



INTRODUCCIÓN

La guía de Laboratorios de la materia de Introducción a la Informática (INF110) fue elaborada con la finalidad de que complementar las clases teóricas de la materia de introducción a la informática e iniciar al estudiante en el arte de programar computadores ya sea usando un computador y/o un dispositivo móvil. La guía tiene 3 partes:

En la primera parte el alumno podrá iniciarse en el uso de las herramientas de desarrollo que serán: Embarcadero RAD Studio con Delphi y PascalGUI.

En la segunda parte el alumno podrá iniciarse en la programación de computadores, para ello se harán ejercicios de programación para resolver ya sea con Delphi o con Pascal.

En la tercera parte el alumno podrá realizar ejercicios de programación usando módulos, ya sea con Delphi o Pascal.

CONTENIDO

- Laboratorio 01. Paquetes de desarrollo
- Laboratorio 02. Estructura de una aplicación
- Laboratorio 03. Componentes 1ra parte (Programación conducida por eventos)
- Laboratorio 04. Componentes 2da parte (propiedades)
- Laboratorio 05. Componentes 3ra parte (Eventos)
- Laboratorio 06. Tipos de Datos y Expresiones
- Laboratorio 07. Estructuras de control Alternativas
- Laboratorio 08. Estructuras de control repetitivas
- Laboratorio 09. Funciones y Procedimientos
- Laboratorio 10. Paso de parámetros
- Laboratorio 11. Módulos
- Laboratorio 12. Realizando un proyecto.



Laboratorio 01. Paquetes de desarrollo

En esta experiencia se verán los conceptos relacionados a paquetes

1. Objetivos.

Conocer las características de un paquete de desarrollo

Conocer las características de Embarcadero RAD Estudio

2. Desarrollo.

2.1 Conceptos.

Proceso. Es una lista de instrucciones que realizan una tarea específica.

Módulo. Es un conjunto de procesos y/o declaraciones que se almacenan en un archivo. Un módulo puede ser utilizado o puede formar parte de uno o más programas.

Aplicación/Programa. Una aplicación es un conjunto de instrucciones que realizan una o más tareas relacionadas a una actividad. Una aplicación está conformada por uno o más módulos, donde uno de ellos es el módulo principal. Ej. Excel, Word, Delphi, etc.

Librería. Es un conjunto de módulos que puede ser utilizado para el desarrollo de aplicaciones.

Paquete. Un paquete es un conjunto de programas que puede ejecutarse individualmente o dentro de un entorno integrado.

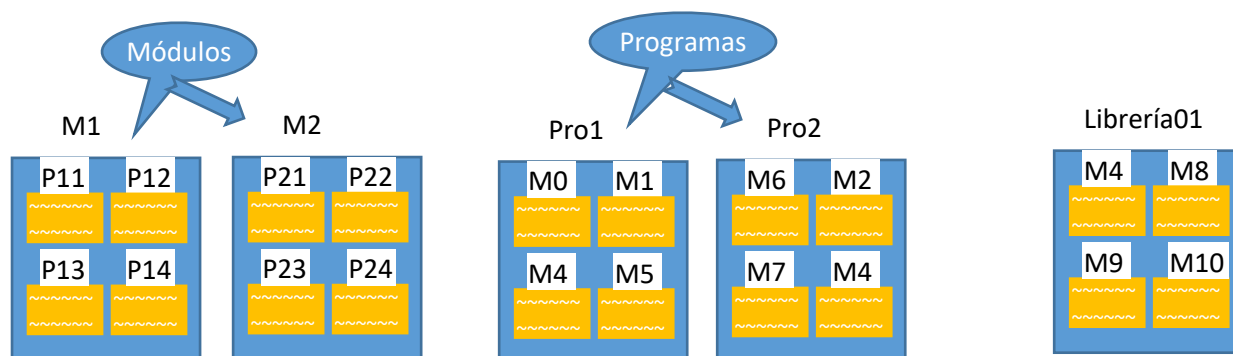
Ej. Microsoft Office (que viene con Word, Excel, etc)

Microsoft Visual Estudio (C#, Visual Basic, Visual C++)

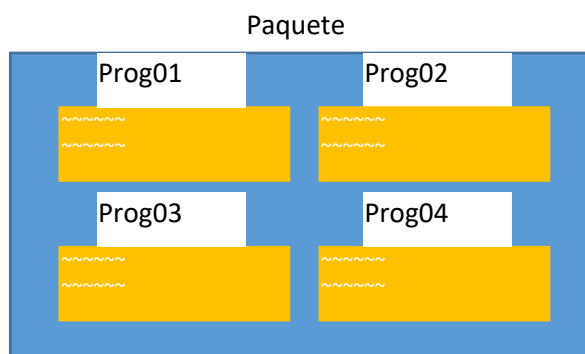
RAD Studio (Delphi, C++ Builder, compilador, Depurador, etc)

Paquete de desarrollo. Un paquete de desarrollo, es un conjunto de programas que permiten el desarrollo de aplicaciones, es decir, es un conjunto de programas para hacer programas. Estos paquetes pueden ser tan sencillos que tengan pocos programas (editor y compilador) Ej. PascalGUI o tan complejos que tengan todas las herramientas Ej. RAD Studio o Visual Studio.

Descripción gráfica de Programa, módulos y librerías.



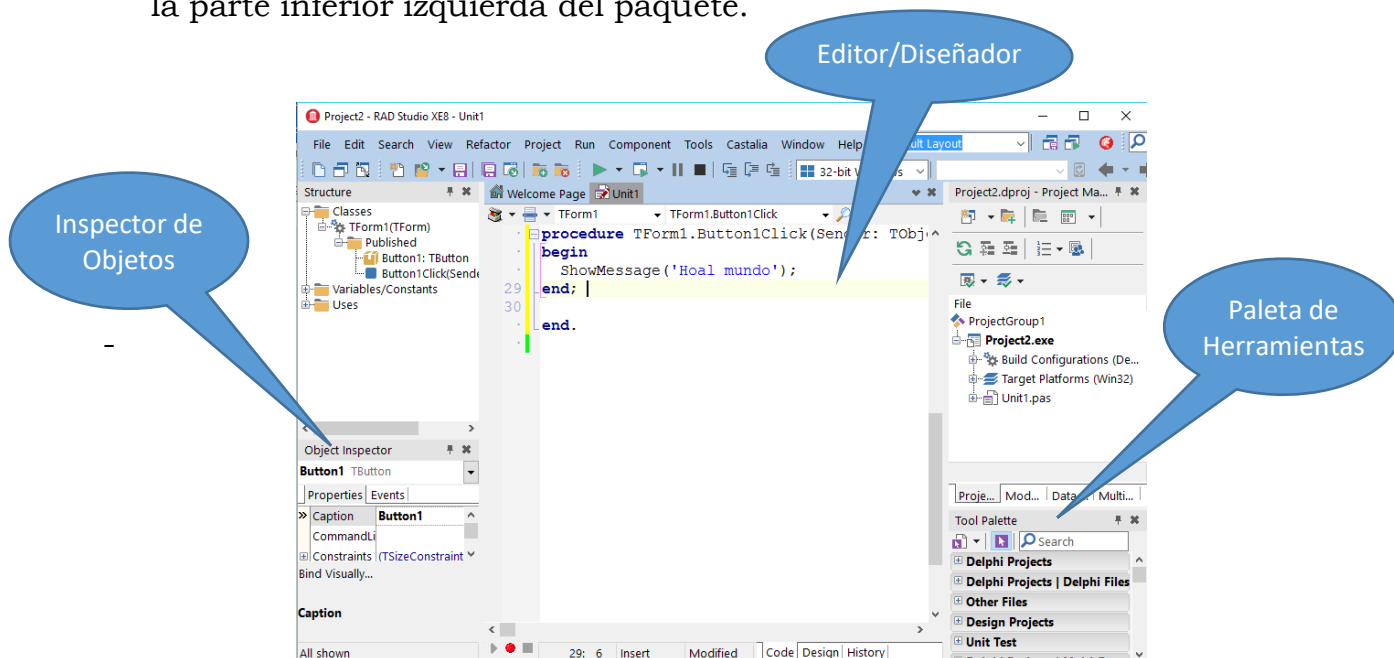
Descripción gráfica de Paquete



2.2 Paquete de Desarrollo RAD Studio

Este paquete está conformado por varios programas denominados herramientas del paquete, los cuales son:

- **Editor.** Es la herramienta del paquete que permite la escritura del código fuente por parte del desarrollador.
- **Compilador.** Es la herramienta que permite compilar el código fuente para generar el código objeto de la aplicación en desarrollo. Se puede acceder a esta herramienta pulsando las teclas **^F9** (control F9)
- **Depurador.** Es la herramienta que nos permite hacer un seguimiento a la ejecución de las instrucciones de un programa con la finalidad de detectar errores. Se puede usar esta herramienta colocando marcadores en el código y/o pulsando las teclas **F4, F7, F8**.
- **Paleta de Herramientas (Tool Palette).** Esta paleta contiene los componentes con los que se podrá armar la aplicación a desarrollar. Los componentes están agrupados por categorías, siendo las principales: Estándar, Additional, Dialogs. Aparece en la parte inferior derecha del paquete.
- **Inspector de Objetos (Object Inspector).** Permite editar los atributos de cada componente colocado en una aplicación en desarrollo. También permite editar los procesos vinculados a los eventos a los cuales reaccionará cada componente colocado en la aplicación en desarrollo. Se encuentra ubicado en la parte inferior izquierda del paquete.





Laboratorio 02. Estructura de una aplicación en Delphi

En esta experiencia se veremos la estructura de una aplicación básica en Delphi

1. Objetivos.

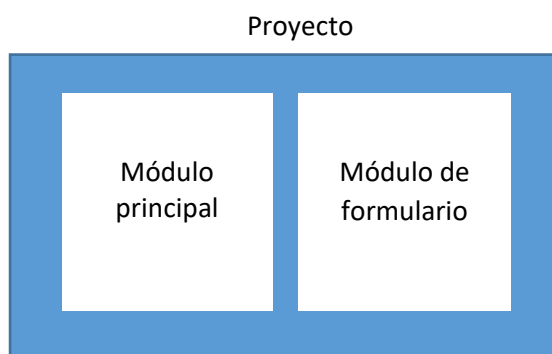
Conocer la estructura en Delphi de una aplicación básica para Windows

Conocer la estructura en Delphi de una aplicación básica para el modo consola

2. Desarrollo.

2.1 Aplicación para Windows. Una aplicación básica para Windows consta de las siguientes partes:

- **Módulo de formulario.** Este módulo se almacena en dos archivos:
 - o **.dfm** Contiene el código de configuración de los componentes del formulario de la aplicación. Éste código es generado por el paquete.
 - o **.pas** Contiene el código fuente (en Delphi) de los procesos implementados por el programador para el formulario.
- **Módulo principal.** Éste módulo se almacena en un archivo que contiene el código fuente del módulo principal que enlaza al formulario. Este archivo tiene extensión **.dpr**
- **Proyecto.** Se almacena en un archivo que tiene el mismo nombre del módulo principal pero con extensión **.dproj** , en él se almacena toda la configuración del proyecto Delphi.



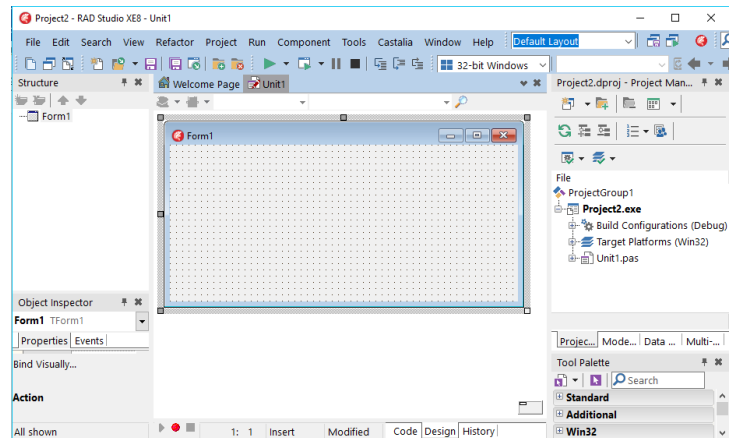
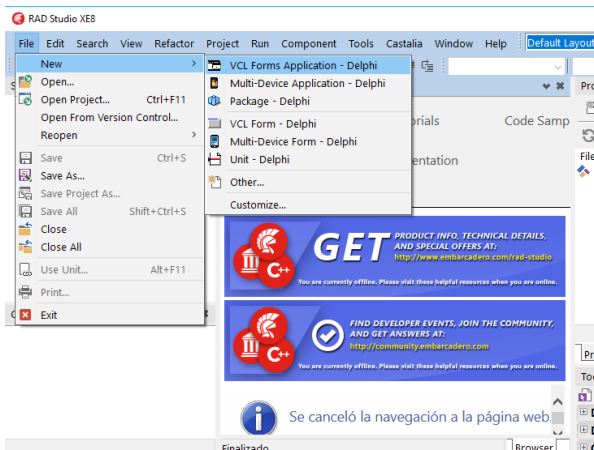
Creando una aplicación para Windows

1° En el Menú Principal seleccionamos **File**

2° Luego seleccionamos la opción **New**

3° Luego seleccionamos la opción **VCL Forms Application - Delphi**

4° A partir de aquí podremos “armar” nuestra aplicación, colocando componentes, programando los eventos y configurando los componentes.



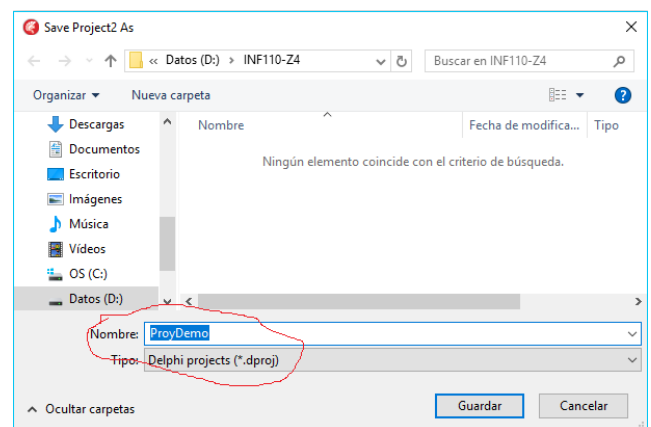
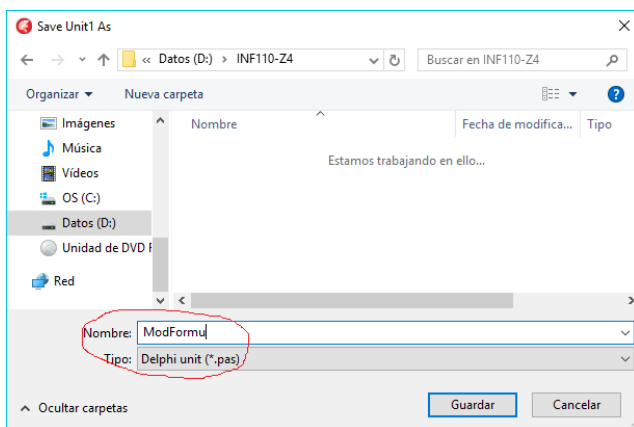
Guardando una aplicación para Windows

1° En el Menú Principal seleccionamos **File**

2° Luego seleccionamos la opción **Save Project As**

3° En este punto el paquete nos pedirá 2 nombres

- El primero será el nombre para el módulo del formulario (Ej. ModFormu) lo que generará dos archivos con el mismo nombre. Uno con la extensión .dfm y el otro con la extensión .pas
- El segundo será el nombre para el proyecto (Ej. ProyDemo) lo que generará dos archivos con el mismo nombre. Uno con la extensión .dpr y otro con la extensión .dproj

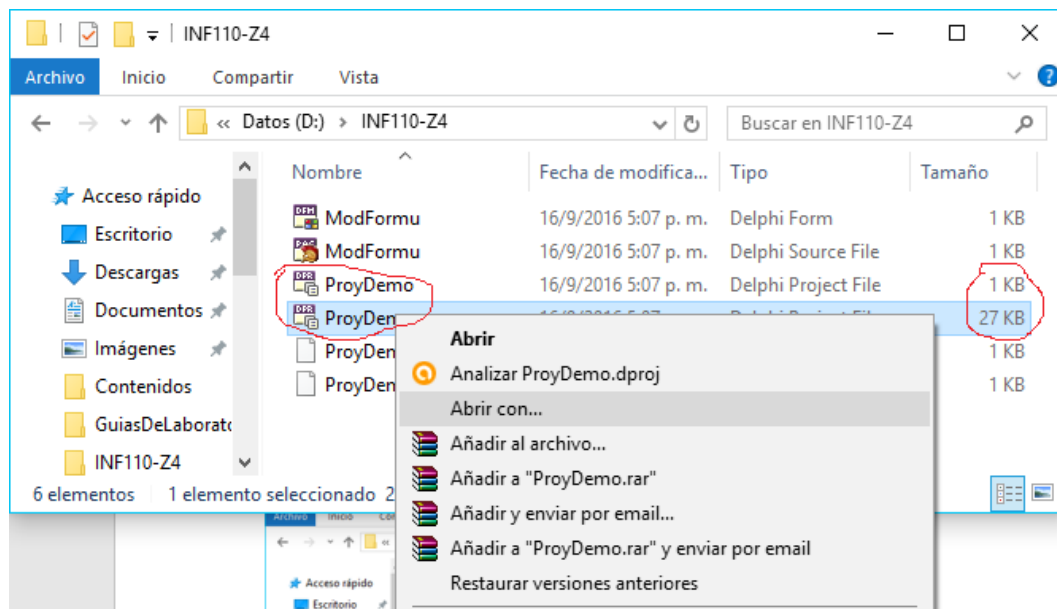


4° Una vez guardado el proyecto podemos continuar trabajando en él o cerrarlo. Para cerrarlo solo Hay que seleccionar del menú principal la opción **File**, luego **Close All**

Verificando el contenido de los archivos guardados

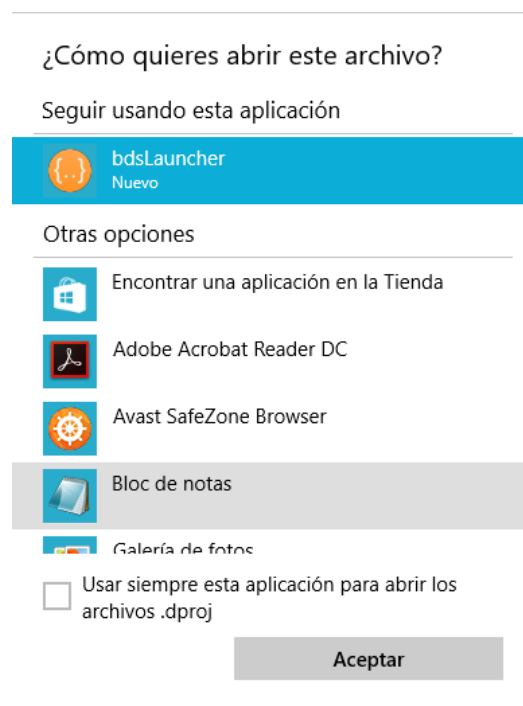
Como se mencionó en la primera parte, dada la estructura de una aplicación Delphi para Windows, se generarán 4 archivos (dos del formulario y dos del proyecto). Para verificar su contenido podremos abrirlo con el Bloc de Notas siguiendo los siguientes pasos:

1° Utilizando el **explorador de archivos** buscaremos el archivo del proyecto. Como hay dos archivos con el mismo nombre seleccionaremos el que ocupa más espacio en disco.



2° Dando clic con el botón derecho del mouse podremos seleccionar la opción **Abrir con**

3° En el menú contextual que aparecerá se debe seleccionar la opción **otras opciones o más aplicaciones**. Luego se debe desmarcar la opción **usar siempre esta aplicación para abrir los archivos .dproj**. Luego se debe seleccionar la aplicación Bloc de Notas.



Siguiendo los pasos anteriormente indicados podremos abrir los 4 archivos, los que se verán mas o menos como en las siguientes figuras.



Guía de Laboratorios de INF110 (Introducción a la Informática)

```
ProyDemo: Bloc de notas
Archivo Edición Formato Ver Ayuda
<Project xmlns="http://schemas.microsoft.com/developer/ms
<PropertyGroup>
  <ProjectGuid>{AB0D0C79-ECD6-4153-9EF8-3C0C5C28814}
  <ProjectVersion>17.2</ProjectVersion>
  <FrameworkType>VCL</FrameworkType>
  <MainSource>ProyDemo.dpr</MainSource>
  <Base>True</Base>
  <Config Condition="'$(Config)'==''">Debug</Config>
  <Platform Condition="'$(Platform)'==''">Win32</Pl
  <TargetedPlatforms>1</TargetedPlatforms>
  <AppType>Application</AppType>
</PropertyGroup>
<PropertyGroup Condition="'$(Config)'=='Base' or '$(B
  <Base>true</Base>
</PropertyGroup>
<PropertyGroup Condition="'$(Platform)'=='Win32' and
  <Base_Win32>true</Base_Win32>
  <CfgParent>Base</CfgParent>
  <Base>true</Base>
</PropertyGroup>
<PropertyGroup Condition="'$(Platform)'=='Win64' and
  <Base_Win64>true</Base_Win64>
  <CfgParent>Base</CfgParent>
  <Base>true</Base>
</PropertyGroup>
<PropertyGroup Condition="'$(Config)'=='Debug' or '$(
  <Cfg_1>true</Cfg_1>
  <CfgParent>Base</CfgParent>
```

```
ProyDemo: Bloc de notas
Archivo Edición Formato Ver Ayuda
program ProyDemo;

uses
  Vcl.Forms,
  ModFormu in 'ModFormu.pas' {Form1};

{$R *.res}

begin
  Application.Initialize;
  Application.MainFormOnTaskbar := True;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.
```

```
ModFormu: Bloc de notas
Archivo Edición Formato Ver Ayuda
unit ModFormu;

interface

uses
  Winapi.Windows, Winapi.Messages, System.SysUtils,
  System.Variants, System.Classes, Vcl.Graphics,
  Vcl.Controls, Vcl.Forms, Vcl.Dialogs,
  Vcl.StdCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

end.
```

```
ModFormu: Bloc de notas
Archivo Edición Formato Ver Ayuda
object Form1: TForm1
  Left = 0
  Top = 0
  Caption = 'Form1'
  ClientHeight = 201
  ClientWidth = 447
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'Tahoma'
  Font.Style = []
  OldCreateOrder = False
  PixelsPerInch = 96
  TextHeight = 13
  object Button1: TButton
    Left = 176
    Top = 72
    Width = 75
    Height = 25
    Caption = 'Button1'
    TabOrder = 0
  end
end
```

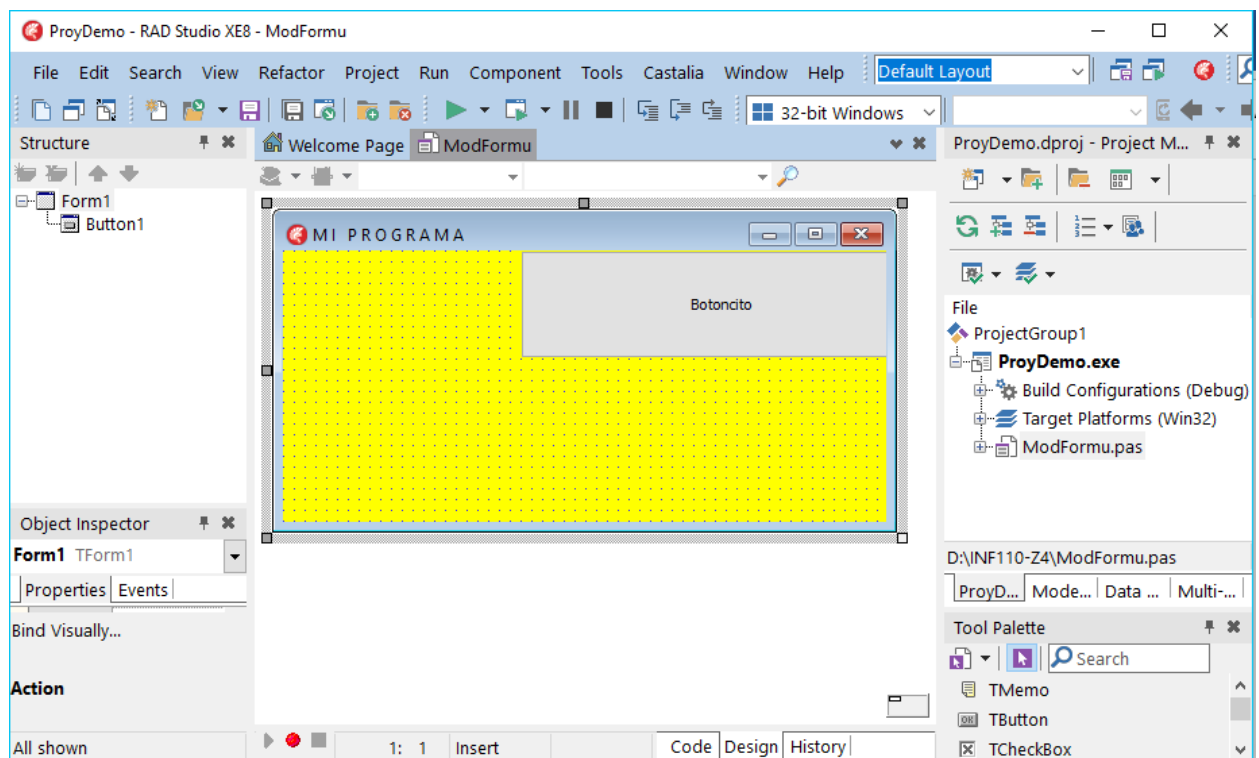
Experimento.

Como se mencionó en el anterior punto, al crear un proyecto Delphi se generan 4 archivos texto, uno de los cuales contiene el código de configuración del formulario (el archivo **.dfm**). A modo de experimento modificaremos el archivo **.dfm**, para ellos lo abriremos con el Bloc de Notas (como se indicó en el punto anterior) y editaremos algunas partes del archivo quedando como se muestra en el gráfico siguiente. Luego se debe guardar y cerrar el Bloc de Notas.


```
ModFormu: Bloc de notas
Archivo Edición Formato Ver Ayuda

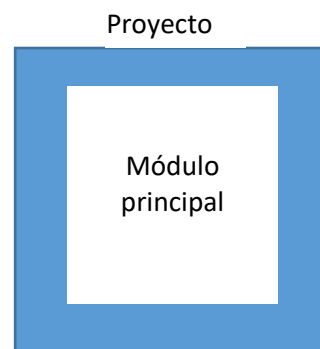
object Form1: TForm1
  Left = 0
  Top = 0
  Caption = 'M I  P R O G R A M A'
  ClientHeight = 201
  ClientWidth = 447
  Color = clYellow
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'Tahoma'
  Font.Style = []
  OldCreateOrder = False
  PixelsPerInch = 96
  TextHeight = 13
  object Button1: TButton
    Left = 176
    Top = 0
    Width = 300
    Height = 80
    Caption = 'Botoncito'
    TabOrder = 0
  end
end
```

Luego de haber hecho las modificaciones indicadas abrir el proyecto con Delphi y tendremos el siguiente resultado.



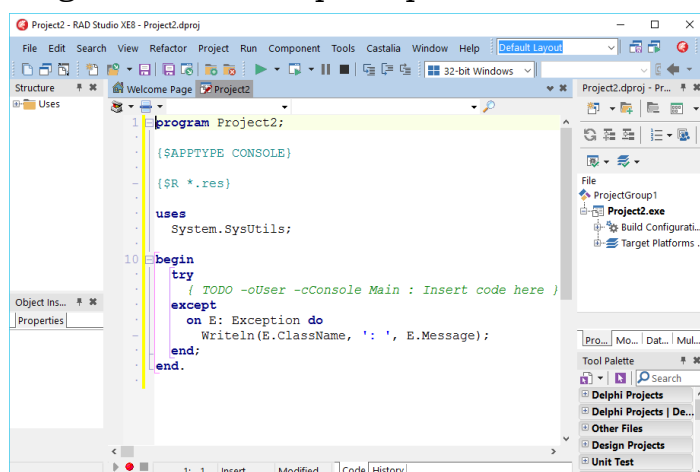
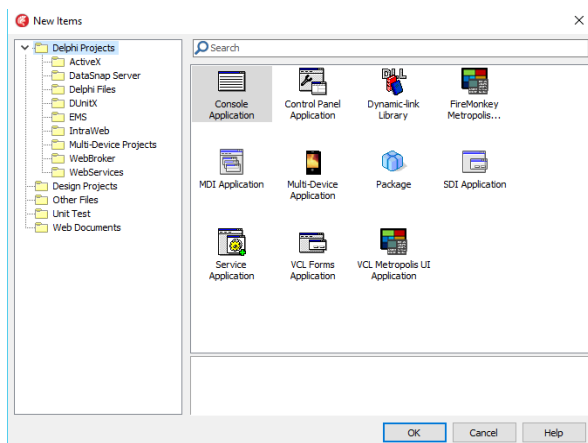
2.2 Aplicación para el modo consola. Una aplicación básica para el modo consola consta de las siguientes partes:

- **Módulo principal.** Este módulo contiene el código fuente del programa que se almacena en un archivo con extensión **.dpr**
- **Proyecto.** Se almacena en un archivo que tiene el mismo nombre del módulo principal pero con extensión **.dproj**, en él se almacena toda la configuración del proyecto Delphi.



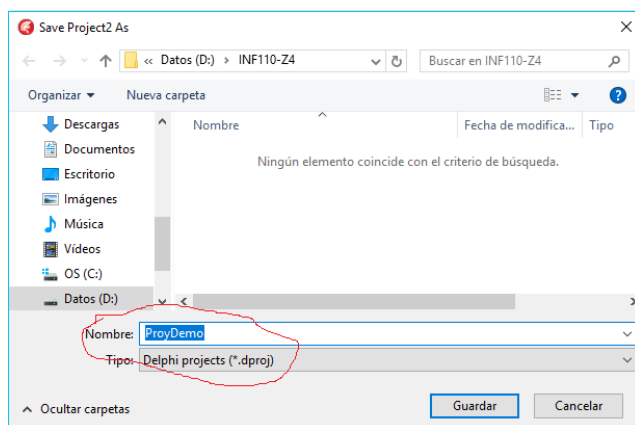
Creando una aplicación para el modo consola

- 1º En el Menú Principal seleccionamos **File**
- 2º Luego seleccionamos la opción **New**
- 3º Luego seleccionamos la opción **Other**
- 4º De la ventana que aparece seleccionar **Console Application**
- 4º A partir de aquí podremos escribir el código en el módulo principal



Guardando una aplicación para el modo consola

- 1º En el Menú Principal seleccionamos **File**
- 2º Luego seleccionamos la opción **Save Project As**
- 3º En este punto el paquete nos pedirá un solo nombre (Ej. ProgDemo) lo que generará dos archivos con el mismo nombre, uno con la extensión **.dpr** y otro con la extensión **.dproj**
- 4º Una vez guardado el proyecto podemos continuar trabajando en él o cerrarlo. Para cerrarlo solo Hay que seleccionar del menú principal la opción **File**, luego **Close All**



Verificando el contenido de los archivos guardados

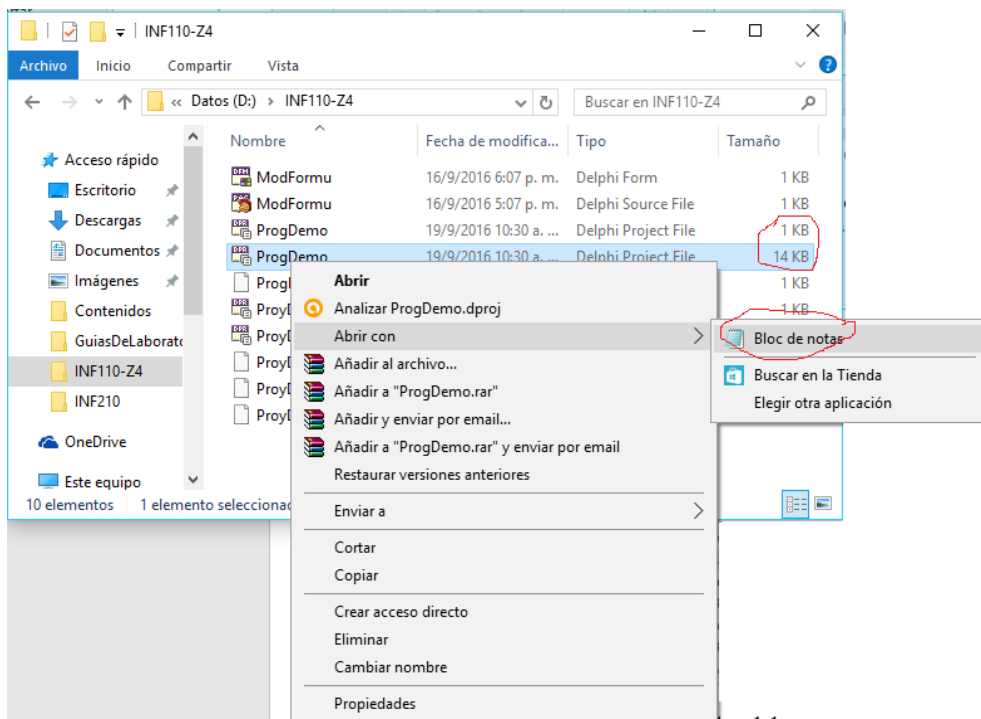
Como se mencionó en la primera parte, dada la estructura de una aplicación Delphi para el modo consola, se generarán 2 archivos. Para verificar su contenido podremos abrirlo con el Bloc de Notas siguiendo los siguientes pasos:

Guía de Laboratorios de INF110 (Introducción a la Informática)

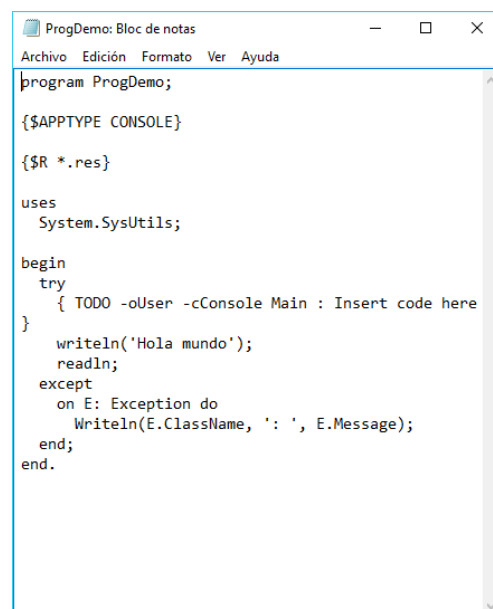
1º Utilizando el **explorador de archivos** buscaremos el archivo del proyecto. Como hay dos archivos con el mismo nombre seleccionaremos el que ocupa más espacio en disco.

2º Dando clic con el botón derecho del mouse podremos seleccionar la opción **Abrir con**

3º En el menú contextual que aparecerá se debe seleccionar la opción **otras opciones o más aplicaciones**. Luego se debe desmarcar la opción **usar siempre esta aplicación para abrir los archivos .dproj**. Luego se debe seleccionar la aplicación Bloc de Notas.



Siguiendo los pasos anteriormente indicados podremos abrir los 2 archivos, que se verán mas o menos como en las siguientes figuras.



Laboratorio 03. Componentes 1ra parte (Programación conducida por eventos)

En esta experiencia se veremos cómo se desarrolla aplicaciones para windows utilizando la programación conducida por eventos

1. Objetivos.

Conocer la secuencia de funcionamiento de la programación conducida por eventos.

Conocer el concepto de componentes

Armar una aplicación similar al **Bloc de Notas**

Armar una aplicación similar a la **Calculadora**

2. Desarrollo.

2.1 Programación conducida por eventos. La programación conducida por eventos es un paradigma de programación en el que la ejecución de los procesos está determinada por los sucesos que ocurran en el sistema, o que los usuarios provoquen, sucesos tales como: pulsar un botón del mouse, presionar una tecla, mover el mouse, tocar la pantalla, etc. En el gráfico siguiente se podrá ver la secuencia de acciones ocurridos por un evento.



1° El usuario provoca el evento (pulsando una tecla, moviendo el mouse, etc).

2° El dispositivo (teclado, mouse, etc) captura el evento y genera un código al respecto. El código es enviado a través de los buses hacia la tarjeta madre donde es almacenado en un buffer (memoria temporal).

3° El SO que está verificando constantemente los buffers, comprueba que ha ocurrido un evento (porque hay datos en algún buffer), extrae el código del buffer, luego comprueba si es o no un evento que él deba atender.

5° El código extraído es enviado a la aplicación activa en ese instante que esté en la zona donde se produjo el evento.

6° La aplicación recibe el código y la atiende si le corresponde, de lo contrario se la deriva al objeto en el cual se produjo el evento.

7° El objeto genera una **reacción** al evento (mediante código de programación).

2.2 Componente. Es un elemento de software ya compilado, que es reutilizable y tiene una interfaz bien definida. Se distribuyen en librerías y tienen poca o casi nada de dependencia con otros componentes. Pueden ser instalados en cualquier lenguaje de programación y utilizados también en cualquier lenguaje.

Los paquetes de desarrollo más importantes traen consigo mismo muchas librerías de componentes lo que facilita a los desarrolladores a realizar aplicaciones en menos tiempo. Por ejemplo, en RAD Studio xe8 se tienen las siguientes librerías de componentes:

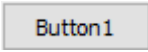




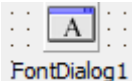
Guía de Laboratorios de INF110 (Introducción a la Informática)

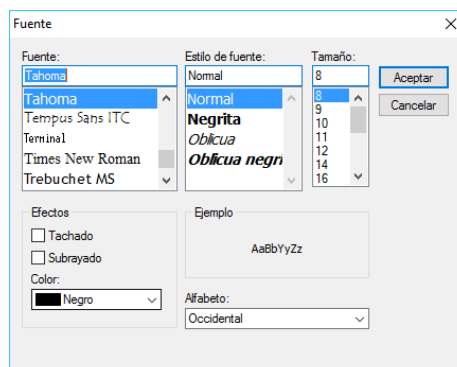
Los componentes pueden ser de dos tipos:

- **Visuales.** Es decir, el usuario los podrá ver tal como aparecen en la aplicación en desarrollo, por ejemplo: El **TButton**, el **TEdit**, Etc. En estos casos el programador podrá configurarlos desde el editor de objetos del paquete o a través de código. También existen componentes que no se visualizan en la aplicación en desarrollo, pero a través de código se pueden hacer visibles para que el usuario pueda interactuar con ellos, por ejemplo, los componentes de diálogo como ser: **TFontDialog**, **TOpenDialog**
- **No visuales.** Es decir, el usuario no podrá verlos como aparecen en la aplicación en desarrollo, sin embargo, el programador si podrá hacer que el usuario interactúe con él de alguna forma, por ejemplo, **TTimer**

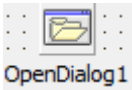


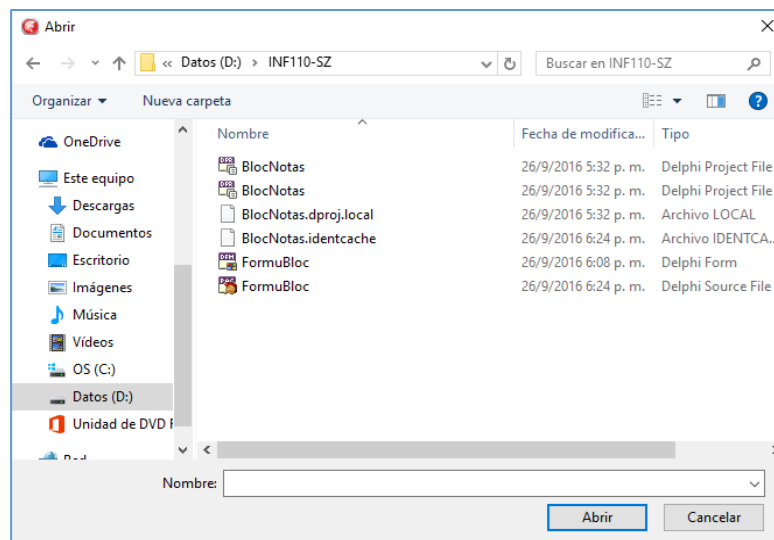
Los componentes más utilizados son los siguientes:

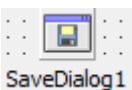
-  **TButton** Sirve para que el usuario active procesos, es decir, cuando él haga clic sobre éste objeto el programa realizará alguna acción (que deberá ser programada).
-  **TEdit** Sirve para que el usuario pueda escribir algo en él utilizando el teclado. Nosotros podremos capturar lo tecleado por el usuario en este componente.
-  **TMemo** Permite al usuario escribir muchas líneas de texto. Al igual que en el Edit nosotros podremos capturar lo tecleado por el usuario solo que será un poco más complejo el proceso. El contenido de este componente podrá ser almacenado en un archivo de texto.
-  **TLabel** Permite colocar etiquetas de texto en las aplicaciones, es decir, el texto que podrá ver el usuario, pero no puede modificarlo.
-  **TMainMenu** Permite colocar y configurar un menú principal para nuestra aplicación. También podremos programar las acciones a realizar de cada opción del menú.
-  **TFontDialog** Permite activar la ventana de diálogo para seleccionar la fuente de letra.

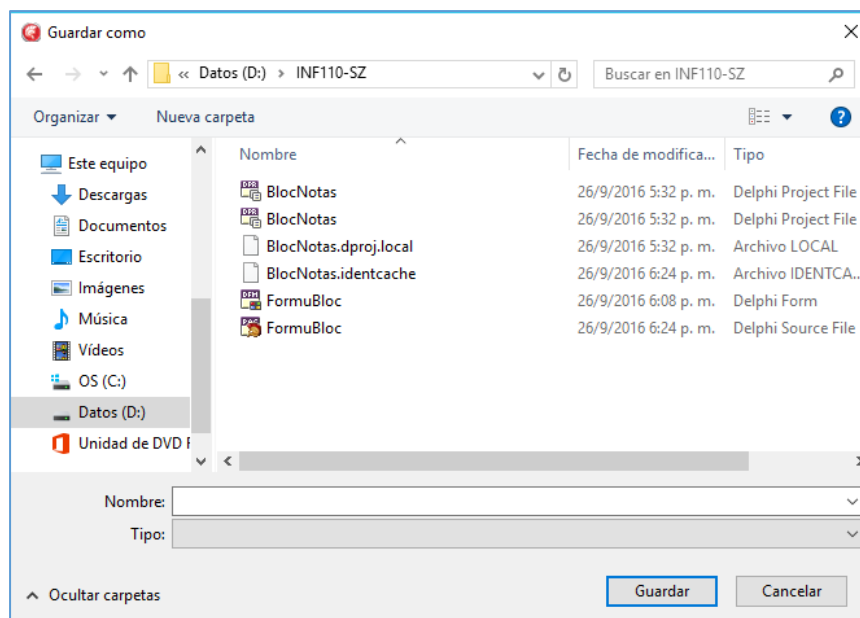


Guía de Laboratorios de INF110 (Introducción a la Informática)

-  **TOpenDialog** Permite activar la ventana de diálogo que le permitirá al usuario seleccionar la carpeta y el archivo a abrir (no abrirá ningún archivo) de allí se puede capturar la ruta hacia el archivo y el nombre del archivo.



-  **TSaveDialog** Permite activar la ventana de diálogo que le permitirá al usuario seleccionar la carpeta y el nombre de archivo con el que se puede guardar algún archivo (no guardará ningún archivo) de allí se puede capturar la ruta y el nombre de archivo.

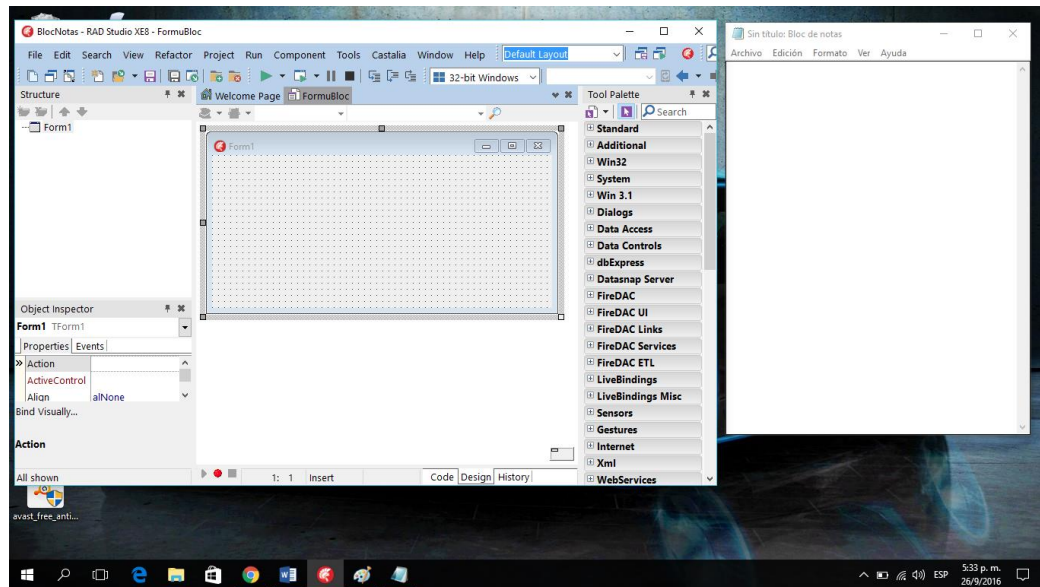


2.3 Experimento 1.

Como experiencia “construiremos” una aplicación similar al **Bloc de Notas**. En esta experiencia solo colocaremos los componentes que se requieren para la aplicación pues el objetivo es conocer los componentes. En el próximo laboratorio aprenderemos a configurar los componentes y en el siguiente aprenderemos a programar los eventos de los componentes.

Guía de Laboratorios de INF110 (Introducción a la Informática)

Primero lo primero, tal como vimos en el Lab02 crearemos una aplicación para Windows (VCL Forms Application – Delphi), la cual ya comienza con una ventana que en tiempo de desarrollo llamaremos **formulario**. Una vez creada la aplicación deberíamos guardarla en una carpeta nueva (por ejemplo, **BlocDeNotas**) y los archivos podrían llamarse por ejemplo **FormuBloc** (para el formulario) y **BlocNotas** para el proyecto. Sería bueno ejecutar la aplicación Bloc de Notas para tomarla como referencia.



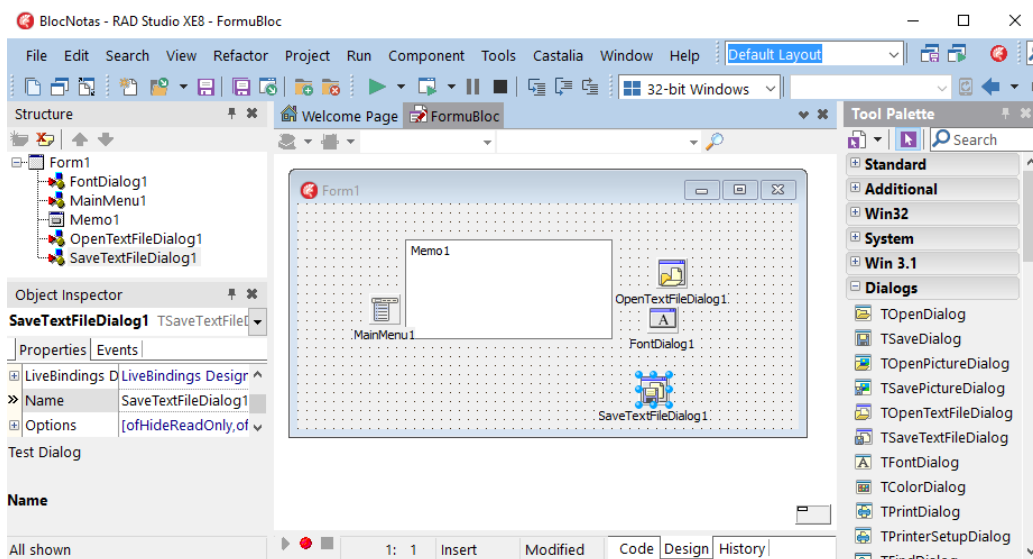
Ahora colocaremos los componentes realizando los siguientes pasos:

- **1ro** colocaremos el componente **MainMenu** (que será el menú principal de nuestra aplicación), para ello seleccionaremos la de la **ToolPalette** (que está en la esquina inferior derecha de Delphi) la opción **Standard**, lo que desplegará una sublista, de ella hacer clic en **TMainMenu**, luego hacer clic en cualquier parte del formulario, lo que dejará un ícono. Este ícono nos permitirá configurar el menú principal de nuestra aplicación (lo que haremos en el próximo laboratorio).
- **2do** Ahora colocaremos el componente **TMemo** (que será el editor de nuestra aplicación), para ello haremos un clic en la opción **TMemo** que está en la **ToolPalette** en la misma paleta Standard que el TmainMenu, luego haremos clic en el formulario.
- **3ro** Ahora colocaremos el componente **TFontDialog** (que permitirá a nuestro usuario cambiar el tipo de letra de nuestro editor). Este es otro componente no visual, que aparecerá al usuario cuando seleccione la opción Fuente que pondremos luego. Igual que en el caso anterior haremos un clic en el componente **TFontDialog** que está en la paleta **Dialogs**, luego haremos un clic en el formulario.
- **4to** Ahora colocaremos el componente **TOpenTextFileDialog** (que permitirá a nuestro usuario abrir un documento de texto). Este es otro componente no visual, que aparecerá al usuario cuando seleccione la opción Abrir que pondremos luego. Igual que en el caso anterior haremos un clic en el componente **TOpenTextFileDialog** que está en la paleta **Dialogs**, luego haremos un clic en el formulario.

Guía de Laboratorios de INF110 (Introducción a la Informática)

- **5to** Finalmente colocaremos el componente **TSaveTextFileDialog** (que permitirá a nuestro usuario guardar el documento en edición). Este es otro componente no visual, que aparecerá al usuario cuando seleccione la opción Guardar que pondremos luego. Igual que en el caso anterior haremos un clic en el componente **TSaveTextFileDialog** que está en la paleta **Dialogs**, luego haremos un clic en el formulario.

Ahora que ya hemos colocado todos los componentes deberíamos guardar la aplicación, la cual debería verse como sigue:

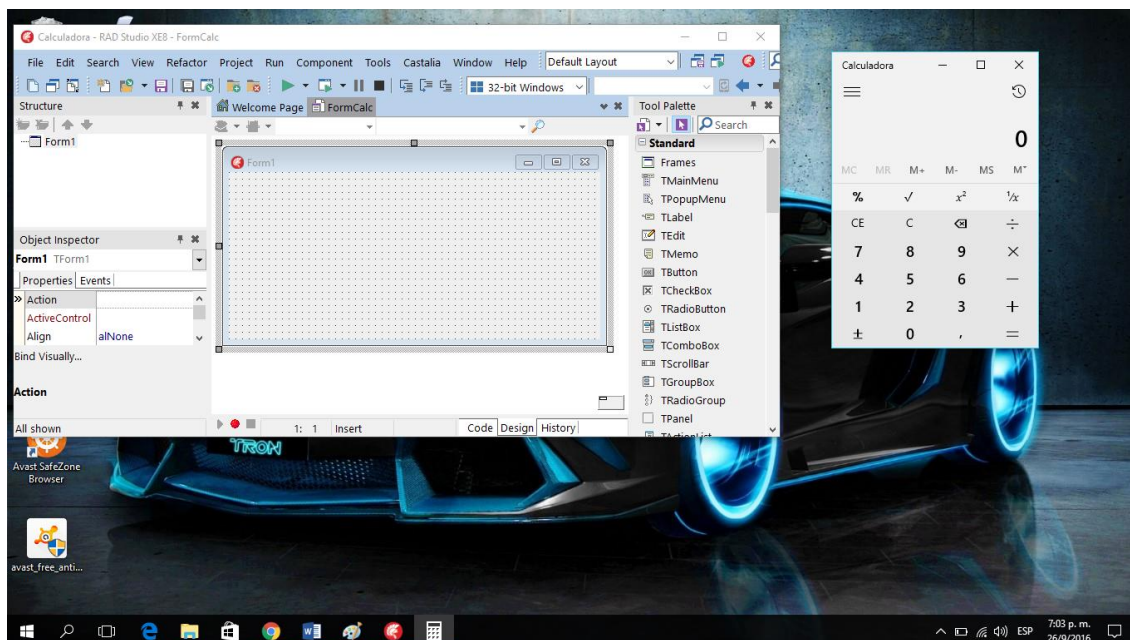


2.4 Experimento 2.

En esta segunda experiencia “construiremos” una aplicación similar a la **Calculadora**. En esta experiencia solo colocaremos los componentes que se requieren para la aplicación pues el objetivo es conocer los componentes. En el próximo laboratorio aprenderemos a configurar los componentes y en el siguiente aprenderemos a programar los eventos de los componentes.

Primero lo primero, tal como vimos en el Lab02 crearemos una aplicación para Windows (VCL Forms Application – Delphi), la cual ya comienza con una ventana que en tiempo de desarrollo llamaremos **formulario**. Una vez creada la aplicación deberíamos guardarla en una carpeta nueva (por ejemplo, **Calculadora**) y los archivos podrían llamarse por ejemplo **FormCalc** (para el formulario) y **Calculadora** para el proyecto. Sería bueno ejecutar la aplicación Calculadora de Windows para tomarla como referencia.

Guía de Laboratorios de INF110 (Introducción a la Informática)



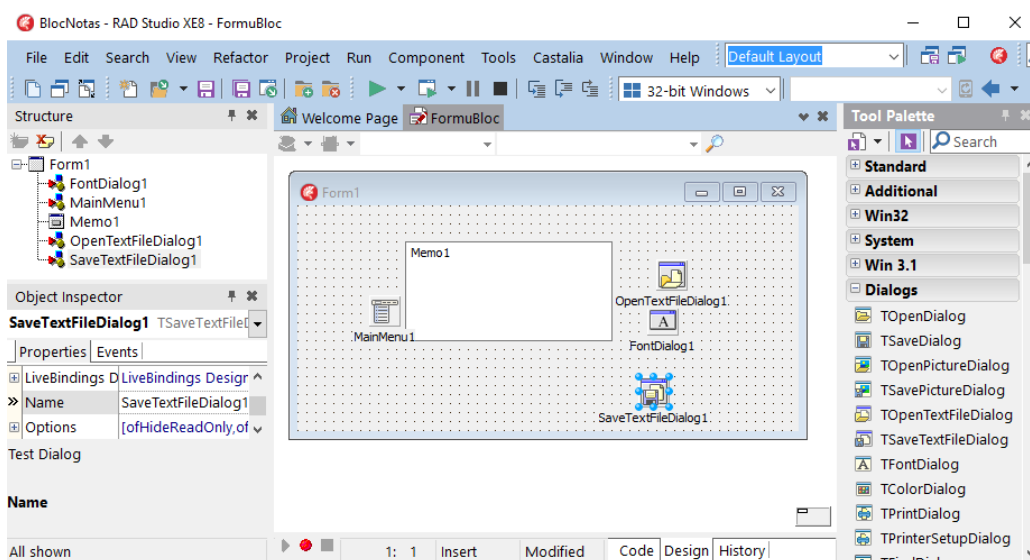
Ahora colocaremos los componentes realizando los siguientes pasos:

- **1ro** colocaremos el componente **Edit** (que será el espacio donde el usuario escribirá las cantidades), para ello seleccionaremos la de la **ToolPalette** la opción **Standard**, lo que desplegará una sublista, de ella hacer clic en **TEdit**, luego hacer clic en la parte superior de formulario. Luego con la ayuda del mouse podremos agrandarlo y ubicarlo como está en la aplicación de referencia.
- **2do** Ahora colocaremos los botones uno por uno (los botones serán las teclas de números y operaciones de la calculadora), para ello haremos un clic en la opción **TButton** que está en la **ToolPalette** en la misma paleta Standard que el TEdit, luego haremos clic en el formulario, luego lo acomodaremos usando el mouse en el lugar correspondiente, también podremos arreglar el tamaño de los botones siguiendo la muestra de la calculadora de Windows.

Ahora que ya hemos colocado todos los componentes deberíamos guardar la aplicación, la cual debería verse como sigue:



Guía de Laboratorios de INF110 (Introducción a la Informática)



siguientes partes:

- **Módulo de formulario.** Este módulo se almacena en dos archivos:
.dfm Contiene el código de configuración de los componentes del formulario de la aplicación. Éste