JpGraph

Table des matières

La librairie PHP JpGraph est une librairie graphique orientée objet de haut niveau permettant de réaliser dynamiquement des graphiques depuis PHP. Elle est installée conjointement avec la librairie graphique GD (supportant la manipulation d'images JPEG et PNG depuis PHP).

1. Introduction

JpGraph est une librairie PHP dédiée à la représentation graphique de données.

JpGraph produit des images. C'est au programmeur de fournir l'ensemble des données nécessaires à la réalisation du graphique voulu.

Si JpGraph peut s'utiliser gratuitement dans le cadre d'un développement non-commercial, il est toutefois nécessaire d'acquérir une licence dans le cadre d'un projet où des profits sont générés.

2. Installation

Vous pouvez télécharger les sources de JpGraph à l'adresse suivante : Téléchargement de la librairie JpGraph.

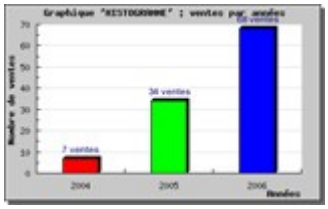
L'archive à télécharger est une archive « .tar.gz ». Pour les utilisateurs de Windows, si vous ne disposez pas d'un programme pour ouvrir un fichier tar.gz, vous pouvez télécharger le logiciel de compression/décompression de données 7-zip.

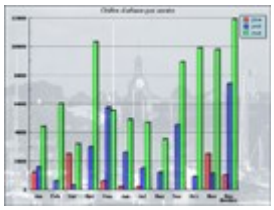
Une fois les fichiers extraits de l'archive, la librairie JpGraph se présente sous la forme d'un ensemble de fichiers PHP, d'un répertoire contenant de nombreux exemples ainsi qu'un répertoire de langues.

Pour fonctionner, JpGraph nécessite que l'extension GD2 (librairie graphique dédiée à la création et à la manipulation d'images) soit disponible.

3. Les principaux graphiques

Voici un aperçu des différents graphiques :

Aperçu	Type de graphique	Description
	**Secteur**	Nombre d'unités vendues sur les différents exercices (2004, 2005 ...)
	**Secteur 3D**	Nombre d'unités vendues sur les différents exercices (2004, 2005 ...)
	**Histogrammes**	Nombre d'unités vendues sur les différents exercices (2004, 2005 ...)

	**Courbe**	Chiffre d'affaires de l'année 2006 par mois
	**Histogrammes groupés**	Chiffre d'affaires par mois et par années sur les différents exercices (2004, 2005 ...)
	**Histogrammes horizontaux**	répartition des ventes par chiffre d'affaires entre les différents types de produits depuis la première vente
	**Histogrammes et courbe**	Chiffre d'affaires et unités vendues pour l'année 2006
	**Histogrammes accumulés**	Chiffre d'affaires 2006 avec répartition du type de vente
	**Radar**	Chiffre d'affaires de l'année 2006

Pour utiliser JpGraph depuis vos applications PHP il faut les includes suivants :

```
include "jpgraph.php";          // librairie JpGraph
include "xxx.php";              // plot extension xxx (voir documentation)
```

- ces graphiques seront générés au format JPEG ou PNG (le format GIF n'étant pas supporté en écriture par la librairie GD utilisée par JpGraph)
- n'utilisez que les "polices de caractère built-in" et pas les polices TrueType afin que les scripts que vous développez soient ensuite portables sur d'autres sites

3.1. Graphique "secteur"

Nous allons voir comment réaliser un **graphique type secteur** (appelé également graphique camembert).

Pour ce premier exemple, **le graphique sera construit étape par étape**. Le processus en sera volontairement plus détaillé que dans les exemples suivants.

Ce dont nous avons besoin pour produire les données :

1. Il faut d'abord **récupérer les données**, à l'aide d'une requête SQL.
2. Sachant que JpGraph attend un tableau de données, nous allons **intégrer ces valeurs dans deux tableaux.**
3. Extraire de la table des ventes les étiquettes (différentes années p r exemple) permettra d'**ajouter une légende** à chaque section du graphique.

Passons ensuite à la **mise en œuvre du graphique** avec la librairie. L'élaboration d'un graphique est assez simple :

1. **Créer un objet** qui représente le type de graphique que nous souhaitons obtenir (ici un PieGraph).
2. **Renseigner différentes propriétés** du graphique, comme les données nécessaires à sa représentation, les légendes, la position du graphique dans son conteneur et bien d'autres paramètres dont nous aurons un aperçu au cours de cet article.
3. Enfin, **provoquer l'envoi de l'image vers le navigateur**.

Voyons le code source du graphique :

```
<?php
// ********************************************************************
// PARTIE : Includes et initialisation des variables
// ********************************************************************

// Inclusion de la librairie JpGraph
include ("../jpGraph/jpgraph.php");
include ("../jpGraph/jpgraph_pie.php");

// Constantes (connection mysql)
define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

// Tableaux de données destinées à JpGraph
$tableauAnnees = array();
$tableauNombreVentes = array();

// ********************************************************************
// PARTIE : Production des données avec Mysql
// ********************************************************************

$sql = <<<EOF
      SELECT
            YEAR(`DTHR_VENTE`) AS ANNEE,
            COUNT(ID) AS NBR_VENTES
```

```
      FROM `ventes`
      GROUP BY YEAR(`DTHR_VENTE`)
EOF;

// Connexion à la BDD
$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

// Sélection de la base de données
@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

// Requête
$mysqlQuery = @mysql_query($sql, $mysqlCnx)
      or die('Pb de requête');

// Fetch sur chaque enregistrement
while ($row = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      // Alimentation des tableaux de données
      $tableauAnnees[] = 'Année ' . $row['ANNEE'];
      $tableauNombreVentes[] = $row['NBR_VENTES'];
      }

// ************************************************************************
// PARTIE : Création du graphique
// ************************************************************************

// On spécifie la largeur et la hauteur du graphique conteneur
$graph = new PieGraph(400,300);

// Titre du graphique
$graph->title->Set("Volume des ventes par années");

// Créer un graphique secteur (classe PiePlot)
$oPie = new PiePlot($tableauNombreVentes);

// Ajouter au graphique le graphique secteur
$graph→Add($oPie);

// Légendes qui accompagnent chaque secteur, ici chaque année
$oPie->SetLegends($tableauAnnees);

// position du graphique (légèrement à droite)
$oPie->SetCenter(0.4);

$oPie->SetValueType(PIE_VALUE_ABS);

// Format des valeurs de type entier
$oPie->value->SetFormat('%d');

// Provoquer l'affichage (renvoie directement l'image au navigateur)
$graph->Stroke();
?>
```

Comme nous l'avons vu au tout début du script PHP, il est nécessaire d'inclure la librairie JpGraph ainsi que la librairie nécessaire à la production du graphique voulu. Dans le cas présent, il s'agit d'un graphique secteur (camembert), nous avons donc inclu le fichier **jpgraph_pie.php**.

```
// Inclusion de la librairie JpGraph
include ("../jpGraph/jpgraph.php");
include ("../jpGraph/jpgraph_pie.php");
```

Afin de créer notre graphique, nous avons utilisé le constructeur de la classe **PieGraph** (elle-même une extension de la classe Graph). Les arguments que nous avons passés sont la largeur et la hauteur.

```
$graph = new PieGraph(400,300);
```

La variable **$graph** est à comprendre au sens de « conteneur s» pour un graphique de type piegraph.

Après avoir effectué quelques paramétrages, nous passons à la création du secteur proprement dit :

```
$oPie = new PiePlot($tableauNombreVentes);
```

L'argument pris est le tableau que nous avons alimenté avec les données issues de notre table des ventes.

Nous ajoutons notre graphique secteur au conteneur.

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

Là encore, nous allons effectuer quelques paramétrages, comme la **légende** et la **position** du graphique.

```
// Légendes qui accompagnent chaque secteur, ici chaque année
$oPie->SetLegends($tableauAnnees);

// position du graphique (légèrement à droite)
$oPie->SetCenter(0.4);
```

Il est également possible de spécifier la façon dont les **valeurs de chaque part** sont restituées, soit de façon **absolue** (comme c'est le cas dans notre exemple) soit de façon **proportionnelle**, avec un pourcentage pour chaque part.

Nous avons choisi la représentation absolue. Pour cela, on utilise la méthode **SetValueType()** et on passe en argument la constante correspondante.

```
$oPie->SetValueType(PIE_VALUE_ABS);
```

La méthode **SetFormat('format')** permet de formater les valeurs en les représentant sous divers formats comme entier ou flottant, mais aussi d'accompagner les valeurs d'une chaîne.

```
// Format des valeurs de type entier
$oPie->value->SetFormat('%d');
```

Pour finir, la méthode **Stroke()** provoque l'affichage du graphique.

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

Cette méthode provoque un envoi de l'image (avec header PHP) directement au navigateur. Pas question, donc, de faire autre chose que de produire l'image dans votre script. Nous verrons néanmoins qu'il est tout à fait possible de créer une image sur disque dur si cette méthode ne vous convient pas. Le format par défaut pour l'image produite par défaut est png.

3.2. Graphique "secteur 3D"

Dans cet exemple, l'objectif est de créer **le même type de graphique** et d'**en personnaliser l'affichage**.

Nous allons donc reprendre les données utilisées précédemment (je ne m'attarderai pas sur la partie PHP, dans laquelle on effectue la récupération des données).

```
<?php
include ("../jpGraph/jpgraph.php");
include ("../jpGraph/jpgraph_pie.php");
include ("../jpGraph/jpgraph_pie3d.php");

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$tableauAnnees = array();
$tableauNombreVentes = array();

// ****************************************************
// Extraction des données dans la base de données
// ****************************************************

$sql = <<<EOF
      SELECT
            YEAR(`DTHR_VENTE`) AS ANNEE,
            COUNT(ID) AS NBR_VENTES
      FROM `ventes`
      GROUP BY YEAR(`DTHR_VENTE`)
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

$mysqlQuery = @mysql_query($sql, $mysqlCnx)
      or die('Pb de requête');

while ($row = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      // Ajouter année devant, c'est pour la légende
      $tableauAnnees[] = "année " . $row['ANNEE'];
      $tableauNombreVentes[] = $row['NBR_VENTES'];
      }

// ************************************
// Création du graphique
// ************************************

// On spécifie la largeur et la hauteur du graph
$graph = new PieGraph(400,300);
```

```php
// Ajouter une ombre au conteneur
$graph->SetShadow();

// Donner un titre
$graph->title->Set("Volume des ventes par années style PIE 3D");

// Quelle police et quel style pour le titre
// Prototype: function SetFont($aFamily,$aStyle=FS_NORMAL,$aSize=10)
// 1. famille
// 2. style
// 3. taille
$graph->title->SetFont(FF_GEORGIA,FS_BOLD, 12);

// Créer un camembert
$pie = new PiePlot3D($tableauNombreVentes);

// ajouter le graphique PIE3D au conteneur
$graph->Add($pie);

// Quelle partie se détache du reste
$pie->ExplodeSlice(2);

// Spécifier des couleurs personnalisées... #FF0000 ok
$pie->SetSliceColors(array('red', 'blue', 'green'));

// Légendes qui accompagnent le graphique, ici chaque année avec sa couleur
$pie->SetLegends($tableauAnnees);

// Position du graphique (0.5=centré)
$pie->SetCenter(0.4);

// Type de valeur (pourcentage ou valeurs)
$pie->SetValueType(PIE_VALUE_ABS);

// Personnalisation des étiquettes pour chaque partie
$pie->value->SetFormat('%d ventes');

// Personnaliser la police et couleur des étiquettes
$pie->value->SetFont(FF_ARIAL,FS_NORMAL, 9);
$pie->value->SetColor('blue');

// Provoquer l'affichage
$graph->Stroke();
?>
```

Fichiers nécessaires à la réalisation du graphique :

```php
include ("../jpGraph/jpgraph.php");
include ("../jpGraph/jpgraph_pie.php");
include ("../jpGraph/jpgraph_pie3d.php");
```

A l'aide de la méthode **SetShadow()**, nous avons ajouté une ombre au conteneur.

Ensuite, nous avons spécifié la police, le style et la taille avec la méthode **SetFont()** :

Pour la création de l'objet représentant notre graphique en 3D, nous avons fait appel à un constructeur différent **`PiePlot3D`**, rendu possible grâce à l'inclusion du fichier

jpgraph_pie3d.php.

L'effet de séparation d'une des parties du graphique a pu être réalisé à l'aide de la méthode **ExplodeSlice()**, qui prend en argument l'index de la partie que l'on souhaite valoriser.

La personnalisation des couleurs se fait avec la méthode **SetSliceColors()**, qui prend comme argument un tableau indexé de chaînes de caractères (les différentes couleurs).

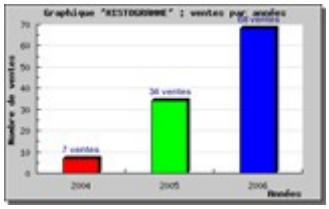
Les couleurs peuvent également être définies en RGB, notation utilisée en Html, par exemple : #FF0000 pour du rouge

Il est également possible de paramétrer la position du graphique, grâce à la méthode **SetCenter()**. Une valeur de 0.5 correspond à un centrage par rapport au conteneur. Plus la valeur est élevée, plus le graphique est décalé à droite (et inversement). Dans l'exemple, le graphique est légèrement décentré à gauche pour laisser de la place à la légende.

JpGraph permet enfin de personnaliser les valeurs présentées, puisque nous avons pu spécifier le texte de sortie ainsi que la police et la couleur.

```
$pie->value->SetFormat('%d ventes');
$pie->value->SetFont(FF_ARIAL,FS_NORMAL, 9);
$pie->value->SetColor('blue');
```

3.3. Graphique "histogrammes"



Après avoir étudié le graphique secteur (ou camembert), voyons maintenant le graphique de type histogramme. Cette sorte de graphique est nommée 'Bar' dans la librairie JpGraph.

Pour créer un graphique de ce type, il faut, en plus de l'inclusion de la librairie de base de JpGraph (fichier jpgraph.php), procéder à l'inclusion du fichier **jpgraph_bar.php**.

Le graphique en histogramme présente les chiffres sous forme de barres verticales (ou horizontales).

Pour ce qui est de la production de données, pas besoin de modifier le code : ce qui était valable pour le graphique secteur l'est également pour l'histogramme.

Le code PHP :

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$tableauAnnees = array();
$tableauNombreVentes = array();

// ********************************************
// Extraction des données dans la base de données
// ********************************************

$sql = <<<EOF
      SELECT
            YEAR(`DTHR_VENTE`) AS ANNEE,
```

```
            COUNT(ID) AS NBR_VENTES
      FROM `ventes`
      GROUP BY YEAR(`DTHR_VENTE`)
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

$mysqlQuery = @mysql_query($sql, $mysqlCnx)
      or die('Pb de requête');

while ($row = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      $tableauAnnees[] = 'Année ' . $row['ANNEE'];
      $tableauNombreVentes[] = $row['NBR_VENTES'];
      }

// ******************
// Création du graphique
// ******************

// Construction du conteneur
// Spécification largeur et hauteur
$graph = new Graph(400,250);

// Réprésentation linéaire
$graph->SetScale("textlin");

// Ajouter une ombre au conteneur
$graph->SetShadow();

// Fixer les marges
$graph->img->SetMargin(40,30,25,40);

// Création du graphique histogramme
$bplot = new BarPlot($tableauNombreVentes);

// Ajouter les barres au conteneur
$graph->Add($bplot);

// Spécification des couleurs des barres
$bplot->SetFillColor(array('red', 'green', 'blue'));

// Une ombre pour chaque barre
$bplot->SetShadow();

// Afficher les valeurs pour chaque barre
$bplot->value->Show();
// Fixer l'aspect de la police
$bplot->value->SetFont(FF_ARIAL,FS_NORMAL,9);
// Modifier le rendu de chaque valeur
$bplot->value->SetFormat('%d ventes');

// Le titre
$graph->title->Set("Graphique 'HISTOGRAMME' : ventes par années");
$graph->title->SetFont(FF_FONT1,FS_BOLD);
```

```
// Titre pour l'axe horizontal(axe x) et vertical (axe y)
$graph->xaxis->title->Set("Années");
$graph->yaxis->title->Set("Nombre de ventes");

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

// Légende pour l'axe horizontal
$graph->xaxis->SetTickLabels($tableauAnnees);

// Afficher le graphique
$graph->Stroke();
?>
```

Après avoir créé notre conteneur comme dans les autres exemples, nous devons fixer la représentation souhaitée pour l'échelle avec la méthode **SetScale()**, logarithmique (log) ou bien linéaire (nous choisirons ici la représentation linéaire).

La construction du graphique proprement dit passe par la création d'un objet de type **BarPlot**. Le constructeur prend comme argument un tableau de valeurs entières.

La méthode **SetShadow()** permet l'ajout d'une ombre à chaque histogramme :

Voici comment afficher la valeur de chaque histogramme : `$bplot->value->Show();`

L'ajout au conteneur du graphique avec la méthode **Add() :** `$graph->Add($bplot);`

Il est également possible de **spécifier des légendes pour les deux axes** : **x** pour l'axe horizontal et **y** pour l'axe vertical.

```
$graph->xaxis->title->Set("Années");
$graph->yaxis->title->Set("Nombre de ventes");
```

Enfin, la méthode **SetTickLabels()** permet de personnaliser les valeurs de légende d'un axe (dans notre exemple, les différentes années). Un tableau indexé des valeurs correspondantes est nécessaire en argument.

3.4. Graphique "courbe"

Abordons un type de graphique que l'on rencontre très fréquemment : **le graphique de type courbe**. Ce type de graphique, constitué de points reliés entre eux est particulièrement approprié pour représenter une progression.

La production des données s'avère un petit peu plus difficile à réaliser cette fois puisque nous souhaitons :

- Récupérer les ventes de l'année 2006
- Comptabiliser le chiffre d'affaires généré par mois

Il nous faudra aussi créer la requête SQL appropriée.

Une fois la production de données effectuée, le reste ne présente pas de difficulté particulière.

Le code PHP :

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$tableauAnnees = array();
$tableauNombreVentes = array();
$moisFr = array('JAN', 'FEV', 'MAR', 'AVR', 'MAI', 'JUI', 'JUL', 'AOU', 'SEP',
'OCT', 'NOV', 'DEC');

// *********************
// Production de données
// *********************

$sql_ventes_par_mois = <<<EOF
      SELECT
            MONTH( `DTHR_VENTE` ) AS MOIS,
            COUNT( `ID` ) AS NOMBRE_VENTE,
            SUM( `PRIX` ) AS PRODUIT_VENTE
      FROM VENTES
      WHERE YEAR( `DTHR_VENTE` ) = '2006'
      GROUP BY MOIS
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

// Initialiser le tableau à 0 pour chaque mois ***********************
$tableauVentes2006 = array(0,0,0,0,0,0,0,0,0,0,0,0);

$mysqlQuery = @mysql_query($sql_ventes_par_mois, $mysqlCnx)
      or die('Pb de requête');

while ($row_mois = mysql_fetch_array($mysqlQuery, MYSQL_ASSOC))
      $tableauVentes2006[$row_mois['MOIS']-1] = $row_mois['PRODUIT_VENTE'];

// ***********************
// Création du graphique
// ***********************

// Création du conteneur
$graph = new Graph(500,300);

// Fixer les marges
$graph->img->SetMargin(40,30,50,40);

// Mettre une image en fond
$graph->SetBackgroundImage("images/mael_white.png",BGIMG_FILLFRAME);

// Lissage sur fond blanc (évite la pixellisation)
$graph->img->SetAntiAliasing("white");
```

```
// A détailler
$graph->SetScale("textlin");

// Ajouter une ombre
$graph->SetShadow();

// Ajouter le titre du graphique
$graph->title->Set("Graphique 'courbes' : volume des ventes 2006");

// Afficher la grille de l'axe des ordonnées
$graph->ygrid->Show();
// Fixer la couleur de l'axe (bleu avec transparence : @0.7)
$graph->ygrid->SetColor('blue@0.7');
// Des tirets pour les lignes
$graph->ygrid->SetLineStyle('dashed');

// Afficher la grille de l'axe des abscisses
$graph->xgrid->Show();
// Fixer la couleur de l'axe (rouge avec transparence : @0.7)
$graph->xgrid->SetColor('red@0.7');
// Des tirets pour les lignes
$graph->xgrid->SetLineStyle('dashed');

// Apparence de la police
$graph->title->SetFont(FF_ARIAL,FS_BOLD,11);

// Créer une courbes
$courbe = new LinePlot($tableauVentes2006);

// Ajouter la courbe au conteneur
$graph→Add($courbe);

// Afficher les valeurs pour chaque point
$courbe->value->Show();

// Valeurs: Apparence de la police
$courbe->value->SetFont(FF_ARIAL,FS_NORMAL,9);
$courbe->value->SetFormat('%d');
$courbe->value->SetColor("red");

// Chaque point de la courbe ****
// Type de point
$courbe->mark->SetType(MARK_FILLEDCIRCLE);
// Couleur de remplissage
$courbe->mark->SetFillColor("green");
// Taille
$courbe->mark->SetWidth(5);

// Couleur de la courbe
$courbe->SetColor("blue");
$courbe->SetCenter();

// Paramétrage des axes
$graph->xaxis->title->Set("Mois");
$graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->SetTickLabels($moisFr);
```

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

Remarque : Il est nécessaire de **prévoir les cas où aucune vente n'a eu lieu** pour un mois donné.

Pour ce faire, on initialise un tableau indexé de 12 valeurs (indice 0 à indice 11) par la valeur zéro. Lors de la récupération, les données (chiffre d'affaires et nombre de ventes) sont affectées au mois qui convient.

```
$tableauVentes2006 = array(0,0,0,0,0,0,0,0,0,0,0,0);

while ($row_mois = mysql_fetch_array($mysqlQuery, MYSQL_ASSOC))
      $tableauVentes2006[$row_mois['MOIS']-1] = $row_mois['PRODUIT_VENTE'];
```

Le reste concerne la mise en œuvre du graphique et sa personnalisation.

La création d'une courbe se fait à l'aide du constructeur **LinePlot($donnees)**, qui prend en argument un ensemble de données (un tableau indexé).

Une image a été ajoutée en fond avec la méthode **SetBackgroundImage().**

La position de l'image est déterminé par la constante passée en second argument. Les différentes valeurs possibles sont :

CONSTANTE	SIGNIFICATION
BGIMG_FILLPLOT	L'image est placée dans la zone du tracé et non dans le conteneur. Elle est redimensionnée au besoin
BGIMG_FILLFRAME	L'image est placée sur l'ensemble de l'image et est redimensionnée au besoin
BGIMG_COPY	L'image est placée telle quelle, sans être centrée ni redimensionnée
BGIMG_CENTER	L'image est centrée sur le graphique mais n'est pas redimensionnée

Paramétrer l'anti-aliasing avec une couleur dominante permet d'éviter la pixellisation (effet escalier) de la courbe tracée (il faut penser à spécifier une couleur proche de la couleur de fond) .

Nous avons également fait apparaître un quadrillage en fond du graphique (sur les deux axes) bleu pour l'axe des abscisses et rouge pour l'axe des ordonnées :

```
$graph->ygrid->Show();
$graph->ygrid->SetColor('blue@0.7');
$graph->ygrid->SetLineStyle('dashed');
$graph->xgrid->Show();
$graph->xgrid->SetColor('red@0.7');
$graph->xgrid->SetLineStyle('dashed');
```

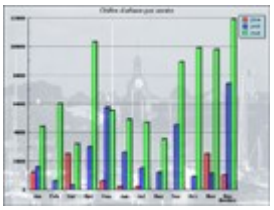
Comme vous pouvez le constater, nous avons changé l'**apparence des points** pour obtenir des ronds remplis de vert:

```
$courbe->mark->SetType(MARK_FILLEDCIRCLE);
$courbe->mark->SetFillColor("green");
$courbe->mark->SetWidth(5);
```

Enfin, pour chaque mois, nous avons affiché une représentation textuelle (un tableau de 12 valeurs crées en début de script) pour l'axe des abscisses.

```
$moisFr = array('JAN', 'FEV', 'MAR', 'AVR', 'MAI', 'JUI', 'JUL', 'AOU', 'SEP',
'OCT', 'NOV', 'DEC');
$graph->xaxis->SetTickLabels($moisFr);
```

3.5. Graphique "histogrammes" groupés



Dans le même esprit que le graphique type courbe, revenons aux histogrammes pour étudier une forme particulière : **les histogrammes groupés**.

Comme son nom l'évoque, ce type de graphique permet la représentation de **plusieurs histogrammes groupés dans le même conteneur**. Cela peut s'avérer très utile lorsque l'on souhaite comparer différentes valeurs.

Nous allons de nouveau représenter le **chiffre d'affaires pour l'ensemble des années concernées** (dans notre base de données) et non plus pour une année donnée.

Les données ne sont pas très difficiles à produire. En effet, une partie du travail a déjà été effectuée dans l'exemple précédent (nous reprendrons une partie de ce travail en ajoutant une étape pour récupérer les années).

Récapitulons ce dont nous avons besoin :

1. Récupérer les **années disponibles**
2. Pour chaque année, récupérer les **valeurs de chaque mois**
3. Pour chaque mois, comptabiliser le **chiffre d'affaires**

Voici le code :

```
<?php

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

// Tableau de données (années et chiffre d'affaires)
$tableauAnnees = array();
$tableauNombreVentes = array();

// **********************
// Extraction des données
// **********************

// Les années
$sql_annees = <<<EOF
      SELECT YEAR( `DTHR_VENTE` ) AS ANNEE
      FROM VENTES
      GROUP BY ANNEE
EOF;

// Pour chaque année récupérer le chiffre d'affaires
```

```
$sql_ventes_par_mois = <<<EOF
	SELECT
		MONTH( `DTHR_VENTE` ) AS MOIS,
		COUNT( `ID` ) AS NOMBRE_VENTE,
		SUM( `PRIX` ) AS PRODUIT_VENTE
	FROM VENTES
	WHERE YEAR( `DTHR_VENTE` ) = %s
	GROUP BY MOIS
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
	or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
	or die('Pb de sélection de la base');

$mysqlQuery = @mysql_query($sql_annees, $mysqlCnx)
	or die('Pb de requête');

// Faire pour chaque année
while ($row_annees = mysql_fetch_array($mysqlQuery, MYSQL_ASSOC)) {
	// Initialiser le tableau à 0 pour chaques mois
	$tableauVentesParAnnees[$row_annees['ANNEE']] = array(0,0,0,0,0,0,0,0,0,0,0,0);

	// Récupérer le chiffre d'affaires par mois de l'année en cours
	$mysqlQuery2 = @mysql_query(sprintf($sql_ventes_par_mois,
$row_annees['ANNEE']), $mysqlCnx) or die('Pb de requête');

	while ($row_mois = mysql_fetch_array($mysqlQuery2, MYSQL_ASSOC))
		$tableauVentesParAnnees[$row_annees['ANNEE']][$row_mois['MOIS']-1] =
$row_mois['PRODUIT_VENTE'];
	}

// **********************
// Création du graphique
// **********************

// Création du graphique conteneur
$graph = new Graph(640,480,'auto');

// Type d'échelle
$graph->SetScale("textlin");

// Fixer les marges
$graph->img->SetMargin(60,80,30,40);

// Positionner la légende
$graph->legend->Pos(0.02,0.05);

// Couleur de l'ombre et du fond de la légende
$graph->legend->SetShadow('darkgray@0.5');
$graph->legend->SetFillColor('lightblue@0.3');

// Obtenir le mois (localisation fr possible ?)
$graph->xaxis->SetTickLabels($gDateLocale->GetShortMonth());

// Afficher une image de fond
$graph->SetBackgroundImage('images/R0011940.jpg',BGIMG_COPY);
```

```
// AXE X
$graph->xaxis->title->Set('Années');
$graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->title->SetColor('black');
$graph->xaxis->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->SetColor('black');

// AXE Y
$graph->yaxis->SetFont(FF_FONT1,FS_BOLD);
$graph->yaxis->SetColor('black');
$graph->ygrid->SetColor('black@0.5');

// TITRE: texte
$graph->title->Set("Chiffre d'affaires par année");

// TITRE: marge et apparence
$graph->title->SetMargin(6);
$graph->title->SetFont(FF_ARIAL,FS_NORMAL,12);

// Couleurs et transparence par histogramme
$aColors=array('red@0.4', 'blue@0.4', 'green@0.4', 'pink@0.4', 'teal@0.4',
'navy@0.4');

$i=0;

// Chaque  histogramme est un élément du tableau:
$aGroupBarPlot = array();

foreach ($tableauVentesParAnnees as $key => $value) {
      $bplot = new BarPlot($tableauVentesParAnnees[$key]);
      $bplot->SetFillColor($aColors[$i++]);
      $bplot->SetLegend($key);
      $bplot->SetShadow('black@0.4');
      $aGroupBarPlot[] = $bplot;
      }

// Création de l'objet qui regroupe nos histogrammes
$gbarplot = new GroupBarPlot($aGroupBarPlot);

// Ajouter au graphique
$graph->Add($gbarplot);
$gbarplot->SetWidth(0.8);

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

La principale difficulté de la création de ce graphique réside dans la construction du groupement d'histogramme. En particulier, pour bien découpler production de données et création du graphique, il a fallu créer une structure (ici un tableau associatif) susceptible de contenir les données produites.

Il faut également que notre structure soit adaptée à la restitution des données effectuée au moment de la mise en œuvre du graphique.

La forme choisie est un **tableau associatif** qui possède pour clé chaque année récupérée dans la base. A chaque année correspond un tableau indexé de 12 valeurs (pour chacun des mois de l'année). Ces valeurs représentent le chiffre d'affaires du mois concerné.

Voici un aperçu du tableau (obtenu avec la fonction **print_r($tableau)**) :

```
Array
(
    [2004] => Array
        (
            [0] => 1200
            [1] => 0
            [2] => 2500
            [3] => 0
            [4] => 600
            [5] => 200
            [6] => 170
            [7] => 0
            [8] => 0
            [9] => 0
            [10] => 2500
            [11] => 1000
        )
...
)
```

Pour chaque année (itération dans le foreach) est créé un objet **BarPlot**. Un paramètre est passé lors de la construction de l'objet. Ce paramètre est le tableau de valeurs correspondant à l'année courante.

```
$bplot = new BarPlot($tableauVentesParAnnees[$key]);
```

Chaque 'BarPlot' est paramétré : **couleur de fond, légende et ombre**

```
$bplot->SetFillColor($aColors[$i++]);
$bplot->SetLegend($key);
$bplot->SetShadow('black@0.4');
```

Le BarPlot est ajouté à un conteneur que nous utiliserons juste après.

```
$aGroupBarPlot[] = $bplot;
```

Enfin, la construction des histogrammes groupés avec l'objet **GroupBarPlot** qui prend en argument le tableau qui contient tous nos BarPlot.

```
$gbarplot = new GroupBarPlot($aGroupBarPlot);
```

3.6. Graphique "histogrammes" horizontaux

Abordons maintenant le graphique de type **histogrammes horizontaux**.

Sa mise en œuvre est rigoureusement identique à celle du type graphique "histogramme" vertical.

Cette fois, nous allons représenter graphiquement la **répartition des ventes par chiffre d'affaires entre les différents types de produits depuis la première vente**.

Pour chaque entrée de la table ventes, le type de vente est spécifié. Nous allons nous appuyer sur ce champ pour grouper les résultats en faisant, pour chaque type de vente, la somme du chiffre d'affaires.

Nous souhaitons également faire apparaître les dates de la première et de la dernière vente concernées. Pour cela, nous ferons une autre requête.

Voyons le script :

```
<?php

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$tabTypesProduits = array();
$tabChiffreAffaire = array();

// *********************************
// Extraction des données dans la base
// *********************************

// Chiffre d'affaires par produit
$sql_ventes_par_produits = <<<EOF
      SELECT `TYPE_PRODUIT` , SUM( `PRIX` ) AS CHIFFRE_AFFAIRE
      FROM ventes
      GROUP BY `TYPE_PRODUIT`
EOF;

// Date première vente et date dernière vente
$sql_max_and_min_date = <<<EOF
      SELECT
            DATE_FORMAT(MIN(`DTHR_VENTE`), '%d/%m/%Y') AS MIN_DATE,
            DATE_FORMAT(MAX(`DTHR_VENTE`), '%d/%m/%Y') AS MAX_DATE
      FROM VENTES
EOF;

// Connexion au serveur mysql
$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

// Sélectionner la base de données "tuto_jp_graph"
@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

// Lancer la requête SQL
$mysqlQuery = @mysql_query($sql_max_and_min_date, $mysqlCnx)
      or die('Pb de requête: ' . mysql_error());

$row = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC);

$min_date = $row['MIN_DATE'];
$max_date = $row['MAX_DATE'];

// Lancer l'autre requête
$mysqlQuery = @mysql_query($sql_ventes_par_produits, $mysqlCnx)
      or die('Pb de requête: ' . mysql_error());

// Récupération du résultat
```

```php
while ($row = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      $tabTypesProduits[] = $row['TYPE_PRODUIT'];
      $tabChiffreAffaire[] = $row['CHIFFRE_AFFAIRE'];
      }

// **********************
// Création du graphique
// **********************

// Paramètres du graphique conteneur
$graph = new Graph(450,400,'auto');
$graph->SetScale("textlin");
$graph->SetShadow();

// Mise à 90 degrés et marges
$graph->Set90AndMargin(70,30,90,30);

// Titre et sous-titre
$graph->title->Set("Répartition des ventes par type de produit");
$graph->title->SetFont(FF_VERDANA,FS_BOLD,10);
$graph->subtitle->Set(sprintf("Entre le %s et le %s", $min_date, $max_date));

// Axe x  ******************************
$graph->xaxis->SetTickLabels($tabTypesProduits);
$graph->xaxis->SetFont(FF_VERDANA,FS_NORMAL,10);
$graph->xgrid->Show(true,true);

// Marges pour les étiquettes
// Par rapport au bord droit du graphique
$graph->xaxis->SetLabelMargin(10);

// AXE y ******************************
// Echelle
$graph->yaxis->scale->SetGrace(20);
// Angle des étiquettes
$graph->yaxis->SetLabelAngle(45);
$graph->yaxis->SetLabelFormat('%d');
$graph->yaxis->SetFont(FF_VERDANA,FS_NORMAL,8);

// Création d'une série d'histogramme
$bplot = new BarPlot($tabChiffreAffaire);

// Ajouter au conteneur
$graph->Add($bplot);

// Alternance de couleur
$bplot->SetFillColor(array("green","red", "blue"));

// Ombres
$bplot->SetShadow();

// Epaisseur des histogrammes
$bplot->SetWidth(0.5);

// Valeurs de chaque histogramme
$bplot->value->Show();
$bplot->value->SetFont(FF_ARIAL,FS_BOLD,12);
$bplot->value->SetColor("blue","darkred");
```

```
$bplot->value->SetFormat('%d Eur.');

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

Pour la production de données, nous avons exécuté une requête qui a produit l'ensemble de données suivant :

TYPE_PRODUIT	CHIFFRE_AFFAIRE
logiciel	25900
materiel	32640
service	62880

Ces valeurs ont été placées dans un tableau intermédiaire afin de fournir les données nécessaires à la création du graphique. Les types de produits seront utilisés pour afficher la légende de l'axe des abscisses et les données serviront pour les différents histogrammes.

```
while ($row = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      $tabTypesProduits[] = $row['TYPE_PRODUIT'];
      $tabChiffreAffaire[] = $row['CHIFFRE_AFFAIRE'];
      }
$graph->xaxis->SetTickLabels($tabTypesProduits);
$bplot = new BarPlot($tabChiffreAffaire);
```

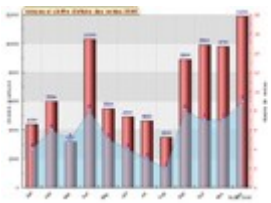
Concernant le paramétrage du graphique proprement dit, nous avons placé le graphique à l'horizontale grâce à l'instruction suivante **Set90AndMargin(...)**

Il a été possible d'ajouter un **sous-titre** pour indiquer les dates de première et de dernière vente (**subtitle->Set(...)**)

Nous avons également modifié l'échelle avec la méthode **SetGrace(...)** afin que les histogrammes laissent de la place à chaque valeur correspondante.

Vous pouvez remarquer que les valeurs illustrant les graduations en haut du graphique sont placées à **45 degrés** en utilisant la méthode **SetLabelAngle(45)**

3.7. Graphique "histogrammes" et "courbe"



Voyons maintenant la possibilité qu'offre JpGraph d'afficher **deux graphiques dans le même conteneur**, en l'occurrence un graphique de type histogramme et un graphique de type courbe.

Pour cet exemple nous allons représenter pour l'année 2006 :

1. Le chiffre d'affaires par mois (histogrammes)
2. Le nombre de ventes par mois (courbe)

La principale difficulté que nous allons rencontrer est la **cohabitation de deux graphiques** qui n'ont pas la même échelle.

Comme nous l'avons vu dans un des exemples précédents, il est possible de récupérer dans une

requête les deux types de données nécessaires à la production du graphique (chiffre d'affaires et nombre de ventes).

Le script :

```
<?php

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$volume_Ventes2006 = array();
$CA_Ventes2006 = array();
$maxVentes = 0;

// **********************
// Extraction des données
// **********************

$sql_ventes_par_produits = <<<EOF
      SELECT
            MONTH( `DTHR_VENTE` ) AS MOIS,
            COUNT( TYPE_PRODUIT ) AS VOLUME_VENTES,
            SUM( PRIX ) AS PRODUIT_VENTES
      FROM VENTES
      WHERE YEAR( `DTHR_VENTE` ) = 2006
      GROUP BY MONTH( `DTHR_VENTE` )
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

$mysqlQuery = @mysql_query($sql_ventes_par_produits, $mysqlCnx)
      or die('Pb de requête: ' . mysql_error());

// Initialisation des tableaux
for ($i=0;$i<=11;$i++) {
      $volume_Ventes2006[$i] = 0;
      $CA_Ventes2006[$i] = 0;
      }

// Récupération des données
while ($row_type_produits = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      $volume_Ventes2006[$row_type_produits['MOIS']-1] =
$row_type_produits['VOLUME_VENTES'];
      $CA_Ventes2006[$row_type_produits['MOIS']-1] =
$row_type_produits['PRODUIT_VENTES'];

      // Récupérer le maximum des ventes
      if ($maxVentes < $row_type_produits['VOLUME_VENTES'])
            $maxVentes = $row_type_produits['VOLUME_VENTES'];
```

```
}

// *********************
// Création du graphique
// *********************

$graph = new Graph(640,480);
$graph->SetScale("textlin");

$graph->SetMargin(50,40,20,40);

// Désactiver le cadre autour du graphique
$graph->SetFrame(false);

// Ajouter un onglet
$graph->tabtitle->Set("Volume et chiffre d'affaire des ventes 2006");
$graph->tabtitle->SetFont(FF_ARIAL,FS_BOLD,10);

// Apparence des grilles
$graph->ygrid->SetFill(true,'#DDDDDD@0.5','#BBBBBB@0.5');
$graph->ygrid->SetLineStyle('dashed');
$graph->ygrid->SetColor('gray');
$graph->xgrid->Show();
$graph->xgrid->SetLineStyle('dashed');
$graph->xgrid->SetColor('gray');
$graph->xaxis->SetTickLabels($gDateLocale->GetShortMonth());
$graph->xaxis->SetFont(FF_ARIAL,FS_NORMAL,8);
$graph->xaxis->SetLabelAngle(45);

// ******************************
// Créer un graphique histogramme
// ******************************

$oHisto = new BarPlot($CA_Ventes2006);

// Ajouter au graphique
$graph->Add($oHisto);

$oHisto->SetWidth(0.6);
$oHisto->value->Show();
$oHisto->value->SetFormat('%d');

// dégradé de rouge vers noir à gauche
// Pour chaque barre
$oHisto->SetFillGradient('#440000', '#FF9090', GRAD_LEFT_REFLECTION);

// Bordure autour de chaque histogramme
$oHisto->SetWeight(1);

// ***********************
// Graphique courbe rempli
// ***********************

$oCourbe = new LinePlot($volume_Ventes2006);

// Couleur de remplissage avec transparence
$oCourbe->SetFillColor('skyblue@0.5');
```

```
// Couleur de la courbe
$oCourbe->SetColor('navy@0.7');
$oCourbe->SetBarCenter();

// Apparence des points
$oCourbe->mark->SetType(MARK_SQUARE);
$oCourbe->mark->SetColor('blue@0.5');
$oCourbe->mark->SetFillColor('lightblue');
$oCourbe->mark->SetSize(6);

// Affichage des valeurs
$oCourbe->value->Show();
$oCourbe->value->SetFormat('%d');

// Echelle des Y pour le nombre de ventes
$graph->SetYScale(0,'lin', 0, $maxVentes * 2);

$graph->xaxis->title->Set("Mois 2006");
$graph->yaxis->title->Set("Chiffre d'affaire");

// Ajouter un axe Y supplémentaire
$graph->AddY(0,$oCourbe);

// Couleur de l'axe Y supplémentaire
$graph->ynaxis[0]->SetColor('red');
$graph->ynaxis[0]->title->Set("Nombre de ventes");
// Envoyer au navigateur
$graph->Stroke();
?>
```

Nous obtenons les données suivantes :

MOIS	VOLUME_VENTES	PRODUIT_VENTES
1	4	4390
2	6	5990
3	5	3200
4	8	10300
5	5	5500
6	4	4900
7	3	4660
8	2	3500
9	8	8900
10	7	9900
11	7	9790
12	9	11890

On trouve dans cet ensemble de données tout ce dont on a besoin : **le mois, le volume et le chiffre d'affaires des ventes**.

Pour la mise en œuvre du graphique, vous pouvez constater que nous avons désactivé l'affichage du

cadre avec la méthode **SetFrame(false)**

Enlever le cadre nous permet aussi d'**ajouter un onglet** au graphique et de le personnaliser :

```
$graph->tabtitle->Set("Volume et chiffre d'affaire des ventes 2006");
$graph->tabtitle->SetFont(FF_ARIAL,FS_BOLD,10);
```

Le graphique histogramme a été créé et ajouté au conteneur comme dans les exemples précédents.

En revanche, l'ajout au graphique de la courbe est quelque peu différent de ce que l'on a eu l'occasion de voir.

On utilise la méthode **addY($donnes)**

L'utilisation de cette méthode permet la création d'une autre axe des ordonnées (y) et d'ajouter un autre graphique dans le même conteneur.

Il est possible d'avoir à **spécifier une autre échelle** pour les graphiques ajoutés. Cela dépend de l'aspect final. Ici, c'est ce que nous avons fait pour des raisons esthétiques, le graphique courbe recouvrant l'autre graphique.

Dans notre cas de figure, il est possible de changer l'échelle avec la méthode **SetYScale()**

Cette méthode prend 4 arguments :

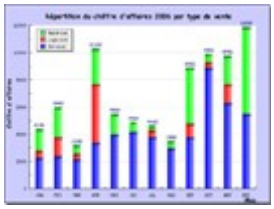
Numéro d'argument	Signification
Arg 1	Indice de l'axe
Arg 2	Type de l'axe (linéaire ou logarithmique)
Arg 3	Valeur min
Arg 4	Valeur max

Pour un affichage harmonieux dans le contexte, le nombre de ventes maximum pour un mois a été multiplié par deux. Cela a eu l'effet recherché : **aplatir le graphique**.

Pour finir, l'axe nouvellement créé peut, lui aussi, être personnalisé (couleur et titre) :

```
$graph->ynaxis[0]->SetColor('red');
$graph->ynaxis[0]->title->Set("Nombre de ventes");
```

3.8. Graphique "histogrammes" accumulés



Une autre caractéristique du graphique type histogramme est la possibilité de représenter un histogramme contenant *n* partie : **les histogrammes accumulés**. Autre caractéristique du graphique type histogramme: la possibilité de représenter un histogramme contenant n parties: **les histogrammes accumulés**.

Pour illustrer cet exemple, nous allons reprendre la représentation du **chiffre d'affaires 2006 par mois**. Mais cette fois-ci, nous ajouterons une information supplémentaire, à savoir le chiffre d'affaires généré pour chaque type de produits.

Par conséquent, nous aurons pour chaque mois un histogramme qui représente le chiffre d'affaire pour tous les types de produits. L'histogramme sera décomposé en parts qui représenteront le chiffre d'affaires généré pour chaque produit.

La production de données est un peu particulière puisqu'il va nous falloir obtenir 3 ensembles de valeurs, le C.A. pour le matériel, pour le logiciel et pour le service de chaque mois.

Ensuite, comme nous allons le voir dans la mise en œuvre du graphique, il suffira de créer un ensemble d'histogrammes pour chaque type de produit et de produire un ensemble d'histogrammes accumulés à partir des histogrammes précédemment créés.

Voici le script :

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$tab_service = array();
$tab_logiciel = array();
$tab_materiel = array();
$moisFr = array('JAN', 'FEV', 'MAR', 'AVR', 'MAI', 'JUI', 'JUL', 'AOU', 'SEP',
'OCT', 'NOV', 'DEC');

// **********************
// Extraction des données
// **********************

$sql_ventes_par_produits = <<<EOF
      SELECT
            MONTH( `DTHR_VENTE` ) AS MOIS,
            UCASE( TYPE_PRODUIT ) AS TYPE_PRODUIT,
            SUM( PRIX ) AS CHIFFRE_AFFAIRES
      FROM VENTES
      WHERE YEAR( `DTHR_VENTE` ) = 2006
      GROUP BY MOIS, TYPE_PRODUIT
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

$mysqlQuery = @mysql_query($sql_ventes_par_produits, $mysqlCnx)
      or die('Pb de requête: ' . mysql_error());

// initialiser les tableaux
for ($i=0;$i<12;$i++) {
      $tab_service[$i] = 0;
      $tab_logiciel[$i] = 0;
      $tab_materiel[$i] = 0;
      }

// remplir les tableaux
while ($row_type_produits = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      if ($row_type_produits['TYPE_PRODUIT'] == 'SERVICE')
                  $tab_service[$row_type_produits['MOIS']-1] =
```

```
$row_type_produits['CHIFFRE_AFFAIRES'];
      if ($row_type_produits['TYPE_PRODUIT'] == 'MATERIEL')
                  $tab_materiel[$row_type_produits['MOIS']-1] =
$row_type_produits['CHIFFRE_AFFAIRES'];
      if ($row_type_produits['TYPE_PRODUIT'] == 'LOGICIEL')
                  $tab_logiciel[$row_type_produits['MOIS']-1] =
$row_type_produits['CHIFFRE_AFFAIRES'];
      }

// **********************
// Création du graphique
// **********************
$graph = new Graph(640,480,"auto");
$graph->SetScale("textlin");
$graph->SetShadow();
$graph->img->SetMargin(60,40,50,40);
$graph->SetMarginColor('#CCCCFF');
$graph->title->Set("Répartition du chiffre d'affaires 2006 par type de vente");
$graph->title->SetMargin(20);
$graph->title->SetFont(FF_COMIC,FS_BOLD,12);

// Créer les ensembles d'histogrammes
$histo_service = new BarPlot($tab_service);
$histo_service->SetFillGradient('blue', '#9090FF', GRAD_VER);
$histo_service->SetLegend('Service');
$histo_service->value->Show();
$histo_service->value->SetFont(FF_ARIAL, FS_NORMAL,7);
$histo_service->value->SetColor('black');
$histo_service->value->SetFormat('%d');

$histo_logiciel = new BarPlot($tab_logiciel);
$histo_logiciel->SetFillGradient('red', '#FF9090', GRAD_VER);
$histo_logiciel->SetLegend('Logiciel');
$histo_logiciel->value->Show();
$histo_logiciel->value->SetFont(FF_ARIAL, FS_NORMAL,7);
$histo_logiciel->value->SetColor('black');
$histo_logiciel->value->SetFormat('%d');

$histo_materiel = new BarPlot($tab_materiel);
$histo_materiel->SetFillGradient('green', '#90FF90', GRAD_VER);
$histo_materiel->SetLegend('Matériel');
$histo_materiel->value->Show();
$histo_materiel->value->SetFont(FF_ARIAL, FS_NORMAL,7);
$histo_materiel->value->SetColor('black');
$histo_materiel->value->SetFormat('%d');

// Créer l'ensemble d'histogrammes accumulés
$gbplot = new AccBarPlot(array($histo_service, $histo_logiciel, $histo_materiel));

// Ajouter l'ensemble accumulé
$graph->Add($gbplot);

// Afficher les valeurs de chaque histogramme groupé
$gbplot->value->Show();
$gbplot->value->SetFont(FF_COMIC,FS_NORMAL,8);
$gbplot->value->SetFormat('%d');

// Position de la légende
```

```
$graph->legend->Pos(0.12,0.12,"left","top");

// Paramétrer les axes X et Y
$graph->yaxis->title->Set("Chiffre d'affaires");
$graph->yaxis->title->SetMargin(20);
$graph->yaxis->title->SetFont(FF_COMIC,FS_BOLD);

$graph->xaxis->title->Set("Mois");
$graph->xaxis->SetTickLabels($moisFr);
$graph->xaxis->title->SetMargin(4);
$graph->xaxis->title->SetFont(FF_COMIC,FS_BOLD);

// Afficher l'image générée
$graph->Stroke();
?>
```

Dans notre exemple, il a été fait le choix de faire une seule requête pour extraire toutes les informations nécessaires. Nous aurions tout aussi bien pu effectuer une requête par type de produit (matériel, logiciel et service).

La requête produite retourne pour chaque mois (de l'année 2006) le chiffre d'affaires des différents types de produits. Cela donne l'ensemble de données suivant (aperçu des les 4 premiers mois) :

MOIS	TYPE_PRODUIT	CHIFFRE_AFFAIRES
1	LOGICIEL	500
1	MATERIEL	1600
1	SERVICE	2290
2	LOGICIEL	1400
2	MATERIEL	2200
2	SERVICE	2390
3	LOGICIEL	400
3	MATERIEL	600
3	SERVICE	2200
4	LOGICIEL	4300
4	MATERIEL	2600
4	SERVICE	3400

La seconde étape est de séparer les données pour **créer un tableau par type de produit** :

```
$tab_service = array();
$tab_logiciel = array();
$tab_materiel = array();

// initialiser les tableaux
for ($i=0;$i<12;$i++) {
      $tab_service[$i] = 0;
      $tab_logiciel[$i] = 0;
      $tab_materiel[$i] = 0;
      }

// remplir les tableaux
while ($row_type_produits = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC)) {
      if ($row_type_produits['TYPE_PRODUIT'] == 'SERVICE')
            $tab_service[$row_type_produits['MOIS']-1] =
$row_type_produits['CHIFFRE_AFFAIRES'];
      if ($row_type_produits['TYPE_PRODUIT'] == 'MATERIEL')
```

```
            $tab_materiel[$row_type_produits['MOIS']-1] =
$row_type_produits['CHIFFRE_AFFAIRES'];
      if ($row_type_produits['TYPE_PRODUIT'] == 'LOGICIEL')
            $tab_logiciel[$row_type_produits['MOIS']-1] =
$row_type_produits['CHIFFRE_AFFAIRES'];
      }
```

La couleur du fond a été personnalisée en utilisant la méthode **SetMarginColor()**

Chaque histogramme (matériel, logiciel et service) a été produit en créant un objet **BarPlot** en prenant comme argument les ensembles de données correspondants.

Après avoir généré les histogrammes par type de produits, nous les avons regroupés en créant une instance de l'objet AccBarPlot, lui-même prenant en argument un tableau des histogrammes tout justes produits.

La position par défaut de la légende ne convenait pas (elle recouvrait un des histogrammes). La méthode **Pos()** a permis de déplacer la légende à gauche.

3.9. Graphique "radar"

Voyons un autre type de graphique : le **graphique de type radar**. Les valeurs passées à ce graphique sont représentées par des points placés circulairement autour d'un point central. A chaque valeur correspond un axe.

La production de données ne pose pas de problème, car le code source est quasiment le même que celui que l'on a vu plus haut.

La mise en œuvre du graphique ne présente pas non plus de difficulté particulière.

Le code :

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

define('MYSQL_HOST', 'localhost');
define('MYSQL_USER', 'root');
define('MYSQL_PASS', '');
define('MYSQL_DATABASE', 'tuto_jp_graph');

$tab_materiel = array();
$moisFr = array('JAN', 'FEV', 'MAR', 'AVR', 'MAI', 'JUI', 'JUL', 'AOU', 'SEP',
'OCT', 'NOV', 'DEC');

// **********************
// Extraction des données
// **********************

$sql_ventes_par_produits = <<<EOF
      SELECT
            MONTH( `DTHR_VENTE` ) AS MOIS,
            UCASE( TYPE_PRODUIT ) AS TYPE_PRODUIT,
            SUM( PRIX ) AS CHIFFRE_AFFAIRES
      FROM VENTES
      WHERE YEAR( `DTHR_VENTE` ) = 2006
            AND
            TYPE_PRODUIT = 'MATERIEL'
```

```
      GROUP BY MOIS, TYPE_PRODUIT
EOF;

$mysqlCnx = @mysql_connect(MYSQL_HOST, MYSQL_USER, MYSQL_PASS)
      or die('Pb de connxion mysql');

@mysql_select_db(MYSQL_DATABASE)
      or die('Pb de sélection de la base');

$mysqlQuery = @mysql_query($sql_ventes_par_produits, $mysqlCnx)
      or die('Pb de requête : ' . mysql_error());

// initialiser le tableau
for ($i=0;$i<12;$i++)
      $tab_materiel[$i] = 0;

// remplir le tableau
while ($row_type_produits = mysql_fetch_array($mysqlQuery,  MYSQL_ASSOC))
      if ($row_type_produits['TYPE_PRODUIT'] == 'MATERIEL')
            $tab_materiel[$row_type_produits['MOIS']-1] =
$row_type_produits['CHIFFRE_AFFAIRES'];

// *********************
// Création du graphique
// *********************

// Création du conteneur type radar
$graph = new RadarGraph (480,480,"auto");

// Représentation logarithmique
// Permet le lissage en changeant l'échelle
$graph->SetScale("log");

// Paramétrage de l'apparence du grahique
$graph->title->Set("C.A. des ventes de matériel 2006");
$graph->title->SetFont(FF_VERDANA,FS_NORMAL,12);
$graph->SetTitles($moisFr);

// Position du graphique par rapport au centre
$graph->SetCenter(0.35,0.55);
// Cacher les marques
$graph->HideTickMarks();
// Couleur de fond
$graph->SetColor('#cccccc@0.3');
$graph->axis->SetColor('blue@0.5');
$graph->grid->SetColor('blue@0.5');
$graph->grid->Show();
$graph->axis->title->SetFont(FF_ARIAL,FS_NORMAL,10);
$graph->axis->title->SetMargin(5);

// Créer les points
$plot1 = new RadarPlot($tab_materiel);

// Ajouter les points au graphique
$graph->Add($plot1);

// Couleur de la ligne
$plot1->SetColor('red');
```

```
// Epaisseur de la ligne qui relie les points
$plot1->SetLineWeight(1);
// Couleur de remplissage
$plot1->SetFillColor('red@0.8');
// Apparence des points
$plot1->mark->SetType(MARK_SQUARE);

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

JpGraph permet la **représentation logarithmique d'un graphique**.

Pour cela, nous utiliserons la méthode **SetScale('log')** (changement du type de l'échelle) de l'objet créé lors de la construction du graphique **.**

Remarque : l'échelle par défaut est 'lin' comme linear, c'est à dire linéaire.

4. Fonctions mathématiques

La création de représentation graphique d'une fonction mathématique nécessite une quantité de code à produire dérisoire.

Prenons pour l'exemple la représentation graphique de la fonction mathématique x^2.

Pour créer le graphique, il faudra inclure les librairies suivantes :

1. jpgraph.php
2. jpgraph_line.php
3. jpgraph_utils.inc.php

La création de ce type de graphique est simple :

- Création de la fonction à l'aide d'une méthode dédiée
- Ensuite, génération des différents points qui constitueront la courbe.
- Enfin, comme pour un graphique courbe, paramétrage et affichage de la représentation graphique.

Voici le code :

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

// Générer une fonction
$f = new FuncGenerator('$x*$x');

// Créer une série de points
list($xdata,$ydata) = $f->E(-9,9);

// Nouveau conteneur
$graph = new Graph(450,350,"auto");
$graph->SetScale("linlin");

// Titre
$graph->title->Set('Exemple de fonction : "x au carré"');

// Placer l'axe Y au centre
$graph->yaxis->SetPos(0);
```

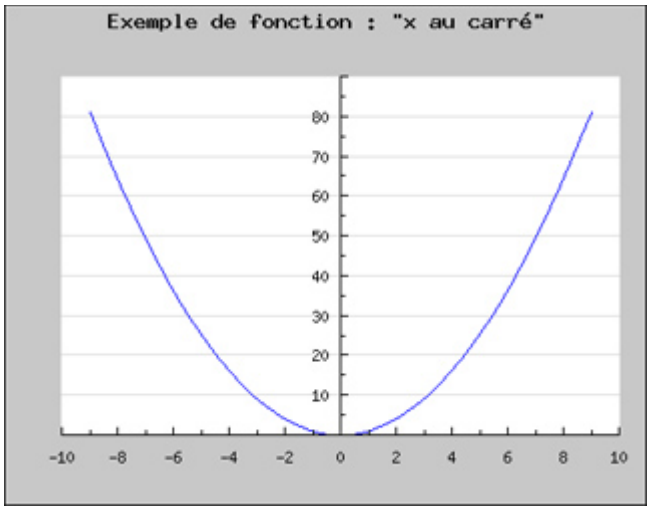
```
// Cacher première et dernière légende
$graph->yaxis->HideFirstLastLabel();

// Créer une courbe
$courbe = new LinePlot($ydata,$xdata);

// Ajouter la courbe au graphique
$graph→Add($courbe);
$courbe->SetColor('blue');

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

Résultat :



Revenons sur les quelques instructions spécifiques à ce type de graphique :

D'abord création de la fonction mathématique « x au carré » :

```
$f = new FuncGenerator('$x*$x');
```

ensuite vient la création de la liste de points correspondant en passant un point de référence. Ici on utilise un point de coordonnées -9 pour les abscisses et 9 pour les ordonnées. Cette liste de points est composée d'un tableau indexé d'une cinquantaine de valeurs pour chaque axe.
La méthode retourne donc deux tableaux :

```
list($xdata,$ydata) = $f->E(-9,9);
```

5. Quelques astuces

5.1. Créer une image sur le disque dur

Vous avez pu remarquer qu'un script produit par défaut une image directement renvoyée vers le navigateur.

Il est possible de modifier ce comportement en spécifiant le nom de l'image en argument de la méthode **stroke()**.

```
$graph->Stroke('graphique_secteur.png');
```

5.2. Changer le format des images

JpGraph crée par défaut ses images au format png. Là encore, il est possible de changer ce comportement en précisant le format avec la méthode **SetImgFormat(format)** :

```
$graph->img->SetImgFormat('gif');
```

5.3. Passer des paramètres à la création d'une image

L'utilisation courante de JpGraph se fait en spécifiant le nom du script PHP comme source de l'image crée :

```
<img src="monGraphique.php" width="240" height="240" />
```

Il peut être avantageux de passer des paramètres lors de la production Html car cela permet de rendre la création du graphique encore plus dynamique :

```
<img src="monGraphique.php?2004=7&2005=34&2006=68" width="240" height="240" />
```

Ici, on reprend les données exploitées dans l'exemple « Histogramme » (§ 3.3.).

Dans ce cas, les données seront dans ce cas récupérées dans le script responsable de la création du graphique, en utilisant la variable superglobale **$_GET** ou **$_REQUEST**

```
$data_2004 = $_GET['2004']; // Valeur = 7
$data_2005 = $_GET['2005']; // Valeur = 34
$data_2006 = $_GET['2006']; // Valeur = 69
...
```

6. Exemples en ligne

- http://enacit1.epfl.ch/php/jpgraph/src/Examples/testsuit.php?type=1
- http://enacit1.epfl.ch/php/jpgraph/src/Examples/testsuit.php?type=2