QESPISQA KAWSAY

CULTURA LIBRE

Redes Virtuales usando Qemu

DreamWeaver 8 en GNU/Linux

Gestor de Iconos de Escritorio IconMgr

Conexión a Postgresql en FreePascal

FreePascal y Lazarus en FreeBSD

Entrevista a: Alonso Cárdenas Commiter Arequipeño de FreeBSD

http://www.aqpglug.org.pe IRC. server irc.freenode.net canal #aqpglug

aqpglug

Noticias

Es el evento de difusión de Software Libre más grande en Latinoamérica.
Se realiza desde el año 2005 y su principal objetivo es promover el uso del software libre, dando a conocer al público en general su filosofía, alcances, avances y desarrollo.
El FLISOL 2007 se efectuará el día sábado 28 de abril. ¿Cómo me entero si en mi ciudad se realizará el evento? Sigue este enlace:
http://installfest.info/FLISOL2007/Ciudades

Gaim cambia de nombre, ahora es Pidgin.

El cliente de mensajería instantánea multiprotocolo GAIM cambió su nombre a Pidgin. Este se debio a que AOL (America On-Line, dueña de la marca "AIM") amenazara con demandar a Sean Egan, el creador de GAIM.
El acuerdo legal entre ambas parte concluyó también con el cambio de la denominación de libgaim a libpurple y gaim-text a Finch.
http://www.pidgin.im/

El 8 de abril del presente se hizo el anuncio oficial de la liberación de la version estable 4.0 llamada "etch", esto despues de 21 meses de constante trabajo.
http://www.debian.org/News/2007/20070408

Al mismo tiempo se anuncio la release 6 de la version anterior 3.1 estable de Debian.
Para descargarlo:
http://www.debian.org/CD/

Índice

Número 1 - Abril 2007

Editor
Luis Revilla

Colaboradores
Alonso Cárdenas
Ayax Fernández
Abraham Montaño
Carlos Zúñiga

Diseño y diagramación
Ayax Fernández

Carátula
José Antonio Rodríguez

AQPGLUG

www.aqpglug.org.pe

Lista de correo:
revista@listas.aqpglug.org.pe

Editorial

Después ya de algunos meses, que surgiera la idea de desarrollar una revista; que tratase de temas referidos a las tecnologías libres, de tener continuas reuniones y cumplir con todas las tareas respectivas para su realización, finalmente se obtuvo los frutos deseados con este, el primer número.

La revista representa para nuestra comunidad AQPGLUG un reto, para demostrarnos de lo que somos capaces de hacer, tenemos varios proyectos en marcha y muchos más esperando, de los cuales son pocos los que se podrán concretar en el presente año.

El desarrollo del software libre en estos últimos años, ha llevado a grandes cambios en diferentes áreas, esto no solo ha afectado a los informáticos, sino que se ha convertido en parte de la vida de muchas personas de distintas especialidades, que gracias a su retroalimentación han hecho crecer al movimiento y se han convertido en esencial para su desarrollo.

En este primer número, se ha incluido diferentes artículos que van desde programación hasta emulación, en nuestro país existen diferentes tipos de proyectos destinados a la difusión o creación de soluciones libres, que merecen el apoyo y difusión de los demás; para ello hemos creado una sección denominada Proyectos Nacionales, que cumplirá este fin.

Esperamos satisfacer nuestras espectativas y aprender de nuestros errores, mi sincero agradecimiento a todas las personas que colaboraron y nos dieron su apoyo para hacer realidad esta revista. Espero que muchas más personas se unan a este esfuerzo y mantengan vivo este sueño.

Editor

Luis Revilla A.

REDES VIRTUALES USANDO QEMU

Escrito por: Luis Revilla Amézquita

Qemu es emulador publicado bajo licencia GPL , que nos permite la ejecucion de sistemas operativos en un ambiente virtualizado.

Introducción

Qemu es un emulador de procesador publicado bajo la licencia GPL , que nos permite la ejecución de sistemas operativos para arquitecturas diferentes como:
- PC (procesadores x86)
- PREP (procesadores PowePC)
- PowerMac (procesadores PowerPC)
- Sun4m (procesadores Sparc).

Logrando obtener maquinas virtuales, por lo tanto tener su propio ambiente de trabajo; lo cual nos permitirá la ejecución de las diferentes aplicaciones nativas de cada sistema operativo.
Durante el desarrollo de qemu se vio en la necesidad de aumentar su rendimiento, para ello se añadió un modulo a nivel kernel llamado "QEMU Accelerator", el cual nos permite acelerar la ejecución de emulación.
La utilización de este modulo no es necesaria para el funcionamiento del emulador QEMU, pero por motivos de rendimiento, su uso se aconseja.

Comandos Básicos

Creación de imágenes de disco

Previamente en una consola ejecutar "man qemu-img" nos mostrara información muy útil en la creación de los discos virtuales, rápidamente crearemos un disco virtual de 1Gbyte y con el formato que aconseja la documentación de qemu el qcow.

```
# qemu-img create -f qcow discoA.qcow 1G
```

Asignación de memoria virtual (Opcional)

Tenemos el disco virtual ahora un punto importante es la memoria virtual utilizada por defecto es 128 Megas, si nuestra intención es que utilize mas memoria virtual se le puede asignar de la siguiente manera:

```
# umount /dev/shm
# mount -t tmpfs -o size=XXm none /dev/shm
```

Donde XX es la cantidad de memoria q se le va asignar "m" indica megas ,por lo tanto si queremos asignarle 250 megas ala memoria virtual de QEMU seria de la siguiente forma.

```
# umount /dev/shm
# mount -t tmpfs -o size=250m none /dev/shm
```

Un punto importante a considerar al asignarle memoria virtual , es tomar en cuenta la memoria de nuestro sistema la memoria física y la de intercambio swap. No se le puede asignar una cantidad de memoria mayor ala que tiene nuestro sistema operativo.

Acelerador QEMU (Opcional)

Para aumentar el rendimiento ala maquina virtual se recomienda usar el modulo de aceleración ,para ello previamente se ha tenido que descargar el código fuente , compilado y instalado correctamente:

```
./configure
make
make install
depmod
```

Para cargar el modulo "modprobe kqemu major=0"

Nota:

Para obtener el código fuente de QEMU y Acelerador QEMU:
http://fabrice.bellard.free.fr/qemu/

Consejos Útiles

Creación de imágenes de disco

```
$qemu-img create -f qcow discoA.qcow 1G
```

El formato propuesto por la documentación es el .qcow
Para más información revisar el man.

Asignación de Memoria Virtual

```
$sudo mount -t tmpfs -o size=XXm none /dev/shm
```

La memoria virtual utilizada por defecto es de 128 Megas.
Antes de cambiar el tamaño de la memoria virtual debemos desmontar la partición:

```
$sudo umount /dev/shm
```

Iniciando QEMU

Previamente se recomienda revisar el manual para conocer algunos parámetros que utilizaremos , ejecute en un terminal el siguiente comandos: qemu -h o man qemu

Los parámetros que nos interesan son los siguientes:

```
Standard options:
-fda/-fdb file   use 'file' as floppy disk 0/1 image
-hda/-hdb file use 'file' as IDE hard disk 0/1 image
-hdc/-hdd file use 'file' as IDE hard disk 2/3 image
-cdrom file     use 'file' as IDE cdrom image (cdrom is ide1 master)
-boot [a|c|d]  boot on floppy (a), hard disk (c) or CD-ROM (d)
-m megs        set virtual RAM size to megs MB [default=128]

Network options:
-net nic[,vlan=n][,macaddr=addr][,model=type]
            create a new Network Interface Card and connect it to
VLAN 'n'
-net tap[,vlan=n][,fd=h][,ifname=name][,script=file]
            connect the host TAP network interface to VLAN 'n' and use
            the network script 'file' (default=/etc/qemu-ifup);
            use 'fd=h' to connect to an already opened TAP interface
```

Colocar un CD de instalacion en la lectora y ejecutar de la siguiente forma:

```
qemu -m 100 -cdrom /dev/cdrom -hda
/path/imagenCreada/discoA.qcow -boot d
```

¿Que le hemos indicado al ejecutar el qemu con estos parametro?

-m 100 Nos indica que vamos asignarle ala memoria virtual 100 megas de los 128 que son asignados por defecto.

-cdrom /dev/cdrom Indica que el dispositivo de la lectora cdrom (dev/cdrom) va hacer usado, pero tambien se le puede indicar que use una imagen ISO, para ello solo se le indica la ruta donde se encuentra esa imagen.

-hda /path/imagenCreada/discoA.qcow Como se lee en su manual hda indica el primer disco IDE, se pueden colocar hasta los 4 discos IDE, para ello solo hay que crear esos discos virtuales con el comando " qemu-img ", por lo tanto tendriamos lo siguiente en la linea de comandos:

```
-hda /path/imagenCreada/discoA.qcow
-hdb /path/imagenCreada/discoB.qcow
-hdc /path/imagenCreada/discoC.qcow
-hdd /path/imagenCreada/discoD.qcow
```

-boot d Como su nombre lo indica es el buteo del sistema al indicarle " d " se iniciara desde la lectora de cdrom. ("a" desde la disketera, "c" desde el primer disco duro IDE hda, "d" cdrom).

Con todos estos pasos se podra iniciar correctamente el emulador de procesador QEMU, lo primero a tener encuenta, como en toda PC primero arracará el BIOS y luego se iniciará desde el CDROM. Con esto tenemos una máquina virtual donde se pueden ejecutar: una distribucion linux , *bsd, algun sistema operativo comercial o también un LiveCD.

Para propositos de este artículo sobre Redes Virtuales, usaremos un livecd llamado: "Damn Small Linux"
http://www.damnsmalllinux.org/

Redes Virtuales

Al tener nuestra propia maquina virtual con un ambiente propio, permitirá tener una red virtual por lo cual podemos entre otras cosas tener acceso a Internet , conexiones ftp, ssh, y otros servicios más.

Pero para obtener todos estos servicios debemos de conocer la implementación para redes del QEMU, hay dos formas de usar la red virtual que a continuación se los indico.

Modo –net user: Es la que viene por defecto, no hay necesidad de colocar el parámetro respectivo -net user. Que consta de un servidor DHCP, que configura automáticamente la red virtual.

```
-net user[,vlan=n][,hostname=host]
            connect the user mode network stack to VLAN 'n' and
send
            hostname 'host' to DHCP clients
```

La configuración DHCP es la siguiente:

```
QEMU Virtual Machine    <----> Firewall/DHCP server <----> Internet
            (10.0.2.x)      |          (10.0.2.2)
                            |
                            ----> DNS server (10.0.2.3)
                            |
                            ----> SMB server (10.0.2.4)
```

Con esto basta para tener una conexión al exterior, lo cual nos permitirá acceder a Internet y a los servicios que ofrece.

Para el acceso a nuestro Host, me refiero a los servicios que ofrece nuestra maquina anfitrión, también se puede acceder desde la red virtual, si tenemos servicios ssh, ftp, web, etc. El acceso es transparente pero aquí viene el problema en este tipo de configuración de la red virtual, no podemos acceder desde nuestro host hacia la red virtual; me explico si nuestro livecd DSL tiene el servicio ssh activo , desde nuestro host que es el anfitrión no podemos acceder por la red virtual hacia el servicio ssh del livecd DSL por lo tanto se necesita otro tipo de implementación.

Modor –net nic –net tap: El acceso es a través de túneles por la interfaz tapX , donde X simboliza el número de la interfaz creada, si se usa esta opción el acceso es transparente en ambos sentidos sin restricciones.

```
-net nic[,vlan=n][,macaddr=addr][,model=type]
            create a new Network Interface Card and connect it to VLAN 'n'
-net tap[,vlan=n][,fd=h][,ifname=name][,script=file]
            connect the host TAP network interface to VLAN 'n' and use
            the network script 'file' (default=/etc/qemu-ifup);
            use 'fd=h' to connect to an already opened TAP interface
```

De la teoría a la práctica demostraremos todo lo indicado con la ayuda livecd DSL; su imagen ocupa 50 megas.

Se usara el modulo de aceleración.
modprobe kqemu major=0

Primera implementación de la red virtual (por defecto)

No hay necesidad de colocar el parámetro -net user.

qemu -m 100 -cdrom /DirectorioCualquiera/dsl-3.0.1.iso -boot d

Como muestra la figura 1 el livecd se ejecuta como si estuviera en una PC real, el servidor dhcp que tiene internamente el QEMU, configura automáticamente la interfaz de red eth0.

En la figura 2 se observa que el acceso a Internet por nuestra red virtual es posible, ahora veamos cual es la configuración que tiene la interfaz de red virtual con ayuda de un terminal, solo considero los valores importantes no toda la información que proporciona estos comandos:

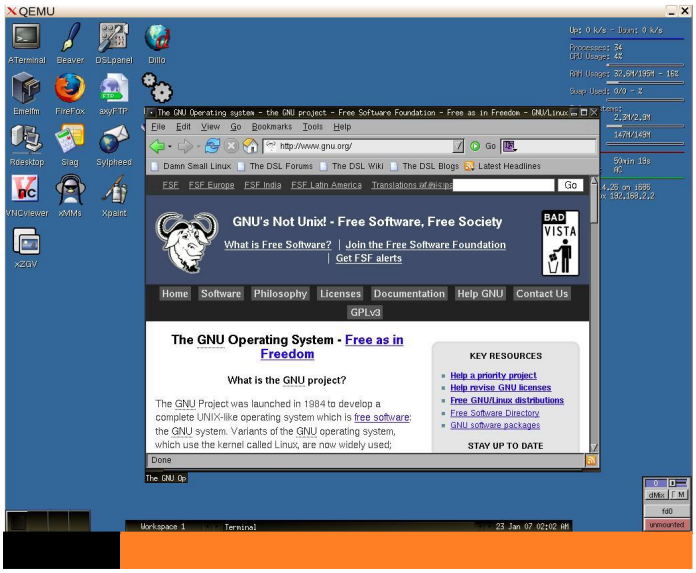
dsl@box:~$ ifconfig

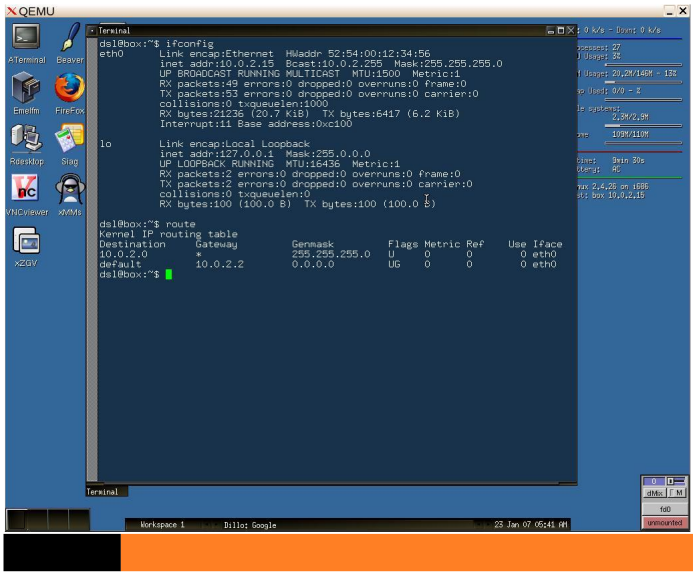
Nos indica que tiene la siguiente IP 10.0.2.15

dsl@box:~$ route

La puerta de enlace utilizada es la 10.0.2.2

Que son los valores que coloca el servidor DHCP a nuestra red. (Ver figura 3)





Ademas se puede verificar si se tiene acceso a los servicios proporcionados por el anfitrión que tiene estas características:

Anfitrión
la interfaz de red eth0 tiene la IP 192.168.1.33
la puerta de enlace 192.168.1.1
Servicios activos ssh, ftp, http.

En el siguiente cuadro se puede observar el acceso de los servicios desde el anfitrión ala maquina virtual y viceversa:

Red virtual QEMU --hacia --> Anfitrión	
ping 192.168.1.33	Conexión aceptada
ssh pc2@192.168.1.33	Conexión aceptada
ftp 192.168.1.33	Conexión aceptada

Anfitrión --hacia ---> Red virtual QEMU	
ping 10.0.2.15	Conexión rechazada
ssh dsl@10.0.2.15	Conexión rechazada

Como se observa en el cuadro la conexión solo se realiza en un solo sentido dela maquina virtual hacia el anfitrión y no viceversa.

Interfaz de red TUN/TAP

Para solucionar este problema se opta por el segundo metodo usando la interfaz de red TUN/TAP, atravez del dispositivo /dev/net/tun .

Para ello se necesita crear los siguientes archivos:
/etc/qemu-ifup
/etc/qemu-ifup-tun

El archivo /etc/qemu-ifup debe de contener lo siguiente:

```
#!/bin/bash
/etc/qemu-ifup-tun $1
```

El archivo /etc/qemu-ifup-tun debe de contener lo siguiente:

```
#!/bin/bash
echo "Interfaz d Red para el Tunel QUEMU -->" $1
ifconfig $1 192.168.2.1
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
route add -host 192.168.2.2 dev $1
```

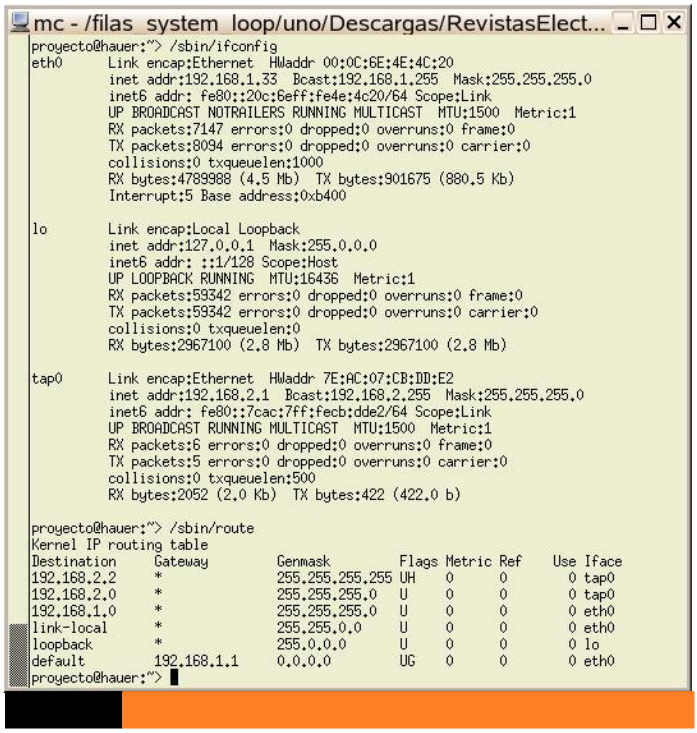
Finalmente ejecutamos nuestra maquina virtual QEMU con la imagen ISO del livecd DSL:
qemu -m 100 -net nic -net tap -cdrom /DirectorioCualquiera/dsl-3.0.1.iso -boot d

Una vez ejecutado nuestro livecd DSL en el QEMU procederemos a configurar nuestra red virtual para ello en un terminal ejecutamos:
dsl@box:~$ sudo ifconfig eth0 192.168.2.2 up
dsl@box:~$ sudo route add default gw 192.168.2.1

Editamos el archivo /etc/resolv.conf
dsl@box:~$ sudo vi /etc/resolv.conf
y modificamos nuestros DNS y colocamos
nameserver 192.168.2.1
nameserver 192.168.1.1

Tenemos que nuestra interfaz de red virtual tiene la dirección IP 192.168.2.2 y la puerta de enlace es 192.168.2.1

En el anfitrión
Se verifica la creacion del dispositivo de red tap0, a la que QEMU le ha asignado la dirección IP 192.168.2.1 para ello se ejecuta el comando "ifconfig" y "route" en una consola , y se despliega la información de todos los dispositivos de red existentes como se muestra en la figura 4.



Enlaces de Interés

QEMU
http://fabrice.bellard.free.fr/qemu/
QEMU Accelerator
http://fabrice.bellard.free.fr/qemu/qemu-accel.html
KQEMU
http://kqemu.sourceforge.net/
Damn Small Linux
http://www.damnsmalllinux.org/

Todos estos pasos básicos son necesarios para poder acceder a los servicios del host anfitrión, el cuadro siguiente se detalla con mas claridad el acceso transparente entre la red virtual y la red del host anfitrión.

Red virtual QEMU --hacia--> Anfitrión	
ping 192.168.1.33	Conexión aceptada
ssh pc2@192.168.1.33	Conexión aceptada
ftp 192.168.1.33	Conexión aceptada

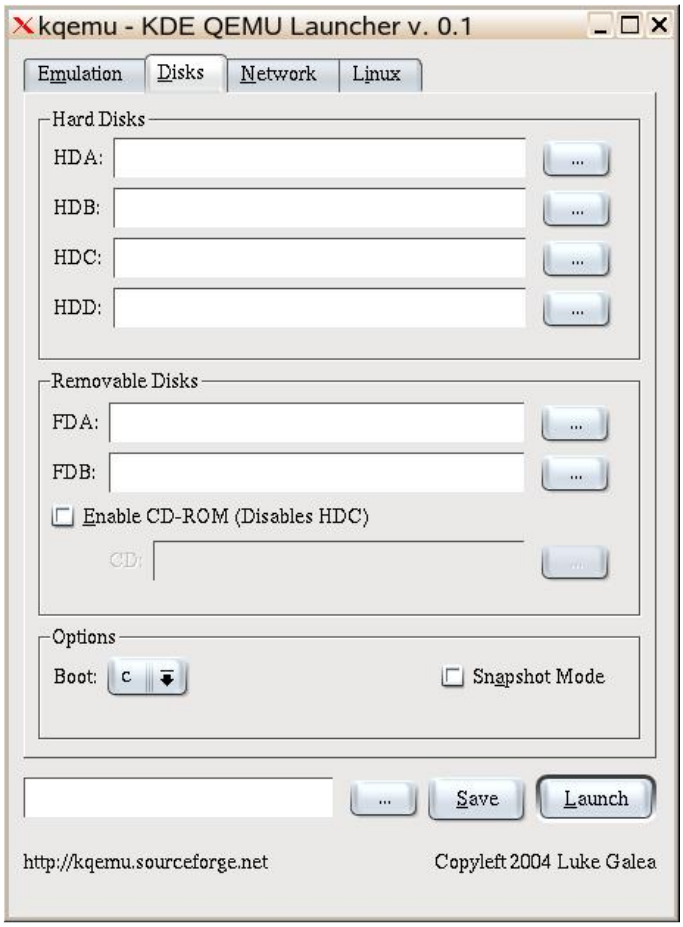
Anfitrión --hacia --> Red virtual QEMU	
ping 192.168.2.2	Conexión aceptada
ssh dsl@192.168.2.2	Conexión aceptada

Nota: La activación del servicio ssh en el livecd DSL se realiza ejecutando el script /etc/init.d/sshd start.

KQEMU Interfaz gráfica

Un tema aparte es la interfaz gráfica kqemu , que permite un uso mas agradable para el usuario.

Como lo muestra la figura 5 , su uso es muy sencillo con la selección adecuada de sus opciones permite la ejecución de qemu en pocos pasos.



Conclusión

Las maquinas virtuales son de gran ayuda, ya sea para el campo de la investigación o para la demostración del funcionamiento de cualquier sistema operativo. Espero que este articulo sobre la configuración de la red virtual les se ha de utilidad.

Entrevista a:

Alonso Cárdenas

Realizada por: Ayax Fernández Rosado

Ayax Fernández: ¿Desde hace cuanto tiempo estas involucrado con el movimiento de Software Libre y a que comunidades o proyectos perteneces?

Alonso Cárdenas: Empece con el software libre en el año 2000 aproximadamente, a raíz de la curiosidad que me despertó conocer acerca de un sistema llamado GNU/Linux. Con un grupo de amigos fundamos AQPLINUX, un grupo de usuarios de la ciudad donde radico (Arequipa), después fue cambiando de nombre a LINUXAREQUIPA y finalmente a lo que hoy es AQPGLUG, del cual soy miembro actual. Entre las comunidades y proyectos en los cuales estoy involucrado puedo nombrar PHP es español, BSD Perú, PCBSD en español, ElDemonio, FreeBSD y algunos otros mas.

A.F.: ¿Cuál fue tu primera distribución o sistema operativo libre?

A.C.: La primera distribución que conocí y utilice fue RedHat, en aquella época en su versión 6.2 y FreeBSD 4.1

A.F.: ¿Cómo es que te relacionas con el proyecto freebsd y decides apoyarlo activamente?

A.C.: Como te comente anteriormente la primera versión de FreeBSD que conocí fue la 4.1 pero solo lo instale: inicie el sistema, coloque mi usuario y eso fue todo lo que hice con el; desde entonces siempre lleve un tema pendiente con FreeBSD, hasta que, aproximadamente a finales del 2004, es que decidí retomar lo que tiempo atrás había dejado. Empece a conocer más acerca de FreeBSD y llegue a un punto en el cual, tenía la necesidad de aportar más cosas y no solo dedicarme a ser un usuario mas; es ahí que empece fundando BSD Perú, y participando reportando algunos problemas que encontraba en mi uso diario de mi sistema.
Utilizando el árbol de ports, en una de las aplicaciones que usaba, el Idesk, que no era actualizado un buen tiempo; entonces, decidí enviar un parche para actualizar el port; a lo que el mantenedor me dijo que si deseaba ser el nuevo mantenedor del port, acepte y ahí empezó mi aventura, para lograr convertirme en un commiter de FreeBSD.

A.F.: ¿Desde hace cuanto eres committer del proyecto FreeBSD, y en que área o áreas apoyas?

A.C.: Oficialmente soy commiter del proyecto FreeBSD desde el mes de Julio del 2006; 20 días antes fui propuesto por mi mentor Renato Bothelo, me propuso hacia el portmgr y convertirme en nuevo integrante del equipo de port commiters.

Actualmente mi trabajo se enfoca en los ports y apoyando en la documentación en español, haciendo la traducción de algunos documentos, como son: la guía de uso de tinderbox y el porter's handbook.

A.F.: ¿Qué proyectos futuros tienes planeados?

A.C.: Ampliar mi privilegios en el árbol CVS: para documentación (doc), source (src) y continuar con algunos proyectos que tenemos pendiente en AQPGLUG.

A.F.: Algunas palabras finales, que desees compartir con nuestros lectores.

A.C.: Bueno todos aquellos que quieran ingresar al mundo de los BSD pueden encontrarnos en el canal #bsd.pe, en los servidores de freenode y no teman hacer preguntas, que todos estamos para aprender.

Gracias por tu tiempo y colaborar con la revista.

Más información

Alonso es desarrollador de FreeBSD, en el área de los PORTS, actualmente vive en Arequipa - Perú.
En sudamérica solo hay 5 desarrolladores de FreeBSD (3 en Brasil, 1 en Peru y 1 en Argentina)

EJECUTAR Dream Weaver 8 en GNU/Linux

Escrito por: Abraham Montaño

Para emular el dreamweaver 8 en una computadora con sistema operativo GNU/Linux necesitaremos la aplicación wine version 9 o superior, en esta ocasion lo haremos con la distribución de GNU/Linux Ubuntu.

Los pasos generales seran iguales para cualquier distribucion de GNU/Linux:

Primero

Debemos instalar wine.

```
$sudo aptitude update
$sudo aptitude install wine
$wine
```

(ejecutaremos wine para crear su espacio de trabajo).

Segundo

Para este ejemplo asumiremos que tenemos instalada una particion con windows y tenemos montada esa partición, procedemos a ejecutar los siguientes comandos:

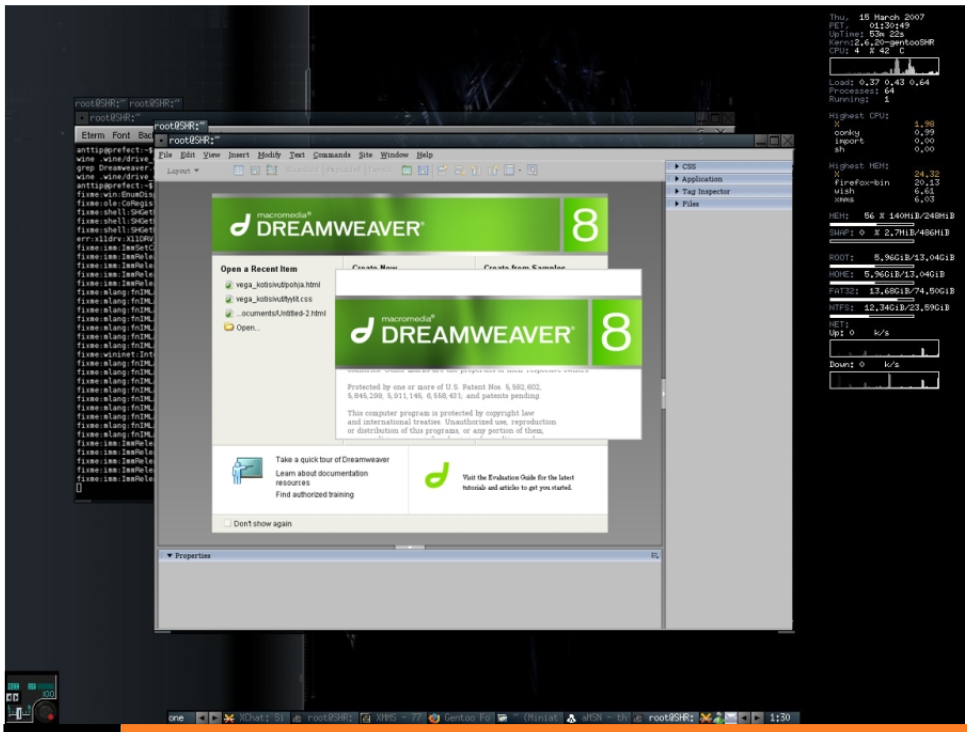


Figura 1 Captura de pantalla de DreamWeaver8 emulada en Gentoo por el autor del tutorial

```
$cd ~/.wine/drive_c/Archivos\ de\ programa/
$cp -R /mnt/windows/Archivos\ de\ programa/Macromedia .
$chmod -R +w Macromedia
$cd ~/.wine/drive_c/windows/system32
$cp -R /mnt/windows/WINDOWS/system32/Macromed/ .
$chmod -R +w Macromed
$cd ~/.wine/drive_c/windows/profiles/All\ Users/Application\ Data/
$cp -R /mnt/windows/Documents\ and\ Settings/All\Users/Datos de
  programa/Macromedia/ .
$chmod -R +w Macromedia
$cd ~/.wine/drive_c/Archivos\ de\ programa/Archivos\ comunes/
$cp -R /mnt/windows/Archivos\ de\ programa/Archivos\ comunes/
Macromedia/ .
$chmod -R +w Macromedia/
```

Tercero

Ahora tendremos que exportar el registro de macromedia que tenemos en windows a un archivo reg para luego cargarlo al registro del wine. En windows ejecutemos regedit y exportemos esta ubicación "HKEY_LOCAL_MACHINE/Software/Macromedia/" a un archivo de nombre macromedia.reg.
Instalamos recode para cambiar el formato del archivo macromedia.reg

```
$apt-get install recode
$recode ucs-2..ascii macromedia.reg
```

Agregamos el registro a wine:

```
$wine regedit macromedia.reg
```

Cuarto

Ahora solo nos queda ejecutar nuestro dreamweaver.

```
$ wine .wine/drive_c/Archivos\ de\
programa/Macromedia/Dreamweaver\8/Dr
eamweaver.exe
```

Con eso ya deberiamos tener funcionando el DreamWeaver (como se muestra en la figura 1).

NOTA: Si usted tiene una particion windows y no piensa eliminarla en vez de copiar los archivos puede crearle los enlaces simbolicos con el comando "ln -s <ruta del directorio o archivo> <nombre del enlace simbolico>"

Instalación de FreePascal y Lazarus en FreeBSD

Escrito por: Alonso Cárdenas

Una guía para la instalación y configuración de FreePascal; un compilador libre, del lenguaje de programación pascal, para arquitecturas de 32 y 64 bits; con soporte para un buen número de plataformas.
Lazarus un conjunto de librerías y un IDE que emula a delphi, todo esto en FreeBSD.

Instalando FreePascal

Existen múltiples formas para instalar FreePascal: desde código fuente utilizando el árbol de ports o también utilizando lo paquetes pre compilados.

Desde Código Fuente

Para instalar desde código fuente, solo hace falta ubicar la ruta de nuestro port, en nuestro caso, el compilador de freepascal se encuentra en /usr/ports/lang/fpc y procedemos a instalarlo.

```
# cd /usr/ports/lang/fpc && make install clean clean-depends
```

o también podemos utilizar la herramienta sysutils/portupgrade

```
# portinstall fpc
```

Con esto automáticamente se descargara todo lo necesario para compilar e instalar FreePascal en nuestro sistema.

Desde Paquetes

Si deseamos utilizar los paquetes pre compilados ejecutaremos lo siguiente:

```
# pkg_add -r fpc
```

Este comando también descargará e instalará todos los paquetes pre compilados de freepascal y todas sus dependencias.

TIPS previos

Actualmente freepascal funciona sin problemas en version 5.x y 6.x de FreeBSD, se recomienda instalarlo en la version 6.x, ya que el soporte de versiones inferiores esta pronta a terminar.
También necesitaremos que nuestro arbol de ports este actualizado, para esto podemos usar la herramienta portsnap, para la actualización de nuestro arbol de ports. Para más información acerca de su uso, puedes ver este enlace:
http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/portsnap.html
La actualización del árbol de ports es opcional, ya que freepascal puede ser instalado tanto desde fuentes como utilizando los paquetes precompilados, ambos metodos de instalación los veremos en el siguiente punto.

Instalando Units

Un unit, es una extensión de freepascal, que le provee de lo necesario para trabajar, por ejemplo: ya sea con alguna base de datos (pgsql, mysql, firebird, oracle, etc.), librería gráfica (png,jpeg, gd, opengl, etc), entre otras.
Para saber de que units contamos en nuestro FreeBSD podemos ejecutar lo siguiente:

```
# cd /usr/ports && make search key="fpc-*" | more
```

Con este comando tendremos una lista de las units disponibles en nuestro árbol de ports.
También tenemos un meta port que nos ayudara a instalar con facilidad las units de freepascal, para utilizarlo ejecutamos.

```
# cd /usr/ports/lang/fpc-units && make install clean clean-depends
```

o usando portupgrade:

```
# portinstall fpc-units
```

Este meta port nos mostrara un menú donde marcaremos aquellas units que deseamos que se instalen en nuestro sistema, también se puede utilizar paquetes pre compilados.
Ver la figura 1.

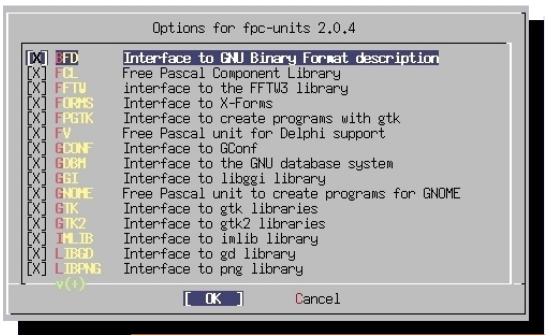


Figura 1 Imagen del fpc-units

Archivos de Configuración

Ya tenemos todo lo necesario para empezar a utilizar freepascal y hacer nuestros primeros programas; solo tenemos que tener en cuenta acerca de los archivos de configuración, tanto del compilador, como del ide.
El archivo de configuración del compilador lo encontraremos en:
ee /usr/local/etc/fpc.cfg

Ahí se definen rutas hacia los units, rutas a las librerías que usaremos según la aplicación que desarrollemos, opciones que se le pasan al compilador para cuando compilemos alguna aplicación, entre otra muchas.
Para ver algunos parámetros que se le pueden especificar al compilar podemos ver la ayuda desplegada cuando ejecutamos:
fpc

El archivo de configuración del ide en modo texto, es almacenado en el directorio home de cada usuario:
ee ~/fp.dsk

Este archivo es generado por el propio ide en modo texto.

Instalando Lazarus

Para instalar lazarus el procedimiento es tan sencillo como lo anteriormente explicado, instalamos lazarus usando nuestro árbol de ports
cd /usr/ports/editors/lazarus && make install clean clean-depends
o usando sysutils/portupgrade
portinstall lazarus
Si queremos usar un paquete pre compilado
pkg_add -r lazarus

Ejecutando el Text-IDE y Lazarus

Para ejecutar el ide modo texto de freepascal ejecutamos
fp
Ver la figura 2.

Ahora ejecutamos lazarus
lazarus
Ver la figura 3.

Lazarus nos mostrara un mensaje acerca que no puede encontrar las fuentes de FreePascal, para esto solo nos hace falta hacer una copia del source de FreePascal en algún lugar que sea accesible para lazarus.

cp /usr/ports/distfiles/freepascal/fpcbuild-2.0.4.tar.gz ~/
tar xfz fpcbuild-2.0.4.tar.gz
mv fpcbuild-2.0.4 fpc-2.0.4
rm fpcbuild-2.0.4.tar.gz

Ahora solo nos falta indicarle a lazarus que use el directorio que acabamos de renombrar.

Para esto nos vamos al menú de lazarus llamado [opciones de entorno],y en la sección [Directorio de fuentes de FPC] le damos la ruta a las fuentes que acabamos de descomprimir ~/fpc-2.0.4/fpcsrc, también no olvidemos darle la ruta al [make path] hacia /usr/local/bin/gmake.

NOTAS FINALES

Es recomendable instalar la documentación de FreePascal, una muy buena y completa documentación.
cd /usr/ports/lang/fpc-docs && make install clean clean-depends

El formato de los archivos es pdf; así que utilicen el visor de pdf's que mas les guste.

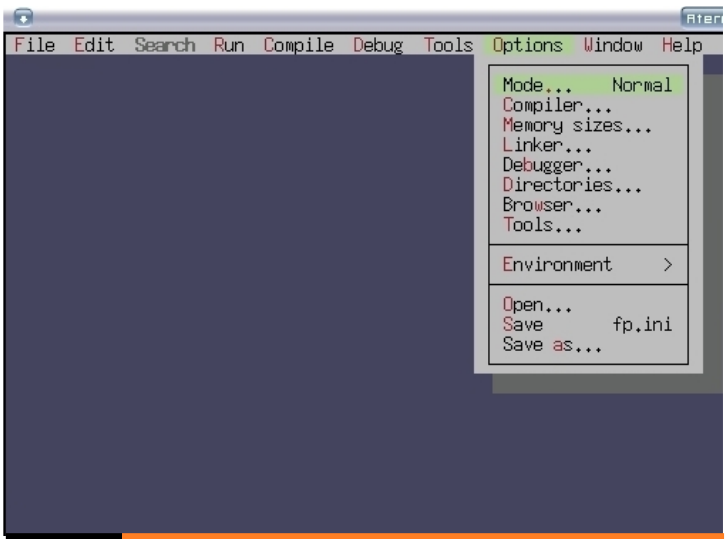


Figura 2 Entorno de programación de FreePascal

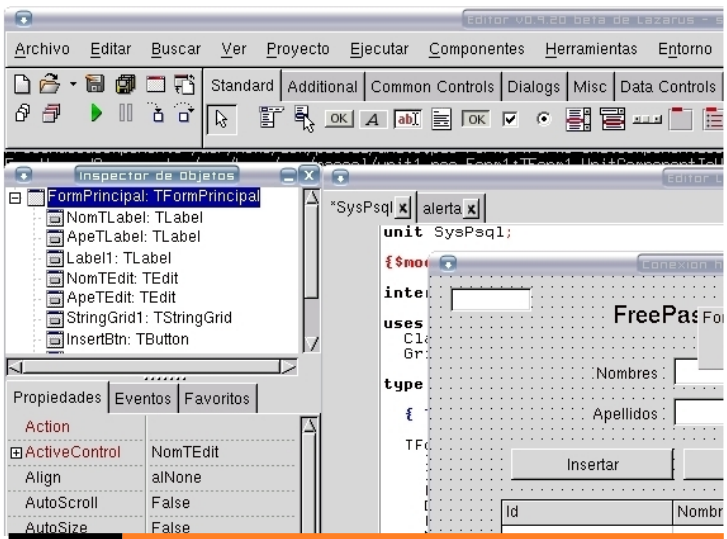


Figura 3 Entorno de Lazarus

ENLACES DE INTERÉS

FreeBSD:
http://www.freebsd.org

FreePascal:
http://www.freepascal.org
http://www.freepascal.org/wiki/index.php/Main_Page

Lazarus:
http://lazarus.freepascal.org
http://wiki.lazarus.freepascal.org/index.php/Main_Page

FLISOL 2007

28 de abril del

www.installfest.info

Conexión a Postgresql en FreePascal

Escrito por: Alonso Cárdenas

Veremos lo sencillo que es el poder conectarnos a nuestra base de datos PostgreSQL usando Freepascal y Lazarus, y para esto vamos a desarrollar una sencilla aplicación.

Iniciando Postgresql

Una vez instalado PostgreSQL crearemos una base de datos y una tabla donde realizaremos nuestras pruebas. Para esto necesitamos tener PostgreSQL ejecutándose:
su postgres o también # su pgsql
Este dependerá del sistema operativo que estemos usando (en nuestro caso GNU/Linux o *BSD).

Creando la BD

Para crea una base de datos, ejecutamos lo siguiente:
createdb freepascal

Luego la tabla que usaremos.
psql template1
Bienvenido a psql 8.0.10, el terminal interactivo de PostgreSQL.

Digite: \copyright para ver los términos de distribución
\h para obtener ayuda sobre comandos SQL
\? para obtener ayuda sobre comandos internos
\g o punto y coma (;) para ejecutar consulta
\q para salir

```
template1=# \c freepascal
freepascal=# CREATE TABLE test (id serial, nombres varchar(30), apellidos varchar(30));
freepascal=# INSERT INTO test (DEFAULT,'Jose Alonso','Cardenas Marquez');
freepascal=# INSERT INTO test (DEFAULT,'Jose Antonio','Rodriguez Segura');
freepascal# \q
# exit
```

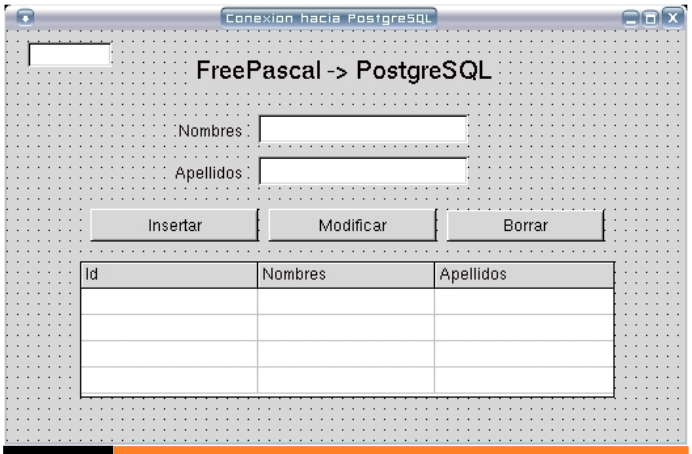


Figura 1 Formulario creado en Lazarus

Desarrollando la aplicación

Primero creamos un nuevo formulario, para esto, ubicamos en el menú de Lazarus y elegimos "Archivo/Nuevo formulario". Ahora empezaremos a crear el contenido que utilizaremos para nuestras pruebas. (Ver la figura 1)

Esto constará de algunos TLabel, TButton, TEdit y un TStringGrid, todos estos controles podemos encontrarlos en la barra de controles, seccion "Standard" y "Additional".

No olvidemos ver las propiedades de cada control, que usaremos en nuestra aplicación, como son "Caption" y "Name"; para editar nuestra grilla: nos posicionamos en ella, seleccionamos la opción "Edit StringGrid".

A continuación asignaremos algunos nombres a los controles.

Id (Caja de texto de la esquina superior izquierda)
Name: idTEdit
Visibility: False
Nombres
Name: NomTEdit
Apellidos
Name: ApeTEdit
Formulario
Name: TestForm
Grilla
Name: DatStringGrid

Una vez terminado de diseñar nuestro formulario empezaremos a programar; para esto situamos en el editor de código de Lazarus. (Ver figura 2)

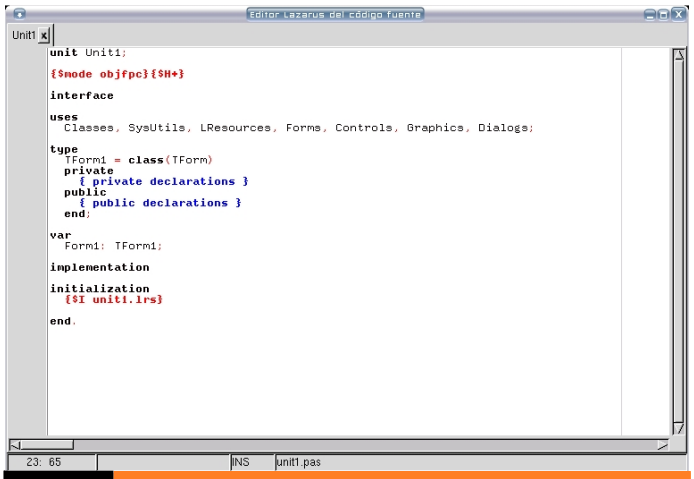


Figura 2 Editor de código del Lazarus

A continuación asignaremos algunos nombres a los controles. En la sección "uses" agregaremos postgres para indicarle al compilador que usaremos la unit de postgres en nuestra aplicación:

```
uses
  Classes, SysUtils, LResources, Forms, Controls, Graphics, Dialogs,
postgres;
```

En la seccion "var" declararemos algunas variables que utilizaremos:

```
var
  login,pwd,pghost,pgport,pgoptions,pgtty,dbname : Pchar;
  nRows,i,c : longint;
  conn : PPGConn;
  res : PPGresult;
```

Crearemos 3 procedimientos que llamaremos de nuestra aplicación para iniciar una conexión, mostrar los datos en nuestro TStringGrid y vaciar los TEdit que usamos en nuestro formulario. Para eso vamos a la sección "implementation" y agregamos nuestros procedimientos:

```
//Primer Procedimiento
procedure pgsqlEmptyText();
begin
    { Vaciamos los Tedit nombres, apellidos y el id oculto }
    TestForm.NomTEdit.Text := '';
    TestForm.ApeTEdit.Text := '';
    TestForm.IdTEdit.Text := '';
end;
```

```
//Segundo procedimiento
procedure pgsqlConection();
begin
    { host de nuestra base de datos }
    pghost := 'localhost';
    { Puerto de conexión a la base de datos }
    pgport := NiL;
    { Alguna opción extra de conexión }
    pgoptions := NiL;
    { Tipo de terminal }
    pgtty := NiL;
    { Nombre de la base de datos }
    dbname := 'freepascal';
    { Usuario de conexión para la base de datos }
    login := 'pgsql';
    { Clave de conexión para la base de datos }
    pwd := '';
    { Abrimos una conexión a nuestra base de datos con los
datos anteriores }
    conn := PQsetdbLogin(pghost, pgport, pgoptions, pgtty,
dbName, login, pwd);
end;
```

```
//Tercer procedimiento
procedure pgsqlFillGrid(conn : PPGConn);
var
  SQL : AnsiString;
begin
    { Armamos nuestra sentencia SQL }
    SQL := 'SELECT * FROM test ORDER by id ASC';
    { Realizamos una consulta a la tabla }
    PQexec(conn, PChar(SQL));
    { Capturamos el resultado }
    nroRows :=  PQntuples(res);
    { Generamos tantas filas en la grilla como filas tenga el
resultado del query }
    TestForm.DatStringGrid.RowCount := nRows+1;
    { Recorremos el resultado y vamos poniendo los datos en la
grilla }
    for i := 0 to nRows-1 do
    begin
        c := i+1;
        TestForm.DatStringGrid.Cells[0,c] := PQgetvalue(res, i, 0);
        TestForm.DatStringGrid.Cells[1,c] := PQgetvalue(res, i, 1);
        TestForm.DatStringGrid.Cells[2,c] := PQgetvalue(res, i, 2);
    end;
    { Limpiamos el resultado }
    PQclear(res);
end;
```

Con esto tenemos listos algunos procedimientos extra, que utilizaremos en nuestra aplicación; ahora asignaremos algunos procedimientos a eventos de nuestro formulario, botones y grilla. Empezaremos por los botones de Insertar, Modificar y Borrar.

Seleccionamos el boton de "Insertar" y en el "Inpector de Objetos/Eventos" seleccionamos el evento OnClick, llenamos el campo que contendra el nombre de la función que llamara cuando hagamos un click a dicho boton en nuestro caso llamaremos a la función "InsertBtnClick" y presionamos enter.

Automaticamente sera llamado el "Editor de codigo de Lazarus" referenciando a dicha función. (Continua en la siguiente hoja)

El contenido de la función InsertBtnClick será el siguiente:

```
procedure TTestForm.InsertBtnClick(Sender: TObject);
var
  SQL : AnsiString;
begin
    { Llamamos a la función de conexión }
    pgsqlConection();
    { Armamos nuestro sentencia SQL, para insertar un nuevo
      registro con los datos de NomTEdit y ApeTEdit }
    SQL := 'INSERT INTO test VALUES(DEFAULT,''' + NomTEdit.Text +
    ''','' + ApeTEdit.Text + ''')';
    { Ejecutamos el SQL }
    PQexec(conn, PChar(SQL));
    { Llenamos la grilla con los datos de la tabla }
    pgsqlFillGrid(conn);
    { Finalizamos la conexión }
    PQfinish(conn);
    { Limpiamos las cajas TEdit }
    pgsqlEmptyText();
end;.
```

Repetimos los pasos anteriores para el botón "Modificar" y "Borrar", asignándole como funciones al evento OnClick "EditBtnClick" y "DeleteBtnClick" respectivamente, y con el siguiente contenido:

```
procedure TTestForm.EditBtnClick(Sender: TObject);
var
  SQL : AnsiString;
begin
    { Llamamos a la funcion de conexión }
    pgsqlConection();
    { Armamos nuestra sentencia SQL, para actualizar un registro
      segun los datos de IdTEdit, NomEdit y ApeTEdit }
    SQL := 'UPDATE test SET nombre='''+ NomTEdit.Text+''',
      apellidos='''+ApeTEdit.Text+''' WHERE id=' + IdTEdit.Text;
    { Ejecutamos la sentencia }
    PQexec(conn, PChar(SQL));
    { Llenamos la grilla con los datos de la tabla }
    pgsqlFillGrid(conn);
    { Cerramos la conexion a la base de datos }
    PQfinish(conn);
    { Limpiamos las cajas TEdit }
    pgsqlEmptyText();
end;

procedure TTestForm.DeleteBtnClick(Sender: TObject);
var
  SQL : AnsiString;
begin
    { Llamamos a la funcióon de conexión }
    pgsqlConection();
    { Armamos nuestra sentencia SQL, para eliminar un registro
      segun el dato del IdTEdit }
    SQL := 'DELETE FROM test WHERE id=' + IdTEdit.Text;
    { Ejecutamos la sentencia }
    PQexec(conn, PChar(SQL));
    { Llenamos la grilla con los datos de la tabla }
    pgsqlFillGrid(conn);
    { Cerramos la conexión a la base de datos }
    PQfinish(conn);
end;
```

Asignamos a nuestro "TStringGrid" la función "GridSelectCell" en el evento "OnSelectCell"

```
procedure TTestForm.GridSelectCell(Sender: TObject; Col,
  Row: Integer; var CanSelect: Boolean);
begin
    { Limpiamos las cajas TEdit }
    pgsqlEmptyText();
    { Capturamos el número de fila seleccionada en la grilla y
      llenamos los Tedit }
    TestForm.NomTEdit.Text := TestForm.DatStringGrid.Cells[1,Row];
    TestForm.ApeTEdit.Text := TestForm.DatStringGrid.Cells[2,Row];
    TestForm.IdTEdit.Text := TestForm.DatStringGrid.Cells[0,Row];;
end;
```

Y por último asignaremos una función a un evento de nuestro formulario, seleccionamos nuestro formulario y asignamos al evento "OnCreate" la funcion "FormCreate" con el siguiente contenido:

```
procedure TTestForm.FormCreate(Sender: TObject);
begin
    { Llamamos a la función de conexión }
    pgsqlConection();
    { Llenamos la grilla con los datos de la tabla test }
    pgsqlFillGrid(conn);
    { Cerramos la conexión a la base de datos }
    PQfinish(conn);
end;
```

Ahora solo nos queda grabar los cambios y compilar nuestra aplicacion, para esto presionamos F9, si todo los pasos anteriores han sido seguidos correctamente miraremos nuestra aplicación ejecutándose. (Ver figura 3)

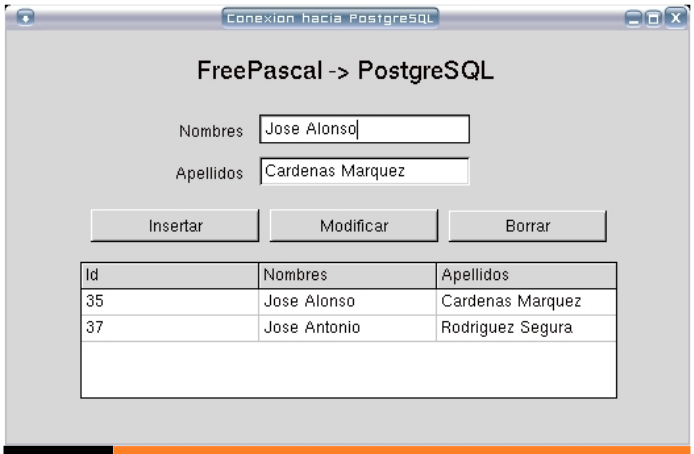


Figura 3 Resultado de nuestra aplicación

ENLACES DE INTERÉS

FreePascal:
http://www.freepascal.org
http://www.freepascal.org/wiki/index.php/Main_Page

Lazarus:
http://lazarus.freepascal.org
http://wiki.lazarus.freepascal.org/index.php/Main_Page

Gestor de Iconos de Escritorio

IconMgr

Escrito por: Luis Revilla Amézquita

El proyecto IconMgr que actualmente esta en desarrollo tiene por finalidad añadir iconos de escritorio en gestores de ventanas (window managers) que no soporten esta función (fluxbox, icewm, fuwm2, etc). Además, gestionar el fondo de pantalla de nuestro escritorio.

Historia

La idea nació debido alas ciertas carencias que tiene el Idesk http://idesk.sourceforge.net/ que es un programa que tiene la misma función que el IconMgr que permite gestionar los iconos de escritorio y el fondo de pantalla.

Ademas el Idesk esta casi abandonado su ultima versión (0.7.5) y modificación data de noviembre del 2005 según su pagina web.

Desarrollo y situación actual del proyecto

Actualmente ya tiene casi todas las funcionales que contiene el Idesk y se ha agregado otras características tales como:

- Colocación de los Iconos de escritorio.
- El texto que nos indica el nombre de la aplicación (Caption) y información adicional del programa a lanzar (ToolTip).
- Fondo de pantalla.
- Archivos de configuración.
- Colocación de temas (themes) como fondo para el texto que se coloca tanto en el caption y tooltip.
- Posibilidad de cambiar las fuentes y color del texto.
- El lanzamiento o ejecución de la aplicación respectiva para cada icono de escritorio.

Versiones disponibles

Actualmente el proyecto esta en fase de desarrollo aun no esta disponible el código para su uso , pero muy pronto ya se tendrá una versión Beta.

Actualmente el IconMgr esta alojado en la WIKI[1] del AQPGLUG.
Donde se puede encontrar información sobre su desarrollo actual y algunas capturas y videos realizados.

Características Técnicas

IconMgr esta escrito en el lenguaje de programación C.

Dependencias

- Xorg devel
- Imlib2

Instalación

Una vez obtenido el código se aplican los pasos típicos de una compilación:

```
./configure
make
make install
```

Ejecutar IconMgr que esta ubicado en /usr/local/bin/IconMgr

Continúa >>

[1] http://www.aqpglug.org.pe/wiki/doku.php?id=start:proyectos:iconmgriconmgrlnk
http://iconmgr.sourceforge.net/

En el momento de la instalación crea la siguiente estructura de directorios en la carpeta del usuario:

```
$HOME/.IconMgr/
$HOME/.IconMgr/Background/              --> para imágenes de fondo de pantalla
$HOME/.IconMgr/Config/Background    --> archivo de configuración usado para
                                            el fondo de pantalla, aquí se puede modificar
                                            para indicar que imagen se va utilizar como
                                            fondo.
$HOME/.IconMgr/Config/IconMgrConf  --> archivo de configuración.
$HOME/.IconMgr/Links/                   --> ubicación de los archivos *.lnk, que
                                            representan los iconos de escritorio
$HOME/.IconMgr/Themes/Caption/      --> temas
$HOME/.IconMgr/Themes/ToolTip/       --> temas
$HOME/.IconMgr/icons/                   --> iconos personalizados
$HOME/.IconMgr/ttfonts/                 -->  fuentes truetype personalizadas
```

La instalación por defecto se realiza en:

```
/usr/local/share/IconMgr/          --> tiene la misma estructura de los sub directorios
                                       que en la instalación del $HOME del usuario.
```

Archivos de Configuración

Estructura de los archivos lnk: Representan a cada icono del escritorio:

```
table Icon
  Caption: Consola
  ToolTip.Caption: Terminal
  Command: /usr/bin/xterm
  Icon: /usr/local/share/IconMgr/icons/terminal.png
  Width: 48
  Height: 48
  X: 10
  Y: 70
end
```

Estructura del archivo de configuración del fondo de pantalla:

```
table background
   imagen:
/usr/local/share/IconMgr/Background/fondo.jpg
end
```

Al ejecutar IconMgr, que es el binario obtenido de la compilación, verifica la presencia en el $HOME del usuario, la existencia del directorio ./IconMgr; si no existiera, se realiza la búsqueda en /usr/local/share/IconMgr.

Una vez encontrado el directorio ./IconMgr verifica la presencia de los archivos de configuración del fondo de pantalla y de los iconos de escritorios ubicados respectivamente en:

Para el fondo de pantalla:
$HOME/.IconMgr/Config/Background

Para cada link que agregamos al escritorio:
$HOME/.IconMgr/Links/

Cada link esta definido en un archivo:
.gaim.lnk
.xv.lnk
.xterm.lnk

¿Cuando comenzo el proyecto?

En enero del 2007 se propuso la idea de desarrollar un programa que tenga la misma función que el conocido Idesk.
Lo que se quería era desarrolar un gestor de iconos de escritorio, que consuma menos recursos del sistema, hemos logrado este primer objetivo, obteniendo un consumo mucho menor comparado con el Idesk.

Próximas mejoras

Mejor soporte en el tema de los tipos de fuentes. El uso del estandar XML (XML es un lenguaje de marcas), para el uso de los archivos de configuracion. Ademas de algunos efectos visuales.

Desarrolladores

Actualmente solo hay 2 desarrolladores:
Luis Revilla A. Encargado del desarrollo del IconMgr.
mail: lrevilla@aqpglug.org.pe

Abraham Montaño Encargado del desarrollo la aplicación gráfica de configuración llamada IconMgr-LNK.
mail: abraham.montano@gmail.com

CONSOL

Trujillo – Perú

Próximamente...

Octubre – 2007