHART 4. It can be activated by the 4th argument errorFlags of the external function xmCH_DrawChart(). Error flags are discussed in detail in combination with the external function xmCH_GetErrorMessage().

- The border of the graphics primitives AddRect(), AddFrame(), AddOval(), AddEllipse(), AddRoundRect(), AddRoundFrame(), AddSlice() and AddArc() is drawn inside of the defined rectangle and not centered along the rectangle bounds as in xmCHART 3. For example:

```
OpenDrawing(225;250)

  AddText(110;20;"xmCHART 4";Verdana;12;1;80 80 80;center)
  AddText( 62;45;"AddFrame()\nAddEllipse()";Courier New;10;bold;80 80 80;center)
  AddText(162;50;"AddPath()";Courier New;10;bold;80 80 80;center)

  OpenView(10;60;225;200)
    /* Left square and oval */
    AddFrame(20;20;60;60;10.0;80 80 80)
    AddEllipse(20;110;60;60;10.0;80 80 80)

    /* Right square and oval created as path */
    AddPath(M 120 20 l 60 0 l 0 60 l -60 0 z;;;10.0;80 80 80)
    AddPath(M 120 140 a 60 0 30 30 0 0 0 a -60 0 30 30 0 0 0 z;;;10.0;80 80 80)

    /* Vertical lines */
    AddLine( 20;5; 20;185;0.5;magenta)
    AddLine( 80;5; 80;185;0.5;magenta)
    AddLine(120;5;120;185;0.5;magenta)
    AddLine(180;5;180;185;0.5;magenta)

    /* Horizontal lines */
    AddLine(5; 20;195; 20;0.5;magenta)
    AddLine(5; 80;195; 80;0.5;magenta)
    AddLine(5;110;195;110;0.5;magenta)
    AddLine(5;170;195;170;0.5;magenta)
  CloseView()

CloseDrawing()
```



Incompatibilities

File paths

In xmCHART 4 the file path syntax becomes more compliant with Unix standards:

https://en.wikipedia.org/wiki/Path_(computing)

A leading slash means in xmCHART 4 an absolute path (full path) — in xmCHART 3 a file path with a leading slash defines a path relative to the current database's location.

Absolute file paths in macOS:

/Volumes/volume/folder/fileName

Examples:

```
SaveAsPNGFile("MacintoshHD/users/john/desktop/graph123.png")          // Error (xmCHART 3).
SaveAsPNGFile("/users/john/desktop/graph123.png")                     // Okay.
SaveAsPNGFile("/Volumes/MacintoshHD/users/john/desktop/graph123.png") // Okay.
SaveAsPNGFile("/Volumes/BackupHD/graphs/graph123.png")                // Okay.
```

Absolute file paths in Windows OS:

/drive:/folder/fileName

A slash in front of the drive letter is optional and can be omitted.

Examples:

```
SaveAsPNGFile("/C:/users/john/desktop/graph123.png") // Okay.
SaveAsPNGFile("C:/users/john/desktop/graph123.png")  // Okay.
SaveAsPNGFile("/Z:/graphs/graph123.png")             // Okay.
SaveAsPNGFile("Z:/graphs/graph123.png")              // Okay.
```

Relative file paths in macOS and Windows OS:

The file path is relative to the current database's location.

Examples:

```
SaveAsPNGFile("graph123.png")                                    // Okay.
SaveAsPNGFile("sub_directory/graph123.png")                      // Okay.
SaveAsPNGFile("/sub_directory/graph123.png")                     // Error (xmCHART 3).
SaveAsPNGFile("sub_directory/sub_sub_directory/graph123.png")  // Okay.
SaveAsPNGFile("/sub_directory/sub_sub_directory/graph123.png") // Error (xmCHART 3).
```

New file path options in xmCHART 4 (macOS and Windows OS):

The file path is relative to the current database's location.

Examples:

```
SaveAsPNGFile("./graph123.png")                  // Path relative to current directory.
SaveAsPNGFile("../graph123.png")                 // Path relative to parent directory.
SaveAsPNGFile("../../graph123.png")              // Path relative to grandparent directory.
SaveAsPNGFile("../../sub_directory/graph123.png") // Path relative to grandparent directory.
SaveAsPNGFile("~/graphs/graph123.png")           // Path relative to user's home directory,
                                                 // same as, for example:
                                                 // Mac: /users/john/graphs/graph123.png
                                                 // Win: /C:/users/john/graphs/graph123.png
```

Deprecations

- OpenDrawing(width;height;format;antialiasing;targetName)
  The arguments antialiasing and targetName are ignored in xmCHART 4.
  targetName has been replaced by the 2nd parameter fileName of the external function xmCH_DrawChart().

- The last 3 arguments location, adjustment and isProportional are ignored:
  BackgroundPict(sourceType;sourceName;location;adjustment;isProportional)
  ChartBackgroundPict(planeIndex;sourceType;sourceName;location;adjustment;isProp.)
  AddPicture(left;top;width;height;sourceType;sourceName;location;adjustment;isProp.)

- The obsolete macOS PICT format has been removed in xmCHART 4. As a consequence, the output Function SaveAsPICTFile() is ignored.

- The 3rd argument creatorType is ignored in all output functions:
  SaveAsBMPFile(fileName;fileFlag;creatorType;resolution)
  SaveAsGIFFile(fileName;fileFlag;creatorType;resolution)
  SaveAsJPGFile(fileName;fileFlag;creatorType;compression;resolution)
  SaveAsPDFFile(fileName;fileFlag;creatorType;title;subject;author)
  SaveAsPNGFile(fileName;fileFlag;creatorType;resolution)
  SaveAsSVGFile(fileName;fileFlag;creatorType;title;description;comment)
  SaveAsTIFFile(fileName;fileFlag;creatorType;resolution)

- SaveAsPDFFile(fileName;fileFlag;creatorType;title;subject;author;creator;keywords)
The last 2 arguments creator and keywords are ignored.

- The fileFlag constant throwError has been deprecated in all output functions:

- The argument shadowPattern has been removed in function ShadowStyle() and in all background functions:
  AxisLabelBackground()
  AxisMajorTickLabelBackground()
  AxisMinorTickLabelBackground()
  Background()
  ChartBackground()
  LabelBackground()
  LegendBackground()
  PieChartCenterLabelBackground()
  PieChartInnerLabelBackground()
  TitleBackground()

- The schemeIndex constant none has been deprecated in all color scheme functions:
  BorderColorScheme(schemeIndex;colorVariant)
  FillColorScheme(schemeIndex;colorVariant)
  LineColorScheme(schemeIndex;colorVariant)
  SymbolColorScheme(schemeIndex;colorVariant)
  Instead, use the new constant classic or simply set the argument schemeIndex to 0 for future projects. For example:
  ```
  FillColorScheme(0)       // Okay.
  FillColorScheme(classic) // Okay.
  FillColorScheme(none)    // Not recommended, constant "none" is deprecated.
  ```

- The clipboard (`sourceType` constant `clipboard`) and the export function `SendToClipboard(resolution)` have been deprecated.

- The built-in gradients (`sourceType` constant `resource`) have been deprecated. Instead, you should use the much more flexible color gradients.