NÚMERO 4 AÑO 0

papirux

La Revista Libre

ISSN 2013-259X

Redes (I)

eyeOS

Algoritmo y Programación

Revolución Tecnológica

Linux From Scratch

en la nube

KIG (II)

Capitán Sevilla

Creative Commons

papirux
La Revista Libre

Índice

Equipo:

Coordinación:
Sergi Caparrós

Colaboradores:
Daniel Quiroga - Juan Carlos - Sergi Caparrós
Pere - Carlos Sánchez - Fausto Mauricio Lagos
Apokalíptica79

Diseño y Maquetación:
Pablo Axeitos

Correctores ortográficos:
Akilex90 - Elmer - Fredy Enrique Lopez -
Joferrue - Jorge Arios - Jorge Marchante -
Solaris - Sergi Caparrós

Editorial

El mundo sigue cambiando minuto a minuto, con nuevas tecnologías; convirtiéndolo en una guerra silenciosa por ver quién es el mejor.

Mientras que algunos se preguntan por qué elegir Linux, surgen nuevos modelos de computadoras, procesadores mas rápidos, y con mayor capacidad. Al mismo tiempo, algún colaborador alrededor del mundo hace un cambio en el código fuente de algún programa, para beneficiar así a toda la comunidad.

El Software Libre, crece, y crece a pasos agigantados; intimidando a los grandes, ofreciendo opciones. Hoy por hoy, el kernel de Linux no es el único en ofrecer una solución libre, debido a la existencia de otros grandes proyectos como OpenSolaris, Hurd, o inclusive los BSD.

Con todas estas opciones, es momento de elegir; y esto es la razón de Papirux, dar información a los usuarios, que nuestro mensaje libre llegue a todos, dotando de conocimiento. Es así como queremos que nuestro mensaje se de a escuchar: Libre.

Disfruten la lectura, Papirux 4.

Félix, el Gatuno123

Revolución Tecnológica en Marcha

Daniel Quiroga

¿Será este el fin de la informática como la conocemos?

Desde que se inició el movimiento del Software Libre hace 25 años (1983), desde que Linus Tordvalds escribió la primera versión del kernel Linux hace 18 años (1991), bases estas sobre las cuales se han construido, no solamente sistemas operativos y aplicaciones libres, si no también filosofías y estilos de vida. Hemos participado en una revolución tecnológica, una guerra sin cuartel en la cual muchas empresas e ideas igualmente buenas han desaparecido, una guerra que no sólo se ha librado en el campo de la programación, sino también en el de la economía, como es el caso de Netscape vs IE; una batalla que se libró en todos los campos: ideológicos, económicos, legales, etc.

Durante los casi cuatro años que llevo metido en el Software Libre, siempre pensé (equivocadamente) al ver una nueva versión que esto había llegado al punto máximo de su perfección, sobre todo ahora que el soporte de hardware está tan adelantado, llegando a estar parejo con "la competencia" en muchos aspectos. Lo cual junto con el bajón que implicó para Microsoft los problemas con MS Windows Vista, ha hecho esta fase de la guerra un poco lenta, y siempre con una gran ventaja hacia nosotros. Sin embargo presumo que esta fase está a punto de acabar por las siguientes razones:

MS Windows 7:

He probado la Beta, y es francamente impresionante, bastante más ligero y con una presentación gráfica algo mejor, con este producto Microsoft podrá entrar de nuevo con fuerza en la guerra (o al menos eso espero) ya que este tipo de carreras fomentan el avance de la tecnología, tal como pasó durante la guerra fría.

Ubuntu 9.04 Jaunty Jackalope:

Se ha hablado mucho, en la blogosfera de esta versión, he estado probando las alphas y pinta muy bien, tiene características nuevas por todas partes, y ni siquiera hablemos de Karmic Koala, que será aún mejor.

Debian 5.0 Lenny:

Así es, el "estado más puro del alma linuxera" llegó a su versión 5.0. Como siempre no hacen el 'release' hasta que el sistema sea perfecto. Su estabilidad y rendimiento están muy por encima de los de cualquier otra versión de GNU/Linux excepto Gentoo.

Gentoo 2008.0:

Como siempre la más ligera de las distribuciones. Con su sistema de paquetes: 'portage' y la invaluable habilidad de poder ser compilado para lograr el mejor rendimiento del sistema.

Como podemos ver, los mejores sistemas operativos, en mi opinión, tienen versiones nuevas y listas para entrar en batalla.

Sin embargo hay más razones para que se anuncie una nueva era en la historia de la computación: por doquier surgen sistemas operativos nuevos, totalmente nuevos, con características nuevas, que hasta ahora no se habían visto. Voy a hablarles un poco de algunos:

ReactOS:

Es una plataforma libre que se desarrolla con la idea de proveer un sistema operativo libre que pueda reemplazar a Windows.

El disco de instalación trae el sistema operativo base en tan solo 35 MB, la instalación es muy parecida a la de Windows 2000/ME/XP aunque desde el primer momento se nota que es algo más. La instalación es bastante rápida. Cuando reinicias descubres que acabas de entrar a un sistema operativo que es muy similar, en su ambiente de trabajo, a Windows 2000. Sin embargo se nota que tiene "algo más". Este sistema no corre, vuela. Permite la instalación de drivers de Windows de manera nativa, y corre excelentes juegos de Windows como Unreal Tournament o Half Life 2.

Más información: http://www.reactos.org

Phantom OS:

Si la carrera del software se puede considerar una guerra, entonces Phantom OS es un arma de destrucción masiva. Este sistema operativo, está en pleno desarrollo, y aún no tiene una versión alpha. Sin embargo promete mucho. Si el programador Dmitry Zavalishin logra llevar a cabo al menos la mitad de lo que planea con su Phantom va a conmover la informática hasta sus bases, aboliendo uno de los paradigmas más profundos de la computación. Hasta ahora cuando la electricidad falla todos los cambios no guardados se pierden y cuando esta regresa hay que volver a encender la computadora teniendo que repetir todo el proceso de arranque. Phantom lo que hace es escribir el estado minuto a minuto al disco. Cuando la energía se pierde el automáticamente re-arranca desde el punto donde se quedó sin que siquiera note que se apagó.

Más información: http://dz.ru/en/solutions/phantom/

EyeOS:

Es un sistema operativo 'cloud computing'. Se aloja en un servidor remoto al cual accede el usuario. En dicho servidor está almacenada la información del usuario, permitiéndole acceder a sus datos desde cualquier lugar.

Más información: http://www.eyeos.org/

Conclusiones:

Como podemos observar, estas diversas opciones el mundo de la informática está teniendo un avance acelerado. Cosas que ayer se consideraban imposibles están siendo cada vez más comunes. Estas nuevas alternativas, características y funcionalidades nos llevan a una gran velocidad hacia lo imposible. Estamos llegando a una época en la cual un procesador de cuatro u ocho núcleos es básico. Tenemos una velocidad básica mucho mayor y con estos rendimientos los sueños más locos se hacen realidad.

No hay duda de que estamos acercándonos a un recrudecimiento en la carrera del avance tecnológico ¿Hacia dónde apunta? Hacia la seguridad y hacia la informática de bolsillo. El viejo PC de escritorio y el computador portátil pronto serán obsoletos y reemplazados por dispositivos del tamaño de una mano con sus mismas características, haciendo que la informática forme parte, aún más, de nuestras vidas. Las alternativas en sistemas operativos se están abriendo muy rápidamente. Ya no hay que decidir entre Windows o GNU/Linux, hay alternativas completamente nuevas que harán las delicias de más de uno.

eyeOS

Apokalíptica79 - Sergi Caparrós

EyeOS es el Sistema Operativo del Cloud Computing. Trabaja online -personal y cooperativamente- con los archivos, Office, calendario, contactos y mucho más. eyeOS es Software Libre.

Para los que no lo conocen es es una herramienta sumamente útil y como bien dice el título es un sistema operativo muy liviano con el cual podrás hacer todos tus trabajos diarios, como por ejemplo usar el procesador de texto, hoja de cálculo, calendarios, y además es personalizable.

A continuación un listado de las aplicaciones que podemos encontrar en eyeOS:

-Oficina: eyeDocs, eyeSheets, eyePresentation, eyeCalendar, eyeContacts, eyePdf.
-Educación: eyePlot, eyeCalc.
-Juegos: eyeTetravex, eyeChess.
-Otras Aplicaciones: eyeFiles, eyeString, eyeArchive, eyeOS Mobile.
-Temas: Default, Light Desktop.
-Accesorios: eyeNotes.
-Administración: eyeProcess, eyeControl, eyeInstaller, eyeSoft.
-Red: eyeFeeds, eyeNav, eyeMail, eyeBoard, eyeUpload, eyeFTP, eyeMessages.
-Multimedia: eyeMp3, eyeVideo.
-MiniAplicaciones(widgets): miniMessages, miniCalendar, minihome.

Ahora vamos a proceder a la instalación:

Requerimientos:

- Server:

El requerimiento principal para una instalación nueva de eyeOS es un servidor web compatible con PHP 5 o mayor.

Se recomienda que se use un servidor con Apache Web Server y PHP 5, pero otros servidores web con soporte para PHP están disponibles para poder trabajar con eyeOS instalado.

EyeOS tiene su propio sistema de archivo virtual y no requiere de una base de datos para trabajar. De todos modos necesitará tener capacidad de subir archivos y directorios y que estén disponibles para poder cambiarles los permisos.

- Cliente (Web Browser):
- Mozilla Firefox 2 y mayor.
- Internet Explorer 6 y mayor.
- Safari 3 y mayor.

Para instalar en nuestro servidor local

1) Descargamos eyeOS:
http://eyeos.org/es/downloads

2) Una vez que la descarga se ha completado vamos a la carpeta que lo contiene y lo descomprimimos. En mi caso descargué el .zip.

unzip eyeOS1.7.0.1-2.zip

3) Al extraer correctamente aparecerá una carpeta con el siguiente nombre:
eyeOS

4) Ahora en la barra de navegación colocamos:
http://localhost/eyeOS/index.php
Y listo. Es así de sencillo.

Para instalar en un servidor contratado (el típico hosting externo)

Procedemos a la descargar del fichero "Tar.gz | Instalador y Actualizador | 2.2 MB" del sitio http://eyeos.org/es/downloads. En el momento de escribir este artículo la versión disponible es la 1.8.0.3.
Extraemos el contenido del fichero a nuestro disco duro y éste lo subimos a su vez (descomprimido) a nuestro servidor externo mediante FTP.

Una vez subido a nuestro servidor, visitamos la URL del lugar donde lo hayamos subido (www.midominio.xxx). Nos aparecerá la ventana de Instalación de eyeOS. Nos solicita la contraseña del usuario root, el nombre del sistema (algo así como el nombre de la empresa), y una última opción que permite que los usuarios puedan darse de alta ellos mismos y crear un usuario para acceder al sistema (sistema abierto) o por el contrario tenga que ser un administrador quien dé alta a los usuarios en el sistema (sistema cerrado). Pulsamos sobre el botón "!Instalalar eyeOS! y en unos segundos estará instalado nuestro sistema eyeOS.

La primera pantalla que vemos ahora es la de bienvenida, para validarnos en el sistema. Deberemos entrar con el usuario "root" y la contraseña que hayamos elegido. Fijaros que en la ventana de validación (login) tenemos la opción de crear una cuenta nueva (si hemos dejado el sistema abierto a nuevos usuarios). De momento sólo disponemos del idioma en inglés.

Para cambiar el idioma, valídate como root. En la parte inferior de tu escritorio eyeOS verás una barra de color verde. Al pulsar sobre esta barra se despliega un menú y deberás elegir la opción "System Preferences". En el menú de la izquierda elige "Language" y ahora pulsa sobre "Get more translations".

Se abrirá el navegador de eyeOS y nos llevará a la URL:
http://eyeos.org/?section=downloads&part=translations&lang=en donde aparecerá un listado de todos los idiomas disponibles, entre ellos "Spanish (Español)". Lo guardamos en nuestro disco duro (nuestro PC de verdad).

Ahora podemos cerrar las ventanas abiertas en nuestro sistema eyeOS. A la derecha de nuestro escritorio virtual eyeOS tenemos unas opciones. Una de ellas dice "Upload your files". Elegimos el método simple (a mí personalmente me gusta más) y subimos el fichero de idioma al escritorio virtual eyeOS.

Una vez tenemos el fichero de idioma en nuestro escritorio virtual pulsamos sobre él y nos aparecerá un mensaje emergente preguntado si realmente queremos instalar el paquete en el sistema (el sistema eyeOS, claro está). Aceptamos la instalación y se instalará sin problema.

Ahora podemos ir de nuevo a "System Preferences" › "Language" y seleccionar el idioma "Español (Spanish)". También podemos hacer que sea el idioma predeterminado de los usuarios. Para ellos, en la esquina infereior derecha de la ventana "System Preferences" seleccionamos las opciones de "[Default User]" › "Language" y de nuevo elegimos "Español (Spanish)" como idioma de los usuarios.

Si cerramos la sesión veremos que la pantalla de bienvenida aparece ahora en español como idioma predeterminado. Además nos deja seleccionar el inglés o cualquier otro idioma que hayamos instalado.

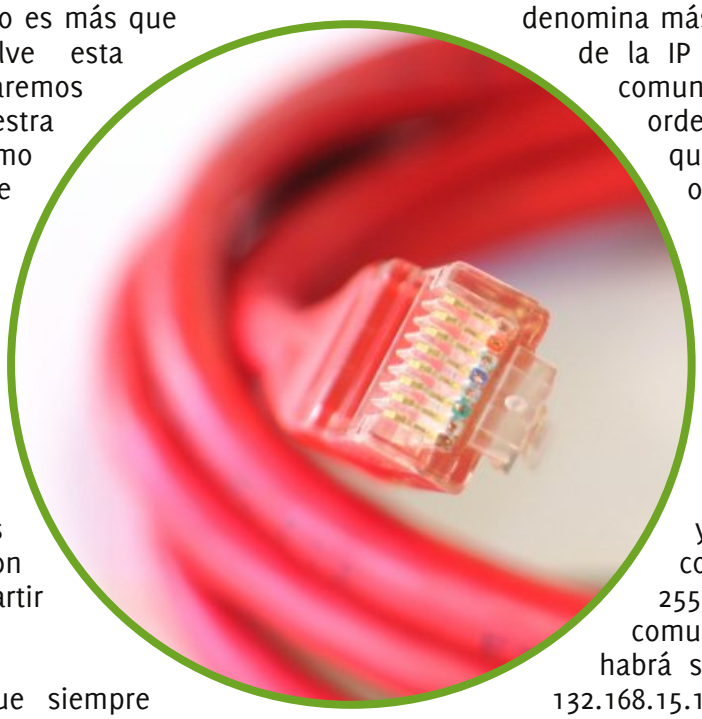
Para obtener más información podemos visitar http://eyeos.org/es
Si quieres obtener aplicaciones para tu nuevo Sistema Operativo Virtual visita:
http://www.eyeos-apps.org/

Redes en Linux (I)

Juan Carlos

Con este artículo, quiero empezar a dar unas nociones sobre redes y como aplicarlas a nuestro sistema Linux, en este caso me centraré en Ubuntu, que es el que se supone tenemos la mayoría de los que leemos esta revista, aunque en el resto de Linux es prácticamente igual.

Para entender cómo funcionan las redes se necesita conocer los fundamentos de las direcciones TCP/IP. En internet encontraréis muchísima información al respecto sobre los protocolos mencionados. Supongo que también sabéis que los ordenadores se comunican entre sí por estos protocolos aunque nosotros usemos nombres por la fácil compresión de estos respecto a una dirección IP, en realidad los ordenadores no se comunican con un nombre sino por una IP (cuando nosotros ponemos en nuestro navegador "www.google.es", en realidad lo que estamos haciendo es decirle al ordenador que nos comunique con otro ordenador cuya IP es 209.85.229.103 ; claro está que para que este proceso se realice, el ser humano ha ideado una forma de traducir los nombres en direcciones IP, que es lo que se llama DNS, que no es más que un servidor que resuelve esta situación y del que ya hablaremos e instalaremos uno en nuestra máquina para ver cómo funciona, pues este tipo de servidores los podemos poner en nuestra red interna y funcionar por medio de nombres y no con las IP).



Una dirección IP no es más que un conjunto de números separados por puntos gracias a los cuales podemos comunicarnos con otro ordenador, compartir carpetas, impresoras etc...

Tened en cuenta que siempre que estemos haciendo prácticas de este tipo debemos trabajar como root en el sistema, porque sino no os dejará hacer algunas configuraciones, ya que el tema de redes es cosa de administradores.

Como mejor se ven las cosas es con un ejemplo ilustrativo, que podéis hacer ahora en vuestro ordenador si tenéis acceso a internet. Para ello nos vamos a la Consola de nuestro GNU/Linux, allí ponemos lo siguiente "nslookup www.google.es" nos dará como resultado unas IP, poned una cualquiera en vuestro navegador (en la barra de direcciones), es decir en lugar de poner el nombre, poned la IP y os conducirá a la página de Google. El comando "nslookup" como veis nos da información sobre resolución de nombres, es bastante práctico. A lo largo de los diferentes artículos iremos viendo algunos comandos interesantes.

Respecto a las IP tienen un formato parecido a esta 192.168.15.1, pueden ir del 0 al 255 como máximo, es decir el mínimo que podríamos utilizar sería 0.0.0.0 y el máximo 255.255.255.255, claro está que, como todo, hay una serie de normas para usarlos, sobre todo a la hora de salir a internet no sirve cualquier número pues esto está regulado formalmente por las Autoridades y tienen lo que se llama IP públicas. Sin embargo en una pequeña red que montemos en nuestra casa o en una red de una empresa podemos jugar con estas IP como queramos.

A toda IP va asociada otra IP, lo que se denomina máscara de red, y dependiendo de la IP y de la máscara podemos comunicarnos con otros ordenadores o no; es decir, si queremos que dos ordenadores o más se comuniquen entre sí debemos seguir una serie de reglas, porque sino, no habrá comunicación entre esos ordenadores, por ejemplo: si tenemos el ordenador 1 con una IP 192.168.15.1 con una máscara de red 255.255.0.0 y tenemos otro ordenador 2 con IP 132.168.15.1 y máscara 255.255.0.0, no habrá comunicación entre ellos. Sí la habrá si ponemos en ordenador 1 132.168.15.1 máscara 255.255.0.0 y en el ordenador 2 132.168.15.2 máscara 255.255.0.0.

Por supuesto dos ordenadores que estén dentro de la misma red no pueden usar la misma IP, porque tampoco se comunicarían, habría un conflicto en la red. Todo esto lo digo para aquel que no tiene conocimientos de redes y quiera conocer el tema en profundidad sólo se tiene que ir a un buscador de Internet y tendrá toda la información.

Para las prácticas vamos a utilizar dos ordenadores en red o bien dos máquinas virtuales (claro está que pueden ser más) a cada uno le vamos a poner una IP para que se puedan comunicar entre ellos y podamos ver los archivos y carpetas correspondientes. Un ordenador hará de servidor (es decir el que presta los servicios) y otro el cliente (es decir el que quiere coger o ve esos recursos). El cliente prácticamente no hay que configurarlo (salvo ponerle la IP), en el servidor habrá que configurar algunos ficheros.

En primer lugar debemos saber qué tarjetas de red tenemos, que IP le vamos a poner y si los ordenadores se comunican.

Cuando veáis un símbolo como este "#" quiere decir que estamos en la consola como root y lo que ponga después significará el comando que voy a utilizar, después de cada comando pulsaremos "ENTER".

Para saber la configuración de mi tarjeta y la IP, en la consola usamos lo siguiente:

```
#ifconfig
```

Nos saldrá la IP de nuestra máquina junto con la máscara de red, así como el nombre de la tarjeta, ejemplo:

```
eth1 Link encap:Ethernet direcciónHW
00:1e:90:29:6f:75
inet dirección:192.168.15.2
Difusión:192.168.15.255
Máscara:255.255.255.0
```

eth1: este es el nombre de mi tarjeta de red.
HW: es es la dirección MAC.
Dirección: 192.168.15.2: esta es la IP
Máscara: 255.255.255.0: esta es la máscara.

La dirección MAC es una dirección física, dicho en otras palabras es el número de la tarjeta física de red. Esta dirección física es siempre la misma y cada tarjeta tiene una diferente. Se puede cambiar, pero lo dejo para más adelante, porque ahora no nos afecta.

Si queremos saber los nombres de nuestras tarjetas de red, podemos verlo en el siguiente archivo:

```
#cat /etc/udev/rules.d/70-persistent-
net.rules
```

Si queremos ponerle otro nombre, no tenemos más que editar el fichero y guardar los cambios, pero tendréis que reiniciar la máquina para que sean efectivos dichos cambios. Así en lugar de salirnos "eth1" como salía antes, puede salirnos otro nombre, el que le hayamos puesto, eso es a gusto vuestro.

Ejemplo de este fichero:

```
# This file maintains persistent names
for network interfaces.
# See udev(7) for syntax.
#
# Entries are automatically added by the
75-persistent-net-generator.rules
# file; however you are also free to add
your own entries.

# PCI device 0x10b7:0x9000 (3c59x)
SUBSYSTEM=="net", ACTION=="add",
DRIVERS=="?*",
ATTR{address}=="00:10:4b:aa:70:03",
ATTR{type}=="1",
KERNEL=="eth*", NAME="eth0"

# PCI device 0x10ec:0x8136 (r8169)
SUBSYSTEM=="net", ACTION=="add",
DRIVERS=="?*",
ATTR{address}=="00:1e:90:29:6f:75",
ATTR{type}=="1",
KERNEL=="eth*", NAME="eth1"
```

Como podéis observar tengo dos tarjetas "etho" y "eth1", si queréis cambiarlo no tenéis más que sustituir ese nombre por el que queráis y ya estará listo.

Cambiar la IP de mi tarjeta: podemos hacerlo de dos formas:

```
#ifconfig eth1 192.168.15.1/24 up
```

O También esta:

```
#ifconfig eth1 192.168.15.1 netmask
255.255.255.0 up
```

Las dos líneas que he puesto son exactamente iguales.

También podemos ver las tarjetas en el siguiente fichero:

```
#cat /etc/network/interfaces
```

Saldrá algo como esto:

```
auto lo
iface lo inet loopback
```

Esta línea siempre debe de estar en este fichero.

```
iface eth1 inet dhcp
aitp eth1
```

Esta línea puede variar. En este caso está configurada para un servidor dhcp y es la tarjeta "eth1". Si le hubiésemos cambiado el nombre como indique más arriba aparecería aquí el nombre correspondiente.

Estas líneas también pueden ser cambiadas. Si por ejemplo vamos a usar una IP estática en lugar de hacerlo desde la consola lo podemos insertar aquí y no tendremos problemas de configuración como puede ocurrir si hacemos los cambios desde consola. Por ejemplo podemos poner lo siguiente:

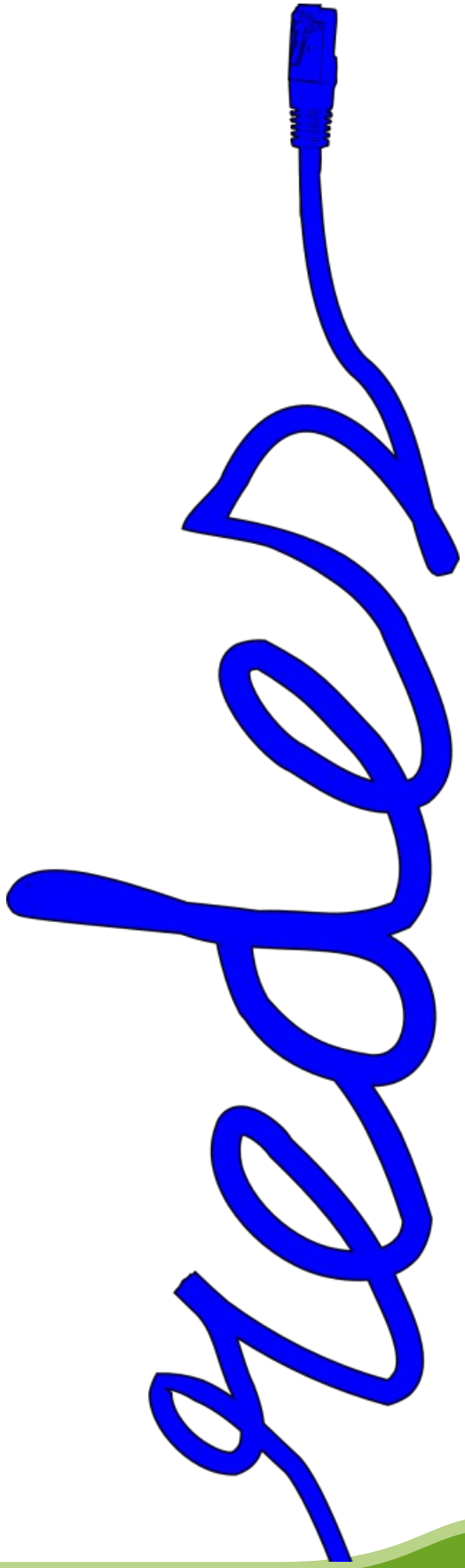
```
auto eth1
iface eth1 inet static
address 192.168.15.1
(esta es la IP de la tarjeta)
netmask 255.255.255.0
(esta la máscara de red)
gateway 192.168.1.1
(esta sería una puerta de enlace y lo pondríamos en caso de
utilizar un router, pero no es necesario si no vamos a
enrutar).
```

Para comprobar si dos máquinas entre sí tienen conexión se suele utilizar el comando ping. Por ejemplo si ahora tenéis conexión a Internet, desde la misma consola podéis poner lo siguiente: ping www.google.es, saldrá algo parecido a esto:

```
# ping www.google.es
64 bytes from ww-in-f104.google.com
(209.85.229.104): icmp_seq=1 ttl=238
time=91.6 ms
64 bytes from ww-in-f104.google.com
(209.85.229.104): icmp_seq=2 ttl=238
time=101 ms
```

Esto quiere decir que tenemos comunicación con dicha máquina.

Para los más novatos en el tema recomendaría que pusieseis dos IP con sus respectivas máscaras (una a cada máquina) a dos maquinas que tengáis conectadas en red y comprobéis si hacen ping entre ellas.



Creative Commons

Sergi Caparrós - scaparros@papirux.org

Creative Commons [1] es un tipo de licencia que nos permite beneficiarnos a todos. Podemos sacar provecho de trabajos realizados por otros, siempre bajo las condiciones de Creative Commons. Fue fundada por Lawrence Lessing quien la presidió hasta marzo de 2008.

La licencia Creative Commons están inspiradas en la licencia GPL (General Public License) de la FSF (Free Software Foundation). Podríamos traducir Creative Commons como "bienes comunes creativos" y en vez de estar enfocado a licenciar el Software, está enfocado a licenciar textos, música u otros tipos de creaciones intelectuales.

Las licencias restrictivas como el Copyright, lejos de ayudar a los autores a proteger sus obras, hacían que les copiaran las obras sin ofrecer nada a cambio, ni una mención, ni una obra derivada con su nombre, nada. Al licenciar tu obra bajo Creative Commons estás asegurándote que tu obra llegue a muchas más personas a través de obras derivadas de la tuya. Consigues que puedan utilizar tu obra, siempre que reconozcan tu autoría y eso conlleva llegar a millones de personas y ganar reputación, entre otros beneficios.

También puedes beneficiarte tú mismo al hacer un trabajo y utilizar material bajo licencia CC. Te por el ahorro de tiempo y beneficias al autor original de la obra al hacerla llegar a una multitud más amplia de personas.

Tenemos varias series de licencias Creative Commons para licenciar nuestra obra bajo aquella que más nos convenga o interese. El autor es el encargado de elegir el tipo de licencia más conveniente a su obra. Aquellos que quieran que terceras personas utilicen su obra bajo determinadas condiciones lo pueden hacer. Creative Commons propone tener "algunos derechos reservados" y no "todos los derechos reservados" como dice el Copyright.

Hay un total de seis licencias Creative Commons para escoger:

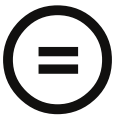
Reconocimiento (Attribution):

En cualquier explotación de la obra autorizada por la licencia hará falta reconocer la autoría.

No Comercial (Non commercial):

La explotación de la obra queda limitada a usos no comerciales.

Sin obras derivadas (No Derivate Works):



La autorización para explotar la obra no incluye la transformación para crear una obra derivada.

Compartir Igual (Share alike):

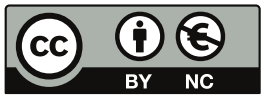
La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

Con estas cuatro condiciones combinadas se pueden generar las seis licencias que se pueden escoger:

Reconocimiento (by):

Se permite cualquier explotación de la obra, incluyendo una finalidad comercial, así como la creación de obras derivadas, la distribución de las cuales también está permitida sin ninguna restricción.

Reconocimiento - No Comercial (by-nc):



Se permite la generación de obras derivadas siempre que no se haga un uso comercial. Tampoco se puede utilizar la obra original con finalidades comerciales.

Reconocimiento - No Comercial - Compartir Igual (by-nc-sa):

No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Reconocimiento - No Comercial - Sin Obra Derivada (by-nc-nd):

No se permite un uso comercial de la obra original ni la generación de obras derivadas.

Reconocimiento - Compartir Igual (by-sa):

Se permite el uso comercial de la obra y de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Reconocimiento - Sin Obra Derivada (by-nd):

Se permite el uso comercial de la obra pero no la generación de obras derivadas.

[1] Creative Commons España: http://es.creativecommons.org/

Prisión o Libertad

Pere , miembro de FACIL (http://www.facil.cat)

A menudo, cuando alguien oye hablar de Linux, de Software Libre y código abierto le pica el gusanillo de la curiosidad y entonces indaga, pregunta, busca y acaba encontrando a otro que lleva ya un tiempo en este mundo.

Las dudas acostumbran a ser abundantes, las preguntas, como es lógico, no están muy estructuradas y muy a menudo nosotros pretendemos poner al día a nuestro interlocutor con una charla de media hora.

Hablamos de GPL, de las diferentes distribuciones de nuestro sistema preferido, de entornos gráficos, de gestores de ventanas, de que no debemos decir Linux, si no GNU/Linux,... en definitiva, le inflamos la cabeza hasta unos límites que pueden facilitar el colapso neuronal. Que pesados somos algunos!...

Parece que hayamos olvidado como comenzamos nosotros, lo vemos como si fuera la prehistoria, cuando en la mayoría de los casos estamos hablando, como mucho, de tres o cuatro años. Los inicios son duros, pero cuesta acordarse, sabemos que merece la pena el esfuerzo y por eso nos duele que nuestro interlocutor no pueda disfrutar de todo lo que nosotros disfrutamos, nos duele que se tenga que preocupar por su ordenador, en lugar de utilizarlo, nos duele que se sienta preocupado porque no entiende nada de las licencias de los programas que utiliza, nos duele que se sienta estafado cuando ha comprado un programa que le ha costado un pastón y el ordenador le dice que no puede utilizarlo, simplemente porque ha tenido una avería y le han cambiando unas piezas de la máquina.

Entonces, con la mejor de las intenciones, atacamos, molestamos y mareamos al personal. Es una reacción visceral, no lo podemos evitar. Pero el peligro de todo es que nuestro interlocutor acabe pasando del tema y se resigne a sufrir como hasta ahora y renovando su sistema una vez más cuando desde Redmond le dicen que el nuevo sistema será mucho mejor, con menos problemas, más bonito y elegante que el que tiene su máquina. Entonces pagará otra vez, la impresora dejará de funcionar por arte de magia, su ordenador será mucho más lento que antes y tendrá que hacer un cursillo para aprender un poco lo que ha cambiado.

Pero en el fondo la cuestión no es tan complicada como parece, y a la vez mucho más difícil de lo que podemos pensar. La elección de la que estamos hablando es, simplemente, escoger entre la libertad o la prisión. En la prisión la comida es gratuita, el orden está garantizado, la cama no se paga y si, encima, nuestras preferencias sexuales coinciden con las del sexo de los compañeros, entonces puede ser el paraíso.

Por algún motivo que no entiendo, la mayoría preferimos la libertad, a pesar que haya que luchar para vivir, a pesar de los riesgos, a pesar de la violencia exterior, a pesar de las tempestades, el precio de las cosas, las manipulaciones, los chantajes, a pesar del miedo y de la inseguridad, a pesar de todo, la libertad no nos parece demasiado cara, nunca demasiado difícil, nunca demasiado incómoda.

Simplemente queremos que nuestra máquina haga lo que queremos que haga. Queremos saber como lo hace, queremos pasar a los amigos ese programa que tanto nos gusta y tan útil encontramos, sin que esto pueda ser considerado un delito. Como le pasamos la receta de pato con peras, o le dejamos el último libro que nos ha gustado. Queremos no tener que pagar para probar si un determinado programa nos es útil o no, queremos poder añadir a ese programa una prestación que nos parece obvia pero que no viene de serie. No queremos estar escogiendo entre pagar o robar, preferimos regalar y coger aquello que nos hace falta y, si nos es útil, no nos será tan difícil compensar con algún euro a quien ha estado trabajando para hacernos la vida más fácil.

La alternativa a nuestra alternativa es oscura. Imaginad por un momento que no podemos saber que la fuerza de atracción de la gravedad es directamente proporcional a la masa e inversamente proporcional al cuadrado de la distancia, sin pagar unas patentes, y que si enseñamos esto a nuestra vecina, el hecho podría ser considerado delito. Este es el mundo que nos espera si dejamos morir el Software Libre.

Un programa es una herramienta, un programa es una idea, lo podemos regalar sin tener que prescindir. No es como un coche o la "tele", o el riñón izquierdo, no es material, es conocimiento y nadie, repito, nadie tiene derecho del monopolio del conocimiento.

Esta es nuestra lucha, no importa mucho si nuestro programa es un poco más avanzado o no, es libre y puede crecer. Dejemos que crezca, que se desarrolle, que nadie frene su camino, al final será mejor, seguro! Y como mínimo, además, es nuestro.

Linux From Scratch

Carlos Sánchez - cargabsj175@gmail.com

GNU/Linux desde el principio...

Linux from scratch[1], **Linux desde cero** o simplemente **LFS**, es un libro escrito originalmente por Gerard Beekmans[2], es una distribución muy peculiar que la diferencia de las demás distribuciones, ya que ésta no posee paquetes binarios o scripts para su instalación, sólo se dispone de manuales de instrucciones y el código fuente para la elaboración de un GNU/Linux a la medida. El libro se distribuye de manera libre y está disponible desde su Web Oficial[3] por si quieres embarcarte en este colosal proyecto :).

La construcción de un sistema LFS requiere de esfuerzos y gran empeño y es muy útil para todo aquel que desee aprender el funcionamiento de un sistema GNU/Linux.

Para mantener enfocado a **LFS** en el objetivo principal (La construcción de un sistema base GNU/Linux) y sin perder el espíritu del original ha nacido otro proyecto con el nombre de **Beyond Linux From Scratch**[4] (**Más allá de Linux desde cero**) o **BLFS**, que agrega instrucciones para construir más que sólo el sistema base, por ejemplo, construir un servidor gráfico **X.org**[5], entornos de escritorio como GNOME[6] o KDE[7], soporte de impresoras y escáneres, redes, etc.

Con el correr del tiempo han agregado otros proyectos inspirados en original que son los siguientes (con sus nombres y una breve descripción):

1. ALFS (Automated Linux From Scratch)[8]: Es un proyecto que crea una plataforma genérica para un sistema extensible; constructor e instalador de paquetes de manera automatizada.
2. CLFS (Cross Linux From Scratch)[9]: Al igual que LFS, contiene las instrucciones para la creación de un sistema desde cero, con la diferencia de que se usa compilación cruzada para soportar diferentes plataformas como AMD, Sparc v9, x86_64.
3. HLFS (Hardened Linux From Scratch)[10]: Enfocado en la seguridad principalmente, proporcionando las herramientas para el "endurecimiento" del sistema GNU/Linux.
4. Hints (Las recetas)[11]: Son pequeños documentos creados por la comunidad que explican cómo construir paquetes que no se mencionan en los libros anteriores.
5. LFS LiveCD[12]: Es el disco que contiene un sistema anfitrión con todas las características necesarias para comenzar a construir un sistema LFS, así como el libro y el código fuente necesarios.
6. LFS Patches Project[13]: Es un repositorio de parches mantenido por la comunidad para los paquetes contenidos en el libro LFS.

Aunque Linux From Scratch y sus derivados están todos escritos en el idioma inglés, existe un proyecto de traducción al español disponible desde http://www.escomposlinux.org/lfs-es/. Actualmente algo abandonado.

En conclusión, los requisitos esenciales para comenzar a armarte tu propio GNU/Linux son los siguientes:

1. ¡Paciencia y muchas Ganas!
2. Un sistema anfitrión que cumpla con los requisitos necesarios (**SE RECOMIENDA ALTAMENTE EL LIVECD**). Aunque está un poco viejo, cumple con las características ideales de un sistema anfitrión.
3. Los Libros (LFS, BLFS, etc.).
4. El código fuente.
5. Y ¡A compilar se ha dicho!

Enlaces en éste artículo:

1. http://www.linuxfromscratch.org/
2. http://en.wikipedia.org/wiki/Gerard_Beekmans
3. http://www.linuxfromscratch.org/lfs/download.html
4. http://www.linuxfromscratch.org/blfs
5. http://www.x.org/wiki
6. http://www.gnome.org/
7. http://www.kde.org/
8. http://www.linuxfromscratch.org/alfs/
9. http://cross-lfs.org/
10. http://www.linuxfromscratch.org/hlfs/
11. http://www.linuxfromscratch.org/hints/
12. http://www.linuxfromscratch.org/livecd/
13. http://www.linuxfromscratch.org/patches/

Demostración del Teorema de Pitágoras con Kig

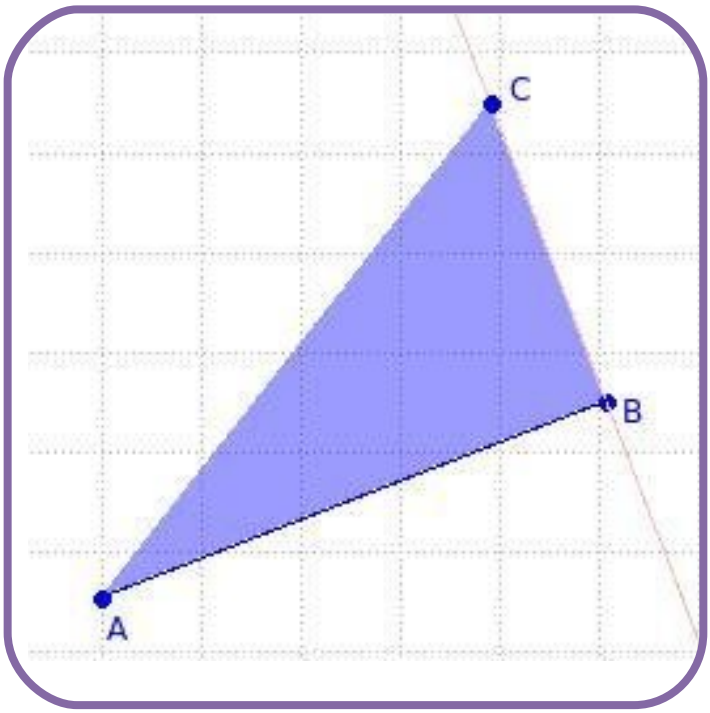
Lic. Fausto Mauricio Lagos Suárez - pirataxoo7@hotmail.com

En esta primera entrega de los mini tutoriales del procesador geométrico KIG, recreamos la demostración del Teorema de Pitágoras, construcción con el cual aprendemos a utilizar la herramienta de traslación y la asignación de atributos.

EL TRIÁNGULO RECTÁNGULO

- Crear un segmento y una recta perpendicular a él por uno de sus extremos.

- Sobre la perpendicular, con la herramienta ‹Point›se crea un punto para completar los tres vértices del triángulo.

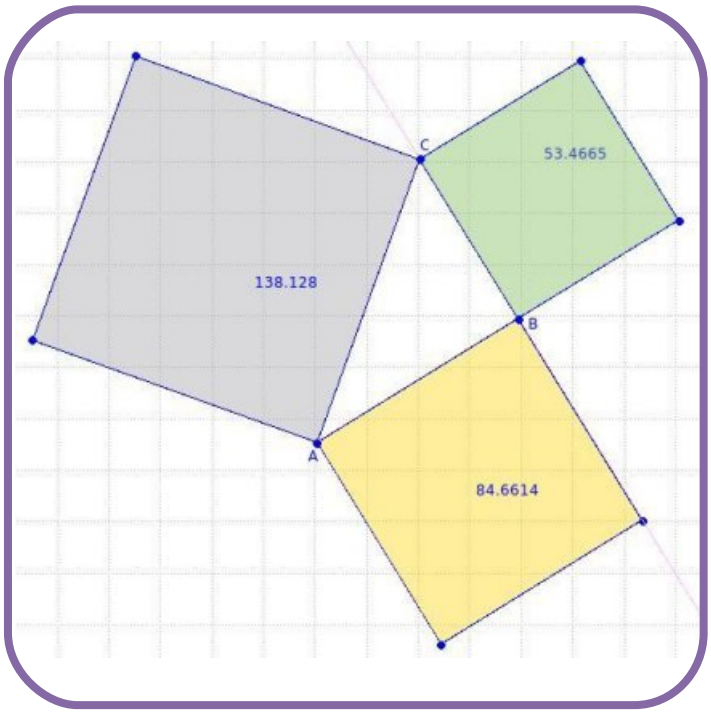


- Con el menú contextual sobre cada punto puede utilizar la herramienta ‹Set name› para nombrar cada vértice del triángulo.

- Utilizando la herramienta ‹Segment› crear el triángulo uniendo los tres puntos A, B y C.

- El siguiente paso es construir los cuadrados sobre cada lado del triángulo con la herramienta ‹Square› dando click sobre cada par de vértices del triángulo. Una forma de acceder a la herramienta ‹Square› es a través del menú Objects, submenú Poligons, o buscándola en el menú contextual sobre el área de trabajo.

- En el menú contextual sobre cada cuadrado se pueden encontrar la opción ‹Add Text Label› desde donde se pueden utilizar la opción ‹Surface› para encontrar el área de cada cuadrado. Otras propiedades importantes a las que se puede acceder desde el menú contextual sobre cada cuadrado son la opción ‹Construct› se pueden resaltar los vértices de cada cuadrado ‹Vertices of a polygon›, y los lados de cada cuadrado con la opción ‹Sides of a polygon›, estas opciones es bueno aplicarlas porque se utilizarán más adelante en la construcción. Es recomendable que después de resaltar los vértices y los lagos de cada cuadrado se oculten las superficies de cada cuadrado, para ello con el menú contextual sobre cada superficie se utiliza la herramienta ‹Hide›.



- Sobre cada cuadrado, en el menú contextual, opción ‹Construct› se encuentra la opción ‹Center of Mass of the Vertices›con la cual se encuentran el centro de cada uno de los cuadrados.

- Lo primero es dividir el cuadrado del cateto mayor, en trapecios, para ello con la herramienta ‹Parallel› se traza una recta paralela a la hipotenusa que pase por el centro del cuadrado del cateto mayor, igualmente se traza una perpendicular a la hipotenusa que pase por el centro del cateto mayor.

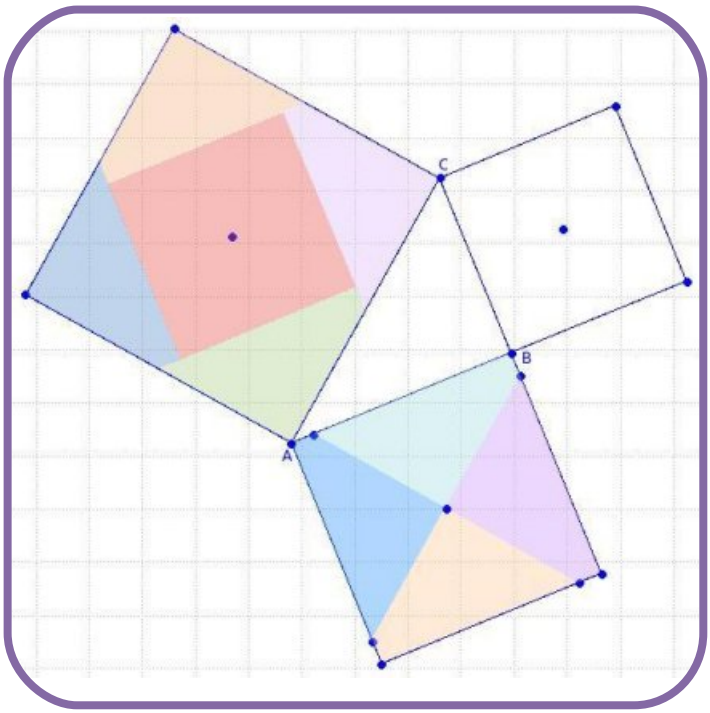
- Encontrar lo puntos de intersección entre la paralela y el cuadrado y entre la perpendicular y el

Demostración del Teorema de Pitágoras con Kig

cuadrado. Para ello es necesario tener ya visualizados los lados del cuadrado como se indicó anteriormente.

- Para construir los trapecios con la herramienta ‹Polygon by its vertices›se debe seguir en orden, primero el centro del cuadrado, luego el punto cercano al vértice, después el vértice, a continuación el punto lejano a dicho vértice y para terminar nuevamente el centro del cuadrado. Si se desea cambiar las propiedades de algún objeto se puede utilizar el menú contextual sobre el objeto, por ejemplo si se quiere cambiar el color de la superficie de los trapecios, en el menú contextual sobre cada trapecio se utiliza la opción ‹Set Color›.

¡¡¡KIG cuenta con una muy amplia gama de colores y variadas paletas que se pueden seleccionar en la ventana de personalización del color ‹Custom color›!!!



- El siguiente paso es utilizar la herramienta ‹Vector›, la cual puede encontrarse en el menú Objects opción Vectors and Segments o en la opción Start del menú contextual sobre el área de trabajo, con esta herramienta se crean los vectores de translación desde el centro de los cuadrados de los catetos hasta el centro del cuadrado del cateto de la hipotenusa.

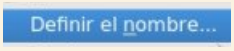
¡¡¡Es necesario utilizar ‹Unhide all› en el menú contextual sobre el área de trabajo para visualizar el cuadrado del cateto menor, que luego se transladarlos ya se puede volver a ocultar junto con los demás cuadrados.!!!

- Después de crear los vectores de translación utilizando la herramienta ‹Translate›, que se puede encontrar en el primer botón de la barra de herramientas derecha, o en Objects – Transformations, se selecciona primero el cuadrado del cateto menor y después su respectivo vector de translación, luego cada uno de los trapecios su respectivo vector de translación hasta mover las cinco piezas al cuadrado sobre la hipotenusa.

HERRAMIENTAS UTILIZADAS

Punto de Intersección

Nombrar



Recta Paralela

Cuadrado

Recta Perpendicular

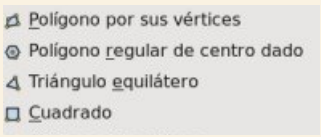
Vector

Punto sobre Objeto

Desplazar

Segmento

Menú Construcción



- ENHORABUENA!!! Ahora ya pude arrastrar los vértices del triángulo rectángulo para comprobar que siempre el área del cuadrado sobre la hipotenusa queda cubierta por las áreas de los cuadrados sobre cada uno de los catetos.

Aquí finaliza el primer mini-tutorial del procesador geométrico KIG, espero que se haya hecho evidente el gran potencial de esta herramienta y quedando a la espera de sus opiniones, nos veremos en la próxima edición con la segunda parte, gracias.

Algoritmo y Programación

APOKALYPTICA79

CAPÍTULO 1

Algoritmo: método para resolver problemas que consiste en dividir el problema en un número finito de pasos elementales e indicar claramente el orden de ejecución de los mismos.

Programación: es la transformación del algoritmo en algo comprensible por la computadora, para ello debe ser escrito en el lenguaje de programación de acuerdo con las reglas de sintaxis del mismo. Consiste en transformar los pasos elementales del algoritmo en sentencias o instrucciones de un programa de computadora.

1.1 RESOLUCIÓN DE PROBLEMAS CON COMPUTADORA:

Las computadoras son capaces del más exacto procesamiento de datos a la más alta velocidad, tienen la habilidad de comparar datos y ejecutar diferentes operaciones de acuerdo a los resultados de la comparación. Pero tienen las siguientes limitaciones:

- Confianza del programa: una computadora hace solamente lo que está programada para hacer.

- Claridad en la lógica: puede procesar únicamente aplicaciones expresadas en un número finito de pasos dirigidos, donde cada paso debe ser específicamente definido.

- Adecuación de la aplicación: la programación es una tarea humana, lenta y costosa, por ello todas las tareas no reeditables para las aplicaciones del procesamiento de datos.

Escribir programas para computadoras es una actividad que requiere una metodología científica, repetible y comprobable, para llegar a su fin.

Las fases en la construcción de un programa para resolver un problema mediante la computadora son las siguientes:

- Análisis del problema.
- Diseño del algoritmo.
- Programación.
- Ejecución y pruebas.

1.1.1 Análisis del problema:

El análisis consiste en estudiar el problema planteado para obtener una idea clara y concisa de los pasos necesarios para proponer un modelo para su solución. Este método no puede existir sin que se hayan especificado con claridad todos y cada uno de los componentes estructurales del problema. Esta etapa consiste en describir el modelo que mejor se adapte a la estructura del problema que estemos observando.

Para resolver un problema con una computadora hay que disponer de los datos de entrada, estudiar el tratamiento que se ha de realizar a dichos datos, la información que se desea obtener como resultado y de que manera debe presentarse. Después de analizar el problema, se han de conocer claramente tres cosas.

- Datos de entrada de que se dispone.
- Proceso o tratamiento que ha de realizarse con estos datos.
- Información de salida deseada.

ENTRADA ------> PROCESO ------> SALIDA

1.1.2 Diseño del Algoritmo:

Un algoritmo es un método para resolver problemas, una vez analizado el mismo se precisa diseñar un algoritmo que indique claramente los pasos a seguir para resolverlo.

Para realizar un determinado proceso, se le debe suministrar a la computadora una fórmula para la resolución de un problema (algoritmo), cuyo diseño debe ser independiente de la computadora que resuelve el problema.

El diseño del algoritmo, requiere en la mayoría de los casos, creatividad y conocimientos profundos de la técnica de programación. Es decir, la solución de un problema se puede expresar mediante un algoritmo.

En esta etapa se realizará una representación gráfica clara y detallada que muestre la secuencia en que se deben ejecutar las diferentes operaciones. Estas representaciones gráficas son las herramientas utilizadas para el análisis de la programación y pueden ser: diagramas de flujo, pseudocódigos y/o tablas de decisión.

1.1.3 Programación:

Una vez que el diagrama de flujo o el algoritmo de resolución del problema está definido se pasa a la fase de codificación del programa en cualquier lenguaje cuyo resultado será el programa fuente el cual sigue las reglas de sintaxis que el lenguaje escogido exija.

Después de codificado el programa, se introduce en la computadora mediante unos programas especiales llamados editores.

Una vez dentro de la computadora, el programa debe ser traducido al único lenguaje que éste entiende: Lenguaje de máquina. Dicha operación se realiza mediante el correspondiente programa traductor o compilador del lenguaje en el que está escrito el programa.

PROGRAMACIÓN ›› ANÁLISIS ›› ALGORITMO ›› CODIFICACIÓN ›› EDICIÓN ›› TRADUCCIÓN

1.1.4 Edición y Pruebas:

Antes de dar por finalizada cualquier labor de programación, es fundamental preparar un conjunto de datos lo más representativo posible del problema, que permitan probar el programa cuando se ejecute y así verificar los resultados.

Cuanto más exhaustivas sean las pruebas de un programa, mayor seguridad de que éste funcione correctamente y menor posibilidad de errores.

El programa se considera terminado cuando se han realizado pruebas y ensayos de su fiabilidad con el conjunto de datos seleccionados y otros nuevos, hasta incluso con datos reales, y no se encuentren errores de ningún tipo.

1.2 CONCEPTO DE ALGORITMO:

Algoritmo: es un conjunto ordenado y finito de pasos que especifican la secuencia de operaciones que se han de realizar para resolver un problema.

Los algoritmos son independientes del lenguaje de programación en que se expresan como así también de la computadora que se ejecuten. Un algoritmo se puede expresar en distintos lenguajes de programación y en computadoras distintas, pero el algoritmo, los pasos a seguir para la solución del problema es siempre el mismo.

Los algoritmos son más importantes que los lenguajes de programación e incluso que las computadoras, dado que los lenguajes de programación son solo un medio para expresar un algoritmo y las computadoras la herramienta que los ejecuta. Es imprescindible que el algoritmo elegido para resolver el problema sea absolutamente claro, sin ambigüedades y además contemple todas y cada una de las posibles situaciones que puedan presentarse durante la resolución del mismo.

1.2.1 Características:

Se puede observar que el número de operaciones que realiza un algoritmo es finito siempre y cuando sus datos sean adecuados. Por consiguiente, el número de operaciones que necesitamos realizar al ejecutar un algoritmo dependerá de los datos del problema y solamente se conocerá al ejecutar este.

Un algoritmo debe ser:

- Preciso: debe indicar el orden de realización de cada paso.
- Definido: si se ejecuta dos veces el algoritmo con los mismos datos éste debe dar el mismo resultado.
- Finito: debe finalizar en algún momento o sea tener un número finito de pasos.

Todo algoritmo tiene tres partes: entrada, proceso y salida, y sus pasos describen la información de la entrada en la salida.

1.3 METODOLOGÍAS PARA RESOLVER UN PROBLEMA:

Una vez que el algoritmo esté diseñado se debe proceder a representarlo gráficamente mediante algún método de programación. Una vez realizado el algoritmo se procede a su escritura en algún lenguaje de programación para su posterior ejecución.

Esta representación gráfica independiza al algoritmo del lenguaje de programación elegido, permitiendo de esta manera que pueda ser codificado indistintamente en cualquier lenguaje. Para conseguir este objetivo se necesita que el algoritmo sea representado gráficamente, de modo que las sucesivas acciones puedan ser traducidas fácilmente a un programa, es decir su codificación.

1.3.1 Diagramas de flujo:

Diagrama o esquema que utiliza símbolos estándar y que tiene los pasos del algoritmo escritos en esos símbolos unidos por flechas, llamados flujo de datos o líneas de flujo, que indican la secuencia en que se deben ejecutar las órdenes, instrucciones o sentencias.

La naturaleza gráfica de los diagramas de flujo facilita inicialmente la comprensión de dichos algoritmos.

Los diagramas de flujo pueden ser de dos tipos:

- Diagramas de flujo del sistema: diagramas destinados a describir el flujo de información entre los distintos soportes físicos de un sistema informático. Reflejan operaciones normales para el desarrollo del proceso, que realizan los componentes utilizados en un programa.
- Diagrama de flujo de proceso u ordinograma: son las órdenes en secuencia que se deben dar a la máquina para la resolución del programa.

El uso de diagramas de flujo como herramienta de programación tiene beneficios que se detallan:

- Rápida comprensión de las relaciones.

- Se pueden usar como modelos de trabajo para el diseño de nuevos programas.

- Documentación adecuada de los programas.

- Produce una codificación eficaz en los programas.

- Depuración y pruebas ordenadas de programas.

- Fácil de traducir a cualquier lenguaje de programación.

Limitaciones o inconvenientes:

- Los diagramas complejos y detallados suelen ser laboriosos en su planteamiento y realización.
- No existen normas fijas para la elaboración de los diagramas de flujo que incluyan todo lo que el usuario desea introducir.

Estructuras Básicas:

1. Secuencia: se compone de un grupo de acciones que se realizan y en el orden que están escritas, sin posibilidad de omitir ninguna de ellas. Las tareas se suceden de forma tal que la salida de una de ellas es la entrada de la siguiente y así sucesivamente hasta el final del proceso.

2. Alternativa o Selectiva: permite seleccionar entre dos grupos de acciones dependiendo de que una determinada condición se cumpla o no. Estas estructuras se utilizan para tomar decisiones lógicas. Las condiciones que se especifican usan expresiones lógicas y usan la figura geométrica en forma de rombo. Estas estructuras pueden ser Simples o Dobles.

- Simples: solo obliga a realizar acciones si se cumple la condición. El no cumplimiento de la condición implica que no se realizará ninguna acción.
- Doble: el cumplimiento o no de la condición lógica obliga a la ejecución de diferentes grupos de acciones.

3. Iteración o Repetitiva: repetir una o varias instrucciones un número determinado de veces que vendrá determinado por una condición. Esta condición se conoce como condición de salida.

Existen dos tipos de iteraciones:

- Hacer Mientras: se caracteriza porque la condición de salida del bucle está situada al comienzo del mismo, es decir las acciones la hace mientras se cumple determinada condición.Cuando se ejecuta una estructura de este tipo, lo que primero se hace es evaluar la condición, si la misma es falsa no se realiza ninguna acción. Si la condición resulta verdadera entonces se ejecuta el cuerpo del bucle. Este mecanismo se repite mientras la condición sea verdadera.

- Hacer Hasta: se caracteriza porque la condición que controla la realización de las acciones del bucle está al final del mismo. En este tipo de iteración las acciones se repiten mientras la condición sea falsa, lo opuesto a la estructura hacer mientras. Este tipo de bucle se usa para situaciones en las que se desea que un conjunto de instrucciones se ejecute al menos una vez antes de comprobar la condición de iteración.

1.3.2 Pseudocódigo:

Cuando hablamos de pseudocódigo nos referimos a una imitación o simulacro de instrucciones reales abreviadas para las computadoras.

El uso de esta herramienta tiene la ventaja en la planificación de un programa.

El pseudocódigo es una especie de programa borrador o una forma de plantear un proceso de forma tal que su traducción a un lenguaje de alto nivel sea más sencilla para el programador.

Algoritmo de la tortilla de patatas:

1 Cortar patatas en tacos

2 Calentar aceite en sartén

3 Añadir patatas a sartén

4 Añadir cebolla a sartén

5 Freír patatas y cebolla

6 Batir huevos en cuenco

7 Añadir patatas y cebolla al cuenco

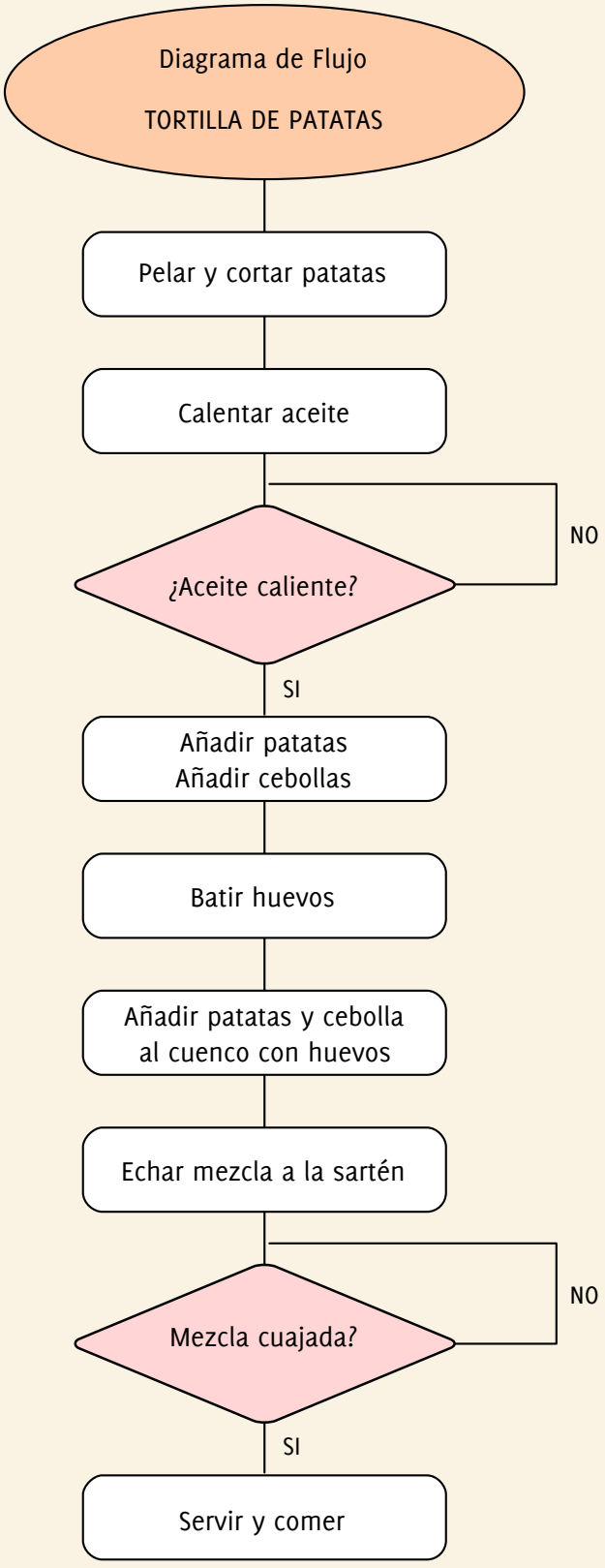
8 Quitar aceite sartén

9 Echar la mezcla de cuenco a la sartén

10 Cuajar un poco y dar la vuelta

11 Terminar de cuajar

Diagrama de flujo de la tortilla de patatas:



Capitán Sevilla, El Remake (I)

Sergi Caparrós - scaparros@papirux.org

Para descargar el juego lo mejor es visitar [1] y descargar los paquetes (.deb). Si nuestra distribución no está basada en Debian, siempre podemos bajar los fuentes y compilarlos. El juego está disponible también para Windows. Se distribuye bajo licencia GPL.

Desde su página web [2] podemos descargar las distintas versiones del juego y los manuales (tanto inglés como en español) de los cuales hemos obtenido esta documentación (bajo licencia CopyLeft).

Introducción

Amanece un día tranquilo sobre Sevilla. Mariano López se dirige a su trabajo como transportista de embutidos. Mientras tanto, en el centro de aparatos interestratosféricos de Santiponce, se preparan para el lanzamiento del primer cohete espacial enteramente español, diseñado por el prestigioso Doctor Torrebruno, eminente físico nuclear que tuvo que abandonar el proyecto debido a un extraño desorden mental.

Lo que nadie esperaba es que el cohete sufriera un brusco cambio en su trayectoria, y cayese justo encima del camión en el que Mariano López transportaba un cargamento de morcillas. Mariano, inconsciente, queda tendido en la calzada junto a sus embutidos y a los restos del camión, que se esparcen por toda la zona del siniestro. Horas más tarde se despierta hambriento, e ingiere una caja de sus morcillas afectadas por la radiación.

Pero, ¡¡Qué diablos está sucediendo!? Mariano se está transformando: sus músculos crecen, su estatura aumenta, su vulgar rostro se convierte en el de un apuesto súper-héroe, el Capitán Sevilla, que descubre que todo ha sido un plan del malévolo Torrebruno, con la intención de conquistar el Mundo empezando por Sevilla.

PRIMERA PARTE: Los mutantes atacan Sevilla

En la primera parte del juego te encontrarás en Sevilla, con sus edificios, obras, parques, etc. Tu misión consiste en localizar el cohete espacial que te permitirá viajar al planetoide - manicomio Congrio, escenario de la segunda parte. Pero habrás de ser cauto, ya que la explosión nuclear ha transformado en mutantes a muchos de los habitantes de la ciudad, que están dispuestos a defender a Torrebruno en su malévolo plan.

EL MENÚ

1. Empezar a jugar

2. Nombre del idioma actual: Selecciónalo para cambiar el idioma

3. Configuración

3.1. Configurar Juego

3.1.1. Iniciar en fase: Sirve para intercambiar la fase inicial entre las dos que componen el juego. Si no hemos llegado nunca a la primera parte, o jugamos desde un soporte de sólo lectura, tendremos que introducir la contraseña obtenida al finalizar la primera parte

3.1.2. Dificultad: Ajústala a tu gusto entre 3 niveles diferentes

3.2. Configurar Pantalla

3.2.1. Nivel de detalle: 5 niveles de detalle gráfico, para que la fluidez del juego no se resienta por tarjetas gráficas de poca capacidad.

3.2.2. Scanlines: Activa o desactiva este filtro de emulación de TV a tu antojo

3.2.3. Resolución: Elige entre 640x480 (resolución "nativa"), 800x600 ó 1024x768.

3.2.4. Modo de Pantalla: Juega con el juego en ventana o a pantalla completa

3.3 Configurar Sonido

3.3.1. Volumen de FX

3.3.2. Volumen de música

3.4. Configurar Controles

3.4.1. Sistema de control: Teclado, Joystick (si hay alguno disponible), o ambas cosas.

3.4.2. Definir teclado

4. Extras

4.1. Volver a ver la introducción del juego

4.2. Ver créditos

5. Salir al S.O.: Abandona el juego, y vuelve al Sistema Operativo.

Si tras unos segundos no pulsas ninguna tecla, el juego iniciará un modo de demostración.

Debes evitar el contacto con los siguientes personajes:

CURRO:

Es un obrero con muy malas pulgas. En cuanto te divise irá directo hacia ti. Evita como puedas el contacto e intenta destruirle.

SEBASTIÁN, MANOLITO y JOSELITO:

Son tres amigos a cual más travieso que, además, han caído bajo el control mental de Torrebruno. Con sus trastadas te harán pasar malos momentos pero no les hagas nada; al fin y al cabo son niños.

MANOLO EL CAMARERO:

Sádico y vil personaje afectado por la radiación que, sin embargo, solo te atacará cuando estés bajo los efectos de una morcilla. Si le das un par de buenos derechazos, conseguirás la morcilla que lleva en su bandeja.

LONG PRETTY WILLOBY:

Pistolero a sueldo contratado esta vez por el profesor Torrebruno. Creció en un ambiente hostil que le condujo al vicio y a la corrupción. Ten cuidado con él.

FATTY BLACK JONES:

Compañero inseparable de LONG PRETTY WILLOBY es incluso más hábil que éste manejando la "pipa". Un consejo: no le pierdas de vista. Estos dos degenerados aparecen cuando llevas mucho tiempo en la misma pantalla.

DUMBO PINK:

No, no tienes una copa de más encima. Se trata del último espécimen de los ELPHANTUS VOLADORUS. Aunque parezca inocente y simpático no te fíes de él.

ROBOTROIDE GEOSINCOPADO:

No te pierde de vista y en cualquier momento te suelta un chorro de electrones hidrolizados.

HAMBURGUESOIDE VOLATIL:

No intentes comértela ¡DESTRÚYELA!

DUSTBIN:

El hedor que despide es insoportable. Trata de alejarte de él.

MARY GROUPY:

Insaciable fan del CAPITÁN SEVILLA. No la dejes que se acerque: lleva "malas" intenciones

RETORTERO AEROSTÁTICO:

Unidad autónoma de vigilancia aeroespacial.

BRAVE BULL:

Es un toro bestial que te persigue. No le dejes nunca que se te acerque.

GORGOJO CARRASQUERO:

Pequeño alien que se encargará de proteger ciertas zonas de la pantalla. Es pequeño, pero no por ello inofensivo.

ALIENÍTICUS ALIENANTE:

Fiera brutal, originaria de las profundidades del planetoide Congrio, ha sido enviado por Torrebruno dentro de su diabólico plan.

CHEPPY TRON:

Androide acorazado casi impenetrable.

ATTACK SOLDIER:

Está entrenado para matar. Cuidado con su fotodisparo de megaprotones.

Aparte de estos enemigos, tendrás muchas más sorpresas que evitar: Mutantes que han ocupado las casas de Sevilla y te lanzarán objetos, sierras automáticas que intentarán mutilarte, ríos de lava que evitar, etcétera.

Todo un camino de aventura, hasta que consigas llegar a la Base de Lanzamiento de Aparatos Interestratosféricos, donde hay preparado un cohete con destino al planetoide Congrio, donde podrás enfrentarte a Torrebruno, y salvar Sevilla y el Mundo.

Controles

· Movimiento: Cursores
· Fuego: Control izquierdo
· Selección: Enter

Utiliza el control de Selección para comerte las morcillas cuando seas Mariano, y para elegir entre los distintos superpoderes del Capitán Sevilla.

Utiliza el control de Fuego para pegar puñetazos, cuando seas Mariano, y para utilizar el superpoder seleccionado cuando seas el Capitán.

Cuando representes a Mariano en el juego tus posibilidades son pocas. Trata de evitar por todos los medios que algún personaje te toque; si lo hace perderás una vida. Por lo demás dedícate a dar puñetazos para defenderte y busca alguna morcilla para transformarte en el CAPITÁN SEVILLA.

Sin embargo, algunos de los enemigos no te atacarán mientras vayas con esa apariencia.

Enlaces:

[1]http://computeremuzone.com/ficha.php?id=754&pg=develop
[2]http://computeremuzone.com/ficha.php?id=754

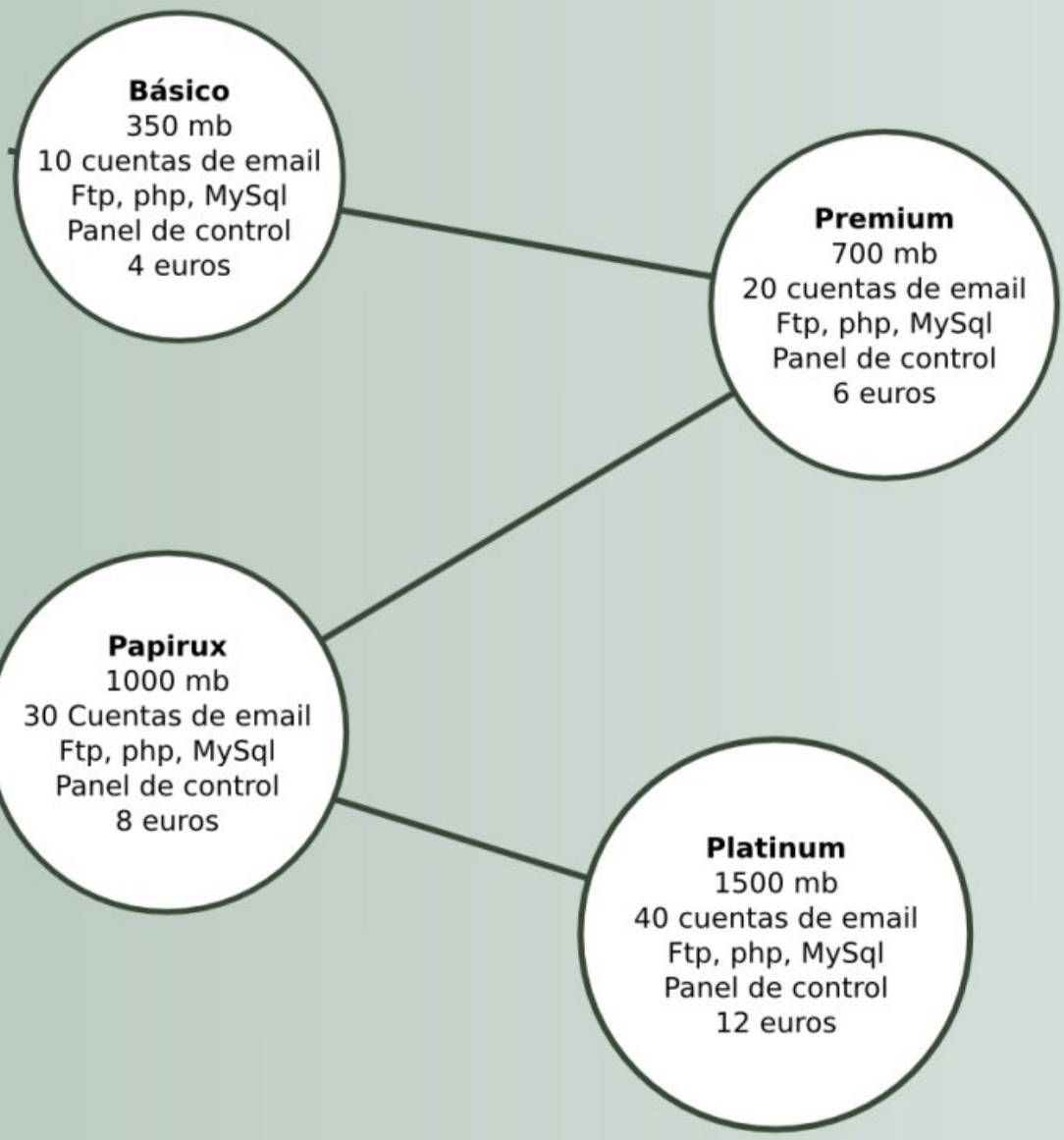
Equipo de desarrollo

Programación: Luis I. García Ventura
Gráficos: Daniel Celemín García
Música: Daniel Celemín, David Cañadas y Augusto Ruíz

SFERA HOSTING

Todos nuestros productos disponen de transferencia ILIMITADA
Promoción especial para los lectores de Papirux: El primer mes es gratis en cualquiera de los productos (excepto compra de dominios)

Planes WebHosting



Sitio: http://www.sferahosting.com/
E-mail: comercial@sferahosting.com

*Pagando un año se regala un mes.

¿Te interesa colaborar con Papirux?

Puedes enviarnos tus ideas, propuestas, artículos, opiniones a nuestra dirección de correo:

papirux@papirux.org

También puedes contactar con nosotros mediante **IRC** en el canal **#papirux** en **freenode.org**, o suscribirte a nuestro grupo de correo en **GoogleGroups**.