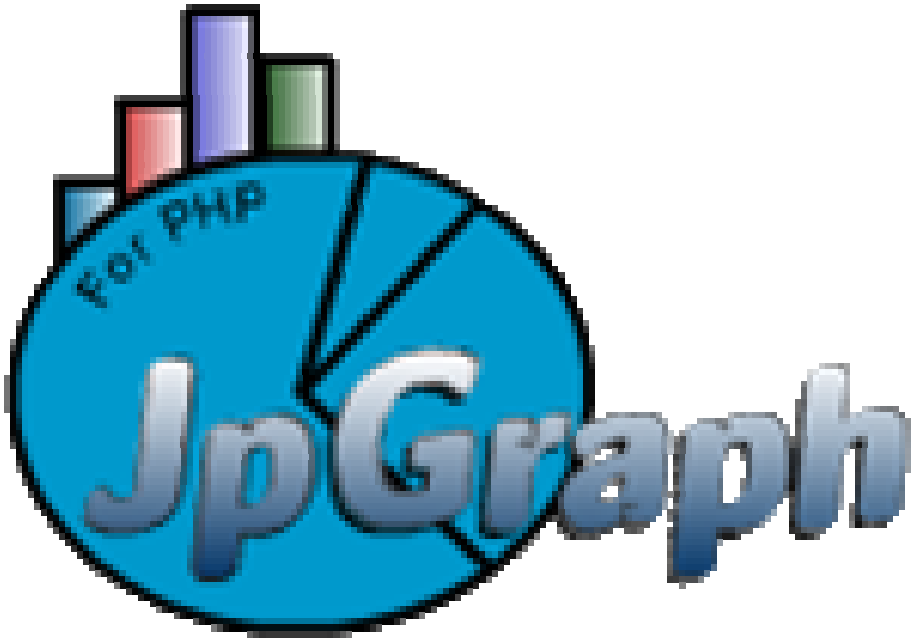


JpGraph – eine Einführung

Vortrag vom 03.06.2004 auf der PHP-Usergroup Hannover

von Frank Staude <staude@trilos.de>

Inhalt

1. Einleitung

David hat es in seiner Einführung in der GD-Lib am 2.10.2003 bereits geschrieben.
"*Ich sehe keinen Sinn darin, Linien und Flächen in Bildern zu generieren, also lasse ich dieses Anwendungsgebiet außen vor. Wer Diagramme dynamisch generieren möchte, kommt an JpGraph (www.aditus.nu/jpgraph) nicht vorbei, die einem diese ganze Arbeit gut abnimmt.*"

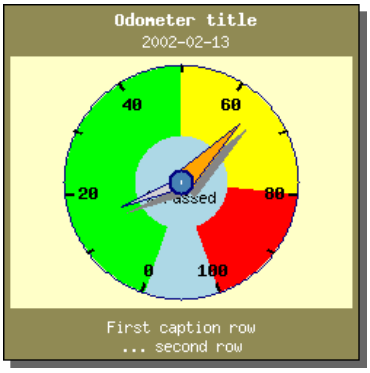
2. Features

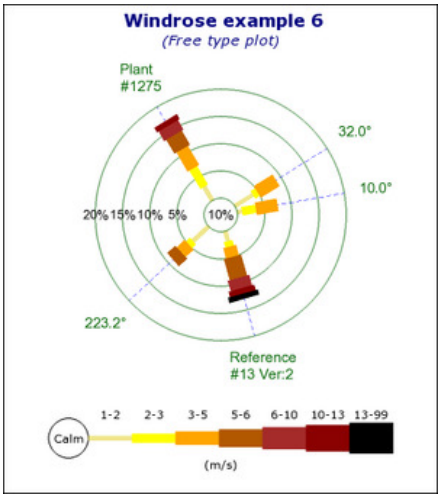
(eine Auswahl)

- Erzeugt kleine Grafiken. Die durchschnittliche Größe einer Grafik 300x200 Pixel ist um die 2 Kb.
- JpGraph arbeitet sowohl mit der GD als auch mit der GD2 Bibliothek zusammen und kann selbstständig erkennen welche vorhanden ist.
- Unterstützt Gantt Diagramme (für z.B: Projektplanungssoftware)
- Unterstützung von TTF-Fonts zur Diagrammbeschriftung.
- Kann Gif, PNG und JPG erzeugen.
- Über 400 benannte Farben
- Hintergrundbilder in Graphen sind möglich.
- Serverentlastung durch cachingmechanismen.
- Barcodeerzeugung (in der Kommerziellen Version)
- Sehr gute Dokumentation (150 Seiten mit vielen Beispielen und Klassenreferenz)
- Unterstützung von Alpha blending
- Über 200 Länderflaggen eingebaut
- Diverse Graphentypen (siehe "Diagrammübersicht")

3. Lizenz der JpGraph

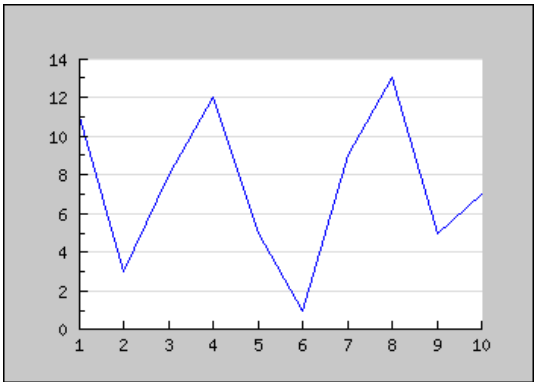
Die JpGraph Bibliothek ist in einer Duallizenz erhältlich. Für nicht-kommerzielle Anwendungen unterliegt sie der QLP (Qt-Lizenz) und die Professional Version für kommerzielle Anwendungen (inkl. Barcodegenerierung, Windrosen und Odometer). Eine Single-Server Lizenz kostet 85 EUR + MwSt. Weitere Informationen zur Lizenz finden sich unter http://www.aditus.nu/jpgraph/jpg_proversion.php

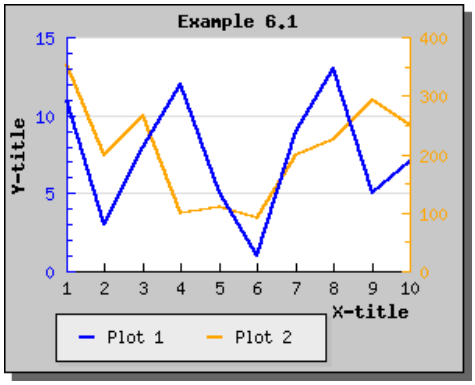


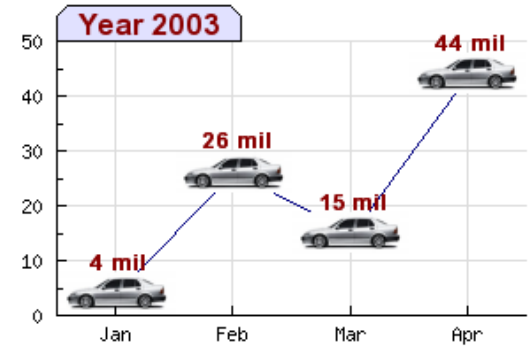


4. Diagrammübersicht

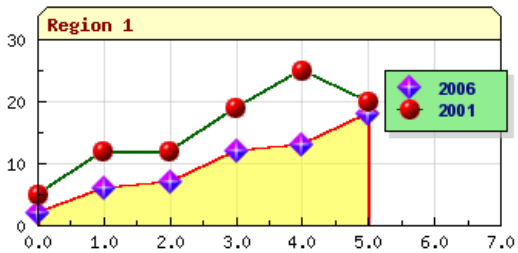
Ein Auswahl möglicher Diagramme/Graphen die mit JpGraph möglich sind.
Alle hier abgebildeten Diagramme sind aus der Dokumentation und sind dort inkl. Sourcecode beschrieben.

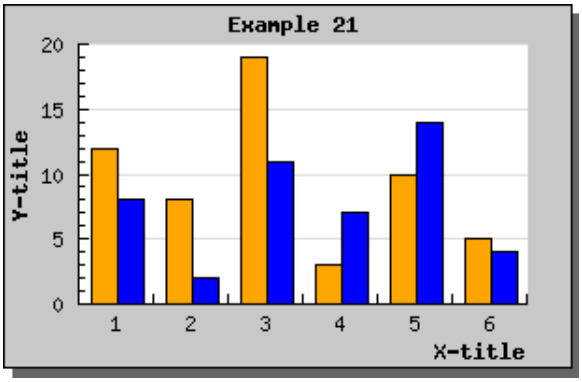


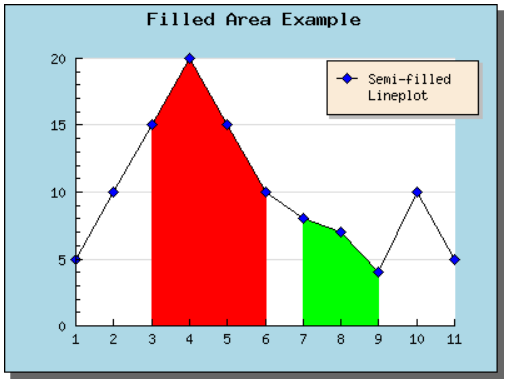


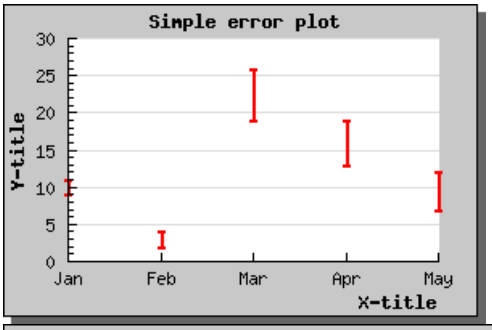


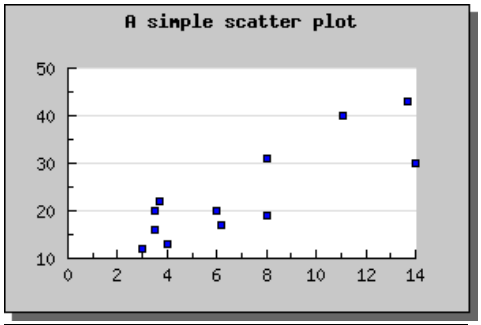
Using Builtin PlotMarks

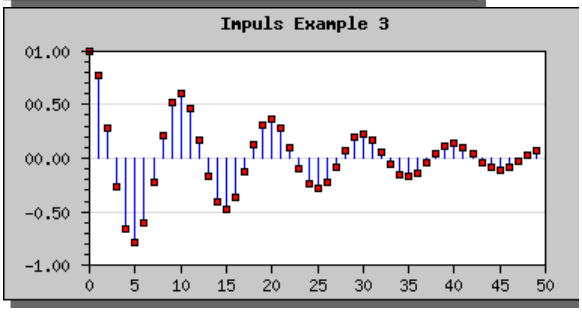


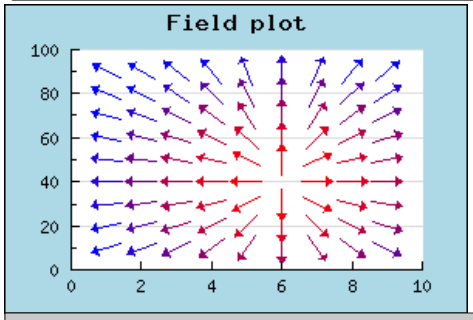


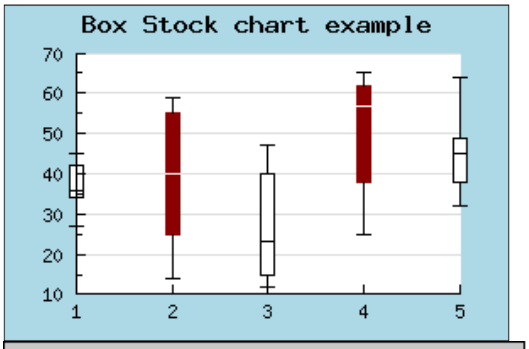


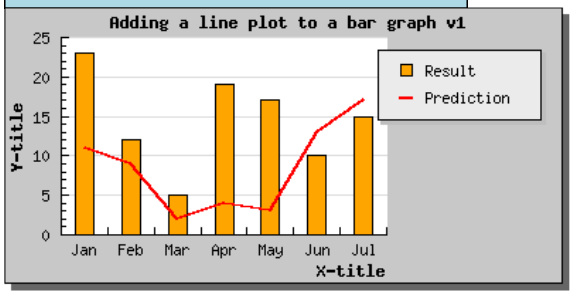

Simple error plot
Y-title
X-title
Jan
Feb
Mar
Apr
May

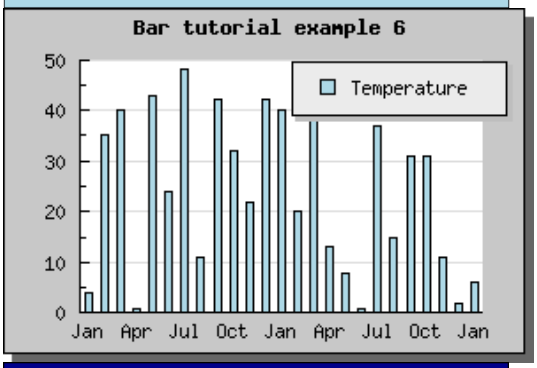

A simple scatter plot

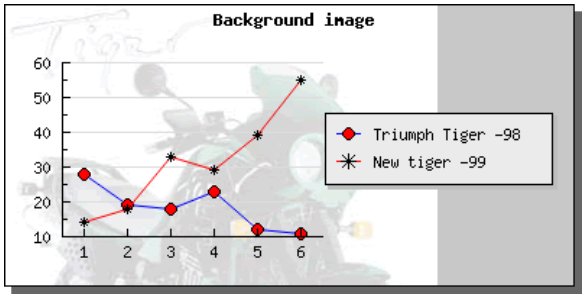

Impuls Example 3

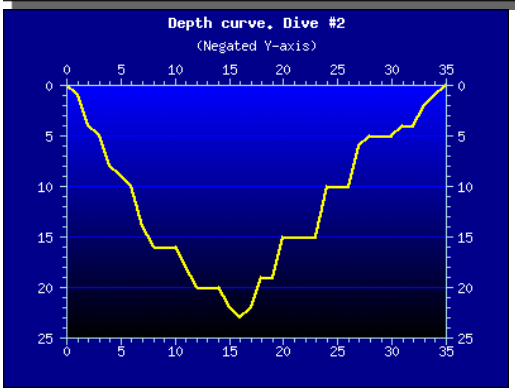

Field plot

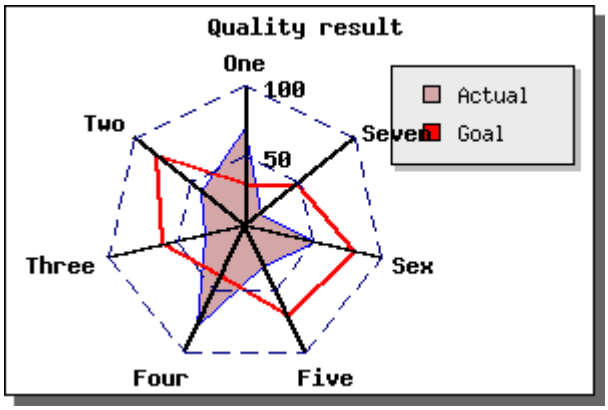

Box Stock chart example

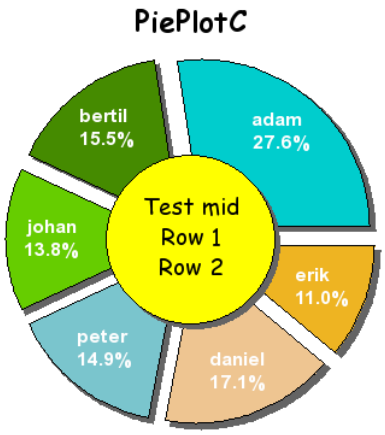

Adding a line plot to a bar graph v1
Result
Prediction
Y-title
X-title
Jan
Feb
Mar
Apr
May
Jun
Jul

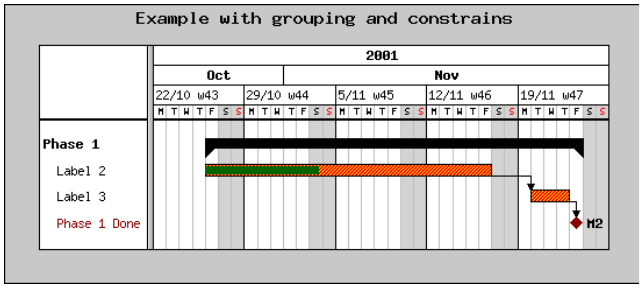

Bar tutorial example 6
Temperature
Jan
Apr
Jul
Oct
Jan
Apr
Jul
Oct
Jan

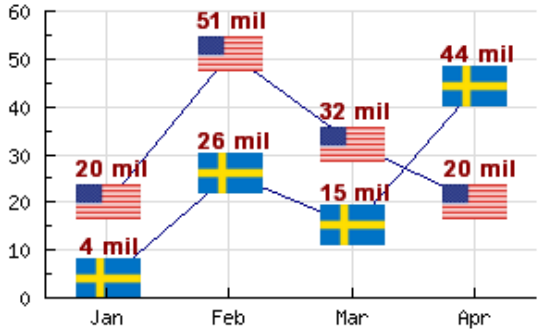

Background image
Triumph Tiger -98
New tiger -99

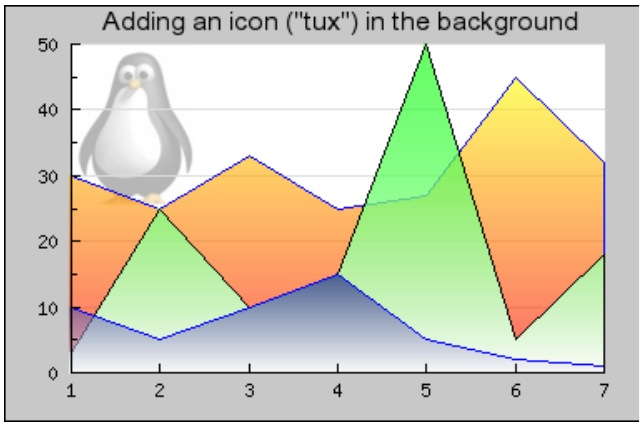

Depth curve, Dive #2
(Negated Y-axis)


Quality result
One
Two
Three
Four
Five
Sex
Seven
100
50
Actual
Goal



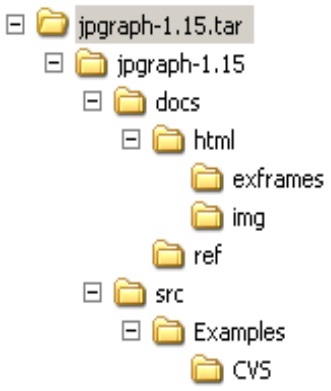






5. Installation

Nach dem entpacken des Archivs hat man zwei Verzeichnisse vorliegen. Docs und src. In Docs befindet sich die Dokumentation und in src der PHP-Code der JpGraph. Den Inhalt des src Verzeichnisses in ein Verzeichnis auf dem Server kopieren aus dem man es included oder welches man in den includepfad aufnimmt.



Im Detail werden folgenden Dateien benötigt.

Jpgraph.php – ist die Hauptdatei die immer included wird.

Und den Plot-Erweiterungen

jpgraph_log.php
jpgraph_line.php
jpgraph_bar.php
jpgraph_error.php,
jpgraph_scatter.php
jpgraph_spider.php
jpgraph_pie.php
jpgraph_pie3d.php
jpgraph_gantt.php
jpgraph_radar.php

jpgraph_polar.php
jpgraph_regstat.php
jpgraph_stock.php
jpgraph_gradient.php
jpgraph_gb2312.php
jpgraph_plotmark.php
jpgraph_flags.php
imgdata_*.inc
flags*.dat
jpgraph_canvas.php
jpgraph_canvtools.php

6. Tortendiagramm

Los geht's…Als erstes bauen wir ein einfaches Tortendiagramm.

Dazu benötigen wir:

Die JpGraph-Lib

include ("jpgraph/jpgraph.php");

Außerdem das Plotmodul für Tortendiagramme.

include ("jpgraph/jpgraph_pie.php");

Nun brauchen wir noch ein paar Werte

```
$data = array(40,60,21,33);
```

Als nächstes bauen wir die Grafik. Dazu erzeugen wir ein neues Grafikobjekt in der Größe 300 x 200 Pixel.

$graph = new PieGraph(300,200,"auto");

Nun noch dem ganzen einen Titel geben

$graph->title->Set("Example 1 Pie plot");
$graph->title->SetFont(FF_VERDANA,FS_BOLD,14);
$graph->title->SetColor("black");

Die Werte kommen nun in ein Tortenobject

```
$p1 = new PiePlot($data);
```

Den Graphen den das Tortenobject liefert tragen wir in die Grafik ein.

```
$graph->Add($p1);
```

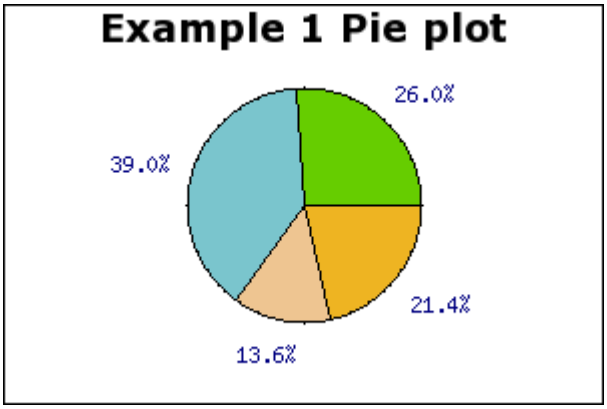
Und nun noch ausgeben.

$graph->Stroke();

Das gesamte PHP-Sktipt sieht nun so aus:

```
<?
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_pie.php");

$data = array(40,60,21,33);

$graph = new PieGraph(300,200,"auto");

$graph->title->Set("Example 1 Pie plot");
$graph->title->SetFont(FF_VERDANA,FS_BOLD,14);
$graph->title->SetColor("black");

$p1 = new PiePlot($data);

$graph->Add($p1);
$graph->Stroke();
?>
```

Und liefert als Ergebnis folgende Grafik:



Das sieht nun noch ein bisschen langweilig aus.

Als erstes erzeugen wir ein array mit den Beschriftungen:

```
$legend = array("Perl","PHP","C","Tcl");
```

Anschließend wird die Legende dem Graphen hinzugefügt.

```
$p1->SetLegends($legend);
```

Der Graph muss nun noch ein bisschen nach links, damit rechts Platz für die Beschriftung ist.

```
$p1->SetCenter(0.4);
```

Und nun noch ein bisschen Schatten, damit es hübscher aussieht.

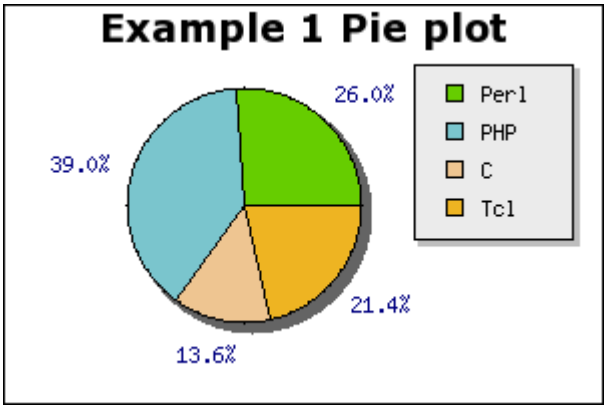
```
$p1->SetShadow();
```

Damit sieht unser Programm nun so aus

```
<?
```

```
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_pie.php");

$data = array(40,60,21,33);
$legend = array("Perl","PHP","C","Tcl");

$graph = new PieGraph(300,200,"auto");

$graph->title->Set("Example 1 Pie plot");
$graph->title->SetFont(FF_VERDANA,FS_BOLD,14);
$graph->title->SetColor("black");

$p1 = new PiePlot($data);
$p1->SetLegends($legend);
$p1->SetCenter(0.4);
$p1->SetShadow();
$graph->Add($p1);
$graph->Stroke();
?>
```

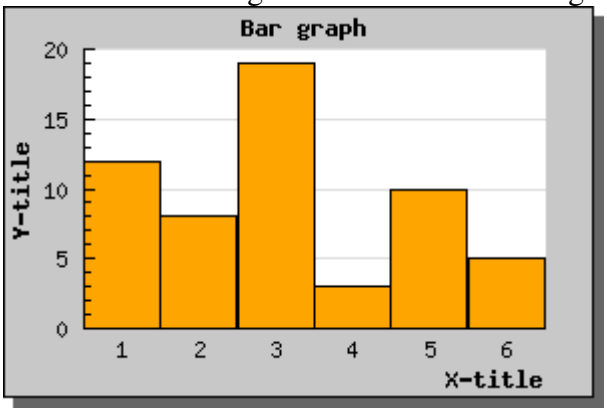
Und erzeugt diese Grafik.



Ok, machen wir Schluss mit den Torten und bauen lieber "Schoko-Riegel".

7. Balkendiagramm

Als nächstes Erzeugen wir dieses Balkendiagramm.



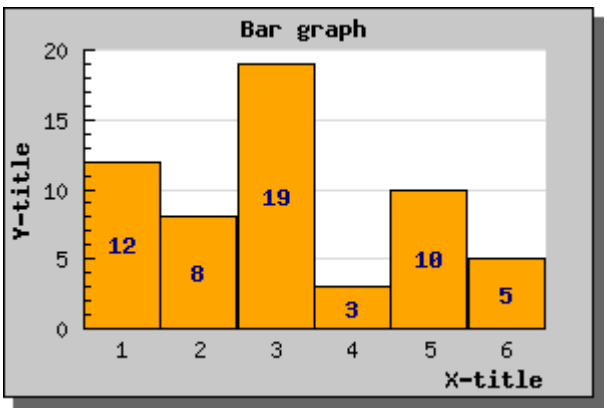
Der Programmcode dazu sieht folgendermaßen aus.

```
<?php
include ("jpgraph/jpgraph.php");
```

```
include ("jpgraph/jpgraph_bar.php");

$datay=array(12,8,19,3,10,5);

$graph = new Graph(300,200,"auto");
$graph->SetScale("textlin");

$graph->SetShadow();

$graph->img->SetMargin(40,30,20,40);

$bplot = new BarPlot($datay);
$bplot->SetFillColor('orange');
$bplot->SetWidth(1.0);
$graph->Add($bplot);

$graph->title->Set("Bar graph");
$graph->xaxis->title->Set("X-title");
$graph->yaxis->title->Set("Y-title");

$graph->title->SetFont(FF_FONT1,FS_BOLD);
$graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);

$graph->Stroke();
?>
```

Die absoluten Zahlen der Balken als Beschriftung wären nett? Dann fügen wir sie hinzu. Die folgenden Zeilen ergänzen die fehlende Beschriftung:

```
$bplot->value->Show();
$bplot->value->SetFormat('%d');
$bplot->value->SetFont(FF_FONT1,FS_BOLD);
$bplot->SetValuePos('center');
$graph->Add($bplot);
```



Der gesamte Code sieht nun so aus:

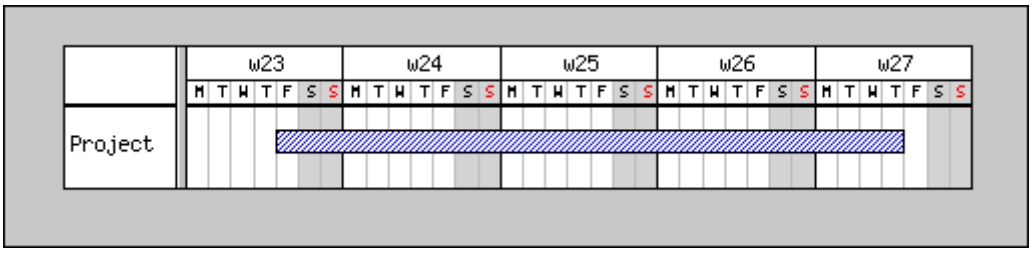
```
<?php
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_bar.php");

$datay=array(12,8,19,3,10,5);
```

```
$graph = new Graph(300,200,"auto");
$graph->SetScale("textlin");

$graph->SetShadow();

$graph->img->SetMargin(40,30,20,40);

$bplot = new BarPlot($datay);
$bplot->SetFillColor('orange');
$bplot->SetWidth(1.0);

$bplot->value->Show();
$bplot->value->SetFormat('%d');
$bplot->value->SetFont(FF_FONT1,FS_BOLD);
$bplot->SetValuePos('center');
$graph->Add($bplot);

$graph->title->Set("Bar graph");
$graph->xaxis->title->Set("X-title");
$graph->yaxis->title->Set("Y-title");

$graph->title->SetFont(FF_FONT1,FS_BOLD);
$graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);

$graph->Stroke();
?>
```

8. Gantt-Diagramm

Damit jeder von Euch dann seine eigenen Vorträge für die PHPUG planen kann, erzeugen wir jetzt noch ein Gantt-Diagramm. Das Projekt beginnt morgen und endet pünktlich am 1.7. zum nächsten Treffen.

```
<?php
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_gantt.php");

$graph = new GanttGraph (0,0, "auto");

$activity = new GanttBar (0,"Project", "2004-06-04", "2004-07-01");
$graph->Add( $activity);

$graph->Stroke();
?>
```



Schon nicht schlecht - hat auch schon eine gewisse Ähnlichkeit mit MS Project, aber noch fehlen ein paar Dinge. Wie viel ist schon erledigt? Wann sind Meilensteine? Und nicht jeder rechnet in Kalenderwochen.

Als erstes fügen wir de Monate hinzu.

```
$graph->ShowHeaders( GANTT_HDAY | GANTT_HWEEK | GANTT_HMONTH);
```

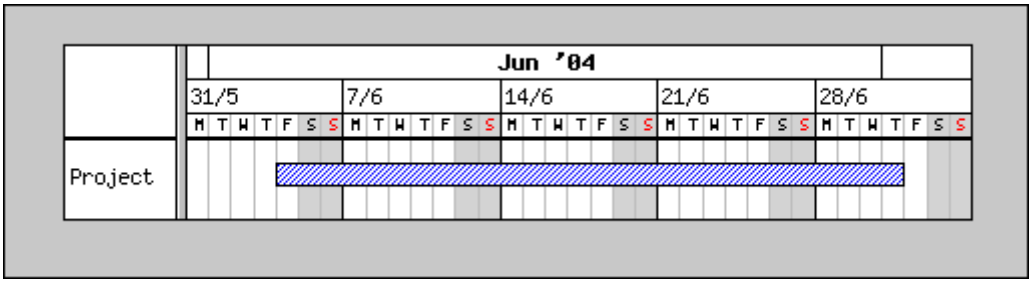
Dann statt Kalenderwoche das Datum des ersten Tages der Woche.

```
$graph->scale-> week->SetStyle(WEEKSTYLE_FIRSTDAY);
```

Nun ändern wir die Monatsnamen in die Kurzschreibweise, so dass auch das Jahr mit angezeigt wird.

```
$graph->scale-> month-> SetStyle( MONTHSTYLE_SHORTNAMEYEAR2);
```

Nun sieht es so aus:



Fehlt noch der Meilenstein.

```
$milestone = new MileStone(1,"Vortrag gehalten","2004-07-01","Vortrag");
$graph->Add( $milestone);
```

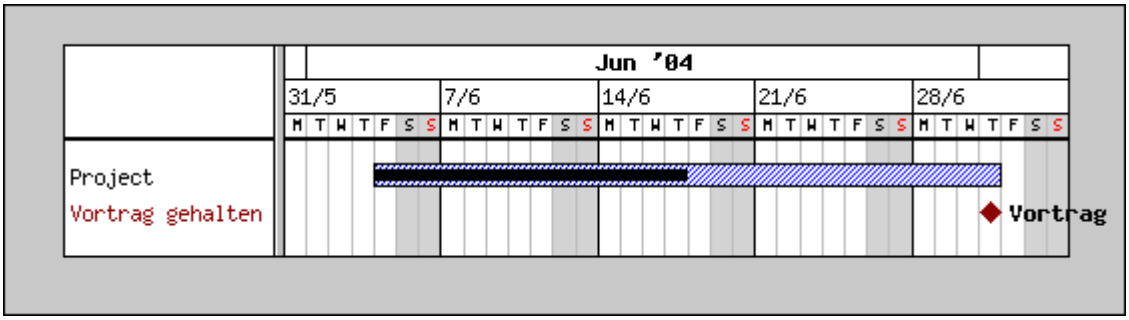
Und die Anzeige, wie viel des Projekts schon erledigt wurde (z.B: 50%)

```
$activity->progress->Set(0.5);
```

Dann ist der vollständige Code jetzt:

```
<?php
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_gantt.php");

$graph = new GanttGraph (0,0, "auto");

$graph->ShowHeaders( GANTT_HDAY | GANTT_HWEEK | GANTT_HMONTH);
$graph->scale-> month-> SetStyle( MONTHSTYLE_SHORTNAMEYEAR2);
$graph->scale-> week->SetStyle(WEEKSTYLE_FIRSTDAY);

$activity = new GanttBar (0,"Project", "2004-06-04", "2004-07-01");
$activity->progress->Set(0.5) ;
$graph->Add( $activity);

$milestone = new MileStone(1,"Vortrag gehalten","2004-07-01","Vortrag");
$graph->Add( $milestone);

$graph->Stroke();
?>
```

Und erzeugt diese Grafik.



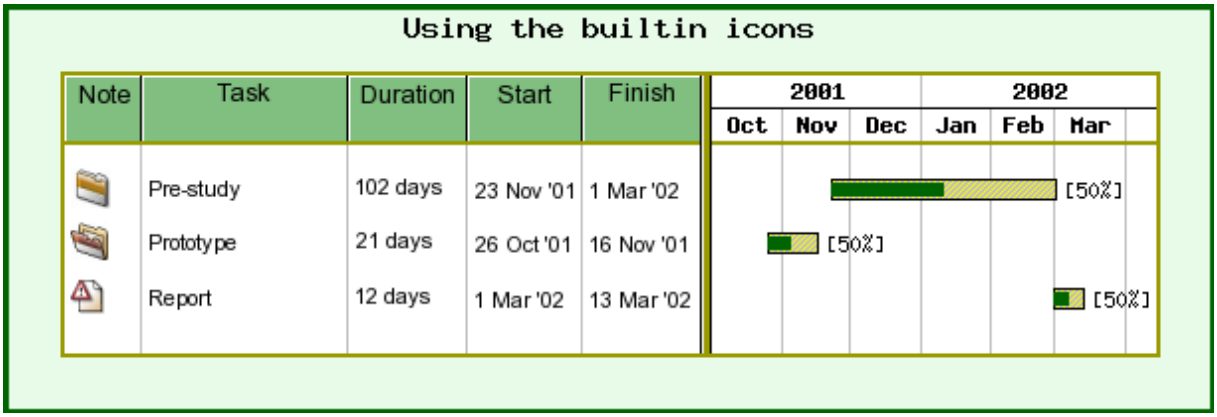
Hier noch mal ein weiteres Beispiel aus der Dokumentation was man mit Gantt-Diagrammen und den eingebauten Icons machen kann.

Dieser Code:

```
<?php
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_gantt.php");

// Basic Gantt graph
$graph = new GanttGraph();
$graph->title->Set("Using the builtin icons");

// Explicitely set the date range
// (Autoscaling will of course also work)
$graph->SetDateRange('2001-10-06','2002-4-10');


// 1.5 line spacing to make more room
$graph->SetVMarginFactor(1.5);

// Setup some nonstandard colors
$graph->SetMarginColor('lightgreen@0.8');
$graph->SetBox(true,'yellow:0.6',2);
$graph->SetFrame(true,'darkgreen',4);
$graph->scale->divider->SetColor('yellow:0.6');
$graph->scale->dividerh->SetColor('yellow:0.6');

// Display month and year scale with the gridlines
$graph->ShowHeaders(GANTT_HMONTH | GANTT_HYEAR);
$graph->scale->month->grid->SetColor('gray');
$graph->scale->month->grid->Show(true);
$graph->scale->year->grid->SetColor('gray');
$graph->scale->year->grid->Show(true);

// For the titles we also add a minimum width of 100 pixels for the Task
name column
$graph->scale->actinfo->SetColTitles(
    array('Note','Task','Duration','Start','Finish'),array(30,100));
$graph->scale->actinfo->SetBackgroundColor('green:0.5@0.5');
$graph->scale->actinfo->SetFont(FF_ARIAL,FS_NORMAL,10);
$graph->scale->actinfo->vgrid->SetStyle('solid');
$graph->scale->actinfo->vgrid->SetColor('gray');

// Uncomment this to keep the columns but show no headers
//$graph->scale->actinfo->Show(false);

// Setup the icons we want to use
$erricon = new IconImage(GICON_FOLDER,0.6);
```

```
$startconicon = new IconImage(GICON_FOLDEROPEN,0.6);
$endconicon = new IconImage(GICON_TEXTIMPORTANT,0.5);

// Store the icons in the first column and use plain text in the others
$data = array(
    array(0,array($erricon,"Pre-study","102 days","23 Nov '01","1 Mar '02")
          , "2001-11-23","2002-03-1",FF_ARIAL,FS_NORMAL,8),
    array(1,array($startconicon,"Prototype","21 days","26 Oct '01","16 Nov
'01"),
          "2001-10-26","2001-11-16",FF_ARIAL,FS_NORMAL,8),
    array(2,array($endconicon,"Report","12 days","1 Mar '02","13 Mar '02"),
          "2002-03-01","2002-03-13",FF_ARIAL,FS_NORMAL,8)
);

// Create the bars and add them to the gantt chart
for($i=0; $i<count($data); ++$i) {
    $bar = new
GanttBar($data[$i][0],$data[$i][1],$data[$i][2],$data[$i][3],"[50%]",10);
    if( count($data[$i])>4 )
        $bar->title->SetFont($data[$i][4],$data[$i][5],$data[$i][6]);
    $bar->SetPattern(BAND_RDIAG,"yellow");
    $bar->SetFillColor("gray");
    $bar->progress->Set(0.5);
    $bar->progress->SetPattern(GANTT_SOLID,"darkgreen");
    $bar->title-
>SetCSIMTarget(array('#1'.$i,'#2'.$i,'#3'.$i,'#4'.$i,'#5'.$i),array('11'.$i
,'22'.$i,'33'.$i));
    $graph->Add($bar);
}

// Output the chart
$graph->Stroke();

?>
```

Liefert dieses:



9. Der Cache

JpGraph hat ein eingebautes Cacheing, um die Serverlast zu verringern. Die Verwendung ist relativ simpel. Als erstes muss das Caching aktiviert werden. Dazu müssen in der Configdatei jpg-config.inc 3 Variablen gesetzt werden. (Bei früheren Versionen der JpGraph wurde dies in der jpgraph.php vorgenommen.).

Die Variable CACHE_DIR gibt an, welches Verzeichnis verwendet werden soll. Der Webserver muss Schreibrechte für dieses Verzeichnis besitzen. Die Variablen USE_CACHE und READ_CACHE müssen auf true gesetzt werden. Nun kann man den Cache verwenden.

Um den Cache zu verwenden, muss man beim Initialen Aufruf von Graph nun einen Timeout angeben.

```
$graph = new Graph(300,200,"auto",60);
```

Beim ersten Aufruf wird die Grafik erzeugt und in das Cacheverzeichnis gespeichert. Bei jedem weiteren Aufruf dieses Scipts innerhalb von 60 Minuten wird das Bild aus dem Cacheverzeichnis ausgeliefert und es wird kein weiterer Code für diesen Graphen ausgeführt.

10. Ende

Hier möchte ich enden mit der Einführung in die JpGraph Bibliothek. Wie ihr gesehen habt, ist es wirklich leicht zu recht ansehnlichen Ergebnissen zu kommen – gleichzeitig ist sie aber extrem leistungsfähig und dazu noch sehr gut dokumentiert. Alles zusammen also ein Stück Software, bei dem es sich wirklich lohnt, sich damit zu beschäftigen.

11. Links

- http://www.aditus.nu/jpgraph
  die JpGraph-Bibliothek
- http://dev.macxonline.net/gdlib/
  Einführung in die GB-Lib von David (auch verlinkt unter http://www.phpug-hannover.de/material.php)
- http://www.aditus.nu/jpgraph/jpg_proversion.php
  Informationen zur Lizenz von JpGraph