02/01/11 - Curs de programació en PHP

Aquest és un petit curs per a aprendre a programar des de 0.

No tothom és capaç d'estructurar la ment i les idees i ser capaç de crear programes, però amb aquest petit curs podràs descobrir si t'agrada i n'ets capaç.

Dedico aquest curs a la Eli i a la Yazmina, dues persones amb moltes ganes d'aprendre i voluntat i capaces.

Introducció: Posant nom a les coses

Què vol dir programar?. De seguida hi arribarem, però primer hem de comprendre coses com què són els ordinadors.

Molt simplificat els ordinadors són uns ginys, que mitjançant uns valors d'entrada, proporcionen uns resultat. Per exemple, enviant-los 1+1 proporcionen un resultat de 2.

Els ordinadors realment funcionen amb impulsos elèctrics, que es transformen en uns i zeros. Si els voltatges són positius representen un 1, en cas contrari un 0.

Imaginem-nos un gran magatzem.

Hi ha una sèrie d'entrades de càrrega i descàrrega que poden ser portes d'entrada (càrrega), de sortida (descàrrega) o d'entrada i sortida (input/output dirien en anglès).

Dins d'aquest gran magatzem hi ha un operari, que seguint instruccions d'un paper, agafa els paquets que li donen per les portes d'entrada, els manipula i els desa als prestatges del magatzem o bé els retorna per una porta de sortida.

Aquest operari és el cervell de l'ordinador o CPU (Central Processing Unit). Les prestatgeries equivaldrien a la memòria. I les portes d'entrada/sortida del magatzem a les portes d'entrada/sortida de l'ordinador (els ports USB són ports d'entrada sortida, el port VGA on connectem el monitor és un port de sortida, etc...)

Com més ràpid és el processador, més feina pot fer per segon. La velocitat dels ordinadors la mesurem en Megahertzos o avui en dia en Gigahertzos.

Les instruccions s'executen als ordinadors marcades per un tic de rellotge de

quartz, que són molt i molt precisos, una mica s'executen seguint el ritme dels tambors de les galeres, cada cop de tambor (clic de rellotge), una instrucció.

El prefix Mega vol dir 1.000 i Giga 1.000.000, i un hertz és una oscil·lació o tick del rellotge per segon, per tant un Gigahertz són un milió d'operacions per segon.

Un ordinador a 2 Ghz hauria de ser més ràpid que un a 1 Ghz. Però no sempre és així. Els nous models fan les operacions cada cop més ràpid, i una ordre que pot tarda 5 clics de rellotge en un ordinador més antic, pot trigar un sol clic de rellotge en un ordinador modern. Per tant ens podem trobar que un ordinador modern a 1 Ghz sigui més ràpid que un ordinador més antic a 2 Ghz.

Per últim, un ordinador amb dual core, seria com dir que al magatzem hi treballen dos operaris independents, per tant, un sol paquet el recullen, el processen i l'entreguen a la mateixa velocitat, però si hi ha més d'un paquet, els dos operaris despatxen dos paquets en el mateix temps que un ordinador amb un sol nucli o core en processen un.

Els ordinadors només fan el que prèviament els hem dit. Els operaris segueixen les instruccions per a manegar els paquets amb una llista d'ordres, i aquesta llista d'ordres és el que anomenem un programa.

Programar és dir-li a l'ordinador què ha de fer mitjançant un llistat d'instruccions que anomenem programes.

Nivells en programació

Les instruccions que segueix l'operari de la fàbrica (el processador) no són com les nostres.

L'operari no entén "recull la caixa i deixa-la a la taula". L'operari de l'exemple entén instruccions més senzilles com ara:

? Dóna una passa endavant

? Fira 90 graus a la dreta

? Hi ha algun objecte davant?

I encadenades com:

1. Dóna una passa endavant

2. Hi ha algun objecte davant?

3. Si hi ha un objecte recull l'bjecte

4. Si no hi ha un objecte torna al pas 1

En això consisteix la feina del programador. En crear programes, o sia llista d'ordres, per a que l'ordinador (l'operari) faci el que volem.

Per això els programadors han de tenir en compte tots els possibles problemes.

Per a programar utilitzem llenguatges de programació.

Els llenguatges de programació són similars als llenguatges que parlem els humans en diferents països: n'hi ha de diferents amb diferents estructures gramaticals.

Depenent de si el llenguatge que emprem és més similar al que entén l'ordinador (passa, passa, hi ha alguna cosa?, passa, agafa, gira 180 graus, passa, passa, hi ha la taula? desa a la taula...) o més similar al que entenem els humans (porta la caixa a la taula) parlem de llenguatges de baix nivell, nivell mig o alt nivell.

Els llenguatges de baix nivell són els que se semblen molt al que entén la màquina.

Els d'alt nivell són els que se semblen molt al que parlem els humans.

En principi en un llenguatge d'alt nivell no sabem ben bé què fa la màquina per sota, mentre que en un llenguatge de baix nivell controlem tots el que fan els seus components.

Un exemples més real de programació en codi màquina, que és el de més baix nivell seria:

```
ETIQUETA_INICI:Array push axArray push bxArray push dxArray xo
r ax,axArray mov dx,TIMER1_CNTArray in al,dxArray or al,alArra
y jnz ETIQUETA_ES0Array xor ax,axArray out dx,axArray mov dx,T
IMER0_CNTArrayETIQUETA_ES0:Array mov bx,axArray mov ax,_dt_ymp
prArray sub ax,bx
```

L'objectiu d'aquest codi és veure que no enteneu res.

Si enteneu aquest codi deixeu de llegir que vindré jo a que em doneu classes si és que la Nasa ?no us contracta i us deixa una estoneta lliure.

El més interessant és la línia que diu jnz ETIQUETA_ES0. jnz és una de les instruccions bàsiques que mira si la última operació matemàtica ha retornat 0, (jnz significa jump if not 0, és a dir, salta a la etiqueta si no és 0).

Respireu, sé que és complicat. L'objectiu no és comprendre aquest codi si no veure la diferència entre baix nivell i alt nivell.

En un llenguatge de baix nivell tenim cura i treballem amb totes les peces internes de l'ordinador.

Per exemple, unes petites memòries (ax, dx, cx, ex) que serveixen per a fer les operacions bàsiques com **mov ax, dx** que vol diu mou el valor que hi ha a la memòria dx a la memòria ax.

Com aquestes memòries tenen molt poca capacitat, 8 bytes en el cas de ax, cx, el que fa que només puguem tenir valors entre 0 i 255, hem de vigilar quan fem sumes, que no ens passem per tal no obtenir resultat erronis.

Programar en ensamblador o en altres llenguatges de baix nivell és complicat, és molt difícil controlar un error, i un programa per a un tipus d'ordinador no servirà per a un altre ordinador o un mòbil.

Per això avui en dia molt pocs programen en baix nivell, i queda reservat per a alguns enginyers que fan controladors (drivers) on la velocitat és crucial o bé no hi ha una altra manera de fer-ho que emprar certs components de la màquina.

La majoria de gent avui en dia programa en llenguatges d'alt nivell, on no t'has de preocupar de quines peces hi ha per sota.

Exemple de llenguatges d'alt nivell són: PHP, Java, microsoft .NET.

Un llenguatge de mig nivell (entre la màquina i els humans) és el C.

El llenguatge de baix nivell és l'ensamblador (el de mov ax, dx) que és el mes semblant al que parla la màquina (0 i 1 en realitat).

Compilar o interpretar

Precisament, perquè l'ordinador només entén 0 i 1, necessitem algun programa que transformi els nostres programes en que es poden llegir i entendre al llenguatge que entén la màquina, que s'anomena codi màquina o codi binari.

L'ordinador només entén uns ? voltatge positiu o zeros.

Al principi els programadors programaven amb uns i zeros, però era òbviament molt difícil de mantenir programes d'aquesta manera.

Trobar errors i fer modificacions era una feina realment complicada.

Per això es van crear els compiladors.

Els compiladors agafen un programa fet en un llenguatge d'alt nivell, mig o baix, i el transformen en codi màquina, que és l'únic que l'ordinador finalment entén.

Però no tots els programes es compilen.

Hi ha uns llenguatges que es diuen interpretats. Això vol dir, que enlloc de compilador, hi ha un intèrpret que en temps real tradueix el llenguatge de programació a codi màquina.

Els llenguatges interpretats mai són tan ràpids com els que es compilen per aquest motiu, perquè l'intèrpret els tradueix sobre la marxa i ha de fer les traduccions del llenguatge a codi màquina.

Diem que interpreta el codi en temps real o on-the-fly (al vol).

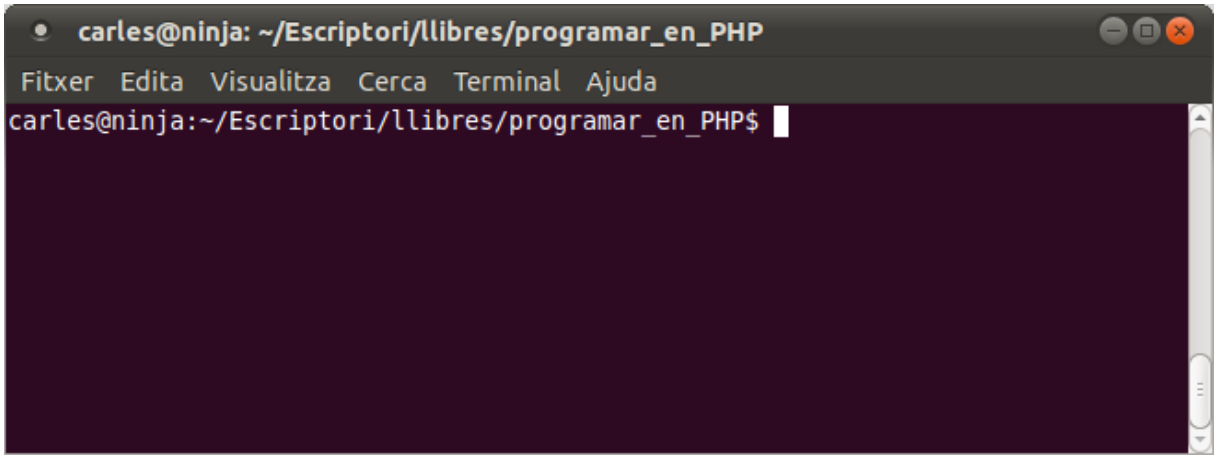
PHP és un llenguatge interpretat. Això fa que certament no sigui tan ràpid com el codi en C compilat, però el fet de ser interpretat ofereix alguns avantatges com el fet que un codi en PHP funciona en un Linux, en un Windows, en un Mac, en un Solaris... simplement és qüestió de crear intèrprets per als ordinadors (o mòbils) que sigui i tots els programes escrits en PHP ja funcionaran.

Per exemple, parem atenció a aquest codi en PHP:

```
<?phpArray echo 'Hola Catalunya!\n';Array?>
```

Aquest codi imprimirà per pantalla el missatge Hola Catalunya en qualsevol ordinador amb PHP instal·lat en que l'executem.

No és genial?. :-)



A la imatge el fitxer hola.php executat en el meu Linux. En un ordinador amb windows produeix la mateixa resposta.

Els llenguatges interpretats també s'anomenen de vegades llenguatges de script (script és un guió en anglès).

Perquè PHP?

Per diversos motius.

PHP és un programari lliure, això vol dir que el podem utilitzar sense pagar res, és molt potent, ve preparat amb moltes llibreries per a fer moltes coses com connectar-se a bases de dades, comprimir en format zip, etc... i està orientat a treballar en web.

Facebook per posar un exemple, utilitza PHP, així com la majoria d'empreses líders d'Internet.

Segons la Wikipedia el 75% dels servidors Web d'Internet utilitzen PHP com a llenguatge de programació.

PHP s'integra amb els principals servidors web.

Per tant, per aprendre a programar en PHP necessitarem instal·lar en el nostre ordinador un servidor web, per exemple Apache, també i PHP.

Podem instal·lar un programa com ara Wamp, que inclou en un Apache, MySql, i PHP.

Per programar ens podem instal·lar un programa com Notepad++ que és gratuït i posa colors a les instruccions PHP. Hi ha altres IDE (entorns de desenvolupament) més potents, però ja ho anirem veient. Poc a poc.

De moment instal·lem Wamp i Notepad++.

Ho normal és que necessiteu la versió de 32 bits:
http://www.wampserver.com/en/dl32.php

Adreça curta Twitter d'aquest article: http://wp.me/pzeab-1It

Traduir a l'Anglès. Translate to English	Compartir: ||