UNIVERSIDAD LA SALLE

ACM5CLICK

Revista de Programación

ANEXO Manual de Java

La Paz – Bolivia

02 - 2011

Editorial

ACM5CLICK sigue con la continuidad de las grandes revistas, con el aporte de la investigación y las aplicaciones por parte de los estudiantes de la materia de Programación II.

ACM5CLICK es un pequeño aporte a la producción intelectual de la Universidad la Salle, la cuál continua apoyando a la producción de los estudiantes y docentes, por lo cual nuestro eterno agradecimiento.

La revista siempre brillará por su originalidad, en este número el valor agregado esta en el anexo que es un mini manual del lenguaje de programación JAVA,

La esperanza es seguir creciendo y que cada número siempre sea mejor, por lo cual, esperamos que este número sea del agrado de todos.

Atentamente,

Walter Carvajal

ÍNDICE

Pag.

"CENTRO MULTIMEDIA

Diego Rafael Sanabria Peñaloza
drsp22@hotmail.com
Universidad La Salle

Resumen.

El centro multimedia es un programa el cual reúne distintas aplicaciones de uso diario tales como música o internet en un solo lugar.

1. INTRODUCCIÓN

Muchas veces una sola aplicación no basta a la hora de entretenernos, a veces necesitamos juntarlas o incluso tener un rápido acceso a estas, desde música a videojuegos entre otros.

1.1 Registro

El registro de los usuarios hace un poco más limitado al programa y no tan abierto al público en general. El registro es directo y sencillo, está asociado a una base de datos creada para este programa y destinada a almacenar todos los usuarios con sus respectivas contraseñas.

1.2 Música

La música es una aplicación en el programa de uso más general. La mayoría de las personas le gusta escuchar música mientras realiza otras actividades, tales como trabajar o jugar videojuegos. Esta aplicación esta relacionada con el componente Windows media player y aun botón el cual abre una ventana con una dirección ya establecida.

1.3 Videos

Los videos actúan de forma similar a lo que es el método de uso de la música. Se hace uso también de un botón que abre una ventana con una dirección diferente, capaz de cargar los archivos deseados haciendo uso del Windows media player.

1.4 Videojuegos

El centro multimedia también contiene accesos directos a lo que son tres tipos de videojuegos, los cuales están ubicados e instalados en el ordenador. Hace uso también de botones, los cuales llaman al ejecutable de cada uno de los programas.

1.5 YouTube

Otra cualidad del centro multimedia es que tiene un rápido acceso a lo que son páginas web. Muchas veces se nos impide entrar a distintas páginas por muchas razones o el procedimiento para entrar es muy moroso. Solo basta con seguir las instrucciones en la ventana y el acceso será rápido.

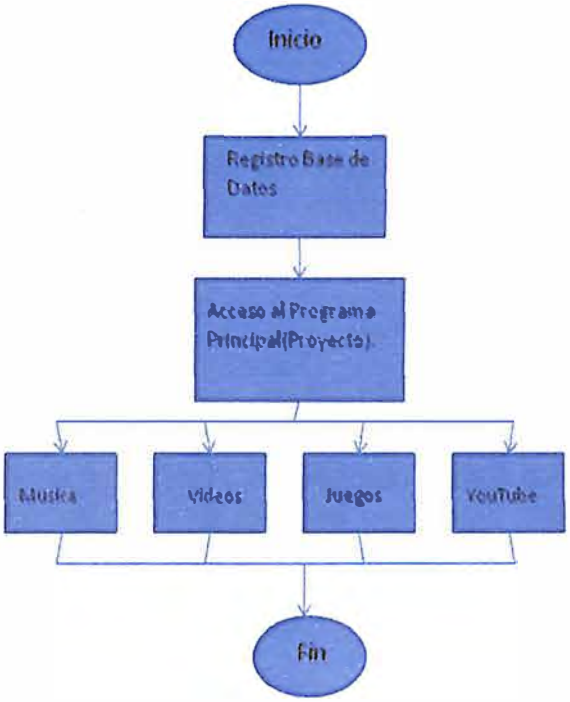
PROBLEMA

Este programa surge a partir de la necesidad nuevas formas de entretenimiento, en un ambiente monótono, donde no existen muchas o que también llegan a ser repetitivas.

2. MARCO TEORICO

Me baso en el uso de Programación con Base de Datos y Programación Orientada a Eventos.

3. ALGORITMOS



4. COMPONENTES Y METODOS

Forms, buttons, label, menú strip, openfile dialog, picture box, text box, Windows media player, System Process Diagnostics Start, Conexión Base de Datos, Access 2012, web browser.

5. PRUEBAS DE CORRIDO

6. CONCLUSIONES

Se ha hecho un programa cuyo fin es entretener de distintas formas sin tener que gastar mucho tiempo en acceder a estas.

7. RECOMENDACIONES

Sírvase a abrir las ventanas de Ayuda, facilitan el uso de las aplicaciones.

8. REFERENCIAS

- http://www.trucoteca.com/trucos-juegos/pc/starcraft-3934.html
- http://www.juegomania.org/trucos/pc/7325

"ROMPECABEZAS"

ANGELICA BETZI QUISPE VISALLA
betzi_unnuevodia@hotmail.com
UNIVERSIDAD LA SALLE

Resumen.

Con todo lo aprendido en el manejo del C# en el Visual Studio 2008 busco crear una aplicación que permita hacer que un juego tan simple como los rompecabezas, sea algo divertido.

1. INTRODUCCIÓN

Los juegos es un entretenimiento, tanto para niños, jóvenes, hasta adultos. Y constituye la ocupación principal de los mismos.
Por medio de los juegos, más que todo los niños experimentan, aprenden, etc. Es por eso que no se debe limitar al niño en esta actividad.
Por lo cual he decidido crear un juegomuy clásico que son los rompecabezas.

a) Rompecabezas

Los rompecabezas o puzles son piezas comúnmente planas que combinadas correctamente forman una figura, un objeto o una escena.

Fueron inventados en 1762 por el londinense John Spilsbury y un siglo después empezaron a fabricarse en serie.

Varían por su forma, tamaño, tema, material con que están hechos y grados de dificultad de acuerdo a la cantidad y la forma de sus piezas; pueden ir desde 15 hasta las 12, 0000 piezas.

Por diversión, entretenimiento o como una forma de relajarse; armar rompecabezas es una actividad tanto para chicos como para grandes y de la cual se pueden obtener muchos beneficios.

b) Tipos de rompecabezas

Entre sus tipos están los que tienen piezas que se enlazan entre sí pero no tienen tablero. Aparte están los que sus piezas se insertan en un tablero, donde cada pieza tiene un orificio único.

La variedad es muy amplia tan sólo por la cantidad de piezas que puede tener un rompecabezas: 15, 25, 75, 100 500, 1000 hasta 12000.

Por otro lado algunos contienen una imagen de cómo debe quedar armado, lo que le facilita al niño el proceso de análisis-síntesis; en este caso ya no es necesario anticipar el todo sino solamente de reconstruirlo. Hay otros que no muestran el modelo armado por lo que es necesario construir diferentes hipótesis sobre el mismo.

Aquellos que tienen cortes sinuosos e irregulares en sus piezas, facilitan la estrategia del armado ya que se pueden guiar por el encaje de las mismas para su construcción; otros ofrecen cortes rectos, pudiendo ser más difíciles de armar porque sus pistas son menos.

2. PROBLEMA

La manera de jugar y crear juegos ha cambiado con los avances de la tecnología.
Por lo cual buscar un juego seguro para la mente de un niño es un poco complicado.
Este proyecto implica en que el usuario pueda seleccionar el nivel de dificultad, y a medida que se juegue pasar de un niveles

3. MARCO TEORICO

Me baso en el uso de Programación Orientada a Objetos (POO), Y el funcionamiento de movimientos de piezas dentro de c#.

4. OBJETOS

Objetos:
Label, GrupBox, TextBox,Button, PictureBox, Timer, Buton,

5. CONCLUSIONES

En fin he creado una herramienta más para poder disfrutar de los juegos por medio del programa realizado.

ANEXO. MANUAL DE JAVA

Introducción

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre diciembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java aún no lo es).

El lenguaje Java se creó con cinco objetivos principales:

1. Debería usar la metodología de la programación orientada a objetos.
2. Debería permitir la ejecución de un mismo programa en múltiples sistemas operativos.
3. Debería incluir por defecto soporte para trabajo en red.
4. Debería diseñarse para ejecutar código en sistemas remotos de forma segura.
5. Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos, como C++.

Instalación

Después de descargar e instalar The Java SE Development Kit (JDK) necesitamos decirle a nuestro computador donde está el compilador de JAVA, para esto vamos a **MI PC** y hacemos clic derecho con el botón del mouse y vamos a **Propiedades** y después nos vamos a la pestaña donde dice **Opciones Avanzadas**. Damos Clic en Variable de entorno. Luego buscamos la variable llamada "**Path**" en las Variables del Sistema y hacemos click en **modificar**.

Al final del campo llamado "**valor de variable**" escribimos la ubicación del compilador de JAVA, por ejemplo: "**;C:\Java\jdk1.6.0_02\bin**".

En la misma ventana de Variable de entorno. Creamos una variable llamada **Classpath**, en entrada ingresamos la dirección de la variable. En este caso la dirección es ;**C:\Java\jdk1.6.0_02\src.zip** Para terminar hacemos clic en **Aceptar** y cerramos todas las ventanas. Ahora vamos al **Menú Inicio -> Todos los programas -> Accesorios -> Símbolo del sistema.**
Aparecerá ante nosotros una Consola de DOS, en ella escribimos **java** y hacemos un **enter** y después tecleamos **javac** para ver si el sistema encuentra el interprete de JAVA. Si nos sale algún error, hay que volver atrás a revisar la configuración de las variables de entorno del sistema.

Tipos de datos

A toda variable que se use en un programa, se le debe asociar (generalmente al principio del programa) un tipo de dato específico.
Un tipo de dato define todo el posible rango de valores que una variable puede tomar al momento de ejecución del programa y a lo largo de toda la vida útil del propio programa.

Los tipos de datos más comunes en java son:

Keyword	Description	Size/Format
(integers)		
"byte"	Byte-length integer	8-bit two's complement
"short"	Short integer	16-bit two's complement
"int"	Integer	32-bit two's complement
"long"	Long integer	64-bit two's complement
(real numbers)		
"float"	Single-precision floating point	32-bit IEEE 754
"double"	Double-precision floating point	64-bit IEEE 754
(other types)		
"char"	A single carácter	16-bit Unicode character
"boolean"	A boolean value ("true" or "false")	true or false

Para el caso de strings se debera usar la Clase String que tiene dos constructores, de momento entenderemos esto último como dos maneras de crearse, ej;

1. String nombre= new String();
2. String ciudad= new String("Bolivia");

Estructuras

Estructuras Selectivas

Sentencia if

La sentencia if (o si condicional), le permite a un programa decidir, mediante la evaluación de una condición, ejecutar una u otra acción o acciones.

La sintaxis General es La siguiente:

if(condicion1)

Accion1;

En donde:

Condición 1: Representa una expresión que puede ser del tipo booleana.

Acción 1: es la acción que, al evaluar la condición como verdadera, se ejecutará.

Si son varias acciones, van entre llaves.

Sentencia elseif

Esta estructura, es una consecuencia de las estructuras if anidadas, sus formato es el siguiente:

if(condicion1)

Sentencia 1;

elseif(condicion2)

Sentencia 2;

elseif(condicion3)

Sentencia 3;

else

Sentencia n;

Funciona de la siguiente manera.

Se evalúa la primera condición, si resulta verdadera se ejecuta la sentencia 1, y se continúa con la ejecución del programa; de lo contrario, de evalúa la condición 2, si resulta verdadera, se ejecuta la sentencia 2, de lo contrario se evalúa la condición 3 y asi sucesivamente. Si al evaluar todas las condiciones, ninguna resulta verdadera, se ejecuta el bloque del else.

Sentencia switch

Esta sentencia, permite ejecutar, una u otra u otra acción, al evaluar una condición, cuyo resultado es el que indica que bloque (o bloques) de instrucciones se van a ejecutar.

Su sintaxis es la siguiente:

switch(expresión)

case 1:

Sentecia 1;

break;

case 2:

Sentecia 2;

break;

default:

Sentencias;

break;

En donde, expresión es una condición que, al evaluarla, nos indicará que camino debemos seguir. Además ésta puede ser, una expresión entera char, byte, int y short. Además que, la expresión constante que acompaña a la palabra reservada *case* debe ser del mismo tipo que *expresión.* La cláusula default es opcional y puede omitirse.

Estructuras Repetitivas

En Java podemos encontrar tres tipos de ciclos:

- Entrada Asegurada (while)
- Ciclo Controlado Por Contador (for)
- Hacer Mientras (do.. while)

while(condición)

Acción;

Funciona de la siguiente manera; primero evalua la condicion, si da como resultado cierta realiza la accion, luego vuelve a evaluar la condicion; si su resultado es falso, se sale del ciclo y continua con la ejecucion del programa. Hay que tener mucho cuidado, cuando trabajamos con ciclos, ya que podemos caer en un ciclo infinito, es decir que

nunca se sale de él. Por lo cual en las acciones debemos siempre colocar algo que haga que se modifique el resultado de la condición, lo cual puede ser una bandera, un contador o un acumulador.

Formato:

while (condición)

sentencia;

}

for(valor inicial; condición; incremento)

accion;

Donde:

Valor inicial: es el valor con el cual inicializamos nuestra variable de control.

Condición: si la cumple, ejecuta la acción o acciones e incrementa o decrementa la variable de control, sino la cumple la condición, se sale del ciclo.

Incremento; que puede ser positivo o negativo (decremento).

Formato:

for(valor inicial; condición; incremento)

sentencia;

Ciclo Do... while

Es te ciclo funciona de la siguiente manera, realiza la acción o conjunto de acciones, luego evalúa una condición de resultar cierta vuelve a realizar la/s accion/es. Cuando sea falsa, se sale del ciclo.

Formato :

do {

sentencia;

} while(<condicion>);

La diferencia fundamental, entre el ciclo while y do...while, es que en este ultimo, las sentencias se realizarán por lo menos una vez, en cambio, con while, solo se cumplirán *mientras* se cumpla la condición, lo cual puede ser nunca.

Herencia y Polimorfismo

Tipos de herencia

En java existen dos tipos de herencia, herencia simple y herencia múltiple.

Herencia simple

Un objeto puede extender las características de otro objeto y de ningún otro, es decir, que solo puede heredar o tomar atributos de un solo padre o de una sola clase.

```
public class Mamifero{
   private int patas;
   private String nombre;
   public void imprimirPatas(){
     JOptionPane.showMessageDialog(null," Tiene + patas +
"patas\n","Mamifero",JOptionPane.INFORMATION_MESSAGE);
    }
    public Mamifero(String nombre, int patas){
      this.nombre = nombre;
      this.patas = patas;
    }
  }

  public class Perro extends Mamifero {
    public Perro(String nombre){
      super(nombre, 4);
    }
  }

  public class Gato extends Mamifero {
    public Gato(String nombre){
      super(nombre, 4);
    }
  }

public class CrearPerro {
  public static void main(String [] args) {
    Perro perrito = new Perro("Pantaleon");
    perrito.imprimirPatas();   /*Está en la clase mamífero*/
  }
}
```

Herencia múltiple

Un objeto puede extender las características de uno o más objetos, es decir, puede tener varios padres.

Limitaciones de la herencia

Todos los campos y métodos de una clase son siempre accesibles para el código de la misma clase.

Para controlar el acceso desde otras clases, y para controlar la herencia por las subclases, los miembros (atributos y métodos) de las clases tienen tres modificadores posibles de control de acceso:

- Public: Los miembros declarados public son accesibles en cualquier lugar en que sea accesible la clase, y son heredados por las subclases.
- Private: Los miembros declarados private son accesibles sólo en la propia clase.
  Protected: Los miembros declarados protected son accesibles sólo para sus subclase.

El polimorfismo

El polimorfismo es un concepto de la programación orientada a objetos que nos permite programar en forma general, en lugar de hacerlo en forma especifica. En general nos sirve para programar objetos con características comunes y que todos estos compartan la misma superclase en una jerarquía de clases, como si todas fueran objetos de la superclase. Esto nos simplifica la programación.

Existen varias formas de polimorfismo:

- Cuando invocamos el mismo nombre de método sobre instancias de distinta clase.
- Cuando creamos múltiples constructores.
- Cuando vía subtipo asignamos una instancia de una subclase a una referencia a la clase base.

Características de Java

Orientado a objetos

La primera característica, orientado a objetos ("OO"), se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para OO, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones. Así, los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. El principio es separar aquello que cambia de las cosas que permanecen inalterables. Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema software. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos.

Otra de las grandes promesas de la programación orientada a objetos es la creación de entidades más genéricas (objetos) que permitan la reutilización del software entre proyectos, una de las premisas fundamentales de la Ingeniería del Software. Un objeto genérico "cliente", por ejemplo, debería en teoría tener el mismo conjunto de comportamiento en diferentes proyectos, sobre todo cuando estos coinciden en cierta medida, algo que suele suceder en las grandes organizaciones. En este sentido, los objetos podrían verse como piezas reutilizables que pueden emplearse en múltiples proyectos distintos, posibilitando así a la industria del software a construir proyectos de envergadura empleando componentes ya existentes y de comprobada calidad; conduciendo esto finalmente a una reducción drástica del tiempo de desarrollo.

Independencia de la plataforma

La segunda característica, la independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo, tal como reza el axioma de Java, "'write once, run anywhere'".

El recolector de basura

En Java el problema de las fugas de memoria se evita en gran medida gracias a la recolección de basura (o *automatic garbage collector*).

El programador determina cuándo se crean los objetos y el entorno en tiempo de ejecución de Java (Java runtime) es el responsable de gestionar el ciclo de vida de los objetos. El programa, u otros objetos pueden tener localizado un objeto mediante una referencia a éste. Cuando no quedan referencias a un objeto, el recolector de basura de Java borra el objeto, liberando así la memoria que ocupaba previniendo posibles fugas (ejemplo: un objeto creado y únicamente usado dentro de un método sólo tiene entidad dentro de éste; al salir del método el objeto es eliminado). Aun así, es posible que se produzcan fugas de memoria si el código almacena referencias a objetos que ya no son necesarios– es decir, pueden aún ocurrir, pero en un nivel conceptual superior. En definitiva, el recolector de basura de Java permite una fácil creación y eliminación de objetos, mayor seguridad y puede que más rápida que en C++.

Ff.aro_f¿würl<s de .Java

Framework define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

En el desarrollo de software, un framework es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un pro ecto.

Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

Existen diferentes tipos de Frameworks los más conocidos son Struts, Spring, JSF, Maven, Hibernate, etc.

El Spring Framework (también nocido simplemente como Spring) es un framework de código abierto de desarrollo de aplicaciones para la plataforma Java. La primera versión fue escrita por Rod Jonhson, quien lo lanzó primero con la publicación de su libro Expert One-on-One Java EE Oesign and Oevelopment (Wrox Press, octubre 2002). También hay una versión para la plataforma .NET, Spring .NET.

El framework fue lanzado inicialmente bajo Apache 2.0 License en junio de 2003. El primer gran lanzamiento hito fuede la versión 1.0, que apareció en marzo de 2004 y fue seguida por otros hitos en septiembre de 2004 y marzo 2005.

A pesar de que Spring Framework no obliga a usar un modelo de programación en particular, se ha popularizado en la comunidad de programadores en Java al ser considerado como una alternativa y sustituto del modelo de Enterprise JavaBean. Por su diseño el framework ofrece mucha libertad a los desarrolladores en Java y soluciones muy bien documentadas y fáciles de usar para las prácticas comunes en la industria.

Mientras que las características fundamentales de este framework pueden emplearse en cualquier aplicación hecha en Java, existen muchas extensiones y mejoras para construir aplicaciones basadas en web por encima de la Plataforma empresarial de Java (Java Enterprise Platform).

Webgrafía

http://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

http://es.wikipedia.org/wiki/Spring_Framework

http://www.luiskano.net/blog/2007/09/07/como-instalar-java-jdk-en-windows-xp/

http://www.programacionfacil.com/java:tipos_de_datos

http://www.monografias.com/trabajos42/manual-de-java/manual-de-java3.shtml
http://es.wikipedia.org/wiki/Herencia_(inform%C3%A1tica)

http://dis.um.es/~lopezquesada/documentos/IES_1011/DFSI/curso/UT6/java/cap9/index.html