Introducción

Power Builder

Qué es Power Builder ?

PowerBuilder es un ambiente para desarrollar aplicaciones gráficas. Usando PowerBuilder, usted puede fácilmente desarrollar poderosas aplicaciones grafica que accesa a servidores de base de datos. PowerBuilder provee todas las herramientas que Ud. necesita para construir aplicaciones industriales , tales como , contabilidad, sistemas de manufactura, etc. PowerBuilder es un entorno de desarrollo comprensivo para construir aplicaciones cliente /servidor de alto desempeño para la familia Windows, que combina una interface gráfica intuitiva con un poderoso lenguaje de programación orientado a objetos.
Power Builder soporta multi-plataformas desarrolladas y desplegadas. Por ejemplo, Ud. puede desarrollar una aplicacion usando PowerBuilder bajo windows(Win'95 o Win NT) y desplegar la misma aplicacion -sin hacer cambios- sobre máquinas Win 3.11, Macintosh, o Unix.

Desarrolo en Internet: Power Builder incluye herramientas que le permiten construir aplicaciones basadas en Web y extender la existencia de su aplicacion al Internet. Es un front-end que puede interactuar con la mayoría de DBMS basados en ODBC

Acerca de los Pintores(Painters)

Se puede construir los componentes de una aplicación usando pintores, los cuales proveen una variedad de herramientas para construir objetos. Power Builder provee un pintor para cada tipo de objeto que se puede construir. Por ejemplo: se puede construir una ventana con el Pintor de Ventanas(Window painter).

Acerca de Eventos y Scripts

Las aplicaciones de Power Builder son manejadoras de eventos: los usuarios controlan el flujo de la aplicación . Cuando un usuario hace click en un botón, elige una opción de un menú, o ingresa datos en una caja de texto,un evento se dispara. Se encribe codigo(script) que especifica el proceso que deberia suceder cuando el evento acurre.
Por ejemplo, Buttons tiene un evento Clicked. Se escribe un script(código) para el evento clicked de Buttton que especifica que sucede cuando el usuarios hace click en el botón.
Se escribe Script usando PowerScript del lenguaje Power Builder.
Un **Script** consiste de comandos de PowerScript, funciones, y sentencias que realizan un proceso en respuesta a un evento.

Acerca de las Librerias

Se puede gravar objetos, tales como , ventanas y menúes, en Librerias de PowerBuilder(archivos .PBL). Cuando se corre una aplicación, PowerBuilder recupera los objetos de la libreria. PowerBuilder provee un pintor Library(Libreria) para manejar librerias.

Creando un ejecutable

Cuando se ha creado una aplicación completa, se puede crear un ejecutable de la aplicacion para dar a diferentes usuarios y lo utilicen.

Explicación de Front-End y Back-End

Un front-end es un constructor de interfaces, es una herramienta de programación donde se definen los formatos mediante los cuales se van a visualizar y manipular los datos. Un back-end es la herramienta que almacena los datos y los entrega al front-end para su manipulación

Explicación de Cliente/Servidor.

Cliente/Servidor es una organización de procesos, donde un proceso específico al que se le denomina servidor se dedica exclusivamente a atender los requerimientos que le envían, un grupo de procesos denominados clientes.

Objetos de PowerBuilder

· PowerBuilder es una herramienta orientada a objetos.
· Cada objeto tiene sus propios atributos y eventos

Objeto

Un objeto es cualquier entidad o cosa que se pueda representar o concebir mediante una serie de características que lo definan

Atributos

Un atributo es una característica que define al objeto.

Evento

Es una circunstancia a la cual se asocia una porción de código de programación, que se ejecuta cuando el evento se dispara. Ej: click del mouse, al abrir una ventana, al hacer doble-click, etc..
Cada evento tiene asociado un espacio en donde se puede programar, a este espacio se le conoce como script.

Los tipos de objetos más importantes son:

- Aplicación
- Ventana
- Menús
- DataWindows, ChildDatawindows
- Gráficos
- ListBox
- DropDownListBox
- Multilineedi
- CommandButton

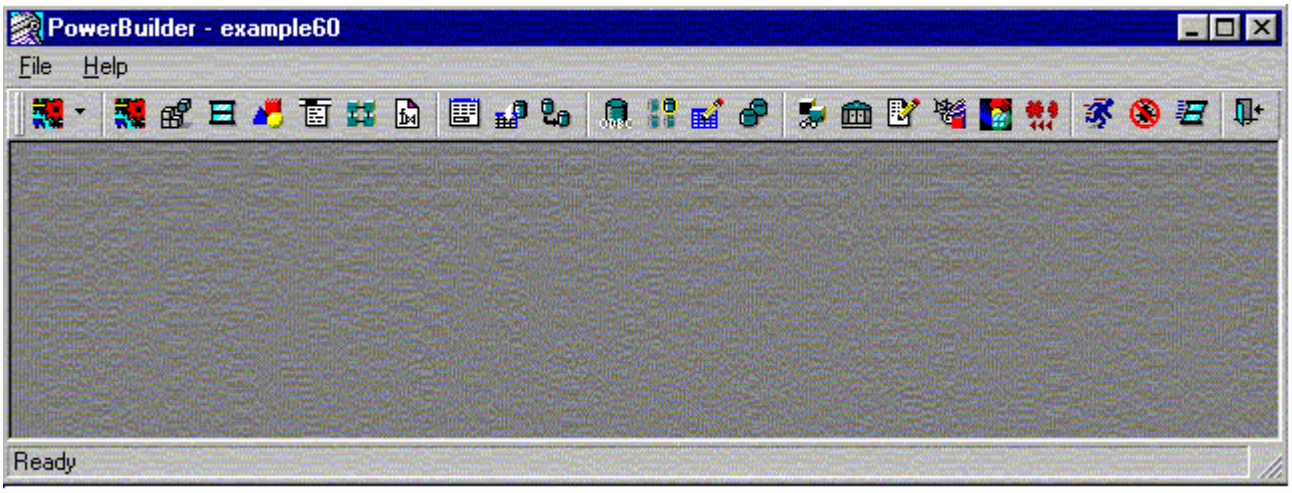
- PictureButton
- Editmask
- Checkbox
- Radiobutton
- Groupbox, etc...

Definición de SQLCA
SQL Communications Area(SQLCA) es un objeto transaccional. Un objeto transaccional es el área de comunicación entre el script (lugar donde se programa un evento) y la base de datos. PowerBuilder define este tipo de objeto para facilitar la comunicación con la base de datos desde el código de programación. Este objeto es accesado por default, cada vez que se utiliza una sentencia SQL dentro de un script.

Convenciones de Nomenclatura de Objetos
En PowerBuilder se acostumbra utilizar la primera letra del objeto con un underscore antes del nombre en sí de la instancia del objeto
Ej: ventana w_alumno, datawindow dw_calculo, cb_cerrar, etc.. w_ ventanas, dw_ datawindows, m_ menus, wf_ funciones de ventana, em_ editmask, cb_ commandbutton, dddw_ dropdowndatawindow, etc..

El Entorno de PowerBuilder

Cuando se empieza PowerBuilder, se abre una ventana que contiene un Menú y una barra(PowerBar). Se puede utilizar pintores de PowerBuilder para crear ventanas, menues, tables para una base de datos, y otros objetos que se necesite para la aplicación.



Acerca del PowerBar

La barra PowerBar se despliega cuando se inicia una sesión en PowerBuilder. PowerBar es un parte principal de control para construir aplicacioones en PowerBuilder. Desde esta

barra se puede abrir un pintor de PowerBuilder , correr una aplicación, personalizar el Power Builder de acuerdo a sus necesidades.

Si desea saber como utilizar cada uno de los pintores, haga click en la barra de herramientas que está a continuación.

Lista de los Pintores(Painters)

Pintor	Lo que realiza
Aplicacion	Especifica información acerca de la aplicación, tales como , su nombre, las librerias de Power Builder en las cuales los objetos de la aplicacion serán grabados
Project	Crea un ejecutable especificando los componentes que conforman la aplicación
Window	Construye una ventana que será usada en la aplicación
User Object(Objetos de usuario)	Construye objetos credaos por el usuario, que se puede grabar y usar repetidamente en una ventana.
Menu	Construye menues que las ventanas lo utilizarán
Structure(Estructura)	Define estructuras(grupo de variables) para usarlas en la aplicacion
Function(Función)	Construye funciones para desempeñar un proceso específico en la aplicación
DataWindow	Construye objetos inteligentes llamados objetos DataWindow que presentan información de una Base de datos.
Report(Reporte)	Construye vistas previas o reportes(un objetos DataWindow sin la capacidad de actualizar)
Run Report(ejecuta un reporte)	Realiza una vista previa de un reporte
Query(Consultas)	Realiza consultas graficas con sentecias SQL SELECT .
Data PipeLine(Tuberia de datos)	Transfiere datos desde una fuente de datos a otra
Configurar ODBC	Define una Base de Datos que usa ODBC
Databases Profiles(Perfiles de Base de datos)	Define y usa nombres puestos como parametros para conectarse a una base de datos en particular
Table(Tabla)	Crea tablas para una base de datos, altera tablas existentes, define claves primarias, relaciones entre tablas, indices

Database(Base de datos)	Sirve para mantener una base de datos, un usuario puede controlar el acceso a la base de datos y manipular los datos de una base de datos.
Database Administration(Administrador de base de datos)	Desempeña tareas de administracion de una base de datos,tales como, mantenimiento por parte del usuario y seguridades.
Browser	Permite visualizar acerca de los objetos del sistema y los objetos de una aplicación, tales como, propiedades, eventos, funciones ,variables globales.
Library (Libreria)	Crea y mantiene librerias de objetos de PowerBuilder.
File Editor(Editor de archivo)	Edita archivos de texto, tales como, fuentes, archivos de inicialización.
Run(Correr)	Ejecuta la aplicación actual que esta cargada en PowerBuilder
Debug	Corre una aplicacion paso a paso, permitiendo colocar puntos de quiebre.
Run Window(Corre una ventana)	Corre una ventana sinple en la aplicación.
System Options (Opciones de sistema)	Coloca preferencias del PowerBuilder, tales como, camino(path) de inicialzación perfiles preferidos, etc.
Help	Invoca a la ayuda en línea de PowerBuilder. Presionar F1. o elegir la Opción Help

Pintor Aplicación(Application Painter)

Vista Global de un Objeto Aplicación
Una **Aplicación** es una colección de ventanas de PowerBuilder que desempeñan actividades relacionadas.

El ***Objeto Aplicación*** es el punto de entrada dentro de la ventana que desempeña estas actividades. Cuando un usuario corre una aplicación, el script(código) que es escrito en los eventos son disparados en el objeto Aplicación.

Eventos en el objeto Aplicación

Evento	Lo que ocurre cuando se dispara
Close	Cuando el usuario cierra la aplicación.
ConnetionBegin	Cuando una aplicación Cliente intenta establecer una conexión a la aplicación servidor. Este evento se dispara solo cuando una aplicación servidor esta corriendo en un ambiente distribuido.
ConnectionEnd	Cuando la conexión de una aplicación cliente es terminada. Este evento se dispara solo cuando una aplicación servidor esta corriendo en un ambiente distribuido.
Idle	Cuando la Función Idle ha sido llamada en el script del objeto aplicación y se especifica el número de segundos que han transcurrido cuando el mouse o teclado no están en actividad.
Open	Cuando el usuario corre la aplicación.
SystemError	Cuando en tiempo de ejecución ocurren serios errores.

Creando una Nueva Aplicación

El Primer Paso para Construir una nueva aplicación en PowerBuilder es crear un objeto aplicación para la aplicación.

Pasos para crear una aplicación

1. Haga click en el pintor Aplicación de la barra de herramientas PowerBar y aparece un ambiente de trabajo del objeto Application.

2. Selecione del menú principal en la opción **File** la opción **New** y luego aparece una ventana en donde ingresa un nuevo nombre o elige uno ya existente para el archivo .PBL principal.

3. Luego aparece la ventana (Fig. 1) en donde se debe ingresar el nombre de la aplicación. Además puede ingresar un comentario de la aplicación (este esopcional).

4. Por último debe elegir la libreria en donde se va almacenar todos los objetos creados en la aplicación y presione OK.

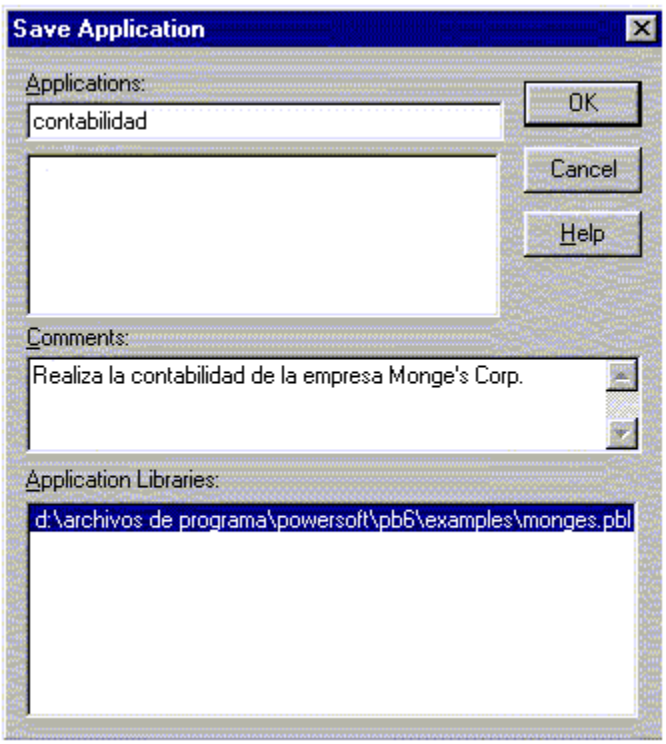


Fig. 1

Pintor Proyecto (Project Painter)

Acerca del Pintor Project:
Existen dos maneras básicas de empaquetar y crear una aplicación.

1. Como un archivo ejecutable independiente que contiene todos los objetos de la aplicación.
2. Como un archivo ejecutable y una o más librerias dinámicas que contienen objetos que son ligados en tiempo de ejecución.

El pintor Project permite que por una línea de flujo la generación de archivos ejecutables y librerias dinámicas. Cuando se quiere contruir una objeto project, se debe especificar los siguientes componentes de la aplicación:

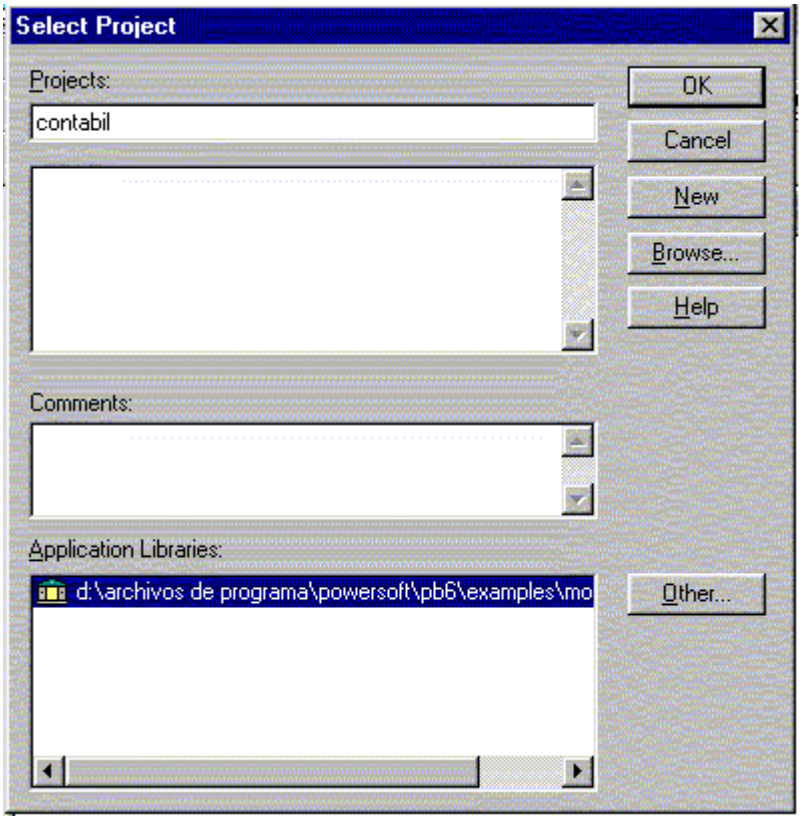
- Nombre del archivo ejecutable
- Cuales de las librerias quiere que se distribuyan como librerias dinámicas.
- Qué opciones de contrucción desea para usar en el proyecto.
- Qué opciones de generación de código desea usar.

Una vez que se ha definido el proyecto , ya se puede construir la aplicación con solo hacer click en el botón Build(contruir).

Contruyendo una Aplicación :

Pasos para crear un proyecto

1. Haga click en el pintor Project de la barra de herramientas PowerBar y aparece la pantalla siguiente:



2. Puede elejir un proyecto que exista ya o ingresar uno nuevo. Si ya existe elija el archivo que ya existe y presiona el botón *OK* y si es uno nuevo presiona el botón en *New*(nuevo).

3. Luego aparece un espacio de trabajo para crear el protecto (Fig. 2), en donde se ingresa el nombre del archivo ejecutable y varias de la opciones que se pueden ver en la figura(más adelante se explican estas opciones para construir una aplicación.)

4. Una vez que ha ingresado todos los datos, en el menú pricipal en la opción ***Design***(diseño) elija la opción ***Build Project***(contruir proyecto) para contruir un ejecutable de la aplicación.

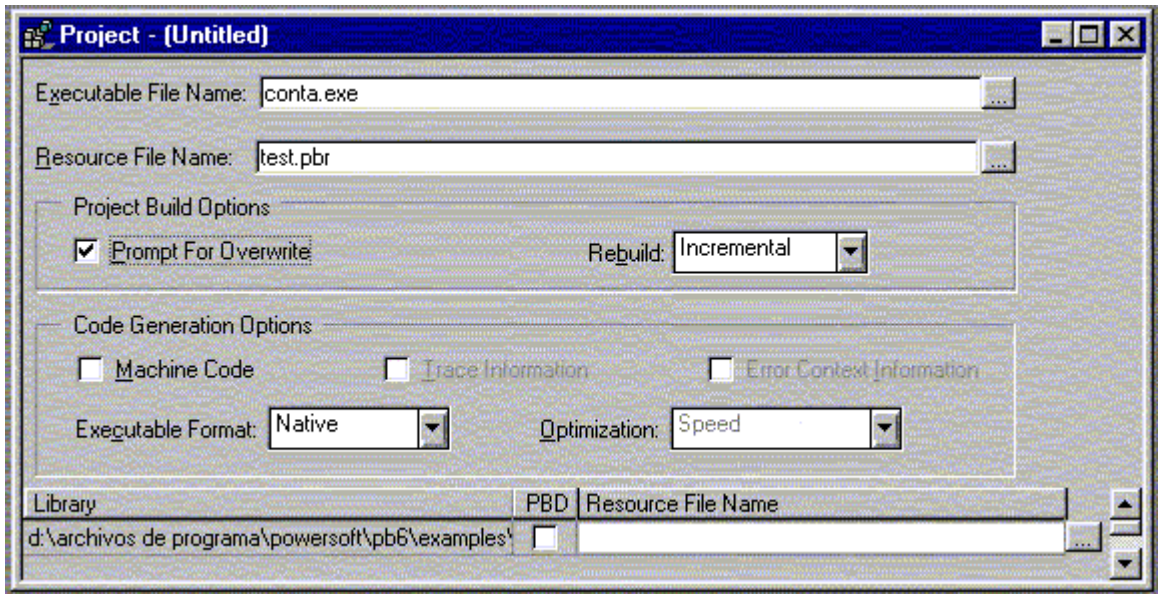


Fig. 2

Explicación de las Opciones para contruir una Aplicación:
Opciones del Pintor Project:
Executable File Name*(Nombre del archivo ejecutable)*
El nombre que se especifica para el ejecutable debe tener una extensión .EXE en la plataforma Windows.

Resource File Name*(Nombre del archivo de recursos)*
Se necesita especificar un archivo de recursos en PowerBuilder(.PBR File) para el archivo ejecutable si dinamicamente se hace referencia a recursos (tales como bitmaps y iconos) en algún script y si se quiere incluir recursos en el archivo ejecutable en lugar de tener que distribuir los recursos separadamente.

Project Build Options (Opciones para construir un proyecto)
Prompt Overwrite
PowerBuilder sobrescribe algunos archivos creados cuando esta construyendo una aplicación. Elija esta opción si desea sobreescribir. ***Rebuild***
Especifica una de las dos o Full o Incremental de una lista deplegable Rebuild indicando si se se quiere regenerar todos los onjetos en la librerias de la aplicación antes de hacerlo ejecutable ejecutable y librerias dinámicas, opción Full. Si elije Incremental, PowerBuilder regenera aquellos objetos referenciados por objetos que han cambiando desde la última vez que se contruyó la aplicación.

Code generation options (Opciones para generación de código)
MUchas de estas opciones están deshabilitadas if el compilador no es soportado por la plataforma. El codigo de compilación esta soportado sobre 32 bits en Windows, Unix y Mac.

Machine Code (código de maquina)
Selecione esta opción si Ud, quiere generar código compilado el lugar de Pcode.

Trace Information (Copia de la información)
Selecciona esta opción cuando desea crear un archivo de copia cuando corre el codigo compilado.

Error context Information (Contexto de información de errores)
PowerBuilder Despliega un contextode información, tales como un objeto, eventos, algún script para errores en tiempo de ejecución.

Excutable Options
Si elije Native es sobre 32 Bits o sino sobre 16 bits. ***Dynamic Library Options (Opciones de librerias dinámicas)***
Se puede reducir el tamaño del archivo ejecutable para distribuir algunos de los objetos requeridos en una libreria dinámica.

Pintor Ventana (Window Painter)

Vista Global de una Ventana

Las Formas Windows(ventanas) dan una interface entre el usuario y una aplicación de PowerBuilder. Las ventanas(windows) pueden desplegar información, pedir información a un usuario, y respoder a las acciones que realiza el usuario con el mouse y teclado.
Una ventana consiste de :

- **Propiedades** que definen la apariencia de la ventana y su comportamiento.
- **Eventos** Una ventana tiene eventos como otros objetos de PowerBuilder
- **Controles** ubicados dentro de la ventana. Controles como: CheckBoxes,CommandButton, etc.

Tipos de Ventanas

- Main
- PoPup
- Child
- Response
- Múltiple Document Interface **(MDI) Frame**
- MDI Frame con MicroHelp

Main Windows (Ventana Pricipales)

Las Main Windows son ventanas independientes que actuan de forma independiente con el resto de las ventanas.

Si usa una Main Window como un ancla para su aplicación. La primera ventana que la aplicación abre es una main window - a menos que Ud. haya contruido una aplicación con

Multiple Document Interface (MDI) , en este caso la primera ventana que se abre es una MDI Frame.

Si Ud. quiere que una ventana siempre esté a disposición del usuario, que puede ser desplegada en cualquier momento, en cualquier parte de la pantalla.

Popup Windows (Ventanas Popup)

Las Ventanas Popup son abiertas desde otra ventana, que en la mayoria de los casos llegan a ser padres de las ventanas popup.

Las ventanas popup son utilizadas a menudo como ventanas de soporte. Por ejemplo: si se tiene una ventana que contiene información principal, tales como una lista de películas. Se puede usar una ventana Popup para permitir al usuario vea en detalle los datos de una película en particular.
Utilizando open **Open(popupwindow,parentwindow)**, por ejmeplo:
open(w_popup,w_padre).

Child Windows (Ventana Hijo)

Las ventanas hijo son siempre abiertas desde el interior de una ventana Main o Popup, que llegan a ser padres de la ventana hijo(Child window). *Una ventana Hijo existe solo dentro de una ventana padre.* Las ventanas Hijo(Child window) no pueden tener menúes, y nunca se consideran como ventanas activas. Una ventana hijo(Child window) se cierra cuando se cierra la ventana que es padre.

Response Window (Ventana de respuesta)

Las ventanas de respuesta solicitan información desde el usuario. Estas ventanas siempre son abiertas desde el interior de otra ventana(padre). Por lo general, una ventana de respuesta es abierta luego que algún evento ha ocurrido en la ventana padre.

Las ventanas de respuesta son de modo aplicación(application modal). Esto es, cuando una ventana de respuesta es desplegada, por tanto esta ventana se hace activa(obtiene el enfoque) y las demas ventanas de la aplicación no son accesibles mientras el usuarios no responda a la ventana de respuesta(response window).
Suelen usarse como cajas de mensajes, para dar información cuando ocurre un error, cuando se ha realizado alguna tarea, como se muestra en la figura siguiente.

MDI Frame Windows (marco de interface de multiples documentos)

Una MDI Frame es un marco de ventana en el que se puede abrir multiples ventanas como documentos o Sheets(hojas) y moverse entre las hojas(sheets).
Hay dos tipos de ventanas MDI Frame:

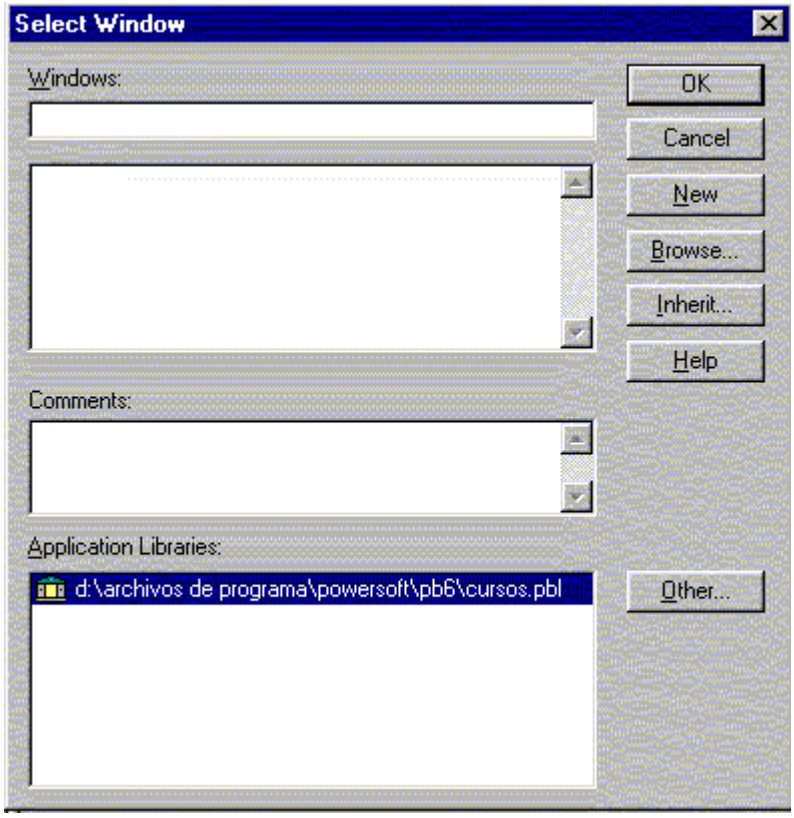
- MDI Frame
- MDI Frame with MicroHelp(con micro ayuda)

Construyendo una ventana Nueva

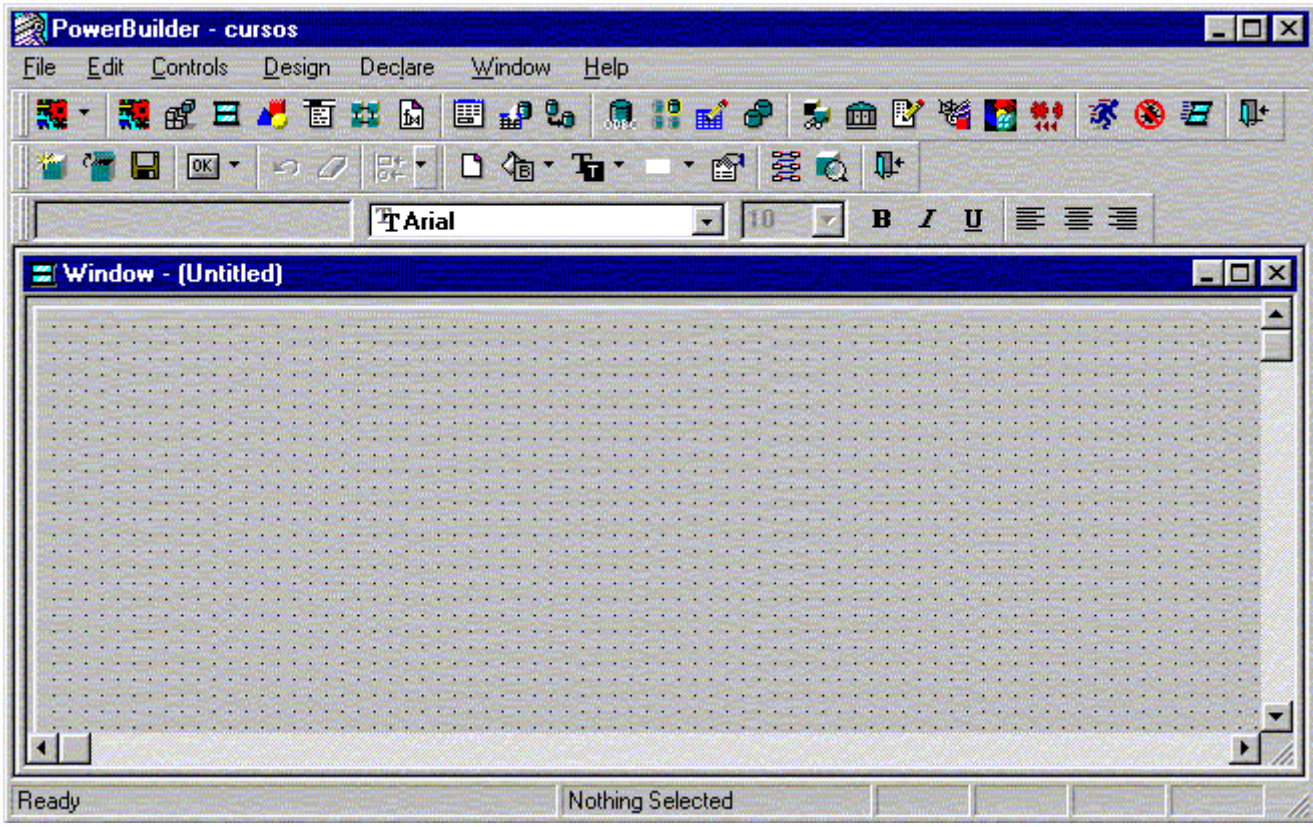
Esta sección describe cómo construir Windows del rasguño.Usted usará esta técnica para crear ventanas que no están basadas sobre ventanas existentes.

Abriendo el Pintor Window(ventana)

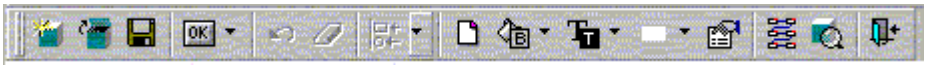
1. Haga click en el botón del pintor Window de la barra de herramientas PowerBar y aparece una ventana en donde puede cargar una ventana ya existente o crear una nueva.



2. Haga Click en el botón *New*(nuevo) para crear una ventana nueva y aparece un espacio de trabajo para crear nuestra nueva ventana.
   Ademas, el botón *OK* , sirve para cuando ya existe una ventana y queremos trabajar sobre ella o modificarla, entonces elegimos una ventana y presionamos *OK*.
   Existe también el botón *Inherit* , esta opción es cuando queremos crear una ventana nueva , pero heredamos todas las características de una ventana que ya existe.



3. Aparecen dos barra de herramientas la barra del pintor (PainterBar) que trabaja de la misma manera como en otros pintores.



4. El pintor Window tiene una Barra de estilos(StyleBar) que se usa para asignar propiedades al texto.

Definiendo las propiedades de la ventana

Cada ventana y control tiene un estilo que determina como apareceran al usuario.
El estilo de una venta comprende lo siguiente:

- Tipo
- Apariencia Básica
- Posicipon inicial sobre la pantalla
- Botones
- Puntero a la pantalla.

Para definir el estilo de la ventana debe elejir de diferentes hojas de la página de propiedades de la ventana, como se muestra en la figura siguiente, en donde cada tiempo va cambiando y muestra todas las hojas de propiedades:

Propiedad General

- Primero debe elegir el tipo de ventana de una lista.(Main Popup,MDI Frame, Child,Response)
- El título de la ventana .
- Una ventana Main o MDI Frame deben tener asociado un menú mientras que las ventanas Hijo o Response no pueden tener nunca un menú.
  Para asociar una ventana con un menú, debe hacer click en *Browser* y elegir un menú que previamente se debe haber creado.
- Elegir color de la ventana *Window Color*, una etiqueta en la opción *Tag*.
- *La Opción Control Menú* ,esta opción para la mayoría de tipos de ventanas, despliega el casillero de cerrar la ventana en la parte superior derecha de la ventana. En la ventana Response no tiene efecto esta opción.

Propiedad Position (posición)

- Elija la hoja *Position*, de la ventana propiedades.
- A la ventana se puede ubicarle en la posición que se quiera en la pantalla con solo haciendo click en la ventana y arrastrar con el mouse a la posición deseada , o también poniendo en los ejes X y Y la posición deseada, además se puede poner el ancho(width) y altura(height) de la ventana.
- Además existe las opciones Normal, Maximizada o Minimizada.

Propiedad Pointer (puntero)

- Elija la hoja *Pointer*, de la ventana propiedades.
- Aqui se puede elegir diferentes tipos de punteros del mouse,Ej: Flecha(arrow) etc, o se puede elegir otro archivo (.CUR) haciendo click en Browser.

Propiedad Icon (icono)

- Elija la hoja *Icon*, de la ventana propiedades.
- En esta opción puede elegir el tipo de ventana que desee, si es la ventana principal de la aplicación entonces el tipo debe ser application!,o si una ventana es tipo response entonces puede elegir el tipo información(Information!) , etc.
- Aqui se puede elegir diferentes tipos de iconos de ventanas ,Ej: application! etc, o se puede elegir otro archivo (.ICO) haciendo click en Browser.

Propiedad Scroll (desplazar)

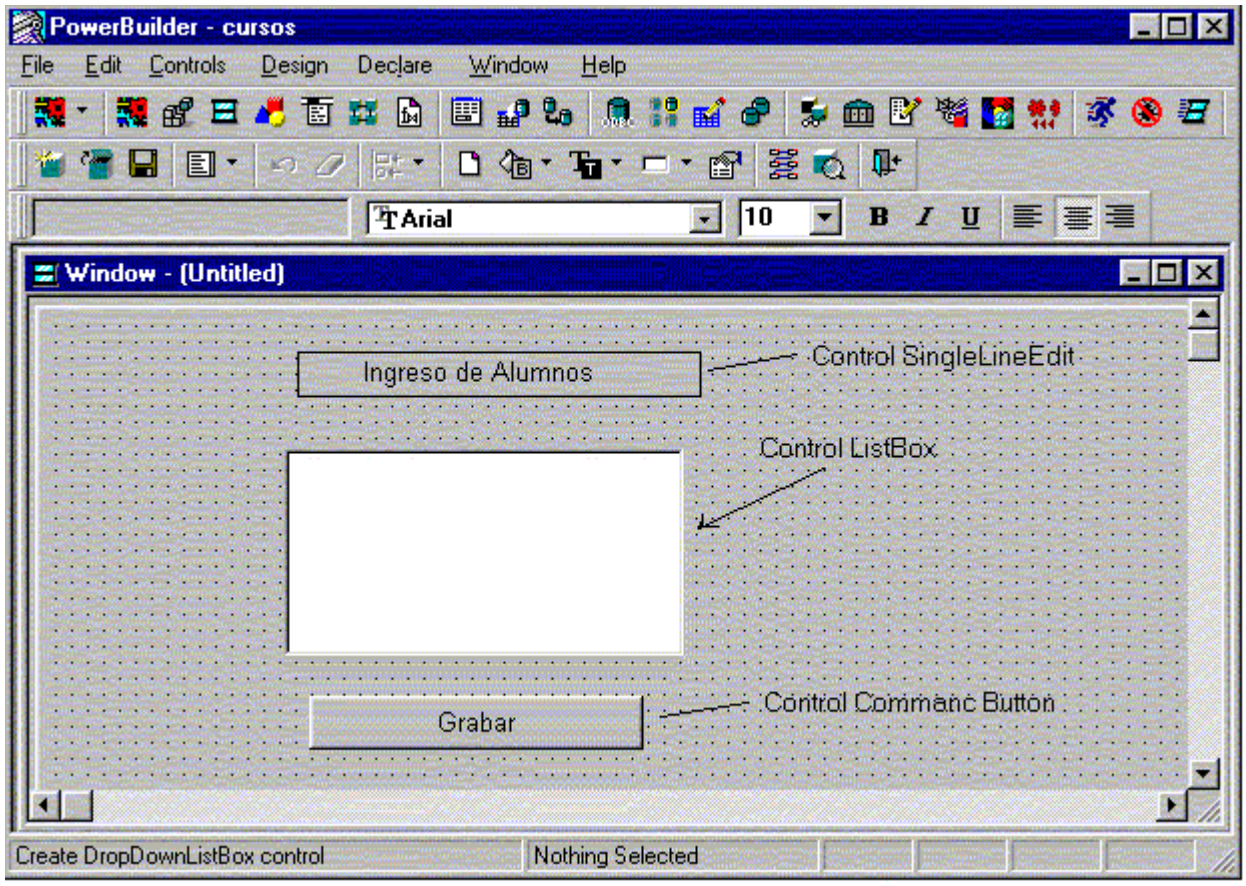
- Elija la hoja *Scroll*, de la ventana propiedades.
- Aquí puede poner el número en unidades de desplazamiento cuando se hace click en la barra de desplazamiento de la ventana, el default es O.
- Si se quiera tener Barra de desplazamiento horizontal elija en el casillero *HscrollBar* .
  Units Per Column: el número de unidades para desplazarse de Izquierda a derecha o viceversa, cuando el usuario hace click en la flechas izquierda o derecha de la barra horizontal de desplazamiento.
  Column per Page: El número de columnas que se desea desplazar cuando el usuario hace click sobre la barra horizontal.
- Si se quiera tener Barra de desplazamiento vertical elija en el casillero *VscrollBar*
  Units Per Line: el número de unidades para desplazarse de arriba a abajo o viceversa, cuando el usuario hace click en la flechas de arriba o abajo de la barra vertical de desplazamiento.
  Lines per Page: El número de líneas que se desea desplazar cuando el usuario hace click sobre la barra vertical.

Propiedad ToolBar (barra de herramientas)

- Elija la hoja *ToolBar*, de la ventana propiedades.
- En esta opción Ud. puede elegir en que lugar quiere ubicar la barra de herramientas PowerBuilder, ya sea en la parte superios, a la derecha, ect, o tambien puede dar una localización dando valores a las coordenadas X y Y.

Luego de poner las propiedades a la ventana haga click en ***OK***.

Agregando Controles
Cuando se construye una ventana, se puede colocar controles(tales como: CheckBos, CommadButton, etc.) dentro de la ventana para pedir y recibir información del usuario y presentar información para el usuario.
Despues de colocar un control en la ventana, se puede definir el estilo, moverlo, escribir codigo(script) para que el control responda de acuerdo a un evento.

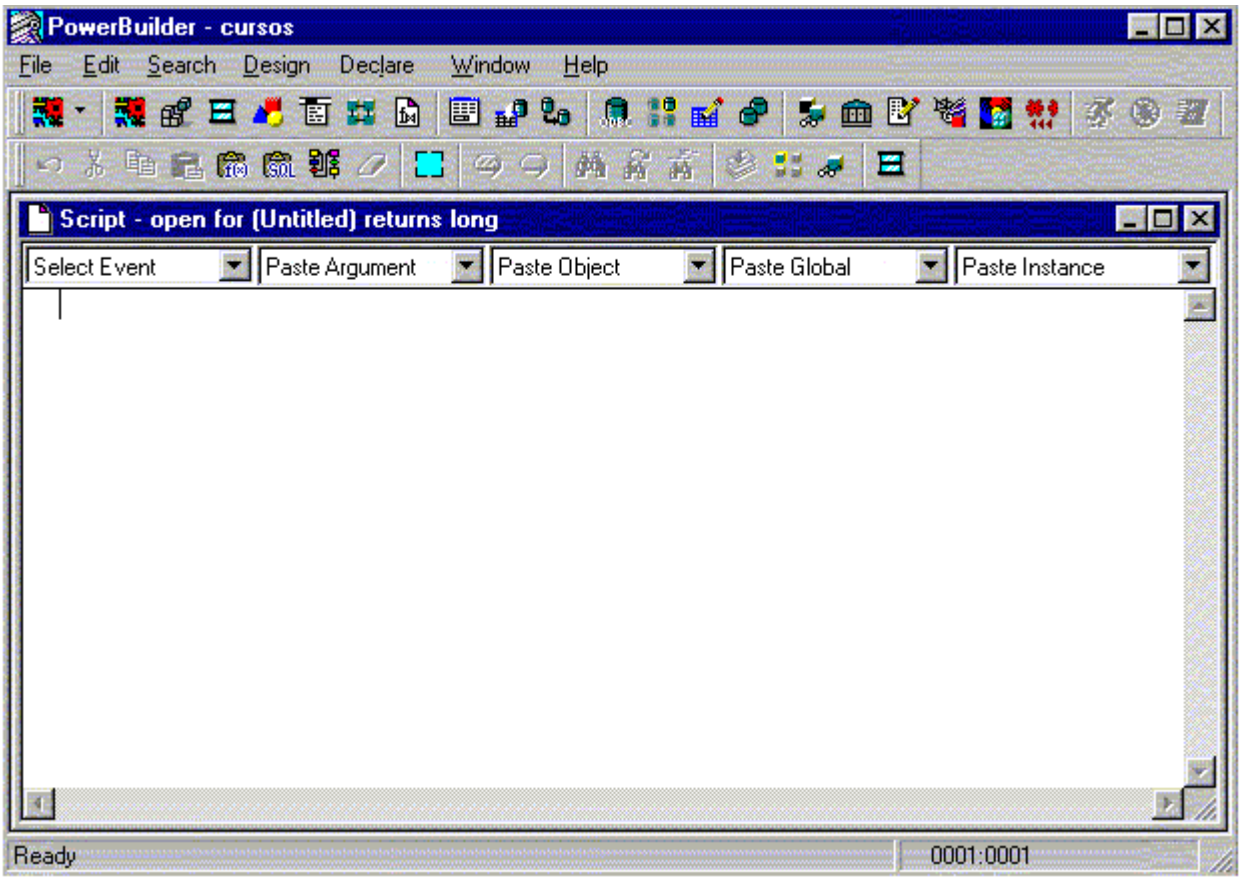


Escribiendo Código(Script) en una ventana

Se puede escribir código(script) en los eventos de una ventana y además en los controles que estan colocados dentro de la ventana.

Para escribir código(script) para un ventana o un control, coloque el mouse sobre la ventana (o el control) y haga click derecho con el mouse y elija la opción ***script*** o por el menú elija la opción ***Edit*** y luego seleccione ***script*** o haga click sobre el icono ***script*** en la barra de herramientas, entonces puede elejir el evento y escribir el código respectivo.

Por ejemplo: puede elegir el evento ***OPEN*** , este evento se dispara cuando se abre la ventana y ecribir el script correspondiente.



Además de elegir el evento en el cual vamos a poner codigo(script), podemos pegar argumentos, objetos ya existenes, variables globales y variables de instancia.

Acerca de los eventos para Ventanas
Las ventanas tiene varios eventos los más importantes son:

Evento Se dispara cuando	
Activate	cuando se abre la ventana, antes del evento open
Close	cuando se cierra la ventana
Open	cuando se abre la ventana

Definiendo sus propios eventos
Ud. también puede definir sus propios eventos,llamados Eventos de Usuario(User Events) para una ventana o control, estos eventos usan la función **TriggerEvent** para disparar un evento creado por un usuario.

Acerca de funciones para Ventanas y controles
Además puede también definir funciones para la ventana el cual puede llamar desde el codigo(script) de cualquier evento de la ventana y de los controles.En las funciones creadas se puede pasar parámetros.

Ud. puede definir su propio nivel de funciones para sus ventanas para facilitar la manipulación de sus ventanas.

Usando Herencia para construir una ventana
Cuando se construye una ventana que hereda todas las caracteristicas (estilo, evento, funciones, estructuras, variables, controles y scripts)de una ventana ya existente. Todo le que se tiene que hacer es modificar las caracteristicas heredadas de acuerdo a las situación actual.

Cómo heredar de una ventana ya existente
Ejemplo: Asumimos que la aplicación tiene una ventana w_padre_ingreso que tiene:

- Un título (Ventana padre para ingreso de datos)
- Un texto de dice: Ingreso de datos
- Un DataWindow en donde se ingresa los datos
- Y dos botones de comando para Grabar los datos o para cancelar.

A continuación mostramos la figura de la ventana descrita anteriormente:

Ahora asumimos que necesitamos construir otra ventana para llevar a cabo un proceso similar. Necesitamos heredar de la ventana padre ingreso , pero ahora para ingresar datos de un alumno especificamente.
Para construir esta ventana tenemos tres opciones:

1. Construir una nueva ventana con esas características de la forma como se explicó anteriormente.
2. Modificando la ventana existente(w_padre_ingreso), y luego grabándola con un nuevo nombre.
3. Usar herencia para construir la ventana que hereda todas las características de una ventana que ya existe(w_padre_ingreso), en otras palabras , construir una ventana decendiente de otra.

Usando las ventajas de la herencia (paso 3)
Usar herencia tiene algunas ventajas:

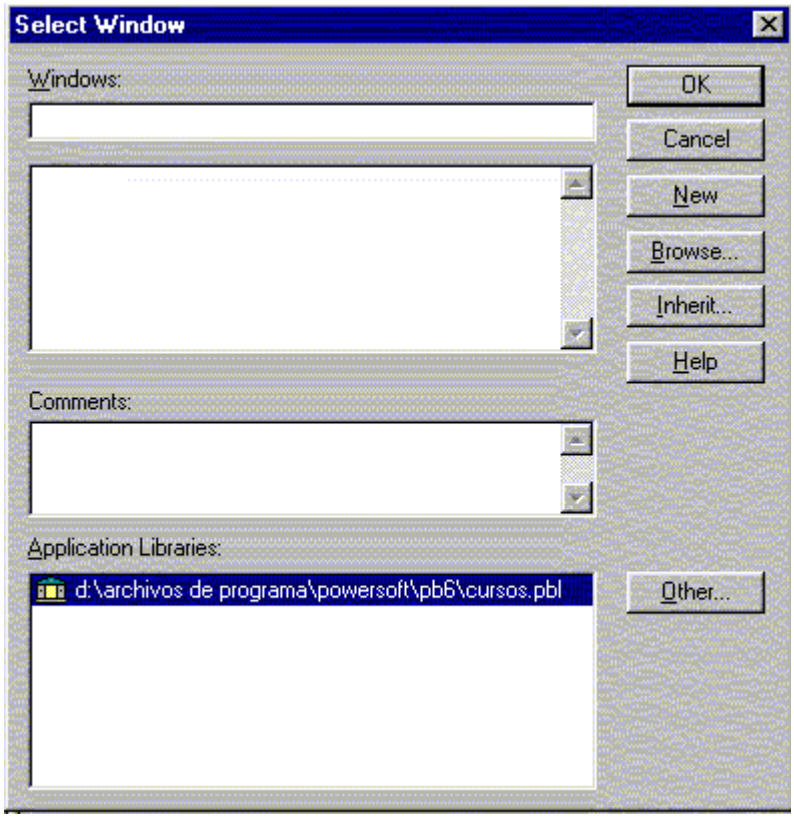
- Cuando Ud. realiza cambios en la ventana *padre*, los cambios son reflejados en todas las ventanas descendientes(hijas). Ud. no tiene que hacer manualmente cambios en los descendientes como lo deberia hacer en una copia(paso 2).
- Los descendientes heredan código script del padre, por tanto Ud. no tiene que reinmgresar el código y agregarlo a la ventana.
- Se obtiene consistencia en el código y en las ventanas de la aplicación.

Como usar herencia para construir una ventana

Esta sección describe cómo construir Windows del rasguño.Usted usará esta técnica para crear ventanas que están basadas sobre ventanas existentes.

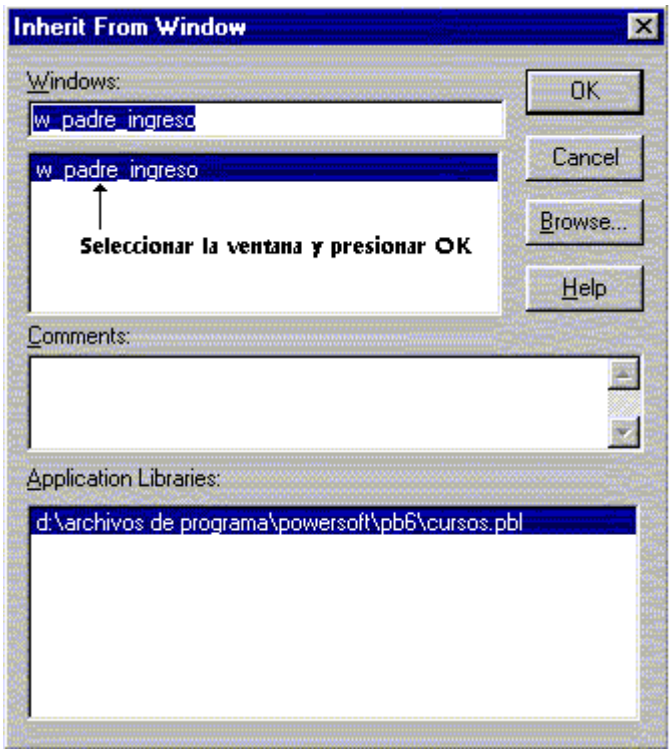
Abriendo el Pintor Window(ventana)

1. Haga click en el botón del pintor Window de la barra de herramientas PowerBar y aparece una ventana de donde podemos heredar una ventana

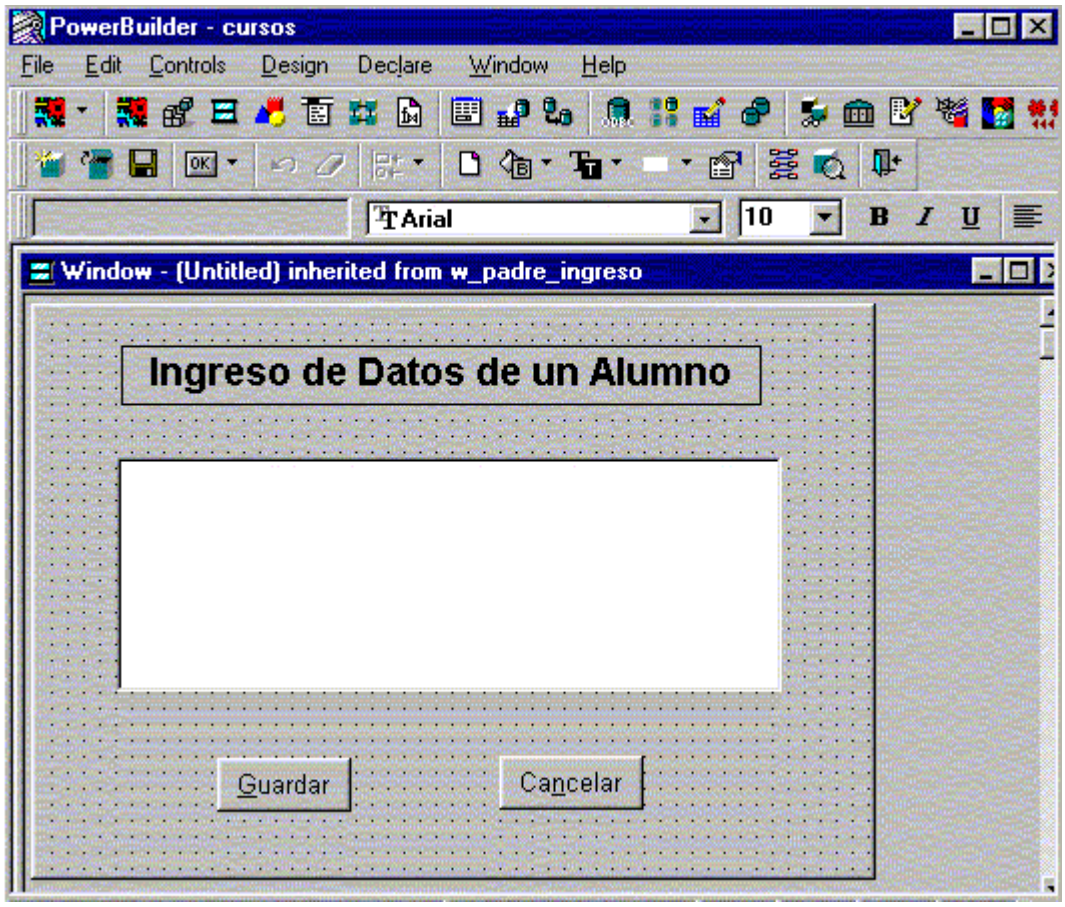


2. Haga Click en el botón *Inherit*(heredar) para heredar de una ventana que ya existe.

3. Luego aparece otra ventan, en donde debemos seleccionar la ventana de la cual vamos heredar, en este caso, w_padre_ingreso.



4. Luego que hemos seleccionado la ventana , nos aparece la nueva ventana heredada, en la cual podemos ponerle un nombre(w_ingreso_alumnos), y agregarle las características propias para esta nueva ventana. Podemos cambiar las características de la ventana. agregar controloles, contruir nuevos scripts pasra otros eventos, agregar codig´po a los scriptis existentes, referenciar a las funciones y eventos de la ventana padre, declarar nuevas variables, etc.



5. Si Ud. no necesita de algún control heredado, ud. puede harcerlo invisible a ese control en la ventana descendiente.

Pintor Objetos de Usuario (User Objets Painter)

Vista de un Objeto de Usuario Las aplicaciones a menudo tienen características comunes. Por ejemplo, Ud. podría muchas veces reutilizar esas características como las siguientes:

- Un Botón Cerrar, que realiza un cierto número de operaciones y luego cierra una ventana.
- Un ListBox que lista todos los departamentos.
- Un control DataWindow que lleva a cabo un chequeo del mismo tipo de error.
- Procesos que se realizan en varias partes del sistema.

Si Ud. está usando en la misma aplicación características repetidas, Ud. podría definir un Obejto de usuraio(User Objets): Ud. define el objeto de usuario una vez en el pintor Objeto de usuario(User Objets) y puede usar tantas veces como lo necesite.

Hay dos tipos de Objetos de usuario:

- Visuales
- Clase(Class)

Objetos de Usuarios Visuales
Existen tres Tipos de objetos de usuarios visuales:

- ***Standard:*** Un objeto de usuario visual standard hereda las características de un control standard de Power Builder. Usted Modifica las características y hace un control específico para su aplicación.
  Por ejemplo: Ud. asume frecuentemente usar un CommandButton llamado Cerrar, que cierra una ventana padre , el script sería : close(parent).

- ***Custom(creado por el usuario):*** Un objeto de usuario visual custom son objetos que tienen varios controles que funcionan como una unidad. Ud. puede pensar de un objeto de usuario visual custom como una ventana que es una unidad simple y es usada como un control.
  Digamos que Ud. usa un grupo de botones, cada uno de ellos realiza un proceso standard , si ud. construye un objeto de usuario Custom que contiene todos los botones, Ud. puede colocar todos estos botones en una ventana como una unidad , osea coloca el objeto usuario.

- ***External:*** Un objeto de usuario visual external contiene contoles de objetos en el sistema subyacente de las ventanas(Underlying windowing system) que fueron creadas fuera de PowerBuilder. Se puede usar una DLL(Libreria dinámica) hecha im Power Builder para crear un objeto de usuario externo.

Objetos de Usuario tipo Clase (Class)
Un Objeto de Usuario tipo Clase le permite que reutilice un conjunto de reglas de negocios u otros procesos que actúan como una unidad pero que *no son componentes visuales*. Por ejemplo, Ud. podría definir una clase que calcule comisiones de ventas o estadísticas sobre análisis de desempeño.
Hay dos tipos de Objetos de usuario tipo Clase:

- ***Standard:*** Este Objeto hereda las características de un objeto propio del PowerBuilder y que no es visual, tales como, un objeto Transaccional(transaction Object) o un objeto de error (Error Object).
  Un uso importante de este objeto, es cuando se usa herencia de un objeto Transaccional que hace llamadas a procedimientos en una base de datos remota desde la aplicación mismo.

- ***Custom:*** Estos objetos son diseñadas por Ud. mismo, propios, que encapsulan propiedades y funciones no visibles al usuario. No se derivan de Objetos de PowerBuilder. Ud. los define y los crea como unidades de procesamiento que tienen componentes no visuales.
  Por ejemplo: para calcular comisiones en una aplicación, Ud. puede definir un objetode usuario tipo Clase Custom que contiene propiedades y funciones creadas por el usuario que hace el proceso de cálculo de comisiones. Siempre que Ud.

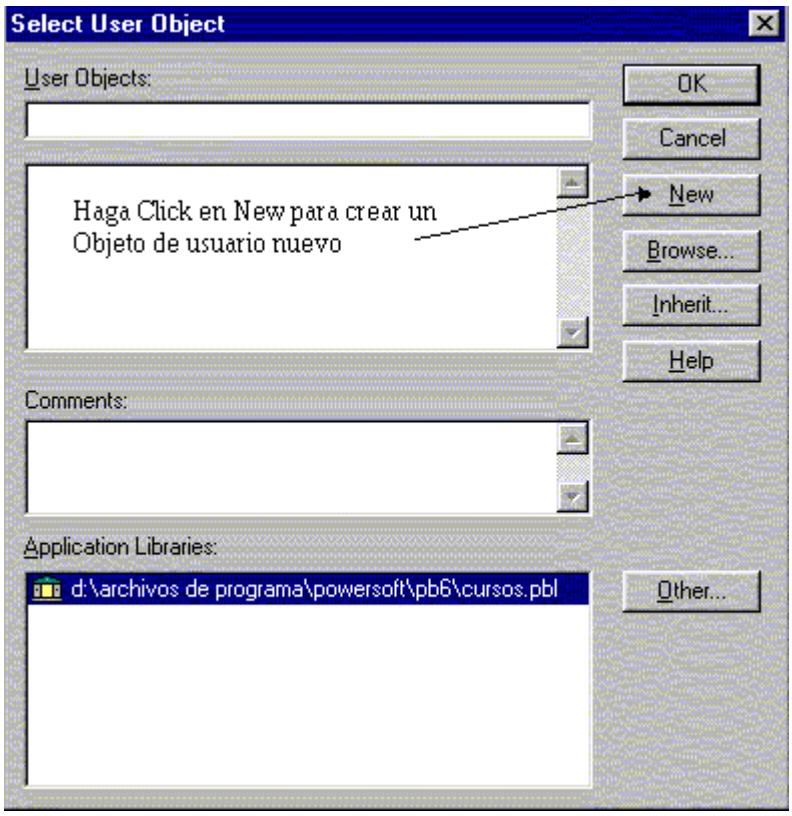
necesite usar este procesamiento, Ud. crea una instancia del objeto de usuario en un script , desde el cual puede accerder a la lógica del objeto de usuario.

Contruyendo un Objeto de Usuario (User Object)

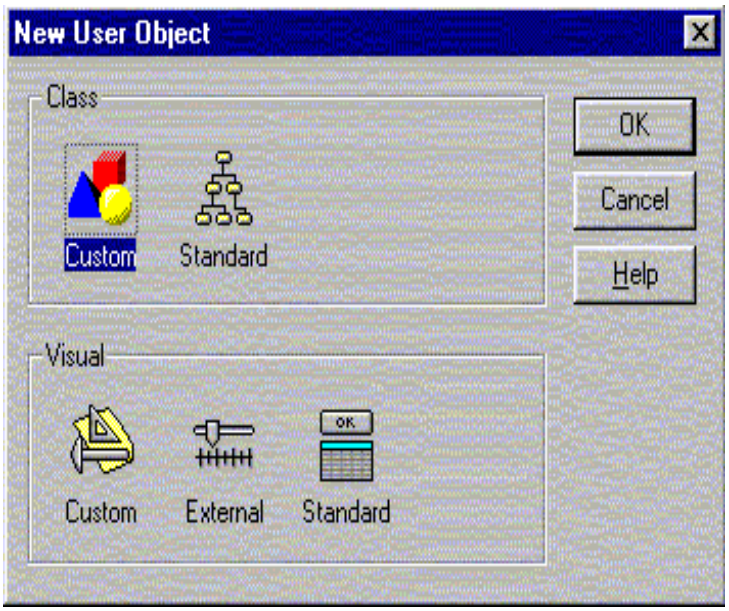
Ud. puede crear un objeto de usuario empezando desde cero, o puede crear objetos de usuario que hereden un estilo determinado , eventos, funciones, estructuras, variables y código(script) de un objeto de usuario que ya existe.

Como contruir un Objeto de Usuario Nuevo

1. Haga Click en Pintor de Objetos de Usuario(User Object) en la barra de herramientas PowerBar.
2. Luego debemos hacer click en el botón ***New*** para cear un nuevo Objeto de usuario.



3. Por último, en la pantalla de Objetos de Usuario, debemos elegir el tipo de Objetode Usuario que se desee crear, de tipo Clase o Visual, luego que elige el tipo haga click en ***OK***.



Pintor Menú (Menu Painter)

Acerca de los Menúes y Objetos Menú(Menu Objects)
Todas las ventanas en una aplicación , excepto en las ventanas de respuesta(Response) y las ventanas hijo(Child), deberían tener menúes. Los **Menúes** son listas de comandos relacionados u opciones(elemento del menú) que un usuario puede seleccionar en la ventana que esta actualmente activa. Cada opción en un menú es definida como un *Objeto Menú(Menu Objetc)* en PowerBuilder. Los **Objetos Menú** se pueden desplegar en una barra de menú o en menúes DropDown o Cascada.

Acerca del uso de menúes
Para usar menúes en PowerBuilder se puede construir de dos formas:

- ***En la barra de menú de una Ventana:*** Los menúes para ventanas(Windows) son asociados con una ventana en el Pintor ventana(Window Painter) y se despliega siempre que que la ventana es abierta.
- ***Como Menúes Popup:*** estos menúes se despliegan solamente cuando en un script se ejecuta la función PopMenu.

Acerca de como diseñar un menú
PowerBuilder da una completa libertad cuando se va a diseñar un menú. Pero se debería seguir ciertos convenios para que el ambiente operativo este en orden para hacer fácil el uso de la aplicación. Por ejemplo: Ud. debería mantener menúes simples y consistentes . Se debería agrupar en opciones relacionadas en un menú dropdown. Debería usar menúes en cascada para economizar y restrigir opciones en un solo nivel.

Acerca de cómo construir Menúes
Cuando se construye un menú , Ud. debe:

- Especificar la apariencia y comportamiento de los objetos menú colocando sus propiedades.
- Construir scripts(código PowerScript) que determinen como responde a un evento en los objetos menú. Para que soporte estos scripts, Ud. puede declarar funciones, estructuras y variables para el menú.

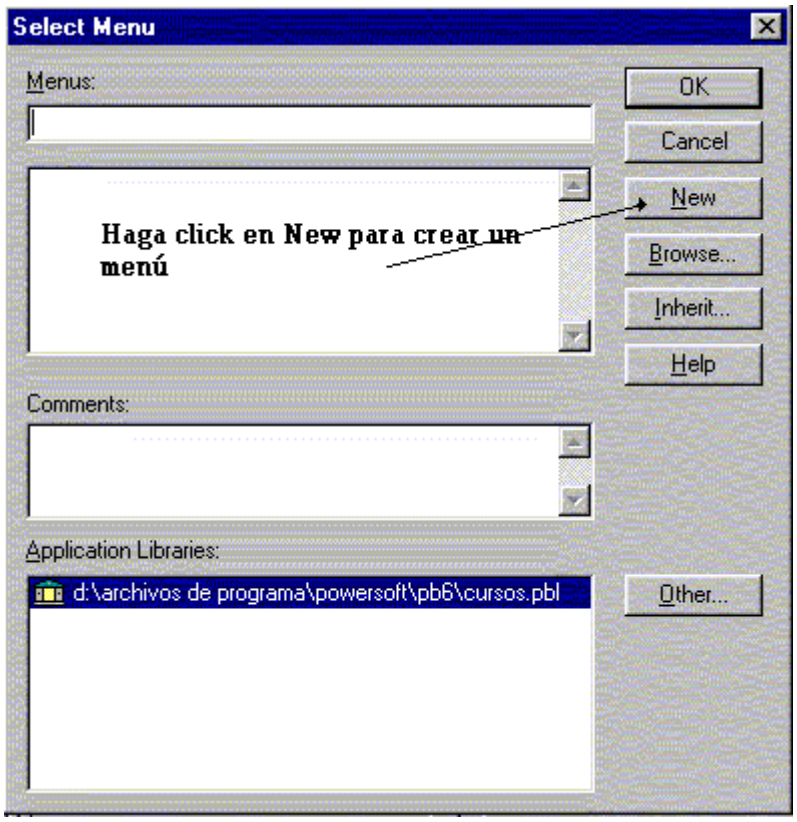
Hay dos maneras para construir un menú:

1. Construir un menu desde cero.
2. Construir un menú que herede un estilo, estructuras, variables y scripts(código) de un menú ya existente.

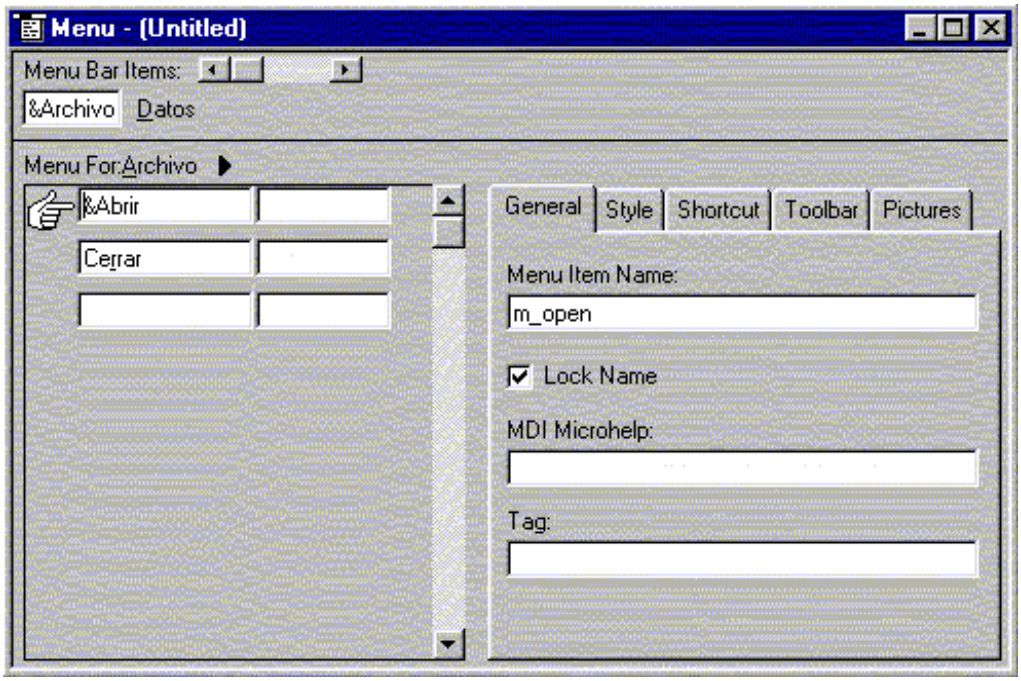
Construyendo un Menú nuevo

Abriendo el Pintor Menú(Painter Menu)

1. Haga click en el pintor Menú de la barra de herramientas PowerBar para que le aparezca una ventana que me permite hacer un menú nuevo o heredar de uno ya existente.



2. Luego haga Click en el botón ***New*** para contruir un menú nuevo. El pintor Menú despliega un espacio de trabajo pra crear el nuevo menú.



El pintor Menú tiene varios areas de trabajo en las cuales se especifica las diferentes partes de un menú:

Menú Bar: Seleccione un Objeto de menu ya existente de la barra de menú(menu bar) o uno nuevo al inicio de la ventana para crear un menú. El menú en la imagen de la ventana anterior tiene dos objetos en la barra de menú(menu bar): Archivos y Datos.

Menúes DropDown y en Cascada: Seleccione un objeto Menú existente de un menú Dropdown o en cascada o especifique uno nuevo en la hoja de menu izquierda. La imagen anterior muestra 2 opciones en el menú dropdown Archivo: Abrir y Cerrar.

Propiedades de los Objetos Menú: Especificando la apariencia y comportamiento del objeto menu selecionado en la hoja derecha de la ventana de menú. Ud. puede especificar propiedades para las opciones de la barra de menú y opciones en el menú. En la imagen anterior se muestra en la página de propiedades ***General*** para la opcion del menu dropdown Abrir,por ejemplo su nombre es *m_open*.

Pintor Estructura (Structure Painter)

Qué son las Estructuras(Structures)?
Una ***estructura*** es una colección de una o más variables relacionadas de un mismo o diferente tipo de datos agrupados bajo un mismo nombre. En Algunos lenguajes como COBOL o Pascal las estructuras son llamadas *registros*.
Las estructuras le permiten referirse como entidades relacionadas como una unidad mas que individualmente.
Hay dos tipos de estructuras:

1. ***Estructuras Globales:*** Estas estruturas no están asociadas a ningún objeto en su aplicación. Ud. puede declarar una instancia de la estructura y referirse ala instancia en algún script de la aplicación.

2. ***Estructura a nivel-de un objeto:*** Estas estructuras están asociadas a un tipo particular de, ventana(window), menú, u objeto de usuario, o con el objeto Aplicación. Estas estructuras pueden siempre ser usadas en un script pero solo de objeto dado. Se puede además se puede escoger como hacer estructuras accesibles desde otros scripts.

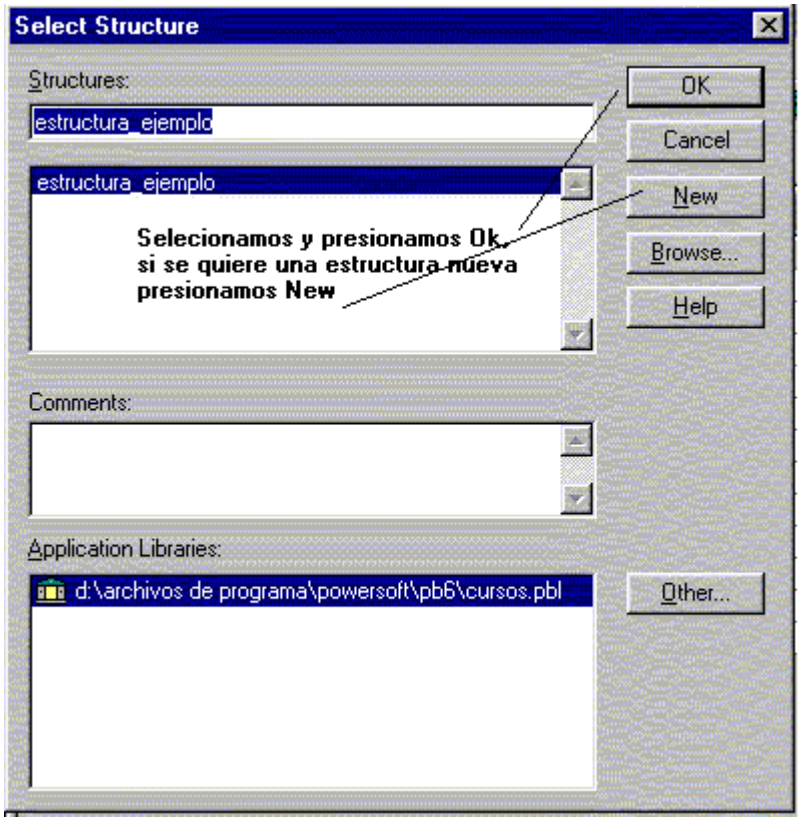
Definiendo estructuras:

1. **Abrir el pintor de Estructura(Structure Painter)**

    - ***Abrir el pintor Estructura para crear una estrcutura Global***

        1. Haga Click en el botón del pintor estructura en la barra de herramientas PowerBar y aparece una ventana de diálogo para selecionar una estructura

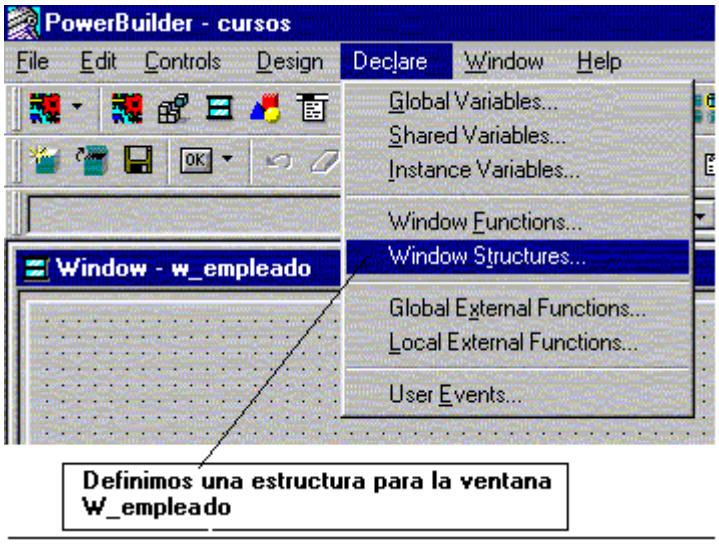
2. Si existe ya una estructura , seleccione la estructura y haga click en ***OK*** o haga click en el botón de New para declarar una estructura nueva.



- ***Abrir el pintor Estructura para crear una estrcutura a nivel de un objeto***

1. Abra el Pintor para el objeto que Ud. quiere declarar una estructura Ud. puede declarar estructuras para ventanas(windows), menus, objetos de usuario, o aplicaciones. Por ejemplo, si Ud. quiere declarar una estructura para una ventana w_empleado abra el pintor de ventana(painter window) y seleccione w_empleado.

2. Luego de haber elegido la ventana w_empleado, del menú la opción *Declare* , luego seleccione la opción apropiada, en este caso, *Window Structure*, si son otros objetos, Menu Structure, User object Structure, o Aplicación Structure.



Luego aparece una ventana de diálogo, donde debemos elegir una estrcutura ya existente y presionamos OK, o si desea declarar uno nuevo presione el botón New.

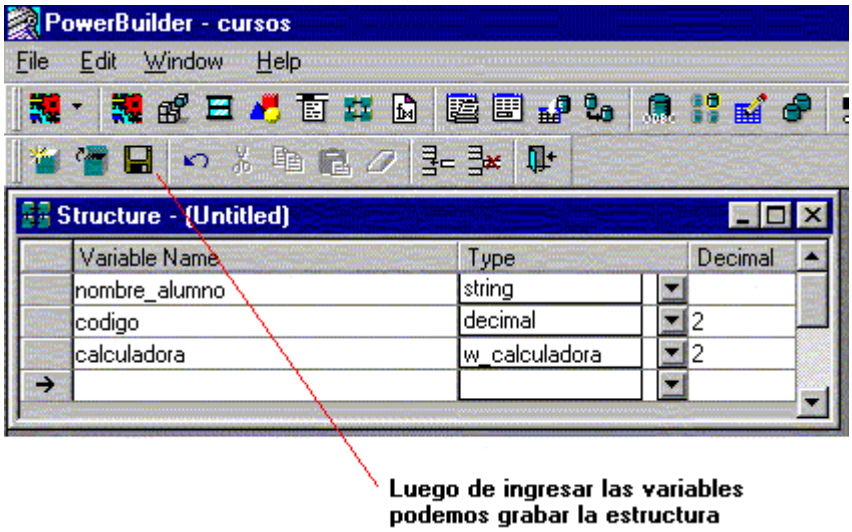
2. **Definir las variables que componen la estructura**

    1. Ingrese el nombre de las variables que usted quiere incluir en la estructura.

    2. Ingrese el tipo de datos de la variable. El default para la primera variable es string(cadena). Se puede especificar cualquier tipo de datos de PowerBuilder, incluyendo tipo de datos standard, tales como enteros(integer) y Cadenas(string), asi como tambipen objetos y controles, tales como ventanas(window), MuiltiLineEdit, ect.
    Además se puede especificar cualquier tipo de datos que Ud. ha definido. Por ejemplo, si Ud. está usando una ventana llamada w_calculadora que Ud. tiene definida y quiere incluir en una estructura la ventana, escriba *w_calculadora* como un tipo de datos.(No se puede seleccionar w_calculadora de la lista , ya que ésta solo muestra tipos propios creados por PowerBiulder).
    Una estrcutura como variable
    Una variable en una estructura puede ser en sí otra estructura. Especifique el nombre de la estructura como un tipo de variable de datos.

    3. Ingrese el número de decimales en el lugar de los DEC si el tipo de datos es decimal. El default es 2.

    4. Repita los pasos hasta que haya terminado de ingresar todas la variables.



3. **Grabar la estructura**
   El nombre de la estructura puede tener un máximo de 40 caracteres.
   Se puede adoptar una convención para poner el nombre a una estructura. Por ejemplo, si está definiendo una estructura global, de puede llamar con el prefijo **s_** (s_nombre), y si una estructura es a nivel de un objeto , su nombre con el prefijo **os_** (os_datos_alumno).
   Una vez haya definido la estrcutura, se debe grabar y ponerle un nombre, adenás puede ponerse comentarios, pero solo cuando es una estructura global).

Usando Estructuras

Pintor Funciones (Function Painter)

Qué son funciones definidas por el usuario ?

El Lenguaje PowerScript tiene muchas funciones que vienen construidas por PowerBuilder. Pero Ud. podría encontrar que necesita programar el mismo proceso una y otra vez. Por ejemplo, Ud. podría necesitar llevar a cabo cierto cálculo en varias partes de una aplicación o en diferentes aplicaciones. En tal situación ud. querrá crear una ***Función definida por el usuario*** para realizar el proceso.

Una Función definida por el usuario es una colección de sentencias que desempeñan algún proceso. Ud. puede usar el pintor de Función(Painter Function) para definir funciones credas por el usuario. Después de definir la función y grabar dentro de una librería, alguna aplicación accesará a esa librería para usar la función.

Hay dos tipos de definir funciones creadas por el usuario:

1. ***Funciones Globales:*** que no están asociadas a ningún objeto en la aplicación y que están siempre accesibles desde cualquier parte de la aplicación.

2. ***Funciones a nivel-de un objeto:*** que están definidas para un tipo particular de ventana(window), menu, u objeto de usuario, o para un objeto aplicación. Estas

funciones son parte de la definición de un objeto y pueden siempre ser usadas en el script solo de ese objeto. Ud. puede escoger y hacer que estas funciones sean accesibles desde otro script también.

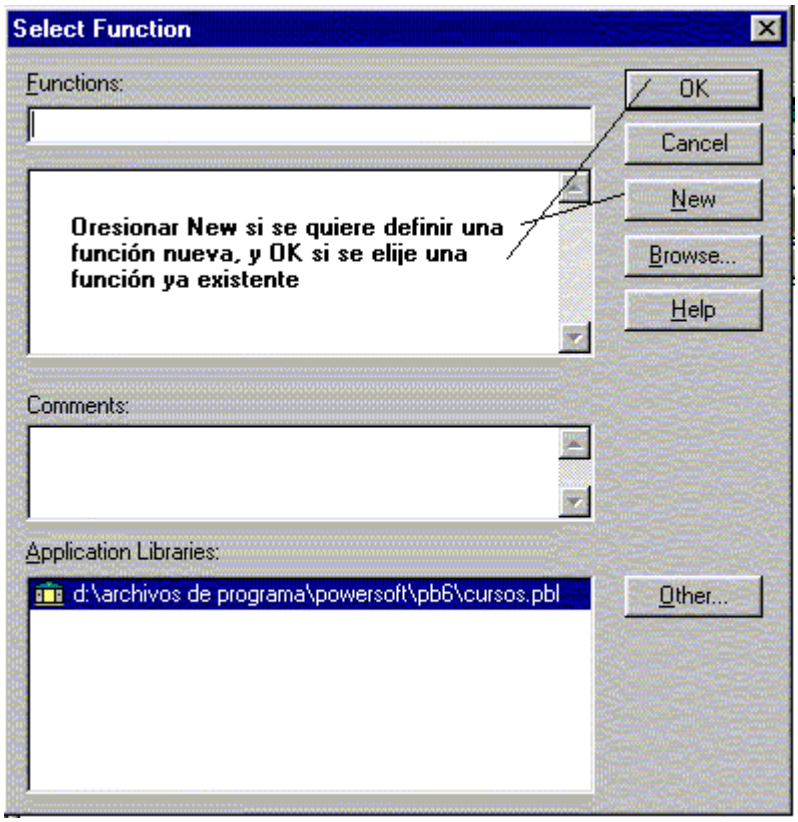
Definiendo Funciones creadas por un usuario

1. ***Abrir el pintor Función(Function painter)***

   - ***Abrir el pintor Función para crear una Función Global***

     1. Haga Click en el botón del pintor función en la barra de herramientas PowerBar y aparece una ventana de diálogo para selecionar una función

     2. Si existe ya una función , seleccione la función y haga click en ***OK*** o haga click en el botón de **New** para declarar una función nueva.



   - ***Abrir el pintor Función para crear una función a nivel de un objeto***

     1. Abra el Pintor para el objeto que Ud. quiere declarar una función Ud. puede declarar funciones para ventanas(windows), menus,

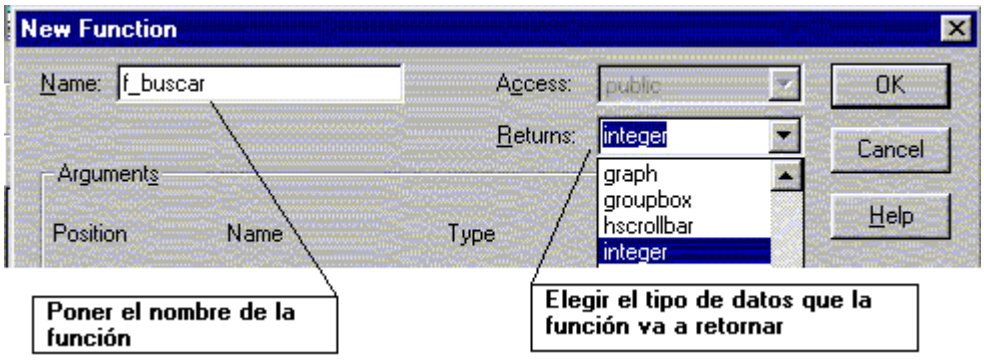
objetos de usuario, o aplicaciones. Por ejemplo, si Ud. quiere declarar una función para una ventana w_empleado abra el pintor de ventana(painter window) y seleccione w_empleado.

2. Luego de haber elegido la ventana w_empleado, del menú la opción *Declare* , luego seleccione la opción apropiada, en este caso, *Window Functions*, si son otros objetos, Menu Functions, User object Functions, o Aplicación Functions.

Luego aparece una ventana de diálogo, donde debemos elegir una función ya existente y presionamos OK, o si desea declarar una nueva presione el botón New.

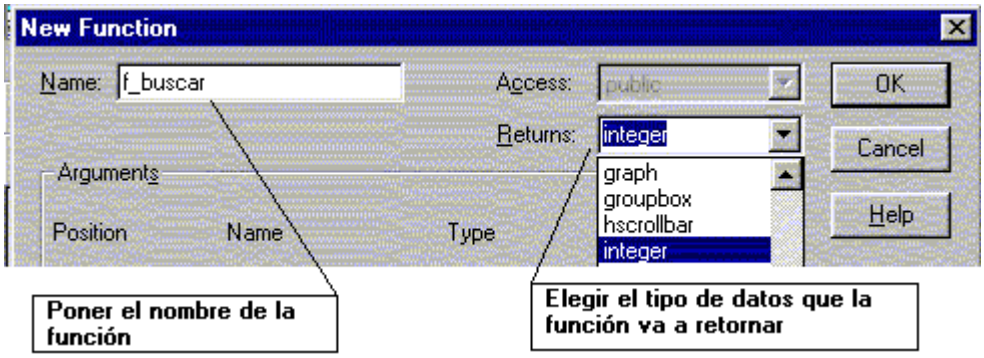
2. ***Poner el nombre de la función***
   El Nombre de las funciones pueden tener hasta 40 caracteres.
   Es buena idea desarrollar una convención de nombres para las funciones creadas por el usuario asi Ud. puede reconocer las funciones fácilmente y distinguirlas de las funciones propias de PowerBuilder.
   Por ejemplo: se podría dar el nombre a las funciones globales con un prefijo **f_** por ejemplo: (f_encontrar) y para funciones a nivel de objetos con un prefijo **of_** por ejemplo: of_chequearpadre, of_refreshwindow.

3. ***Definir el tipo que va a retornar la función***
   Muchas funciones desempeñan algunos procesos y entonces deben retornar un valor. Este valor puede ser el resultado de un procesamiento o un valor que indica si la función se ejecutó satisfactoriamente o no. Si tiene su función que retornar un valor, ud. necesita definir el tipo que va ha retornar(*return type*), que especifica el tipo de dato del valor que se va a retornar.
   Cuando de va a definir una función, ya sea global o a nivel de objeto, podemos elegir el tipo que se va a retornar, como se muestra en la figura:



Si se quiere que la función no retorne ningún valor, se debe elegir ***None*** de la lista *return type*.

4. **Para una función a nivel de un objeto, definir un nivel de acceso**
   La ventana de diálogo tiene un lista dropdown que tiene los tipos de acceso a una función.



En esta ventana se especifica el nivel de acceso de la función -el lugar desde el cual Ud. puede llamar a la función en una aplicación.

Para Funciones Globales

Las funciones globales pueden ser llamadas desde cualquier parte de la aplicación. En términos de PowerBuilder, estas funciones son **Públicas(Public)**. Además cuando Ud. está definiendo una función global, no se puede modificar el acceso.

Para Funciones a nivel de un objeto

Se puede restringir el acceso a una función a nivel de objetos por un conjunto de niveles de acceso como se muestra a continuación:

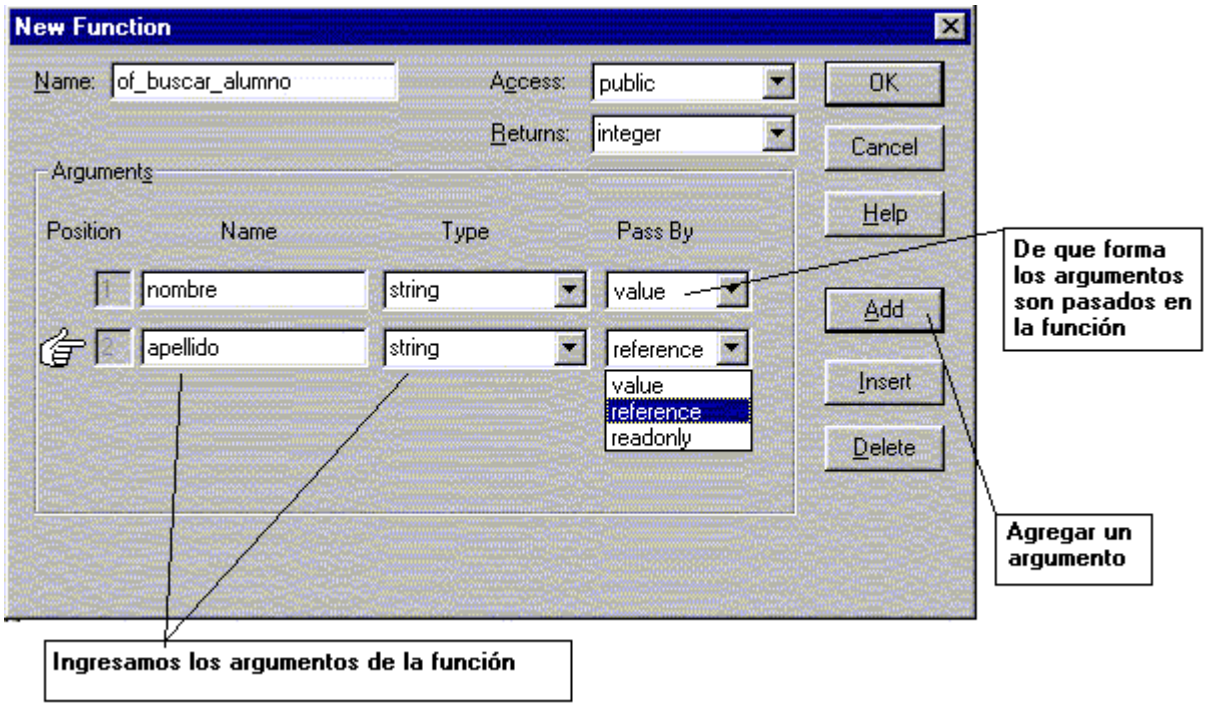
Acceso	Desde donde se puede llamar a la función
Public	Desde cualquier script de la aplicación
Private	Solo en scripts de eventos en el objeto en el cual la función es definida. No se puede llmar a la función desde los objetos decendientes de otro objeto.
Protected	Solo en scripts para objetos en los cuales la función fue definida y para los objetos descendientes del objeto padre.

Si una función está solamente para ser usada dentro de un objeto, debería definir el acceso como private o protected. De esta forma. ud. garantiza que la función nunca sea llamada inapropiadamente desde fuera del objeto.(En términos orientados a objetos, definiendo funciones como private o protected se ***encapsula*** la función dentro del objeto).

- ***Definir los argumentos para la función***

Como en las funciones hechas en PowerBuilder, las funciones definidas por el usuario pueden tener un determinado número de argumentos o ninguno. Ud. declara los argumentos y sus tipos cuando define la función.
Pasos para definir los argumentos:

1. Poner el nombre del argumento(el orden como especifica los argumentos es el orden cuando se llama a la función)
2. Declarar el tipo de los argumentos, existen varios tipos:
    - Tipos de datos propios del PowerBuilder(Integer, real, etc)
    - Tipos de Objetos(tales como window) u objetos específicos(tales como w_empleado)
    - Objetos de usuario
    - Controles (tales como CommmandButtons)
3. Declarar como se quiere que los argumentos sean pasados, existen tres tipos para pasar los argumentos:



- ***Por referencia:*** Cuando se pasa un argumento por referencia, la función tiene acceso a los argumentos originales y se puede cambiar los datos directamente.

- ***Por valor:*** Cuando se pasa por valor, se esta pasando a la función una copia temporal y local del argumento. La función puede cambiar el valor de la copia local del argumento dentro de la función, pero el valor del argumento no es cambiado desde el script que es llamada la función.

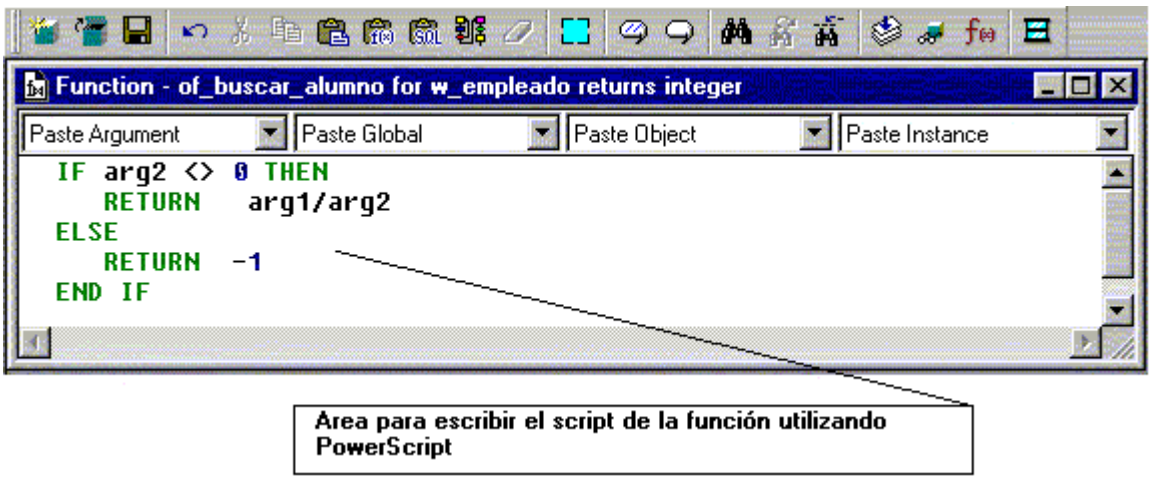
- ***Solo lectura(ReadOnly):*** Cuando pasa un argumento solo lectura, el valor de la variable está disponible en la función, pero es tratada como una constante. Este tipo provee un gran desempeño para valores como cadenas (strings), Bolbs, Date, time, DateTime, por que no crea una copia del dato que es pasado.

4. Si se quiere agregar otro argumento haga click en el boton **Add** y repita los pasos del 1 al 3.

5. Finalmente haga click en **OK**.

- ***Implementar el código para la función***

El pintor Función, una vez que se ha definido la función con sus parámetros, despliega un espacio de trabajo para poder implementar el codigo en la función, esto es, con PowerScript.



Una función definida por el usuario puede contener sentencias PowerScript, sentencias SQL integradas y llamadas a funciones propias del PowerBuilder, y otras funciones.
Para retornar una valor se utiliza la **sentencia RETURN**:
RETURN ***expesion***
Ejemplo:

```
IF arg2 <> 0 THEN
                RETURN   arg1/arg2
             ELSE
                RETURN  -1
             END IF
```

Pintor DataWindow (DataWindow Painter)

Introducción a los objetos DataWindow

Un Objeto DataWindow es un objeto que se usa para recuperar, presentar y manipular datos de una base de datos relacional u otra fuente de datos(tales como una tabla de excel o un archivo de dBase).

Los objetos DataWindow tienen el conocimiento acerca de los datos que ellos están recuperando. Ud. puede formatos para desplegar los datos, estilos de presentación.

Cómo usar los objetos dataWindow

Antes que Ud. pueda usar el objeto DataWindow en una aplicación, necesita construir el objeto. Se debe utilizar el pintor Datawindow(Datawindow painter), que le permite crea y editar objetos DataWindow. Adicionalmente, permite hacer archivos PSR(PowerSoft Report) que además le permitirían usar en una aplicación. Un archivo PSR contiene la definición de un reporte (esencialmnete objetos DataWindow sin actualizar tablas) asi como el contenido de datos en este reporte cuando el archivo PSR fue creado.

Ejemplos de objetos DataWindow

Ud. puede desplegar datos de la mejor forma de presentación para el usuario:

Estilos de Edición(Edit styles)

Si una columna puede tomar solamnete u pequeño numeros de valores, ud. puede hacer aparecer los datos como botones de radio(radio buttons) en un objeto datawindow y el usuario sabe que debe elegir uno de ellos.

Formatos de presentación (Display formats)

Si una columna despliega un número telefónico , salarios, o fechas, ud. puede especificar el formato apropiado para el dato.

Reglas de validación

Si una columna puede tomar números solamente en un rango específico, ud. puede especificar una regla simple de validación para la columna, sin escribir ningún código, y así asegurarse que el usuario ingrese datos válidos.

Reforzar los Objetos DataWindow

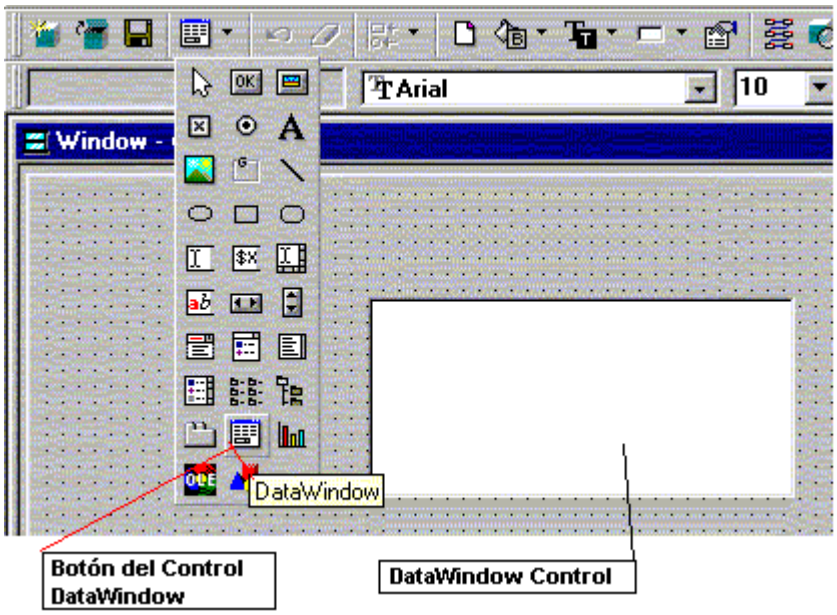
Si ud. quiere mejorar la presentación y manipulación de los datos en un objeto DataWindow, ud. puede incluir campos calculados(computed fields), imágenes(pictures) y gráficos que son ligados directamente a los datos recuperados por el objeto.

Esta sección describe los procesos sobretodo para la creación y uso de los objetos DataWindow.

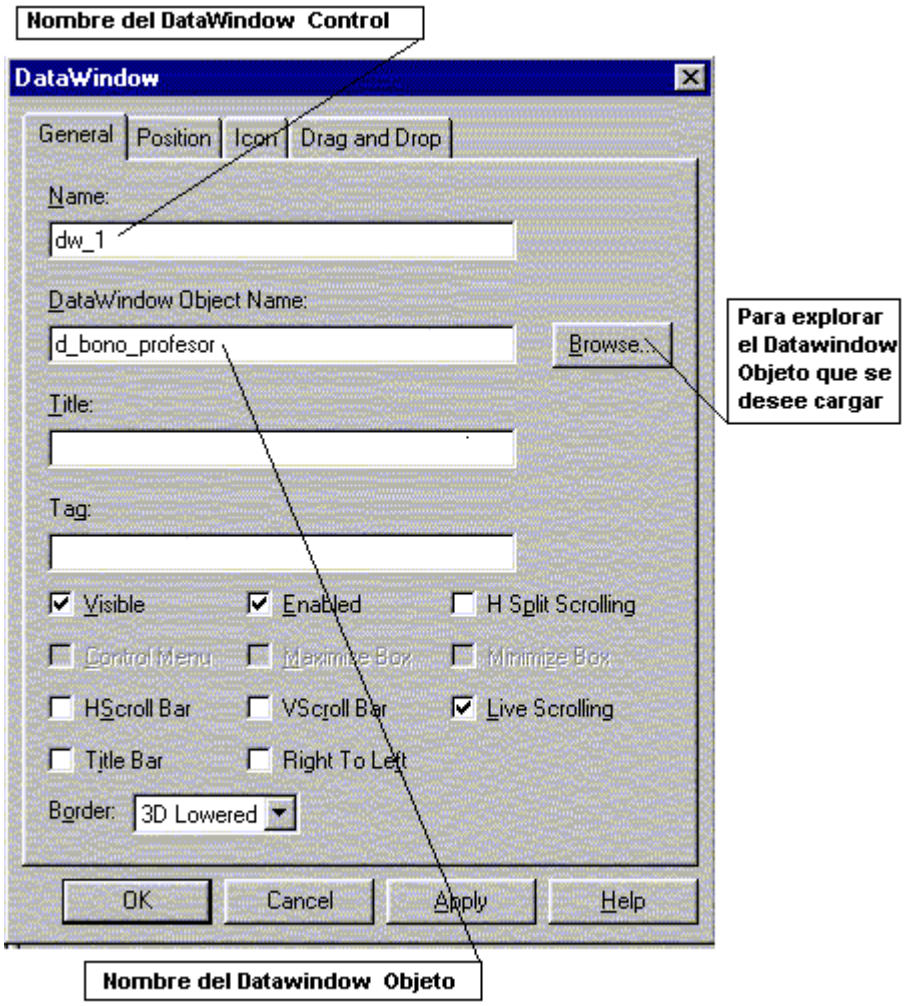
Usando los objetos Datawindow

1. Construir un objeto DataWindow (o archivo PSR) haciendo click en el botón del Pintor DataWindow. En este pintor, Ud. define la fuente de datos, estilo de presentación y todas las otras propiedades del objeto, tales como, reglas de validación, ordenamiento y filtros y gráficos.

2. Colocar un control DataWindow en una ventana(o en objeto de Usuario).
   A través de este control su aplicación se puede comunicar con el objeto datawindow que ha creado con el pintor DataWindow.



3. Asociar el control DataWindow con el objeto DataWindow. Hacemos doble click sobre el control datawindow o hacemos click derecho sobre el control datawindow para ver las propiedades del control y poder ligar con el datawindow objeto.



4. Escribir código script en una ventana para manipular de control DataWindow y su contenido.
   Por ejemplo: Ud. puede usar la función Retrieve de PowerScript para recuperar datos dentro de un control DataWindow.
   Ud. puede escribir scripts para el control DataWindow y tratar de manipular errores, compartiendo datos entre controles DataWindow.

5. Escribir código para controlar un proceso un proceso que es iniciado cuando ocurre un evento en el control DataWindow.
   Ud. puede escribir scripts para el control DataWindow y tratar de manipular errores, compartiendo datos entre controles DataWindow.

Objetos Datawindow versus Reportes

Pintor Reporte (Reporte Painter)

Los Reportes presentan datos. El pintor Reporte en PowerBuilder provee de muchas maneras para presentar los datos. Ud. podría requerir de reportes tabulares con filas y columnas llenas de información. A veces un gráfico o un crosstab es una mejor manera para presentar los datos.

Los reportes en PowerBuilder pueden además estar con etiquetas para envíar por correo o muchos reportes jerarquizados que se encuentran en la misma página. PowerBuilder además tiene reportes de forma libre(FreeForm) que le permite colocar texto, datos, líneas, cajas de texto, y gráficos en cualquier parte que Ud. desee.

Reportes versus Objetos Datawindow

Para crear un reporte nuevo se hace de la misma manera como para crear un nuevo objeto DataWindow.
Construyendo un Reporte

Pintor Consulta (Painter Query)

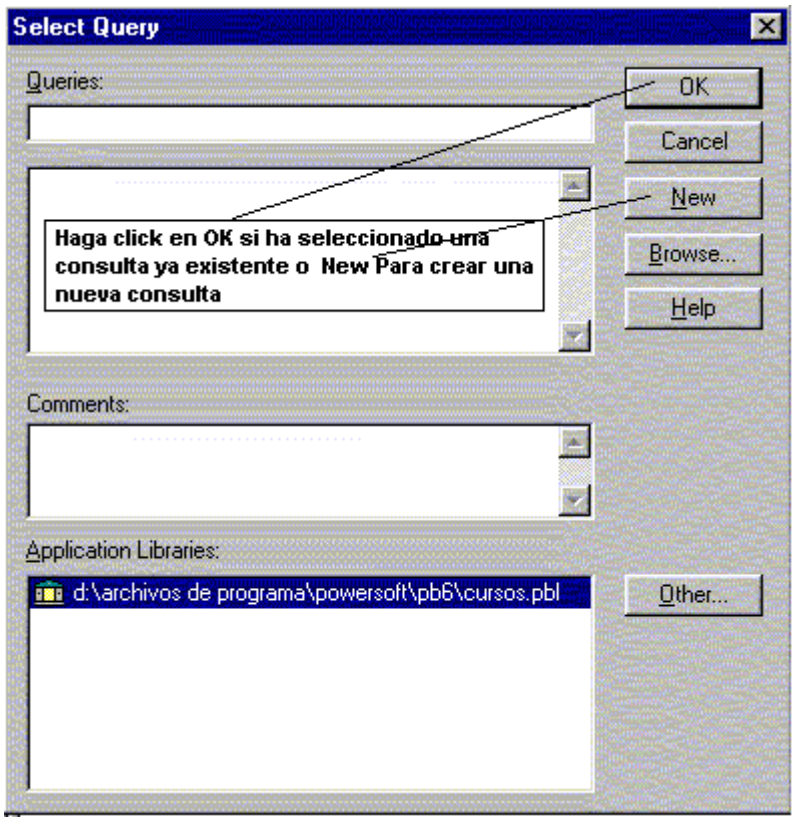
Una Consulta es una sentencia SQL SELECT creada con el pintor Consulta(Query Painter) y grabada con un nombre y puede ser usada repetidamnete como una fuente de datos para un objerto DataWindow.

Las consulta ahorran tiempo, porque ud. especifica todos los requerimientos de datos solo una vez. Por ejemplo, se puede especificar las columnas, cuales filas se van a recuperar, y el ordenamiento de una consulta. Las veces que Ud. quiera crea objetos DataWindow usando estos datos, simplemente especificando la Consulta(Query) como la fuente de datos.

Definiendo una Consulta(Query)

1. Haga click en el botón del Pintor Consulta(Query painter) en la barra de herramientas PowerBar.

2. Haga Click en el Botón **New** para crear una Consulta nueva en la ventana de diálogo o elija una consulta ya existente y haga click en **OK** para modificar la consulta. Entonces se despliega un área de trabajo para crear o modificar una consulta.



3. Seleccione las tablas de la ventana de diálogo que aparece y luego las columnas que desea que se despliegen en la consulta.