L'intèrpret PHP

Esquema general d'ús de l'intèrpret PHP

L'intèrpret PHP (http://www.php.net) és el programa que s'encarrega d'executar scripts escrits en el llenguatge PHP (http://php.net/manual/es). Aquests scripts es poden escriure de forma autònoma en un fitxer de text per tal de ser executats invocant l'intèrpret des d'un terminal d'Ubuntu com si fossin scripts Bash o Python (l'intèrpret en aquest cas és la comanda *php5*, instal.lable amb el paquet "php5-cli" i configurable des de l'arxiu "/etc/php5/cli/php.ini") però el més habitual és que el codi PHP s'escrigui embegut dins de codi HTML/CSS. Això (incloure codi PHP enmig de codi HTML/CSS) permet ampliar enormement les capacitats i funcionalitat de la pàgina web, convertint-la, de fet, en tot un programa sencer capaç d'accedir a fitxers, a bases de dades, executar comandes, etc.

El procés d'executar una pàgina HTML/CSS+PHP (a partir d'ara, per simplificar, direm "pàgina PHP" amb el benentès de que no es pot escriure una pàgina web sense una mínima estructura HTML obligatòria) és, simplificant molt, així:

1) El client HTTP (un navegador) sol.licita una pàgina PHP (a través de la barra de direccions, per exemple) a un determinat servidor HTTP (Apache, per exemple)

2) Qualsevol servidor HTTP per sí sol només és capaç de retornar pàgines que continguin codi HTML, CSS i/o Javascript. En detectar que la pàgina demanada incorpora codi PHP, li traspassa la pàgina sol.licitada a l'intèrpret PHP

3) L'intèrpret PHP agafa aquesta pàgina i processa tot el codi PHP que hi hagi, obtenint com a resultat una pàgina HTML pura (deixant intacte el possible codi CSS i/o Javascript que hi pugui tenir). Per exemple,si inicialment la pàgina conté una línia PHP tal com *print Date("Y/m/d");* , l'intèrpret la substituirà per la cadena "2015/11/23" (o la data que sigui). Les línies PHP poden realitzar molts tipus d'accions diverses, com ara l'accés a bases de dades (que és el que més veurem en aquest curs); en aquest cas, la pàgina HTML pura retornada contindrà les dades concretes obtingudes després d'haver realitzat la consulta pertinent en aquell moment. En qualsevol cas, la pàgina HTML pura serà retornada per l'intèrpret PHP al servidor HTTP

4) El servidor HTTP ara ja podrà retornar al client la pàgina HTML amb el resultat d'haver executat el codi PHP que tenia inicialment.

NOTA: La gran diferència entre el llenguatge PHP i el llenguatge Javascript és que aquest últim és un llenguatge "client"; és a dir, són els navegadors els qui interpreten codi Javascript (a l'igual que codi HTML i/o CSS). En canvi, el llenguatge PHP és un llenguatge "servidor" perquè només l'intèrpret PHP (invocat per un servidor HTTP) és capaç d'entendre codi escrit en aquest llenguatge: si un navegador per si sol intentés obrir una pàgina PHP no funcionaria correctament perquè no pot entendre-la

Maneres d'executar l'intèrpret PHP dins d'un servidor web

Hi ha diferents maneres de realitzar el punt 2) de la llista anterior (és a dir, de cridar per part del servidor HTTP a l'intèrpret PHP). Depenent dels seus "pros" i "contres", ens interessarà més utilitzar una manera o una altra; i segons la decisió que prenem, la configuració del servidor web serà diferent. Bàsicament, hi ha tres opcions:

A) Cridar a l'intèrpret PHP funcionant com un programa independent de l'executable Apache (mode "CGI"). Això significa que cada cop que s'ha de realitzar el punt 2), l'Apache crida a un executable PHP extern i aquest es posa en marxa sempre com si fos la primera vegada (rellegint un altre cop la seva configuració particular, etc.). L'intèrpret PHP s'executarà amb uns permisos propis diferents dels de l'Apache i, de fet, fins i tot podria funcionar en una màquina separada diferent.

B) Cridar a l'intèrpret PHP funcionant com una llibreria dinàmica interna (en Windows, una "dll"; en Linux una "so") pertanyent al propi executable Apache (mode "mòdul"). Això significa que

l'intèrpret PHP forma part del propi servidor web i, per tant, es posa en marxa en iniciar-se l'Apache (llegint la seva configuració en aquell moment només) i es manté llest mentre l'Apache està funcionant sense la necessitat de cap executable extern. De fet, l'intèrpret PHP s'executarà amb els permisos propis de l'Apache.

C) Darrerament s'està implementant una tercera manera de cridar a l'intèrpret PHP anomenada "FastCGI" (concretament el subtipus "PHP-FPM") el qual és semblant en concepte al "CGI" però molt més ràpid ja que internament utilitza uns sistemes de comunicació entre el servidor Apache i l'intèrpret PHP diferents del mode "CGI" clàssic. Una altra diferència és que en aquest mode l'intèrpret PHP funciona com un servei/dimoni del sistema (en comptes d'un executable estàndar com pasava al mode "CGI" tradicional).

L'avantatge més obvi del mode "mòdul" respecte els altres dos és la velocitat i la no repetició de tasques preparatòries, però també està la facilitat de guardar l'"estat de la sessió" (és a dir, compartir dades comunes) entre dues peticions, cosa que amb els modes (Fast)CGI és més difícil perquè en ells cada petició és "la primera". D'altra banda, l'inconvenient més important del mode "mòdul" és que només es pot fer servir amb el worker "prefork"; a més, en aquest mode el procés Apache incorpora una "motxilla" PHP que sempre ocupa memòria del sistema independentment de si les pàgines demanades són PHP (dinàmiques) o HTML/CSS/Javascript pures (estàtiques).

Instal.lació de l'intèrpret

La instal.lació en el mode "CGI" tradicional no el veurem perquè el mode "FastCGI" és equivalent però millor.
NOTA: A partir d'ara es considerarà que treballem sobre una màquina Ubuntu. A Fedora els noms dels paquets involucrats són diferents: el paquet "php5-cli" d'Ubuntu es converteix en "php-cli" (i l'executable de terminal s'anomena *php* en comptes de *php5)*; el paquet "libapache2-mod-php5" es converteix en "php" i el paquet "php5-fpm" es converteix en "php-fpm".

Instal.lació mode "mòdul":

Podrem fer servir a Ubuntu l'intèrpret PHP que ve en forma de mòdul d'Apache si instal.lem el paquet "libapache2-mod-php5" (aquest paquet té com a dependència, entre d'altres, al paquet "php5-common", comú -tal com diu el seu nom- a la resta de modes de l'intèrpret) i seguidament l'activem com qualsevol altre mòdul (per exemple amb la comanda *a2enmod php5*) i reiniciem el servidor Apache (*systemctl restart apache2*). Ja està.

L'intèrpret/mòdul PHP té un arxiu de configuració anomenat "/etc/apache2/mods-available/php5.conf" que no caldrà modificar però que és interessant estudiar-lo. Per exemple, allà trobem la directiva AddType establerta convenientment per tal d'indicar quins arxius (identificats per les seves extensions) seran tractats com arxius de tipus PHP (concretament, de tipus MIME application/x-httpd-php); aquesta directiva permet, doncs que l´Apache sàpiga què n´ha de fer d´aquests tipus de fitxers (és a dir: que sàpiga que els ha de passar a l´intèrpret/mòdul PHP per a què aquest els processi, o en altres paraules,, la decisió presa al punt 2) de l'esquema de la primera pàgina d'aquest document es defineix amb aquesta directiva).

També hi ha un altre arxiu de configuració més general, "/etc/php5/apache2/php.ini" on podem establir moltes altres coses. Un exemple: definir la zona horària que farà servir l'intèrpret a qualsevol script on hi apareguin dates (línia *date.timezone*, consultar *http://php.net/manual/en/timezones.php*). Un altre exemple (¡MOLT ÚTIL!): activar la possibilitat de veure al navegador els errors de sintaxi presents al nostre codi PHP (línia *display_errors*, molt recomanable posar-la a "On" quan estem desenvolupant la nostra web però molt important posar-la a "Off" en producció, per seguretat). Altres línies interessants són: *error_reporting* (per definir el grau d'importància que ha de tenir l'error per tal de mostrar-lo o no), *error_log* (per definir la ruta del fitxer de registre on es guardaran els missatges d'error; només funciona si *log_error* val "On"), *max_input_time* (per definir el nº màxim de segons que un script es quedarà esperant dades provinents de l'exterior), *max_execution_time* (per definir el nº màxim de segons que un script romandrà executant-se), *memory_limit* (per definir el nº màxim de MB que un script pot consumir de RAM), etc, etc.

Tal com s'ha comentat prèviament, aquest mode de l'intèrpret només funciona amb el MPM "prefork". És per això que es pot veure que a l'hora d'instal.lar aquest mòdul, automàticament es desinstal.la qualsevol altre MPM que hi hagués al nostre servidor i es posa en marxa el "prefork".

Instal.lació mode "FastCGI" (FPM):

Podrem fer servir a Ubuntu l'intèrpret PHP que ve en aquest mode si instal.lem el paquet "php5-fpm" (el qual utilitza el mòdul "proxy_fcgi" de l'Apache per funcionar, però aquest ja ve instal.lat per defecte; també té com a dependència el paquet "php5-common", comú -tal com diu el seu nom- a la resta de modes de l'intèrpret). Un cop instal.lat, la idea és configurar el servei Apache per a què sàpiga reenviar (a través del mòdul "proxy_fcgi") les sol.licituds de pàgines PHP al servei PHP-FPM. Els passos per aconseguir-ho són:
NOTA: Existeixen altres mòduls de tipus FastCGI a més del PHP-FPM, però es configuren de forma diferent i no són tan eficients com ell és per això que no els estudiarem. Concretament, es tracten dels paquets "libapache2-mod-fcgid" (desenvolupat per l'equip d'Apache) i "libapache2-mod-fastcgi" (desenvolupat per http://www.fastcgi.com)

1) Activar (si no ho està ja) el mòdul "proxy_fcgi", així: *a2enmod proxy_fcgi*

2) Esquivar un "bug" de la versió actual d'Apache. Resulta que hi ha dos mètodes per comunicar el servidor web amb el PHP-FPM, i el mètode per defecte ("unix socket") no funciona, així que hem de canviar a l'altra ("tcp/ip socket"). Aquest segon mètode implica que el servidor PHP-FPM estigui escoltant en un port TCP (posarem el 9000) per on rebrà les sol.licitud reenviades pel mòdulo "proxy_fcgi" de l'Apache. Per tant, farem que el PHP-FPM escolti en aquest port modificant la línia "listen" actual de l'arxiu "/etc/php5/fpm/pool.d/www.conf" per aquesta altra: ***listen=127.0.0.1:9000***
NOTA: Un altre arxiu de configuració molt important per l'intèrpret PHP-FPM és "/etc/php5/fpm/php.ini"

3) Dir a l'Apache que redireccioni les peticions de pàgines PHP al port 9000. Això es pot fer a nivell de servidor global, de VirtualHost o de <Directory>...nosaltres optarem per la segona opció. Així doncs, hem d'obrir l'arxiu de configuració del virtualhost escollit i escriure la següent directiva dins de la secció <VirtualHost ...> :
ProxyPassMatch ^/(.*\.php(/.*)?) fcgi://127.0.0.1:9000/ruta/real/de/la/carpeta/DocumentRoot
NOTA: Si no haguéssim d'haver realitzat el pas 2), el valor de la línia anterior hauria d'haver sigut *ProxyPassMatch ^/(.*\.php(/.*)?) unix:/var/run/php5-fpm.sock* Per a més informació, http://wiki.apache.org/httpd/PHP-FPM

4) Iniciar els dos servidors: *systemctl start php5-fpm* i *systemctl start apache2*

Primeres proves

Sigui quina sigui la manera que haguem escollit per instal.lar l'intèrpret PHP, el primer que farem serà comprovar que funcioni. Per això haurem de crear dins del DocumentRoot d'algun VirtualHost actiu del nostre servidor Apache (per defecte això és la carpeta "/var/www/html") un arxiu de texte anomenat "index.php" (vigileu que aquest nom estigui indicat a la directiva DirectoryIndex) amb el següent contingut:

```
<html><body><?php
phpinfo();
?></body></html>
```

Com es pot comprovar, la manera d'incrustar codi PHP dins del codi HTML és mitjançant les etiquetes especials "<?php" (inici codi PHP) i "?>" (final codi PHP). El que fa la funció *phpinfo()* -no oblideu el punt i coma del final...en aquest sentit, la majoria de sintaxi de PHP és heretada de C- és generar una taula informativa de color lila amb multitude de dades de configuració de l'intèrpret PHP. Ara no ens aturarem en esbrinar el significat d'aquestes dades (encara que, per exemple, es pot veure fàcilment a l'apartat "Server/API" quin mode d'intèrpret PHP està funcionant): només ens interessa veure si aquesta taula apareix o no. Per fer això, haurem d'escriure a la barra del navegador una direcció tal com http://nostra.ip (o http://nostra.ip/index.php, si voleu). Si es veu la taula lila, tot ha anat perfecte.

¿Què es veuria al navegador si substituïm la funció *phpinfo();* del codi anterior per la línia *print Date("Y/m/d");?*

Fer possible que l'intèrpret PHP pugui treballar amb servidors MySQL

La instal.lació per defecte de l'intèrpret PHP no proporciona tota la funcionalitat possible del llenguatge. Això fa que per poder utilitzar diferents aspectes del llenguatge PHP calgui instal.lar prèviament algun paquet extra, a mode d'"ampliació" de l'intèrpret bàsic. Això es fa per no engreixar d'entrada un intèrpret del qual potser no aprofitaríem moltes de les seves capacitats (capacitats que estan sempre igualment consumint recursos).

Una d'aquestes funcionalitats que no vénen en la instal.lació per defecte de l'intèrpret PHP és la de poder contactar amb servidors MySQL. Per tant, si volem manipular bases de dades allotjades en aquest tipus de servidors, hauríem d'instal.lar obligatòriament un paquet extra anomenat "php5-mysql" (a Fedora s'anomena "php-mysql") i seguidament, reiniciar l'Apache (en el mode "mòdul") o el PHP-FPM (en el mode "FastCGI") per tal de què el nostre servidor detecti aquesta nova funcionalitat afegida.

Un símptoma clar de no haver fet el pas anterior és voler escriure un codi PHP contenint diferents funcions d'accés i manipulació de bases de dades MySQL i obtenir, en voler-lo mostrar en un navegador, un error del tipus "Undefined function xxxx". Tal com diu aquest missatge, l'error indica que l'intèrpret PHP no reconeix una funció específica, la qual només serà entesa amb la instal.lació de l'"ampliació" corresponent.