

Maquetado de formularios, controles y estilos

Vamos a construir una pequeña aplicación en la se va a comprobar las diferencias entre aplicarlo o no formato a un formulario. También se va a ver cómo intercambiar datos entre formularios y por último, cómo aplicar estilos generales a la aplicación, estilos mediante clave y estilos por código.

La aplicación le va a pedir al usuario los datos de un evento, los va a validar, y se los va a volver a mostrar

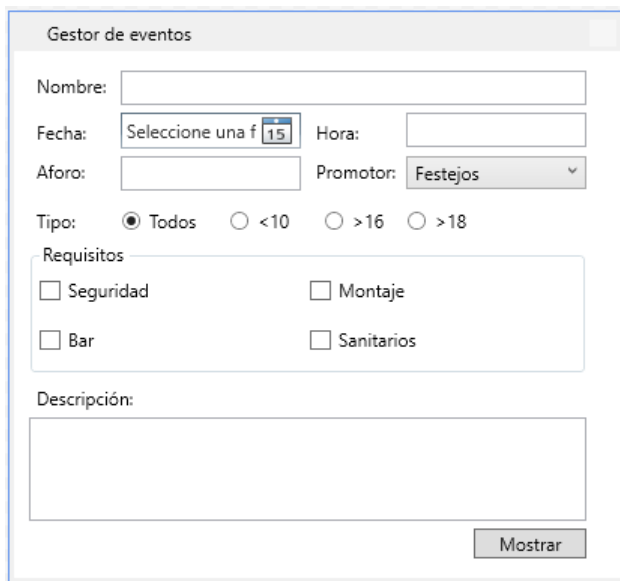
Antes de empezar con la presentación vamos a necesitar definir una clase de negocio denominada “Evento” con los siguientes campos:

- Nombre, de tipo cadena
- Fecha, de tipo Datetime
- Hora, de tipo Datetime
- Aforo, de tipo entero
- Tipo, de tipo TipoEventoEnum
- Descripción, de tipo cadena
- Seguridad, de tipo Booleano
- Bar, de tipo Booleano
- Montaje, de tipo Booleano
- Sanitarios, de tipo Booleano
- Promotor, de tipo Booleano

El tipo enumerado “TipoEventoEnum” tiene los siguientes valores:

- Todos
- Menor de 10
- Mayor de 16
- Mayor de 18

El formulario principal tiene el siguiente aspecto:



The screenshot shows a Windows-style application window titled "Gestor de eventos". Inside, there is a form with the following elements: a text box for "Nombre"; a date picker for "Fecha" showing "15"; a text box for "Hora"; a text box for "Aforo"; a dropdown menu for "Promotor" with "Festejos" selected; a group box "Tipo" containing four radio buttons: "Todos" (selected), "<10", ">16", and ">18"; a group box "Requisitos" containing four checkboxes: "Seguridad", "Montaje", "Bar", and "Sanitarios"; a text area for "Descripción"; and a "Mostrar" button at the bottom right.

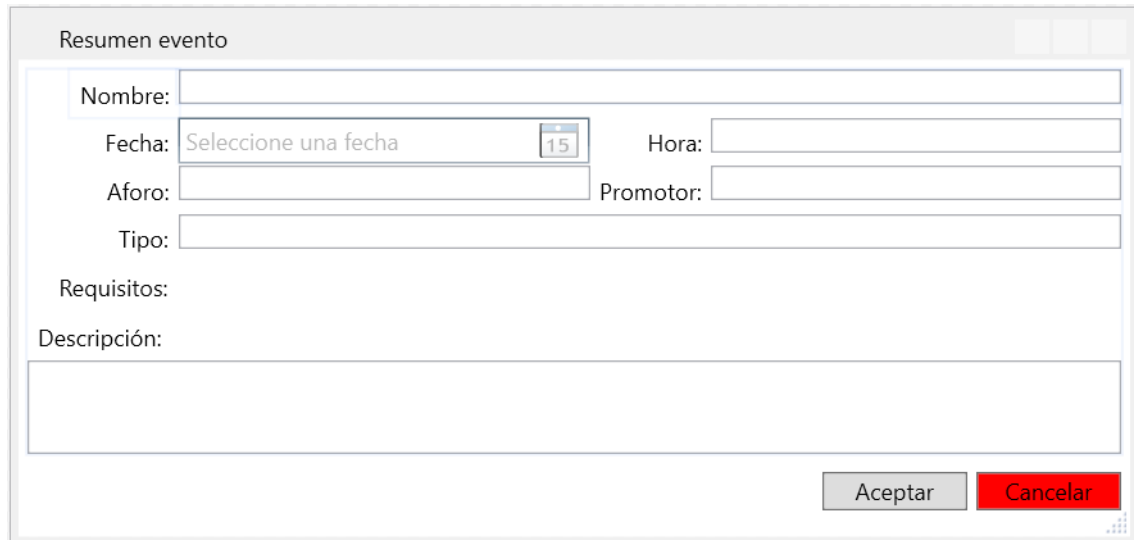
Consideraciones formulario principal:

- La ventana tiene título y no es redimensionable
- La ventana tiene de elemento raíz un Grid por lo que los controles se redistribuirán de forma manual
- El campo fecha tiene el formato de fecha corto y es obligatorio
- El campo hora tiene el formato hh:mm y es obligatorio
 - o Solo se permiten números, “:”, borrar y tabular
 - o Al abandonar el campo comprobar que la hora es válida
 - o Intenta dar formato automáticamente, es decir, que el usuario no necesite escribir “:”, si no que aparezca automáticamente si el tercer dígito es un número.
- El campo aforo con formato de número entero y es obligatorio
- El campo festejos es un desplegable con los siguientes valores: Deportes, Festejos (por defecto), Educación y Externo
- En los controles, el campo tipo se agrupan con el nombre “rbTipo”.
 - o Gestiona la modificación del valor del campo con un evento común y una variable de respaldo
- El campo requisitos es un GroupBox cuyo contenido se distribuye en un grid con dos columnas y 2 filas. En cada celda va uno de los datos de tipo Booleano del evento.
- El campo descripción acepta múltiples líneas y tiene activado el corrector ortográfico. El texto que exceda el espacio disponible se recortará
- Al pulsar el botón se realizarán las siguientes acciones.
 - o Se validarán los datos.
 - En caso de que algún dato sea erróneo se mostrará una caja de mensaje con el error, un icono de información y un botón de aceptar.
 - Tras detectarse un error el control invalido recibirá el foco.
 - o Se abrirá de forma modal la ventana resumen, esta ventana recibe un objeto de tipo evento con los valores indicados por el usuario.
 - En función del DialogResult de la ventana mostrar una caja de mensaje con el texto “Datos confirmados” o “Datos rechazados”.

Teoría eventos

- PreviewKeyDown -> se produce cuando se pulsa una tecla. El parámetro “e” contiene un par de propiedades interesantes:
 - o Key, enumerado con el valor de la tecla pulsada.
 - o KeyboardDevice, teclado en el que se produce el evento. La función “IsKeyDown” permite conocer si se está pulsando una tecla adicional como el shift o el control.
- PreviewTextInput -> se produce con cada nueva cadena que se añade al control, útil para introducir automáticamente el separador de hora “:”.
- TextChanged -> se produce tras modificar el texto.

El formulario de resumen tiene el siguiente aspecto:



Consideraciones

Consideraciones

- La ventana tiene título y es redimensionarle agarrando desde la esquina inferior derecha.
- La ventana tiene de elemento raíz un DockPanel.
 - o En la base se ancla un grid con dos botones alineados a la derecha.
 - o El resto del espacio lo ocupa un grid con 4 columnas y 7 filas.
 - Las columnas con nombre de campos ajustan su tamaño automáticamente.
 - La fila con los datos de los requisitos ajusta su altura automáticamente.
 - La fila con los datos de la descripción ocupa el doble de espacio por defecto.
 - Ajusta las alineaciones y extensión de las celdas según la imagen.
- Todos los controles asociados a datos o están deshabilitados o son de solo lectura, según lo que permita su tipo.
- El campo de fecha es un selector de fecha.
- El campo requisitos es un TextBlock cuyo contenido tiene el siguiente formato:
 - o Por cada booleano de la clase evento con valor true se añade una línea dentro de un "Run" con una descripción del campo.
 - o Entre líneas debemos añadir un salto de línea.
 - o Las líneas pares tendrán de color de fondo "Colors.Aqua".
 - o Las líneas impares tendrán de color de fondo "Colors.LightGreen".
- Si no se indica lo contrario el tipo de dato es caja de texto.
- El botón aceptar está asociado a la tecla Intro, al pulsarlo se establece el DialogResult de la ventana a true.

- El botón cancelar está asociado a la tecla Esc, al pulsarlo se establece el DialogResult de la ventana a false.

Respecto a los estilos

- Todos los textos de la aplicación son del tipo “Comic Sans M”.
- Los estilos de las filas pares e impares se establecen mediante código según se van procesando los datos.
- El botón cancelar tiene de color de fondo el color “Red” y cuando el ratón está encima el color “OrangeRed”, además este estilo se almacena en la ventana y tiene el nombre “resaltar”.
 - NOTA: Es muy fácil, existe una opción en el diseñador para sacar explícitamente al código xaml el formato por defecto de un control **seleccionado**.