

# Manipularea dinamică a imaginilor în PHP

## Un exemplu de reprezentare pe Web a graficelor de funcții

– Sabin Corneliu Buraga

În cadrul acestui articol, ne propunem să ilustrăm o parte dintre facilitățile oferite de serverul de aplicații PHP 4 în ceea ce privește prelucrarea imaginilor. Pentru aceasta vom scrie un script PHP care va putea reprezenta grafic oricărei funcții a cărei formulă va fi introdusă de utilizator prin intermediul unui formular.

Dar mai întâi să facem cunoștință cu funcțiile de manipulare a imaginilor. PHP pune la dispoziție o pleiadă de funcții pentru desena de puncte, linii, poligoane, arce de cerc, pentru controlul culorii și fonturilor, plus posibilitatea de a încărca și salva imagini în popularele formate GIF (*Graphics Interchange Format*), PNG (*Portable Network Graphics*) sau JPEG (*Joint Picture Experts Group*). Singura cerință este să fie instalată biblioteca GD în sistem. Pentru *gd-1.6* sau versiuni mai vechi se suportă numai formatul GIF, iar pentru versiuni mai recente se oferă suport pentru formatul PNG. Prelucrarea fișierelor în formatul JPEG se va putea realiza dacă este instalată biblioteca *jpeg-6b*. În marea majoritate a cazurilor, aceste biblioteci sunt disponibile în orice distribuție actuală de Linux.

Unele dintre cele mai utile funcții sunt:

- `imagecreate()` – va crea o imagine de dimensiuni precizate de programator și va returna un identificator de imagine, similar identificatorului de fișier (*handler*) returnat de primitivile `creat()` sau `open()`. Toate celelalte funcții de prelucrare a imaginilor vor folosi acest identificator (număr întreg).
- `imagecreatefromgif()` – va crea o imagine pornind de la un fișier în format GIF care va fi încărcat. Funcția va returna un identificator de imagine în caz de succes sau 0 în caz de eroare.
- `imagecreatefrompng()` – ca mai sus, dar se va încărca o imagine în format PNG.
- `imagecreatefromjpeg()` – ca mai sus, dar pentru imagini în format JPEG.
- `imagesx()` – furnizează lățimea unei imagini (în pixeli).
- `imagesy()` – furnizează înălțimea unei imagini (în pixeli).
- `getimagesize()` – returnează dimensiunea unui fișier în format GIF, PNG sau JPEG, plus lățimea și înălțimea imaginii conținute de acesta. Funcția nu necesită prezența bibliotecii GD.
- `imagetypes()` – furnizează ce tipuri de formate grafice sunt suportate de versiunea PHP care rulează pe serverul Web.
- `imagesetpixel()` – va desena un pixel având o anumită culoare la coordonatele precizate. Punctul de coordonate (0, 0) se consideră a fi în stânga-sus. Culoarea va fi un întreg desemnând un identificator de culoare returnat de funcția `imagecolorallocate()`.
- `imageline()` – va reprezenta un segment de dreaptă dat de coordonatele a două puncte, cu o anumită culoare. Pentru a desena linii întrerupte se poate utiliza `imagedashedline()`.
- `imagerectangle()` – va desena o formă rectangulară.
- `imagepolygon()` – va desena un poligon; coordonatele punctelor vor fi furnizate prin intermediul unui tablou.
- `imagearc()` – va desena o elipsă parțială, această funcție putând fi folosită și pentru reprezentarea arcelor de cerc.

- `imagechar()` – va insera un singur caracter în cadrul imaginii la coordonatele specificate, utilizându-se familia de fonturi implicită (se pot stabili diferite dimensiuni de reprezentare). Pentru a desena vertical un caracter se poate utiliza funcția `imagecharup()`.
- `imagestring()` – va avea aceeași acțiune ca și `imagechar()`, dar va fi figurat un șir de caractere, nu doar un unic caracter. Pentru a reprezenta un șir de caractere scris vertical vom folosi funcția `imagestringup()`.
- `imagefill()` – reprezintă o funcție de umplere cu o anumită culoare, pornind de la coordonatele precizate. Alte funcții înrudite cu aceasta sunt `imagefilledrectangle()`, `imagefilledpolygon()` sau `imagefilltoborder()`.
- `imagecolorallocate()` – va alocă un identificator de culoare pentru imagine, pornind de la componentele RGB corespunzătoare (trei întregi între 0 și 255). Alte funcții utile sunt `imagecolorat()`, `imagecolorset()`, `imagecolorexact()` sau `imagecolorresolve()`.
- `imagecolordeallocate()` – este funcția care va elibera un identificator de culoare alocat în prealabil de funcția `imagecolorallocate()`.
- `imagecolortransparent()` – va stabili care culoare va fi considerată transparentă pentru o anumită imagine (utilă pentru generarea de GIF-uri transparente).
- `imagecolorstotal()` – furnizează numărul total de culori ale paletii asociate unei imagini.
- `imageinterlace()` – setează sau inhibă proprietatea de întrețesere (*interlace*) a unei imagini.
- `imagecopy()` – va copia o zonă dintr-o imagine și o va amplasa în cadrul altei imagini, posibil la alte coordonate. Pentru a modifica și dimensiunile zonei copiate (efectul *resize*) se poate folosi `imagecopyresized()`.
- `imagegif()` – reprezintă imaginea desemnată de un identificator de imagine la ieșirea standard (se trimite direct navigatorului Web) sau o salvează într-un fișier stocat în sistemul de fișiere al serverului Web. Implicit, formatul va fi GIF87a. Dacă s-a setat proprietatea de transparentă prin funcția `imagecolortransparent()`, imaginea va fi generată în format GIF89a.
- `imagepng()` – ca mai sus, dar se va salva în formatul PNG.
- `imagejpeg()` – ca mai sus, dar se va salva în formatul JPEG (se dă posibilitatea programatorului de a stabili și gradul de compresie, cu pierderi, a imaginii rezultate).
- `imagedestroy()` – distruge identificatorul de imagine și eliberează zona de memorie alocată informațiilor despre aceasta.

### Reprezentarea graficelor de funcții

Folosind o parte dintre funcțiile enumerate mai sus, vom concepe un script PHP capabil să reprezinte grafic orice funcție a cărei formulă va fi furnizată de utilizator. Acesta va introduce prin intermediul unui formular XHTML formula funcției de reprezentat (de exemplu:  $A * x * x + B$  pentru a reprezenta grafic funcția  $f(x) = Ax^2 + B$ ) și valorile constantelor care intervin în această formulă ( $A = 20$ ;  $B = 5$ ; de pildă – va trebui introdus cod PHP, constantele din formulă fiind de fapt variabile al căror nume va trebui prefixat de simbolul „\$“). Vom conveni ca toate

### Listing 1. Exemplu de script PHP

```

<?php
/* Reprezentarea grafica a unei functii date de utilizator
*/

// Constante glogale
define ("OX_MIN", 0.1); // valoarea minima pe Ox
define ("OX_MAX", 100); // valoarea maxima pe Ox
define ("PAS", 0.5); // incrementul pe Ox
define ("INALTIME", 300); // inaltimea maxima a imaginii
// generate
define ("LATIME", 150); // latimea maxima a imaginii
// generate

// Functie de generare a functiei de calcul
function generare_functie ($continut)
{
// antetul functiei generate
$antet = "function calcul (\$cod, \$x)\n";
$antet = $antet . "\n";
$antet = $antet . "eval (\$cod);\n";
// partea de sfirsit a functiei generate
$final = "\n\n";
// converteste majusculele in variabile PHP, de asemeni
"x"
$rezult = ereg_replace ("([A-Zx])", "$\\1", $continut);
// generarea functiei
$rezult = $antet . "return (" . $rezult . ");" . $final;
return ($rezult);
}

// Functie de creare a imaginii care va contine
reprezentarea functiei
function creare_image()
{
// culoarea de afisare a graficului (variabila gloabala)
global $culoare_grafic;

$latime = OX_MAX / PAS;
$inaltime = INALTIME;

$image = @imagecreate ($latime, $inaltime)
or die ("Nu exista suport GD");
// alocam culorile de reprezentare:
// pentru fundal - alb
$culoare_fundal = imagecolorallocate ($image, 255, 255,
255);
// pentru axe - gri
$culoare_axe = imagecolorallocate ($image, 120, 120,
120);
// pentru grafic - albastru
$culoare_grafic = imagecolorallocate ($image, 0, 0,
255);

// generam mai intii fundalul
imagefilledrectangle ($image, 0, 0, $latime - 1,
$inaltime - 1,
$culoare_fundal);
// generam axele
imagedashedline ($image, 0, 0, 0, $inaltime-1,
$culoare_axe);
imagedashedline ($image, 0, LATIME, $latime - 1,
LATIME, $culoare_axe);
// afisam valorile pe axa Ox
imagestring ($image, 2, 5, LATIME + 5, "0",
$culoare_axe);
imagestring ($image, 2, $latime - 20, LATIME + 5,
OX_MAX, $culoare_axe);
// afisam valorile pe axa Oy
imagestring ($image, 2, 2, 2, "+" . $inaltime / 2,
$culoare_axe);
imagestring ($image, 2, 2, $inaltime - 20, "-" .
$inaltime / 2, $culoare_axe);
// afisam vertical un text
imagestringup ($image, 3, $latime - 20, $inaltime - 5,
"Grafic", $culoare_axe);

return ($image);
}

// Functie de reprezentare a graficului
function grafic ($image, $x, $y)
{
global $culoare_grafic;
// ultimele coordonate ale graficului
static $x_anterior = OX_MIN;
static $y_anterior = 0;

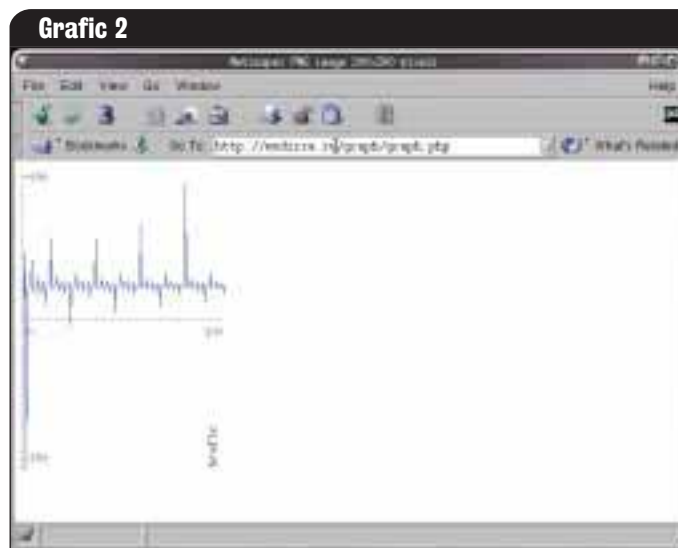
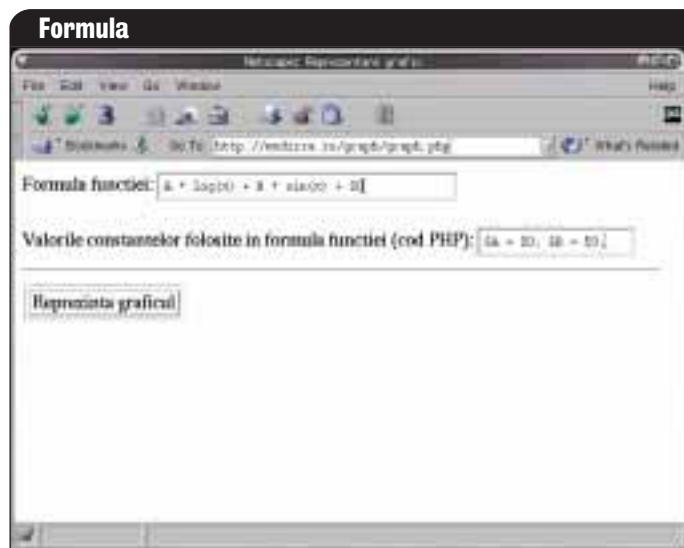
if ($x_anterior != OX_MIN)
imagine ($image, $x_anterior / PAS, INALTIME -
($y_anterior + LATIME),
$x / PAS, INALTIME - ($y + LATIME),
$culoare_grafic);
$x_anterior = $x;
$y_anterior = $y;
}

// Verificam modul de invocare a script-ului
if (!isset ($formula)) // nu a fost furnizata formula
functiei de reprezentat
{
// se va genera formularul XHTML
print ("<html>\n<head><title>Reprezentare
grafic</title></head>\n");
print ("<body bgcolor=\\"white\\" text=\\"black\\">");
print ("<form action=\\"" . basename($PHP_SELF) . "\"
method=\\"post\\">");
print ("<p>Formula functiei: ");
print ("<input type=\\"text\\" name=\\"formula\\" value=\\"A *
x + B\\" size=\\"40\\" />");
print ("<p>Valorile constantelor folosite in formula
functiei (cod PHP): ");
print ("<input type=\\"text\\" name=\\"cod\\" value=\\"$A =
2; $B = 5;\\" />");
print ("<hr /><input type=\\"submit\\" value=\\"Reprezinta
graficul\\" />");
print ("</form>\n</body>\n</html>\n");
}
else // invocarea script-ului la apasarea butonului de tip
'submit'
{
// translatam formula functiei in cod PHP
$functie = generare_functie ($formula);
// echo ("<pre> $functie </pre>");
// evaluam codul generat
eval ($functie);
// cream imaginea care va contine graficul
$image = creare_image();
// construim graficul
for ($x = OX_MIN; $x < OX_MAX; $x += PAS)
{
// apelam functia generata dinamic
$y = calcul ($cod, $x);
grafic ($image, $x, $y);
}
// setam bitul de intretesere a imaginii (interlace)
imageinterlace ($image, 1);
// setam culoarea pentru transparenta
imagecolortransparent ($image, $culoare_fundal);
// trimitem graficul generat catre navigatorul Web:
// setam tipul continutului (MIME)
header ("Content-type: image/png");
// trimitem imaginea
@imagepng ($image) or die ("Acest format nu este
recunoscut");
}
?>

```

constantele să le reprezentăm cu majuscule, iar singura necunoscută să fie  $x$ . Utilizatorul va putea introduce orice funcție matematică recunoscută de interpretorul PHP, deoarece evaluarea formulei funcției se va baza pe funcția PHP `eval()`. Scriptul va genera o funcție `calcul()` conținând codul PHP al formulei funcției de reprezentat. Funcția `cal-`

`cul()` va fi apelată în cadrul procesului de generare a graficului pentru a calcula valoarea  $y = f(x)$  pentru fiecare  $x$  variind între constantele `OX_MIN` și `OX_MAX` cu pasul `PAS`. După figurarea axelor de coordonate și generarea graficului, imaginea va fi trimisă navigatorului Web în formatul PNG, pentru aceasta expediind mai întâi câmpul de antet

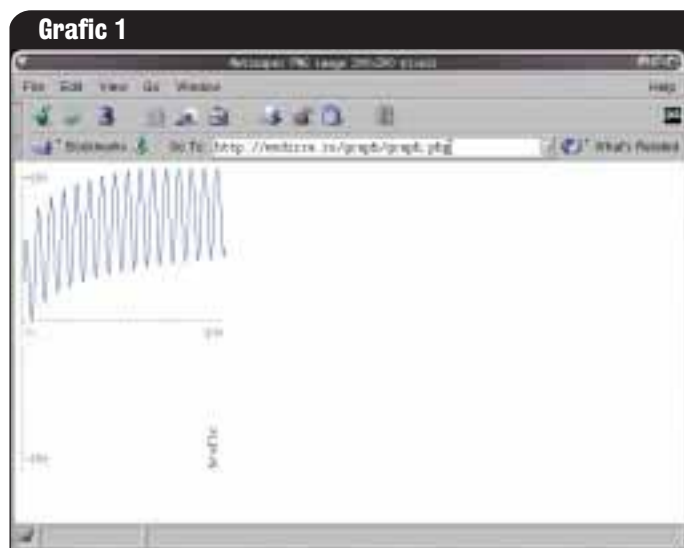


HTTP *Content-type* care va avea valoarea MIME *image/png*. Imaginea PNG va avea setate proprietățile de transparență și întregesere.

Codul sursă al scriptului PHP (folosind mare parte din funcțiile de manipulare grafică descrise mai sus) este prezentat în *Listing 1*.

Imaginea graficului funcției va fi de dimensiuni 200x300, fiind figurate și axele de coordonate și valorile minime și maxime pe axele  $Ox$  și  $Oy$ . Graficul funcției va fi reprezentat cu culoare albastră, iar axele în gri. Funcția de calculare a valorii funcției de reprezentat va fi generată de funcția `generare_functie()`. S-au folosit expresiile regulate – prelucrate prin funcția `ereg_replace()` – pentru ca fiecare constantă care apare în formula funcției de reprezentat să fie prefixată de simbolul „\$” pentru a fi interpretată drept variabilă de către PHP. La fel pentru necunoscuta  $x$ . Astfel, scriptul PHP generează *in mod dinamic* cod PHP pentru fiecare formulă de funcție introdusă de utilizator. Acest cod va fi evaluat de interpretorul PHP prin intermediul funcției predefinite `eval()`. De remarcat faptul că astfel utilizatorul va putea insera *orice cod PHP valid* ceea ce va conduce la breșe de securitate a serverului Web! Recomandăm cititorului să găsească o metodă mai sigură de interpretare a formulei funcției de reprezentat grafic și să se asigure că acea funcție poate fi reprezentată.

Unele apeluri de funcții au fost prefixate de simbolul „@” pentru ca mesajele de eroare să nu fie trimise spre afișare navigatorului. Posibilele mesaje de eroare vor putea fi consultate prin intermediul variabilei globale `$php_errormsg`.



### Exemple de rulare

Exemplele de mai jos au fost executate pe un sistem Linux Red Hat 7.0, având instalat serverul Apache 3.1.14, cu PHP 4.0.2 și biblioteca GD versiunea 1.8.3.

Utilizatorul va putea introduce cele prezentate în figura „Formula”.

În urma apăsării butonului „Reprezinta graficul” va fi generat graficul din figura „Grafic 1”.

Pentru expresia  $(A * \tan(x) + 3) / (C * \log(x) + 2) + 33$  cu  $A = 50$  și  $C = 10$  va fi generată funcția PHP:

```
function calcul ($cod, $x)
{
    eval ($cod);
    return ((A * tan($x) + 3) / (C * log($x) + 2) + 33);
}
```

iar reprezentarea grafică va fi cea din figura „Grafic 2”.

### În loc de concluzii

După cum am văzut din exemplul de mai sus, utilizarea funcțiilor de manipulare grafică puse la dispoziție de PHP nu prezintă dificultăți. De dorit ar fi fost ca programatorul să aibă la dispoziție și altele, pentru a realiza prelucrări mai sofisticate de imagini (de exemplu, modificarea unor proprietăți precum luminozitatea sau contrastul ori posibilitatea de a realiza rotiri sau alte transformări). Dar chiar și așa, PHP oferă câteva instrumente folositoare pentru generarea dinamică a imaginilor și salvarea acestora în diferite formate.

*Sabin Corneliu Buraga este doctorand în Computer Science la Universitatea „Al.I. Cuza” din Iași, putând fi contactat prin e-mail la adresa busaco@infoiasi.ro și pe Web la <http://www.infoiasi.ro/~busaco/>. ■ 98*

### Referințe

- ✓ T.Ratschiller, T.Gerken – „Web Application Development with PHP 4.0”, New Riders Publishing, 2000
- ✓ M.Sârbu – „PHP: Hypertext Preprocessor”, PC Report, vol.9, 08 (95), aug.2000
- ✓ \*\*\* – Biblioteca GD: <http://www.boutell.com/gd/>
- ✓ \*\*\* – Biblioteca jpeg-6b: <ftp://ftp.uu.net/graphics/jpeg/>
- ✓ \*\*\* – PHP.net: <http://www.php.net/>
- ✓ \*\*\* – PHP Builder: <http://www.phpbuilder.com/>