

Entorno de desarrollo NetBeans

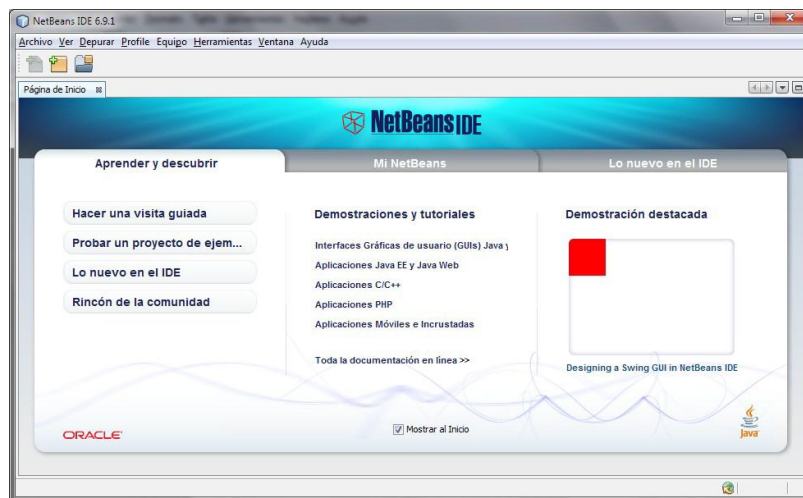
NetBeans IDE: Es un entorno de desarrollo - una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

Instalar NetBeans:

Para ejecutar NetBeans necesitará cumplir los requisitos técnicos detallados a continuación.

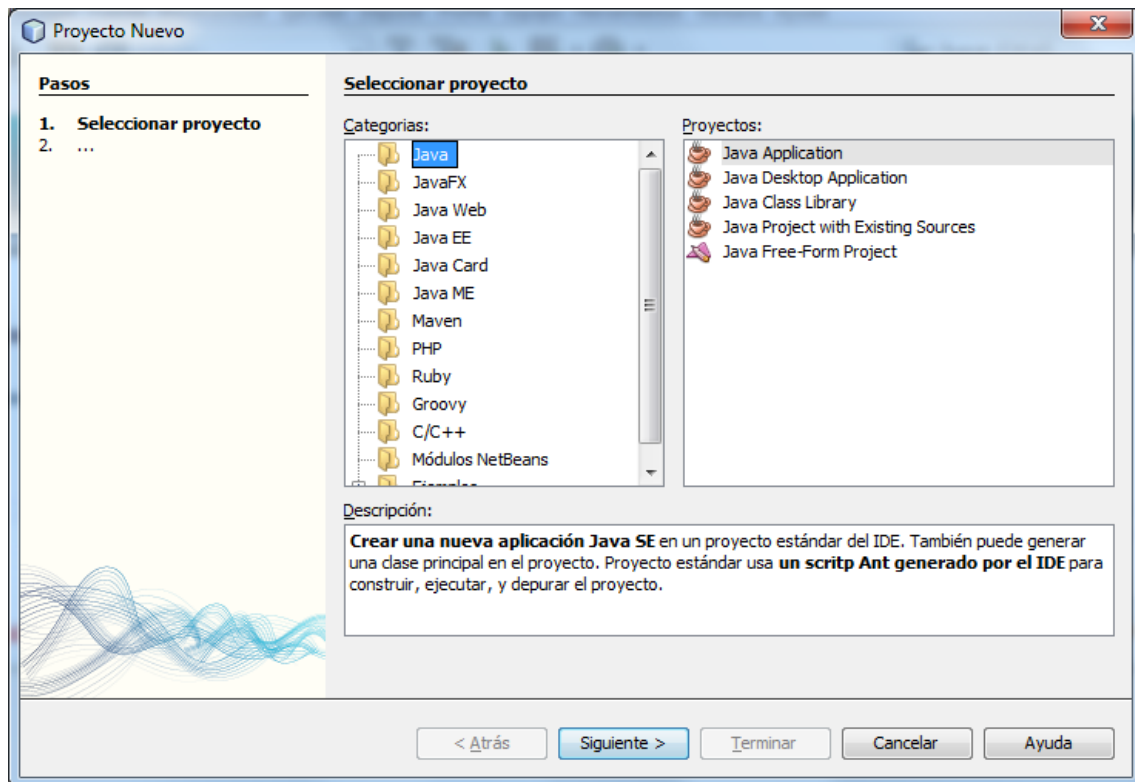
Componente	Características
Una plataforma compatible: NetBeans IDE6.9	Windows XPSP3/Vista/7; Linux (x86/x64); Solaris (x86/x64); Solaris (sparc); Mac OS X; OS independent.
Espacio de disco suficiente:	Dependiendo de la versión que desea instalar la que menos utiliza es de 750 MB y la completa de 1 GB como requerimiento.
Memoria RAM Suficiente:	512 MB como mínimo, 2GB recomendado
Java JDK 6 Update 13:	La versión 6.9.1 del IDE no se puede instalar o ejecutar con JDK 5.0.

Después de instalado el entorno grafico de desarrollo NetBeans e iniciarlo, se mostrará de esta manera:



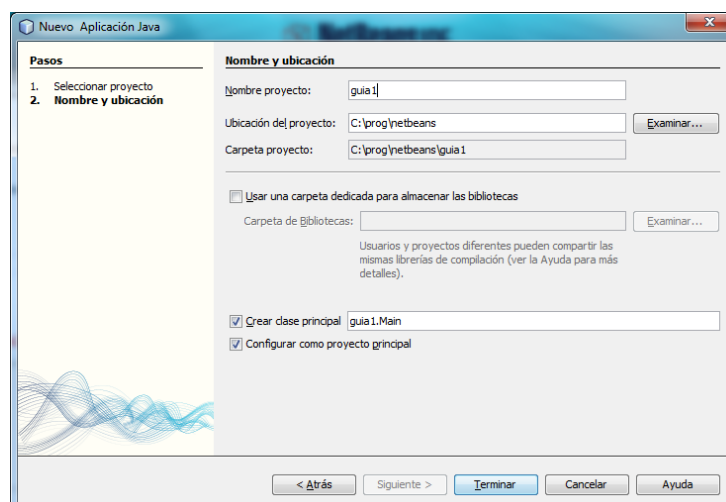
Parte I.- Abriendo un Nuevo Proyecto (consola):

1. En el IDE, seleccione Archivo> Proyecto Nuevo .
2. En el asistente de Proyecto, ampliar la categoría de Java y seleccione Java Application, como se muestra en la siguiente figura. A continuación, haga clic en Siguiente.



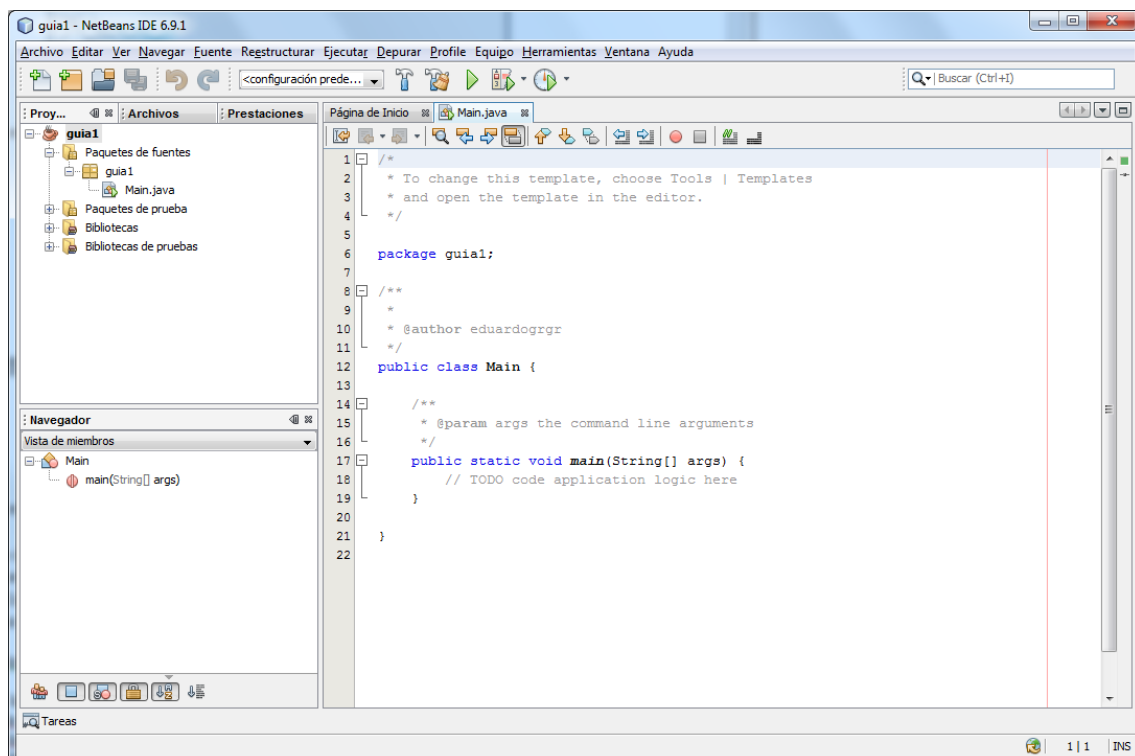
3. En el Nombre y ubicación de la página del asistente, haga lo siguiente (tal y como se muestra en la siguiente figura. Seguiremos utilizando la carpeta **C:\Prog** para almacenar las prácticas realizadas en el módulo, pero ahora crearemos una subcarpeta llamada NETBEANS donde se guardarán las prácticas realizadas en este segundo trimestre):

- En el campo Nombre del proyecto, es **guia1**.
- En el campo Crear la clase principal, es **guia1.Main**.



4. Haga clic en Terminar.

El proyecto es creado y abierto en el IDE. Debe ver los siguientes componentes:



- **Panel de Proyectos**, que contiene una vista en árbol de los componentes del proyecto, incluyendo archivos de origen, las bibliotecas que depende de su código, y así sucesivamente. A través de la expansión o contracción de los nodos en el árbol uno puede ir navegando a través de los elementos que conforman el proyecto.
- **Panel de Archivos**: permite navegar a través de las carpetas del proyecto. A veces algunos elementos pueden no aparecer en el árbol del panel de proyectos, pero si en el panel de Archivos. Por ejemplo, los archivos de salida al ejecutar o debugear un proyecto.
- **Panel de Prestaciones**: maneja conexiones a algunos servicios. Los más comunes son los de Bases de datos. Son una ayuda sumamente útil si uno realiza proyectos que interactúan con servidores de base de datos.
- **Panel del Navegador**, que usted puede utilizar para navegar rápidamente entre los elementos seleccionados dentro de la clase.
- La **ventana de Edición de Código** con un archivo llamado Main.java abierto.

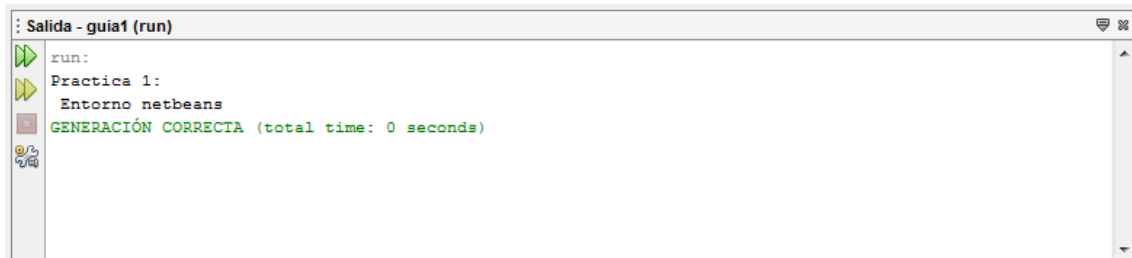
Agregamos la siguiente línea dentro del método main, exactamente después del comentario //TODO code application logic here

System.out.println("Practica 1: \n Entorno netbeans");

Luego compilamos y ejecutamos el programa con el botón



la ejecución aparecerá en el **Panel de salida**:

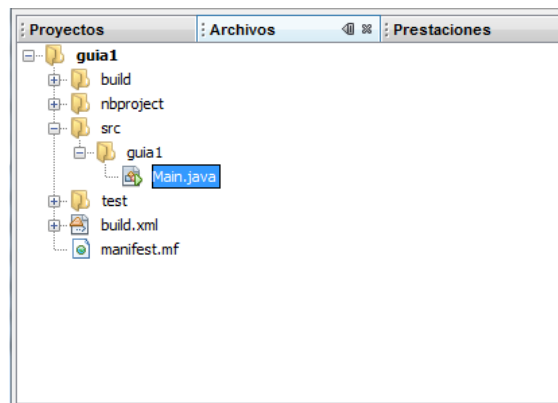


Tener en cuenta las siguientes consideraciones:

- Java es sensible a mayúsculas y minúsculas. El no utilizar la combinación apropiada de letras minúsculas y mayúsculas para un identificador, generalmente produce un error de compilación.
- Cuando usted guarda su declaración de clase **public** en un archivo, el nombre de éste debe ser el nombre de la clase, seguido de la extensión **".java"**. Para nuestra aplicación, el nombre del archivo es **Main.java**.
- Todas las extensiones de clases en Java se guardan en archivos que terminan con la extensión **".java"**.
- Es un error que una clase **public** tenga un nombre de archivo que no sea idéntico al nombre de la clase (más la extensión .java). Por lo tanto, es también un error que un archivo contenga dos o más clases **public**.
- Si un archivo contiene la declaración de una clase, es un error que no finalice con la extensión **.java**. Si se omite esa extensión, el compilador de java no podrá compilar la declaración de la clase.

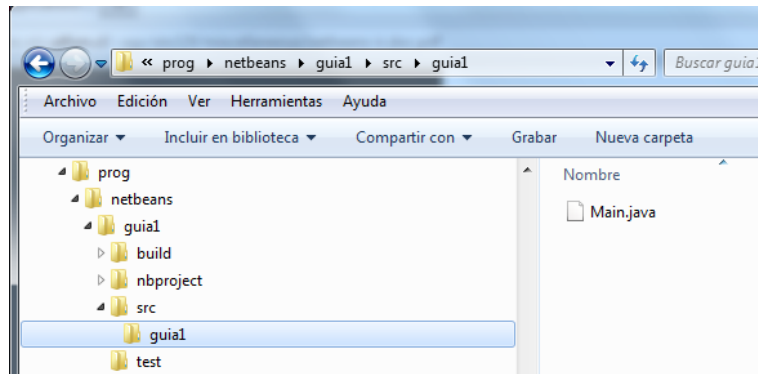
Ubicación de los Archivos de un Proyecto.

En la figura siguiente se muestra el panel de archivos, tras haber generado nuestro proyecto:



Muestra los directorios y archivos generados al crear el proyecto **guia1**. El código fuente de una clase se guarda en un archivo que tiene el mismo nombre de la clase y con la extensión **".java"**, por ejemplo, Main.java está dentro de la carpeta guia1 (el nombre del paquete) que a su vez está dentro de la carpeta "scr" (donde se almacenan los archivos fuentes de este proyecto) y que a su vez está dentro de la carpeta "guia1" (que contiene todos los archivos del proyecto).

En la siguiente imagen puedes ver la ubicación en nuestro disco duro:




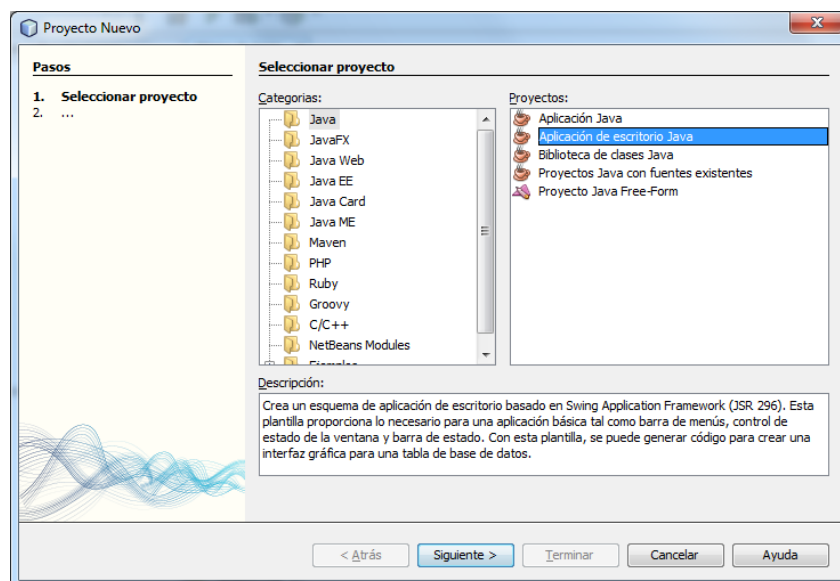
Pasamos a utilizar NETBEANS para crear proyectos de Aplicaciones con Interfaz gráfica.

Para ello cerramos el proyecto actual. (Menú ARCHIVO→CERRAR PROJECT).

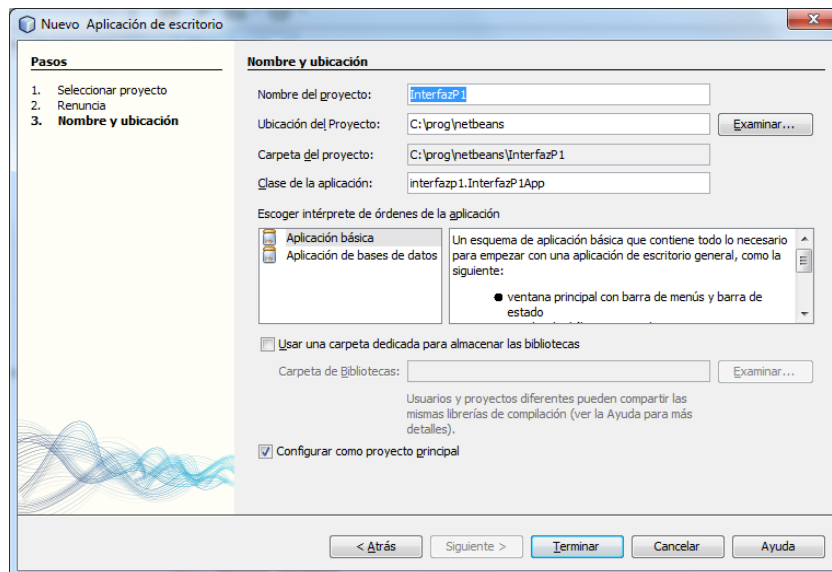
PARTE II. Ejemplo del Desarrollo de un Proyecto con Interfaz Gráfica de Usuario. (GUI)

Vamos a crear un nuevo proyecto llamado **InterfazP1** pero ahora utilizaremos una GUI.

1. Puede presionar el botón  o el menú ARCHIVO -> PROYECTO NUEVO.

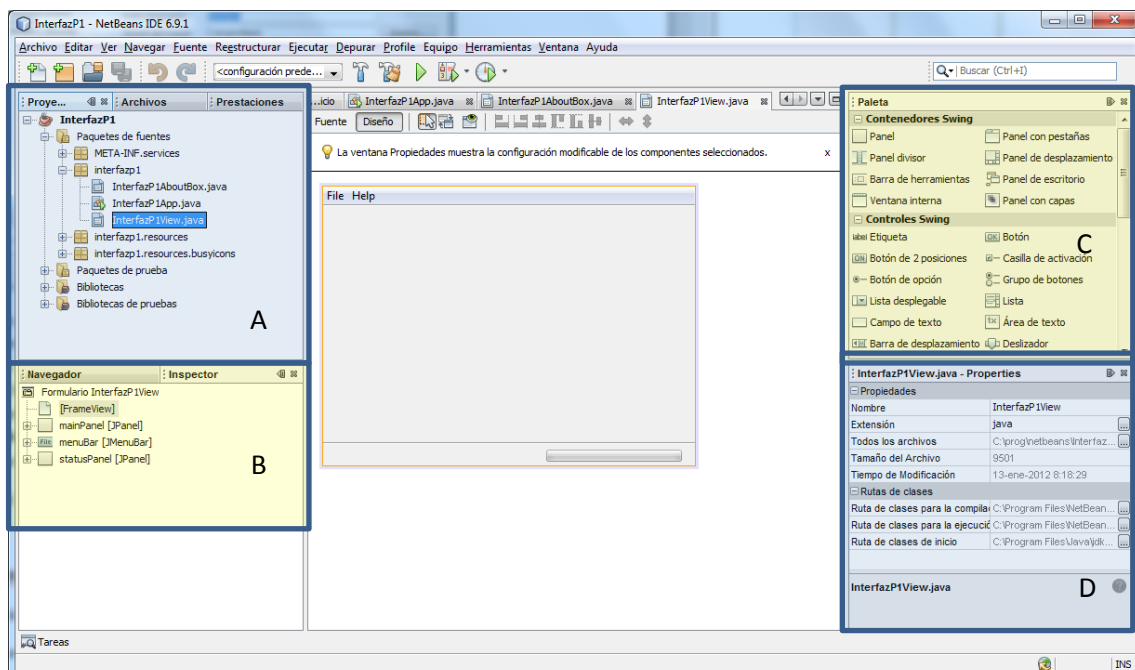


2. Elija el tipo de proyecto. Para el ejemplo elegiremos “Aplicación de escritorio Java” (aplicación con GUI).
3. Hacemos click en Siguiente.



- Elegimos el nombre del proyecto (**InterfazP1**) y la carpeta donde guardarlo. Dejaremos el nombre de la clase main que se sugiere y dejaremos marcado que será nuestro proyecto principal (es el que se ejecutará o debugeará por defecto si uno presiona algunos botones correspondientes como ya veremos en prácticas posteriores). Para este proyecto, en Escoger intérprete de órdenes de la aplicación, seleccionaremos una **Aplicación básica**.
- Presionamos el botón “TERMINAR”.

Veremos algo así:



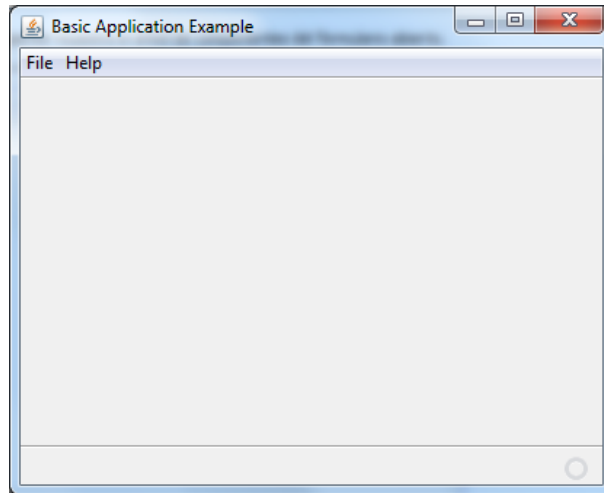
Panel A: Nuestro Proyecto.

Panel B- INSPECTOR: Componentes SWING incorporados en nuestro Formulario.

Panel C- PALETA: Caja de herramientas (Toolbox) con los componentes SWING.

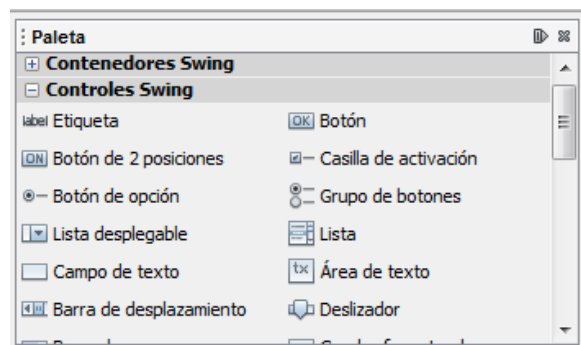
Panel D- PROPERTIES: Propiedades del componente SWING seleccionado o de un elemento elegido en el panel A.

Ejecutar el proyecto con el botón

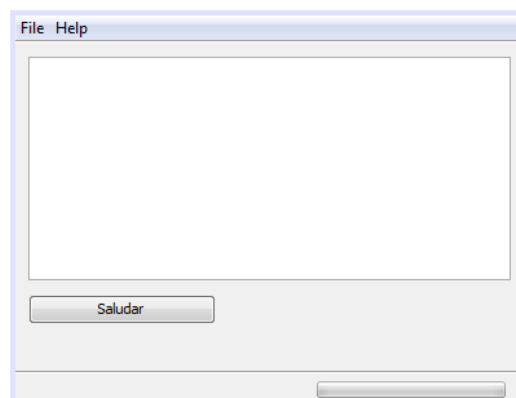


PARTE II. Editando el Form (GUI)

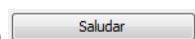
Utilizando la sección controles Swing del panel PALETA.



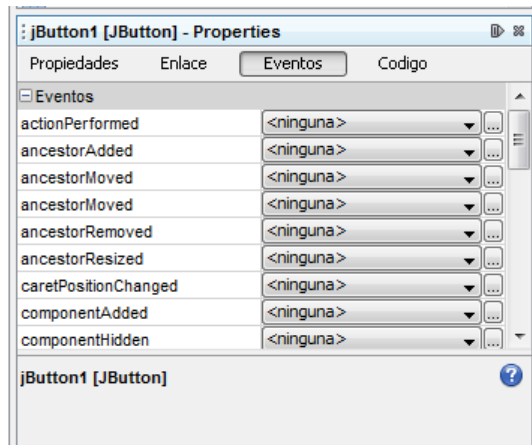
Agreguemos un Área de texto y un botón.



Ahora creamos una acción que ha de ejecutarse cuando hagamos clic en el botón



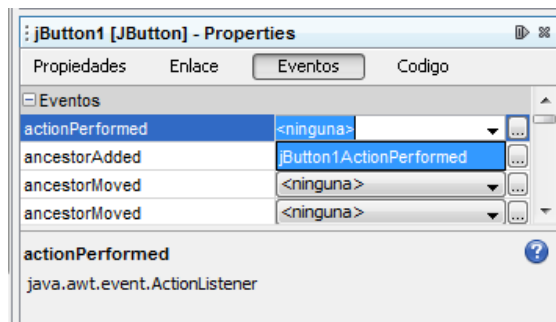
Una forma de hacer esto es, en primer lugar seleccionar el botón y hacer clic en la ficha “EVENTOS” que aparece en el panel de Properties.



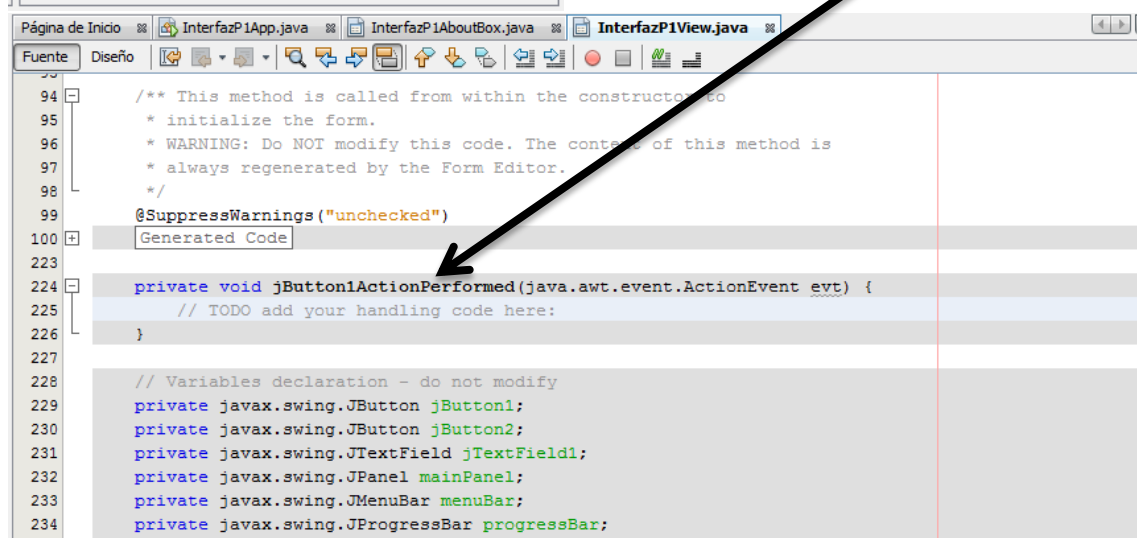
Esta tarjeta muestra una serie de eventos que pueden ser disparados por el botón.

En este caso, el que nos interesa es “actionPerformed”, el cual está al inicio de la lista.

Hacemos clic sobre el campo de texto que aparece a la derecha de “actionPerformed” y que dice “<ninguna>”.



Se despliega una lista donde aparece jButton1ActionPerformed. Al hacer clic en él nos creará automáticamente el método, e incluso nos llevará automáticamente al Editor de código donde introduciremos el código deseado para este Evento.



En la última imagen podemos ver la declaración del método y también están resaltados dos botones en la esquina superior izquierda: “Fuente” y “Diseño”. Estos nos permiten ver el código de la clase o la representación gráfica de su Form (como lo veíamos antes). “Fuente” nos muestra el código y “Diseño” nos muestra la representación gráfica. Claramente sólo las clases que tienen una representación gráfica tienen estos botones.

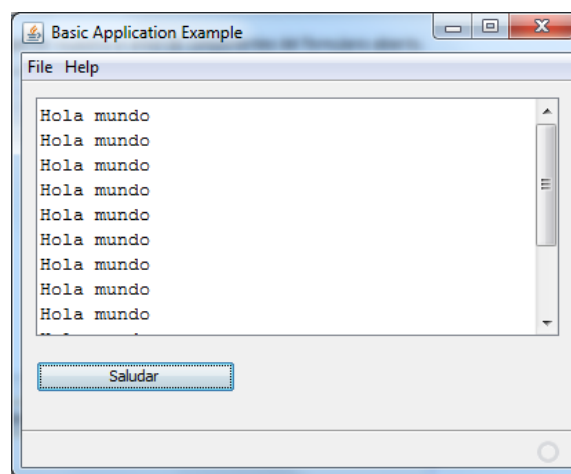
Parte II. Agregando Código.

Vamos a la definición del método que acabamos de crear (¿No sabes como ir a la definición? Recuerda el “Navegador” o también puedes ir al evento “actionPerformed” del botón en Properties”).

Agrega el código que aparece en la imagen.

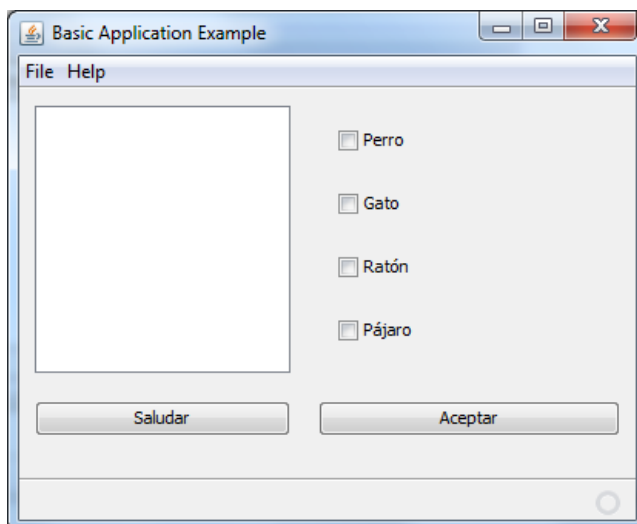
```
222 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
223     JTextArea1.append("Hola mundo \n"); // TODO add your handling code here:  
224 }
```

Como podrás suponer, este código hace que cada vez que se haga clic en el botón se agregue la línea “Hola mundo” en el área de texto de nuestro Form.



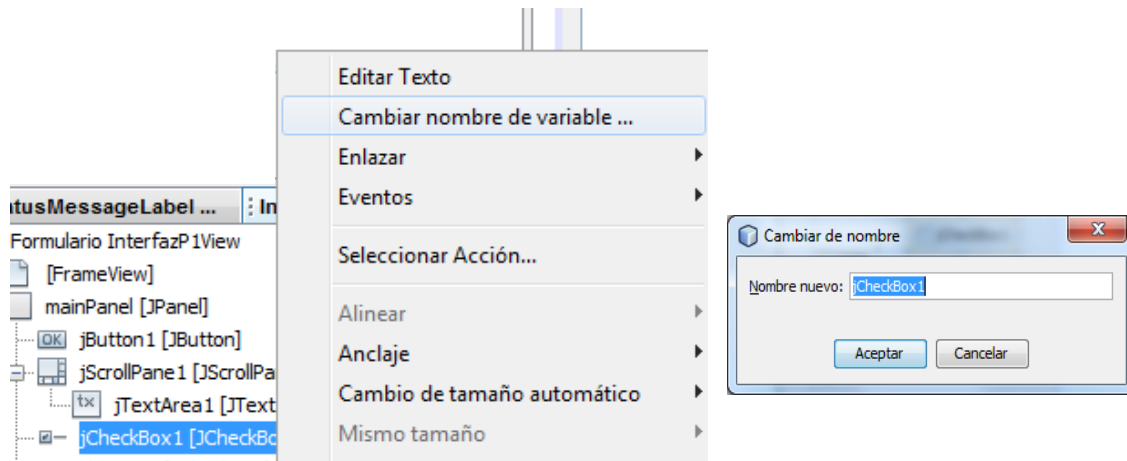
Ejecuta el programa con el botón correspondiente y prueba el resultado.

Rediseñar el formulario para que quede con el siguiente aspecto:

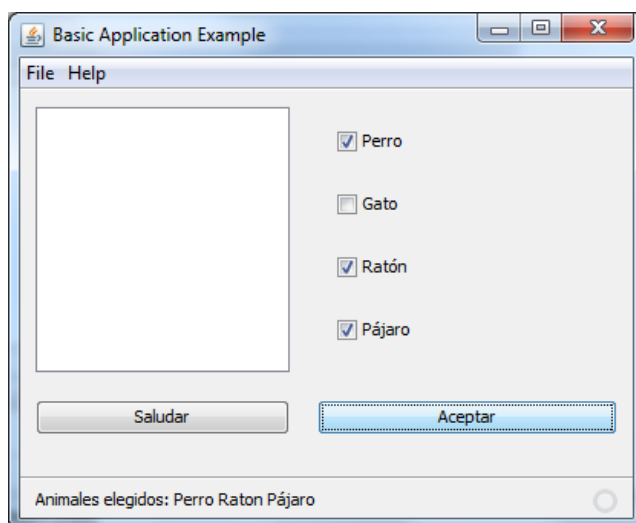


1. Añade cuatro cuadros de verificación. Estos cuadros son objetos del tipo JCheckBox. Cambia el texto de ellos, de forma que aparezca “Perro”, “Gato”, “Ratón” y “Pájaro”.

2. Debe cambiar el nombre de cada uno de dichos objetos. Se llamarán: chkPerro, chkGato, chkRaton, chkPajaro. (**Panel INSPECTOR** seleccionamos el objeto deseado, botón derecho → Cambiar nombre de variable.



Insertar el código necesario para que al pulsar en el botón ACEPTAR de nuestro formulario, ocurra lo siguiente:



En la etiqueta statusMessageLabel que creó automáticamente NETBEANS al crear nuestra aplicación, aparecerá un mensaje indicando qué animales han sido “seleccionados”.

Comprimir las carpetas **guia1** e **InterfazP1** correspondientes a la Práctica 1 de Netbeans y subirla a la plataforma.

FIN PRIMERA PRÁCTICA NETBEANS