ELABORACIÓN DE LINEAMIENTOS PARA EL DESARROLLO DE SOFTWARE CON HERRAMIENTAS GNU EN LINUX

CHAMORRO HERNÁNDEZ MARIA

CUESTA MONTES INDIRA

GARIZABALO VISBAL ANTONIO

OLMOS NAVARRO YAIR

UNIVERSIDAD SIMON BOLIVAR

FACULTAD DE INGENIERIA DE SISTEMAS

AREA DE FORMATIVA IV

BARRANQUILA – ATLÁNTICO

2006

CONTENIDO

INTRODUCCION

En la actualidad, los sistemas de información basados en computadoras son de gran importancia en las actividades cotidianas y en la toma de decisiones en cualquier clase de negoció, ocupando así un lugar especial en el área que se implante, ya que hoy en día la tecnología es utilizada de muchas formas directa o indirectamente para trabajar con mayor facilidad creando un servicio mas eficiente.

Es importante reconocer que un software ó un sistema de información es realizado bajo unos lineamientos lógicos con la documentación correspondiente que ayude a facilitar el manejo del lenguaje de programación y la información necesaria de cualquier sistema es almacenada en una base de datos, donde la combinación de estos dos componentes es esencial para que un sistema de información funcione adecuadamente.

De tal forma los sistemas informáticos se elaboran de acuerdo al Sistema Operativo en donde se implementen ya sea en el Sistema Operativo de Microsoft ó al Sistema Operativo de Linux de tecnología GNU. Existen infinidad de herramientas que permiten el desarrollo de software, reconociendo también que en el mercado se manejan las herramientas GNU para el desarrollo de software libre.

Las herramientas que más predominan en el mercado para el desarrollo de software GNU han incluido las versiones modificadas como son el caso de Gambas y Eclipse, que hará parte esencial del contenido de esta investigación plasmado en una documentación. El propósito es ayudar al aprendizaje de estas herramientas GNU, para el desarrollo de software libre en los estudiantes de Ingeniería de Sistemas de la Universidad Simón Bolívar.

1. PLANTEAMIENTO DEL PROBLEMA

1.1 DESCRIPCION DEL PROBLEMA

A través del estudio que se les ha realizado a los estudiantes de la Universidad Simón Bolívar en el programa de Ingeniería de Sistemas se ha logrado establecer que la formación académica de estos discentes se basa en el estudio y manejo de herramientas comerciales bajo licencias para el desarrollo software.

Estas circunstancias conlleva a que los estudiantes de Ingeniería de Sistemas no logren tener una visión completa de otras herramientas que se encuentran implementadas en el mercado y que esta situación del pensamiento de los discentes no permite crear un nuevo paradigma para el desarrollo de software de manera fácil y gratuita.

1.2 FORMULACION DEL PROBLEMA

¿De qué manera contribuiría en la formación profesional de los estudiantes de Ingeniería de Sistemas un cambio del paradigma en el desarrollo de software?

1.3 SISTEMATIZACION DEL PROBLEMA

- ¿Cómo se aplicaría este nuevo paradigma en el desarrollo de software?

- ¿Cómo se podrá implementar las reglas necesarias para el desarrollo de software libre?

- ¿De qué manera los lineamientos ayudarán a afianzar los conocimientos de Herramientas GNU?

- ¿Qué metodología se podría utilizar para llevar a cabo el desarrollo de software?

- ¿Cómo motivar a los estudiantes de la Universidad Simón Bolívar para impulsar a la utilización de las herramientas GNU/Linux?

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Afianzar el conocimiento y el manejo de herramientas GNU/Linux en los estudiantes de Ingeniería de Sistemas de la Universidad Simón Bolívar a través de los lineamientos para el desarrollo de software libre con el fin de optimizar el desempeño de los estudiantes.

2.2 OBJETIVOS ESPECIFICOS

- Elaborar la documentación de herramientas GNU para el desarrollo de software libre.
- Analizar los lineamientos que permita el manejo y el desarrollo de software libre con las herramientas GNU de Gambas y Eclipse.
- Conocer los entornos gráficos de las herramientas para el desarrollo de software GNU en Gambas y Eclipse.
- Ilustrar a través de conceptos y ejemplos como una metodología de constructivismo en los estudiantes para el desarrollo de software libre.
- Motivar a los estudiantes de la Universidad Simón Bolívar para que interactúen con las herramientas GNU/Linux.

3. JUSTIFICACIÓN DEL PROYECTO

Después de realizar un estudio detallado a los estudiantes de Ingeniería de Sistemas de la Universidad Simón Bolívar acerca de sus conocimientos del manejo y desarrollo de herramientas comerciales bajo licencias, es necesario realizar esta investigación para guiarlos mediante los lineamientos que ayudan a facilitar el aprendizaje y el manejo de las herramientas GNU/Linux, que permite a los estudiantes tener una visión clara de otros tipos de herramientas bajo las licencias de libre distribución. Esta investigación busca romper con el paradigma de los estudiantes de Ingeniería de Sistemas de la Universidad Simón Bolívar acerca de las herramientas GNU que ofrecen ventajas provechosas para el desarrollo y construcción de software de alta calidad de manera fácil y rápida.

4. MARCO DE REFERENCIA

4.1 MARCO TEORICO

Los lineamientos ayudan a tener una guía clara de los aspectos importantes que se necesitan para aprender. El manejo de los lineamientos son necesarios ya que se definen como el conjunto de pasos, etapas, conductas que pueden ayudar a la consecución de conocimientos ó cualquier deseo de aprender.

UNÍX, al igual que los demás sistemas operacionales, es un programa de control para computadores, pero además suministra un conjunto de programas de utilidad que permiten al usuario efectuar sus propios desarrollos.
UNÍX[1] incluye cientos de programas de utilidad que realizan las funciones más comunes requeridas por los usuarios, desde el manejo de archivos y dispositivos hasta rutinas de calculo y manejo de redes, en un ambiente multiusuario porque permite que varias personas utilicen el computador al mismo tiempo e interactivo además admite la comunicación es directa con el computador desde terminales.

La eficiencia y portabilidad de UNÍX se debe al cumplimiento de estándares básicos en su diseño y a que la mayoría de los módulos están desarrollados en

---

1 - http\\wipipedia\enciclopedia_libre\Linu_GNU\Linux.pdf\html
- WELSH, Matt. Instalación y Primeros Pasos: Linux.pdf (Traducción: Proyecto Lucas, versión en castellano) Copyright c 1992-1996. p, 19.

el lenguaje de programación C, que es un lenguaje de alto nivel con una alta independencia de la máquina.

La productividad de UNÍX se basa en su orientación a administrar los recursos del computador en un ambiente multi-tareas.

UNÍX[2] utiliza un Shell para el procesar los requerimientos de los usuarios. El Shell es una utilidad que interpreta el comando dado por el usuario e invoca el programa respectivo para su ejecución. El Shell opera como una interfaz de usuario y puede ser distinta para cada usuario en un sistema dado.

UNÍX cuenta un sistema de archivos que le permiten manejar una gran cantidad de archivos de manera eficiente. Utiliza una estructura jerárquica de árbol, en la que cada usuario tiene un directorio de trabajo con la cantidad de subdirectorio que desee, y la posibilidad de compartir archivos con otros usuarios.

UNIX maneja criterios de seguridad para controlar el acceso a los sistemas y sus recursos, incluyendo los archivos, directorios y unidades. Los dispositivos son tratados como archivos en UNÍX lo cual permite el re-direccionamiento de la salida en los procesos y su condición como entrada a otro proceso. Tanto la entrada como la salida son independientes del dispositivo.

---

2 - http\\wipipedia\enciclopedia_libre\Linu_GNU\Linux.pdf\html
- WELSH, Matt. Instalación y Primeros Pasos: Linux.pdf (Traducción: Proyecto Lucas, versión en castellano) Copyright c 1992-1996. p, 23.

Los sistemas UNÍX se caracterizan por combinar de forma armónica estos detalles. Son utilizados actualmente en la mayoría de las organizaciones públicas y privadas que necesitan comunicarse de la mejor: UNÍX, es probablemente el sistema operativo más versátil.

La Free Software Foundation (Fundación de Software Libre, FSF) y en esta el proyecto GNU como una forma de recuperar el espíritu cooperativo de los primeros días de la computación y posibilitar nuevamente la cooperación sacando los obstáculos impuestos por los dueños del software propietario.

El proyecto GNU[3] consiste en el desarrollo de sistemas y un juego de aplicaciones totalmente libre y compatible con UNÍX. El proyecto incluye desarrollar una versión libre de cualquier aplicación que no se disponga libre. De esta forma, una computadora puede estar equipada con un 100% de software libre; esto incluye el sistema operativo y todos los programas que se necesiten para realizar cualquier tarea, ya que sin un sistema operativo no puede utilizarse una computadora, se tomo esto como un punto de partida para el proyecto GNU.

Linux[4] es, a simple vista un sistema operativo, es una implementación de libre distribución UNÍX para computadoras personales (PC), servidores, y estaciones

---

3 http\\monografia.com\GNU.html

4 - http\\wipipedia\enciclopedia_libre\Linu_GNU\Linux.pdf\html
- WELSH, Matt. Instalación y Primeros Pasos: Linux.pdf (Traducción: Proyecto Lucas,

de trabajo. Fue desarrollado para i386 y soporta los Procesadores i486, Pentium, Pentium pro y Pentium II, así como los clones AMD y Cyrix. También soporta maquinas basadas en SPARC, DEC Alpha, PowerPC/PowerMac y Mac/Amiga Motorola 68x0.

Como sistema operativo, Linux es muy eficiente y tiene un excelente diseño. Es multitarea, multiusuario, multiplataforma y multiprocesador, en plataformas Intel corre en modo protegido, protege la memoria para que un programa no pueda hacer caer al resto del sistema, carga solo las partes de un programa que se usan, comparte la memoria entre programa aumentando la velocidad y disminuyendo el uso de memoria, usan un sistema de memoria virtual por paginas, utiliza toda la memoria libre para la caché, permite usar bibliotecas enlazadas tanto estática como dinámicamente, se distribuye con código fuente, usa hasta 64 consolas virtuales, tiene un sistema de archivos avanzado pero puede usar los de los otros sistemas y soporta redes tanto en TCP/IP como en otros protocolos.

Su objetivo inicial es propulsar el software de libre distribución junto con su código fuente para que pueda ser modificado por cualquier persona, dando rienda suelta a la creatividad. El hecho de que el sistema operativo incluya su propio código fuente expande enormemente las posibilidades de este sistema. Este método también es aplicado en numerosas ocasiones a los programas que corren en el sistema, lo que hace que podamos encontrar muchísimos programas útiles totalmente gratuitos y con su código fuente.

4.2 MARCO CONCEPTUAL

Características de Linux

Linux es un sistema operativo completo con multitarea y multiusuario, es decir, que pueden trabajar varios usuarios simultáneamente en él, y que cada uno de ellos puede tener varios programas en ejecución.

El sistema Linux es compatible con ciertos estándares de UNIX a nivel de código fuente, incluyendo el IEEE POSIX.1, System V y BSD. Fue desarrollado buscando la portabilidad de las fuentes: encontrará que casi todo el software gratuito desarrollado para UNIX se compila en Linux sin problemas. Y todo lo que se hace para Linux (código del núcleo, drivers, librerías y programas de usuario) es de libre distribución.

El núcleo de Linux es capaz de emular por su cuenta las instrucciones del procesador, soporta diversos sistemas de ficheros para guardar los datos.

Linux[5] implementa todo lo necesario para trabajar en red con TCP/IP. Desde administradores para las tarjetas de red más populares hasta SLIP/PPP, que permiten acceder a una red TCP/IP por el puerto serie. También se implementan PLIP (para comunicarse por el puerto de la impresora) y NFS (para acceso remoto a ficheros). Y también se han portado los clientes de TCP/IP, como FTP, telnet, NNTP y SMTP.

---

5 - http\\wipipedia\enciclopedia_libre\Linu_GNU\Linux.pdf\html
- WELSH, Matt. Instalación y Primeros Pasos: Linux.pdf (Traducción: Proyecto Lucas, versión en castellano) Copyright c 1992-1996. p, 11.

El núcleo soporta ejecutables con paginación por demanda. Esto significa que sólo los segmentos del programa que se necesitan se cargan en memoria desde el disco. Las páginas de los ejecutables son compartidas mediante la técnica copy-on-write, contribuyendo todo ello a reducir la cantidad de memoria requerida para las aplicaciones.

Con el fin de incrementar la memoria disponible, Linux implementa la paginación con el disco: puede tener hasta 256 megabytes de espacio de intercambio o "swap" en el disco duro. Cuando el sistema necesita más memoria, expulsará páginas inactivas al disco, permitiendo la ejecución de programas más grandes o aumentando el número de usuarios que puede atender a la vez. Sin embargo, el espacio de intercambio no puede suplir totalmente a la memoria RAM, ya que el primero es mucho más lento que ésta.

La memoria dedicada a los programas y a la caché de disco está unificada. Por ello, si en cierto momento hay mucha memoria libre, el tamaño de la caché de disco aumentará acelerando así los accesos.

Los ejecutables hacen uso de las librerías de enlace dinámico. Esto significa que los ejecutables comparten el código común de las librerías en un único fichero. También pueden enlazarse estáticamente cuando se deseen ejecutables que no requieran la presencia de las librerías dinámicas en el sistema. El enlace dinámico se hace en tiempo de ejecución, con lo que el programador puede cambiar las librerías sin necesidad de recopilación de los ejecutables.

Herramientas GNU

Estas son algunas herramientas para la elaboración de software.

Gambas, es una herramienta de desarrollo visual de aplicaciones muy similar a los conocidos programas comerciales Microsoft Visual Basic o Borland Delphi[6].

Con Gambas se pueden hacer aplicaciones o programas con interfaz gráfica de forma muy rápida, pues integran un diseñador de formularios o ventanas, un editor de código, un explorador de clases, un visor de ayuda, etc.
Este tipo de herramientas han sido siempre muy habituales en la plataforma Microsoft Windows, pero para Linux no existían tantas, o bien no estaban tan depuradas. Podemos encontrar Kdevelop, Kylix o VDK Builder.

Hay que destacar que en el desarrollo de aplicaciones en Linux hay una larga tradición y costumbre de emplear muchas herramientas diferentes, cada una especializada en una tarea en concreto (p. Ej., un compilador, un editor, un depurador, cada uno por separado), por lo que este tipo de herramientas integradas (IDE) no han aparecido hasta hace poco.

Eclipse es una poderosa herramienta que permite integrar diferentes aplicaciones para construir un entorno integrado de desarrollo[7].

---

6 - http://gambas.gnulinex.org/gambas_tutorial.pdf. P, 1.
- http:// gambas.sourceforge.net/

7 - www.eclipse.org\Desarrollo de Plug-ins.pdf\\. P, 1.

Es un proyecto de desarrollo de software opensource, que está dividido en tres partes: el proyecto Eclipse Proyect, Eclipse Tools, y Eclipse Technology Proyect. Mediante Eclipse se puede crear diversas aplicaciones como ser sitios Web, programas Java, C++ y Enterprise Java Beans. Su principal aplicación es JDT, herramienta para crear aplicaciones en Java. Otras aplicaciones pueden ser integradas a Eclipse en forma de plug-ins, que son reconocidos automáticamente al iniciar el mismo. Como Eclipse está escrito en Java, para su funcionamiento se debe tener instalado el JRE (Java Runtime Enviroment). Eclipse detecta automáticamente la ubicación de la JRE instalada. Las funcionalidades que otorga Eclipse se localizan de dos formas diferentes: en un pequeño núcleo conocido como el Plataforma Runtime o en forma de plug-ins. La plataforma Eclipse esta construida en base a plug-ins.

4.3 MARCO LEGAL

Según el artículo 99 de la ley de propiedad intelectual indica: "sección de uso, salvar a prueba de contras, de intransferible y no exclusiva", es decir, no eres propietario del programa, sólo se tiene derecho a usarlo en un ordenador o tantos como permita expresamente la licencia y no puede ser modificado el programa aunque se realicen copia privada del programa, no se podrá distribuir públicamente.

La ley no dice expresamente nada sobre la aplicación del derecho de comunicación pública a los programas de ordenador mientras que el artículo

20.2 de dicha ley considera la comunicación pública del acceso a las bases de datos cuando estas incorporen o constituyan obras protegidas.

La licencia **GPL** o General Public License, desarrollada por la free software foundation, del cual se pueden instalar y usar un programa GPL en un ordenador sin limitación alguna, podrá hacerse gratuitamente, siempre y cuando respete las 3 libertades, de esta forma se entiende que no se cobre por el programa en si, sino por los servicios que esto implica:

La primera libertad es la de usar el programa.
La segunda la de poder modificar el programa.
La tercera la de distribuir el programa modificado o no.
La licencia GPL obliga a incluir el código fuente en su distribución, siendo imposibles cambiar la licencia al programa, al distribuirlo.

Preámbulo de las licencias GNU

Las licencias que cubren la mayor parte del software están diseñadas para quitarles a los usuarios la libertad de compartirlo y modificarlo. Por el contrario, la Licencia Pública General de GNU pretende garantizar la libertad de compartir y modificar software libre, para asegurar que el software es libre para todos sus usuarios. Esta Licencia Pública General se aplica a la mayor parte del software del la Free Software Foundation y a cualquier otro programa si sus autores se comprometen a utilizarla. (Existe otro software de la Free Software Foundation que está cubierto por la Licencia Pública General de GNU para Bibliotecas).

Cuando hablamos de software libre, estamos refiriéndonos a libertad, no a precio. Licencias Públicas Generales están diseñadas para asegurarnos de que tenga la libertad de distribuir copias de software libre (y cobrar por ese servicio si quiere), con el código fuente y realizar modificaciones al software o usar fragmentos de él en nuevos programas libres.

Para proteger los derechos de los usuarios se necesitan algunas restricciones que prohíban a cualquiera que obtenga software libre negarle a los usuarios la posibilidad de tenerlos. Estas restricciones se traducen en ciertas obligaciones que le afectan si distribuye copias del software, o si lo modifica.

Por ejemplo, si distribuye copias de algún programa, sea gratuitamente, o a cambio de una contraprestación, debe dar a los receptores todos los derechos que tiene.

Para el software libre se protegen sus derechos con la combinación de dos medidas:

1. Se coloca el software libre bajo copyright
2. Se ofrece la licencia, que le da permiso legal para copiar, distribuir y/o modificar el software.

También, para la protección de cada autor y se debe tener en cuenta que aunque se trata de software libre no se proporcione ninguna garantía de este. Si el software es modificado y se distribuye, los receptores deben saber que lo que se tiene no es el original, de forma que cualquier problema introducido por otros no afecte a la reputación de los autores originales.

Por último, cualquier programa libre está constantemente amenazado por patentes sobre el software. Se debe evitar el peligro de que los redistribuidores de un programa libre obtengan patentes por su cuenta, convirtiendo de facto el programa en propietario. Para evitar esto, se debe saber que cualquier patente será utilizada para el uso libre únicamente.

Los términos exactos y las condiciones para la copia, distribución y modificación del software se exponen a continuación.

Términos y condiciones para la copia, distribución y modificación

1. La licencia se aplica a cualquier programa u otro tipo de trabajo que contenga una nota colocada por el poseedor del copyright diciendo que puede ser distribuido bajo los términos de esta Licencia Pública General. En adelante, «Programa» se referirá a cualquier programa o trabajo que cumpla esa condición y «trabajo basado en el Programa» se referirá bien al Programa o a cualquier trabajo derivado de él según la ley de copyright. Esto es, un trabajo que contenga el programa o una parte de él, lo cual implica las modificaciones y/o traducido en otro lenguaje. Por lo tanto, la traducción está incluida sin limitaciones en el término «modificación». Cada concesionario (licenciatario) será denominado «usted».

   Cualquier otra actividad que no sea la copia, distribución o modificación no está cubierta por esta Licencia, está fuera de su ámbito. El acto de ejecutar el Programa no está restringido, y los resultados del Programa

están cubiertos únicamente si sus contenidos constituyen un trabajo basado en el Programa, independientemente de haberlo producido mediante la ejecución del programa. El que esto se cumpla, depende de lo que haga el programa.

2. El usuario puede copiar y distribuir copias literales del código fuente del Programa, según lo has recibido, en cualquier medio, supuesto que de forma adecuada y bien visible publique en cada copia un anuncio de copyright adecuado y un repudio de garantía, mantenga intactos todos los anuncios que se refieran a esta Licencia y a la ausencia de garantía, y proporcione a cualquier otro receptor del programa una copia de esta Licencia junto con el Programa.

   Puede cobrar un precio por el acto físico de transferir una copia, y puede, según su libre albedrío, ofrecer garantía a cambio de unos honorarios.

3. Pueden modificar su copia o copias del Programa o de cualquier porción de él, formando de esta manera un trabajo basado en el Programa, y copiar y distribuir esa modificación o trabajo bajo los términos y condiciones del apartado 1, anteriormente expuesto, agregando además que cumpla con las siguientes condiciones:

   a. Debe hacer que los ficheros modificados lleven anuncios prominentes indicando que los ha cambiado y la fecha de cualquier cambio.

b. Debe hacer que cualquier trabajo que distribuya o publique y que en todo o en parte contenga o sea derivado del Programa o de cualquier parte de él sea licenciada como un todo, sin carga alguna, a todas las terceras partes y bajo los términos de esta Licencia.

c. Si el programa modificado lee normalmente órdenes interactivamente cuando es ejecutado, debe hacer que, cuando comience su ejecución para ese uso interactivo de la forma más habitual, muestre o escriba un mensaje que incluya un anuncio de copyright y un anuncio de que no se ofrece ninguna garantía (o por el contrario que sí se ofrece garantía) y que los usuarios pueden redistribuir el programa bajo estas condiciones, e indicando al usuario cómo ver una copia de esta licencia. (Excepción: si el propio programa es interactivo pero normalmente no muestra ese anuncio, no se requiere que su trabajo basado en el Programa muestre ningún anuncio).

Estos requisitos se aplican al trabajo modificado como un todo. Si partes identificables de ese trabajo no son derivadas del Programa, y pueden, razonablemente, ser consideradas trabajos independientes y separados por ellos mismos, entonces esta Licencia y sus términos no se aplican a esas partes cuando sean distribuidas como trabajos separados. Pero cuando distribuya esas mismas secciones como partes de un todo que es un trabajo basado en el Programa, la distribución del todo debe ser según los términos de esta licencia, cuyos permisos para otros

licenciatarios se extienden al todo completo, y por lo tanto a todas y cada una de sus partes, con independencia de quién la escribió.

Por lo tanto, no es la intención de este apartado reclamar derechos o desafiar sus derechos sobre trabajos escritos totalmente por usted mismo. El intento es ejercer el derecho a controlar la distribución de trabajos derivados o colectivos basados en el Programa.

Además, el simple hecho de reunir un trabajo no basado en el Programa con el Programa (o con un trabajo basado en el Programa) en un volumen de almacenamiento o en un medio de distribución no hace que dicho trabajo entre dentro del ámbito cubierto por esta Licencia.

4. Puede copiar y distribuir el Programa (o un trabajo basado en él, según se especifica en el apartado 2, como código objeto o en formato ejecutable según los términos de los apartados 1 y 2, supuesto que además cumpla una de las siguientes condiciones:
    a. Acompañarlo con el código fuente completo correspondiente, en formato electrónico, que debe ser distribuido según se especifica en los apartados 1 y 2 de esta Licencia en un medio habitualmente utilizado para el intercambio de programas, o
    b. Acompañarlo con una oferta por escrito, válida durante al menos tres años, de proporcionar a cualquier tercera parte una copia completa en formato electrónico del código fuente correspondiente, a un coste no mayor que el de realizar físicamente la distribución del fuente, que será distribuido bajo las

condiciones descritas en los apartados 1 y 2 anteriores, en un medio habitualmente utilizado para el intercambio de programas,

c. Acompañarlo con la información que se recibe ofreciendo distribuir el código fuente correspondiente. (Esta opción se permite sólo para distribución no comercial y sólo si usted recibió el programa como código objeto o en formato ejecutable con tal oferta, de acuerdo con el apartado b anterior).

Por código fuente de un trabajo se entiende la forma preferida del trabajo cuando se le hacen modificaciones. Para un trabajo ejecutable, se entiende por código fuente completo todo el código fuente para todos los módulos que contiene, más cualquier fichero asociado de definición de interfaces, más los guiones utilizados para controlar la compilación e instalación del ejecutable. Como excepción especial el código fuente distribuido no necesita incluir nada que sea distribuido normalmente (bien como fuente, bien en forma binaria) con los componentes principales (compilador, kernel y similares) del sistema operativo en el cual funciona el ejecutable, a no ser que el propio componente acompañe al ejecutable.

Si la distribución del ejecutable o del código objeto se hace mediante la oferta acceso para copiarlo de un cierto lugar, entonces se considera la oferta de acceso para copiar el código fuente del mismo lugar como distribución del código fuente, incluso aunque terceras partes no estén forzadas a copiar el fuente junto con el código objeto.

5. No puede copiar, modificar, sublicenciar o distribuir el Programa excepto como prevé expresamente esta Licencia. Cualquier intento de copiar, modificar sublicenciar o distribuir el Programa de otra forma es inválida, y hará que cesen automáticamente los derechos que te proporciona esta Licencia. En cualquier caso, las partes que hayan recibido copias o derechos de usted bajo esta Licencia no cesarán en sus derechos mientras esas partes continúen cumpliéndola.
6. No está obligado a aceptar esta licencia, ya que no la ha firmado. Sin embargo, no hay hada más que le proporcione permiso para modificar o distribuir el Programa o sus trabajos derivados. Estas acciones están prohibidas por la ley si no acepta esta Licencia. Por lo tanto, si modifica o distribuye el Programa (o cualquier trabajo basado en el Programa), está indicando que acepta esta Licencia para poder hacerlo, y todos sus términos y condiciones para copiar, distribuir o modificar el Programa o trabajos basados en él.
7. Cada vez que redistribuya el Programa (o cualquier trabajo basado en el Programa), el receptor recibe automáticamente una licencia del licenciatario original para copiar, distribuir o modificar el Programa, de forma sujeta a estos términos y condiciones. No puede imponer al receptor ninguna restricción más sobre el ejercicio de los derechos aquí garantizados. No es usted responsable de hacer cumplir esta licencia por terceras partes.
8. Si como consecuencia de una resolución judicial o de una alegación de infracción de patente o por cualquier otra razón (no limitada a asuntos

relacionados con patentes) se le imponen condiciones (ya sea por mandato judicial, por acuerdo o por cualquier otra causa) que contradigan las condiciones de esta Licencia, ello no le exime de cumplir las condiciones de esta Licencia. Si no puede realizar distribuciones de forma que se satisfagan simultáneamente sus obligaciones bajo esta licencia y cualquier otra obligación pertinente entonces, como consecuencia, no puede distribuir el Programa de ninguna forma. Por ejemplo, si una patente no permite la redistribución libre de derechos de autor del Programa por parte de todos aquellos que reciban copias directa o indirectamente a través del usuario, entonces la única forma en que podría satisfacer tanto esa condición como esta Licencia sería evitar completamente la distribución del Programa.

Si cualquier porción de este apartado se considera inválida o imposible de cumplir bajo cualquier circunstancia particular ha de cumplirse el resto y la sección por entero ha de cumplirse en cualquier otra circunstancia.

No es el propósito de este apartado inducirle a infringir ninguna reivindicación de patente ni de ningún otro derecho de propiedad o impugnar la validez de ninguna de dichas reivindicaciones. Este apartado tiene el único propósito de proteger la integridad del sistema de distribución de software libre, que se realiza mediante prácticas de licencia pública. Mucha gente ha hecho contribuciones generosas a la gran variedad de software distribuido mediante ese sistema con la

confianza de que el sistema se aplicará consistentemente. Será el autor/donante quien decida si quiere distribuir software mediante cualquier otro sistema y una licencia no puede imponer esa elección.

Este apartado pretende dejar completamente claro lo que se cree que es una consecuencia del resto de esta Licencia.

9. Si la distribución y/o uso de el Programa está restringida en ciertos países, bien por patentes o por interfaces bajo copyright, el tenedor del copyright que coloca este Programa bajo esta Licencia puede añadir una limitación explícita de distribución geográfica excluyendo esos países, de forma que la distribución se permita sólo en o entre los países no excluidos de esta manera. En ese caso, esta Licencia incorporará la limitación como si estuviese escrita en el cuerpo de esta Licencia.
10. La Free Software Foundation puede publicar versiones revisadas y/o nuevas de la Licencia Pública General de tiempo en tiempo. Dichas nuevas versiones serán similares en espíritu a la presente versión, pero pueden ser diferentes en detalles para considerar nuevos problemas o situaciones.

    Cada versión recibe un número de versión que la distingue de otras. Si el Programa especifica un número de versión de esta Licencia que se refiere a ella y a «cualquier versión posterior», tienes la opción de seguir los términos y condiciones, bien de esa versión, bien de cualquier versión posterior publicada por la Free Software Foundation. Si el

Programa no especifica un número de versión de esta Licencia, puedes escoger cualquier versión publicada por la Free Software Foundation.

11. Si quiere incorporar partes del Programa en otros programas libres cuyas condiciones de distribución son diferentes, escribe al autor para pedirle permiso. Si el software tiene copyright de la Free Software Foundation, escribe a la Free Software Foundation: algunas veces hacemos excepciones en estos casos. Nuestra decisión estará guiada por el doble objetivo de de preservar la libertad de todos los derivados de nuestro software libre y promover el que se comparta y reutilice el software en general.

AUSENCIA DE GARANTÍA

12. Como el programa se licencia libre de cargas, no se ofrece ninguna garantía sobre el programa, en todas la extensión permitida por la legislación aplicable. Excepto cuando se indique de otra forma por escrito, los poseedores del copyright y/u otras partes proporcionan el programa «tal cual», sin garantía de ninguna clase, bien expresa o implícita, con inclusión, pero sin limitación a las garantías mercantiles implícitas o a la conveniencia para un propósito particular. Cualquier riesgo referente a la calidad y prestaciones del programa es asumido por usted. Si se probase que el Programa es defectuoso, asume el coste de cualquier servicio, reparación o corrección.
13. En ningún caso, salvo que lo requiera la legislación aplicable o haya sido acordado por escrito, ningún tenedor del copyright ni ninguna otra parte

que modifique y/o redistribuya el Programa según se permite en esta Licencia será responsable ante usted por daños, incluyendo cualquier daño general, especial, incidental o resultante producido por el uso o la imposibilidad de uso del Programa (con inclusión, pero sin limitación a la pérdida de datos o a la generación incorrecta de datos o a pérdidas sufridas por usted o por terceras partes o a un fallo del Programa al funcionar en combinación con cualquier otro programa), incluso si dicho tenedor u otra parte ha sido advertido de la posibilidad de dichos daños.

Cómo aplicar términos a un nuevo programa realizado.

Si desarrolla un nuevo programa y quiere que sea del mayor uso posible para el público en general, la mejor forma de conseguirlo es convirtiéndolo en software libre que cualquiera pueda redistribuir y cambiar bajo los términos.

Para hacerlo, se deben realizar los siguientes anuncios al programa. Lo más seguro es añadirlos al principio de cada fichero fuente para transmitir lo más efectivamente posible la ausencia de garantía. Además cada fichero debería tener al menos la línea de «copyright» y un indicador a dónde se especifique el anuncio completo.

<En una línea se debe indicar el nombre del programa y una rápida idea de qué hace.>

Copyright (C) 20aa <nombre del autor>

Indicando que este programa es software libre, puede redistribuirlo y/o modificarlo bajo los términos de la Licencia Pública General de GNU según es publicada por la Free Software Foundation, en versión **N** de dicha Licencia o bien (según su elección) de cualquier versión posterior.

Este programa se distribuye con la esperanza de que sea útil, pero sin ninguna garantía, incluso sin la garantía mercantil implícita o sin garantizar la conveniencia para un propósito particular. Véase la Licencia Pública General de GNU para más detalles.

Deberán recibir una copia de la Licencia Pública General junto con el programa. Si no se recibe, escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.

Debe añadir también información de cómo contactar con el usuario mediante correo electrónico y postal.

Si el programa es interactivo, haga que muestre un pequeño anuncio como el siguiente, cuando comienza a funcionar en modo interactivo:

Nombre del programa versión **N**, Copyright (C) 20aa nombre del autor

Nombre del programa no ofrece absolutamente ninguna garantía. Para más detalles escriba «show w».

Los comandos hipotéticos «show w» y «show c» deberían mostrar las partes adecuadas de la Licencia Pública General. Por supuesto, los comandos que

use pueden llamarse de cualquier otra manera. Podrían incluso ser pulsaciones del ratón o elementos de un menú (lo que se ajuste mas a su programa).

Si el es estudiante de Universidad debe firmar una «renuncia de copyright» para el programa, si es necesario. A continuación se cita un ejemplo, realice cambios en los nombres según sea conveniente:

> Prueba, Inc. mediante este documento renuncia a cualquier interés de derechos de copyright con respecto al programa nombre del programa (Describe lo que realiza el programa) escrito por Nombre del estudiante.
>
> <firma de Nombre del estudiante>, Día mes año
>
> Nombre del estudiante, cargo.

Esta Licencia Pública General no permite que incluya sus programas en programas propietarios. Si su programa es una biblioteca de subrutinas, puede considerar más útil el permitir el enlazado de aplicaciones propietarias con la biblioteca. Si este es el caso, use la Licencia Pública General de GNU para Bibliotecas en lugar de esta Licencia[8].

---

[8] http://gugs.sindominio.net/licencias/gples.html

5. DISEÑO DE LA INVESTIGACIÒN

5.1 TIPO DE ESTUDIO

El tipo de estudio Investigativo es el que más se ajusta a las características del proyecto ya que este comprende la descripción, análisis, diseño y se realiza de manera gradual para cumplir con el objetivo final, que es proponer un nuevo paradigma que abarque los lineamientos para el desarrollo de software con herramientas GNU en Linux.

5.2 LINEA DE INVESTIGACIÓN

La investigación esta marcada en la línea de desarrollo, evaluación, implementación de estándares y paradigmas en Ingeniería del Software, por ser herramientas que se caracterizan por permitir el desarrollo de aplicativos en función del crecimiento institucional basado en tecnología avanzadas.

Que será desarrollado con una metodología de Filosofía de constructivismo y sociocrítico los cuales consisten; el primero en construir activamente nuevos conocimientos a medida que interactúa con su entorno y el segundo paradigma en contribuir en base a los conocimientos empíricos.

El paradigma Constructivista: Todo lo que usted lee, ve, oye, siente y toca se contrasta con su conocimiento anterior y si encaja dentro del mundo que hay en su mente, puede formar nuevo conocimiento que se llevará consigo. Este conocimiento se refuerza si puede usarlo con éxito en el entorno que le rodea. No sólo es usted un banco de memoria que absorbe información pasivamente,

5.3 CRONOGRAMA

Semanas → / Actividad ↓	Año 2005																			
	Febrero				Marzo				Abril				Mayo				Junio			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Propuesta		x	x	X	x															
Recolección de información						x	x	X												
Plantea-miento del problema									x	x										
Objetivos											x	x								
Justificación													x	x						
Marco de referencia															x	x				
Hipótesis	NO APLICA																			
Metodología																	x			

Semanas → Actividad ↓	Año 2005																			
	Junio				Julio				Agosto				Septiembre				Octubre			
	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
Entrega de la 1ª versión del proyecto				X																
Estudio del proyecto				X	x	x														
Reconstru-cción del proyecto							x	X	x											
Ingeniería de requisitos										x	x	x								
Análisis del sistema													x	x	x					
Análisis Relacional "Modelos"															x	x	x			
Análisis del diseño del sistema																		x	x	X
Entrega y sustentación de la 2ª versión del proyecto																				

Semanas → Actividad ↓	Año 2005				Año 2006															
	Noviembre				Febrero				Marzo				Abril				Mayo			
	34	35	36	37	38	39	40	41	42	43	44	45	46	42	49	50	51	52	53	54
Entrega y sustentación de la 2ª versión del proyecto			x	x	x															
Correcciones de formativa II						x	x	x	x											
Ordenar la información de las herramientas a utilizar							x	x	x	x	x									
Interactuar u practicar con las herramienta									x	x	x	x								
Seleccionar ejemplos para la documentación										x	x	x	x							
Seleccionar ejercicios de taller										x	x	x	x							

Correcciones de las actividades anteriores											x	x	x	x	x					
Interacción y aprendizaje con el moodle												x	x	x	x	x				
Montar la documentación al modle													x	x	x	x	x			
Arreglo de la documentación del modle Presentar anticipo del proyecto														x	x	x	x	x	x	x

Año 2006

Semanas → Actividad ↓	Junio		Agosto				Septiembre				Octubre				Noviembre			
	55	56	57	58	59	60	61	63	64	65	66	67	68	69	70	71	72	73
Entrega y sustentación formativa III	X	X																
Arreglos de la formativa III			X	X	X	X												

Reconstrucción de la metodología						x	x	x										
Implementar capitulaje de las herramientas							x	x	x	x								
Modificar errores del Modle								x	x	x	x							
Preparación y pruebas de las herramientas									x	x	x	x						
Entrega final formativa IV									x	x	x	x						

6. POBLACIÓN Y MUESTRA

6.1 Procedimientos aplicados para la selección de los participantes.

Para el desarrollo de esta investigación se han tomado el área de la sede de Ingeniería de la Universidad Simón Bolívar, con algunos estudiantes de Ingeniería de Sistemas acerca de los conocimientos que tienen de las herramientas GNU, en el cual se les ha efectuado el estudio.

6.2 Elección de los participantes

Un grupo de estudiantes de diversos semestres de Ingeniería de Sistemas de la Universidad Simón Bolívar se prestaron para realizar el cuestionario de conocimiento de las herramientas GNU.

7. PRESENTACION DEL EQUIPO DE INVESTIGADORES

7.1 RECURSOS HUMANOS

El equipo de investigadores de este proyecto de investigación esta conformado por los estudiantes de Ingeniería de Sistemas de Décimo Semestre de la Universidad Simón Bolívar:

Maria Chamorro Hernández

Indira Cuesta Montes

Antonio Garizabalo Visbal

Yair Olmos Navarro

Cuentan con los conocimientos acerca de las herramientas GNU/Linux, lo cual a facilitado a que esta investigación se realice.

Para todo proyecto de investigación es importante decidir cuales son los recursos humanos, físicos, y financieros con los que el equipo de trabajo cuenta.

Durante el desarrollo de este proyecto de investigación hemos contado con recursos físicos esenciales como:

- La disponibilidad de los equipos de cómputos.
- La disponibilidad de la sala de informática de la institución con acceso a Internet.(salón 101)
- Prestamos de discos 3 ½.

7.2 RECURSOS FINANCIEROS

Para la elaboración de este proyecto de investigación hemos tenido una serie de gastos necesarios para la realización de esta investigación.

GASTOS:

- Transporte $ 300.000
- Materiales informáticos $ 620.000

  (Discos de 3 ½, memoria de 256, usb, tinta)
- Impresión de informes $ 450.000
- Fotocopias $ 350.000
- Acceso a Internet $ 250.000
- Almuerzos y refrigerio cuando el personal que trabajaba en la doble jornada en el desarrollo de la investigación. $ 300.000

7.3 RECURSOS TECNOLOGICOS

Hardware:

- Monitor de 14`Pulgadas con resolución de 1024 bits.
- Borrad MSM 2002.
- Procesador AMD Talón 2.8
- Memoria 256 DDR.
- Disco duro de 40 GB.
- Teclado con conector USB.
- Mouse con conector USB.
- Impresora Canon S200x.

Software:

- Windows XP.
- Instalador de Fedora 4.
- Instalación de la herramienta Gambas
- Instalación de la herramienta Eclipse

8. ESTRATEGIA PARA LA RECOLECCIÓN DE DATOS

8.1 FUENTES PRIMARIAS

Las fuentes directas que influyen en la construcción del proyecto son específicamente las investigaciones concernientes de libros, publicaciones, documentos, foros y memorias de los usuarios consultados por Internet correspondientes a los conceptos de las herramientas a utilizar para el contenido del Documento. La técnica primordial que se tomo como fuente para llevar a cabo la realización de esta investigación se contemplo a través de un cuestionario que se realizo a un grupo de estudiantes de universidad Simón Bolívar, con el fin de obtener información conocida por los estudiantes sobre las herramientas GNU y así tener los argumentos necesarios para la elaboración de esta investigación.

8.2 FUENTES SECUNDARIAS

Las fuentes secundarias que influyen en la construcción del proyecto se definen de las personas que están directamente incluidos en la elaboración del proyecto como los estudiantes y profesores que permiten que la investigación cumpla con sus objetivos principales y su finalidad principal de plantear a través de un documento el nuevo paradigma en la elaboración de software.

9. ANALISIS DE RECOLECCION DE DATOS

HERRAMIENTAS GNU

Este es un listado de herramientas para el desarrollo software de acceso libre, compiladores que se pueden obtener fácilmente bajándolos en Internet sin costo alguno. Están organizados de la siguiente forma:

Compiladores

- Gcc
- Free pascal
- Gnu Pascal
- DotGNU

Lenguaje de Programación

- Perl
- Python
- PHP
- Gprolog
- Kdevelop
- Kylix

- Quanta plus
- Eclipse
- Pascal Develop
- Lazarus
- Qt Designer
- Gambas
- Amaya
- SharpDevelop

9.1 CLASIFICACION

Compiladores

- Gcc: Es un compilador de GNU, que contiene front-end para C, C++, Objetive-C, Frortran, Java y Ada.
- Free Pascal: Es un compilador Pascal semánticamente compatible con TP 7.0, así como la mayoría de las versiones de Delphi.
- GNU Pascal: Es un compilador Pascal de GNU; que utiliza a gcc como back-end. Es compatible con Borland Pascal, y soporta muchas de sus unidades.
- Dot GNU: software libre para Webservices y para programación en c#. Los Dot GNU se están desenvolviendo rápidamente para los desktop y aplicaciones del servidor, líder industrial y proveedor de soluciones de software libre, consistiendo en tres desarrollos principales (agregando componentes con el tiempo).

Lenguajes de Programación

- Perl: Es un lenguaje interpretado de alto nivel, optimizado para procesar archivos de texto arbitrarios, extraer datos de ellos, e imprimir informes basados en esta información.
- Python: Es un lenguaje de programación interpretado, interactivo y orientado a objetos que combina una gran potencia, con una alta claridad en la sintaxis.
- PHP: Es un lenguaje de programación interpretado, fuertemente orientado a desarrollo Web.
- KDevelop: Puede generar aplicaciones genéricas o GUIs usando QT, KDE o GNOME, en C/c++.
- Quanta Plus: Es un entorno de desarrollo para HTML con soporte PHP. Esta diseñado para el desarrollo rápidos de sitios Web.
- Eclipse: Es un entorno de desarrollo integrado, especialmente orientado a proyectos en Java.
- Lazarus: herramienta de desarrollo de fuente libre y abierta para el recopilador de Freepascal, además lazarus es un lenguaje de programación que permite crear interfaces graficas, aplicaciones en modo consola. Lazarus proporciona un editor de fuente personalizado y visor visual para paquetes.
- Qt Designer: se suele utilizar en conjunto con KDevelop, aunque es multiplataforma. Sirve para generar ficheros, que contienen la interfaz

grafica de un programa que utilice el *toolkit Qt*. Se puede generar código automáticamente a partir de los ficheros creados con Qt Disigner.

- Gambas: Entorno de desarrollo para un lenguaje Basic orientado a objetos.
- Amaya: Es un editor y navegador de documentos XHTML (paginas Web). El objetivo de Amaya es la creación de documentos XHTML que cumplan fielmente las normas más actuales del W3C.
- SharpDevelop: es un entorno de desarrollo integrado para la plataforma.NET. Soporta las versiones del framework de Microsoft y de ximian, soporta desarrollo de interfaces, clases, namespaces y proyectos creados en c# y en Microsoft Studio .NET.

9.2 EVALUACION

Gambas: gambas es una herramienta de desarrollo visual de aplicaciones muy similar a los conocidos programas comerciales de Microsoft Visual Basic o Borland Delphi.

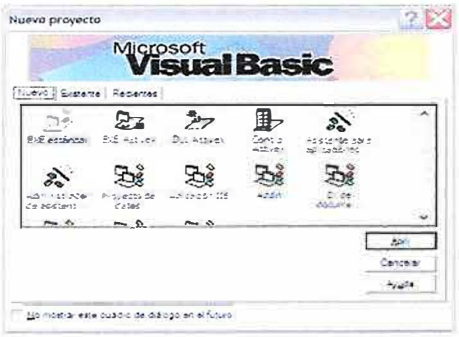
Con gambas se pueden hacer aplicaciones o programas con interfaz grafica muy rápida, pues integran un diseñador de formulario o ventanas, un editor de código, un explorador de clases, un visor de ayuda. Este tipo de herramientas han sido siempre muy habituales en la plataforma Microsoft, pero para Linux no existían tantas, o bien no estaban tan depuradas. Hay que destacar que en el desarrollo de aplicaciones en Linux emplean muchas herramientas diferentes,

cada una especializada en una tarea en concreto (por ejemplo; un compilador, un editor, un depurador, etc., (cada uno por separado).

Gambas es una herramienta que en palabras permite la creación de programas potentes, de forma fácil y sencilla. El lenguaje de programación que se utiliza es una versión del "viejo" BASIC, con el fin de acercar el desarrollo de aplicaciones a personas no expertas en programación.

Eclipse: Es una excelente herramientas de programación que permite crear aplicaciones tanto para **Windows** como para **Linux** y es **GNU**. Este programa permite crear ventanas, botones, menús y cualquier otro elemento de una forma fácil e intuitiva.

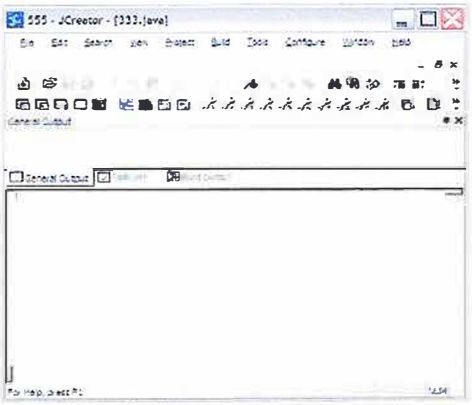
9.3 CUADRO COMPARATIVO

Windows	Linux
Visual Basic	**Gambas**
	
Es un lenguaje de programación de la Microsoft que permite a los usuarios desarrollo de software Visual Basic permite la edición para aplicaciones del sistema de programación de Visual Basic, incluida en Microsoft Excel, Microsoft Access y muchas otras aplicaciones Windows. ➢ El acceso a datos le permiten crear bases de datos como Microsoft SQL Server, Oracle entre otras. ➢ Incorporación de tecnología ActiveX Propias de visual o otras que hallan sido creadas. ➢ Creación de archivos ejecutables (.exe). ➢ Proporciona técnicas de programación necesarias para trabajar con servidores Web. ➢ Puede trabajar con protocolos y aplicaciones en Dynamic HTML de Internet. ➢ Incorporación de tecnología ADO (ActiveX Data Objects). ➢ Compatibilidad para descarga de documentos ActiveX en Internet Explorer. ➢ Permite la visualización,	Gambas es una herramienta de desarrollo visual de aplicaciones muy similar a los conocidos programas comerciales de Microsoft Visual Basic o Borland Delphi. ➢ Se puede hacer aplicaciones o programas con interfaz grafica muy rápida. ➢ Contiene un editor de código. ➢ Un explorador de clases, un visor de ayuda. ➢ Trabaja con la plataforma Linux. ➢ Contiene un compilar y depurador para manejar los errores.

edición y navegación de un conjunto de datos.

- Utiliza controles (Etiquetas), TextItem, botones de control, casillas de verificación entre otras.

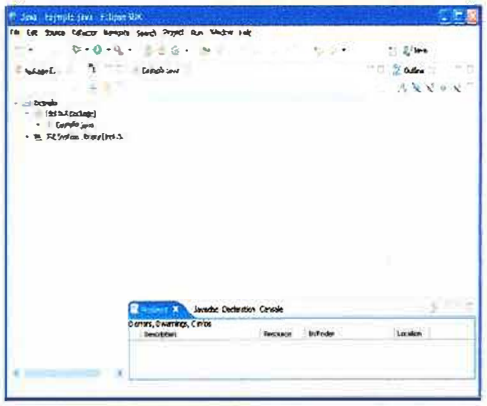
Java



- Permite fácilmente el desarrollo tanto de arquitecturas cliente-servidor como de aplicaciones distribuidas.
- Consistentes en crear aplicaciones capaces de conectarse a otros ordenadores y ejecutar tareas en varios ordenadores simultáneamente, repartiendo por lo tanto el trabajo.

Java incorpora en su propio ***API*** funcionalidades de realizar aplicaciones.

Eclipse



- Eclipse es una plataforma diseñada para facilitar el desarrollo.
- En términos de diseño la plataforma yace de los plug-ins que se dispone a los usuarios.
- Eclipse es diseñado para ser ejecutada bajo múltiples sistemas operativos brindando una integración robusta.
- La plataforma Eclipse esta estructurada como un conjunto de subsistemas los cuales son implementados en uno o más Plug-ins.

10. CAPITULOS PROPIOS DE LA INVESTIGACION

LIBRO DE GAMBAS

CAPITULO 1. INTRODUCCIÓN A LA PROGRAMACIÓN EN GAMBAS

1 INTRODUCCION A GAMBAS

1.1 QUÉ ES GAMBAS

1.2 CARACTERÍSTICAS DE GAMBAS

1.3 PROGRAMACIÓN CON LINUX

1.4 INSTALACIÒN Y DESINSTALACIÒN DE GAMBAS

1.5 ESTRUCTURA DE UN PROYECTO EN GAMBAS

1.6 COMPONENTES DE GAMBAS

1.7 ENTORNO DE PROGRAMACIÓN EN GAMBAS

1.8 UN NUEVO PROYECTO EN GAMBAS

1.9 ENTORNO DE DISEÑO EN GAMBAS

CAPITULO 2. INTRODUCCION A LA PROGRAMACION

2 PROGRAMACIÓN EN GAMBAS

2.1 ELEMENTOS DE UN PROGRAMA EN GAMBAS

2.2 METODOS ESPECIALES EN GAMBAS

2.3 DIFERENCIAS REFERENTES A GAMBAS

2.4 APLICACIONES MULTIIDIOMA EN GAMBAS

2.5 PROCESO DE TRADUCCION EN GAMBAS

2.6 EVENTOS KEY PRESS

2.7 VARIABLES

2.8 FUNCIONES

2.9 CONEXIONES CON BASES DE DATOS

CAPITULO 3. EJERCICIOS

3 EJERCICIOS EN GAMBAS

3.1 EJEMPLO 1 HOLA MUNDO

3.2 EJEMPLO 2 HOLA MUNDO 2

3.3 EJEMPLO 3 HOLA MUNDO 3

AGRADECIMIENTOS

LIBRO DE ECLIPSE

CAPITULO 1. INTRODUCCION A LA PROGRAMACION A ECLIPSE

CAPITULO 2. INTRODUCCION A LA PROGRAMACION A ECLIPSE

CAPITULO 3. EJERCICIOS

CONCLUSION

El manejo de las herramientas libres abre un nuevo paradigma para la manipulación y en el desarrollo de software, este proyecto plantea un nuevo estudio en el desarrollo de software que se asimila a dos herramientas que se maneja en el contenido académico de la Universidad Simón Bolívar de la facultad de Ingeniería de Sistemas, como lo son en la asignatura de Ingeniería de Software, la herramienta de Visual Basic y en la asignatura de Tecnología orientada a Objetos que se maneja la herramienta de JAVA, en base a los conocimientos de estas dos Herramientas, nos enfocamos al aprendizaje de un nuevo paradigma que es el manejo de herramientas para la elaboración de software, el cual es ampliar los conocimientos para la elaborar Software con herramientas GNU, en las que Investigamos, encontramos y hemos dado como resultado tangible este proyecto convertido en un libro de investigación.

A medida que transcurrían los semestres con la Asignatura de formativa contribuíamos poco a poco en la elaboración del libro, en las que planteábamos partes interesantes en los capítulos aprendimos en la utilización de las nuevas herramientas, y comprendimos de manera a priori la descarga de las herramientas, conocer los entornos gráficos de las herramientas y realizar ejercicios pequeños, sencillos de aplicaciones como el "HOLA MUNDO", que se encuentran detallada mente en la construcción del proyecto de

investigación, el cual nos sentimos plenamente orgullosos y gratamente el tener el placer de ofrecer a los estudiantes de Ingeniería de Sistemas el desarrollo de esta Investigación que contribuirá satisfactoriamente en los conocimientos de los participantes que den un nuevo y un gran cambio en el desarrollo de software.

RECOMENDACIÓNES DEl PROYECTO

En la realización de esta investigación se elaboró una documentación que contiene la información necesaria del paralelo que existe entre dos herramientas para el desarrollo de software como son Visual Basic, Java frente a su similar de Gambas, Eclipse para los sistemas operativos Windows y Linux respectivamente con el fin apoyar a los desarrolladores, programadores, diseñadores y todas esas personas que hacen parte esencial en la elaboración de software brindándole la opción de conocer herramientas que son amenas, sencillas, rápidas, potentes de utilizar.

En la elaboración de software con las herramientas GNU descritas anteriormente permiten por sus características una buena relación entre software y los desarrolladores de los proyectos, además que cada proyecto realizado pueden ser publicados como software libre, es decir que cualquier persona interesada con conocimientos básicos de programación puedan realizarle mejoras a los proyectos con el propósito de aumentar su calidad como producto final.

Como punto de principal de esta investigación se realizó la documentación para que sirva de guía a todas aquellas personas que deseen adquirir nuevos conocimientos acerca de herramientas para el desarrollo de software libre y

que perciban de este como una necesidad de aprendizaje para avanzar en la formación como profesionales.

El contenido de esta investigación esta fortalecido con un curso virtual (e-learning), que se encuentra ubicado en el Moodle de educación virtual de la Universidad Simón Bolívar, en el espacio de cursos libres. Para participar del curso virtual que tiene por nombre HERRAMIENTAS GAMBAS Y ECLIPSE es necesario tener una contraseña que será **"HGNU"** para acceder y obtener toda la información necesaria como conceptos, interfaces, gráficos, ejemplos entre otros que ayudaran ha facilitar el interactuar con las herramientas.

Para que los esfuerzos realizados durante esta investigación tengan un mayor aprovechamiento, se tiene pensado solicitar a los profesores de ingeniería de sistemas de la Universidad Simón Bolívar en las asignaturas de ingeniería de software elaborar talleres de aprendizaje que permitan tener la inducción acerca de los conocimiento básicos como conceptos, funcionamiento e importancia de utilizar las herramientas GNU para Linux.

La documentación contenida en la investigación es muy similar al contenido del curso virtual que se encuentra en el Moodle con la finalidad de que se tenga relación entre los dos componentes anteriormente mencionados y cuenta con descarga de los ejercicios para un mayor aprendizaje.

GLOSARIO

Proyecto GNU: Fue creada para financiar servicios que pueden desarrollar y modificar múltiples usuarios para desarrollar software de amplia productividad.

Herramientas GNU: Software de libre distribución, la denominación de software GNU engloba de forma general a cualquier software desarrollado y mantenido por el proyecto GNU.

UNÍX: Sistema operativo multitarea, multipuesto y multiusuario.

Linux: Sistema operativo derivado de UNIX que, manteniendo la generalidad de sus características, como el ser multitarea y basado en bibliotecas dinámicas. Versión del Kernel de Unix System V Reléase 3.0 desarrollado para PC con microprocesadores 80386 y superiores. Creado por el sueco Linus Torvalds en colaboración con otros muchos usuarios repartidos por todo el mundo. Linux se distribuye de manera gratuita con código fuente a traves de BBS e Internet.

Gambas: Es una herramienta, que permite la creación de programas potentes, de forma fácil y sencilla. El lenguaje de programación que se utiliza es una versión del "viejo" BASIC.

Eclipse: Es una excelente herramienta de programación que permite crear aplicaciones tanto para Windows como para Linux y es GNU. Este programa permite crear ventanas botones, menús y cualquier otro elemento de una forma fácil e intuitiva.

Aplicación (es): Programa o conjunto de programas de computadora que tiene por objeto realizar una determinada tarea de forma automática.
Programa diseñado para asistir en la realización de una tarea especifica, tal como un procesador de textos, contabilidad o gestión de inventario.

Base de datos: Son archivos compuestos por registros donde cada uno contiene campos junto con un conjunto de operaciones para realizar búsquedas, ordenaciones, reordenaciones y otras funciones.

Código (s): Conjunto de instrucciones de un programa. Representación de sentencias humanas legibles escritas por un programador en un lenguaje de programación.

Caja de herramientas: Conjunto de rutinas predefinidas y normalmente precompiladas que pueden ser utilizadas por un programador para escribir un programa para una maquina, entorno o aplicación en particular.

Componente (s): Una parte discreta de un sistema o estructura mas grande. Una rutina de software modular individual que se ha compilado y enlazado dinámicamente preparadas para utilizarse con otros componentes o programas.

Conexión (ces): Enlace físico por medio de cables, cable de fibra óptica u otro medio entre dos o mas dispositivos de comunicaciones. Una conexión a base de datos se refiere a la comunicación, interrelación de un programa con una base de datos.

Control (es): Se puede definir como la administración de una computadora y sus habilidades de procesamiento de modo que se mantenga el orden mientras que se llevan a cabo las tareas y actividades. Los controles mas comunes son botones, que permiten que el usuario seleccione opciones y barras de desplazamiento que a su vez permitirán al usuario navegar por ellas.

GUI: Interfaz grafica de usuario basada en graficas que incorpora iconos, menús despegables y un Mouse.

GPL: General Public License, licencia Pública que permite la utilización, modificación e instalación de programas o software en un ordenador sin limitación.

GNU/Linux: Sistema operativo GNU que en la mayoría de los casos utiliza un kernel Linux, lo que conlleva a nombrarlo GNU/Linux.

Shell: Programa independiente o constituir un elemento básico de un sistema operativo. Proporciona una comunicación directa entre el usuario y el propio sistema operativo, y así facilita la ejecución de órdenes o comandos del sistema y de los programas que se ejecutan en él.

TCP/IP: Transmisión Control Protocoló/Internet Protocoló, protocolo de control de transmisiones / protocolo de Internet, protocolos usados para el control de la transmisión en Internet. Permite que diferentes tipos de ordenadores o computadoras se comuniquen a través de redes heterogéneas.

IDE: (Entorno de Desarrollo Integrado), es un software que consiste en un editor de texto, un compilador herramientas build-Automation, generación de código automático y un depurador (debugger) todo esto a partir de un interfaz de usuario, también se puede decir que es un ambiente en el que están presentes todas las herramientas necesarias para diseñar, ejecutar y probar una aplicación.

Software: Conjunto de programas, instrucciones, métodos y procedimientos relacionados con la explotación funcionamiento y manejo de un sistema de proceso de datos.

Hardware: son todos los elementos o equipos que componen la computadora utilizados para el funcionamiento de la misma.

Programa: Conjunto de instrucciones preparadas al objeto de resolver un determinado problema mediante la utilización de una computadora. Un programa se escribe en un lenguaje de programación y se convierte al lenguaje de la maquina por medio de programas denominados ensambladores y compiladores.

Directorio(s): Es el modo de organizar y agrupar archivos para que el usuario no se vea abrumado por una lista de archivos.

Entorno (s): Configuración de recursos disponibles para el usuario, entorno se refiere al hardware y al sistema operativo ejecutándose.

Formulario (s): Es un documento estructurado con espacios reservados para acceder a la información y que a menudo contiene codificación especial.
Son ventanas estructuradas, cuadros u otro elemento de presentación autónomo con áreas predefinidas para introducir o cambiar información, también conocida como filtro visual de los datos.

Herramienta (s): Conjunto de rutinas diseñadas para permitir a los desarrolladores escribir programas mas fácilmente en una determinada computadora, sistema operativo o interfaz de usuario.

IDE: (Entorno de Desarrollo Integrado), es un software que consiste en un editor de texto, un compilador herramientas build-Automation, generación de código automático y un depurador (debugger) todo esto a partir de un interfaz de usuario, también se puede decir que es un ambiente en el que están presentes todas las herramientas necesarias para diseñar, ejecutar y probar una aplicación.

Información: Se refiere al significado que selle da a los datos, según el valor y el significado que se le quiera dar, las computadoras procesan los datos sin tener constancia de lo que estos representan en realidad.

Interfaz (ces): Tipos de entornos que representan en la pantalla programas, archivos y opciones por medio de iconos, menús y cuadros de dialogo.
Punto en que se tiene lugar la conexión entre dos elementos de tal forma que ambos puedan trabajar en cooperación.

Lenguaje (s): Conjunto de símbolos y reglas utilizados para conducir información. Cualquier lenguaje artificial utilizado para definir una secuencia de instrucciones que la computadora podrá finalmente procesar y ejecutar.

Menú (s): Lista de funciones disponibles en las que el usuario puede hacer una selección en una aplicación. Los comandos aparecen como opciones dentro de una barra de menús, situadaza en la parte superior de la ventana de la aplicación.

Método (s): Se refiere a los procesos realizados por un objeto cuando recibe una orden de mensaje.

Módulo (s): Colección de rutinas y estructuras de datos que realiza una tarea particular o implementa un tipo abstracto de datos particulares.
Los módulos constan de dos partes la interfaz y la implementación privada, la primera muestra las constantes, tipos de datos, variables y rutinas a los que puede acceder, la segunda contiene el código fuente de la aplicación.

Objeto (s): Son variables que constan de rutinas y datos, que es tratada como una entidad discreta. Se puede decir que es una representación visual.

Propiedad (es): Se refiere a las características o parámetros que contiene un objeto o dispositivo por ejemplo donde se incluyen datos tales como tipo, nombre, fecha de elaboración entre otros.

Redes: Grupo de computadores y dispositivos asociados que son conectados para tener facilidad de comunicaciones. Una red puede implicar conexiones permanentes, como cables o las conexiones temporales realizadas a través de teléfono u otro enlace de comunicación.

Variable (s): Se refiere al lugar de almacenamiento con nombre que puede contener datos y que el valor asociado a esta debe ser modificado durante la ejecución de un programa.

Ventana(s): representación visual de un programa, aplicación e interfaces graficas en una porción de pantalla y que puede contener documentos o mensajes que son activados mediante movimientos del cursor, ordenes y entradas de textos actuales.

Aplicaciones: Programas o conjunto de programas de computadora que tiene por objeto realizar una determinada tarea de forma automática.

Datos: Termino usado para describir las señales con las cuales trabaja la computadora.

Tecnología: Conocimiento general que se aplica al proceso a través del cual los seres humanos diseñan herramientas y máquinas para incrementar su control y su comprensión del entorno material.

Lineamientos: Sucesión continúa de una Conducta o comportamiento en una determinada dirección, Orientación o estilo de un arte o de un saber cualquiera que se deseé aprender.

BIBLIOGRAFIA

- ✓ Internet, http://WWW.MONOGRAFIAS.COM
- ✓ Internet, http://linuxfocus.org (Manual Gambas: Basic para Linux)
- ✓ Internet, http://wiki.gnulinex.org/gambas/ (Manual Visual Introduction ToGambas)
- ✓ Internet, http://gambas.sourceforge.net (Manual Gambas tutorial.pdf)
- ✓ http://gambas.gnulinex.org/gambas_tutorial.pdf.
- ✓ http:// gambas.sourceforge.net/
- ✓ www.eclipse.org\Desarrollo de Plug-ins.pdf\\.
- ✓ http\\wipipedia\enciclopedia_libre\Linu_GNU\Linux.pdf\html
- ✓ WELSH, Mat. Instalación y Primeros Pasos: Linux.pdf (Traducción: Proyecto Lucas, versión en castellano) Copyright c 1992-1996.
- ✓ JOYANES AGUILAR, Luís y MUÑOZ CLEMENTE, Antonio. Microsoft Visual Basic 6.0.

ANEXOS

CUESTIONARIO

Marca con una X la respuesta.

1. ¿Has escuchado sobre herramientas GNU/Linux?

   SI _____ **NO**_____

2. ¿Sabes que son herramientas GNU/Linux?

   SI _____ **NO**_____

3. ¿Sabes manejar algunas de estas herramientas?

   SI _____ **NO**_____

4. ¿Te gustaría que los lineamientos de herramientas de GNU/Linux fueran dinámicos y entendible?

   SI _____ **NO**_____

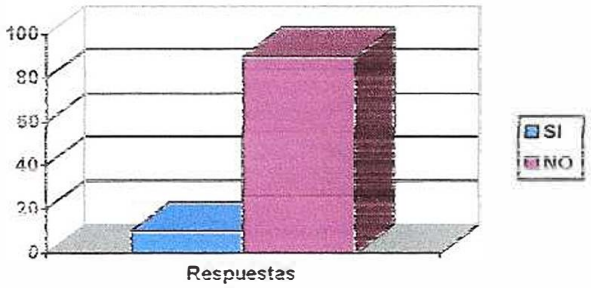
5. ¿Crees que los lineamientos de herramientas GNU/Linux enriquecerán los conocimientos de los estudiantes?

   SI _____ **NO**_____

TABULACION

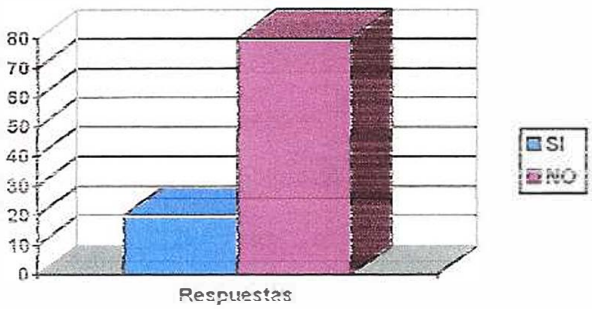
1. ¿Has escuchado sobre herramientas GNU/Linux?

El resultado de esta pregunta se resume en que el 10% contesto SI y el 90% contesto NO.



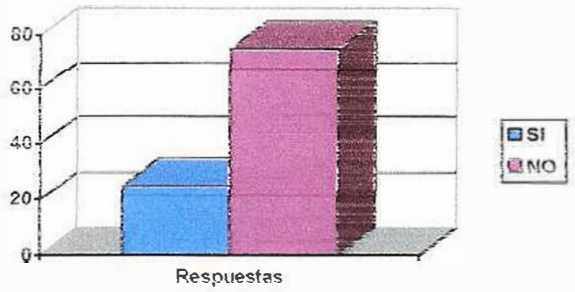
2. ¿Sabes que son herramientas GNU/Linux?

El resultado de esta pregunta se da en que el 20% contesto SI y el 80% contesta NO.



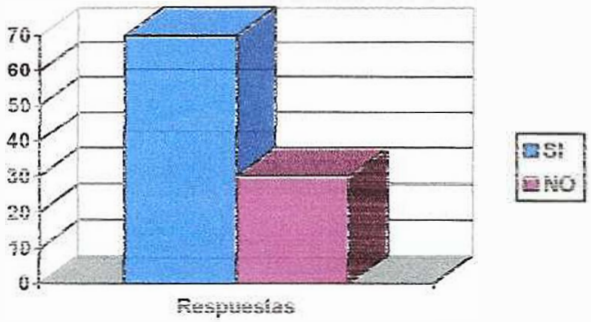
3. ¿Sabes manejar algunas de estas herramientas?

El resultado de esta pregunta se muestra en que el 25% contesta SI y el 75% contesta NO.



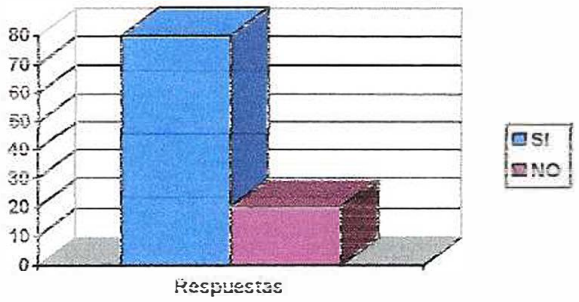
4. ¿Te gustaría que los lineamientos de herramientas de GNU/Linux fueran dinámicos y entendible?

El resultado de esta pregunta se resume en que el 70% contesto SI y el 30% contesto NO.



5. ¿Crees que los lineamientos de herramientas GNU/Linux enriquecerán los conocimientos de los estudiantes?

El resultado de esta pregunta se da en que el 80% contesta SI y el 20% respondió NO.



HERRAMIENTAS GAMBAS Y ECLIPSE

CHAMORRO HERNANDEZ MARIA
CUESTA MONTES INDIRA
GARIZABALO VISBAL ANTONIO
OLMOS NAVARRO YAIR

UNIVERSIDAD SIMÓN BOLÍVAR
FACULTAD DE INGENIERIA DE SISTEMAS
BARRANQUILLA
2006

INTRODUCCION

Este libro fue elaborado con fines educativo buscando incentivar a los discentes de la Universidad Simón Bolívar a la programación de un nuevo paradigma, dando a conocer conceptos y ejemplos que ayudaran a facilitar el manejo de las herramientas de desarrollo de software libre.

Será de utilidad estar familiarizado con los conceptos básicos de las herramientas de Visual Basic y Java, ya que el contenido temático del libro va enfocado al manejo y al aprendizaje de las herramientas Gambas y Eclipse, permitiendo mediante imágenes los entornos gráficos de cada lenguaje, guiando al usuario de manera clara y precisa en el proceso de aprendizaje.

AGRADECIMIENTO

A Dios padre por permitimos aprender día a día todos los conceptos necesarios para llevar acabo este gran esfuerzo, A los Ingeniero Sergio Jiménez, Fabio Moya, Javier Henriquez y Freddy Briceño por brindarnos su colaboración para la realización de este libro.

Especialmente agradecemos a la Universidad Simón Bolívar por brindarnos las Herramientas

TABLA DE CONTENIDO

Pagina

GAMBAS

CAPITULO 1
1. INTRODUCCIÓN A LA PROGRAMACIÓN EN GAMBAS

Este capitulo contiene los conceptos básicos que se necesitan para el manejo e implementación de Gambas con cada uno de los componentes, entornos, estructuras y elementos indispensables en la programación para la elaboración de los proyectos como lo son métodos y variables que ayudaran a facilitar el manejo de las herramientas.

1.1 INTRODUCCIÓN A GAMBAS

Gambas es uno de los lenguajes de programación que permite facilitar a los programadores e incluso a los principiantes lograr perfectamente desarrollar aplicaciones para Linux. Hasta ahora el entorno más habitual para el desarrollo de aplicaciones era el Basic (Acrónimo de Beginners All-Purpose Symbolic Instruction Code) Que era el IDE de visual Basic de Microsoft. Gambas ha ido perfeccionando su entorno de desarrollo incorporando conexiones a bases de datos y un entorno muy útil para los procesos de conexiones a redes.

Gambas logra incorporar un conjunto de herramientas de gran potencia, que ayudan fácilmente en la realización de aplicaciones.

1.2 ¿QUÉ ES GAMBAS?

Gambas es una herramienta de desarrollo visual de aplicaciones muy similar a los conocidos programas comerciales Microsoft Visual Basic o Borland Derphi. Con Gambas se pueden hacer aplicaciones o programas con interfaz gráficas

de forma muy rápida, pues integra un diseñador de formularios o ventanas, un editor de código, un explorador de clases, un visor de ayuda y otros elementos que facilitan trabajo al desarrollador.

Gambas fue creado por Benoit Minisini quien se basó en la utilización del "viejo" Lenguaje Basic. Puede sorprender que se halla escogido un lenguaje tan básico e incluso limitado con es el Basic, pero no hay que olvidar que uno de los objetivos de la herramienta es acercar el desarrollo de aplicaciones a personas tan expertas en la programación.

1.3 CARACTERÍSTICAS DE GAMBAS

LAS PRINCIPALES CARACTERÍSTICAS DE GAMBAS SON:

- Un proyecto en gambas se almacena bajo un directorio. El archivador transforma la estructura del directorio del proyecto en un fichero ejecutable único.
- La compilación de un proyecto requiere solamente la compilación y ejecución de las clases modificadas. Cada referencia externa de una clase se soluciona dinámicamente en tiempo de ejecución.
- Gambas tiene un componente de arquitectura que permite ampliar el lenguaje de programación. Cualquier persona que no conozca gambas puede escribir componentes como librería compartidas que agreguen dinámicamente a nuevas clases nativas al intérprete. El componente de arquitectura se documenta en la enciclopedia de Wiki.
- Por defecto, el intérprete de gambas es un programa y el componente de arquitectura lo utiliza para escribir en el utilizador de interfaz grafica del lenguaje de programación.
- Cuando se utiliza la interfaz grafica de gambas se coloca en ejecución como un componente, donde gambas puede ser independiente de cualquier caja de herramientas. Cualquier programador puede escribir

un programa y elegir la caja de herramientas más adelante: Gtk+, Qt entre otras.

- Cualquier ventana o caja de dialogo se puede utilizar como un control. En la versión de visual Basic se debe usar el control ActiveX.
- Los proyectos de gambas pueden ser fácilmente traducidos a cualquier lenguaje.

1.4 PROGRAMACIÓN CON LINUX

El concepto de desarrollo de aplicaciones se ha extendido últimamente con el uso de Linux hasta el escritorio del usuario.

Linux se esta interesando en el desarrollo de aplicaciones, diseñando programas para cumplir con este fin de soportar fallos en el sistema, buscando con un entorno de desarrollo visual lograr unir correctamente los conceptos de programación orientada a objeto, programación estructurada y modulación.

Linux también esta siendo empleada para las respuestas a las necesidades de los usuarios tales como leer correos electrónicos, navegar en la Web, ediciones de textos entre otros.

Con esta tendencia se puede encontrar varios entornos de desarrollo de software donde únicamente estaremos resaltando del lenguaje de programación gambas para la programación orientada a eventos y Eclipse con la programación orientada a objetos.

1.5 INSTALACIÓN Y DESINSTALACIÓN DE GAMBAS

1.5.1 INSTALACIÓN DE GAMBAS

Para la instalación se hace necesario contar con el instalador, actualmente la ultima versión es la 1.0.10 disponible para descarga desde la pagina http://gambas.sousceforge.net/ en el apartado descargas (download).

Para configurar gambas en una distribución de Linux es necesario tener en cuenta la compatibilidad entre el programa y la distribución, la siguiente tabla muestra la compatibilidad actual entre estos dos:

Distribución de Linux	Versión	Estado	Notas
ArkLinux	Alpha 11.1	OK	
Conectiva	10	OK	
Debian	Woody Sarge/Sid	OK	Mirar el archivo README.DEBIAN en el paquete fuente.
Fedora	Core 4	OK	
	Core 3	OK	
Gentoo	1.4	OK	
Lindowsspire		?	Algunos problemas no resueltos con QT Styles.
Linex		OK	
Mandriva	10.x	OK	
QiLinux		OK	
RedHat	9.0	OK	Se debe actualizar la librería Qt o modificar Makefile.am manualmente. Vea el archivo README.READHAT en
	8.0	OK	Se debe compilar la librería Qt 3.2 y quizás algunas otras. Vea el archivo README.READHAT en el paquete fuente.
Slackware	10.1	OK	
	10	OK	
	9.1	OK	
SuSe	9.3	OK	
	9.2	OK	
	9.1	OK	

	9.0	OK	Algunos problemas con la forma en que SuSe establece el LANG y LC_* system environment variables.
Xandros	2.0	OK	

Antes de instalar gambas y después de haber revisado lo anterior es necesario instalar los siguientes paquetes en el sistema:

- The X11 development packages.
- The Qt 3 development packages.
- The KDE 3 development packages if you want to compile the KDE component.
- The PostgreSQL, MySQL, SQLite or ODBC development packages if you want to compile database drivers.
- The libcurl development packages (version 7.10.7 or greater) if you to compile de network-curl component.
- The SDL and SDL_mixer development packages if you want to compile the SDL component.
- The libxml and libxsit development packages if you want to compile the xml components.
- The GTK+ development packages for the GTK component.

Para la instalación se deben seguir los siguientes pasos:

a. Abrimos una ventana de terminal y descomprimimos el instalador utilizando el comando bunzip2 y vemos en contenido del mismo:

[linux@home ~]$ cd ~

[linux@home ~]$ bunzip2 gambas-1.0.10.tar.bz2

[linux@home ~]$ tar xf gambas-1.0.10.tar

[linux@home ~]$ ls

b. Un directorio llamando gambas a sido creado con el contenido de archivo, debemos entrar en este directorio:

[linux@home ~]$ cd gambas-1.0.10

c. Ahora es necesario que el instalador analice nuestro sistema y determine si contamos con los requerimientos necesarios, además de adaptar la configuración a nuestro sistema:

[linux@home gambas-1.0.10]$ ./configure

Nota: En la pantalla empezarán a aparecer los nombres de los componentes y sus estados.

d. Si la configuración y compilación finaliza sin errores podemos ejecutarla así:

[linux@home gambas-1.0.10]$ make

Nota: Si al ejecutar este comando la terminal no realiza ninguna operación es necesario verificar los componentes y paquetes del sistema necesarios para gambas.

e. Una vez se termina la compilación sin problema se procede a la verdadera instalación de gambas, para ello debemos conocer la contraseña del super usuario (root):

[linux@home gambas-1.0.10]$ su –c "make install"

Nota: Una vez se ejecuta este comando lo primero que le pedirá será la contraseña (password) del root.

f. Una vez terminado el proceso anterior se puede crear una acceso directo para la aplicación o ejecutarla desde la opción ejecutar un programa del menú de la distribución.

1.5.2 DESINSTALAR GAMBAS

Para desinstalar gambas es necesario estar conectado como root y desde una ventana de terminar, remover el directorio de instalación que por defecto es /opt/gambas

[root@home ~] # rm –rf /opt/gambas

1.6 ESTRUCTURA DE UN PROYECTO EN GAMBAS

La estructura de los proyectos en gambas, se orientan como en toda la realización de proyectos en una fase de diseño.
Gambas es un entorno grafico de desarrollo totalmente libre y sencillo, es un interprete Basic para Linux aunque no es compatible con VB, pero si tan potente y fiable como este. Con objetivo principal centrada en la creación rápida y ágil de interfaces graficas para aplicaciones.
La mayor ventaja de gambas esta vinculada a sus librerías dinámicas que funcionan a modo de componentes (Red, gráficos para QT, sonido con SDL) y un asistente para la creación de paquetes RMP y DEB, así como la distribución de fuentes en formatos GZ y BZ2, que viene con un sistema de documentación ejemplar.

1.7 COMPONENTES DE GAMBAS

Los componentes actúan como "plugins" para Gambas, añadiendo nuevas clases al intérprete. Cada proyecto incluye la lista de componentes a cargar en el momento de su ejecución; se encuentran en las "Propiedades del Proyecto".

Un proyecto Gambas sin ningún componente es una simple aplicación de consola. Para convertirse en una aplicación gráfica, debe utilizar al menos el componente gb.qt. Para acceder a bases de datos, debe utilizar el componente gb.db.

- gb - Clases internas, nativas de Gambas
- gb.compress - Librería de Compresión
- gb.db - Componente de Acceso a bases de datos
- gb.debug - Ayuda para la depuración de proyectos Gambas
- gb.eval - Evaluador de expresiones Gambas
- gb.net - Componente de red
- gb.qt - Componente gráfico basado en la librería QT de componentes
- gb.qt.editor - Editor de Gambas con resaltado de sintaxis
- gb.qt.ext - Extensiones del componente gráfico QT
- gb.qt.kde - integración y scripts para KDE
- gb.qt.kde.html - Navegador KDE
- gb.sdl - Librería basada en SDL
- gb.sdl.sound - Librería de sonido basada en SDL
- Gob. - Compatibilidad con VB
- gb.xml.libxml - Herramientas XML basadas en libxml
- gb.xml.libxml.rpc - Cliente XML-RPC basado en libxml y libcurl
- gb.xml.libxml.xslt - Herramienta XSLT basada en libxslt

- **gb**

Este componente contiene todas las clases incluidas por defecto en el intérprete.

Clases

Application, Byte[], Class, Classes, Collection, Component, Components, Date[], Error, File, Float[], Integer[], Object, Object[], Process, Settings, Short[], String, String[], System, Variant[], gb.

- **gb.compress**

Este componente permite comprimir y descomprimir archivos, así como leer y escribir en archivos comprimidos utilizando los métodos habituales en archivos. También permite comprimir y descomprimir cadenas en memoria.

Clases

Compress, Uncompress.

Actualmente, el componente puede utilizar dos algoritmos, gracias a las librerías zlib y libbzip2. Los programas más conocidos que utilizan estos algoritmos son las utilidades de línea de comandos "gzip" y "bzip2", por lo que podrá utilizar archivos generados por estas utilidades.

Este componente emplea internamente las librerías "libz" y "libbz2", por lo que debe tenerlas instaladas en los equipos que hagan uso del componente de compresión. Si sólo desea utilizar uno de los dos algoritmos, no es necesario que instale ambas librerías.

- **gb.db**

Este componente le permite acceder a los siguientes sistemas gestores de bases de datos

- ✓ PostgreSQL
- ✓ MySQL
- ✓ Sqlite (experimental)

Advertencia: Tanto PostgreSQL como MySQL son sistemas basados en arquitectura cliente / servidor, lo que significa que la conexión se realiza con un proceso servidor. Sqlite es una base de datos en un archivo o en memoria, por lo que no hay servidor al que conectarse. Por tanto, el usuario necesita introducir una ruta completa al archivo que forma la base si no se encuentra en la carpeta de trabajo de la aplicación, o incluir la ruta hacia la carpeta donde se encuentran las bases de datos en la variable de entorno

GAMBAS_SQLITE_DBHOME.

Clases

Connection, DB, Result, Table.

- **gb.debug**

Este componente permite al entorno de desarrollo depurar proyectos Gambas.

Clases

Debug.

- **gb.eval**

Este componente se emplea para evaluar expresiones **Gambas** en tiempo de ejecución.

El componente se carga automáticamente si utiliza la función Eval ().

Clases

Expression.

- **gb.net**

  Este componente permite implementar clientes y servidores de red con Gambas. También permite el trabajo con puertos serie.

  Clases

  DnsClient, Net, SerialPort, ServerSocket, Socket, UdpSocket

- **gb.qt**

  Este componente implementa las clases de la Interfaz Gráfica de Usuario. Está basado en la librería QT

  Clases

Align	Cursor	GridView	Picture	ToggleButton
Application	Desktop	IconView	PictureBox	ToolButton
Arrange	Dialog	Image	Printer	TreeView
Border	Drag	Key	ProgressBar	Window
Button	Draw	Label	RadioButton	Windows
CheckBox	Drawing	Line	Scroll	
Clipboard	DrawingArea	ListBox	ScrollView	
Color	Fill	ListView	TabStrip	
ColumnView	Font	Menu	TextArea	
ComboBox	Fonts	Message	TextBox	
Container	Form	Mouse	TextLabel	
Control	Frame	Panel	Timer	

- **gb.qt.editor**

Este componente implementa un editor de código fuente para Gambas, y es utilizado por el IDE de Gambas.

Clases

Gambas editor.

- **gb.qt.ext**

Este componente contiene algunas clases adicionales de la interfaz gráfica, que son específicas de la librería QT.

Clases

Dial, LCDNumber, ScrollBar, Slider, SpinBox, Splitter, TableView, TextView, Workspace.

- **gb.qt.kde**

Este componente transforma una aplicación Gambas en una aplicación KDE. Esto permite a la aplicación gobernar otras aplicaciones KDE utilizando el protocolo DCOP.

Clases

Application, ColorBox, DCOPRef, DatePicker, Dialog, KDEApplication, URLLabel.

- **gb.qt.kde.html**

Este componente permite el uso del khtml, buscador Web en su programas de gambas.

Clases

WebBrowser

- **gb.sdl**

Este componente permite el manejo de CD de audio. Puede listar las unidades de CDROM detectadas con la clase CDRoms y manejarlas con la clase CDRom.

Clases

CDRom, CDRoms.

- **gb.sdlsound**

Este componente permite reproducir sonidos en Gambas.

En el futuro, los métodos de la librería SDL para gráficos y animaciones también estarán soportados.

Este componente puede gestionar hasta 32 canales de sonido, los cuales pueden reproducir sonidos desde memoria, más una pista musical que puede reproducir música desde un archivo. El mezclado se realiza en tiempo real.

Clases

Channel, Channels, Music, Sound.

- **gb.vb**

Este componente implementa una serie de funciones que imitan el comportamiento similar al de Visual Basic.

Clases

Vb

- **gb.xml.libxml**

  Este componente le permite a usted leer y escribir archivos XML. Este es basado en libxml2, y actualmente, es una versión beta, entonces asegúrese si ud quiere usar esto para sus programas.

  Clases

  XmlDocument, XmlNode, XmlReader, XmlReaderNodeType, XmlWriter

- **gb.xml.libxml.rpc**

  Clases

  RpcType

  XmlRpc

- **gb.xml.libxml.xslt**

  Clases

  Xslt

1.8 ENTORNO DE PROGRAMACIÓN EN GAMBAS

El entorno de programación en gambas contiene un conjunto de herramientas que ayudan en la elaboración de aplicaciones. En se puede activar varios documentos a la vez. La creación de nuevos proyectos en gambas esta completamente asistida, con herramientas tales como gestor de base de datos, variables globales, soporte de clases y herencia. Además de las propiedades y métodos que contienen los formularios, controles se puede incluir el autoacompletado de código que ayuda a facilitar la codificación de procedimientos y funciones en una determinada aplicación.

1.9 UN NUEVO PROYECTO EN GAMBAS

Cuando se ejecuta Gambas la primera ventana que se activa es la de bienvenida de Gambas

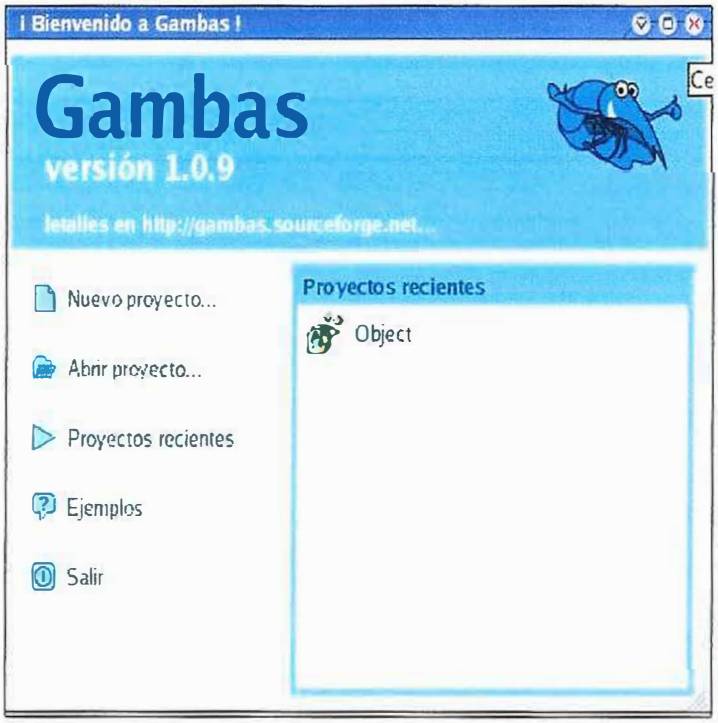


Figura1.1 ventana de bienvenida de Gambas

Donde se deberá escoger la opción Nuevo Proyecto, luego notaremos que automáticamente se activa la ventana de bienvenida al asistente de creación de Gambas que permitirá crear un Nuevo Proyecto. Gambas permitirá crear el Nuevo Proyecto desde cero o desde otro proyecto ya existente.

Figura1.2 ventana de creación de un proyecto en Gambas

Luego de acceder presionando el botón siguiente del asistente de creación de proyectos de Gambas, se activa la ventana en la que se muestran los iconos relativos a los diferentes diseños para los tipos de proyecto que se pueden elegir para un Nuevo Proyecto en Gambas (fig1.3 Gambas).

Figura1.3 ventana de selección del tipo de proyecto en Gambas

Luego de acceder pulsando el botón siguiente, se activara una ventana de selección del proyecto donde hay que escribir algunos datos como nombre y titulo del proyecto (fig1.4 Gambas).

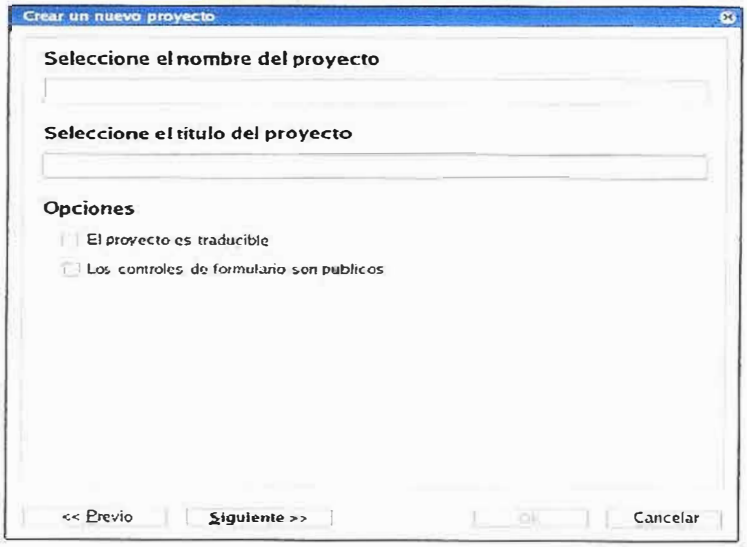


Figura1.4 ventana de selección para escribir el nombre del proyecto en Gambas

Si el usuario presiona el botón siguiente sin haber escrito el nombre del proyecto, Gambas le indicara por medio de un mensaje que debe escribir un nombre del proyecto que desea realizar (fig1.5 Gambas).

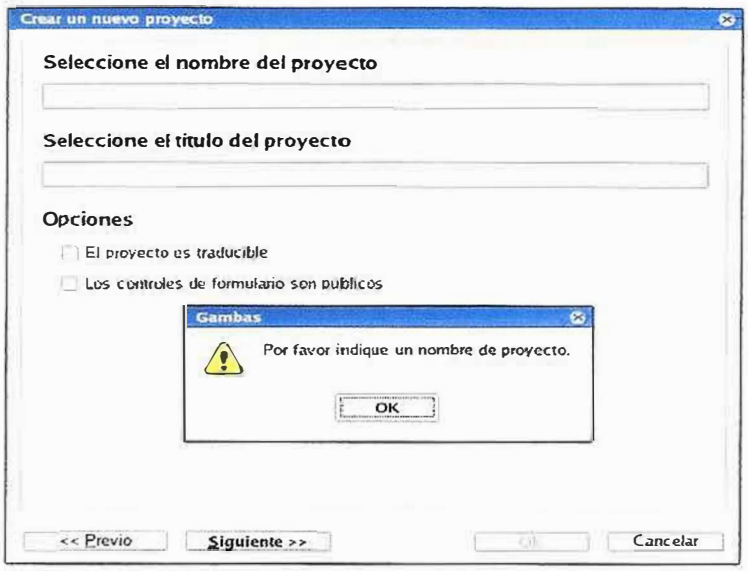


Figura1.5 ventana de selección para escribir el nombre del proyecto en Gambas, cuando presenta errores.

Además que Gambas agrega opciones como si quieres que el proyecto sea traducible o no y que los controles de los formularios sean públicos, estas dos instrucciones no sean necesarias de colocar. Luego de haber escrito el nombre del proyecto y acceder presionando el botón siguiente de la ventana de selección, se activa una ventana de selección donde se debe especificar la carpeta apartir de la cual se creara una carpeta donde se almacenaran los archivos del proyecto Ejemplo:"**/home/sistema**", donde automáticamente se ha creado una carpeta con "**/home/sistema/Nombre del Proyecto**", presione siguiente (fig1.6 Gambas).

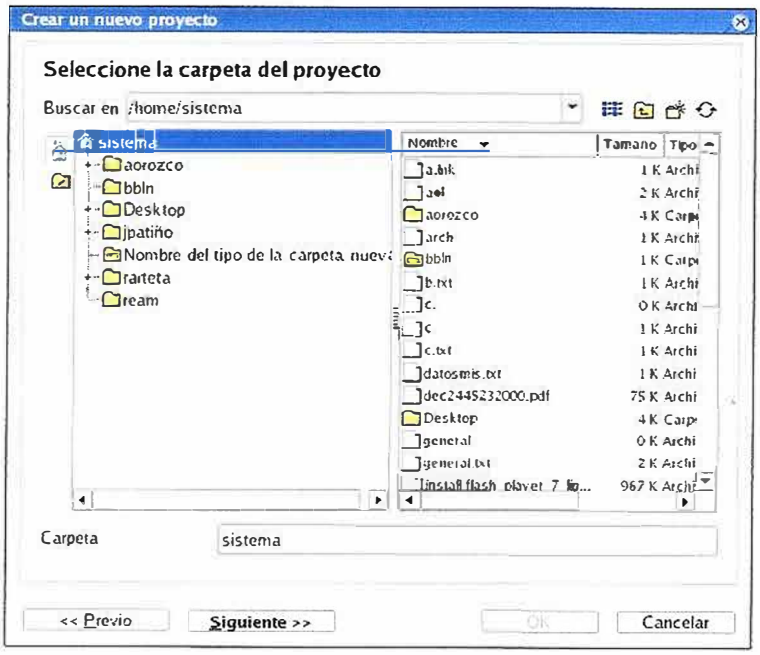


Figura1.6 ventana de selección para escribir el nombre del directorio a que pertenece el proyecto en Gambas.

Gambas después de haber detallado toda la información, activa una ventana que indica que la información obtenida se encuentra disponible para crear un Nuevo Proyecto en Gambas, presionar Ok (fig 1.7 Gambas)

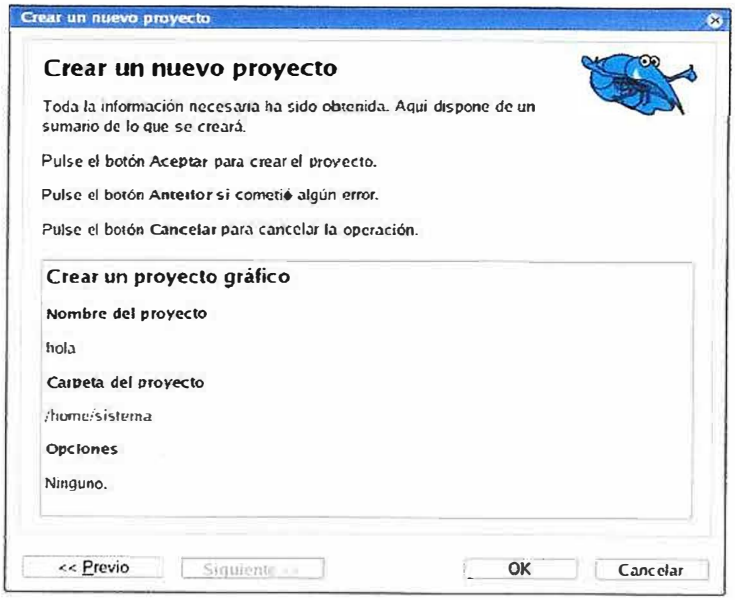


Figura1.7 ventana que la información del proyecto en Gambas.

Entonces deberá aparecer el entorno de desarrollo de gambas que en la cabecera del proyecto tiene el nombre del proyecto que ha creado, con el árbol del proyecto que presenta la opciones para crear clases, formularios, módulos y otros datos del proyecto, también se activa la barra de herramientas (fig1.8 Gambas)

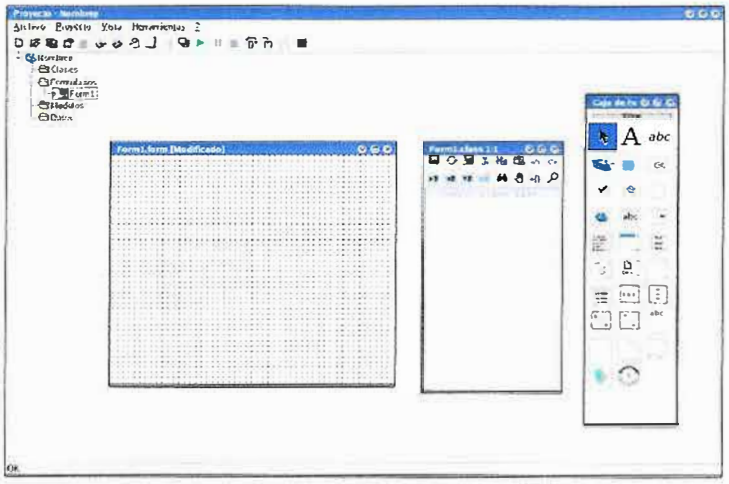


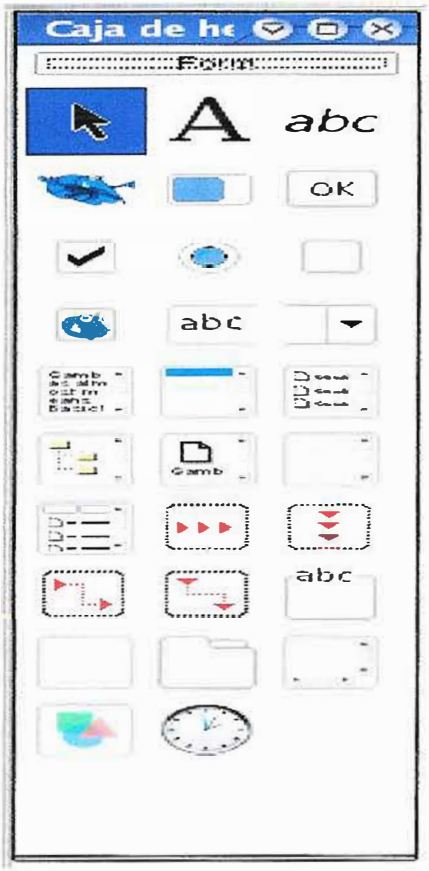
Figura1.7 ventana del proyecto en Gambas con el editor de clases, caja de herramientas y formularios.

1.10 ENTORNO DE DISEÑO EN GAMBAS

1.10.1 LA CAJA DE HERRAMIENTAS DE GAMBAS

En la caja de herramientas de gambas están contenidos los controles que son usados para diseñar las interfaces de usuario y a través de ellos lograr establecer la gestión de tareas de una aplicación en particular.

Los diferentes controles que se pueden encontrar en la caja de herramientas son los siguientes:

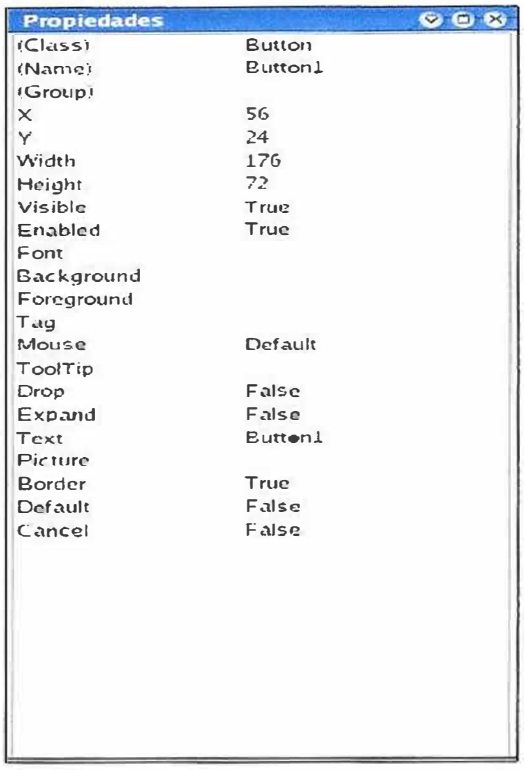


- **Label**: Muestra un texto como una etiqueta, que o permite que el usuario pueda modificar en tiempo de ejecución.
- **Imagen**: Permite almacenar y mostrar una imagen, donde se puede adaptar el tamaño de la imagen especificada en el control, aumentándolo o disminuyéndolo de tamaño.
- **ProgressBar:** Es una barra de estado de procesos que proporciona al usuario una herramienta para el seguimiento de la ejecución de un determinado proceso de una aplicación.
- **Checkbox:** Son casillas de verificación que permite la elección de uno o varios controles en las aplicaciones. No son excluyentes lo que indica que se pueden elegir varios a la vez.
- **Texbox:** Muestra un cuadro de texto donde los usuarios pueden comunicarse con una tarea especifica de una aplicación.

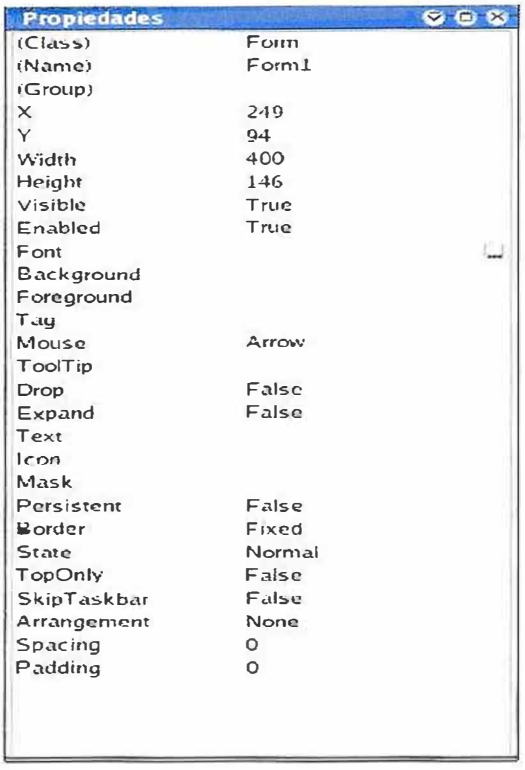
- **Combobox:** Lista un grupo de elementos donde se puede elegir uno o varios controles. Permite incluir nuevos elementos en tiempo de ejecución.
- **Listbox:** Puede mostrar una lista de elementos donde los usuarios puedan elegir uno o varios a la vez.
- **Timer:** Permite que se pueda ejecutar tareas dependiendo de los valores que se puedan establecer en este control para tener una demora o intervalo de tiempo de este control temporizador.
- **Frame:** Permite incluir controles del mismo tipo, organizados independientes entre si, de otros de su misma clase dentro de un mismo formulario.
- **TabStrip:** Es un control que permite incluir sobre un formulario de un cuadro de dialogo con varias paginas.
- **ListView:** Es un control que muestra elementos en una de cuatro vistas diferentes. Puede organizar elementos en columnas con o sin encabezados, mostrar iconos y textos adjuntos.
- **ScollBars:** Control que permite activar una barra de desplazamiento (Horizontal o vertical), ambas barras o ninguna.

1.10.2 PROPIEDADES DE LOS CONTROLES Y FORMULARIOS

Una de las ventanas importantes para gambas es la de las propiedades de los controles y los formularios donde se puede determinar los valores o acciones que estos requieran.



Propiedades	
(Class)	Button
(Name)	Button1
(Group)	
X	56
Y	24
Width	176
Height	72
Visible	True
Enabled	True
Font	
Background	
Foreground	
Tag	
Mouse	Default
ToolTip	
Drop	False
Expand	False
Text	Button1
Picture	
Border	True
Default	False
Cancel	False



Propiedades	
(Class)	Form
(Name)	Form1
(Group)	
X	249
Y	94
Width	400
Height	146
Visible	True
Enabled	True
Font	
Background	
Foreground	
Tag	
Mouse	Arrow
ToolTip	
Drop	False
Expand	False
Text	
Icon	
Mask	
Persistent	False
Border	Fixed
State	Normal
TopOnly	False
SkipTaskbar	False
Arrangement	None
Spacing	0
Padding	0

Las propiedades de los formularios o controles permiten tener una visión del aspecto que estos deben presentar en tiempo de ejecución, después de haberle asignado un valor o acción. Se pueden modificar su estilo, aspecto, el nombre que presenta, color que presenta, altura, anchura entre otros.

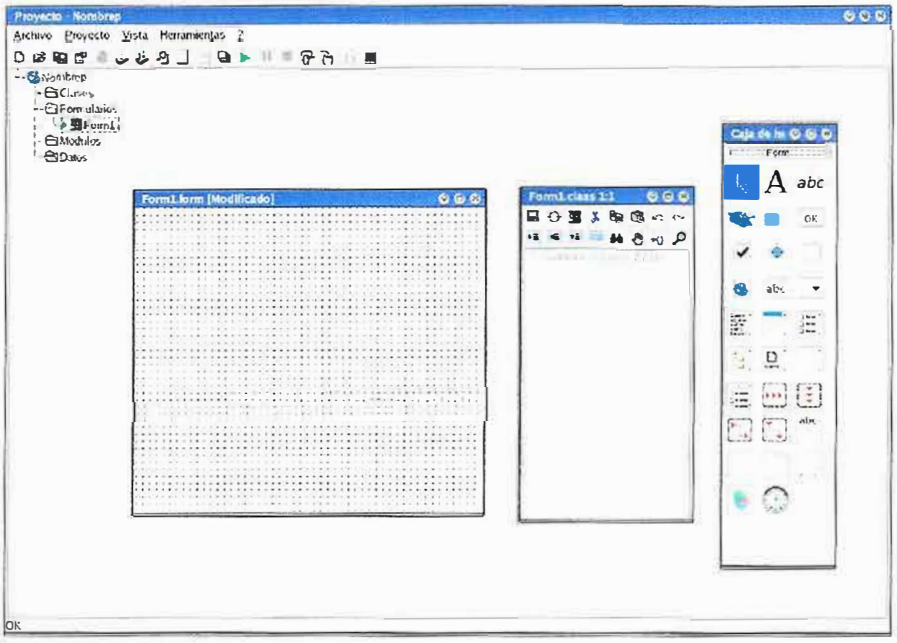
1.10.3 LA VENTANA DE UN PROYECTO EN GAMBAS

La ventana de un proyecto en gambas le presenta a los usuarios un conjunto de herramientas que pueden ser utilizados en un proyecto. Estas herramientas que se muestran pueden ser formularios, clases, controles, módulos, datos, entre otros. Estas herramientas se presentan en una lista en esquema y

organizadas por niveles, que se despliegan cuando presionas con clic sobre cualquier elemento, que deseas visualizar.

Los proyectos de Gambas se definen con un directorio que tiene un archivo.project dentro, y todos los archivos en ese directorio.

Gambas utiliza dos archivos separados para trabajar la creación del objeto formulario con archivo.form y el código de cada formulario con un archivo.class.



En la ventana se puede apreciar las herramientas que se activan cuando se crean un proyecto Nuevo.

1.10.4 EDITOR DE CÓDIGO DE GAMBAS

El editor de código de Gambas es donde se deberá realizar la escritura de código fuente de los formularios, módulos y controles que hacen parte principal de la interfaz grafica de usuario y responder a los sucesos o eventos. Para activarla puede ubicarse con el botón activo del ratón sobre cualquier control

que se quiera escribir código, presionar doble clic, notara que aparecerá el editor de código y en el inicio de este contiene palabras claves o palabras reservadas propias del control que identifican las funciones o procedimientos (se deben observar en **Negrilla**).En el editor de código de Gambas se puede copiar, pegar, borrar, entre otros (fig1.9 Gambas).

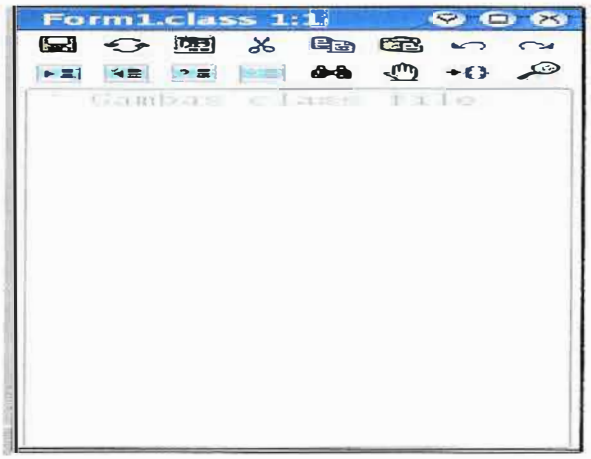


Figura1.7 ventana del editor de código en Gambas.

1.11 ELEMENTOS DE UN PROGRAMA EN GAMBAS

Se puede decir que un programa esta formado por formularios, módulos, controles, datos ente otros. Pero toda aplicación realizada debe incluir un grupo

de métodos, funciones, eventos, variables que ayudaran a realizar operaciones y obtener resultados.

La interfaz gráfica puede ofrecer una serie de eventos para modelar el comportamiento de un programa, no obstante, podemos crear nuestros propios eventos personalizados, y no sólo en programas gráficos, si no también en programas de consola, ya que la gestión de eventos de Gambas es independiente del entorno gráfico.

El código de cada módulo está parcialmente aislado del código de todos los demás, y de igual modo sucede con los formularios y clases y puede ser posible tener dos módulos que contengan una función con el mismo nombre.
Las variables pueden ser declaradas al principio del módulo, formulario o clase, antes de cualquier de realizar cualquiera función.

1.12 LOS MÉTODOS ESPECIALES DE GAMBAS

Los métodos especiales de Gambas son métodos declarados en las clases, cuyo nombre comienza con un carácter guión abajo y son llamados por el intérprete en algunas situaciones especiales.

1.12.1 MÉTODOS ESPECIALES

- **_init**
- **_exit**
- **_new**
- **_free**
- **_next**
- **_get**
- **_put**
- **_call**
- **_unknown**

EL MÉTODO ESPECIAL _init

El método **_init** es llamado cuando las clases son cargadas por el intérprete. Este método puede ser declarado con la forma:

GB_STATIC_METHOD (" _init ", NULL, MyClass_init, NULL)

Utilice este método para una inicialización específica de clases.

EL MÉTODO ESPECIAL _exit

El método **_exit** es llamado cuando las clases son descargadas por el intérprete. Este método puede ser declarado con la forma:

GB_STATIC_METHOD (" _exiit ", NULL , MyClass_exit , NULL)

Utilice este método para una salir de una clase específica.

EL MÉTODO ESPECIAL _new

El método **_new** es llamado cuando un nuevo objeto de la clase son creados. Este método puede ser declarado con la forma:

GB_METHOD ("_new ", NULL, MyClass_new, parameters)

Este método puede tomar cualquier valor en los parámetros que se le asigne y retorna valor de cero.

El método _new decide que parámetros debe utilizar para los nuevos operadores de parámetros.

Utilice este método para inicializar nuevamente la creación de un objeto.

Nota: Todos los campos de los objetos estructurados son seleccionados en cero.

EL MÉTODO ESPECIAL _free

Este objeto puede ser utilizado cuando un nuevo objeto de las clases se destruye, Este método puede ser declarado con la forma:

GB_METHOD ("_free ", NULL, MyClass_free, null)

Este método no toma ningún parámetro y retorna cero.
Utilice este método para limpiar un objeto por ejemplo: Los objetos estructurados contienen referencias para los valores String o los objetos.

Otra forma se puede decidir crear los objetos en memoria.

EL MÉTODO ESPECIAL _next

El método _next puede ser utilizado en Gambas para los programas que utilizan para cada enumeración de iteración del objeto. Este método puede ser declarado con la forma:

GB_ METHOD ("_next ", return type, MyClass_next, null)

Otra forma de declarar este método es:

GB_STATIC_METHOD ("_next ", return type, MyClass_next, null)

Este método no recibe parámetros y puede retornar datos enumerados, los métodos pueden ser estáticos, las clases deciden si pueden ser enumerados, los objetos o no.

Este método puede retornar el valor cero, se le pueden asignar la enumeración de los objetos con uno para cada instrucción escritos.

Internamente los métodos _next implementados, se pueden utilizar para funciones específicas de Gambas para la programación de interfaces:

GB.GetEnum puede dar el punto para la enumeración en el buffer. Utilice los buffer para almacenar los estados de la enumeración.

Se pueden almacenar solamente 12 bytes dentro del buffer.

Nota: Los buffer son inicializados en cero al principio de la enumeración, así que se pueden detectar los estados particulares de la enumeración.

Cuando la enumeración es alcanzada, se pueden usar funciones **GB.StopEnum** para indicarle a la intérprete y retornar inmediatamente con la implementación de la función.

EL MÉTODO ESPECIAL _get

Con los métodos especiales _get son utilizados con los operadores [], que son usados en los objetos o en las clases para insertar datos.

Este método puede ser declarado con la forma:

GB_ METHOD ("_get ", return type, MyClass_get, parameter)

Otra forma de declarar este método es:

GB_ STATIC_METHOD ("_get", return type, MyClass_get, parameter)

Los métodos _get pueden ser estáticos, los operadores [] toman la decisión de si son utilizados en las clases o en los objetos.

El método puede tomar cualquier parámetro, estos parámetros pueden ser declarados entre los operadores [Ejemplo]. Por ejemplo la instrucción **Val= MyObject [X , Y]** decide la causa del llamado del método _get con **X** y **Y** como parámetros.

Este método puede retornar los valores de los datos extraídos de los objetos o las clases.

EL MÉTODO ESPECIAL _put

Con los métodos especiales _put son utilizados con los operadores [], que son usados en los objetos o en las clases para insertar datos.

Este método puede ser declarado con la forma:

GB_ METHOD ("_put ,null, MyClass_put, parameter)

Otra forma de declarar este método es:

GB_ STATIC_METHOD ("_put", null, MyClass_put parameter)

Los métodos _put pueden ser estáticos, los operadores [] toman la decisión de si son utilizados en las clases o en los objetos.

El método puede tomar cualquier parámetro, estos parámetros pueden ser declarados entre los operadores [Ejemplo]. Por ejemplo la instrucción **MyObject [X , Y] =Val** decide la causa del llamado del método _put con **X** y **Y** como parámetros.

Este método puede retornar los valores de los datos extraídos de los objetos o las clases.

EL MÉTODO ESPECIAL _call

Con los métodos especiales _call son utilizados en las clases o objetos que son utilizados en una determinada función.

Este método puede ser declarado con la forma:

GB_ METHOD ("_call , return type , MyClass_call, parameter)

Otra forma de declarar este método es:

GB_ STATIC_METHOD ("_call", return type , MyClass_call, parameter)

Los métodos _put pueden ser estáticos, las clases pueden ser componentes para usarlos en una función, no en los objetos.

El método puede tomar cualquier parámetro y retornar cero,

EL MÉTODO ESPECIAL _unknown

Con los métodos especiales _call pueden ser utilizados cuando los intérpretes no hallan un método o propiedad de la clase para la declaración de símbolos.

Este método puede ser declarado con la forma:
GB_ METHOD ("_unknown", "V" , MyClass_ unknown, " . ")

Este método designa una variable para el numero de argumentos y retornar un valor variante (Variant value).

Para la implementación de funciones pueden declararse:

- Los **GB.IsProperty** indica cual puede ser el los métodos para **_unknown** se deben utilizar para los símbolos usados en una propiedad o un método.
- Los **GB.GetunKnown** indica cual puede ser el nombre de los métodos para **_unknown** se deben utilizar para los símbolos.
- Los **GB.NParam** indica el retorno de valores de los números de argumentos usados en las funciones.

La aplicación de las clases de los componentes **qb.qt.kde** puede ser una opción buena de usar:

Con el llamado de DCOP los interpretes no conocen los métodos existentes, cuales son los parámetros y así si el método especial **_unknown** es el adecuado.

1.13 DIFERENCIAS REFERENTES A GAMBAS

- En Visual Basic el código de cada formulario y el objeto formulario están incluidos en el mismo archivo. Gambas usa dos archivos separados para ello: archivo.form y un archivo.class.

- Extensiones de los ficheros:

Visual Basic	Gambas	Tipo de Archivo
.vbp	.project (en un directorio individual)	Archivo de definición del proyecto
.bas	.module	Módulo
.cls	.class	Archivo de clase
.frm	.form	Archivo de definición del formulario
.frx	(lo que te de la gana)	Archivos de recursos binarios

- Los proyectos de Gambas se definen con un directorio que tiene un archivo.project dentro, y todos los archivos en ese directorio. Visual Basic permite tener múltiples archivos de proyecto en distintos directorios, o usar el mismo archivo fuente de un directorio en distintos proyectos, lo que tiene sus ventajas y sus inconvenientes.

- Las medidas de la pantalla se hacen en Visual Basic en "twips", que son un 1/1440 de pulgada; en Gambas se hacen en píxeles reales.

- Los controles de los formularios son privados por defecto en los programas de Gambas. Se puede cambiar este comportamiento en las Propiedades del Proyecto, marcando el recuadro "Los controles del formulario son públicos".

- En versiones anteriores de Gambas a la 0.97, el operador de división / devolvía un entero cuando sus operandos eran todos enteros. Por ejemplo, PRINT 9 / 2 devolvía 4 y no 4.5. Desde Gambas 0.97 la división funciona más o menos igual que la división de Visual Basic, con el operador \ disponible para forzar una división de enteros.

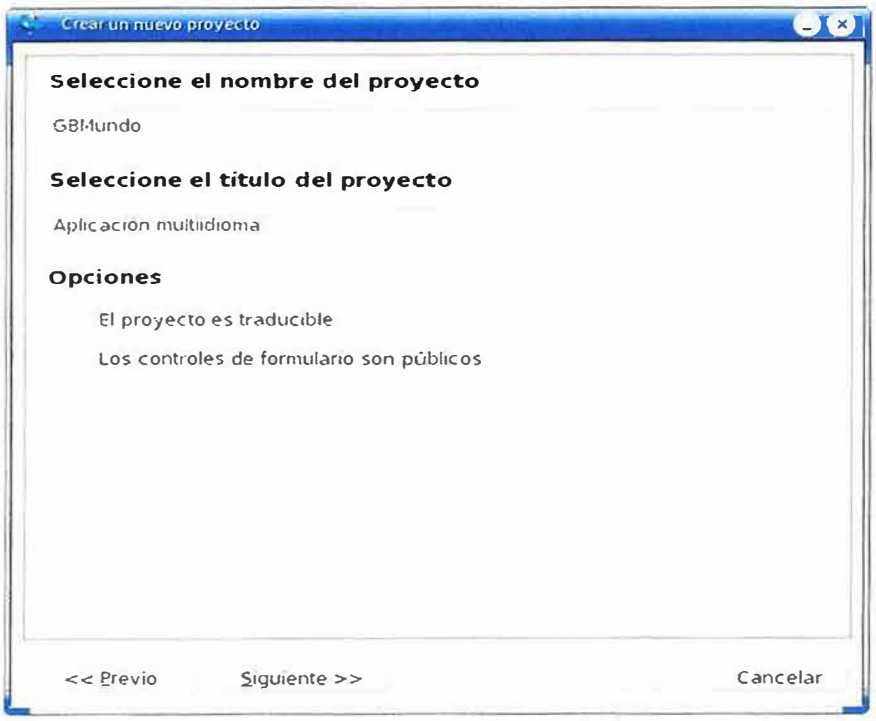
Las funciones de conversión Str$(), Val(), CStr()... se comportan de forma distinta. Por ejemplo, Str$() y Val() usan la configuración de idioma puesta en Gambas, mientras que no hacen eso en Visual Basic. Lea la documentación atentamente para más detalle, pero parece que el comportamiento de Gambas es más lógico: -).

1.14 APLICACIONES MULTIIDIOMA EN GAMBAS

Gambas incorpora soporte nativo para crear aplicaciones que se distribuyan entre usuarios que hablan idiomas distintos. Un proyecto ha de ser marcado como "Traducible" para que esto sea posible.

1. En una nueva aplicación

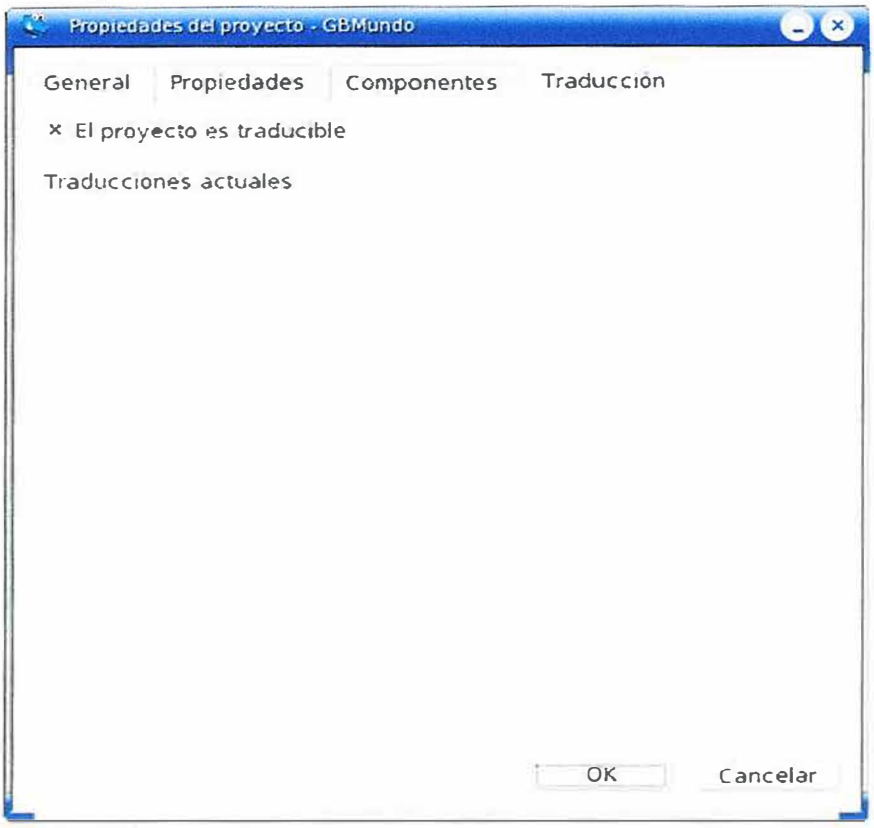
1. Crea un nuevo proyecto gráfico con Gambas y detente en el punto donde se solicita el nombre del proyecto. Observarás que puedes marcar un "check" que indica que el proyecto es traducible. Márcala.



2. El proyecto gestionará de forma automática los textos a partir de ese momento.

2. En una aplicación ya existente

1. Acude al menú "Proyecto" y dentro de este pulsa "Propiedades".
2. En el asistente de propiedades del proyecto, acude a la pestaña "Traducción" y marca la opción "El proyecto es traducible". Acepta los cambios.



3. Trabajo con aplicaciones "traducibles"

1. Todos los textos de la interfaz gráfica, se gestionan de forma automática, de modo que al acudir al gestor de traducciones, los encontrarás disponibles tras cualquier cambio o adición en la interfaz.
2. En cuanto a los textos del programa, hemos de indicar explícitamente que deseamos tenerlos disponibles para su traducción, ya que podría tratarse de textos que no se deben traducir, como, por ejemplo, cadenas SQL, o etiquetas de XML. Si en un código no traducible, indicamos las cadenas como en este ejemplo:

```
3. PUBLIC SUB Button1_Click()
4.
5.     Message.Info ("Hola Mundo")
6.
```

```
7. END
```

En el caso de un texto ha traducir, hemos de delimitar la cadena textual con signos de paréntesis:

```
PUBLIC SUB Button1_Click ()

  Message.Info ( ("Hola Mundo") )

END
```

El compilador sabe distinguir automáticamente entre los paréntesis que corresponden al método, y los referidos a la cadena.

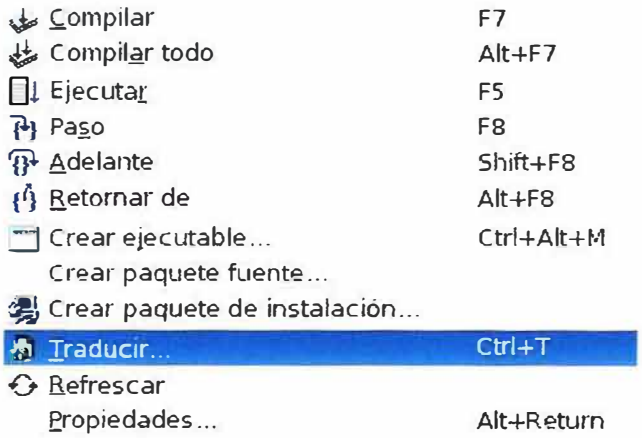
8. Por último, ten en cuenta que hay signos especiales o "de escape" en Gambas; así, por ejemplo, el símbolo " & " se utiliza para indicar en la interfaz gráfica, que esa tecla será el atajo de teclado para acceder al menú o botón, por ejemplo. Otros símbolos tienen significado similar al que tienen en C o algunas shells, como es el caso de "\n" (retorno de carro). el gestor de traducciones, es capaz de comprobar la coherencia de caracteres especiales escritos en la versión original, y en la traducción.

1.15 PROCESO DE TRADUCCION EN GAMBAS

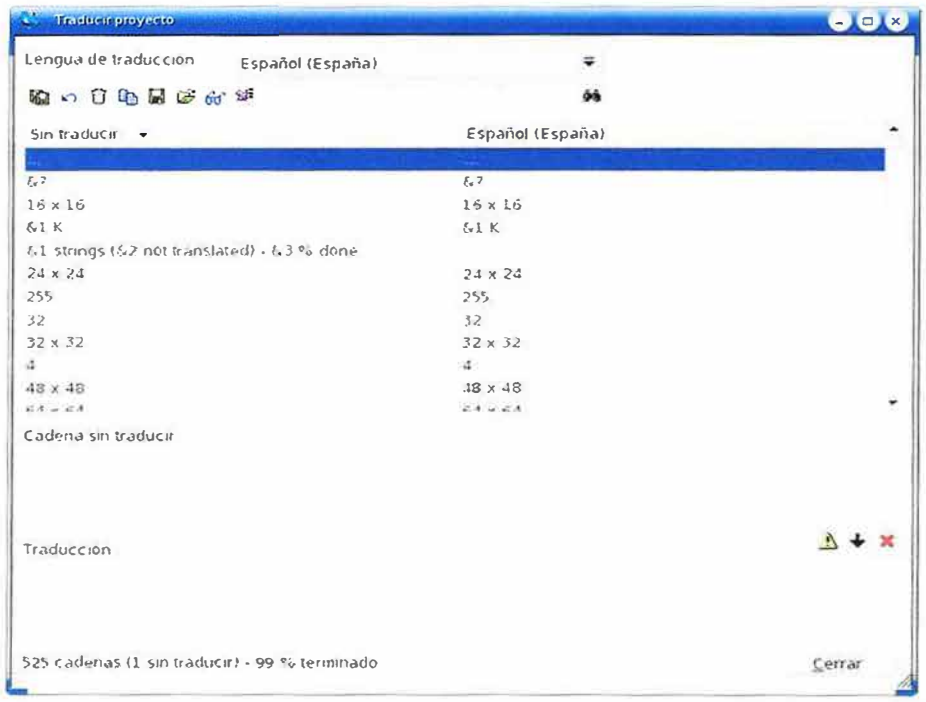
Vamos a tomar como ejemplo el propio entorno de desarrollo Gambas. Cuando descargamos una versión de Gambas en paquete fuente, y descomprimimos el paquete tar.bz2, se genera una carpeta llamada gambas-X.XX (X.XX depende de la versión). Dentro de esta carpeta, hay dos subcarpetas:

- ✓ **app/gambas:** contiene el propio entorno de desarrollo.
- ✓ **app/gambas-database-manager:** contiene el gestor de bases de datos.

Pues bien, ejecutemos Gambas, indiquemos "Abrir proyecto", busquemos la carpeta gambas-X.XX/app/gambas, y abramos este proyecto. A continuación, acudamos al menú "Proyecto", seleccionando la opción "Traducir".



Una vez seleccionada la opción, entramos en el asistente de traducción. Hemos de elegir el idioma que deseamos traducir. Si ya habíamos trabajado previamente en ella, aparecerán los textos sin traducir, así como los ya traducidos. Todas las cadenas sin traducir, así como las introducidas con posterioridad a la última fase de traducciones, aparecerán con la traducción en blanco.



Cada vez que pulsamos con el botón izquierdo en una de las cadenas en la primera lista, en las listas inferiores se nos muestra la cadena original, y en la parte inferior disponemos de la caja de texto para escribir la traducción correspondiente. Cuando cambiamos a otra cadena, la que habíamos editado queda guardada durante esta sesión (es necesario pulsar "Guardar traducción" para que quede guardado permanentemente).

El resto de las opciones son las siguientes:

- Guardar traducción: los cambios realizados en esta sesión, se guardan y quedan disponibles para los usuarios del programa en ese idioma.

- Recargar traducción: se borran todos los cambios realizados en la última sesión, y se vuelve al estado anterior.

- Borrar traducción: elimina todos los textos traducidos.

- Duplicar una traducción: hace una copia de la traducción que había en otro idioma, de modo que nos sirva como base para el idioma a traducir. Puede ser útil, por ejemplo, para hacer una traducción en "Español (variante Argentina)", basada en la original "Español (variante España)". Lo mismo ocurre con el Francés de Bélgica o Canadá respecto al de Francia, que puede llegar a ser confuso según la nacionalidad del usuario, si no se realizan adaptaciones.

- Exportar una traducción : convierte la traducción en un fichero estándar tipo ".por", usado por la mayoría de los programas escritos en C y C++ sobre GNU/Linux, FreeBSD y software libre para otros sistemas operativos. este formato permite enviar la traducción realizada a otras personas, en un sistema de desarrollo donde se trabaje en diferentes lugares. Este es el caso de Gambas: una vez escrita la traducción, se exporta a formato .por, y se envía a la lista de desarrollo de Gambas.

- Importar una traducción: proceso inverso al anterior. El coordinador recibe un fichero ".por", y lo absorbe para incorporarlo a la versión de programa.

- Verificar traducción: imprescindible antes de dar por buena una nueva versión, comprueba que todos los símbolos especiales (por ejemplo '&') coinciden entre los textos originales y los traducidos. No

debe utilizarse, ni exportarse nunca, una traducción en la cual el verificador da un mensaje de error.

- ❖ Buscar la siguiente cadena no traducida: nos dirige directamente a la siguiente cadena pendiente de traducir desde la anterior traducción, de modo que sea más fácil traducir conforme se incorporan nuevas cadenas al programa.

1.16 EVENTO KEY PRESS

Con Gambas se puede utilizar la clase 'Key' para conocer las teclas que pulsa el usuario sobre cualquier control gráfico. Cada control, posee dos eventos para indicarnos la acción del usuario:

- ✓ **KeyPress :** se activa cuando el usuario aprieta la tecla
- ✓ **KeyRelease :** se activa cuando el usuario suelta la tecla

Estos eventos no disponen de parámetros, ya que la clase 'Key' almacena en cada momento el código de la tecla que se acaba de apretar o soltar. Es una clase estática, por la tanto no es necesario que creemos ningún objeto de esa clase para utilizarlo:

```
...
PUBLIC SUB Button1_KeyPress ()

   Message.Info (Key. Code)

END
...
```

Cuando se pulse una tecla con el foco situado sobre el botón, se mostrará un mensaje con el código de la tecla pulsada. Si pulsa por ejemplo la flecha

derecha, devolverá el valor 4116. Con ello ya se sabe que ese código corresponde a la flecha derecha. Si se desea que el evento responda sólo a la pulsación de una/s determinada/s teclas:

```
...
PUBLIC SUB Button1_KeyPress ()

   IF Key. Code = 4116 THEN
      Message ("Has pulsado la tecla de flecha derecha")
   END IF

END
...
```

También podemos valernos de las constantes que aporta la clase "Key" para olvidarnos de los extraños códigos numéricos de cada tecla:

```
...
PUBLIC SUB Button1_KeyRelease ()

   IF Key. Code = Key. Right THEN
      Message ("Has soltado la tecla de flecha derecha")
   END IF

END
...
```

Por último, el evento KeyPress es cancelable, de modo que podemos evitar que la pulsación de una tecla llegue al programa, para cancelar, por ejemplo, una acción del usuario no admitida en ese momento:

```
...
PUBLIC SUB Button1_KeyPress ()

   IF key. Code <> key. Tab THEN STOP EVENT

END
...
```

Se observa en el ejemplo, que se evita la cancelación del evento cuando la tecla "Tab" es pulsada, dada la importancia de esta tecla para moverse por los formularios sin necesidad del ratón.

La clase "Key" aporta también los valores 'Alt.', 'Code', 'Control'y 'Shift' para saber si estas teclas estaban pulsadas a la vez que la principal.

1.17 VARIABLES

Los programas almacenan la información de uso inmediato en memoria RAM. Trabajando al más bajo nivel, se deposita la información indicando la dirección de la memoria RAM directamente. Esto es muy engorroso, especialmente cuando el número de datos a almacenar es muy grande.

Para solucionarlo, los lenguajes de programación aportan el concepto de "variable": es nombre al cual nos referimos para almacenar información u obtenerla, de direcciones de memoria que se gestionan sin que a nosotros nos afecte directamente.

Nosotros indicaremos un nombre, por ejemplo "MiValor", asignaremos valores con el operador " = " (MiValor = 3), y haciendo referencia a ese nombre

podremos leer su contenido (PRINT MyValor, mostraría en pantalla "3" en este ejemplo).

Según el lenguaje de programación se habla de que no está "tipado", o que está "tipado", en función del modo en que usa las variables. En un lenguaje "no tipado", cada variable puede almacenar cualquier cosa, (un número, una cadena de texto, etc.). Esto puede estar bien para programar muy rápido, pero da lugar a graves errores en aplicaciones medianas y grandes. Gambas, por el contrario, es un lenguaje "tipado", y esto significa que de modo previo al uso de cualquier variable, forzosamente hemos de "declararla", especificando el tipo de datos que va a contener. Estos son algunos ejemplos de "declaraciones" con Gambas:

```
DIM MyVar As integer
PRIVATE Respuesta As String
PUBLIC Suma As Float
```

En estos ejemplos, y sin que nos interese ahora la primera parte de estas declaraciones (DIM, PRIVATE,PUBLIC), estamos indicando que deseamos utilizar una variable llamada "MyVar" que almacenará números enteros, otra llamada "Respuesta" que almacenará cadenas de texto, y otra llamada "Suma" que almacenará números con coma flotante (es decir, es capaz de almacenar números con parte decimal).

1.18 FUNCIONES

En los primeros tiempos de la informática, y en realidad, en la actualidad los microprocesadores siguen trabajando así, los programas eran una ristra larguísima de ordenes, una detrás de otra, sin más posibilidad de "estructuración" que dar saltos hacia a delante o hacia atrás en el código.

Más adelante se impuso un modelo más "estructurado": dividir el código en bloques separados, llamados "funciones" o "métodos". Un programa completo estará formado por funciones, pequeños bloques de código a los que se le envían una serie de valores de los cuales se hace responsable para procesarlos, y retornar un valor, si procede.

Cada programa en Gambas empieza con una función "Main", o principal, y a lo largo de su código, esta función envía datos a otras funciones, y, si es necesario, recibe sus resultados. Este proceso se denomina "llamada", una función "llama" a otra función, esta puede "llamar" a otras, etc. Los valores que se envían para ser procesados, se denominan "parámetros".

Para indicar a Gambas el inicio de una función, emplearemos básicamente esta nomenclatura:

```
PUBLIC FUNCTION nombre_de_la_funcion (p1 As TipoDato, p2 As TipoDato...) As TipoDato

o bien

PUBLIC SUB nombre_de_la_funcion (p1 As TipoDato, p2 As TipoDato...)
```

Respecto a la palabra "PUBLIC", no nos preocuparemos por ahora, pero observa que podemos indicar "FUNCTION" o "SUB". La primera palabra significa que el código de la función va a retornar algún valor, mientras que "SUB" significa que se trata de una función que no retornará ningún valor, como fue el ejemplo de nuestro segundo tutorial, donde emitíamos un mensaje, pero no retornábamos ningún valor.

nombre_de_funcion: Es el nombre con que llamaremos a la función desde otras partes del código, y será un identificador único para ese fragmento de código en todo el programa.

(P1 As TipoDato, p2 As TipoDato...): A continuación, se indican las variables que se pasarán como parámetros, separadas por comas, e indicando el tipo de la variable, del mismo modo que en las declaraciones antes explicadas. Según lo que vaya a hacer la función emplearemos más o menos parámetros.

As TipoDato: Si hemos indicado la palabra clave "FUNCTION", indicaremos al final el tipo de dato que vamos a devolver. Dependiendo del tipo de parámetros que utilicemos, estos pueden también utilizarse en ocasiones para devolver el resultado deseado, al margen de este valor retornado.

Unos ejemplos:

Una función que no devuelve ningún valor y no necesita parámetros:

PUBLIC SUB Mi función ()

Una función que no devuelve ningún valor y necesita un valor entero como parámetro:

PUBLIC SUB Otra función (MiValor As Integer)

Una función que devuelve una cadena de texto y necesita un valor entero y un valor de cadena:

```
PUBLIC FUNCTION OtraMas (MiDato As Integer, MiCadena As String) As String
```

ÁMBITO

No todas las funciones y variables se utilizan desde todo el código, hay varias razones para limitar las zonas desde donde se puede acceder a una variable o función: desde razones de rendimiento, por ejemplo, es mejor tener variables que se creen y se destruyan tras ser usadas, que tener un "monstruo" en memoria con todas las variables permanentemente almacenadas, hasta razones de comodidad para el programador, ya que es mejor poder definir el mismo nombre para ciertas variables, en zonas "aisladas" del código, que tener que inventar miles de nombres a lo largo de un gran programa.

Gambas permite verías técnicas para separar o aislar el "ámbito" de las variables y funciones. Los programas escritos en Gambas, constan de "módulos", "formularios", y "clases". Pues bien, el código de cada módulo está parcialmente aislado del código de todos los demás, y de igual modo sucede con los formularios y clases. Es posible tener dos módulos que contengan una función con el mismo nombre:

```
Módulo 1:
Public SUB MiFuncion (Dato As Integer)
Módulo 2:
Public SUB MiFuncion (Dato As Integer)
```

Si nos encontramos dentro de otra función del módulo 1, y queremos llamar a la función "MiFuncion", directamente lo indicaremos así en el código:

```
...
MiFuncion (34)
...
```

Al llamar a "MiFuncion", el intérprete sabe que nos referimos a la función que se encuentra dentro del mismo módulo. Ahora bien, si nos encontramos dentro del módulo 2, o de cualquier otro módulo, clase o formulario, hemos de especificar el módulo en el que se encuentra la función. Así, si desde el módulo 2 indicáremos:

```
...
Mifuncion (9654)
...
```

Realmente estaríamos llamando a la función con ese nombre que se encuentra dentro del módulo 2. Hemos de especificar que deseamos utilizar la del módulo 1, indicando el nombre del módulo, un punto, y el nombre de la función:

```
Modulo1.MiFuncion (9654)
```

La segunda técnica para delimitar el ámbito de la función, es indicar al intérprete si deseamos que esa función sea accesible desde otro módulo, formulario o clase. Si indicamos la palabra "PUBLIC", delante de la declaración de la función, esta podrá ser llamada desde otros módulos, clases o formularios, pero si indicamos "PRIVATE", sólo podremos acceder desde el propio módulo (se dice también que "no es visible" desde otros módulos)

En cuanto a las variables, tenemos los mismos casos que con las funciones, y otro más:

Las variables pueden ser declaradas al principio del módulo, formulario o clase, antes de cualquier función, en este caso emplearemos las palabras "PRIVATE" y "PUBLIC" de igual modo que con las funciones, y estas variables existirán mientras exista el módulo, clase o formulario. Pero también podemos indicar una variable dentro de una función, empleando ahora la palabra clave "DIM", y en este caso la variable sólo existe desde que se entra en la función, hasta que se sale, y no es accesible en absoluto desde ninguna otra zona del código fuera de la función.

```
PRIVATE Var1 As Integer
PUBLIC Var2 As String

PUBLIC FUNCTION SumaEspecial (V1 As Integer, V2 As Integer) As Integer

  DIM Media As Integer
  Media= (V1+V2)/2
  RETURN Media+V1+V2
END
```

1.19 CONEXIÓN CON BASES DE DATOS

En este manual[1] contiene una aplicación sencilla con base de datos MySQL, para que todos los programadores que quieran desarrollar Software en Linux

[1] POLO, Steven, MANUAL DE GAMBAS. Pagina 34.

encuentren en aquí una alternativa. A continuación vamos a crear una pequeña aplicación con una base de datos MySQL, en la cuela se podrá:

- Guardar un registro
- Actualizar un registro
- Eliminar un registro
- Buscar un registro
- Examinar todos los datos de la base de datos

Adjuntando el código fuente explicado paso a paso, se asume que el lector ya tiene los conocimientos
Previos de programación y de Visual Basic
Para el siguiente ejemplo se trabajará con una tabla llamada áreas, que representan las diferentes áreas de conocimiento de un colegio que puede ser adaptado de acuerdo a su necesidad.

NOMBRE: areas		
CAMPO	TIPO	
CodArea	varchar(5)	primary key
Nombre	varchar(50)	not null

1. Crear un modulo: Un modulo es un archivo que contiene todas las variables globales y/o funciones que puedes utilizar en todos los formulario del proyecto.
Para crear el modulo nos vamos al navegador de clases y hacemos clic derecho y escogemos la opción
Nuevo>modulo, y le asignamos el nombre de Mglobal .
Si puedes observar se creó un archivo llamado **Mglobal .module**

Al hacer doble clic sobre ese archivo (el Modulo nuevo creado) se mostrará la ventana de códigos.

en ellas escribirás el siguiente código:

```
'Variables globales
PUBLIPUBLIC db AS NEW Connection 'Para
PUBLIC rs AS Result
PUBLIC strSQL AS String
'Funcion para hacer la Conexión a la base de datos '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
PUBLIC FUNCTION Conexión () AS Boolean
db.close ' Se Cierra la conexión con la base de datos
'Datos de la conexión
'si la conexiones es fija debe configurarla aquí como esta
'tu debes cambiar los parámetros de acuerdo a tu conexión
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Mete los parámetros globales para a conexión con la BD
db.Host = "localhost" ' Nombre del Host
db.Type = "mysql" ' Tipo de base de datos
db.Name = "colegio" ' Nombre de la Base de Datos
db.Login = "root" ' Nombre del usuario
db.Password = "" ' Contraseña de acceso
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
db.Open
RETURN TRUE
CATCH
Message.error (ERROR. text)
RETURN FALSE
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Todas las variables y funciones se deben declarar como públicas para que puedan ser utilizadas por los demás formularios, aquí se definen las siguientes variables:

db: de tipo conexión para abrir y cera la conexión con la base de datos

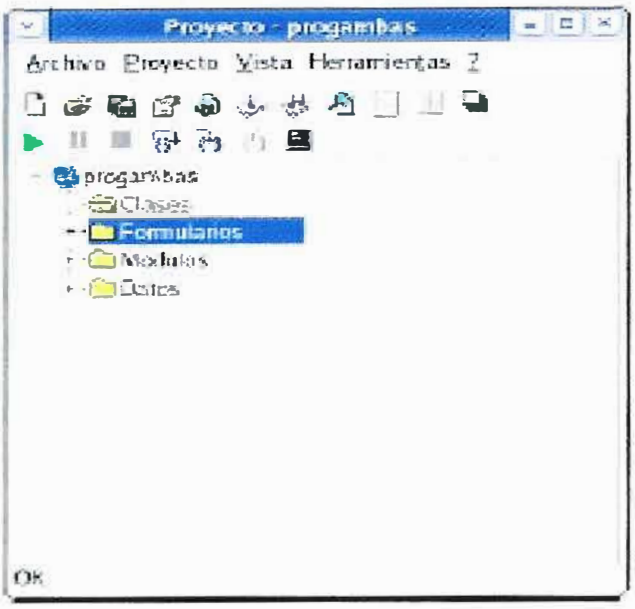
rs: para poder realizar las consultas

StrSQL: Variable de tipo String para colocar aquí todos los consultas SQL como (IINSERT, DELE,Etc),

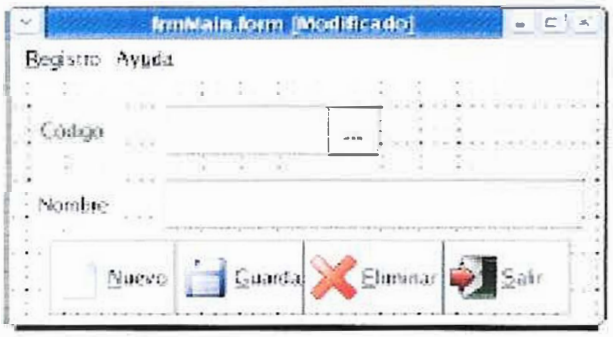
Luego viene la función conexión () que es de tipo booleana que retorna true si se realizó la conexión o false si ocurre un error mostrando el respectivo mensaje en el catch.

2. Crear el formulario principal:

Para crear el formulario principal se hace clic derecho sobre el inspector de proyecto y se crea un nuevo formulario como muestra la figura.



y creamos un nuevo formulario, le asignas un nombre. En nuestro caso se llamará frmMain, el cual debe quedar de esta forma.



Vamos a realizarlo paso a paso.

2.1. Probar la conexión con la base de datos: Para eso hacemos doble click pre el área del formulario y nos lleva al código fuente en el evento Open (). El cual nos debe quedar de la siguiente manera:

```
PUBLIC SUB Form_Open ()
IF (NOT Mglobal.conexion ()) THEN
ME.Close ()
RETURN
END IF
txtcodigo.SetFocus ()
END
```

En el if se llama a la función conexión () se evalúa si retorna falso (es decir si hubo un error),
Sencillamente se cierra el formulario y salimos. Con esta simple instrucción ya puedes ejecutar el Formulario el cual debe mostrar solamente el formulario vació sin ningún mensaje, si esto es así Quiere decir que la conexión se pudo abrir con éxito y podemos avanzar.

2.2. Insertar los controles del Formulario: Se toman los controles de la barra de Herramienta

Objeto Gambas	Propiedad	Valor
TxtBox	Name	txtcodigo
TxtBox	Name	txtnombre
Button	Name	btnnuevo
Button	Name	btnguardar
Button	Name	btneliminar
Button	Name	btnsalir
Button	Picture	btexaminar
Button	Picture	new.png

La propiedad Picture para cada botón se le asigno el icono respectivo, los cuales se encuentran en:

/usr/share/icons/default.kde/32x32/apps/, se deben copiar en el mismo directorio del proyecto para este

Caso los renombre por (save.png, delete.png, exit.png, etc)

2.3. Evento de los Botones: ahora lo que se debe hacer es programar el evento para cada uno de ellos, para el cual se hace doble clic sobre el botón y se escribe el siguiente código

Botón nevó:

```
'Evento de limpiar los controles de texto
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

PUBLIC SUB btnnuevo_Click ()
txtcodigo.Text=""
txtnombre.Text=""
txtcodigo.SetFocus ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Simplemente lo que se hace es limpiar los controles de texto asignándole vació ("") en la propiedad text

Baton guardar:

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

PUBLIC SUB btnguardar_Click ()
IF txtcodigo.Text <> "" AND txtnombre.Text<> "" THEN
'Busca si existe ese código
IF mglobal.buscar ("CodArea","áreas", txtcodigo.Text, 0) THEN
Mglobal.strSQL="UPDATE áreas SET Nombre='" & txtnombre.Text & "'
WHERE CodArea='" & txtcodigo.Text & "'"
ELSE
'Si no existe lo inserta
Mglobal.strSQL="INSERT INTO areas VALUES('" & txtcodigo.Text & "','" &
txtnombre.Text & "')"
END IF
db.Exec(Mglobal.strSQL)
message.Info("Registro Guardado")
btnnuevo_Click
ELSE
message.Warning ("Debe ingresar todos los datos")
END IF
CATCH
Message.error (ERROR. text)
RETURN
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Analicemos que es lo que se hace eb este evento; primeo se valida si los controles de texto no están vació (primer if). Luego continua el siguiente if con la siguiente instrucción:

IF mglobal.buscar ("CodArea","areas", txtcodigo.Text, 0) THEN

Bien buscar () es una función que se debe definir en el modulo principal (global), mas adelante veremos el código ahora solo miremos lo que hace. La función tiene 4 parámetros que son:

"codarea": El nombre del campo en la base de datos como clave para buscar
"areas": Es el nombre de la tabla al que pertenece el campo a buscar
txtcodigo.text: Es el valor del campo a consultar es decir lo que el usuario ingrese en el control de texto 0: es una bandera indica que el campo "codarea" es de tipo String, o 1 si es numérico, en este caso como es de tipo String en la base de datos se ole envía un 0. Esta función retorna true si existe ese registro con esa llave de lo contrario retorna false.

Esta fue diseñada con el objetivo de que si en neutro proyecto va atener mas formulario simplemente se llama la misma función desde los demás formularios Luego de la función buscar(), se realiza la consulta respectiva si esta se actualiza de lo contrario se inserta como un registro nuevo. Con esto nos evitamos el trabajo de de insertar otro botón actualizar.

La función buscar es la siguiente:

''

'Función buscar: busca en la base de datos si existe ese registro en la base de Datos
'es utilizada en todos los formulario en el botón guardar, si esta se actualiza de lo contrario
'se inserta como un registro nuevo
'nomcampo: nombre de campo a buscar
'nomtabla: nombre de la tabla
'valor: valor a consultar

'tipo: puede ser de dos tipos 0 ---> si nomcampo es un String o 1---> si es numérico.

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

PUBLIC FUNCTION buscar (nomcampo AS String, nomtabla AS String, valor
AS String, tipo
AS Integer) AS Boolean
IF tipo = 0 THEN
'Si el nomcampo es de tipo String
StrSQL = "SELECT * FROM " & nomtabla & "WHERE " & nomcampo & "='"&
valor & "'"
ELSE
' Si el nomcampo es de tipo numérico
strSQL = "SELECT * FROM " & nomtabla & "WHERE " & nomcampo & "="&
valor & ""

END IF
rs = db.Exec(strSQL)
IF rs.Count > 0 THEN
'Si existe en la tabla
RETURN TRUE
ELSE
'Si no existe en la tabla
RETURN FALSE
END IF
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Esta debe ser insertada en el modulo principal

Boton Eliminar:

```
PUBLIC SUB btneliminar_Click ()
DIM op AS Integer
```

```
IF txtcodigo.Text<> "" THEN
'valida si desea eliminarlo ----------------------------
Op = message. Warning ("Desea Eliminar este Registro? ","Eliminar","Cancelar")
'
IF op = 1 THEN
IF mglobal.buscar ("CodArea","areas", txtcodigo.Text,0) THEN
Mglobal.strSQL="DELETE FROM areas WHERE CodArea='" &
txtcodigo.Text & "'"
db.Exec (Mglobal.strSQL)
message.Info ("Registro Eliminado!")
btnnuevo_Click
ELSE
message.Warning ("No existe ese registro en la base de Datos")
END IF
END IF
ELSE
Message. Warning ("Debe ingresar El código")
END IF
END
```

Para eliminar un registro basta con que ingrese el código respectivo por eso la instrucción en del primer

if, validando que hallan ingresado el código y haciendo la pregunta respectiva con el message.warnin ()

este ultimo retorna un valor el cual es capturado en la variable op que es

1: si es selecciono el primer botón por default que es eliminar en este caso

2: si presiona OK

3: si es Cancel

Luego que se elimina se llama al evento btnnuevo_Click para limpiar los controles de texto

Botón Salir:

```
PUBLIC SUB btnsalir_Click ()
ME. Close ()
END
```

3. Insertar Menú al proyecto: Para insertar menú solo hay que hacer clic derecho en el área de trabajo del formulario en el modo de diseño y se escoge la opción Editor de Menu

y agregamos el siguiente menú y en los cuadros de texto(nombre y titulo), insertamos los siguientes ítems:

NOMBRE	*TITULO*
Mna	&Registro
mnev	&Nevo
mnguardar	&Guardar
mneliminar	&Eliminar
msale	&Salir

Luego en el formulario hacemos clic sobre cada de las opciones para generar su respectivo evento, el cual ya el código lo tenemos simplemente se invoca cada evento de los botones como sigue

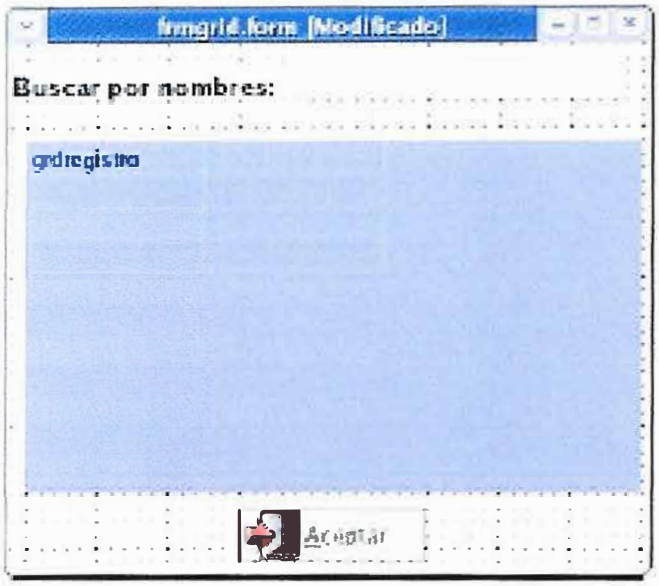
```
' """"Procedimientos del Menu"""""
'opción acerca de
PUBLIC SUB manacer_Click ()
frmautor.Show ()
END
```

```
'Opción Eliminar
PUBLIC SUB mneliminar_Click ()
btneliminar_Click ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Option guardar
PUBLIC SUB mnguardar_Click ()
btnguardar_Click ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Opción Salir
PUBLIC SUB msale_Click ()
ME. Close ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Opción Nuevo
PUBLIC SUB mnuev_Click ()
btnnuevo_Click ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

4. Cargar todos los datos en el grid: Imaginémoslo el siguiente problema ¿Como saber si ya un área fue asignada si no nos sabemos el código?. Lo que se hizo en este caso es crear el botón btexaminar, para que al hacer clic sobre él como abra otro formulario con un grid en donde cargue por default todos las áreas ingresadas ordenadas por nombre, permitiendo además la búsqueda de ellas por nombres.

Para hacer esto debemos crear un nuevo formulario el cual llamaremos frmgrid que se hace como lo vimos en el punto 2,e insertamos los siguientes controles.

Objeto Gambas	*Propiedad*	*Valor*
TxtBox	Name	txtnombre
Label	Text	Buscar por nombres:
Button	Name	btnaceptar
Button	Picture	exit.png
GridWiev	Name	grdregistro
GridWiev	Backround	&HAAAAFF& (Color del grid)



Generamos en le evento Open del formulario haciendo doble clic sobre él y escribimos:

```
PUBLIC SUB Form_Open ()
grdregistro.Columns.Count = 2
grdregistro.Rows.Count = 1
'Para que el encabezado lo coloque en negrilla
grdregistro.Columns.Resizable = TRUE
grdregistro [0,0].Text = "Código"
```

```
grdregistro [0,1].Text = "Nombres "
' De fine el tamaño de la las columnas
grdregistro.Columns [0].Width=52
grdregistro.Columns [1].Width=250
Cargargrid ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Lo primero que se hace es decirle al grid que va a tener inicialmente 2 columnas y una fila y se le asignan los títulos de las dos columnas del grid y se asigna un ancho para cada una de ellas. Luego se invoca un procedimiento local llamado cargargrid (), para que ejecute la consulta al momento de cargar el formulario y llene elgrid con todos los registros.

NOTA: Si su tabla posee muchos registros (miles) no es bueno este método porque seria muy lento es recomendables para un manejo de pocos registros. Para este caso es mejor que se escriba primero la palabra y con un botón o con enter por ejemplo que diga buscar traiga todos los registros que coincidan.

4.1. Procedimiento cargar Gris

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

PUBLIC SUB cargargrid ()
DIM N AS Integer
DIM i AS Integer
IF txtnombre.Text<> "" THEN
mglobal.strSQL = "SELECT * FROM areas WHERE Nombre LIKE '"&
txtnombre.Text & "%' ORDER BY Nombre"
ELSE
mglobal.strSQL = "SELECT * FROM areas ORDER BY Nombre"
END IF
mglobal.rs= mglobal.db.Exec (mglobal.strSQL)
```

```
N=mglobal.rs.Count
IF N > 0 THEN
grdregistro.Clea r()
grdregistro.Font.Bold =TRUE
grdregistro.Font.Size = 14
grdregistro [0,0].Text = "Código"
grdregistro [0,1].Text = "Nombres"
grdregistro.Rows.Coun t=N + 1
grdregistro.Font.Bold=FALSE
grdregistro.Font.Size = 10
FOR i = 1 TO N
Grdregistro [i,0].Text = mglobal.rs!CodArea
Grdregistro [i,1].Text = mglobal.rs!Nombre
mglobal.rs.MoveNext ()
NEXT
END IF
END
```

Este procedimiento comienza validando si se escribe algo en el texto por ejemplo 'a' buscaría todos las áreas que comiencen con esa letra(s), para eso utiliza el operador like de SQL; en caso de que no se escriba nada en el control de texto seleccionaría todos los registros con la consulta y procederá a cargarlos en el grid. Para eso se guarda en la variable N el numero de registros traídos por la consulta para definir de esa manera el numero de filas del grid La instrucción grdregistro.Clear () Limpia el grid grdregistro.Font.Bold =TRUE pone en negrita los títulos del grid luego en el ciclo for se recorrerá todos los registros y se insertan en la tabla en la fila i, columna 0 y 1 con las instrucciones:

grdregistro [i,0].Text = mglobal.rs!CodArea

grdregistro [i,1].Text = mglobal.rs!Nombre

4.2. Evento change del control de texto: Ahora lo que debemos es programar el control de tal manera que a medida que el usuario vaya escribiendo una palabra en el control de texto, cargue automáticamente todos aquellos registros que coincidan con esa ocurrencia de caracteres. Para esto hay que crear el evento change y para hacerlo haga lo siguiente:

- ✓ Seleccione el control de texto txtnombre del formulario frgrid en modo diseño
- ✓ Haga clic derecho sobre este control y seleccione la opción Evento>Changey luego debe aparecer el evento en el código y usted debe agregar la instrucción cargargrid()el cual debe quedar así:

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
PUBLIC SUB txtnombre_Change ()
Cargargrid ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

4.3. Evento DblClick del control GridWiev: Hasta aquí el proyecto ya está funcionando pero

Podemos programarlo de tal manera que al hacer doble clic sobre un registro (una fila), en el grid lo cargue automáticamente en el formulario frmMain para editarlo y realizar las operaciones que queramos sobre ese registro. Para esto hacemos lo siguiente:

- ✓ Seleccione el control GridWiev (grdregsitro), en el formulario frgrid en modo diseño
- ✓ Haga clic derecho sobre este control y seleccione la opción Evento>DblClick y luego debe aparecer el evento en el código y usted

debe agregar la instrucción grdregistro_DblClick () el cual debe quedar así:

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
PUBLIC SUB grdregistro_DblClick ()
frmMain.txtcodigo.Text = grdregistro [grdregistro.Row,0].Text
frmMain.txtnombre.Text = grdregistro [grdregistro.Row,1].Text
ME.Close ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

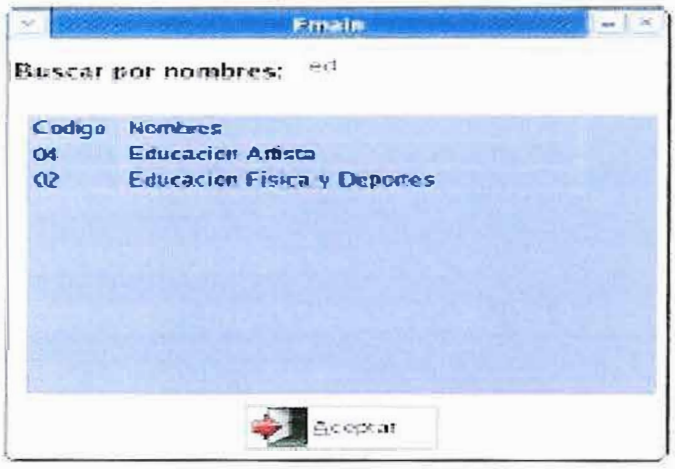
En el cual, se pone en el control de texto txtcodigo del formulario frmMain lo que tiene el grid en la fila actual (grdregistro.Row) con la columna 0, lo mismo se hace con el campo nombre.

4.4. Evento Click del botón de comandos: El botón de comandos btnaceptar la única operación programada es cerrar el formulario por tanto se debe generar el evento click que ya sabemos como generarlo y agregar la instrucción ME.Close() este evento debe quedar así:

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
PUBLIC SUB btnaceptar_Click ()
ME. Close ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Hasta aquí el proyecto esta listo, para ser ejecutado y probado de tal manera que el proyecto en ejecución se verá así:

Y al hacer clic sobre el botón examinar (...) mostrará el otro formulario don de está elgrid



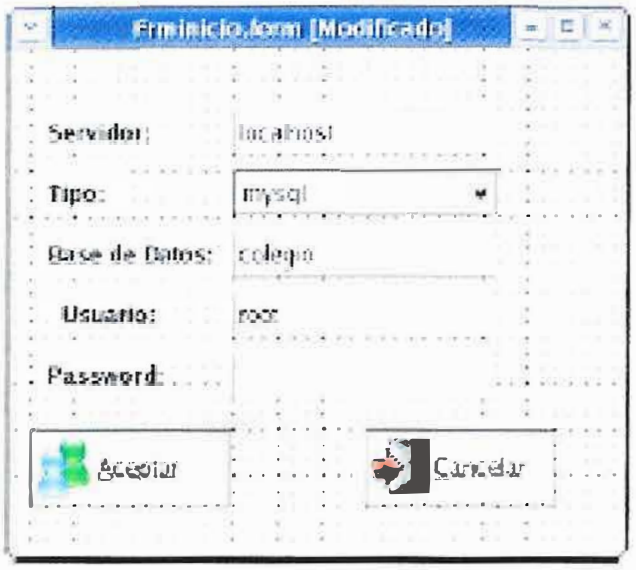
5. Parametrizar la aplicación: Si bien es cierto un programa de computadora nunca va estar terminado a un 100%, que pasaría si creamos el ejecutable y después cuando se quiera ejecutar en una maquina la base de datos no esté de manera local, sino en otro servidor? o ¿Qué pasaría si el nombre de la base de datos una vez creado el ejecutable cambie y/o la contraseña del usuario ? Para eso lo que debemos hacer para cuando esto ocurra es crear otro formulario que capture el nombre del servidor, nombre de la base de datos, nombre del usuario, y porque no el tipo de la base de datos que tal que mas

adelante queramos conectarla con podtgreSQL! Bien lo que debemos hacer es crear otro formulario el cual se llamará frminicio con los siguientes componentes:

Objeto Gambas	*Propiedad*	*Valor*
TxtBox	Name	txtservidor
	Text	localhost
ComboBox	Name	cmbdatos
	List	mysql, postgres
	Text	mysql

Objeto Gambas	*Propiedad*	*Valor*
TxtBox	Name	txtbdatos
	Text	colegio
TxtBox	Name	txtusuario
	Text	root
TxtBox	Name	txtpassword
	Password	True
Button	Name	btnaceptar
	Text	&Aceptar
Button	Name	btncancelar
	Text	&Cancelar

Este nuevo formulario se verá de esta manera:



5.1. Generar el evento click del botón aceptar: Al hacer doble clic sobre el botón btnaceptar del formulario frminicio en el modo diseño nos lleva al evento

.....

btnaceptar_Click () en el cual debemos agregar el siguiente código:

```
PUBLIC SUB btnaceptar_Click()  ' Validación de parámetros.
IF txtservidor.Text ="" THEN
message.Error ("Debe ingresar el servidor de conexión")
RETURN
END IF
IF txtbdatos.Tex t="" THEN
message.Error ("Debe ingresar el nombre de la base de datos")
RETURN
END IF
IF txtusuario.Text ="" THEN
message.Error ("Debe ingresar el nombre de usuario")
RETURN
```

```
END IF
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Mete los parámetros globales para a conexión con la BD                  '
Mglobal.db.Host = txtservidor.Text ' Nombre del Host                     '
Mglobal.db.Type = cmbdatos.Text ' Tipo de base de datos                  '
Mglobal.db.Name = txtbdatos.Text ' Nombre de la Base de Datos            '
Mglobal.db.Login = txtusuario.Text ' Nombre del usuario '
Mglobal.db.Password = txtpassword.Text ' Contraseña de acceso            '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
ME.Close ()
frmMain.Show ()
END
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Como se puede observar las primeras instrucciones es sólo para validar los parámetros obligatorio en el momento de iniciar la sesión como el servidor, tipo de base de datos, nombre de la base de datos y usuario, la contraseña podría ser vacía como lo estamos haciendo en nuestro caso, salvo que usted tenga configurado su servidor de base de datos con un nivel mayor de seguridad.

Posterior mente lo que se hace es en las variables globales declaradas en el modulo db se le asigna de manera dinámica cada vez que usted ingrese y ya dejarían de ser fijas como antes Una vez ingresado estos parámetros, se cierra el formulario actual y se abre el formulario frmMain.

Ahora acomoda esta parametrizado el inicio, se debe cambiar los valores en la función conexión () del modulo principal la cual debe quedar ahora sólo así:

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

```
PUBLIC FUNCTION conexión () AS Boolean
db.close ' Se Cierra la conexión con la base de datos
db.Open
RETURN TRUE
CATCH
Message. Error (ERROR. text)
RETURN FALSE
END
```

Solo se debe abrir la conexión porque todos los parámetros ya fueron inicializados en el formulario de inicio de sesión.

5.2. Configurar el formulario frminicio como Principal: Como este seráel formulario que se debe ejecutar, ya no el frmMain lo que se debe hacer es:

- ✓ Click derecho sobre el navegador de clases sobre el nombre del formulario

5.3. Evento Clic del botón Cancelar: Este botón lo único que debe hacer es salir por tanto este evento debe quedar así:

```
PUBLIC SUB btncancelar_Click ()
ME.Close ()
END
```

Al ejecutarlo nos debe cargar ahora el formulario así:


Inicio de Sesion
Servidor:
Tipo:
Base de Datos:
colegio
Usuario:
root
Password:
Aceptar
Cancelar

1.20 EJERCICIOS

Se desarrollaran los ejercicios los ejercicios necesarios para la práctica de estas herramientas, utilizando cada uno de los contenidos desarrollados en los capítulos anteriores.

1.20.1 EJEMPLO 1 HOLA MUNDO

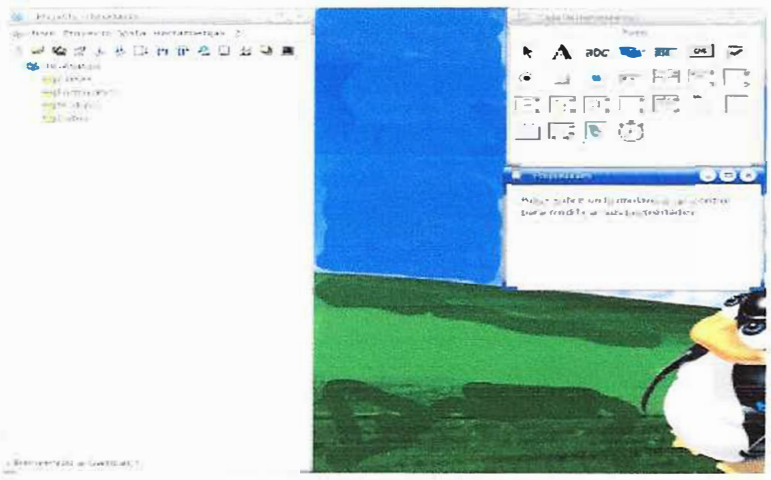
Vamos a seguir una serie de pasos, espera hacer una aplicación muy sencilla, esto nos servirá para familiarizarnos con Gambas, aunque aún desconozcamos muchas cosas.

1. Ejecuta Gambas, pulsa la opción "Nuevo Proyecto", aparecerá el asistente de creación de nuevos proyectos, lee la información, y pulsa "siguiente". Elige la opción "Crear un proyecto gráfico", y pulsa "siguiente". En nombre de proyecto pon "Holamundo", por ejemplo, y en el título, "Proyecto de pruebas".

2. Elige la carpeta a partir de la cual se creará la carpeta donde se almacenarán los archivos del proyecto. Si la creas, por ejemplo, en

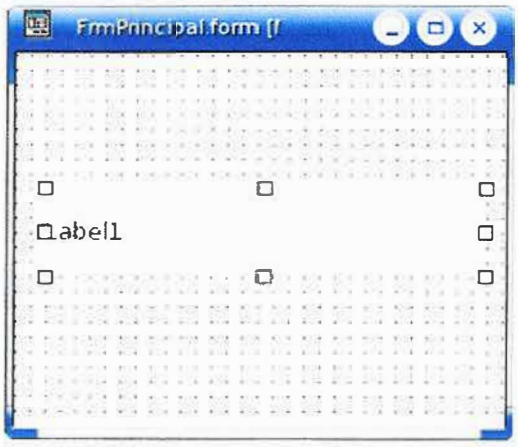
"/home/usuario", se creará una carpeta "/home/usuario/Holamundo". Pulsa "siguiente". Lee la información, y pulsa OK.

3. Aparecerá el entorno de desarrollo. Por ahora no hemos creado nada, verás que en el árbol del proyecto (que aparece con la cabeza de la mascota, y el nombre "HolaMundo"), hay una serie de secciones para depositar las diferentes partes del proyecto. De momento no te preocupes por todas las opciones.

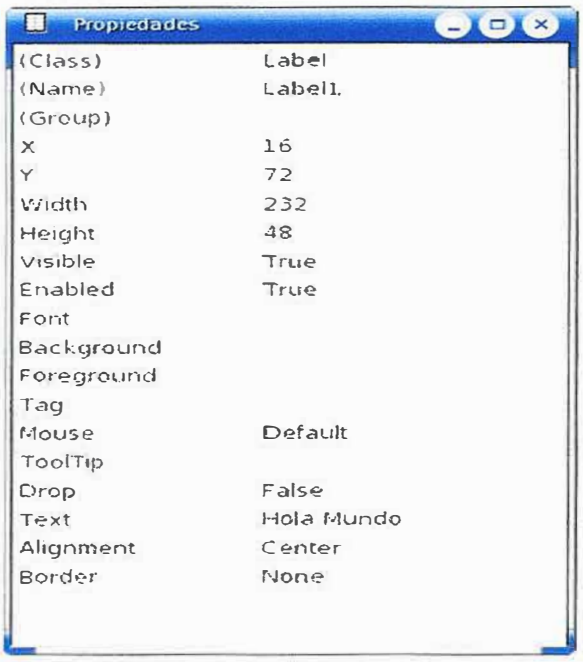


4. Pulsa "Formularios" con el botón izquierdo para seleccionar esta sección, y a continuación pulsa el botón derecho para que aparezca un menú flotante. Si eres zurdo y has configurado el ratón de ese modo, los botones serán los contrarios, derecho para seleccionar, e izquierdo para los menús.
5. En el menú aparece la opción "Nuevo". Sitúate sobre ella, y en el nuevo menú que surge, pulsa sobre la opción "formulario", con el botón izquierdo.

8. Ahora fíjate en la "Caja de herramientas". Aquí se muestran los controles que podemos utilizar para diseñar los elementos de nuestro formulario. De momento utilizaremos una etiqueta. Pulsa con el botón izquierdo en el icono que tiene una letra "A" grande y a continuación sitúate con el ratón en el formulario, y, mientras mantienes pulsado el botón izquierdo, mueve el ratón hacia la derecha y hacia abajo. Verás como la etiqueta se sitúa sobre el formulario y puedes variar su tamaño. Si sueltas el botón del ratón y a continuación pulsas sobre la etiqueta, verás que aparecen en los extremos unos cuadritos. Si te sitúas sobre ellos, mantienes el botón izquierdo pulsado, y mueves el ratón, puedes variar el tamaño de la etiqueta, y en general, de cualquier control que uses. Si te sitúas en el centro de la etiqueta, y mantienes pulsado el botón izquierdo mientras mueves el ratón, cambiarás la posición de la etiqueta en el formulario.



9. Vamos a dar el toque final, pulsa la etiqueta y mira la ventana de propiedades, aquí vemos las características de la etiqueta. Busca la propiedad llamada "Text", y escribe ahí "Hola Mundo". Para que quede más bonito, busca la propiedad "Align", y selecciona "Center".



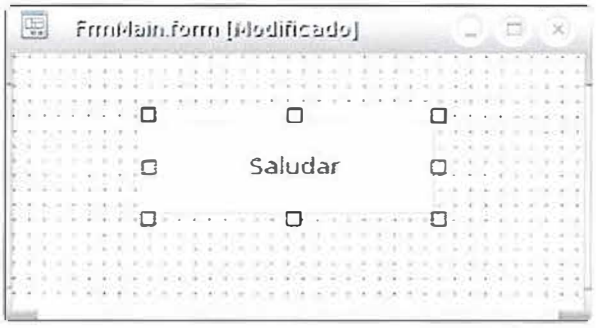
10. Y por último, la magia, ve al menú "Proyecto" de la ventana principal de Gambas, y pulsa en la opción "Ejecutar". ¡Ahí lo tienes! tu programa con una ventana diciendo "¡Hola Mundo!"

11. Ha sido fácil, ¿verdad? No ha sido necesario emplear ni una línea de código, sólo trabajar con el ratón un poco.

1.20.2 EJEMPLO 2 HOLA MUNDO 2

1. Vamos a añadir un botón. Mira ahora en la "Caja de herramientas". Hay un control que parece un botón con el texto "OK". De hecho, es un botón.

2. Sitúalo en el formulario del mismo modo que hiciste con la etiqueta del primer programa, de modo que quede situado a tu gusto. A continuación busca en la ventana de propiedades la propiedad "Text", y cambia su valor por "Saludar".



3. Ahora pulsa dos veces seguidas (o doble click, como le gusta decir a muchos) sobre el botón. Accederás a una ventana de código, en el que te indica esto:

```
4. ' Gambas class file
5.
6. PUBLIC SUB Button1_Click()
7.
8. END
```

9. Vamos a pararnos un momento a ver este código: observa la primera línea, verás que comienza con el símbolo " ' ". Todas las líneas que empiezan por este símbolo, son "comentarios", y significa simplemente que es una nota aclaratoria para el programador, que es ignorada totalmente por el compilador de Gambas. Si quieres, prueba a añadir más comentarios en el código:

```
10. ' Gambas class file
11. ' Este es un ejemplo de programa sencillo
12. ' para aprender algunos conceptos
13. ' básicos sobre Gambas
14.
15. PUBLIC SUB Button1_Click()
16.
17.
18.
19. END
```

20. Observemos las siguientes líneas: "PUBLIC SUB Button1_Click ()" y "END". Estas dos palabras claves definen el principio y final de una "Función". Una función no es más que un conjunto de órdenes que Gambas ejecuta cuando la función es llamada. En este caso, la función se ejecutará cuando se haga "Click" sobre el botón "Button1".

Ahora mismo no hay nada entre el principio y el final de la función, por tanto no pasará nada al pulsar el botón.

21. Diremos a Gambas que ha de mostrar un diálogo de mensaje diciendo "Hola Mundo" cuando se pulse el botón. Para eso, utilizaremos el objeto "Message". Ya hablaremos más adelante acerca de los objetos:

```
' Gambas class file
' Este es un ejemplo de programa sencillo
' para aprender algunos conceptos
' básicos sobre Gambas

PUBLIC SUB Button1_Click()

  Message.Info ("¡Hola Mundo!")

END
```

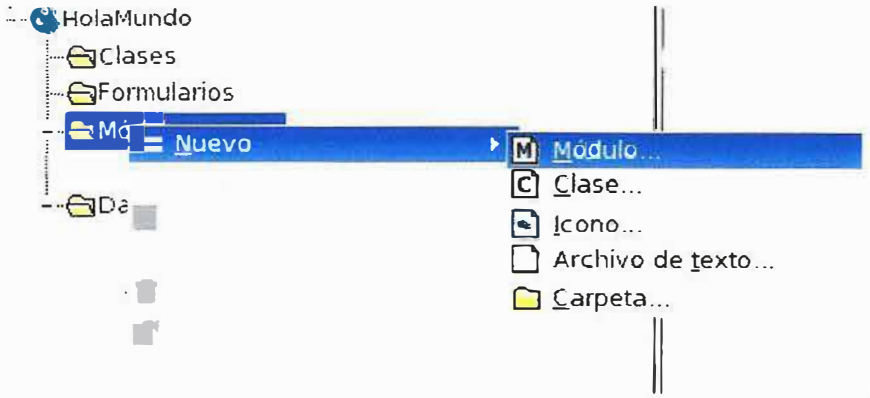
32. Después de escribir "Message." habrás observado que se ha desplegado un menú con varias palabras clave: "Info", "Warning",... Esto es lo que se llama "autocompletado de código", y es una utilidad para mostrarte las diferentes posibilidades que ofrece un objeto, en este caso "Message". Cuando termines este ejemplo, prueba las otras opciones, verás como varía el aspecto del mensaje. También habrás comprobado que tras escribir el paréntesis inicial, aparece una ayuda indicando la sintaxis de este método.

33. Ahora ya solo resta probarlo. Si has hecho todo bien, tras acudir al menú "Proyecto", y pulsar la opción "ejecutar", aparecerá el formulario con nuestro botón. Pulsa el botón, y surgirá el mensaje deseado.

1.20.3 EJEMPLO 3 HOLA MUNDO 3

Ahora vamos a trabajar sin utilizar el componente gráfico. Hay muchos programas que no necesitan de componente gráfico, por ejemplo, por que sólo realizan tareas no visibles para el usuario, o porque quieren consumir pocos recursos, o porque el equipo donde se ejecutará no va a disponer de sistema gráfico. Hay múltiples razones para utilizar sólo la consola, y los ejemplos indicados son sólo unas pocas.

1. Ejecuta Gambas, y selecciona nuevo proyecto. Al llegar hasta el punto donde se selecciona el tipo de proyecto, elige "Crear un proyecto de terminal"



3. Observa que en el código ha aparecido una función, automáticamente, como ocurrió en la segunda parte de este tutorial.

```
4. ' Gambas module file
5.
6. PUBLIC SUB Main()
7.
8. END
```

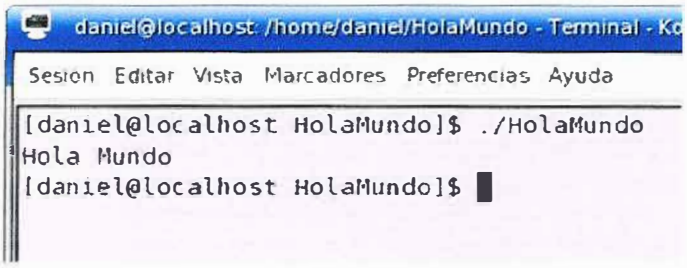
9. Todos los programas cuyo inicio no se encuentra en un formulario, deben tener una función, llamada forzosamente "Main", que será la que se ejecute nada más comenzar el programa. Ahora mismo no hay nada, y si ejecutáramos el programa, nada pasaría. De hecho, este es el programa más simple que puede escribirse con Gambas. Añadiremos ahora el código para mostrar nuestro mensaje. Ya no utilizaremos la clase "Message", que forma parte del componente gráfico de Gambas, si no una función llamada "PRINT" que permite "imprimir" mensajes en la consola:

```
10. ' Gambas module file
11.
12. PUBLIC SUB Main()
13.
```

```
14. PRINT "Hola Mundo"
15. END
```

16. Para ver mejor el resultado del programa, vamos a aprender a compilar un programa. Acude al menú "Proyecto" y pulsa la opción "Crear Ejecutable". Si no te aparece ningún mensaje de error, ya tienes el ejecutable listo para utilizar. ¿Dónde está, por cierto? Pues está en la carpeta del proyecto, así si el proyecto se encuentra, por ejemplo en /home/usuario/HolaMundo, el ejecutable estará dentro de esta carpeta, y con el nombre "HolaMundo".

17. Abre una ventana de terminal, acude a la carpeta, y ejecuta el programa para ver el resultado.



Es el momento de decir también que hay dos tipos de lenguaje de programación, los compilados y los interpretados. Los primeros, al compilar, crean verdaderos binarios, código entendible directamente por el microprocesador del equipo. Los segundos, o bien no disponen de compilador (como la shell bash), o bien el compilador genera un "código intermedio", como es el caso de Java o Gambas. Dicho código intermedio, no es entendible directamente por el microprocesador, si no que un programa llamado intérprete (gbx, en el caso de Gambas), hace de intermediario entre la ejecución real del programa y el código intermedio. La razón de utilizar ese código intermedio, es que es más rápido que el código "humano" que nosotros hemos escrito, aunque más lento que el código binario.

Este planteamiento tiene la ventaja de que el código generado puede ser ejecutado en cualquier máquina para la que exista un intérprete del lenguaje, independientemente de la marca del procesador o el sistema operativo (compila una vez y úsalo en todos sitios, es el lema). Como inconveniente, el rendimiento es menor (no se puede utilizar para aplicaciones que exijan gran rendimiento del equipo), y es necesario tener instalado el intérprete en la máquina.

1.20.4 EJEMPLO 4 OPERACIONES BASICAS

El siguiente ejercicio realiza las operaciones básicas agregando los valores en los TextBox.Text y chequeando los radio Option.


```
PUBLIC SUB btnaceptar_Click ()
  Dim A As Integer
  Dim B As Integer
  Dim Suma As Integer
  Dim Resta As Integer
  Dim Mult As Integer
  Dim Div As Integer
  Dim Result As Integer

  A = Val(TextBox1.Text)
```

```
B = Val(TexTBox2.Text)
If Option1.Value = True Then
  Result = A + B
End If
If Option2.Value = True Then
  Result = A - B
End If
If Option3.Value = True Then
  Result = A * B
End If
If Option4.Value = True Then
  Result = A / B
End If
TextBox3.Text = Result

END

PUBLIC SUB btncancelar_Click ()
ME. Close ()
END
```

EVALUACION

1. ¿Cuales son los pasos para crear un proyecto en Gambas?

2. ¿Cuáles son los pasos para crear un formulario en un proyecto nuevo?

3. ¿Cómo habilitar la caja de propiedades de un control, existen dos formas de hacerlo cuales son?

4. Escriba un formulario que cree un área de texto y proporcione botones para ofrecer las operaciones cortar, copiar y guardar.

ECLIPSE

CAPITULO 2

2. INTRODUCCIÓN A LA PROGRAMACION A ECLIPSE

Este capitulo contiene los conceptos iniciales de la herramienta Eclipse, el cual permitirá al estudiante conocer y entender las ventajas, componentes y la plataforma de ejecución de esta herramienta, principalmente le dará a conocer el entorno grafico de programación con la finalidad de facilitar el manejo de la herramienta Eclipse.

2.1 INTRODUCCIÓN A ECLIPSE

Eclipse es una poderosa herramienta que permite integrar diferentes aplicaciones para construir un entorno integrado de desarrollo (Figura 1).
Es un proyecto de desarrollo de software opensource, que está dividido en tres partes[2]: el proyecto Eclipse Proyect, Eclipse Tools, y Eclipse Technology Proyect.

1. The Eclipse Project es responsable de desarrollar el workbench (banco de trabajo) del IDE Eclipse (la "plataforma" que contiene las herramientas Eclipse), el JDT (Java Development Tools), y el PDE (Plug-In Development Environment) utilizado para ampliar la plataforma.
2. The Eclipse Tools Project se enfoca en la creación de herramientas para la plataforma Eclipse. Entre los subproyectos actuales se incluyen un IDE Cobol, un IDE C/C++, y una herramienta de modelado EMF.
3. The Eclipse Technology Project se enfoca en investigaciones tecnológicas, incubación y educación utilizando la plataforma Eclipse.

Mediante Eclipse se puede crear diversas aplicaciones como ser sitios Web, programas Java, C++ y Enterprise Java Beans. Su principal aplicación es JDT,

[2] Proyecto Pasantía LEAD. Plataforma Eclipse, Desarrollo de Plug-ins. Archivo, PDF. Julio 2004, Pág. 1.

herramienta para crear aplicaciones en Java. Otras aplicaciones pueden ser integradas a Eclipse en forma de plug-ins, que son reconocidos automáticamente al iniciar el mismo. Como Eclipse está escrito en Java, para su funcionamiento se debe tener instalado el JRE (Java Runtime Enviroment). Eclipse detecta automáticamente la ubicación de la JRE instalada. Las funcionalidades que otorga Eclipse se localizan de dos formas diferentes: en un pequeño núcleo conocido como el Plataforma Runtime o en forma de plug-ins. Existe un conjunto de plug-ins que ya vienen con la plataforma. Entre los plug-ins que vienen con la plataforma encontramos: Ant, Compare, Core, CVS, Debug, Help, JDT, JFace, Releng, Scripting, Search, SWT, Text, UI, Update, Team y WebDAV.

La plataforma Eclipse esta construida en base a plug-ins.

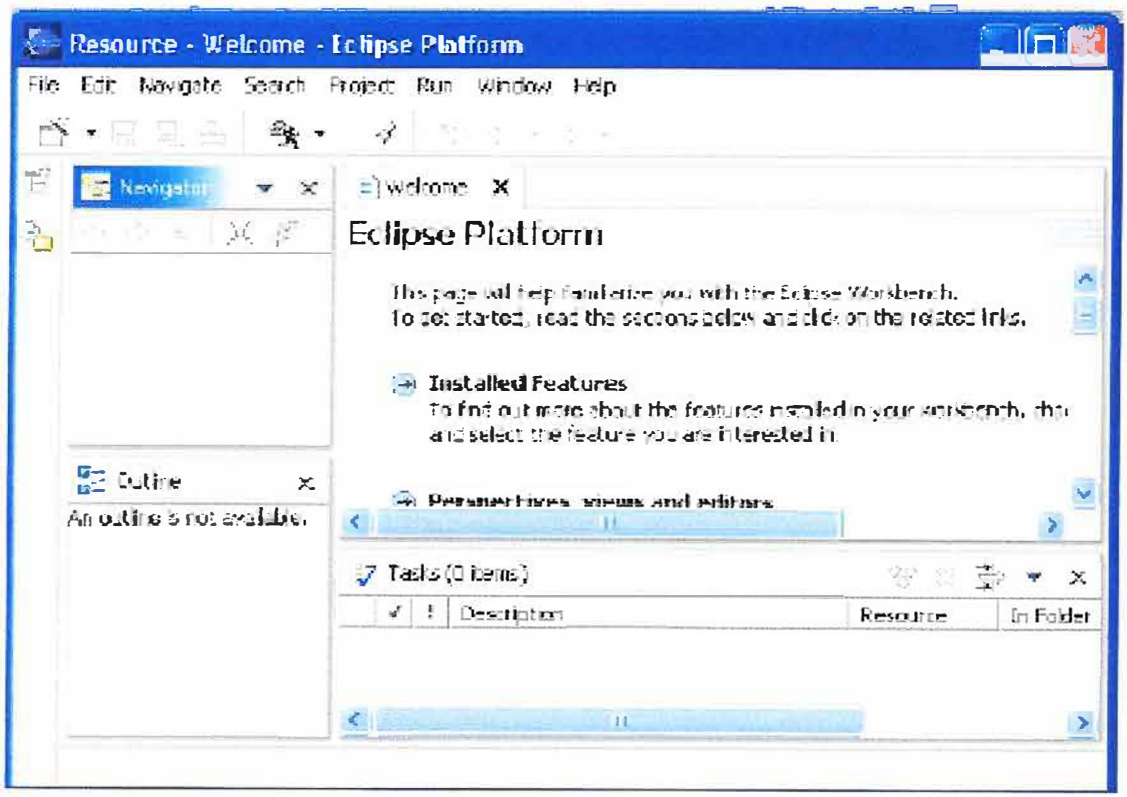


Figura 1 – Plataforma Eclipse

2.2 ¿QUÉ ES ECLIPSE?

Eclipse es una excelente herramienta de programación que permite crear aplicaciones tanto para Windows como para Linux y es GNU, es una herramienta que presenta una plataforma universal para la creación de ventanas, botones, menús y cualquier otro elemento de una forma fácil e intuitiva.

El objetivo primordial de Eclipse[3] es proveer una plataforma libre para alojar de forma integrada herramientas para el desarrollo de aplicaciones, brindar soporte para cualquier lenguaje de programación, atraer a la comunidad de desarrolladores de herramientas a popularizar Java como lenguaje para desarrollar herramientas.

2.3 VENTAJAS DE ECLIPSE

La característica clave de Eclipse es la extensibilidad. Eclipse es una gran estructura formada por un núcleo y muchos plug-ins que van conformando la funcionalidad final. La forma en que los plug-ins interactúan es mediante interfaces o puntos de extensión; así, las nuevas aportaciones se integran sin dificultad ni conflictos.

Ventajas

- Es una herramienta open-source.
- Soporta la construcción de una variedad de herramientas para el desarrollo de aplicaciones.
- Soporta el desarrollo de aplicaciones basadas en GUI y non-GUI.
- Soporta herramientas que manipulan diferentes tipos de archivos como por ejemplo Java, C, C++, EJB, HTML, GIF, etc.
- Corre en una gran cantidad de sistemas operativos incluyendo Windows y Linux.

[3] Ing. CALEGARI GARCÍA, Daniel, Grupo COAL. Eclipse aplicado al Desarrollo Basado en Componentes. Facultad Ingeniería Universidad de la República. 2004. Pág. 7.

- Mediante JDT facilita la creación de aplicaciones programadas en Java.
- Provee a los desarrolladores, herramientas (ej.- PDE) que facilitan la creación de plugins así como un API muy completo que encapsula toda la plataforma haciendo hincapié en constructores de alto nivel pero de alta flexibilidad como Diálogos, Wizards, Viewers, Actions, etc.
- Esto último se logra mediante el uso de tecnologías como JFace y SWT.
- El uso de XML para la declaración de extensiones facilita considerablemente el desarrollo de herramientas evitando incluso, en ciertos casos, la necesidad de generar código Java para proveer cierta funcionalidad.

Desventajas

- Si bien Eclipse es multiplataforma, los plug-ins no tienen por qué serlo.
- Existen Plugins que sólo corren en una plataforma, o que aún no han sido desarrollados para otras.
- Al ser una herramienta open-source, se desarrollan plug-ins que no tienen todas las funcionalidades que tienen en otras herramientas comerciales, como ser IBM Websphere.
- Si bien la documentación del API de la plataforma cubre muy bien los temas básicos se hallan problemas a la hora de realizar cosas de carácter muy específico.
- En algunos casos la información no está, es pobre, o se encuentra distribuía lo que dificulta el entendimiento. Particularmente surgieron problemas a la hora de obtener información acerca de la estructura del Workbench en términos de componentes SWT y JFace, información que fue necesaria para la implementación del Plug-in Listener.

2.4 ARQUITECTURA DE LA PLATAFORMA ECLIPSE

Eclipse es una plataforma[4] que ha sido diseñada para facilitar el desarrollo, en término de diseño. La plataforma no ofrece gran funcionalidad por si sola sino que su valor real yace en el modelo de plug-ins que se pone a disposición del usuario. Ha sido diseñada para se ejecutada bajo múltiples sistemas operativos aun brindando una robusta integración. Eclipse hace uso de un **workbench** común sobre el cual las herramientas pueden ser integradas utilizando los denominados "puntos de extensión".

La plataforma Eclipse está estructurada como un conjunto de subsistemas los cuales son implementados en uno o más plug-ins que corren sobre una pequeña runtime engine. Dichos subsistemas definen puntos de extensión para permitir agregar funcionalidad a la plataforma. A continuación se describen los principales componentes de la plataforma.

Presenta una arquitectura extensible basada en Plug-ins:

PDE	**Plug-in Development Environment**
JDT	**Java Development Tools**
PLATAFORM	**Plataforma Eclipse**
JAVA VM	**Máquina Virtual Java**

2.4.1 PLUGINS

Un plug-in es la mínima unidad de la plataforma que puede ser desarrollado por separado. Se pueden encontrar herramientas pequeñas desarrolladas en un sólo plug-in o herramientas mucho más complejas que se componen de un conjunto de plug-ins que se comunican entre sí. Existen por lo general, dos formas de instalar los nuevos plug-ins en Eclipse. En la mayoría de los casos

[4] -Ing. CALEGARI GARCIA, Daniel, Grupo COAL. Eclipse aplicado al Desarrollo Basado en Componentes. Facultad Ingeniería Universidad de la República. 2004. Pág. 6.
-Proyecto Pasantía LEAD. Plataforma Eclipse, Desarrollo de Plug-ins. Archivo, PDF. Julio 2004, Pág. 1.

sólo hay que descompactar el zip del plug-in en el directorio en el que se encuentra instalado Eclipse.

Existen casos donde la herramienta provee algún programa de instalación, el cuál integra la nueva herramienta. Eclipse provee la utilidad de comenzar el programa con los plug-ins especificados, permitiendo acceder a distintas aplicaciones sin necesidad de levantar todas a la vez al momento de ejecutarlo. Las herramientas integradas a Eclipse operan en archivos del workspace del usuario. El workbench, es la interfaz de usuario de la plataforma. El mismo esta compuesto de un conjunto de vistas, editores y perspectivas. Los editores permiten crear, modificar y salvar objetos, las vistas proveen información acerca de los objetos con los que se están trabajando en el workbench y las perspectivas proveen distintas formas de organización del proyecto.

2.4.2 PLATAFORMA RUNTIME

Implementar la "runtime engine" que se encarga de iniciar la base de la plataforma y de dinámicamente descubrir plug-ins. Un plug-in es un componente estructurado que se describe a sí mismo frente al sistema utilizado un archivo llamado plugin.xml. La plataforma mantiene un registro de aquellos plug-ins instalados así como de las funcionalidades que proveen. Nuevas funciones son agregadas al sistema usando un modelo de extensión común. Los "puntos de extensión" son lugares bien definidos dentro del sistema que permiten ser extendidos por plug-ins. Cuando una herramienta contribuye con una implementación para determinado punto de extensión se dice que agrega una "extensión" a la plataforma. A su vez, cada plug-in puede definir sus propios puntos de extensión de forma tal que puedan ser extendidos por otros. Nótese que este mecanismo de extensión es la única manera de agregar funcionalidad a la plataforma. De esta forma todas las herramientas provistas con el Eclipse SDK no utilizan ningún mecanismo privado para su implementación.

Las extensiones son típicamente escritas en Java utilizando el API de la plataforma. Sin embargo, algunos puntos de extensión tienen asociadas extensiones provistas de forma de ejecutables, componentes ActiveX, o incluso extensiones que han sido desarrolladas mediante lenguajes script. Téngase presente que de forma general, solo un subconjunto del total de la funcionalidad de la plataforma se encuentra disponible para extensiones no realizadas con Java.

Un objetivo muy importante del runtime es que usuarios finales deberían no tener que sufrir desventajas de desempeño o de uso de memoria por aquellos plug-ins que si bien están instalados no están siendo usados. De esta manera un plug-in puede ser instalado y agregado al registro pero el mismo no será activado a menos que una función que provea sea requerida según la actividad del usuario.

2.4.3 SWT

El Standard Widget Toolkit (SWT) de la plataforma Eclipse ha sido el aspecto de mayores rechazos, adhesiones y controversias en la Comunidad de Java. SWT constituye la respuesta de IBM, para el desarrollo de interfaces de usuario, a AWT y a Swing De acuerdo con la documentación oficial de Eclipse, he aquí la descripción del componente Standard Widget Toolkit: "El componente SWT está diseñado para proporcionar acceso eficaz y transportable a los servicios de interfaz de usuario del sistema operativo sobre el cual se implementa". El SWT es un conjunto de componentes gráficos y una biblioteca gráfica integrada con el sistema de ventanas específico del sistema operativo, pero con una API independiente del SO". Tanto SWT como JFace son bibliotecas estructuradas en paquetes con clases e interfaces escritas en Java.

2.4.4 WORKSPACE

Define el API para la creación y gestión de recursos, como lo son proyectos, archivos y carpetas, que son producidos por herramientas y que son almacenados en el sistema de archivos

2.4.5 WORKBENCH

Implementa el aspecto visual que permite al usuario navegar por los recursos y utilizar las herramientas integradas. Particularmente define puntos de extensión para el agregado de componentes como lo son las Vistas, Editores, Menús, Diálogos y Wizards. Además incluye frameworks que son de utilidad para el desarrollo de interfases de usuario. El uso de los anteriores no solo facilita el desarrollo del interfaz de un plug-in sino que asegura que las herramientas provean una apariencia común. El "Standard Widget Toolkit" (SWT) es una biblioteca de bajo nivel independiente del sistema operativo que soporta integración con la plataforma y brinda al desarrollador un API portable. Por otro lado, el framework "JFace UI" ofrece constructores de alto nivel para lo relativo a diálogos, wizards, acciones, preferencias y gestión de controles.

2.4.6 HELP SYSTEM

Implementa un servidor de ayuda optimizado y la facilidad de integración de documentos. Define particularmente puntos de extensión que los plug-ins pueden utilizar para contribuir con ayudas u otro tipo de documentación de forma de libros explorables.

2.4.7 TEAM SUPPORT

Permite a otros plug-ins definir y registrar implementaciones para la programación orientada a un equipo de desarrollo, acceso de repositorios y versionado de artefactos.

2.5 INSTALAR ECLIPSE

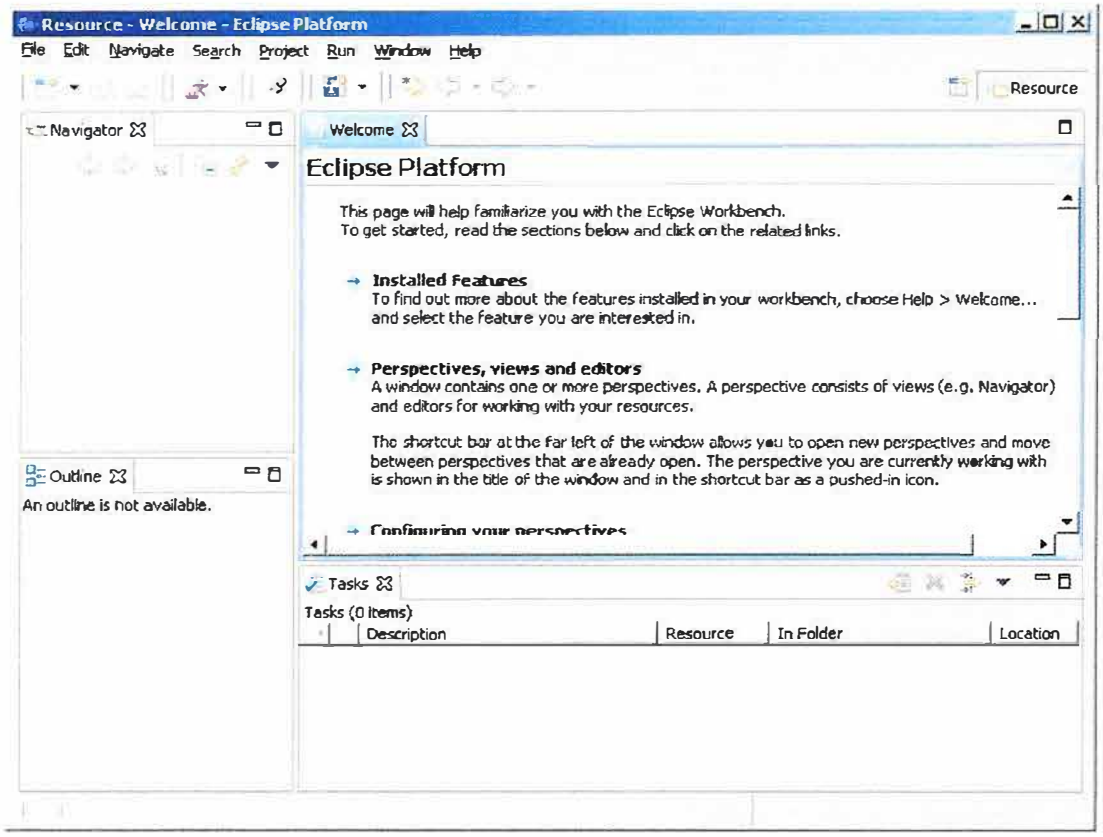
Se estará preguntando donde obtener una copia de Eclipse para su plataforma y qué tiene que hacer para instalarla.
Lo primero que necesita hacer es asegurarse de que tiene disponible un *Java* runtime (JRE) apropiado.
Aunque Eclipse puede compilar código para las versiones 1.3 y 1.4 del JDK, las versiones actuales están diseñadas para una versión 1.3. Si no está seguro de donde encontrar la JVM apropiada para su plataforma, puede encontrar más información en: www.eclipse.org.

Una vez que tenga la JVM, está listo para obtener Eclipse.Visite la página de descargas de Eclipse y coja la última versión liberada para su plataforma. Todas las versiones se entregan como ficheros .zip. Descomprima el archivo en un directorio apropiado y luego lea todos los ficheros que estén presentes en el directorio readme.

Lo crea o no, ya se ha terminado la instalación. Si ha instalado apropiadamente la JVM y ha desempaquetado correctamente el archivo de Eclipse, debería poder ejecutarlo ahora mismo.

Todas las distribuciones binarias proporcionan una aplicación lanzadora en el directorio principal de Eclipse. El nombre de esa aplicación varía entre plataformas: eclipse.exe en Windows, eclipse en Solaris, etc. La primera vez que se ejecute Eclipse, completará algunas pocas tareas que quedan de la instalación (es decir, la creación del directorio workspace para almacenar los ficheros de proyecto) antes de que aparezca el entorno.

2.6 ENTORNO GENERAL DE ECLIPSE



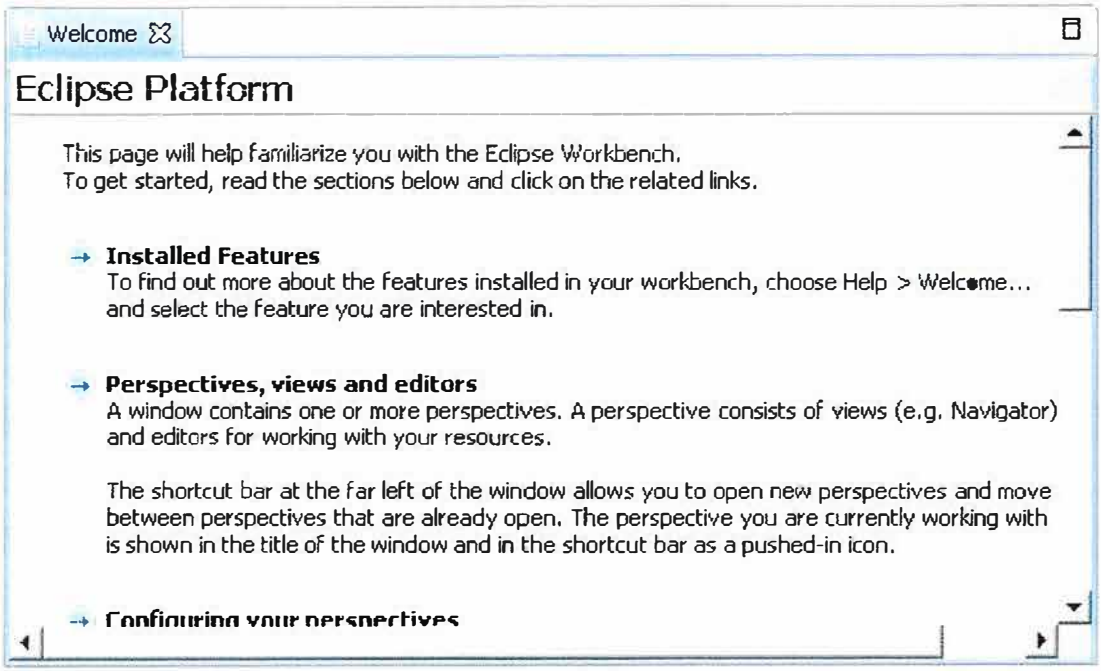
Después de lanzar Eclipse por primera vez, debería ver una ventana como ésta:

2.6.1 BARRA DE HERRAMIENTAS

En ellas se pueden distinguir los siguientes elementos:

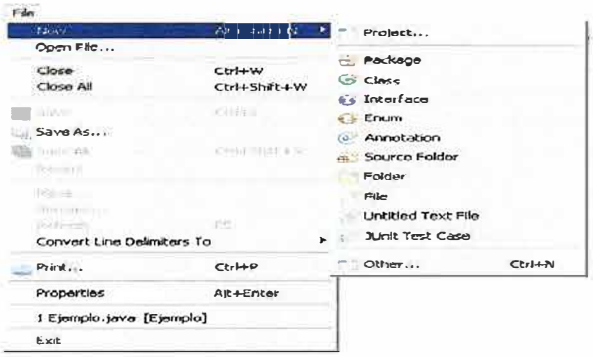
1- La barra de menús
2- La barra de herramientas estándar
3- Ventana de proyecto
4- Salida de compilación

También hay una ventana del editor con pestañas, que inicialmente muestra la página de **Bienvenida**:



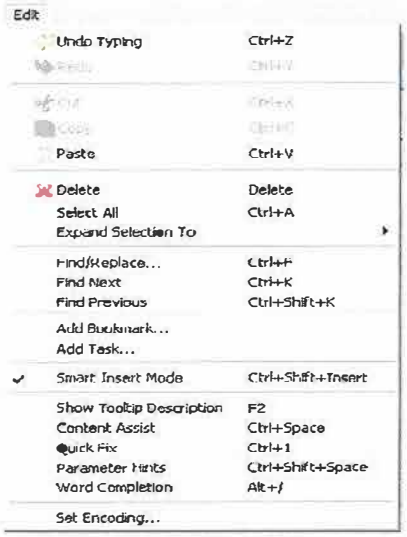
2.6.2 BARRA DE MENÚS

Como puede ver, el IDE Eclipse tiene una barra de menú bastante estándar: Contiene distintas casillas como son File, Edit, Navégate, Search, Project, Run, Windows, Help, cada una de estos posee diversas opciones y comandos cada uno con sus diversas funcionalidades, dependiendo el menú seleccionado.

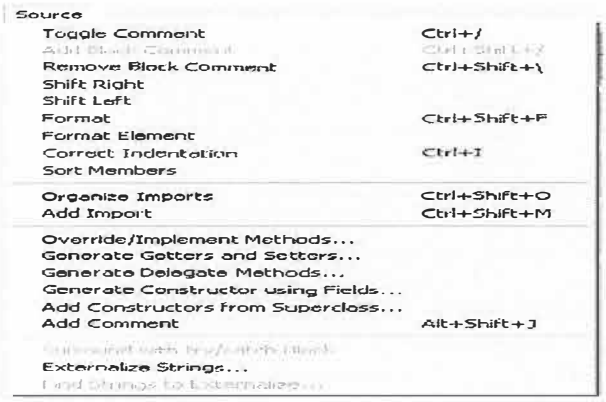


El menú File tiene pocas novedades. Lo más importante es la distinción entre proyectos. Un proyecto reúne y organiza todos los ficheros que componen el

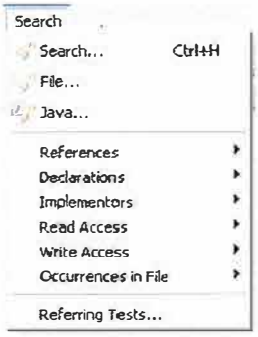
programa o aplicación. Eclipse permite tener más de un proyecto abierto simultáneamente, lo cual puede ser útil en ocasiones. Con el comando New... se crea un nuevo proyecto en la ventana New Project. Con los comandos Open File... se abre un proyecto ya existente.



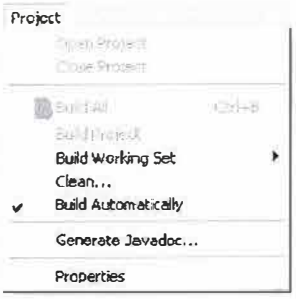
Menú Edit aporta funciones habituales como copiar, cortar, pegar, deshacer, buscar, y eliminar, todas estas funciones se pueden aplicar sobre los objetos o sobre el código.



Menú Source tiene varias funciones que permite hacer, como comentario importar paquetes y entre otros del entorno de desarrollo.



Menú Search tiene los comandos para buscar.



Menú Project se encuentran los comandos para compilaciones del programa.

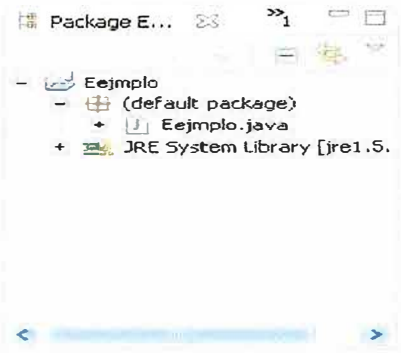
Menú Run aquí están los comandos que permiten ejecutar o correr el proyecto completo o solo una parte de el accionando el comando **Run As,** despliega una lista con todos los formularios del proyecto y se escoge cual se desea ejecutar.

2.6.3 BARRA DE HERRAMIENTAS ESTÁNDAR

La barra de herramientas estándar aparece debajo de la barra de menús, que permite acceder a las opciones más importantes de los menús.

2.6.4 VENTANA DE PROYECTO

Permite acceder a los distintos formularios y clases que componen el proyecto. Desde ella se puede ver el diseño gráfico de dichos formularios, y también permite editar el código que contienen.



2.6.5 SALIDA DE COMPILACIÓN

Muestra el resultado de la labor de compilación, indicando los errores encontrados o indicando que toda esta bien, y a demás arroja el tiempo de compilación.

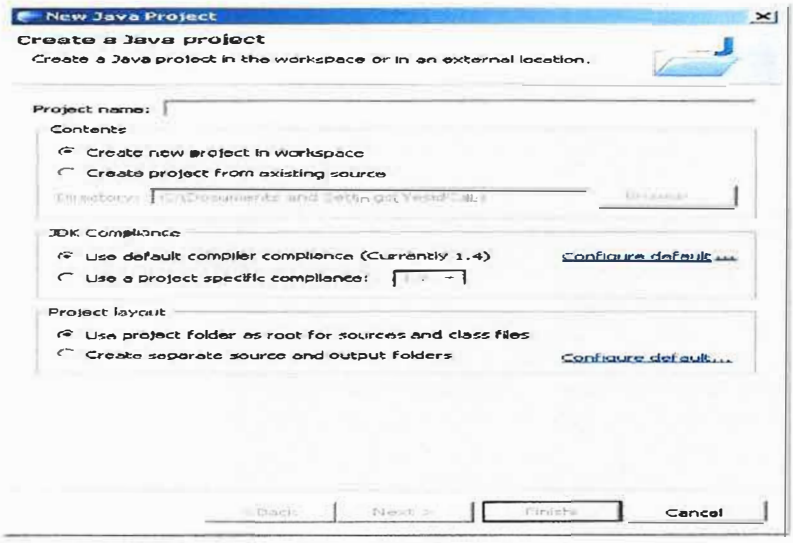
2.7 INTRODUCCIÓN A LA PROGRAMACIÓN

Después de conocer los entornos gráficos de Eclipse el capitulo 1, en este capitulo manejaremos paso a paso la forma de crear un nuevo proyecto en Eclipse, así mismo creando sus clases y los archivos que genera para su ejecución.

2.7.1 NUEVO PROYECTO EN ECLIPSE

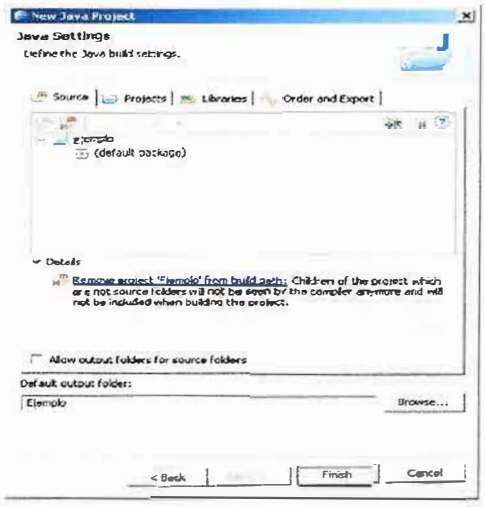
Pasos para crear un proyecto

Paso 1: En el menú archivo se da nuevo proyecto y aparece esta ventana, en donde se especifica el tipo de aplicación.



Ventana nuevo proyecto

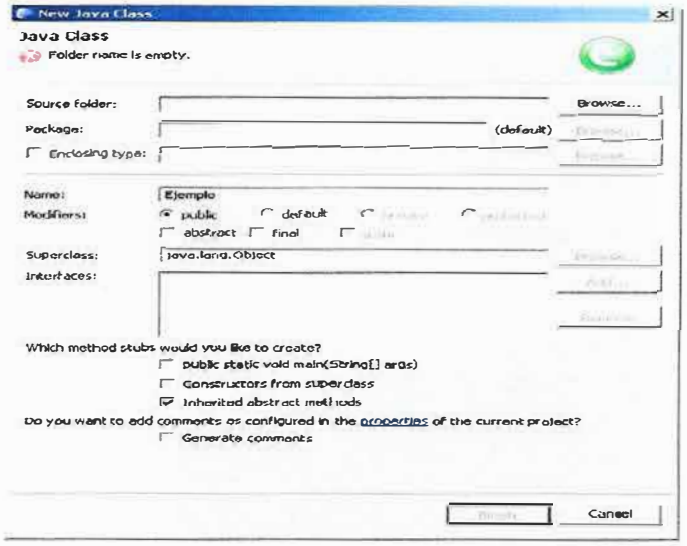
Paso 2: Aparece una nueva ventana en donde se da el nombre del proyecto y la ruta en donde esta guardado.



Ventana nuevo proyecto

2.7.2 NUEVA CLASE

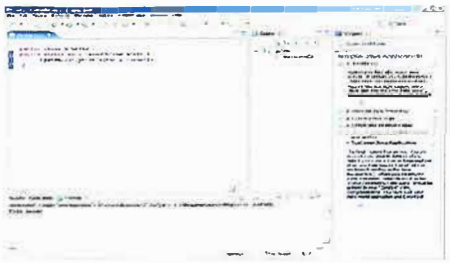
Paso 3: Aparece otra ventana en donde se va a crear la nueva clase y en que ruta se va a guardar.



Ventana nueva clase

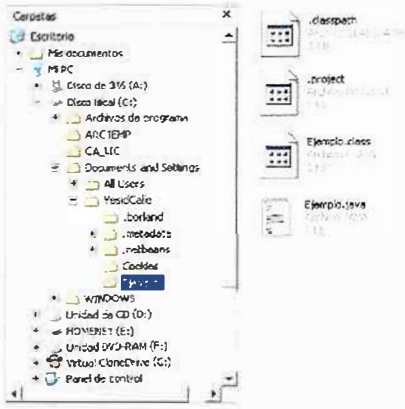
2.8 COMPILACIÓN Y EJECUCIÓN

Paso 4: Por ultimo nuestra aplicación compilada y ejecutada se vera así.



Archivos que genera

Este es el directorio de archivos que genera.



Directorio de Archivos

- ✓ Primero crea una carpeta con el nombre del proyecto, en este caso Ejemplo.
- ✓ Dentro de esta crea cuatro subdirectorios, llamados build, nbproject, src, test y un archivo manifiest.mf.
- ✓ Dentro de la carpeta build crea otro subdirectorios, llamado clases y dentro de este otro subdirectorio llamado javaapplication1, dentro de este último se guardan los archivos punto class del proyecto.

- ✓ En la carpeta nbproject crea uno archivos punto XML y punto properties, y crea un subdirectorio llamado private en donde almacena un archivo llamado private.xml.
- ✓ En la carpeta src crea un subdirectorio llamado javaapplication1 y dentro de este guarda los archivos de texto con extensión punto java.

TIEMPOS DE COMPILACIÓN Y EJECUCIÓN.

Los tiempos de respuesta del **Eclipse 3.1** están tabulados en la siguiente tabla.

Proceso	Tiempo (real)
Abrir y cargar	45 seg.
Nuevo Proyecto	14 seg.
Formulario	3 seg.
Compilación	5 seg.
Debug	5 seg.
Run	2 seg.
Run – Sigle	1 seg.

Para que un programa con SWT pueda compilarse y ejecutarse en el entorno de Eclipse, se necesitan dos pasos previos:

- El primero consiste en indicar al compilador dónde está el código Java de SWT; es decir, dónde se encuentran las clases SWT. Éstas se almacenan en el archivo swt.jar. Para indicar a Eclipse dónde debe buscar el archivo swt.jar hay que seguir varios pasos.
  En primer lugar, sobre el nombre del proyecto de Java, el botón derecho del mouse. A continuación, hay que seleccionar la opción Properties.
  En la ventana que aparece debe seleccionarse Java Build Path (panel de la derecha) y la pestaña Libraries. Seguidamente, debe incluirse swt.jar con el botón Add External JARs. Según el FAQ más actualizado. de Eclipse www.eclipse.org/eclipse/faq/eclipse.

El swt.jar se encuentra – dependiendo de la plataforma– en donde se ejecute:

• win32: INSTALLDIR\eclipse\plugins\org.eclipse.swt.win32_3.0.1\ws\win32\

• Linux GTK: INSTALLDIR/eclipse/plugins/org.eclipse.swt.gtk_3.0.1/ws/gtk/

• Linux Motif: INSTALLDIR/eclipse/plugins/org.eclipse.swt.motif_3.0.1/ws/motif/

•PhotonQNX:INSTALLDIR/eclipse/plugins/org.eclipse.swt.photon_3.0.1/ws/photon/

•MacOSX: NSTALLDIR/eclipse/plugins/org.eclipse.swt.carbon_3.0.1/ws/carbon/

- El segundo paso que se requiere para ejecutar el programa consiste en indicar a Eclipse dónde está la biblioteca compartida que requiere swt.jar. Una manera consiste en introducir, cuando se arranca la máquina virtual de Java, la ubicación de la biblioteca compartida como argumento de ejecución. Para ello, basta elegir el menú Run, escoger Run. Elegir la pestaña Arguments en la parte derecha de la pantalla e introducir en el área de texto VM arguments lo siguiente Djava.library.path="directorio que contiene la biblioteca compartida para SWT". Tal y como sucedía con swt.jar, la biblioteca se encontrará en una ubicación dependiente de la plataforma. Según el FAQ de Eclipse, las ubicaciones serán del estilo.
- Windows:INSTALLDIR\eclipse\plugins\org.eclipse.swt.win32_3.0.1\os\win32\x86.
- LinuxGTK:INSTALLDIR/eclipse/plugins/org.eclipse.swt.gtk_3.0.1/os/linux/x86.
- LinuxMotif:INSTALLDIR/eclipse/plugins/org.eclipse.swt.motif_3.0.1/os/linux/x86.
- PhotonQNX:INSTALLDIR/eclipse/plugins/org.eclipse.swt.photon_3.0.1/os/qnx/x86.

2.9 EJERCICIOS

2.9.1 EJEMPLO 1: SALUDOS.JAVA

```
import org.eclipse.swt.*;
import org.eclipse.swt.widgets.*;

 public class Saludos
{
   public static void main(String args[]){
      // Se crea un display que servirá como contenedor para el
shell.
      Display display = new Display();

      // Se coloca el display dentro del shell.
      Shell shell = new Shell(display);

      // Se establece el tamaño y el texto del shell.
      shell.setSize(250, 120);
      shell.setText("Mi primera aplicación con SWT");
      // Se crea una etiqueta centrada en la ventana y se le da
un texto.
      Label etiqueta= new Label(shell, SWT.CENTER);
      etiqueta.setText("Saludos a los lectores de javaHispano");
      etiqueta.setBounds(shell.getClientArea());

      // Se muestra el shell; un shell es invisible salvo que se
llame a open().
      shell.open ();
```

```
        // Se entra en un bucle para leer y enviar los sucesos del
    usuario.
        // Si no hay ningún suceso que tratar, el método sleep()
    hace que el
        // programa espere el suceso siguiente sin consumir ciclos
    de CPU.
        // Se sale del bucle cuando el usuario cierra el shell.
         while (!shell.isDisposed()) {
           if (!display.readAndDispatch()) {
              display.sleep();
           }
         }
// Se cierra el display y se liberan los recursos que empleaba del sistema
operativo
// Siempre es conveniente liberar explícitamente los recursos
// que no se van a necesitar. En este caso no es necesario llamar a dispose(),
// Pues si se llega a aquí es porque el programa va a terminar, lo que implica
que
// Se liberarán los recursos asociados; pero es una buena práctica
acostumbrarse
// a hacerlo.
         display.dispose ();
       }

    }
```

2.9.2 EJEMPLO 2: BOTON

```
import org.eclipse.swt.*;
import org.eclipse.swt.widgets.*;
import org.eclipse.swt.events.*;
```

```
/**
* Esta clase genera un botón que inicialmente permanece apretado.
* La clase Button admite los siguientes estilos: BORDER, CHECK, PUSH,
* RADIO, TOGGLE, FLAT, ARROW (con UP, DOWN), LEFT, RIGHT y
 CENTER.
   *  BORDER crea un botón con borde.
   *  CHECK crea un botón de casilla de verificación.
   *  PUSH crea un botón estándar (valor por omisión).
   *  RADIO crea un botón de opción.
   *  TOGGLE crea un botón que mantiene su estado pulsado o no
 pulsado.
   *  FLAT crea un botón sin efectos de relieve 3D (plano).
   *  ARROW crea un botón en forma de flecha (UP y DOWN marcan el
 sentido de ésta).
   *  LEFT, RIGHT, CENTER alinean el texto asociado al botón.
   */
      Public class EjemploBoton {

            public static void main(String args[]) {
            // Se crea un display que servirá como contenedor para el shell.
            Display display = new Display();

            // Se coloca el display dentro del shell.
            Shell shell = new Shell(display);

            // Se establece el tamaño y el texto del shell.
            shell.setSize(250, 120);
            shell.setText("Ventana hecha con el SWT");

   // Se establece el tipo, el tamaño y el texto del botón.
```

```
// Un botón TOGGLE permanece presionado tras pulsarlo, hasta que se vuelve a pulsar.
        Button boton = new Button(shell, SWT.TOGGLE);
        boton.setSize(150, 50);
        boton.setText("Soy un botón de tipo TOGGLE");

        // Tratamiento de sucesos: muy similar a como se hace en Swing.
        // Nótese que hay que implementar todos los métodos de cada interfaz
        // Listener.
        boton.addSelectionListener(new SelectionListener() {
          public void widgetSelected(SelectionEvent e) {
            System.out.println("Se ha seleccionado el botón");
          }
          public void widgetDefaultSelected(SelectionEvent e) {
            System.out.println("Se ha seleccionado el botón");

          }
        });

        boton.addMouseListener(new MouseListener() {
          public void mouseDown(MouseEvent e) {
            System.out.println("Se ha movido el ratón hacia abajo");
          }
          public void mouseUp(MouseEvent e) {
            System.out.println("Se ha movido el ratón hacia arriba");
          }
          public void mouseDoubleClick(MouseEvent e) {
            System.out.println("Se ha hecho doble click en el botón");
```

```
            }
        });
            // Se muestra el shell; un shell es invisible salvo que se
    llame a open().
        shell.open();

        // Se entra en un bucle para leer y enviar los sucesos del
    usuario.
        while (!shell.isDisposed()) {
          if (!display.readAndDispatch()) {
            display.sleep();
          }
        }

// Se cierra el display y se liberan los recursos del sistema operativo
asociados.
        display.dispose();
      }

    }
```

2.9.3 EJEMPLO 3: EJEMPLOMENU.JAVA

```
import org.eclipse.swt.SWT;
import org.eclipse.swt.graphics.Image;
import org.eclipse.swt.widgets.*;

/**
 * Esta clase genera una ventana con una imagen y una barra
 de menús
 * Se necesita incluir el paquete org.eclips.swt.graphics.Image
```

```
 * para poder trabajar con imágenes en SWT.
 *
 * La clase Menú actúa como un contenedor para los objetos
Menultem.
 * La clase Menú admite los siguientes estilos: BAR,
DROP_DOWN y POP_UP.
 *
 *  BAR crea una barra de menús.
 *  DROP_DOWN crea un menú desplegable.
 *  POP_UP crea un menú contextual.
 *
 * La clase Menultem admite los siguientes estilos: CHECK,
CASCADE, PUSH, RADIO y  SEPARATOR.
 *
 *  CHECK crea un menú de casilla de verificación.
 *  CASCADE crea un menú en cascada con un submenú.
 *  PUSH crea un elemento estándar de menú.
 *  RADIO crea un menú de opción.
 *  SEPARATOR crea un separador de elementos de menú.
 *
 */

public class EjemploMenu{

    public static void main(String args[]) {
    Display display = new Display();
    Shell shell = new Shell(display);

    shell.setSize(200, 200);
    shell.setText("Ejemplo 1 de menús");
    shell.setImage(new Image(display,"c:\\eclipse.jpg"));
```

```
        Menu menu = new Menu(shell, SWT.BAR);

        shell.setMenuBar(menu);
         shell.open();

        while (!shell.isDisposed()){
           if (!display.readAndDispatch()) {
              display.sleep();
           }
        }

        display.dispose();
     }

  }
```

EVALUACION

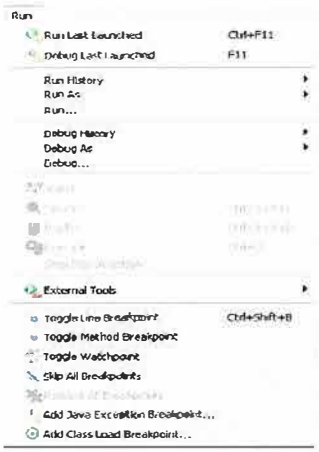
A. Explica brevemente tu concepto acerca de la Herramienta Eclipse y que aplicaciones se pueden desarrollar de esta Herramienta.

B. Realiza una opinión acerca de la plataforma de ejecución de Eclipse y de cada uno de los componentes que brinda su ejecución.

C. Indique a que Barras del entorno grafico pertenecen las siguientes interfaces:

1.

2.



D. Ejercicio propuesto: Primero practica con la instalación de Eclipse siguiendo el enunciado de este segundo capítulo, conoce el entorno de Eclipse con las explicaciones dadas en el primer capítulo creando nuevos proyectos.

AGRADECIMIENTOS

- Queremos agradecer de manera especial a los estudiantes de Ingeniería de Sistemas de la universidad Simón Bolívar, por el aporte de documentación valiosa en Gambas que sirvió como apoyo para la realización de este libro de Gambas.

  - ✓ **Giovanni Mendoza Gonzáles**.
  - ✓ **Hans Peluffo Díaz.**
  - ✓ **Edinson Barrios.**
  - ✓ **Beiro Escobar Solaez.**
  - ✓ **Tatiana Pantoja**
  - ✓ **Alexander Agudelo**
  - ✓ **Steven Polo**
  - ✓ **Paúl Leiva**
  - ✓ **Alfonso Valega**

- También queremos agradecer al docente de la universidad Simón Bolívar; Ingeniero **Sergio Jiménez Martínez**, por su colaboración, asesoría y aporte de materiales en Gambas.

- También queremos agradecer al docente de la universidad Simón Bolívar; Ingeniero **Fabio Moya**, por su colaboración, asesoría y aporte de materiales en Gambas.

ÍNDICE

BIBLIOGRAFIA

- Internet, **http://WWW.MONOGRAFIAS.COM**
- Internet, **http://linuxfocus.org** (Manual Gambas: Basic para Linux)
- Internet, **http://wiki.gnulinex.org/gambas/** (Manual Visual Introduction ToGambas)
- Internet, **http://gambas.sourceforge.net** (Manual Gambas tutorial.pdf)
- **http://gambas.gnulinex.org**/gambas_tutorial.pdf.
- http:// **gambas.sourceforge.net/**
- **www.eclipse.org\Desarrollo de Plug-ins.pdf\\.**
- http\\wipipedia\enciclopedia_libre\Linu_GNU\Linux.pdf\html
- WELSH, Mat. Instalación y Primeros Pasos: Linux.pdf (Traducción: Proyecto Lucas, versión en castellano) Copyright c 1992-1996.
- JOYANES AGUILAR, Luís y MUÑOZ CLEMENTE, Antonio. Microsoft Visual Basic 6.0.