



Manual de usuario



MONTEVIDEO - URUGUAY
BUENOS AIRES - ARGENTINA
CIUDAD DE MÉXICO - MÉXICO
CHICAGO - USA

Av. Italia 6201 - Edif. Los Pinos, P.1
Moreno 4745 P.3
Sánchez Azcona 1550
400 N. Michigan Ave. Suite 1600

(598) 2601 2082
(54 11) 4008 8020
(52 55) 5639 5455
(1 312) 836 9152

1	Introducción:	5
2	Pautas previas:	5
2.1	Pautas previas antes de la aplicación del Pattern K2BEntityServices:	6
2.1.1	Transacciones:	6
2.1.2	Atributos:	7
2.1.3	Determinación de instancia por defecto:	7
2.1.4	Posicionamiento de atributos en las pantallas:	7
2.1.5	Determinaciones de descripciones de atributos en las pantallas:	8
2.1.6	Determinación de control info asociado a los atributos:	9
2.1.7	Generación de filtros por defecto:	10
2.2	Pautas antes de aplicar la instancia:	10
2.3	Pautas antes de ejecutar la aplicación:	10
3	Actualización de transacciones:	11
3.1	Update Transaction = Apply K2BWWStyle:	11
3.1.1	Actualización de web form:	11
3.1.2	Actualización de código:	15
3.1.3	Instanciación automática de la clave foránea:	19
3.2	Propiedades en la instancia sobre la transacción:	21
3.2.1	Update Transaction:	21
3.2.2	GenerateSlotsForFK:	22
3.2.3	Navigation:	22
3.3	Flexibilidad en la invocación de la transacción	22
3.4	Propiedades avanzadas para la transacción:	24
4	Objetos con Grilla	25
4.1	Filtros y condiciones:	26
4.1.1	Mecanismo genérico para la definición de variables:	26
4.1.2	Propiedades asociadas a los filtros:	27
4.2	Órdenes:	27
4.3	Atributos:	28
4.3.1	Variables:	29
4.3.2	Seteo de propiedades de los atributos:	30
4.4	Modos y Acciones:	31
4.4.1	Acciones de usuario:	31
4.4.2	Propiedad RowSelection:	35
4.4.3	Algunas otras propiedades:	36
4.4.4	Acciones con confirmación:	37
4.4.5	Acciones en general and paralel transaction:	38
4.4.6	Condiciones asociadas a las acciones	38
4.4.7	Nodo Standard Actions (K2BEntityServices.config)	40

4.5	Automatic Refresh:	41
4.6	Paginado:	41
4.7	Scroll	43
4.8	Propiedades del WorkWith	44
5	<i>Objecto EntityManager:</i>	45
5.1	Nodo Components.	47
5.1.1	Generación por defecto:	48
5.1.2	TabInfo de los componentes:	48
5.1.3	Componente UserDefined:	48
5.2	Avanzado:	49
5.2.1	Parámetros de los componentes (menos user defined):	49
5.3	Propiedades de interfaz asociadas a las vistas planas:	49
5.3.1	Propiedad NoSkip:	50
5.3.2	Grupos	54
5.3.3	Columnas	55
5.3.4	Right text y left text en atributos y variables	56
5.3.5	Contextual Help	57
5.3.6	Line Separator	58
6	<i>Reportes PDF y Excel.</i>	61
6.1	Particularidades reportes PDF	63
6.1.1	Objeto de Layout	64
6.2	Configuración nodo general	64
6.3	Atributos que van en los reportes y pantalla.	64
7	<i>Propiedades avanzadas del pattern K2BEntityServices:</i>	66
7.1	Settings y configuración general.	66
7.2	Master Pages	67
7.3	Manejo de sesión	69
7.4	Objetos básicos	69
7.4.1	Customización de objetos básicos	70
7.4.2	Recursos	70
7.5	Nomeclatura de objetos	71
7.6	Configuración general	71
8	<i>Propiedades avanzadas de los objetos</i>	73
8.1	Manejo de parámetros	73
8.2	Fixed data en el workwith	73
8.3	FormSections	74
8.3.1	Attributes Section	75
8.3.2	Web Components Section	77
8.3.3	Layout de los form sections	79



8.4	Eventos	80
8.5	Reglas	82
8.6	Variables	82
8.7	Propiedad Prompt	83
8.8	Control Info	84
8.9	Link de los atributos	86
8.10	Tips	86
8.10.1	Creación de grilla sin tabla base.	86
8.10.2	Acción que invoca a un evento de usuario.	87
8.10.3	Cambiar acción standard por acción de usuario	87
9	Customización interfaz K2BTools	88
9.1	Clases del tema	88
9.2	Imágenes	91
9.3	Objeto Layout	92
9.3.1	Layout Web Form	94
9.3.2	Layout Eventos y variables	99
9.3.3	Layout Rules	100
9.4	Pasos para modificar la interfaz generada por las K2BTools	102
10	Seguridad	108
10.1	Actividades dentro de una instancia	109
10.2	API de seguridad	111
10.2.1	Invocación API en transacción	113
10.2.2	Invocación API objeto con grilla	113
10.2.3	Alta de actividades	115
11	Pattern K2BPrompt	116
12	Pattern K2BMultiple Selection	119
12.1	Uso del objeto generado por el pattern múltiple Selection:	119
13	Pattern K2BTrnForm	122



1 Introducción:

K2B Tools son herramientas cuyo objetivo es acelerar y facilitar el proceso de desarrollo de aplicaciones web complejas. Mediante el uso de estas herramientas se aumenta la productividad y se mejora la calidad de las aplicaciones generadas.

Estas herramientas fueron desarrolladas por el equipo técnico de k2b, en base a necesidades surgidas dentro del proyecto k2b, y otros proyectos que hicieron uso de estas herramientas. Mediante todo este feedback, ha sido posible la construcción de una suite de herramientas potentes que se han dado a llamar k2btools.

Las K2BTools son compatibles con las últimas versiones de GeneXus Ev1, y GeneXus Ev2. Están compuestas por cuatro patrones, el pattern K2BEntityServices, el pattern K2BMultipleSelection el K2BPrompt y el K2BTrnForm.

Además se han adicionado las herramientas WebPanelBuilder, para poder generar toda la interfaz web desde las k2btools.

El Pattern K2BEntityServices es el patrón principal de las k2btools, el cual se encarga de definir la arquitectura web de toda la aplicación, permitiendo integraciones con lógica de usuario, o reglas concernientes al negocio. La idea de este patrón es que los desarrolladores se enfoquen aún más en el negocio y se abstraigan de diversos problemas, entre ellos por ejemplo la interfaz. El Pattern K2BEntityServices genera para una entidad paneles que permiten el acceso a los servicios de mantenimiento de la entidad, así como la exploración con sus entidades relacionadas.

A lo largo de este documento se explicará el uso de este patrón, patrón base para las k2btools. Muchas de las propiedades son similares para los otros dos patrones, por tanto en los capítulos correspondientes al pattern K2BPrompt K2BMultipleSelection y K2BTrnForm se centrará en las diferencias.

El primer capítulo tendrá relación con las pautas previas a aplicar en la KB antes de utilizar estos patrones, los siguientes se centrarán en los diferentes objetos generados, las propiedades avanzadas, la seguridad, y los otros patrones que integran las k2btools.

2 Pautas previas:

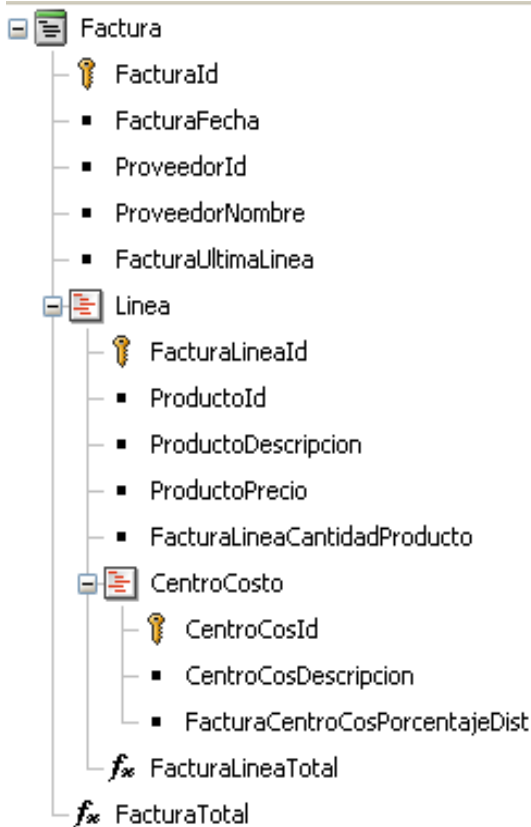


2.1 Pautas previas antes de la aplicación del Pattern K2BEntityServices:

Antes de aplicar el pattern K2BEntityServices existen ciertas pautas que hay que seguir para ayudar al usuario a que al pattern realice las inferencias correctas a la hora de generar la instancia por defecto. Muchas de estas son bastante intuitivas y por lo general se aplican, de todas maneras nunca está de más listarlas.

2.1.1 Transacciones:

- Las descripciones de las transacciones y de sus niveles deben ser descriptivos y deben estar en singular.
 - Ej: Trn001: Pais
- Setear el Description Attribute de cada transacción.
- Ordenar los atributos en las transacciones según el orden en que se desean mostrar en pantalla.
 - Las FK deben estar juntas con su descripción.
- En transacciones de varios niveles, atributos de resumen de nivel inferior, colocarlos a continuación del nivel inferior.
 - Ej: Factura total resumen de nivel Línea. Factura Línea Total resumen de nivel CentroCosto.



2.1.2 Atributos:

- Los titles y column titles de los atributos deben describir a ellos mismos.
 - Ej: Código de País : Código.
- Asignar a los atributos el tipo de control que corresponda cuando va a ser referenciado como clave foránea.

2.1.3 Determinación de instancia por defecto:

El cumplimiento de estas pautas ayudan al Pattern K2BEntityServices a la determinación de la instancia por defecto.

2.1.4 Posicionamiento de atributos en las pantallas:



El Pattern determina el posicionamiento de los atributos según el orden como han sido definidos los atributos en la transacción.

En las transacciones de varios niveles el Pattern K2BEntityServices les da a los atributos de resumen un look & feel especial además de posicionarlos por debajo de las grillas que resumen.

Ej:

Id	0	
Fecha	10/17/2007	
Proveedor	Celulosa Carro Chato SRL	
Linea		
Producto	Mouse	
Precio Unitario	15.00000	
Cantidad	23	
CentroCosto		
Id	Centro de Costo	Porcentaje de distribución
CentroCosto1		12
Total Linea		345.00
Total		345.00

El Pattern K2BTrnWebForm, al generar su instancia por defecto realiza estas mismas inferencias.

2.1.5 Determinaciones de descripciones de atributos en las pantallas:

El pattern K2BEntityServices determina la descripción de los atributos en las pantallas, utilizando el siguiente algoritmo.

- Title, en las vistas planas, para atributos propios de la tabla.
- Column title, en las vistas de grilla, para los atributos propios de la tabla base.
- Contextual title: Cuando son referenciados como foráneo.
- Excepción DA: En caso de ser un atributo DA inferido se toma la descripción del nivel de la transacción del cual el atributo es DA.

Determinación de descripción de DA:

Ejemplo (determinación descripción DA):

Se tiene el atributo ClienteNombre, en el WWCliente se mostrará como descripción Nombre, en cambio en el WWFactura se tomará como descripción la palabra Cliente.



Atributos que son subtipos:

Se toma como descripción de subtipos inferidos cuyo supertipo es DA , la descripción del grupo de subtipos al que pertenece. Ej: Si se tiene la transacción siguiente:

Ciudad
CiudadId*
CiudadNombre
PaisId
PaisNombre

Vuelo
VueloId*
CiudadOrigenId
CiudadOrigenNombre
CiudadDestinoId
CiudadDestinoNombre

Siendo
CiudadOrgienId, CiudadOrigenNombre subtipos de CiudadId y CiudadNombre.
Y CiudadDestinoId y CiudadDestinoNombre subtipos de CiudadId y CiudadNombre.

A la hora de determinar la descripción a tomar en el WWVuelos, correspondiente a CiudadOrgenNombre, o CiudadDestinoNombre, se irá a la descripción del grupo de subtipos, para mostrar de esta forma los títulos CiudadOrigen, CiudadDestino.

2.1.6 Determinación de control info asociado a los atributos.

El Pattern K2BEntityServices le asocia el tipo de control correspondiente a los atributos que aparecen en pantalla de la siguiente forma.

- **Atributos dentro de la grilla:** Se les fuerza el tipo de control a edit. La idea es que cuando se referencia a un atributo se muestre ese atributo y no el atributo inferido a partir de su control info.
- **Determinación del web form por defecto:**
 - **Atributos Claves:** El tipo de control será forzado a edit.



- **Atributos foráneos:** Se toma el tipo de control del atributo.
- **Atributos inferidos mostrados ya visualizados por otro control info:** Esto es para que no sea mostrado dos veces en la pantalla.

2.1.7 Generación de filtros por defecto:

Por defecto el pattern genera filtros por:

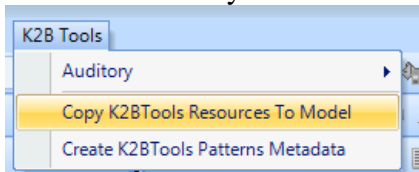
- Description Attribute.
- Description Attribute de los atributos foráneos no instanciados.
 - Por no instanciado se refiere por ejemplo: Filtro de PaisNombre en Ciudades de un Pais.
- Fechas desde y hasta por cada atributo.
 - En el archivo K2BEntityServices.config se puede configurar si se quiere que por defecto se generen los filtros por fecha. Template/Filters : GenerateDateFilters, GenerateDateTimeFilters.

2.2 Pautas antes de aplicar la instancia:

- Verificar que las propiedades caption y description de los WorkWith estén en plural.
- Verificar en cada subworkwith, la propiedad Description este en plural.

2.3 Pautas antes de ejecutar la aplicación:

Antes de ejecutar la aplicación, una vez que está el modelo configurado se debe ir al menú K2BTools y seleccionar Copy K2BTools Pattern Resources.



Esto lo que hace es copiar los objetos básicos de las k2btools al directorio del modelo de ejecución seleccionado.



3 Actualización de transacciones:

La transacción es el único objeto del pattern K2BEntityServices que puede ser actualizado tanto desde Patterns como de GeneXus.

Mediante la propiedad update transaction ubicada en el nodo transaction, se indica de que manera el pattern va a actualizar la transacción. Como predeterminado se tiene la propiedad update transaction en Apply K2BWWStyle. Esta es la forma normal de trabajar y esta primera parte consistirá en explicar cómo es el comportamiento con update transaction en ese valor.

Para complementar el pattern K2BEntityServices, se cuenta con el pattern K2BTrnForm, que permite mantener desde una instancia el web form de la transacción. Veremos en esta sección la convivencia de este patrón con el pattern k2benttiyservices. Detalles específicos del funcionamiento del patrón TrnForm se pueden ver en el capítulo 13.

3.1 Update Transaction = Apply K2BWWStyle:

3.1.1 Actualización de web form:

El Pattern K2BEntityServices lo único que hace es actualizar el contorno del web form de la transacción, el resto el área de datos puede ser actualizada utilizando el patrón k2btrnform. En caso de que no se desee utilizar este patrón también puede ser actualizada a mano.



El Web Form de la transacción va a tener una sección a la que llamaremos área de datos. En este lugar el usuario puede ubicar todos los atributos y hacer las modificaciones que el desee en esa área, sin que el pattern k2bentityservices a la hora de volver a ser aplicado le sobrescriba los cambios realizados.

The screenshot shows a web form titled "Errorviewer: K2BErrViewer". It contains a section labeled "DataAreaBegin" and "DataAreaEnd". Inside this section, there are several input fields and checkboxes:

- Código**: Input field with value "ProductoCodigo".
- Descripcion**: Input field with value "ProductoDescripcion".
- Código**: Input field with value "ProductoClaseCodigo" and a dropdown arrow.
- Compuesto**: Checkbox, currently unchecked.
- Es Stockeable**: Checkbox, currently unchecked.
- Observaciones**: A large text area with the label "ProductoObservaciones" and a vertical scrollbar.

At the bottom of the form, there are two buttons: "Confirmar" and "Cancelar".

El Data Area está delimitado por la primera fila de la tabla

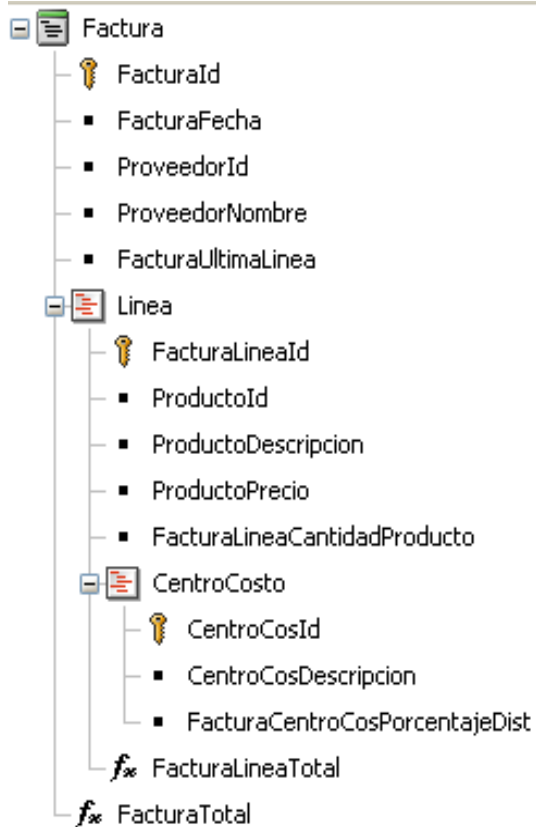
K2BTableTrnDataAreaContainer

y la última fila de esta tabla, según como se visualiza en la figura.

El Pattern K2BEntityServices a la hora de generar el web form coloca los atributos en el orden que fueron definidos en su estructura, asociando el tipo de control especificado en la KB para los atributos foráneos.

Para transacciones de varios niveles el Pattern K2BEntityServices genera grillas para niveles hojas y freestylegrids para los niveles intermedios. El nivel superior siempre es plano.

Ej: Sea la transacción factura.



El primer nivel será plano el segundo un Freestyle grid y el tercero un grid.
La ubicación del DataArea es similar a la ubicación del DataArea en transacciones de un nivel.

DataAreaBegin



Id	Factu
Fecha	FacturaFecha
Proveedor	ProveedorId


Linea												
<input type="checkbox"/>	<table border="1"> <tr> <td>Producto</td> <td>ProductoId</td> </tr> <tr> <td>Precio Unitario</td> <td>ProductoPrecio</td> </tr> <tr> <td>Cantidad</td> <td>Factur</td> </tr> </table>	Producto	ProductoId	Precio Unitario	ProductoPrecio	Cantidad	Factur					
Producto	ProductoId											
Precio Unitario	ProductoPrecio											
Cantidad	Factur											
<table border="1"> <tr> <th colspan="3">CentroCosto</th> </tr> <tr> <th></th> <th>Id</th> <th>Centro de Costo</th> <th>Porcentaje de distribución</th> </tr> <tr> <td><input type="checkbox"/></td> <td>CentroCosId</td> <td>CentroCosDescripcion</td> <td>FacturaCentroCosPorcentaje</td> </tr> </table>		CentroCosto				Id	Centro de Costo	Porcentaje de distribución	<input type="checkbox"/>	CentroCosId	CentroCosDescripcion	FacturaCentroCosPorcentaje
CentroCosto												
	Id	Centro de Costo	Porcentaje de distribución									
<input type="checkbox"/>	CentroCosId	CentroCosDescripcion	FacturaCentroCosPorcentaje									
<table border="1"> <tr> <td>Total Linea</td> <td>FacturaLineaTotal</td> </tr> </table>		Total Linea	FacturaLineaTotal									
Total Linea	FacturaLineaTotal											

Total	FacturaTotal
--------------	--------------


El Pattern K2BEntityServices asume que los atributos de un nivel superior ubicados a continuación de un nivel inferior son atributos de resumen de ese nivel y los genera con un look & feel específico para esto.



En el ejemplo de la factura el Pattern infiere que el atributo FacturaTotal es un resumen del nivel Línea y lo coloca al finalizar la línea. También asume que el atributo FacturaLineaTotal es un resumen del nivel Centro de Costo y lo coloca al final de este. En ejecución:

Id 0
Fecha 10/03/2007 
Proveedor Forestal Sierra de Minas S.A. 

Linea 


☐ **Producto** Mouse
Precio Unitario 15,0
Cantidad 2


CentroCosto 

	Id	Centro de Costo	Porcentaje de distribución
<input type="checkbox"/>	CentroCosto1 	CentroCosto1	31
<input type="checkbox"/>	CentroCosto4 	CentroCosto4	12

Total Linea 30,00

☐ **Producto** Gas-Oil
Precio Unitario 20,000000
Cantidad 0

CentroCosto 

	Id	Centro de Costo	Porcentaje de distribución
<input type="checkbox"/>	CentroCosto1 		0

Total Linea 0,00

Total 30,00

La idea en cuanto a la generación del web form de la transacción por el Pattern K2BEntityServices es que el pattern genera un web form por defecto, y luego el usuario debe modificar el área de datos aplicando el pattern K2BTrnForm o a mano.

3.1.2 Actualización de código:



El Pattern K2BEntityServices genera el código en determinadas secciones llamadas slots, a continuación pasaremos a describir qué son los slots.

3.1.2.1 Slot:

Nomeclatura: `//+ <propietario>.<funcionalidad>[.Fixed]`

El `.Fixed` es opcional e identifica que ese slot es fijo y que no puede ser modificado por el usuario.

El pattern al aplicarse va a generar slots fixed, y no fixed. Al aplicarse el patrón, sobre una transacción pasa lo siguiente según la región en la que está ubicada el código:

- **Código en slots de tipo fixed:** Se vuelve a generar completamente. Si el usuario realiza modificaciones dentro de este slot, las perderá.
- **Código en slots no fixed:** Si no aparecen en el código el pattern los vuelve a generar, si ya estaban en el código el pattern no lo genera de nuevo.
- **Código por fuera de los slots:** El código por fuera de los slots no es sobrescrito por el pattern. Todo lo que el usuario escriba por fuera de estos slots, no lo pierde.

3.1.2.2 Actualización de reglas:

El Pattern K2BEntityService genera las siguientes reglas,

Slot: Parameters

```
//+ K2BES.Parameters
[ Parm(in:&Mode, in:&FacturaId);
  FacturaId = &FacturaId If not &FacturaId.IsEmpty();
  NoPrompt(FacturaId);
//-
```

Esta sección es generada por defecto y el usuario la puede modificar cuando quiera. En ella se encuentra la regla parm que por defecto se genera con el modo y la clave primaria de la transacción, y setea el valor del atributo si este está instanciado. Dado que no es posible tener dos reglas parms en la transacción el pattern también busca una regla parm por fuera de esta sección y de encontrarla la comenta.



Slot fixed para el manejo de la clase del error viewer.

```
//+ K2BES.ErrorViewer.Fixed
[web]
{
    K2BErrViewer.Class = K2BThemeClasses.K2BMessage on AfterInsert, AfterUpdate;
    K2BErrViewer.Class = K2BThemeClasses.K2BWarning If Delete;
}
//-
```

En caso de mostrar un error, coloca un formato diferente si el error se da al eliminar.

Slot para el manejo de la clave, hace que la clave no pueda ser seteada, o sea para el caso de autoenumerados el usuario nunca debe setear la clave,.

```
//+ K2BES.Key
[web]
{
    NoAccept (FacturaId);
}
//-
```

También hay un slot correspondiente a la instanciación automática de la clave foránea, esto lo veremos después.

Para las transacciones de varios niveles genera reglas

```
//+ K2BES.GridRowsSettings
[web]
{
    FreeStyleGridLinea.Rows = 1 If &Mode = TrnMode.Insert;
    FreeStyleGridLinea.Rows = 0 If &Mode = TrnMode.Update Or &Mode = TrnMode.Delete;
    GridCentroCosto.Rows = 1 If &Mode = TrnMode.Insert;
    GridCentroCosto.Rows = 0 If &Mode = TrnMode.Update Or &Mode = TrnMode.Delete;
}
//-
```

Lo que hace esto es que si el modo es insert ya agrega la primera línea en la grilla de cada uno de los niveles. Si es update o delete deja simplemente las que tienen que estar. Esto lo hace para cada grilla correspondiente a cada subnivel de la transacción.

3.1.2.3 Actualización de eventos:

El Pattern K2BEntityServices genera el siguiente código para los eventos:

```
//+ K2BES.Start.Fixed
[web]
{
    K2BGetContext(&Context)
    &BtnCaption = K2BGetButtonCaption.Udp(&Mode)
    Enter.Caption = &BtnCaption
    Enter.Visible = False
    Cancel.Visible = False
    If &Mode <> K2BTrnMode.Display
        Enter.Visible = True
        Cancel.Visible= True
    EndIf
}
//-
```

Esto es para el ocultamiento de los botones de aceptar y cancelar cuando la transacción es invocada en modo display, además de la obtención de los captions de los botones según el modo.



```
Event After Trn
//+ K2BES.AfterTrn.Fixed
[web]
{
    K2BGetTrnContext.Call(&TrnContext)
    If &TrnContext.TransactionName = &Pgmmname
        Do Case
            Case &Mode = K2BTrnMode.Insert
                &Navigation = &TrnContext.AfterInsert
                Do 'DoAfterTrnAction'
            Case &Mode = K2BTrnMode.Update
                &Navigation = &TrnContext.AfterUpdate
                Do 'DoAfterTrnAction'
            Case &Mode = K2BTrnMode.Delete
                &Navigation = &TrnContext.AfterDelete
                Do 'DoAfterTrnAction'
        EndCase
    EndIf
}
//-
```

Generación en el alter trn de código fijo , para configurar la navegación que va a tener la transacción luego de que esta ejecuta la acción.

```
//+ K2BES.EventCancel
[web]
{
    Event 'DoCancel'
        K2BGetTrnContext.Call(&TrnContext)
        Do 'K2BClose'
    EndEvent
}
//-
```

Generación en slot de usuario el evento cancel por si el usuario desea modificarlo para poder acceder a otro panel o pantalla.

3.1.3 Instanciación automática de la clave foránea:

Supongamos que tenemos el ejemplo de facturas y clientes, y se quiere dar de alta una factura en el tab facturas de un cliente, cuando se invoca a la transacción de facturas, se va a querer que el cliente ya este instanciado. Esto presupondría modificar la regla parm,



lo cual implicaría modificar la invocación de todos los objetos que llaman a la transacción.

Sin embargo existe otra forma de tener esto instanciado automáticamente sin tener que hacer modificaciones en la regla parm. El Pattern K2BEntityServices hace uso de la web session, en el panel invocador guarda aquellos atributos que están instanciados, y estos son leídos en la transacción.

Ej: Se tienen las facturas del proveedor, antes de invocar a la transacción de factura se ejecuta la subrutina preparetransaction que se encarga de almacenar en un sdt los atributos que ya están instanciados. En este caso almacena el proveedor en un sdt de nombre TrnContext. (ver código en rojo).

```
Sub 'PrepareTransaction'
    &TrnContext = new()
    &TrnContext.TransactionName = !"Factura"
    &TrnContext CallerUrl = &HTTPRequest.ScriptName + !"?" + &HTTPRequest.QueryString
    &TrnContext.ReturnMode = K2BTrnReturnMode.Stack
    &TrnContext.AfterInsert = new K2BTrnNavigation()
    &TrnContext.AfterInsert.AfterTrn = K2BAfterTrnNavigation.EntityManger
    &TrnContext.AfterUpdate = new K2BTrnNavigation()
    &TrnContext.AfterUpdate.AfterTrn = K2BAfterTrnNavigation.ReturnToCaller
    &TrnContext.AfterDelete = new K2BTrnNavigation()
    &TrnContext.AfterDelete.AfterTrn = K2BAfterTrnNavigation.ReturnToCaller
    &TrnContextAtt = new()
    &TrnContextAtt.AttributeName = !"ProveedorId"
    &TrnContextAtt.AttributeValue = &ProveedorId.ToString()
    &TrnContext.Attributes.Add(&TrnContextAtt)
    K2BSetTrnContext.Call(&TrnContext)
EndSub
```

K2BTrnContext		K2 BTrn Context	
TransactionName	ObjectName	Transaction Name	<input type="checkbox"/>
CallerUrl	URLString	Caller Url	<input type="checkbox"/>
EntityManagerName	ObjectName	Entity Manager Name	<input type="checkbox"/>
EntityManagerNextTaskCode	K2BTabCode	Entity Manager Next Task Code	<input type="checkbox"/>
EntityManagerNextTaskMode	K2BTrnMode	Entity Manager Next Task Mode	<input type="checkbox"/>
ReturnMode	K2BTrnReturnMode	Return Mode	<input type="checkbox"/>
AfterInsert	K2BTrnNavigation	After Insert	<input type="checkbox"/>
AfterUpdate	K2BTrnNavigation	After Update	<input type="checkbox"/>
AfterDelete	K2BTrnNavigation	After Delete	<input type="checkbox"/>
Attributes		Attributes	<input checked="" type="checkbox"/>
Attribute		Attribute	
Attribute Name	VarChar(128)	Attribute Name	<input type="checkbox"/>
Attribute Value	VarChar(1000)	Attribute Value	<input type="checkbox"/>

Se guarda el nombre de la transacción que se va a invocar, y los nombres y valores de atributos instanciados.

Para esto en el evento Start de la transacción factura el pattern K2BEntityServices genera un slot fixed



```
//+ K2BES.FK.Fixed
[web]
{
  If &Mode = K2BTrnMode.Insert
    K2BGetTrnContext.Call(&TrnContext)
    If &TrnContext.TransactionName = &Pgmname
      For &TrnContextAtt in &TrnContext.Attributes
        Do Case
          Case &TrnContextAtt.AttributeName = !"ProveedorId"
            &Insert_ProveedorId.FromString(&TrnContextAtt.AttributeValue)
          EndCase
        EndFor
      EndIf
    EndIf
  EndIf
}
//-
```

Lo primero que hace es preguntar si el nombre de la transacción que está guardado en la sesión es el mismo que el nombre del programa, en caso de serlo recorre todos los atributos y hace un case con todos los atributos foráneos de la factura. En este caso el único foráneo es el proveedor pero podrían ser más.

Luego en las reglas se genera otro slot para igualar el atributo con la variable obtenida de la sesión

```
//+ K2BES.FK.Fixed
  ProveedorId = &Insert_ProveedorId if &Mode = K2BTrnMode.Insert and not &Insert_ProveedorId.IsEmpty();
noaccept(ProveedorId) if &Mode = K2BTrnMode.Insert and not &Insert_ProveedorId.IsEmpty();
//-
```

En muchos casos la instanciación de las claves foráneas ya se tiene resuelto modificando la regla parm, para esto es posible deshabilitar la generación de estos slots, desde el nodo transaction de la instancia yendo a la propiedad GenerateSlotsForFK. El valor <default> lo lee del archivo de settings. En la siguiente sección veremos bien como funciona esto.

3.2 Propiedades en la instancia sobre la transacción:

Estas propiedades se encuentran en el nodo transaction.

3.2.1 Update Transaction:

Indica cómo se va a actualizar la transacción. Hasta ahora se vio su comportamiento con el valor Apply K2BWStyle

- Do Not Update: No actualiza la transacción.
- Only rules and events: Actualiza solo las reglas y los eventos. El web form lo deja como lo puso el usuario.



- Create Default: Vuelve a generar todas las secciones default de nuevo. Esto es: vuelve a generar el web form completo, todas las secciones de código tanto fixed como por defecto.

3.2.2 GenerateSlotsForFK:

Se puede configurar aquí si se va a generar el slot para la instanciación automática de las claves foráneas.

3.2.3 Navigation:

Se especifica la navegación dentro del pattern luego de hacer las diferentes operaciones sobre la transacción.

Estas son After<Operation>:

- Return to caller : Vuelve al web panel llamador.
- EntityManager: Va al siguiente tab del entity manager.
- None: No realiza ninguna acción.

AfterInsert: {Return to caller, EntityManager, None}

AfterUpdate: {Return to caller, EntityManager, None}

AfterDelete: {Return to caller}

Todas estas propiedades poseen el valor <default>, esto lo toma del config. Para esto existe en el K2BEntityServices.config del nodo transaction, de donde se toman los valores por defecto.

3.3 Flexibilidad en la invocación de la transacción

La transacción es generada como web component. Esto permite poder incluirla en cualquier objeto generado. Dentro de patterns, la transacción en modo display es mostrado en el tab general del entity manager.

La idea de tener la transacción como componente apunta a poder reutilizarla sin problemas en cualquier interfaz. Además de esto, brinda la flexibilidad que de ser invocada en varios contextos, incluso no solo desde el contexto de patterns, es posible modificar la navegación sin modificar el código de la transacción.

Esta navegación es tan flexible que es posible indicar que luego de realizar una acción vaya al objeto que el usuario desee simplemente seteando el TrnContext en la web session dentro del objeto llamador.



Explicaremos aquí como funciona esto:

El Objeto K2BTrnContext tendrá las siguientes propiedades:

K2BTrnContext		K2 BTrn Context	
TransactionName	ObjectName	Transaction Name	<input type="checkbox"/>
CallerUrl	URLString	Caller Url	<input type="checkbox"/>
EntityManagerName	ObjectName	Entity Manager Name	<input type="checkbox"/>
EntityManagerNextTaskCode	K2BTabCode	Entity Manager Next Task Code	<input type="checkbox"/>
EntityManagerNextTaskMode	K2BTrnReturnMode	Entity Manager Next Task M...	<input type="checkbox"/>
ReturnMode	K2BTrnReturnMode	Return Mode	<input type="checkbox"/>
AfterInsert	K2BTrnNavigation	After Insert	<input type="checkbox"/>
AfterUpdate	K2BTrnNavigation	After Update	<input type="checkbox"/>
AfterDelete	K2BTrnNavigation	After Delete	<input type="checkbox"/>
Attributes		Attributes	<input checked="" type="checkbox"/>

EntityManagerName: Nombre del entity manager donde se ejecuta la transacción

EntityManagerNextTaskCode: Se setea cuál es la próxima tarea que se va a ejecutar en el entity manager, en el caso actual la próxima tarea será el próximo tab.

EntityManagerNextTaskMode: Se setea cuál será el modo con el que se invocará al entity manager.

Luego se encuentran con las propiedades AfterInsert, AfterUpdate y AfterDelete que pertenecen al SDT K2BTrnNavigation

Ahí se le indica al pattern que acción va a realizar luego de hacer un insert, update o delete.

K2BTrnNavigation		K2 BTrn Navigation	
AfterTrn	K2BAfterTrnNavigation	After Trn	
ObjectToLink	ObjectName	Object To Link	
Mode	Character(3)	Mode	
ExtraParameter	Character(20)	Extra Parameter	

La propiedad AfterTrn permite indicar como va a ser la navegación.

Los valores pueden ser:

- ReturnToCaller: Se va al stack o al callerUrl según corresponda.
- EntityManager: Se invoca al entity manager que se encuentra en la propiedad entitymanagername, pasándole como modo, EntityManagerNextTaskMode, y pasándole como tab EntityManagerNextTaskCode.
- DynamicLink: Permite que la transacción pueda navegar hacia cualquier objeto determinado por los siguientes parámetros.
 - ObjectToLink: Objeto a invocar luego de ejecutar la transacción.
 - Mode: Si es vacío se usará el modo con el que se invocó a la transacción.



- ExtraParameter: Valor del parámetro extra con el que será invocado el objeto.
- La invocación será: Link(ObjectToLink, PKTransacción, ExtraParameter)

3.4 Propiedades avanzadas para la transacción:

En el K2BEntityServices.config:

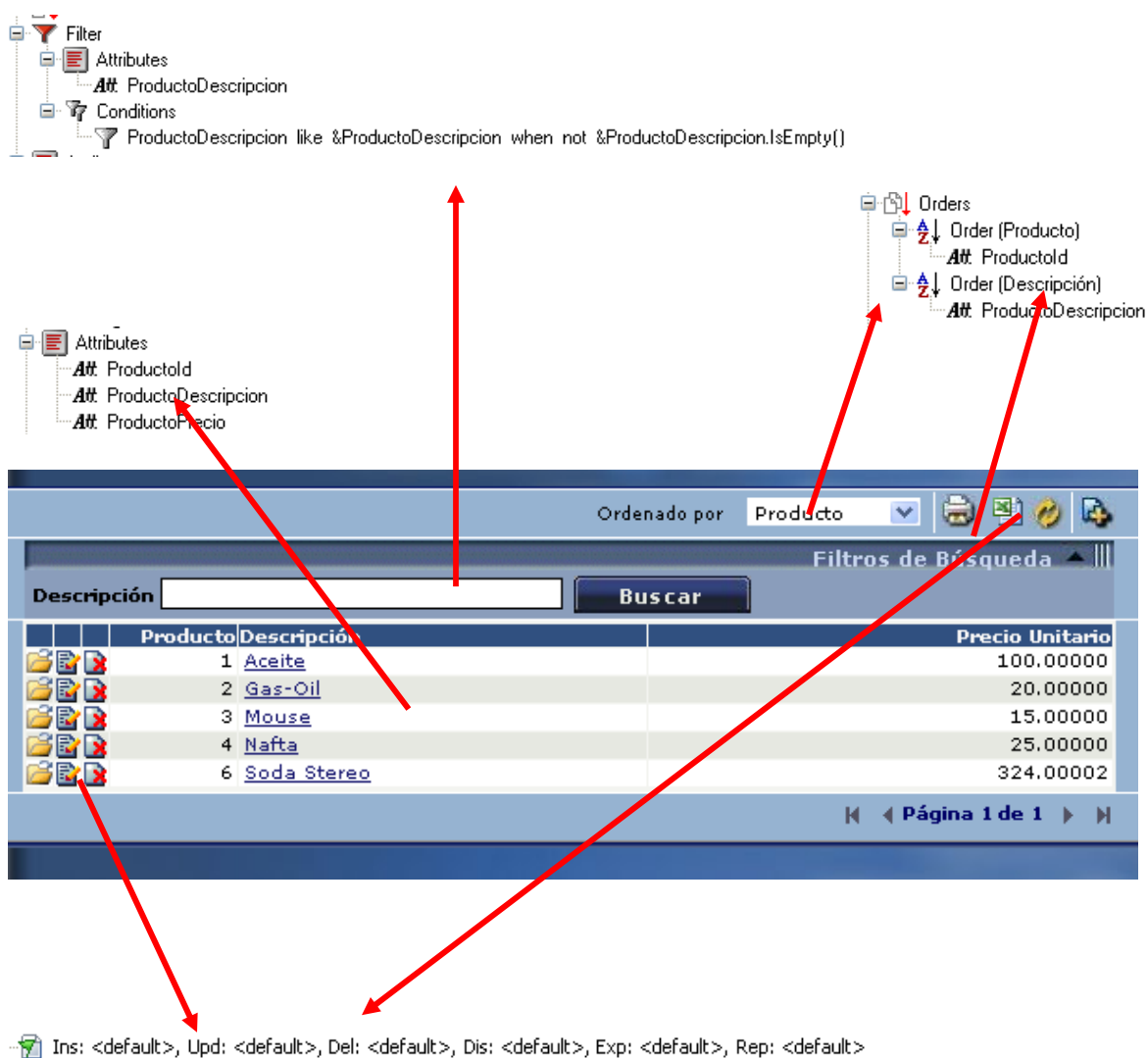
GenerateGetContextForBC: Si el código que invoca al programa K2BGetContext se genera dentro de un tag web o no. Esto es para el caso en que se desee utilizar el context dentro de los bc.

GenerateCheckButton: Indica si a la hora de generar la transacción se va a generar el

Los slots que están comentados son para tener una clave generada sugerida, y el último para tener una clave manual.

4 Objetos con Grilla

El Pattern K2BEntityServices genera objetos con grilla en los tabs de tipo grid y en el workwith. Veremos aquí las propiedades más importantes que podemos encontrar en estos objetos.



The screenshot displays a web application interface for managing products. At the top, there's a navigation pane with a tree structure. The main area shows a grid of products with columns for 'Producto', 'Descripción', and 'Precio Unitario'. The grid contains five rows of data. Below the grid, there's a status bar with various action buttons and a page indicator.

Red arrows indicate the following components:

- Filter:** A tree structure on the left showing a filter for 'ProductoDescripcion' with a condition 'ProductoDescripcion like &ProductoDescripcion when not &ProductoDescripcion.IsEmpty()'.
- Attributes:** A tree structure on the left showing attributes for 'ProductoId', 'ProductoDescripcion', and 'ProductoPrecio'.
- Orders:** A tree structure on the right showing orders for 'Order (Producto)', 'Order (Descripción)', and 'Order (Precio)'.
- Grid:** The main data table with columns 'Producto', 'Descripción', and 'Precio Unitario'. The data rows are:

Producto	Descripción	Precio Unitario
1	Aceite	100.00000
2	Gas-Oil	20.00000
3	Mouse	15.00000
4	Nafta	25.00000
6	Soda Stereo	324.00002
- Status Bar:** A bar at the bottom with buttons for 'Ins', 'Upd', 'Del', 'Dis', 'Exp', and 'Rep', each with a default value.



Estudiaremos cada uno de los componentes de los objetos con grilla:

Estos son los componentes comunes:

- Filtros
- Órdenes
- Atributos
- Acciones
- Modos

En un capítulo posterior estudiaremos componentes avanzados que también se pueden agregar a un objeto con grilla.

4.1 Filtros y condiciones:

Nodo Filters and conditions.

The screenshot shows the K2B Tools interface. On the left, a tree view displays the structure of a project, including a 'Filter' node under 'ProductoDescripcion'. A red arrow points from this 'Filter' node to the 'Filter Attribute' section in the properties panel on the right. The properties panel shows various settings for the filter, including Name, Description, BasedOn, Form, Values, and State.

Filter Attribute	
Name	ProductoDescripcion
Description	Descripción
BasedOn	(none)
Form	
NoSkip	False
Visible	True
ReadOnly	False
ThemeClass	
DescriptionThemeCla:	
Values	
Default	
All	False
Prompt	
State	
SaveState	True

El nodo filtros permite al usuario definir filtros.

Estos filtros se pueden definir basados en atributos o basados en algún dominio.

Patterns infiere el tipo de datos del filtro según el mecanismo genérico para la definición de variables que se explica a continuación.

4.1.1 Mecanismo genérico para la definición de variables:

Esto es válido para todas las secciones donde se permiten ingresar variables, esto es:



- Nodo Filters
- Nodo Variables (se verá en capítulos posteriores)
- Variables en la grilla.
- Componentes de la grilla (se verá en capítulos posteriores)

Patterns infiere el tipo de datos de la variable de la siguiente manera:

- El atributo se define según la propiedad basedon donde se puede basar la variable en un atributo o dominio.
- Si no está seteada la propiedad basedon, si el filtro tiene el mismo nombre que el atributo, queda definida basada en ese atributo.

4.1.2 Propiedades asociadas a los filtros:

SaveState:

El pattern automáticamente para cada filtro salva el estado, esto es si el usuario en ejecución define determinados filtros, y luego navega por otras pantallas, cuando vuelva al objeto con grilla, los filtros van a seguir estando con el último valor seteado. Esto se puede modificar, utilizando la propiedad savestate asociado al filtro, su valor <default> o true indica que se salva el estado correspondiente a ese filtro, seteando en false se puede configurar para no salvar el estado correspondiente a dicho filtro.

ReadOnly:

Permite definir filtros readonly.

NoSkip: Se verá más adelante.

ThemeClass: Clase de tema que tomará el filtro. En caso de no especificarlo toma por defecto la clase que se encuentra en Theme/[WorkWith|SubWorkWith], según el tipo de datos. Si es fecha, tomará el valor de la propiedad DataAttributeInFilter, si es date time tomará el valor de la propiedad DateTimeAttributeInFilter, sino tomará el valor de la propiedad AttributeInFilter.

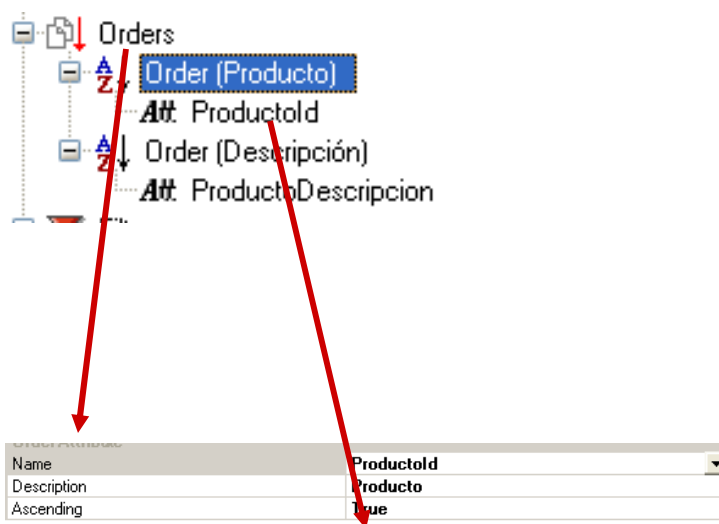
Default: Expresión a ejecutarse en el evento start.

Conditions:

Permite al usuario ingresar condiciones en el grid, que pueden o no estar asociados a algún filtro. Ej: Filtros por Empresa obtenido del contexto (EmpresaId = &Context.Empresa)

4.2 Órdenes:

El nodo órdenes permite definir los diferentes órdenes que se pueden seleccionar para visualizar la grilla.



Cada orden esta compuesto por un conjunto de atributos, para cada atributo se puede setear si el orden es ascendente o descendente utilizando la propiedad ascending.

4.3 Atributos:

Es el nodo principal de las vistas con grilla. Indica que atributos se mostrarán dentro de la grilla.



- attribute: ProductId	
- Definition	
Attribute	ProductId
Description	Id
- Form	
Visible	False
LeftText	
RightText	
ThemeClass	
DescriptionThemeCla:	
NoSkip	False
Format	<default>
- Link	
Autolink	True

Las propiedades que tienen son nombre, la descripción que es lo que se mostrará en el título de la columna.

El atributo puede ser visible o no, en caso de ser invisible se coloca como hidden dentro de la grilla.

ThemeClass:

Indica la clase del tema que tendrá el atributo por defecto toma la clase del tema ubicada en Theme/[WorkWith|SubWorkWith], dependiendo del tipo de control, si es un combo tomará el valor de ComboBoxInGrid, si es un checkbox tomará el valor de CheckBoxInGrid, y sino tomará el valor de AttributeInGrid.

Autolink:

En true, si el atributo es DA genera el link dirigido al entitymanager correspondiente a la instancia.

4.3.1 Variables:

En el nodo attributes es posible también ingresar variables. Para esto parado en el nodo attributes se debe hacer add -> Variables.



variable: ProductPrice	
Definition	
Name	ProductPrice
Description	
BasedOn	ProductPrice
Form	
Visible	True
ReadOnly	True
ThemeClass	
DescriptionThemeClass	
NoSkip	False
Format	<default>
Value	
Value	
ContextualHelp	
ContextualHelpValue	

La definición de variables está explicada en la sección [Mecanismo genérico para la definición de variables](#):

ReadOnly:

Por defecto todas las variables son de solo lectura. Se puede modificar esta propiedad para que el usuario puede ingresarle valor a esta variable.

Value:

Lo que se coloque en value será generado en el evento load de la grilla.

En particular se generará &Variable = <Value>

4.3.2 Seteo de propiedades de los atributos:

Autoresize: Por defecto el Pattern K2BEntityServices para una correcta visualización de la grilla, coloca el primer atributo visible de la grilla con autoresize en false, y el resto de los atributos los coloca con autoresize en true.

La propiedad autoresize del primer atributo visible de la grilla es seteada en false, cuando el largo de su tipo de datos es menor o igual que la propiedad dentro del archivo de configuración Grid/FirstColumnResizeValue.



4.4 Modos y Acciones:

En los objetos con grilla, el Pattern K2BEntityServices permite realizar las siguientes acciones:

- Insert.
- Update
- Delete
- Display
- Export

Estas son las acciones por defecto que genera el pattern en todos sus objetos.

Estas acciones se configuran en el nodo `modes`,



donde se puede indicar a nivel de instancia o de archivo de configuración, cuales de estas operaciones se pueden hacer sobre la instancia. El valor default es leído del archivo de configuración en la propiedad `StandardAction/<Mode>/DefaultMode` ej: `StandardAction/Update/DefaultMode`.

La ubicación por defecto de las acciones es:

Las acciones de update, delete y display se encuentran en la grilla, las acciones de Export se encuentran en Top, y la acción Insert en Top(Right).

4.4.1 Acciones de usuario:

El usuario puede definir sus propias acciones que invoquen a sus propios objetos. Esto lo puede hacer agregando elementos dentro del nodo `actions`. Para esto hay que indicar en la propiedad `GXObject` el objeto al que se va a invocar y agregar en `parameters`, los parámetros con el que se invocará a dicho objeto.

Filter	
[-] action: Action (ChangeCustomerState)	
[-] General	
Name	ChangeCustomerState
Tooltip	Change Customer State
GXObject	ChangeCustomersState
Condition	
ActivityName	
Layout	Bottom
Caption	Change State
[-] Image	
Image	(none)
DisabledImage	(none)
ImageClass	
Target	Standard
+ Button	
[-] MultiRow	
RowSelection	Multiple
ErrorNotSelectedLine	True
ErrorMessage	<default>
[-] Windows	
UseWindowsDataTyp	False
[-] ConfirmPopUp	

Aquí veremos paso por caso cada una de las propiedades.

Layout: Indica la ubicación de las acciones. Las opciones de Layout son las siguientes:

- Top
- Top (Right)
- Top(Left)
- Ingrid
- Bottom



Caption: Solo para los botones. Indica el caption que se va a mostrar correspondiente al botón.

ToolTip: Tool tip asociado al control de la acción.

Image: Nombre de la imagen correspondiente a la acción. Si se setea una imagen entonces el control de la acción será una imagen, en caso contrario se generará un botón.

ImageClass: Si se desea tener una imagen

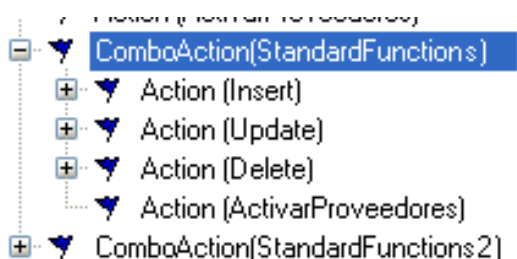
DisableImage: En caso de tener una condition, nombre de la imagen en caso de no cumplirse con la condición.

ButtonClass: En caso de ser un botón se le puede asociar una clase del tema.

Además de estas acciones, para administrar mejor la pantalla existe lo que se llama el ComboAction, que da la posibilidad de agrupar los distintos tipos de acciones dentro de un combo.



Esto se puede hacer con un combo action, especificando adentro cada una de las acciones que va a poseer.

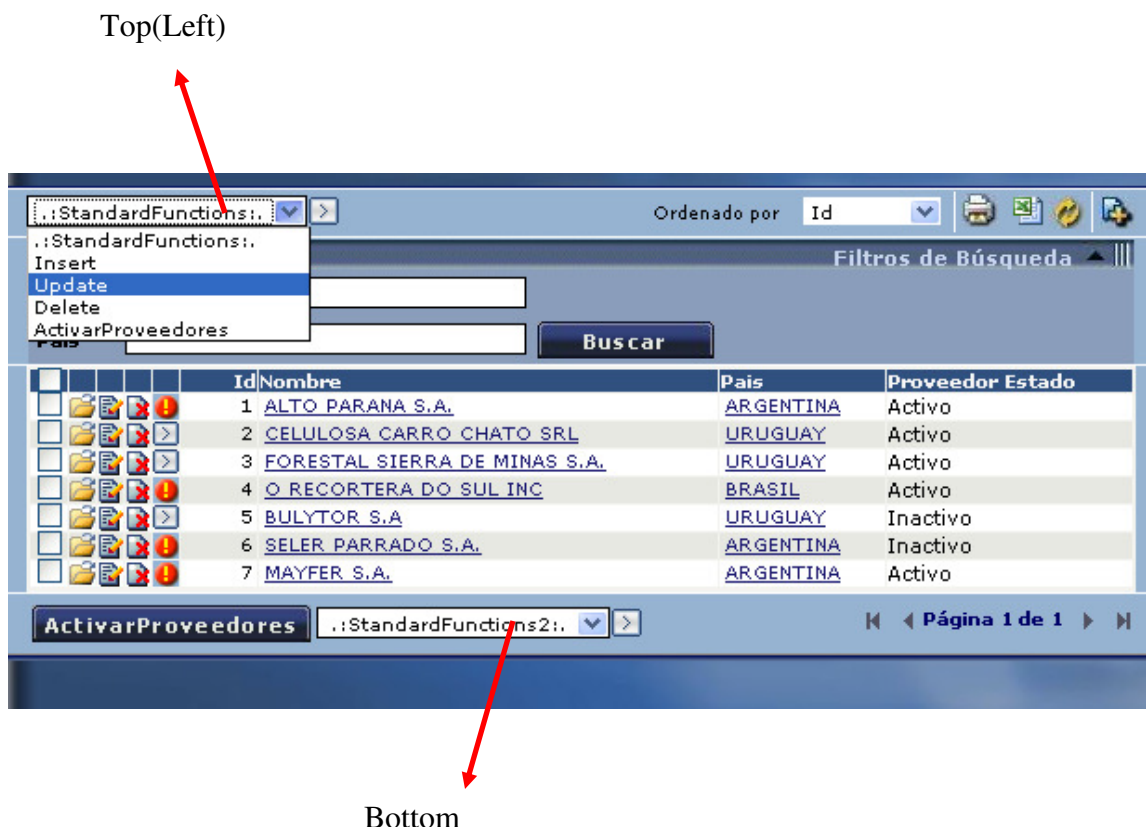


Ahí se especifican las combo actions con cada una de las acciones que van a tener.

comboAction: ComboAction (StandardFunctions)	
Definition	
Name	StandardFunctions
Form	
Caption	PedroStandardFunctions
Layout	Bottom

Las propiedades que puede poseer son nombre, caption y layout.
Caption; Es el nombre que aparece al comienzo del combo.

Los Layouts que posee con Top(Left) y Bottom. En la figura se especifica donde se encuentran estos layouts.



4.4.2 Propiedad RowSelection:

Permite agregar una columna con un checkbox a la izquierda de la grilla, donde se le permite seleccionar al usuario las diferentes filas.

- None: La acción no tiene row selection definido.
- Single: Se permite seleccionar una sola fila en caso contrario error. (solo para acciones combo).
- Multiple: Se permiten seleccionar múltiples filas.

Para las acciones múltiples la acción es invocada con los parámetros que se le indican a la acción, más un parámetro implícito que es un SDT con las líneas seleccionadas. Este SDT es generado por defecto por el pattern, y deberá ser recibido como primer parámetro



de la acción. El nombre de este SDT será <NombredelObjeto>+SDT. Y será de tipo collection.

En el caso del proveedor se llamará WWProveedorSDT, en la siguiente figura se muestra una vista de ese SDT.

WWProveedorSDT		True	WWProveedorSDTItem
■ ProveedorId	ProveedorId		
■ ProveedorNombre	ProveedorNombre		
■ PaisId	PaisId		
■ PaisNombre	PaisNombre		
■ ProveedorEstado	ProveedorEstado		

El programa generado por patterns invocará a dicho objeto con el sdt cargado de las filas seleccionadas.

La aplicación del Pattern, da un error si el GXObject no existe, sin embargo para acciones con RowSelection = Multiple, muchas veces es necesario tener el SDT con las filas seleccionadas para crear el objeto. Para esto es recomendable poner un objeto cualquiera en GXObject, aplicar para que cree el SDT y luego programar el objeto correspondiente a la acción en GX, y finalmente poner en la instancia en GXObject el objeto real.

4.4.3 Algunas otras propiedades:

ErrorNotSelectedLines: Si se va a mostrar algún mensaje de error en caso de que no se seleccione ninguna línea.

ErrorMessage: Mensaje de error en caso de que no se seleccione ninguna línea.

Esto se puede ver en el archivo de configuración

UserDefinedActions/ErrorNoneRowsSelected.

Para las acciones con RowSelection =Single se dispara un error en caso de que más de una línea haya sido seleccionada. El mensaje de error se puede modificar en el archivo de configuración UserDefinedActions/ErrorMoreThanOneRowSelected.

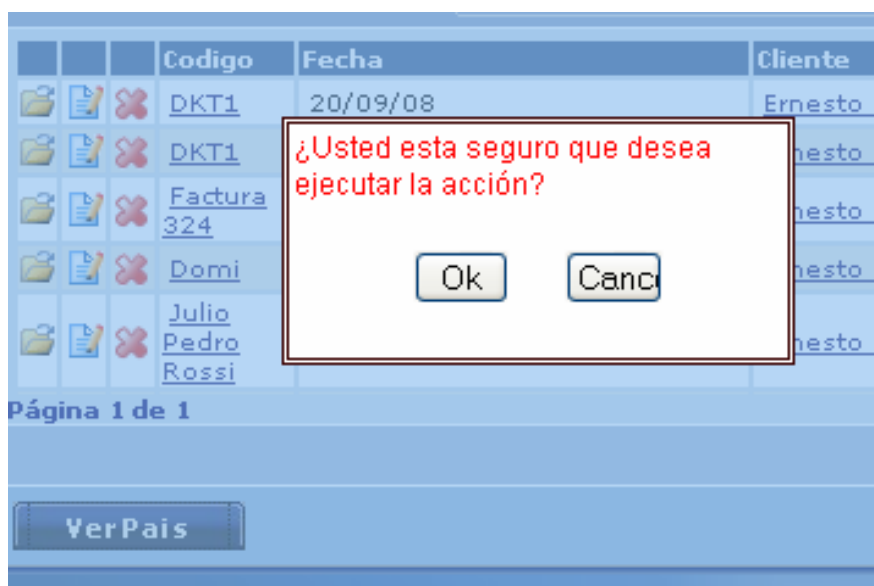
Cuando se genera una acción con un combo, se coloca una imagen a la derecha del combo donde según el item del combo seleccionado se ejecuta la acción correspondiente, la ubicación de esa imagen se puede configurar en el nodo UserDefinedActions

[-] UserActions: UserActions	
[-] ComboActions	
SelectActionInCombo	ActionAddArrow
SelectActionInCombo	Disparar acción
[-] MultiRow	
ErrorMoreThanOneRo	Error : Debe seleccionar solo una línea
ErrorNoneRowsSelect	Error : Debe seleccionar alguna línea

SelectActionInComboToolTip permite configurar el tooltip asociado a la imagen.

4.4.4 Acciones con confirmación:

Dentro de las acciones es posible configurar que la acción sea con confirmación o no. Las acciones con confirmación están funcionando para todo tipos de acciones excepto aquellas que se encuentran dentro del combo o en la grilla.



Dentro del nodo action se encuentra con la categoría ConfirmPopUp

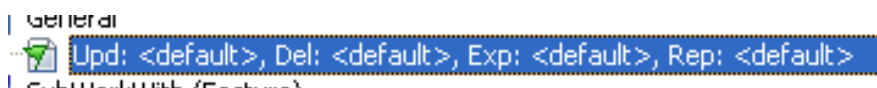


ConfirmPopUp	
Confirm	True
ConfirmMessage	<default>

La propiedad confirm, colocada en true indica que la acción tiene confirmación. En ConfirmMessage es posible escribir el mensaje de confirmación. El default es obtenido en el Settings en el nodo userActions bajo la propiedad ConfirmMessage.

4.4.5 Acciones en general and paralel transaction:

En estos componentes del entity manager, las propiedades asociadas a los nodos se especifica en un nodo tabular modes. Las propiedades son similares a las vistas en las de los objetos con grilla, con la diferencia que no está el modo insert.



El valor default de Update y Delete se obtiene del nodo standard actions en el settings, TabularUpdate y TabularDelete.



4.4.6 Condiciones asociadas a las acciones



Cada acción de usuario, tiene su propiedad condition. Además debajo de modes se cuenta con el nodo InsertCondition, UpdateCondition, DeleteCondition, DisplayCondition, ExportCondition, ReportCondition.

[-] Modes	
Insert	<default>
Update	<default>
Delete	<default>
Display	<default>
Export	<default>
Report	<default>
[-] Security	
InsertCondition	
UpdateCondition	
DeleteCondition	
DisplayCondition	
ExportCondition	
ReportCondition	
[-] Layout	
StandardActionsTopLayout	<default>

En todos estos lugares es posible escribir cualquier expresión booleana incluso la invocación a un udp que retorna un valor booleano.

La forma de trabajo del pattern con condiciones es la siguiente:

- Si la condición se cumple el pattern muestra la imagen.
- Si la condición no se cumple:
 - Si la acción es un boton (layout bottom) se pone invisible.
 - Si la acción es una imagen.
 - Si tiene seteada una imagen de deshabilitada se muestra la imagen de deshabilitada
 - Si no tiene seteada una imagen de deshabilitada entonces se pone invisible la acción.

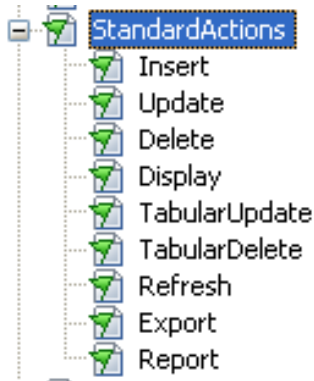
Para las acciones, el valor de la imagen deshabilitada se coloca en la propiedad disableimage, para los modos esto se encuentra en el archivo de configuración, en StandardActions/Action/DisabledImage.

Además de estas condiciones para todas las acciones se controla si se pueden visualizar según un procedimiento de seguridad, este procedimiento en caso de dar falso, pone invisible la acción. El procedimiento de seguridad tiene mayor prioridad sobre las condiciones esto es si una acción no está permitida por el proc de seguridad pero sí lo está por la condition, la acción quedará inhabilitada.

En el nodo modes también es posible configurar el layout que tendrán las acciones top

4.4.7 Nodo Standard Actions (K2BEntityServices.config)






En esta sección se estuvo referenciando a este nodo. Este nodo se configura todo lo relativo a una acción standard.



En la raíz del nodo standard actions, se puede configurar la alineación de las acciones standard top. En caso de poner en layout Left, estas acciones aparecerán a la izquierda.



Este es un ejemplo de cómo quedarían las imágenes a la izquierda.

Trabajar con Proveedor				
Ordenado por Código				
Filtros de Búsqueda				
	Código	Nombre	País	Proveedor Estado
	1	ALMACEN	URUGUAY	Activo
	2	PROVEEDOR	VENEZUELA	Activo
	3	PROVEEDOR S.A	ARGENTINA	Inactivo
	45	AAAAA	URUGUAY	Inactivo
	6	HOLA	URUGUAY	Inactivo

Otras opciones que se pueden configurar son



General	
Caption	Nuevo
Tooltip	Nuevo
Enabled by Default	True
Style	
Image	ActionInsert
DisabledImage	ActionDisabled
ButtonClass	

Caption: Caption del botón en caso de que la acción sea un botón.

Tooltip: Tooltip de la acción.

Enabled by Default : True o False si por defecto va a estar habilitada o no.

Image: Imagen.

DisabledImage: Imagen en caso de que no se cumpla la condición para esa acción.

ButtonClass: Clase del tema para el botón en caso de que la acción sea un botón.

4.5 Automatic Refresh:

En los nodos con grilla, aparece la propiedad automatic refresh. Su valor default es leído desde el archivo de settings, en Grid/GenerateSearchButton.

En caso de estar en true no será necesario la colocación del botón buscar y la grilla se refrescará en la medida que el usuario vaya ingresando los valores. Si se coloca el false, aparecerá el botón buscar y solo se filtrará en la grilla, cuando se precione el botón.

Por lo general para evitar problemas de performance en alto volumen de datos y usuarios concurrentes es recomendable utilizar el automatic refresh en false.

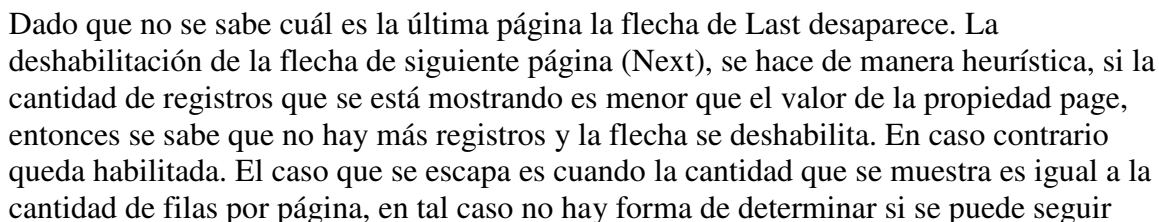
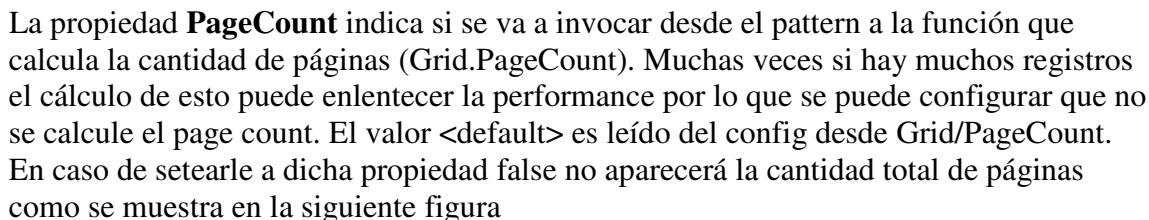
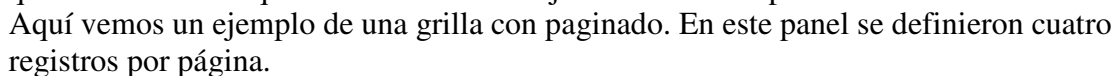
4.6 Paginado:

El paginado es generado cuando se tiene configurada la propiedad automatic refresh en false.

En el nodo correspondiente al objeto con grid se encuentran dos propiedades.

Page	<default>
PageCount	<default>

La propiedad **Page** sirve para indicar si va a haber paginado o no, y en caso de haber paginado, la cantidad de filas que se van a mostrar por página. El valor <unlimited> indica que no se va a paginar, el valor <default> lee del archivo de configuración; esto es





paginado, o no. Ahí se habilita el botón de next y en caso de que el usuario lo presione aparecerá la siguiente página con la grilla vacía.

4.7 Scroll

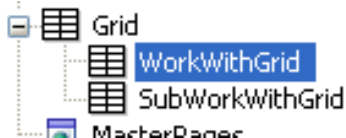
Codigo	Fecha	Cliente	Total
Factura 324	11/11/08	Ernesto Perez	24,00
Domi	11/11/08	Ernesto Perez	12,00
Julio Pedro Rossi	11/11/08	Ernesto Perez	144,00
Codigo23	12/11/08	Ernesto Perez	2772,00
FacturaX	13/11/08	Cliente7	23,00

Es posible definir Scroll dentro de la grilla. Para hacer esto, en el nodo attributes se encuentra con la categoría grid Scholl.

GridScroll	
Scroll	<default>
Height	<default>
Widht	<default>

Ahí se puede configurar si se desea crear un Scroll o no y en caso de querer se puede setear el ancho y alto del mismo.

El valor default es leído del settings y se encuentra ubicado en el nodo grid correspondiente como se muestra en la figura



GridScroll	
Scroll	False
Height	150px
Widht	100%



4.8 Propiedades del WorkWith

Algunas propiedades del nodo workwith.

workWith: WorkWith (Proveedor)	
General	
Name	<default>
Caption	
Description	Proveedor
Page	<default>
PageCount	False
MasterPage	<none>
ObjectProperties	
IsMain	<default>
AutomaticRefresh	<default>

Name: Permite configurar el nombre del objeto que se va a generar.

Caption: Texto que se va a mostrar en el título del trabajar con.

Description: Texto que se va a mostrar en el título del trabajar con cuando el caption es vacío.

IsMain: Permite configurar si el objeto trabajar con es main o no. El valor default es leído de



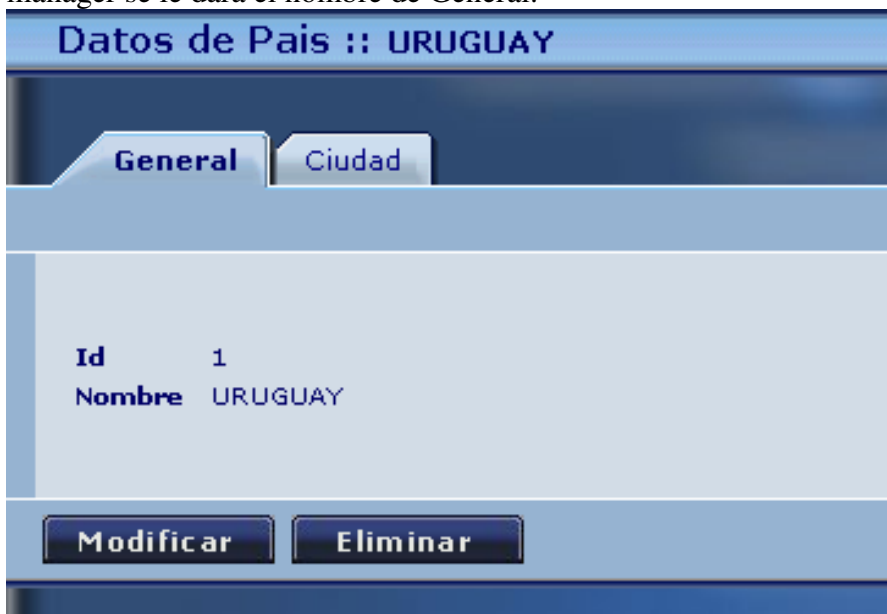
5 Objeto EntityManager:

El EntityManager presenta información asociada a un registro.

Muestra en los tabs los datos relacionados.

Es el objeto que se invoca cuando se presiona el link del DescriptionAttribute.

El primer tab que se muestra en el entitymanager es un tab plano donde se visualiza la transacción base de la instancia en modo display. A este tab o componente del entity manager se le dará el nombre de General.



Datos de Pais :: URUGUAY	
Id	1
Nombre	URUGUAY

Modificar **Eliminar**

Por lo general el resto de los componentes serán similares al workwith, pero en un tab por lo que se les dará el nombre de subworkwith.

Datos de Pais :: URUGUAY

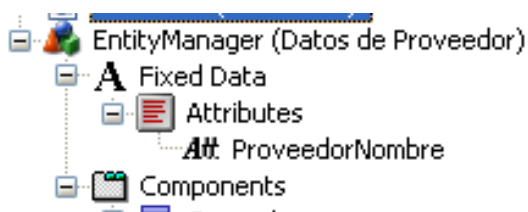
General Ciudad

Nombre

	Id	Nombre
	1	MONTEVIDEO
	2	CANELONES
	3	PUNTA DEL ESTE
	10	ARTIGAS
	11	PAYSANDÚ
	12	SALTO
	13	ROCHA
	14	COLONIA
	15	DURAZNO
	16	TACUAREMBO
	18	CANELONES

El nodo EntityManager está compuesto de los siguientes nodos:

- Fixed Data
- Parameters
- Components



Los parameters son los parámetros del entitymanager. Si el nodo se encuentra vacío será el Modo , la PK de la transacción y el TabCode.



Si se desean modificar los parámetros del entity manager, será posible agregar más parámetros que serán colocados luego de la PK, y antes del TAbCode.

La fixed data esta formada por atributos fijos, que se muestran siempre independientemente de cuál sea el tab seleccionado. En este caso la fixed data es el nombre de Pais.



Por defecto la fixed data es generada con el DA de la transacción base de la instancia, y está configurado para que aparezca en el título del entitymanager. Sin embargo el usuario modificando la instancia puede hacer que este atributo se muestre abajo o colocar más de un atributo en la fixed data. En la siguiente figura se muestran atributos de la fixed data por debajo del título del view. La configuración de la distribución de atributos en la fixed data será vista en la sección noskip.

También es posible agregar variables dentro de la fixed Data y configurar en su propiedad value el valor que tendrá.

5.1 Nodo Components.

Dentro del nodo components es posible agregar un diferente conjunto de objetos, entre estos destacamos los siguientes:



- General: Correspondiente a la transacción en modo display.
- SubWorkWith: Grilla mostrando datos relacionados.
- ParalelTransacton: Vista de una transacción paralela.
- UserDefined: Objeto de usuario a ser mostrado dentro del entitymanager.

5.1.1 Generación por defecto:

A la hora de generar la instancia por defecto, patterns genera un componente general, y luego un subworkwith por cada transacción directamente subordinada a la transacción base de la instancia. También puede generar paralelTransactions por cada transacción paralela, si la propiedad en el archivo de configuración Template/TabsForParallelTransactions está configurada en true.

5.1.2 TabInfo de los componentes:

PageCount	<default>
<input checked="" type="checkbox"/> Tab Info	
Code	ProveedorFacturaWC
Description	Factura
Condition	
<input checked="" type="checkbox"/> ObjectProperties	

Cada componente dentro del entity manager es cargado en un tab.

En cualquier componente del Entity Manager, se cuenta con la propiedad tabInfo, que posee las propiedades Code, Description y Condition.

La propiedad Code es el código interno del tab que debe ser único por cada tab, description es la descripción a mostrar dentro del tab, y en la condition se puede poner cualquier condición que regulará la aparición o no del tab en la pantalla.

Los componentes General, SubWorkWith y Paralel cuentan con una propiedad Name, su valor default se obtiene del archivo de configuración en caso de no tener la propiedad default se deberá setear el nombre del objeto que el pattern generara.

5.1.3 Componente UserDefined:



Definition	
Gxobject	(none)
Tab Info	
Code	
Description	
Condition	

Es posible dar de alta un componente user defined, para mostrar un componente de usuario no generado por el pattern. En este componente las propiedades que se podrán modificar son, GXObject, donde se debe ingresar el objeto a referenciar, y las propiedades asociadas al tab. Se pueden agregar argumentos que serán los argumentos con el que el entity manager invocará al componente de usuario. Por defecto el entity manager siempre invocará al componente con la PK de la transacción base.

5.2 Avanzado:

5.2.1 Parámetros de los componentes (menos user defined):

Cada componente puede tener un nodo parameters. Esto indica la regla parm que va a tener el componente.

La ausencia de nodo parameteres en el subworkwith indica que su regla parm será creada con la PK de la transacción base de la instancia.

El caso en el que la regla parm de un subworkwith es diferente a la del entity manager se da cuando la relación de subordinación es a través de subtipos. En este caso dentro del componente aparece un nodo parameter generado a partir de un subtipo de los atributos PK de la transacción base del entity manager.

5.3 Propiedades de interfaz asociadas a las vistas planas:

Las vistas planas son secciones donde se muestran atributos o variables que no se encuentran adentro de una grilla.

Estas propiedades son válidas para:

- Fixed Data (todos los patterns)
- Atributos en los tabs de tipo tabular
- Filtros (todos los patterns)
- AttributesSection (esto se verá en avanzado).
- Atributos y variables en el web form de la transacción,



Mediante las propiedades de interfaz avanzadas para vistas planas es posible mejorar el look & feel de la aplicación generada, darle más ayudas al usuario y poder agrupar de mejor forma los atributos. Enumeraremos aquí el conjunto de propiedades que hacen posible lograr estas mejoras de interfaz.

The screenshot displays the K2B TOOLS web application interface. At the top, there is a navigation bar with links for / Clients / Invoices / Clients. Below this, the client name 'Client :: John' is shown. The interface is divided into two tabs: 'General' (selected) and 'Additional Information'. The 'General' tab contains the following fields: Id (1), Firstname (John), Surname (Russo), Sex (Male), Birth Date (03/06/85), City Id (1 with a dropdown arrow), and City (Montevideo). Below this is a section for 'Contact Information' with a sub-section 'Address' containing Street (21 de Setiembre), Number (2869), and Apartment (12). Other contact details include Phone Number (7112345) and Mail (John@gmail.com). Below the contact information is a section for 'Occupation Information' with fields for Salary (12.00), Occupation (Programer), and Work Hours (100). At the bottom of the form, there are two buttons: 'Modificar' and 'Eliminar'.

5.3.1 Propiedad NoSkip:

La propiedad NoSkip es una propiedad válida para todas las vistas tabulares y permite agrupar atributos dentro de una pantalla

Por defecto patterns genera todos los atributos planos en filas diferentes (noskip= false). La propiedad noskip en true, indica que el atributo permanecerá en la misma línea que el atributo anterior.

A partir de esto Patterns determina los posicionamientos de los atributos y la forma de generar la tabla.

5.3.1.1 Forma de generar la tabla:

El objetivo a la hora de generar la tabla con los diferentes atributos es mantener siempre la alineación de las descripciones y los atributos; esto es que en una misma columna no haya descripciones y atributos dado que estos tienen diferente alineación y hacen que el web form quede de forma incorrecta.

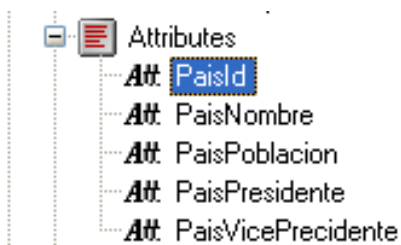
El procedimiento para generar el web form es el siguiente:

Dado un atributo:

- NoSkip en false => a la siguiente línea.
- NoSkip en true:
 - Tiene Descripción => Se cierra la celda del atributo anterior y se generan dos celdas, en una se coloca la descripción y en otra se coloca el atributo-
 - No tiene descripción => No se cierra la celda del atributo anterior y se coloca este atributo al lado.

Dado que cada fila puede tener distinto número de columnas, patterns genera la última celda de una fila, con el colspan restante para que todas las filas posean el mismo tamaño.

Ejemplo:



Atributo	Descripción	NoSkip
PaisId	País	False
PaisNombre		True
PaisPoblacion	Población	False
PaisPresidente	Presidente	False
PaisVicePresidente	Vice-Presidente	True

Resultado:

Pais	Pais PaisNombre		
Población	Pais		
Presidente	PaisPresidente	Vice-Presidente	PaisVicePresidente

5.3.1.2 NoSkip Fixed Data:

Dentro de un entitymanager podemos distinguir dos secciones. Una el título del entitymanager, y otra debajo del título del entitymanager.



Dentro de estas secciones lo que se visualiza es la fixed data. La pregunta que puede surgir es como se configura que un atributo de la fixed data este en el título del entity manager o este debajo del título del entity manager.

Pues bien, la respuesta es la siguiente. El nodo fixed data es el único nodo donde la propiedad noskip en true del primer atributo tiene sentido. Dado que en caso de estar en true es colocado en el título del entity manager.

En primer lugar se coloca la descripción del entity manager, (propiedad description del nodo) y luego todos los atributos que esten en la primera fila uno al lado del otro ignorando su descripción. A partir de la fila siguiente se aplicará el mismo algoritmo del noskip pero los atributos serán posicionados en la fila de abajo y su descripción será tenida en cuenta.

Acá se muestra un ejemplo de cómo fue generada la sección noskip del view de facturas.

Atributo	Descripción	NoSkip
FacturaId	Id	True
FacturaFecha	Fecha	True
ProveedorId	Proveedor	False
ProveedorNombre		True
FacturaTotal	Total	False

5.3.2 Grupos

El nodo group permite agrupar un conjunto de elementos.



En este caso se está mostrando un grupo de nombre fecha, donde se colocan los filtros correspondientes a la fecha.



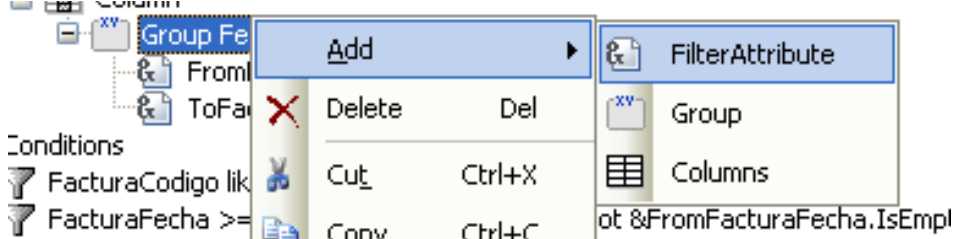
Definition	
ControlName	Fecha
Form	
Caption	Fecha
ThemeClass	
NoSkip	False

ControlName es el nombre de control, si es vacío el pattern generará uno automáticamente.

En la propiedad caption se coloca el título del grupo y en themeClass se ingresa la clase del tema asociada, para asignarle una clase del tema a todo el grupo. En caso de estar vacío tomará la clase FilterGroup.

La propiedad noskip también se usa para indicar si ese grupo se encontrará en la misma fila que los elementos anteriores, o en la fila siguiente.

Como hijo de los grupos se pondrán ingresar atributos/variables y columnas que veremos a continuación.

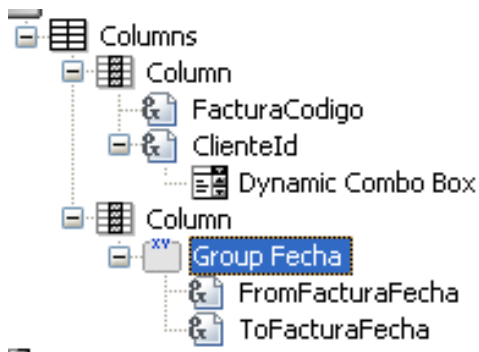


5.3.3 Columnas

Es posible también configurar en qué columnas se mostraran los diferentes elementos. En el siguiente ejemplo se muestra una sección con dos columnas una conformada por el código y cliente, y otra que contiene al grupo fecha.



Esto se implementa en el patrón indicando mediante el nodo columns, donde a partir de él se definen todas las columnas que tendrá la sección.

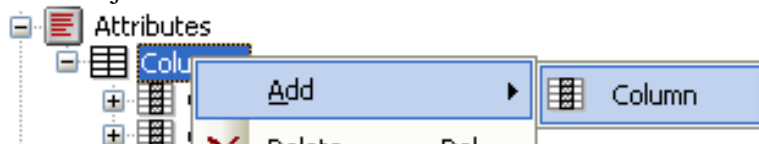


Las propiedades del nodo columns son las siguientes:

Definition	
ControlName	
Form	
ThemeClass	
NoSkip	False



En control name se puede especificar el nombre de la tabla que define la columna. Si no se coloca nada el pattern va a inferir uno. En ThemeClass se puede colocar el nombre de la clase del tema que contendrá la columna y la propiedad noskip referencia sí la tabla que contiene las columnas va a estar en la misma fila que el componente anterior o no. Cómo hijos de la columna tenemos columna.



En cada Column es posible especificar las siguientes propiedades.

Form	
Width	
Height	
Align	Left
VerticalAlign	Middle

En Width es el ancho que va a tener la columna, el Height es el alto. Si se pone vacío el pattern no agrega estas propiedades y deja la columna sin ancho fijo. También es posible configurar la alineación vertical y la horizontal.

Cómo hijo se podrá agregar otros atributos, variables, grupos o columnas.



5.3.4 Right text y left text en atributos y variables

Es posible configurar en los atributos/variables de las secciones planas, un texto diferente a la descripción que puede aparecer tanto a la izquierda como a la derecha. Este texto puede servir para por ejemplo poner de qué unidad es un atributo (% , \$, etc), calificar algún atributo (Sr.) ,etc.

En este ejemplo asociado a los campos de tipo fecha se encuentra en qué formato debe ser entregada la fecha.



Mediante el uso de esta propiedad es posible que el usuario agregue un texto a ser visualizado debajo del campo cuando el usuario de la aplicación comienza a ingresar datos en dicho campo.

La propiedad es context help value y el texto debe ser colocado en comillas. Se permite en este campo por ejemplo concatenar valores de otros atributos, variables, etc.

Clients

Filters

City

Name

Buscar

Select Client Name

	IdName	City
	1 John	Montevideo
	2 Ernest	Montevideo

5.3.6 Line Separator


Permite generar líneas separadoras para ordenar de mejor forma la información.

En la figura de abajo se muestra el uso de esta funcionalidad para separar la información de contacto de cliente, con la de ocupación. Como se muestra en la figura es posible también colapsar información secundaria que en determinado momento no se desea visualizar.

Clients

Client :: Ernest

General

Id 2
 Firstname Ernest Surname Call
 Sex Male
 Birth Date 03/25/09
 City Id 1 
 City Montevideo

ContactInformation

Address

Street Richmond Number 12 Apartment 1

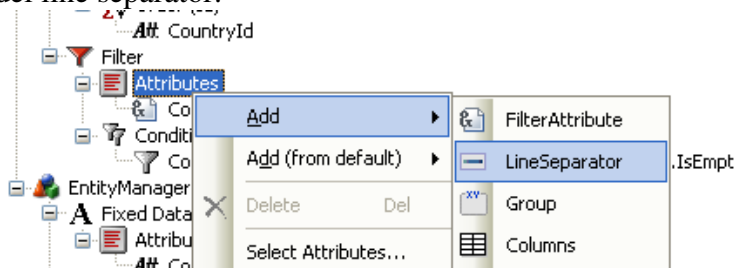
Phone Number 9141111
 Mail ernest@hotmail.com

OccupationInformation

Modificar **Eliminar**

Otro caso de uso puede ser en los filtros, usando line separator es posible definir filtros avanzados, que aparezcan por defecto colapsados, para que el usuario los vea y los ingrese solo cuando considere necesario.

Para utilizar esta funcionalidad en cualquier nodo attributes plano se puede hacer un add del line separator.



Su funcionamiento es el siguiente, cuando el pattern encuentra este nodo genera en el web form una línea separadora. El nodo line separator puede tener hijos, si no tiene, el pattern simplemente colocará esta línea separadora y dividirá los atributos que están arriba de la línea de los que están abajo.

En caso de tener hijos (atributos, grupos, variables) dividirá también los atributos de debajo de los de arriba pero se le incorporará la posibilidad de colapsar los nodos hijos.



[-] Definition	
Label	Separador
Collapsed	False
[-] ThemeClass	
LabelThemeClass	<default>
SectionThemeClass	<default>

Es posible especificar en el nodo line separator si se desea que los atributos aparezcan colapsados o no, según el valor de la propiedad collapsed. También es posible setear el tema del label de la línea, y de la sección.

6 Reportes PDF y Excel.

Todos los web panels generados por patterns tienen la posibilidad de ser exportados tanto a pdf como a Excel. El objetivo de los reportes es mostrar lo mismo que se está mostrando en el web pannel.

Si el web panel que se está mostrando es un objeto con grilla se van a mostrar los atributos del grid ordenados por el orden seleccionado por el usuario, filtrado por el valor de los filtros en ese momento.

Supongamos que se tiene el siguiente trabajar con países:

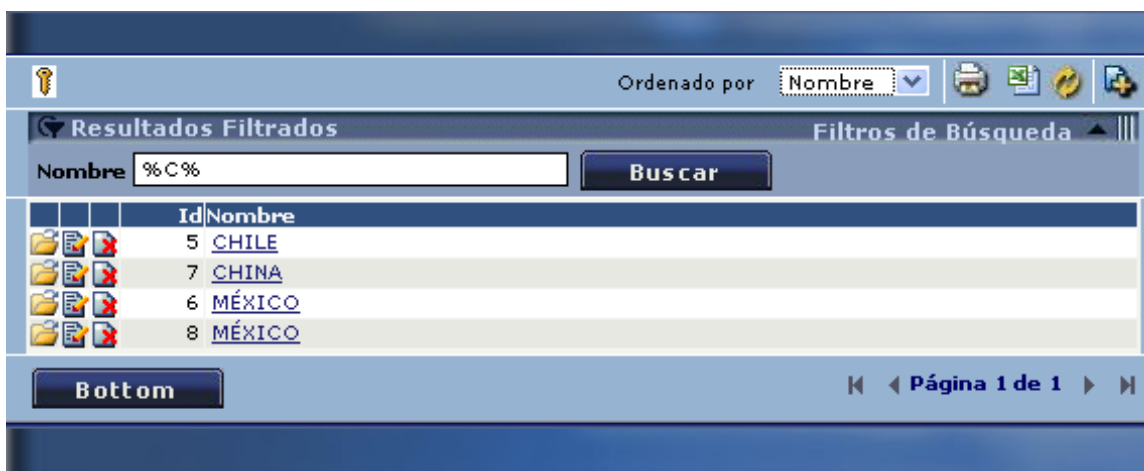


Id	Nombre
1	URUGUAY
2	ARGENTINA
3	BRASIL
4	PARAGUAY
5	CHILE
6	MÉXICO
7	CHINA
8	MÉXICO
9	VENEZUELA
10	BOLIVIA





La exportación asociada será algo como lo siguiente:

Id	Nombre
1	URUGUAY
2	ARGENTINA
3	BRASIL
4	PARAGUAY
5	CHILE
6	MÉXICO
7	CHINA
8	MÉXICO
9	VENEZUELA
10	BOLIVIA

Si se cambia el orden y se agrega un filtro se mostrará lo siguiente:



The screenshot shows a web application interface with a search and filter section. At the top, there is a key icon and a dropdown menu labeled "Ordenado por" with "Nombre" selected. To the right are icons for print, export, refresh, and a plus sign. Below this is a section titled "Resultados Filtrados" with a search input field containing "%C%" and a "Buscar" button. To the right of the search field is a "Filtros de Búsqueda" section with a dropdown arrow. The main content area displays a table with the following data:

	Id	Nombre
	5	CHILE
	7	CHINA
	6	MÉXICO
	8	MÉXICO

At the bottom of the interface, there is a "Bottom" button on the left and a pagination control on the right showing "Página 1 de 1" with navigation arrows.

Paises

Nombre	%C%
--------	-----

Id	Nombre
5	CHILE
7	CHINA
6	MÉXICO
8	MÉXICO

6.1 Particularidades reportes PDF

La diferencia más importante entre los reportes PDF y Excel es que en los reportes PDF se cuenta con una limitante que es el tamaño de la hoja. Para esto el pattern k2bentyservices agrega de los atributos que van en el reporte sólo aquellos que entran en la hoja. El resto de los atributos los ignora.

Veremos a continuación algunas configuraciones que se pueden hacer para los reportes PDF. En el nodo standard actions / Report es posible configurar las siguientes propiedades:

- PaperConfiguration	
PaperType	A4
Orientation	Portrait
LeftMargin	50
RightMargin	50
- LineHeight	
TitleLineWidth	4
ColumnsLineWidth	3
- PrintBlockHeight	
PrintBlockTitleHeight	40
PrintBlockNotAuthoriz	30
PrintBlockFilterHeight	30
PrintBlockColumnsHei	30
PrintBlockRowsHeight	30
- Layout	
ReportLayout	ReportWWLayout

Tipo de Papel: Se pueden seleccionar diferentes tipos de papeles en los que saldrán todos los reportes.

Orientación: Si se desea que el reporte salga apaisado o no.

LeftMargin: Margen izquierdo con respecto a la hoja.

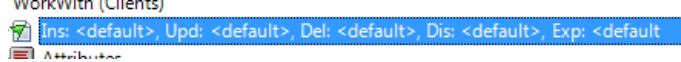
RightMargin: Margen derecho con respecto a la hoja.

TitleLineWidth: Ancho de la línea que separa el título.

Luego el tamaño de cada uno de los printblocks.



Parte de estas configuraciones se pueden modificar por instancia, por ejemplo en el nodo modes de la instancia se puede configurar el tipo de papel y su orientación.



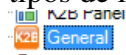
ReportConfiguration	
PaperType	<default>
Orientation	<default>

6.1.1 Objeto de Layout

El reporte está basado en un objeto de Layout, donde se especifica el formato que tiene el reporte. Que cabecal va a tener y qué pie de página. Para esto en la propiedad ReportLayout, ubicada en el nodo standardactions/report del settings se puede asociar un objeto que actúe de Layout del reporte. En este caso se usa el ReportWWLayout. Es posible en el configurar que por ejemplo siempre aparezca una imagen, etc.

6.2 Configuración nodo general

En el nodo K2BTools – General de las preferencias, es posible configurar los diferentes tipos de letra que puede tener el reporte.



ConfirmMessage	¿Are you sure?
Report	
+ Title Font	Courier New, 26pt, style=Bold
+ NotAuthorized For	Courier New, 20pt, style=Bold
+ Filter Desc Font	Courier New, 12pt, style=Bold
+ Filter Field Font	Courier New, 12pt
+ Columns Font	Courier New, 12pt, style=Bold
+ Row Field Font	Courier New, 12pt
SpaceBetweenColumn	50
Label	
NotAuthorizedToVi	You are not authorized to view this report

Ahí es posible configurar la diferente letra de los diferentes campos que pueden haber en el reporte. El título, filtros, títulos y el espacio que hay entre columnas.

6.3 Atributos que van en los reportes y pantalla.



Por defecto el pattern k2bentityservices coloca en los reportes exactamente lo mismo que está en pantalla. De todas formas esto es posible cambiarlo. A partir de la versión 5.0 se cuenta con la propiedad usein asociado al atributo. Esta propiedad permite configurar si el atributo va a ser utilizado en todo (trabajar con, pdf y export) en solo el pdf, en solo el trabajar con, trabajar con y export, y todas las combinaciones posibles como se muestra en la figura.

Att: CustomerFirstName
Att: CustomerLastName
Att: CustomerDNI
Att: CustomerState

UseIn	All
Link	All
Autolink	WorkWith Report Export WorkWithAndReport WorkWithAndExport ReportAndExport

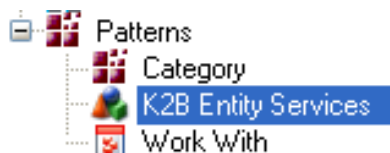


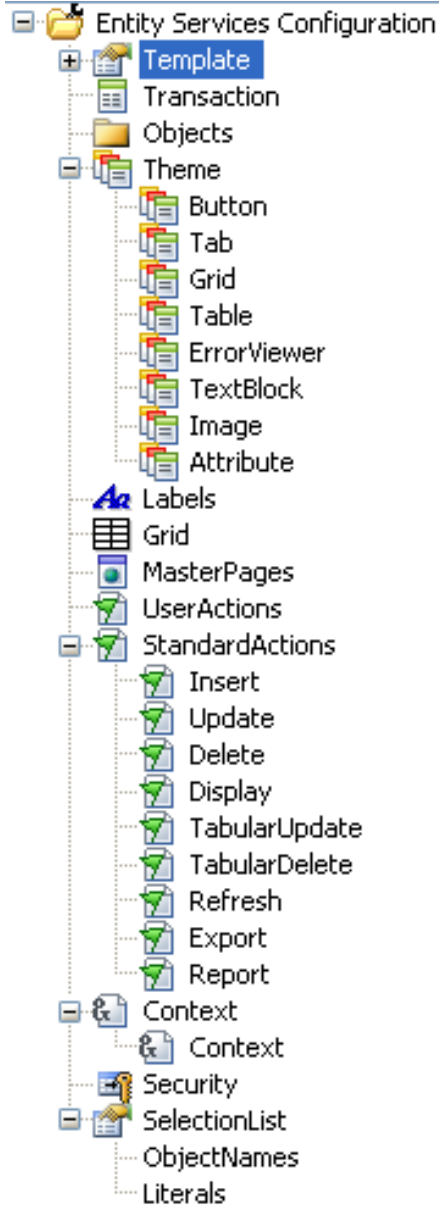
7 Propiedades avanzadas del pattern K2BEntityServices:

7.1 *Settings y configuración general.*

En K2BTools existen dos lugares donde se puede configurar metadata válida para todas las instancias. Uno de estos es el settings donde se encuentra metadata válida para cada una de las instancias del patrón, y la configuración general de las k2btools donde se coloca metadata válida de todas las herramientas que componen las k2btools.

Si se selecciona dentro de preferences, en Patterns el Pattern K2BEntityServices será posible setear metadata válida para todas las instancias.





Aquí nombraremos algunos ejemplos de metadatos que se pueden setear.

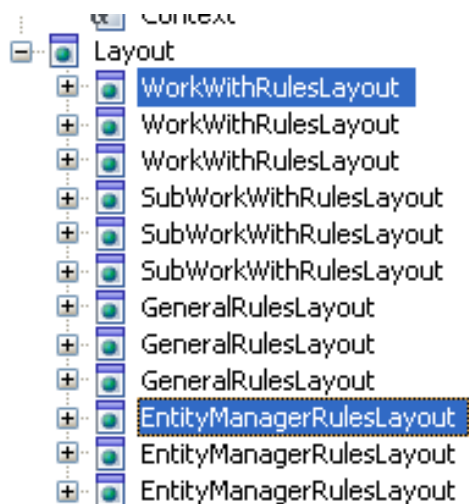
7.2 Master Pages

Por defecto pattern genera cada web panel (workwith y entitymanager) con una master page específica.

La master page puede ser especificada por instancia. Esta propiedad se encuentra en el nodo principal de cada uno de estos objetos. El valor <default> indica que este valor se va a tomar del archivo de configuración.



Dentro del archivo de configuración en el nodo layout correspondiente a la interfaz seleccionada (en modificación de interfaz está esto explicado) es posible configurar que master page se utilizará para el workwith y para el entity manager.



[-] WorkWithRulesLayout: WorkWithRulesLayout	
[-] Layout	
LayoutObject	WWLayout
HasEventsCode	True
HasVariables	True
[-] Master Page	
MasterPage	K2BWWMasterPage

[-] EntityManagerRulesLayout: EntityManagerRulesLayout	
[-] Layout	
LayoutObject	EntityManagerLayout
HasEventsCode	True
HasVariables	True
[-] Master Page	
MasterPage	K2BWWMasterPage
[-] Tabbed View	
TabbedViewWebCom	K2BTabbedView



La master page por defecto se encarga de crear el componente header, que es el que se muestra en todos los objetos y crear el componente donde se muestran los recent links (K2BRecentLinks).

Estos objetos pueden ser customizados y adaptados a la lógica que el desarrollador desee para por ejemplo agregar un footer, modificar la interfaz del header ,etc.

7.3 Manejo de sesión

Por defecto Patterns en el evento start de cada objeto invoca al procedimiento K2BGetContext(&Context). Este procedimiento lo que hace es cargar la información de sesión en una variable de tipo K2BContext para que sea accedida dentro de los objetos generados por patterns. La información que se puede almacenar en esta variable puede ser el usuario conectado, la empresa con la que se está trabajando, etc. Por defecto, la variable Context está basado en un SDT de nombre K2BContext.

K2BContext		K2 BContext
■ User	Character(20)	User
■ Empresa	Numeric(6,0)	Empresa
■ Sucursal	Numeric(6,0)	Sucursal
■ CentroCompra	Numeric(6,0)	Centro Compra
■ Proveedor	Numeric(6,0)	Proveedor
■ usrper	Numeric(6,0)	usrper
■ Peticion	Numeric(6,0)	Peticion

Patterns provee además un objeto para setear el context, K2BSetContext, que dada una variable de tipo Context la setea en el contexto. Por lo general el programa K2BSetContext puede ser utilizada en una pantalla de login para setear en la sesión el usuario cuando este se loguea.

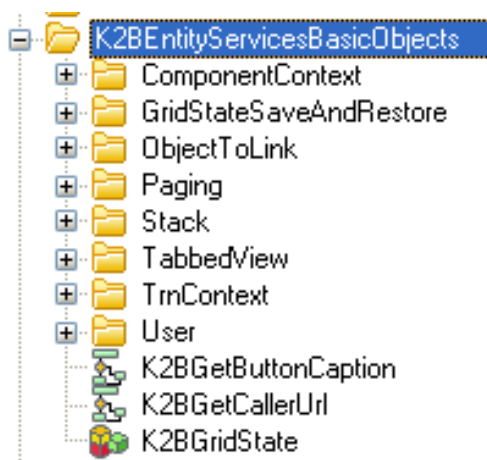
Es posible por cada aplicación definir un context propio. Se pueden definir los nombres de los programas de carga de context en el archivo de configuración.

Context &Context	General	
	Name	Context
	Type	K2BContext
	LoadProcedure	K2BGetContext

7.4 Objetos básicos



El Pattern K2BEntityServices la primera vez que se aplica sobre una base de conocimiento, consolida un conjunto de objetos, los cuales se llaman objetos básicos. Estos objetos se encuentran bajo la carpeta K2BEntityServicesBasicObjects.



7.4.1 Customización de objetos básicos

Algunos de estos objetos pueden ser customizados por el desarrollador del pattern, de forma de adaptarlo a sus necesidades.

Dentro de la carpeta User, se pueden adaptar las master page y los headers, para colocar el header que requiera la aplicación.



En Security se puede modificar los procedimientos de seguridad que por defecto devuelven todos true.

En Session se pueden modificar los programas K2BSessionSet y K2BSessionGet para que modifiquen la forma en que se guardan los datos de contexto. Por defecto se usa la web session, pero esto se puede modificar para que se usen cookies o directamente almacenarla en la base de datos.

En Components se puede modificar las master pages, headers, etc.

7.4.2 Recursos



Se cuenta con un XPZ donde se consolidan también las imágenes. Todas las imágenes de K2B comenzarán con el prefijo K2B y se consolidarán en la KB la primera vez que se aplique el pattern.

7.5 Nomenclatura de objetos

Un punto importante es como el pattern determina cuál es el nombre de los objetos que va a generar. Para esto, se debe ir en el archivo de configuración al nodo Objects.

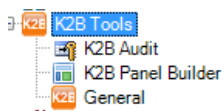
Objects	
EntityManager	EntityManager <TrnName>
WorkWith	WW <TrnName>
GeneralComponent	<TrnName>General
ParallelTransactionCor	<MainTrnName><TrnName>
SubWorkWith	<MainTrnName><TrnName>WC
Export	Export <ObjectName>
Report	Report <ObjectName>

Ahí aparece un conjunto de objetos y como se van a generar.

El Tag <TrnName> será sustituido por el nombre de la transacción en la que se encuentra asociado el objeto. Para determinar el valor de <TrnName> en el nodo subworkwith se toma en cuenta el nombre de la transacción seteada en la propiedad relatedtransaction; mientras que en el workwith se toma el nombre de la transacción padre del workwith. Para los componentes del entity manager se cuenta con el tag <MainTrnName> que será sustituida por la transacción padre del entity manager. Para los objetos generados que están relacionados a un objeto específico, se cuenta con el tag <ObjectName> que es el nombre del objeto generado al que está asociado. Esto es para los reportes asociados a las grillas.

7.6 Configuración general

En la medida que las k2btools fueron avanzando y se fueron incorporando nuevas herramientas y patrones, el settings de cada pattern no alcanzó para poder realizar toda la configuración asociada a los patrones. Para esto se creó en las preferencias un nuevo nodo de nombre K2BTools donde se especifica configuración válida para todas las herramientas que forman parte de las k2btools.





En este nodo, si vamos al nodo general, podemos setear configuración válida para todos los patrones, como puede ser que filtros por defecto se van a generar, tema a utilizar, si asignarle a cada objeto el tema de k2b o que lo tome de las propiedades del modelo, etc. Todo esto está para que sea configurado en la configuración general de las k2btools.

Filters	
Generate Date Filters	True
Generate DateTime F	True
Label Date From	From
Label DateTime From	From
Label Date To	To
Label DateTime To	To
Default Date From	ServerDate() - 30
Default DateTime Fro	ServerDate() - 30
Default Date To	ServerDate()
Default DateTime To	ServerDate() + 1
Theme	
WebTheme	K2B3
Set Object Theme	False
Labels	
OrderedBy	Ordered By
AllInCombo	All
Record Not Found	Record Not Found
TooltipText	
GridLevelHeaderTi	Click for maximize, minimize
GridLevelHeaderTi	Click for maximize, minimize
SelectActionInCom	Execute
Actions	
SelectActionInCombo	K2BActionAddAll
ErrorMoreThanOneR	Error : You must select more than one row

En esta configuración su valor default depende del idioma seleccionado dentro de la KB.

8 Propiedades avanzadas de los objetos

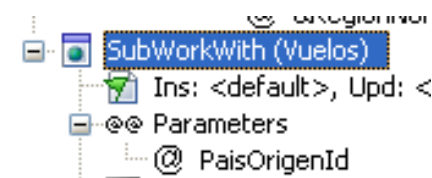
Aquí se listará con más detalle el funcionamiento de los objetos , así como propiedades avanzadas. El objetivo de estas propiedades avanzadas fue siempre aumentar la cantidad de objetos que se pueden generar con el Pattern K2BEntityServices para poder mantener la mayor cantidad de objetos por dentro del Pattern.

8.1 Manejo de parámetros

El Pattern K2BEntityServices genera en todos los objetos su regla parm basado en el nodo parameters. Si el nodo parameters está vacío, en los subworkwiths se tomará como parámetros la PK de la transacción relacionada.

La regla parm la genera con variables y para instanciar el atributo de esa variable agrega una condition att = &att.

Ej:



```
parm(in:&PaisId);
```

Conditions:

```
PaisId = &PaisId;
```

El workwith también puede recibir parámetros. En caso que se quiera recibir una variable basada en un atributo pero que no se genere la condition, se puede pasar en parámetros la variable &att, y así el pattern no genera la condición automática.

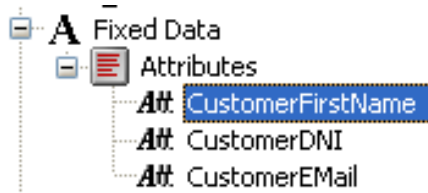
En los componentes si no se encuentra el nodo parameters se toma como parámetros la pk de la transacción padre del entity manager asociado.

8.2 Fixed data en el workwith

En el pattern K2BEntityServices es posible definir Fixed Data en el workwith. La ubicación de la fixed data es similar que dentro del entitymanager. Si el primer atributo tiene noskip en true este y los de la misma fila son colocados en el cabezal del workwith, luego los atributos hijos son colocados abajo.



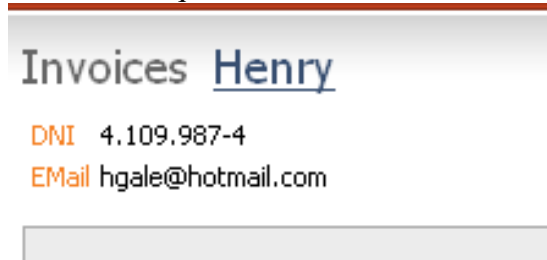
Ej: Se tiene un trabajar con facturas que recibe el proveedor como parámetro como se muestra en la instancia.



Coloco la siguiente fixed data para mostrar con los siguientes valores correspondientes a la propiedad noskip.

Atributo	Descripción	NoSkip
CustomerFirstName		True
CustomerDNI	DNI	False
CustomerEMail	EMail	False

El resultado que se obtiene es el mostrado abajo-

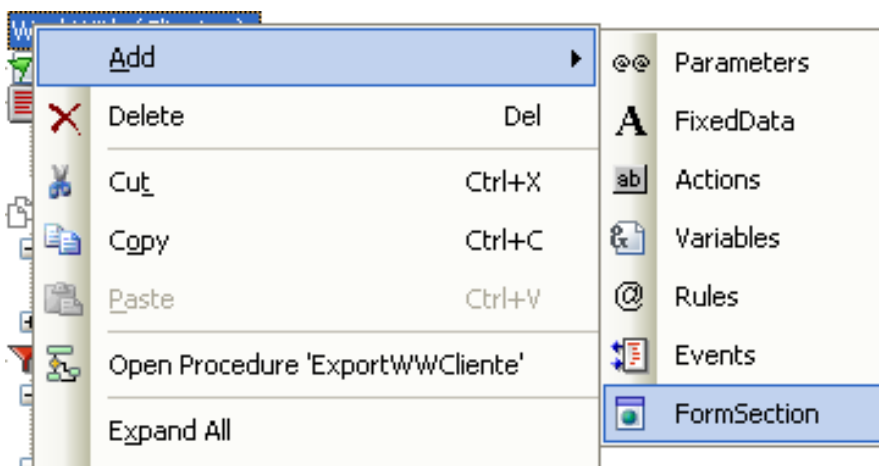


Un tema a tener en cuenta es la diferencia entre la fixed data en el workwith y en el entity manager. En el primero se muestra un conjunto de registros, por lo tanto hay que asegurarse de que lo que se coloca en la fixed data tiene que ser superordinado a los registros que se encuentran en la tabla base. En el entity manager se muestra un único registro por lo tanto la fixed data va a ser única.

8.3 FormSections



Las form sections permiten flexibilizar la interfaz de los objetos generados con el patrón permitiendo integrarlos con otros objetos de interfaz que serán visualizados en la misma pantalla. Parado en cualquier nodo asociado a un objeto con interfaz, es posible dar de alta una form section.



Estas form sections son secciones de interfaz que pueden ser de dos tipos. WebComponent o Attributes.



8.3.1 Attributes Section

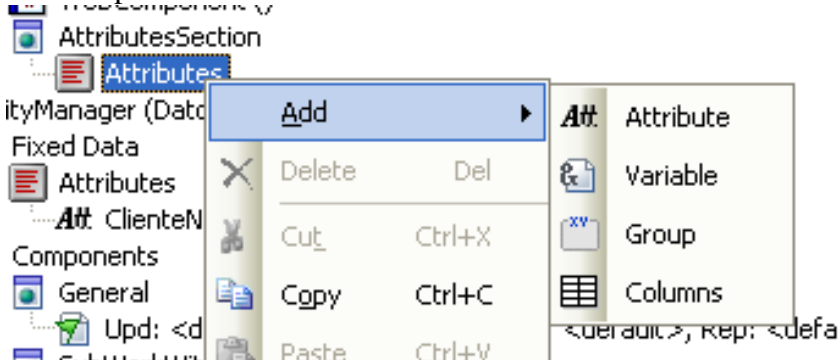
Los AttributesSection se corresponden con una sección donde se mostrarán un conjunto de atributos o variables.

2772,00
23,00
24,00
Total 5996,00

Esto puede servir por ejemplo para mostrar totales debajo de la grilla.

Type	Attributes
Form	
Layout	Bottom
Align	Center

El Type puede ser Attributes o FreeStyleGrid. Por el momento los FreeStyleGrids no están implementados.



Dentro de esta sección de atributos es posible agregar atributos variables, y grupos y columnas.

Las propiedades asociadas a los atributos son similares a las propiedades en cualquier sección plana.

Definition	
Attribute	FacturaTotal
Description	Factura Total
Form	
Visible	True
LeftText	
RightText	
ThemeClass	
DescriptionThemeClass	
NoSkip	False
Format	<default>
Link	
Autolink	True

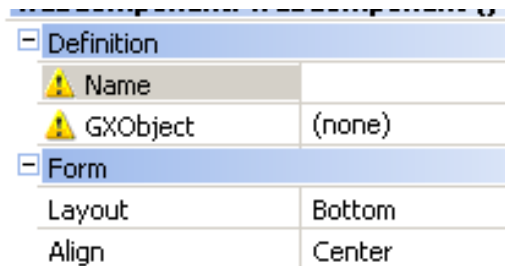
Dentro de las variables ubicadas en las attributes sections es posible definir un OnStartValue, OnRefreshValue y OnLoadValue que serán expresiones a ejecutarse en cada uno de los respectivos eventos del panel. Esto puede servir para calcular totales de lo que se está mostrando en la grilla, o dar la posibilidad de incluir una llamada a un udp.



Definition	
Name	
Description	
BasedOn	(none)
Form	
Visible	True
ReadOnly	True
LeftText	
RightText	
ThemeClass	
DescriptionThemeClass	
NoSkip	False
Format	<default>
Value	
OnStartValue	
OnRefreshValue	
OnLoadValue	

8.3.2 Web Components Section

En los form sections también se pueden definir web components.



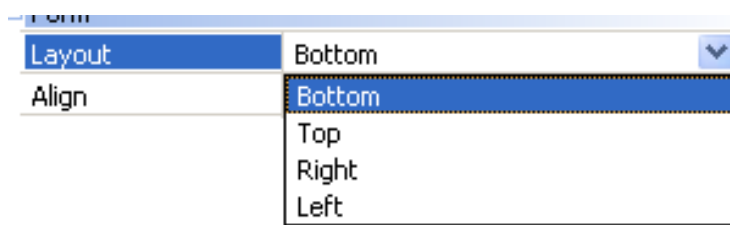
Para esto en GXObject se coloca el nombre del objeto que se desea poner como web component.

En Arguments es posible configurar los argumentos con los que se invocará a ese web component. Este web component será creado en el evento refresh del web panel.

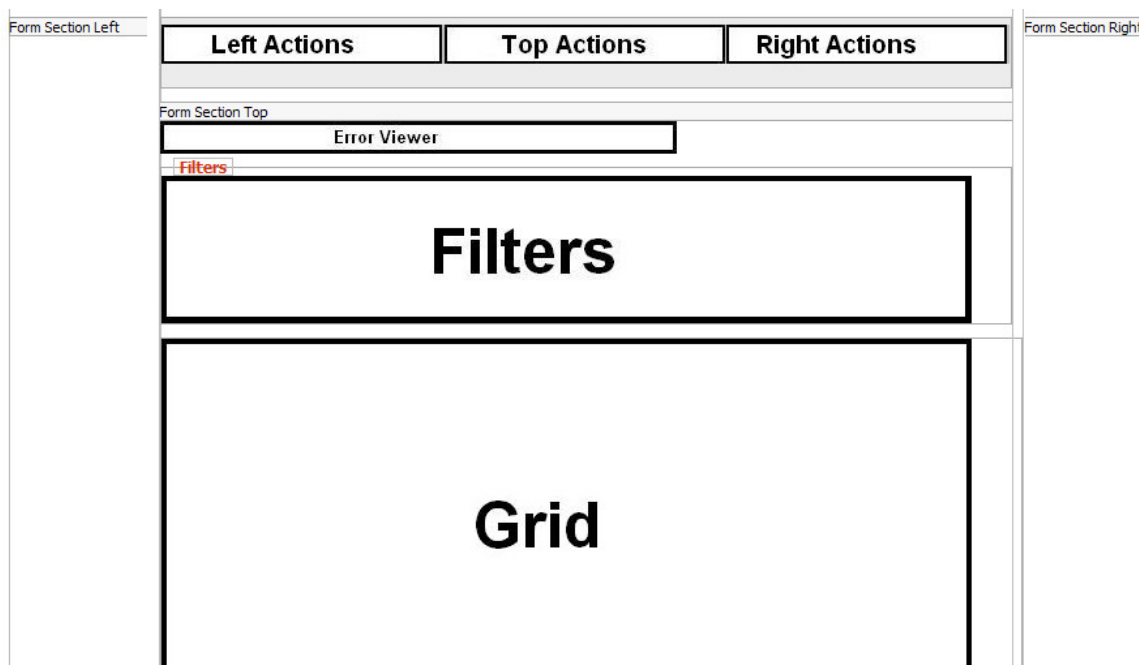


8.3.3 Layout de los form sections

El Layout de las form section se puede especificar dentro de la propiedad Layout.



Esto va a depender de la configuración que se haga dentro del objeto layout asociado al tipo de objeto generado. Por defecto, para los objetos con grilla las form sections top y bottom se encuentran encima o debajo de la grilla, y las left y right a la izquierda y derecha de la misma.



Es posible definir form sections en cualquier objeto generado por el patrón, ya sea objetos tabulares o cualquier tipo de objetos. Abajo se encuentra la ubicación de las form sections en un objeto general.

Description	First Row Fixed Data	Back To WorkWith
Bottom Fixed Data		
Form Section Left	Form Section Top	Form Section Right
	tabbedView	
	Form Section Bottom	
	HiddenItems	

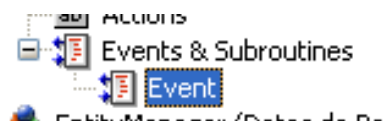
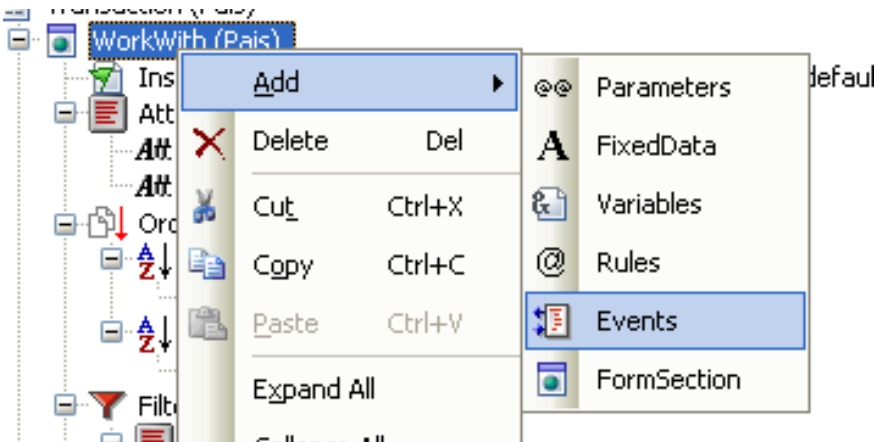
Las form sections también es posible definirlas dentro de los objetos de tipo entity manager como podemos visualizar abajo.

Bottom Fixed Data								
Form Section Left	<table border="1"> <tr> <td>Left Actions</td> <td>Top Actions</td> <td>Right Actions</td> </tr> <tr> <td colspan="3"></td> </tr> </table>	Left Actions	Top Actions	Right Actions				Form Section Right
Left Actions	Top Actions	Right Actions						
	Form Section Top							
	Error Viewer							
	Filters							
	Grid							

Las form sections también pueden ser especificadas dentro del prompt generado por el pattern K2BPrompt.

8.4 Eventos

Muchas veces existe cierto comportamiento que no es posible lograr utilizando las propiedades que aparecen en la instancia del pattern. En muchos casos se desea traer información desde un SDT para mostrar en la grilla, definir algún nuevo evento o subrutina, etc. Para esto el pattern cuenta con la posibilidad que desde el pattern, se pueda definir código a ser ejecutado en el evento o subrutina que el usuario seleccione. Esto es realizado usando el nodo events. Parado en cualquier objeto es posible seleccionar add events



Definition	
Name	
Type	Event
BeginCode	
EndCode	

Estas son las propiedades que poseen los eventos.

En Type se configura si lo que se va a definir es un evento o una subrutina. Para esto solo alcanza con modificar el valor de la propiedad type. En name se selecciona en qué evento se desea agregar el código.

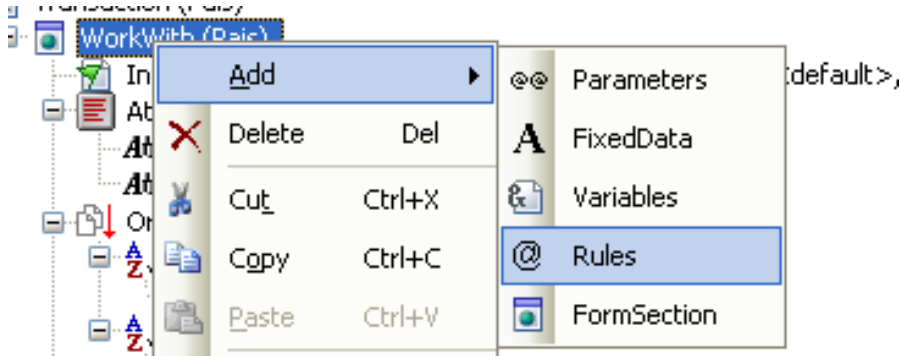
El código puede ser ingresado en algunos de los eventos ya creados por el pattern, esto es en el Start, Refresh, Load, o [MainGrid.Load] que representa el evento load asociado a la grilla. El código es agregado al principio y al final según el código se coloque en BeginCode o EndCode.

En caso de querer definir un evento o subrutina nueva, simplemente alcanza con poner en Name, el nombre del evento nuevo y en Begin Code o End Code la subrutina.

Definition	
Name	
Type	Start
BeginCode	Refresh
EndCode	Load
	[MainGrid.Load]

8.5 Reglas

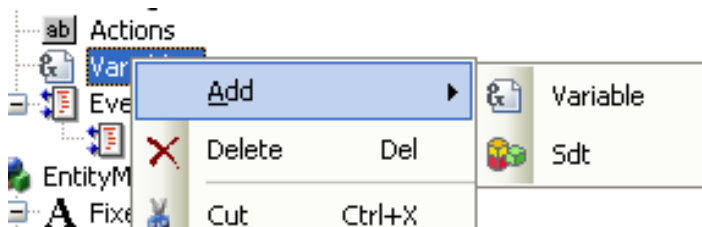
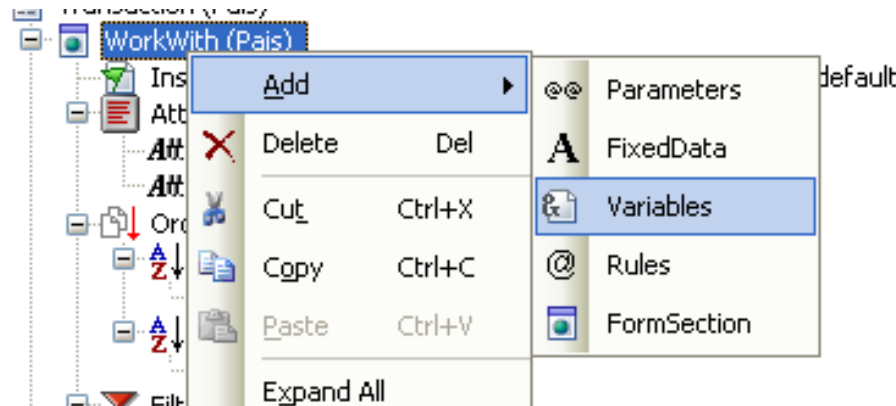
En los objetos con grilla es posible también agregar reglas. Para esto se cuenta con el nodo rules.



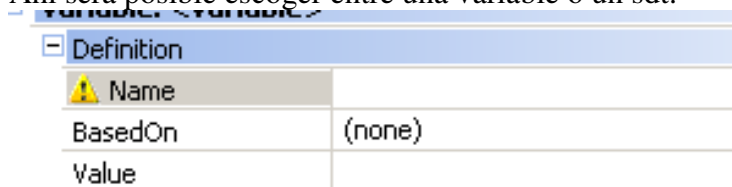
En dicho nodo simplemente se colocan las reglas. Se pueden agregar todas las reglas que uno desee.

8.6 Variables

Las variables que se visualizan en pantalla son por lo general definidas por el usuario, en el nodo correspondiente a su visualización. Sin embargo, existen variables que no se encuentran en el web form, pero sí se encuentran por ejemplo en el código de usuario. Para definir este tipo de variables se cuenta con el nodo variables.



Ahí será posible escoger entre una variable o un sdt.

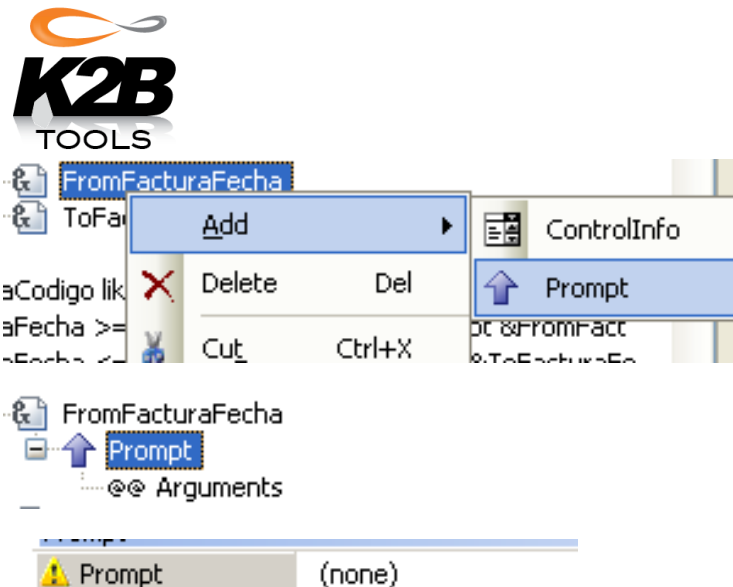


La forma de definir las variables es similar al resto de los nodos donde se definen variables. El value puede ser cualquier expresión a la derecha del igual de la variable que será ejecutada en el evento start.

En SDT solamente se debe colocar el nombre del SDT.

8.7 Propiedad Prompt

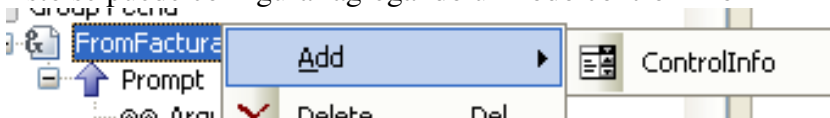
En todos los nodos del pattern, correspondiente a variables en las que se pueden ingresar datos, un ejemplo pueden ser los filtros, se encuentra con la propiedad prompt. Esta propiedad hace que el ingreso de ese campo pueda ser seleccionado mediante un prompt.



En la propiedad Prompt se configura el nombre del prompt.
 En caso de que no se agregue un nodo arguments, el prompt será invocado con el campo al que se le aplica. Sí se define el nodo arguments el prompt será invocado con los argumentos establecidos en ese nodo.

8.8 Control Info

Asociado a cualquier atributo o variable dentro de la pantalla es posible configurar cuál es el tipo de control info que se desea usar.
 Esto se puede configurar agregando un nodo control info





[-] General	
ControlType	Dynamic Combo Box
Suggest	No
NotifyContextChange	False
[-] Suggest	
InputType	Descriptions
FilterOperator	Begins with
CaseSensitive	False
SuggestMaxItems	5
[-] Dynamic Controls	
ItemValue	ClienteId
ItemDescription	ClienteNombre
InstantiatedAttribute	
Conditions	
SortDescriptions	True
[-] Dynamic (ListBox and ComboBox) Controls	
EmptyItem	True
EmptyItemText	GX_EmptyItemText
[-] Check Box	
ControlTitle	
CheckedValue	
UncheckValue	
[-] Radio Button	
RadioDirection	Vertical

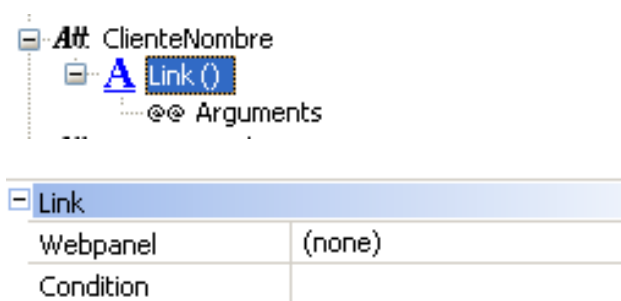
Las propiedades a configurar son las mismas propiedades que permite configurar GeneXus. Dentro de control type se pueden seleccionar todos los control type soportados por GeneXus.

ControlType	Dynamic Combo Box
Suggest	Check Box
NotifyContextChange	Combo Box
Suggest	Dynamic Combo Box
InputType	Dynamic List Box
FilterOperator	Edit
CaseSensitive	List Box
- ... -	Radio Button

8.9 Link de los atributos

Cada atributo o variable generada por el pattern, puede tener la posibilidad de ser puesto en pantalla con un link asociado. El pattern genera los links explícitamente o implícitamente. Implícitamente es cuando el atributo asociado tiene la propiedad autolink en true, en ese caso si el atributo es DA, genera automáticamente un link al entity manager correspondiente a la transacción donde el atributo es DA.

Si el atributo no es DA, se puede definir el link explícitamente, esto es utilizando el pattern agregando el nodo link.



En la propiedad link se puede explicitar el web panel al que se va a linkear y en la condition se puede colocar una condición a evaluar de cuándo se generará el link. En los argumentos se explicitarán los argumentos con los que se invocará a dicho panel.

8.10 Tips

8.10.1 Creación de grilla sin tabla base.

- 1) Poner en el nodo attributes en lugar de atributos variables.
- 2) Escritura del for each de carga de la tabla base: Para esto en el nodo correspondiente al objeto se debe hacer add events , add event y ahí seleccionar como evento el [MainGrid].Load. Ahí en EndCode colocar el código de carga de la grilla. Como sugerencia, escribir primero el código de carga en ese objeto GeneXus luego copiarlo y pegarlo en patterns en la Evento.
- 3) Acciones Ingrid: Patterns genera el código link de las acciones con Grid dentro del evento load. Debido a que es sin tabla base, este código deberá generarse dentro del foreach. Para esto el último paso es copiar dentro del código generado



por patterns el código correspondiente a la función link de las acciones en el evento grid.load y colocarlo en la loadCode dentro del for each

8.10.2 Acción que invoca a un evento de usuario.

Dentro del código de los eventos, es posible colocar eventos nuevos definidos por el usuario. Este evento es asignado a un control de la siguiente manera.

Si se agrega una acción de usuario sin setear el valor de GXobject, se puede hacer que al clicar en la imagen asociada a la acción se invoque al evento creando el evento &actionName.click.

8.10.3 Cambiar acción standard por acción de usuario

Se indica como hacer en patterns para que las acciones update, delete, display, insert, etc apunten a otros objeto y no a la transacción que es como va por defecto en patterns.

En patterns las acciones standard se definen a través del nodo modes. Estando esta propiedad en true Patterns por defecto genera código para la invocación de la transacción en los distintos modos para update, insert y delete, y el llamado al entity manager para el modo display.

Para hacer que estas acciones apunten a otros objetos se debe hacer lo siguiente:

- Poner en false, el modo que se desea personalizar
- Crear una action, cuya imagen sea igual a la imagen correspondiente al ícono de la acción.
 - En caso de querer personalizar las acciones de Update, Delete, Display, colocarle a la action Layout InGrid?
 - En caso de querer personalizar la acción de Insert crear una action de Layout Top(Rigth)
- En GXObject poner el objeto que se desea invocar y en parameters especificarle los parámetros.



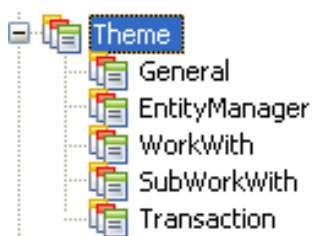
9 Customización interfaz K2BTools

La interfaz generada por el pattern K2BEntityServices puede ser 100% customizable por parte del usuario, para poder obtener el look & feel que se desee.

Cada tipo de objeto generado por el patrón es generado basado en un tema con determinadas imágenes y a partir de un objeto Layout que especifica la ubicación de los diferentes componentes dentro del web form.

9.1 Clases del tema

Todos los objetos generados por el patrón K2BEntityServices, son generados basados en una clase del tema. La clase se puede configurar en el settings, en el nodo theme.



Theme: Theme	
General	
Theme	K2B
SetObjectTheme	True

Ahí se puede configurar cuál es el tema y si el tema se le va a setear a los objetos, o cada objeto tomará el tema del modelo.

Adicionalmente en cada instancia se encuentra con la propiedad theme.

General	
HasOAV	False
AssociatedContext	
Theme	
Theme	(none)

De tener algún tema específico todos los objetos serán generados con esa clase del tema, sino se tomará lo que el usuario configuró en el settings.



Cada control, tabla, atributo, grilla, etc generado por el patrón está basado en alguna clase del tema. La clase del tema que toma cualquier elemento de la interfaz, es configurable a través del settings, y puede tomar diferentes clases del tema según el tipo de objeto que sea (puede tomar un tema si es un subworkwith y otro si es un workwith)
 Esta se puede configurar por tipo de objeto, tal cual se muestra en el nodo theme del settings. Dentro de cada nodo del settings se encuentran categorizados los diferentes tipos de control de los objetos.

ErrorViewer	
ErrorViewer	ErrorViewer
Button	
Button	K2BButton
BigButton	K2BBigButton
BigButtonLength	12
Grid	
Grid	K2BWorkWithGrid
FirstColumnResizeVal	50
TextBlock	
Title	TitleText
PageCount	Page
OrderCaption	GridOrderCaption
FixedDataBottomDes	FixedDataBottomText
FixedDataBottomLeft	LeftText
FixedDataBottomRight	RightText
FilterAttributeDescrip	Filter
FilterAttributeLeftText	LeftText
FilterAttributeRightText	RightText
AttributesSectionBottom	AttributeBoxTitle
AttributesSectionBottomLeft	LeftText
AttributesSectionBottomRight	RightText
AttributesSectionTop	Text
AttributesSectionTopLeft	LeftText
AttributesSectionTopRight	RightText

En este caso por ejemplo se encuentran categorías para el ErrorViewer.
 Para el caso de los botones siempre se le asigna al botón en pantalla la clase Button excepto cuando el botón supera el BigButtonLength. En tal caso se le asignará la clase K2BBigButton.



En los text blocks será posible configurar la clase del tema correspondiente a los diferentes textos que encontramos en pantalla. Title por ejemplo se refiere al title del workwith.

AttributesSectionRight	RightText
<input type="checkbox"/> Image	
Action	K2BActionImageTop
<input type="checkbox"/> Attribute	
AttributeInFixedData	FixedData
AttributeInBottomFix	FixedDataBottomAttribute
AttributeInFilter	SearchAttribute
DateAttributeInFilter	DateSearchAttribute
DateTimeAttributeInF	DateTimeSearchAttribute
OrderByComboBox	OrderByComboAttribute
ComboBox	ComboViewAttribute
AttributeInGrid	GridAttribute
ComboBoxInGrid	ComboAttributeInGrid
CheckBoxInGrid	CheckBoxInGrid
ComboBoxInFixedDal	ComboBoxFixedData
AttributesSectionBott	BoxAttribute
AttributesSectionTop	SearchAttribute

Dentro del tema se puede configurar qué clase tendrán las imágenes correspondiente a las acciones tops y para los atributos la clase seteada dependerá también de qué tipo de control y qué tipo de datos es el atributo que se genera. O sea por ejemplo si es Combo en una grilla se tomará la clase del tema ComboBoxInGrid, si es un CheckBox tomará la CheckBoxInGrid. Si el tipo de datos es una fecha y está en un filtro tomará la clase correspondiente a DateAttributeInFilter.

TableAttributesSection	
TableWebComponent	TableComponentContainer
TableFormSectionLeft	TableComponentContainer
TableAttributesSection	StandardTable
TableAttributesSection	
TableWebComponent	TableComponentContainer
TableFormSectionRight	TableComponentContainer
TableAttributesSection	StandardTable
TableAttributesSection	
TableWebComponent	TableComponentContainer
Group	
FilterGroup	FilterGroup
FixedDataGroup	FixedDataGroup
AttributesSectionTop	FixedDataVerticalGroup
AttributesSectionBottom	FixedDataVerticalGroup
AttributesSectionLeft	FixedDataVerticalGroup
AttributesSectionRight	FixedDataVerticalGroup

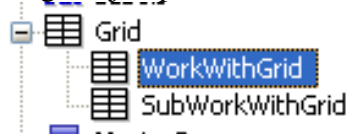
Similar será el comportamiento para las diferentes tablas generadas por el patrón, y los diferentes grupos. Si el grupo está en un filtro se tomará la clase FilterGroup y si está en la fixed data la clase FixedDataGroup.

9.2 Imágenes

Cada imagen que se utiliza puede ser configurada en el pattern k2bentityservices.

Si no se coloca ninguna imagen el pattern por defecto generará un botón.

Algún ejemplo son las imágenes del paginado



Paging Buttons	
FirstButtonCaption	First
FirstButtonTooltip	First
FirstButtonImage	K2BPageFirst
FirstButtonDisableImage	K2BPageFirstDisabled
LastButtonCaption	Last



Donde en el settings se configura qué imagen se toma para ese nodo.
Además de modificar esto se puede modificar la imagen misma, y cambiarla o en caso de que se tengan varios temas, hacerla que para otra clase del tema tome otra imagen haciendo que la imagen sea dependiente del tema.

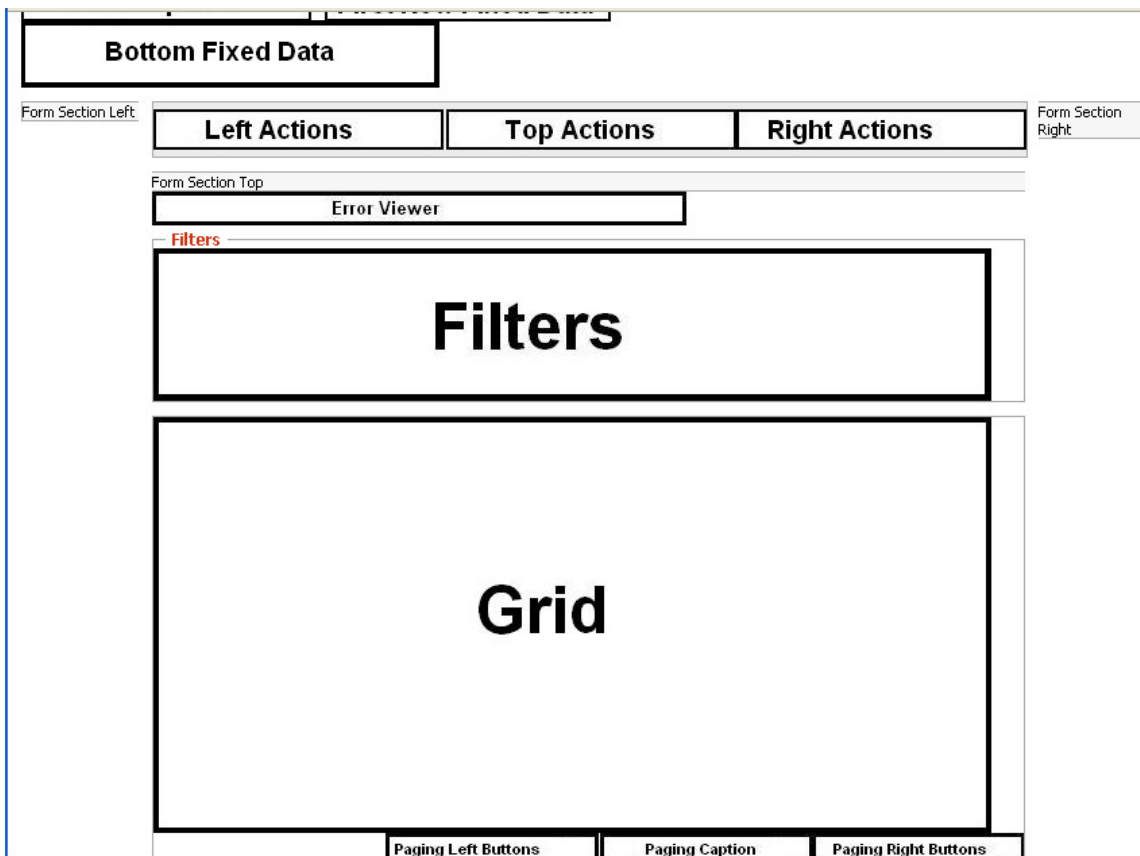
t ☒ Image varies with Themes

ieXusX	K2B
ctionDisplay	gridopen.gif (Default)
ize:16px,16px	Size:14px,14px

9.3 Objeto Layout

Cada tipo de objeto web panel que es generado por el patrón, toma su interfaz de un objeto layout. En este objeto layout se define toda la estructura de componentes de pantalla presente en la aplicación.

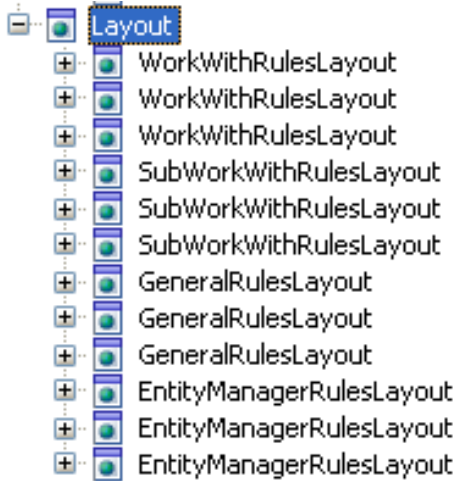
El Objeto Layout es un web panel que el pattern tomará como base para la generación de sus objetos.



En la figura de arriba es posible apreciar un ejemplo de objeto layout. En este ejemplo se indica dónde va a estar ubicada la grilla. A la hora de generar el objeto, el contenedor con nombre Grid será sustituido en tiempo de aplicación del patrón por la grilla conteniendo los atributos definidos en la instancia. Lo mismo se hará con los Filters y con todas las regiones presentes en la pantalla. El desarrollador tiene la flexibilidad de modificar esta estructura, para por ejemplo cambiar la presencia de filtros haciendo que el texto resultados filtrados y la barra que lo contiene no aparezca. Puede hacer que la grilla aparezca en otro lugar, que el paginado aparezca arriba, o especificar en qué lugar serán ubicados las diferentes form sections según el layout. Puede también reestructurar las tablas, para por ejemplo hacer que los objetos de interfaz se agrupen en otro tipo de tablas.

También se puede dar algo de comportamiento a estos objetos permitiendo la inclusión de eventos y definición de variables, las cuales serán agregadas automáticamente por todos los objetos que genera el patrón.

La configuración de los objetos Layouts se da en el archivo de settings. Para cada tipo de objeto generado por el patrón se puede especificar qué objeto de Layout utilizará. Esto se hace a través del nodo Layout



Layout: Layout	
Layout	
WorkWithLayout	WWLayout3
SubWorkWithLayout	SubWWLayout3
GeneralLayout	GeneralLayout3
EntityManagerLayout	EntityManagerLayout3

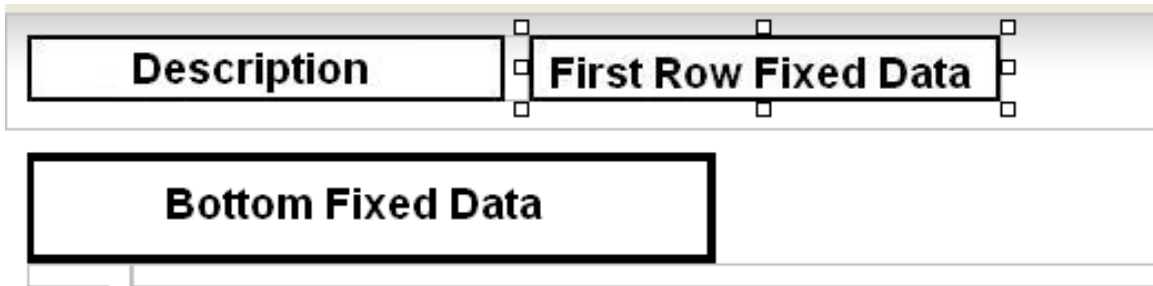
En el nodo Layout se colocará para cada tipo de objeto generado por el patrón, que Layout se utilizará.

9.3.1 Layout Web Form

La parte del web form cuenta con regiones específicas que serán reemplazadas a la hora de generar los objetos por el patrón. Al salvar el archivo de settings donde se le asignan los objetos de layout por tipo de objetos se dispara una validación a los efectos de validar que efectivamente todas las regiones que usa el pattern existan, en caso de no existir se disparará un mensaje de error.

Estas secciones están ubicadas dentro de un div. Enumeraremos aquí las diferentes regiones de div que deben estar presentes en cualquier objeto layout. Esto lo haremos para un Layout aplicado a un workwith. Lo que será reemplazado son los divs. En el WWLayout que viene por defecto en el patrón, adentro de cada div hay una imagen para que el usuario al verlo en el editor de web form reconozca fácilmente de qué trata cada parte de interfaz.

9.3.1.1 Título y Fixed Data



```
<div class="" id="K2BSectionPgmDescriptionContainer">
```



Especifica la ubicación del título del trabajar con. Ese div será sustituido por el título del trabajar con.

```
<div class="" id="K2BSectionFirstRowFixedDataContainer">
```



Será sustituido por la primera fila de la fixeda data.

```
<div class="" id="K2BSectionBottomFixedDataContainer">
```



Especifica donde estará ubicada las siguientes filas correspondientes a la fixed data.

9.3.1.2 Acciones y ordered by

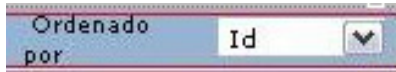


```
<div class="" id="K2BSectionActionsLeftContainer">
```



Especifica la ubicación de las acciones left.

```
<div class="" id="K2BSectionOrderedByContainer">
```



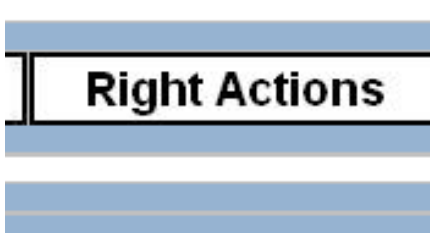
Especifica en qué ubicación estará el ordenado por.

```
<div class="" id="K2BSectionActionsTopContainer">
```



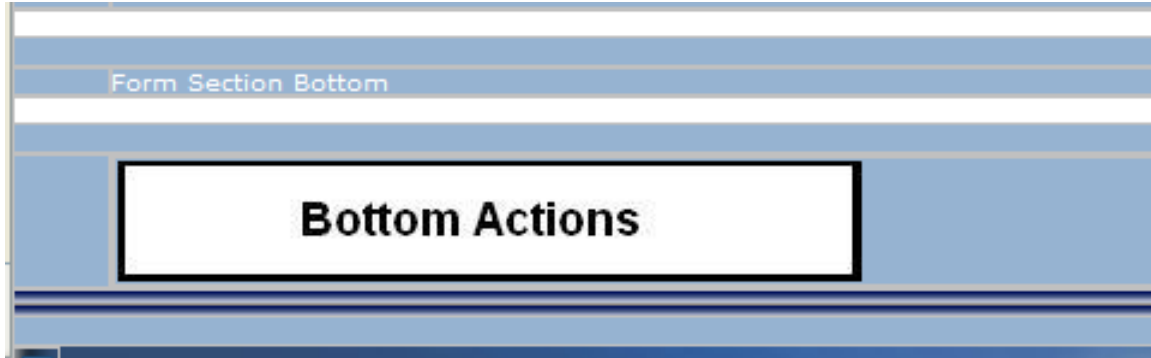
Especifica donde estarán las acciones con layout top.

```
<div class="" id="K2BSectionActionsRightContainer">
```

Especifica donde estarán ubicadas las acciones top right.

```
<div class="" id="K2BSectionActionsBottomContainer">
```



Especifica la ubicación de las acciones con layout bottom.

9.3.1.3 Filtros y Grilla

```
<div class="" id="K2BSectionFiltersContainer">
```



Especifica la ubicación de la sección de filtros dentro del layout.

```
<div class="" id="K2BSectionGridContainer">
```

Grid

Paging Left Buttons

Paging Caption

Paging Right Buttons

Especifica la ubicación de la grilla donde se mostrarán los atributos.

9.3.1.4 Paginado

Paging Left Buttons

Paging Caption

Paging Right Buttons

```
<div class="" id="K2BSectionPagingLeftContainer">
```

Paging Left Buttons

Especifica la ubicación de las flechas del paginado para ir a la página anterior.

```
<div class="" id="K2BSectionPagingPageCountContainer">
```

Paging Caption

Especifica la ubicación del texto que indica la página de la grilla que se está visualizando.

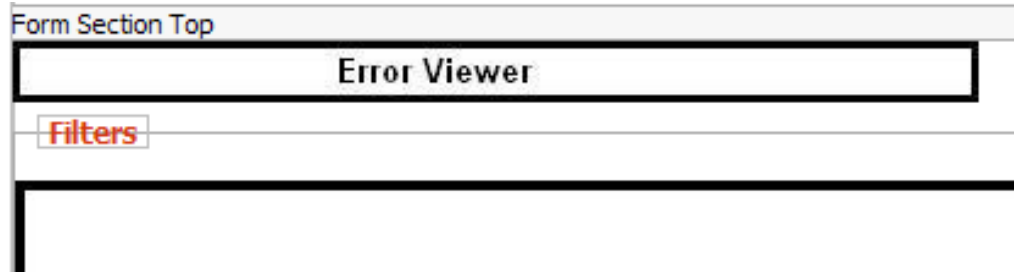
```
<div class="" id="K2BSectionPagingRightContainer">
```

Paging Right Buttons

Especifica la ubicación de las flechas del paginado para ir a la página siguiente.

9.3.1.5 Otras secciones

```
<div class="" id="K2BSectionErrorViewerContainer">
```



Especifica la ubicación del error viewer en el workwith.

```
<div class="" id="K2BSectionHiddenItemsContainer">
</div>
```

Especifica en qué lugar del HTML serán colocados los campos no visibles dentro del objeto.

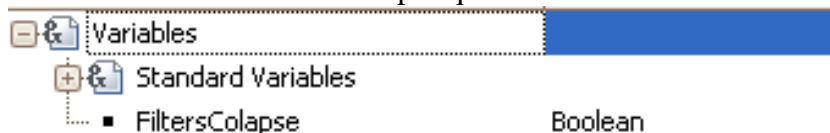
```
<div class="" id="K2BSectionFormSectionTopContainer">
<div class="" id="K2BSectionFormSectionLeftContainer">
<div class="" id="K2BSectionFormSectionRightContainer">
<div class="" id="K2BSectionFormSectionBottomContainer">
```

Especifica la ubicación de las diferentes form sections.

9.3.2 Layout Eventos y variables

Es posible en los objetos de layout definir variables y código de los eventos. Estas variables serán definidas para todos los objetos generados y el código será incluido en todos los objetos. Esto es fundamental para que se puedan customizar en un 100% los objetos generados por el pattern K2BEntityServices.

En este ejemplo se muestra la implementación de la acción de colapsar los filtros. En las variables se define FiltersCollapse que indica si los filtros están colapsados o no.



En los eventos se implementa la acción de colapsar

```

] Event Start
  //+ CollapseFiltersInitialization
  &FiltersCollapse = False
  &FiltersCollapse.Visible = False
  //- CollapseFiltersInitialization
- EndEvent

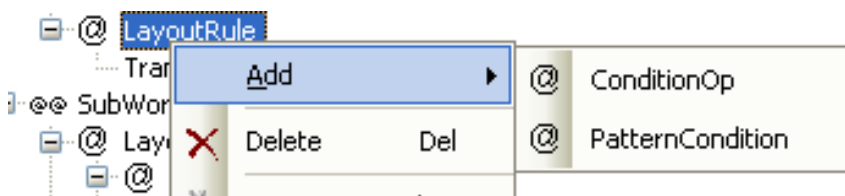
  //+ CollapseFilters
] Event 'CollapseFilters'
] if (&FiltersCollapse = True)
  K2BTableFiltersContainer.Visible = True
  &FiltersCollapse = False
  K2BHideFilters.Class = "K2BButtonUp"
Else
  K2BTableFiltersContainer.Visible = False
  &FiltersCollapse = True
  K2BHideFilters.Class = "K2BButtonDown"
- EndIf
- EndEvent
  //- CollapseFilters

```

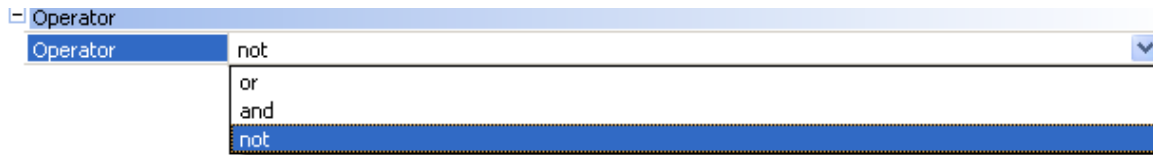
9.3.3 Layout Rules

El objetivo de este elemento del settings, es dar la posibilidad de que bajo determinadas condiciones, no se tomen en cuenta determinadas secciones del objeto Layout, por ejemplo si en el objeto a generar no hay filtros, que no se incluya el texto de resultados filtrados en el web form, y que el código correspondiente a colapsar los filtros tampoco sea generado.

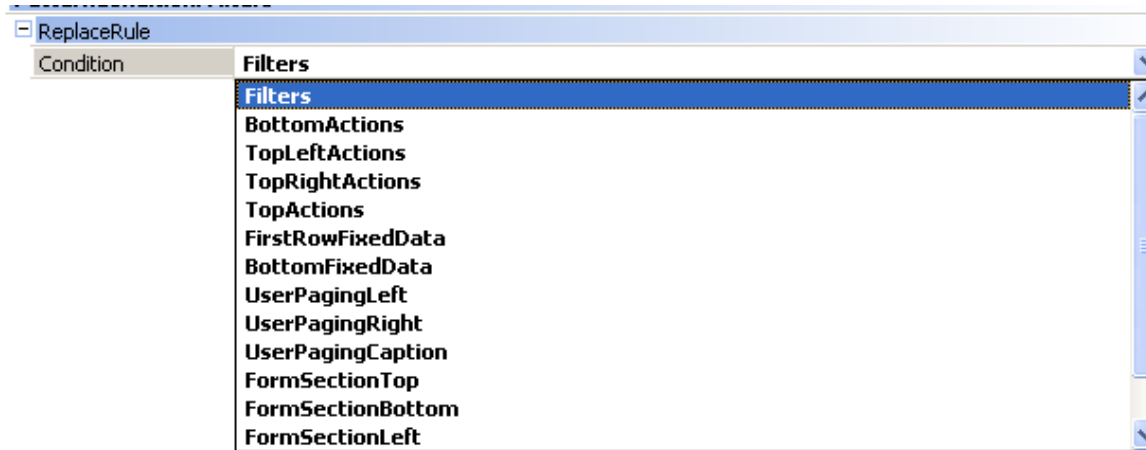
Para esto bajo cada objeto de Layout se encuentra el nodo Layout Rule.



Dentro del LayoutRule se puede agregar un condition operator o un pattern condition.

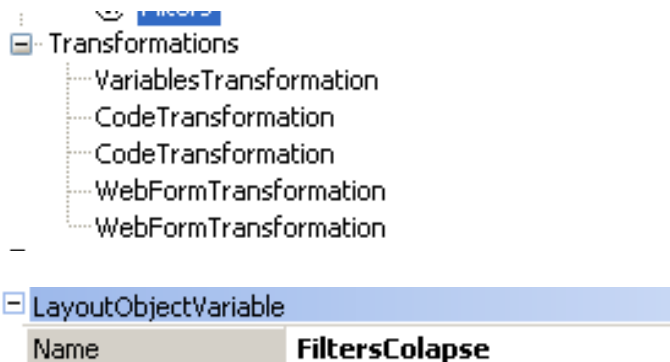


Los operadores posibles a agregar son el not or y and. Ese operador puede ser aplicado a un pattern condition.



Las condiciones se refieren a la presencia de determinado elemento en el web form. Por ejemplo Filters evalúa en true cuando hay filtros, FormSectionTop evalúa en true cuando hay en la instancia un form section top.

De evaluarse la regla en verdadero, entonces se efectuarán una serie de transformaciones dentro del código del objeto Layout especificadas en el nodo transformation que se encuentra como hijo de cada una de las reglas.



En el nodo Variables Transformation se coloca el nombre de la variable que al cumplir la regla en ese nodo de la instancia el patrón no debería definirla.



Un ejemplo es evitar de definir la variable filter collapse cuando no hay filtros.

El nodo Code Transformation permite especificar el nombre del slot que no sería incluido en los objetos generados, y en ApplyParts se especificará si esa regla se aplicará a todas las partes del objeto, solo a los eventos, o reglas.

Es importante ser lo más específico posible en estos casos de forma tal de evitar que el patrón haga parseos en otras regiones. O sea que si de antemano se sabe que en el objeto Layout no existe ese slot en las reglas, lo mejor es poner que aplica solo a eventos.

WebFormTransformation: WebFormTransformation	
ReplaceRule	
Tag	tr
TagId	Filters
HTMLReplaceCode	

WebFormTransformation permite reemplazar secciones de HTML por otras. En tag se coloca el tipo de elemento que se desea reemplazar, en TagId se coloca cuál es el id de algún elemento dentro de ese tag y en HTMLReplaceCode se coloca el código con el cuál se va a querer reemplazar la sección. El reemplazo es de toda esa sección.

Para el ejemplo que se muestra en la figura en caso de no haber filtros, desaparecerá el tr que contiene id Filters.

Mediante estas reglas acompañado de los objetos de layout es posible adaptar 100% la interfaz que genera el pattern K2BEntityServices, para que estos patterns sean adaptados a la customización que el desarrollador necesita.

9.4 Pasos para modificar la interfaz generada por las K2BTools

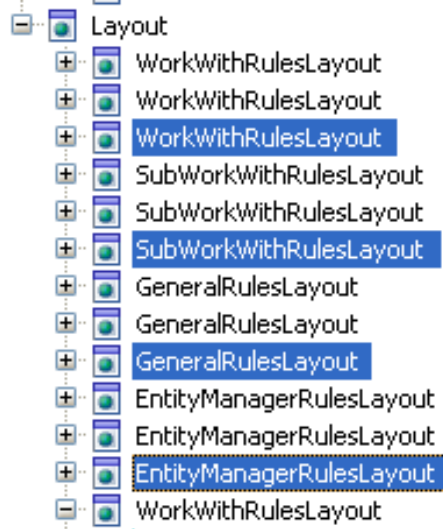
Parte I Puesta a punto:

1. Preparar tema para hacer modificaciones
 - a. Hacer un save as del tema K2B3 (K2BNuevo)
 - b. Setear como tema default del modelo K2BNuevo.
2. Preparar header y master page para customizarlas
 - a. Ir a la carpeta K2BTools-> K2BEntityServicesBasicObjects ->User->Components
 - b. Hacer un save as de K2BHeader3 (K2BHeaderNuevo)
 - c. Hacer un save as de K2BRecentLinks3(K2BRecentLinksNuevo)
 - d. Hacer un save as de K2BWWMasterPage3(K2BWWMasterPageNuevo)



- e. Abrir el objeto K2BWWMasterPageNuevo y en el evento refresh en la línea WCHeader.Object = Create(...) cambiar el nombre K2BHeader3 por K2BHeaderNuevo.
 - f. Abrir el objeto K2BHeaderNuevo y donde referencia a K2BRecentLinks3, cambiar esa referencia por K2BRecentLinksNuevo
 - g. Setearle a los objetos K2BWWMasterPageNuevo, K2BHeaderNuevo y K2BRecentLinksNuevo en las propiedades el tema K2BNuevo.
3. Preparar TabbedView para ser customizado
 - a. Ir a la carpeta K2BTools->K2BEntityServicesBasicObjects->TabbedView y hacer un save as de K2BTabbedView3 a K2BTabbedViewNuevo.
 - b. Setearle el tema K2BNuevo.
4. Preparar objetos de Layout para ser modificados
 - a. Ir a la carpeta K2BTools-> K2BEntityServicesBasicObjects ->User->Components->ObjectsLayout
 - b. Hacer un save as del objeto WWLayout3 (WWLayoutNuevo)
 - c. Hacer un save as del objeto EntityManagerLayout3 (EntityManagerLayoutNuevo)
 - d. Hacer un save as del objeto GeneralLayout3 (GeneralLayoutNuevo)
 - e. Hacer un save as del objeto SubWWLayout3 (SubWWLayoutNuevo)
 - f. Ir al K2BEntityServices settings
 - g. Seleccionar los nodos WorkWithRulesLayout, SubWorkWithRulesLayout, GeneralRulesLayout y EntityManagerRulesLayout correspondiente al Layout 3

Layout	
Layout Object	WWLayout3
Has Events Code	True
Has Variables	True
Master Page	
Master Page	K2BWWMasterPage3



- h. Copiar y pegar bajo el nodo Layout.
- i. Configurar los Layout Objects de las reglas recientemente pegadas
 - i. En WorkWithRulesLayout colocar como LayoutObject WWLayoutNuevo y setear como master page WWMasgerPageNuevo
 - ii. En el EntityManagerRuesLayout colocar como LayoutObject EntityManagerLayoutNuevo y la master page WWMasterPageNuevo y en la propiedad TabbedViewWebComponent el objeto K2BTabbedViewNuevo
 - iii. En los SubWorkWithRulesLayout colocar como Layout Object SubWWLayoutNuevo
- j. Ir al nodo raíz Layout y configurar los objetos de Layout que va a usar el pattern






- i. WorkWithLayout = WWLayoutNuevo
- ii. SubWorkWithLayout = SubWWLayoutNuevo
- iii. GeneralLayout=GeneralLayoutNuevo
- iv. EntityManagerLayout=EntityManagerLayoutNuevo

Parte II Customizaciones de imágenes:

Se comenzará por customizar las imágenes que vienen por defecto en k2btools para adaptarlas a las que se deseen



1. En GeneXus, en FolderView ir a Customization->Images
2. En el filtro, escribir K2B, para ver solo las imágenes que comienzan con K2B. A continuación se modificarán las imágenes que standards del pattern. La forma de hacerlo es:
 - a. Seleccionar el objeto image a cambiar.

Themes			
	K2B	K2B2	K2B3
All Languages	 K2BActionInsert.gif (Default) Size:20px,18px	 actionnew.gif Size:16px,16px	 actionnew.gif Size:16px,16px

b. Hacer click en el tema K2BNuevo y seleccionar la imagen adecuada. Muchas de las imágenes que aparecen en el tema, son imágenes auxiliares, para que se puedan usar en acciones. Aquí se listarán las imágenes relevantes a modificar dentro de las k2btools.

- K2BActionExport: Exportación a Excel en el trabajar con.
- K2BActionDisplay: Visualizar en la grilla del trabajar con.
- K2BActionDelete: Eliminar en la grilla del trabajar con
- K2BActionInsert: Insertar en trabajar con.
- K2BActionRefresh: Refresh en el trabajar con.
- K2BActionReport:Repote pdf en el trabajar con.
- K2BActionUpdate:Actualizar en la grilla del trabajar con.
- K2BButton: Botón para las acciones standard que por defecto se muestra con botón(botones en la transacción, botón update y delete en el tab general, botones para las acciones).
- K2BButtonBig: Cuando el caption supera determinado valor se usa el botón grande en lugar del chico.
- K2BPageFirst: Ir a primera página
- K2BPageFirstDisabled: Ir a primera página deshabilitado.
- K2BPageLast: Ir a última página.
- K2BPageLastDisabled: Ir a última página deshabilitado.
- K2BPageNext: Próxima página
- K2BPageNextDisabled: Próxima página deshabilitada.
- K2BPagePrevious: Página anterior.
- K2BPagePReviousDisabled: Página anterior deshabilitada.

Otras imágenes:

Se puede querer modificar otras imágenes que en la versión standard de las k2btools se muestran como botones. Para esto es posible realizar las siguientes modificaciones.

- Modificar y Eliminar en el objeto general.

Product :: Bread

General Product Storages

Product Id
 Product Description [Bread](#)
 Product Price 42.00

Update **Delete**

- Ir en el settings al nodo standard actions y en tabular update y tabular delete se pueden configurar el caption de la acción o asociar un objeto image para que en lugar de usar un botón, use una imagen.
- Botones de confirmar, verificar o eliminar de la transacción.

Product

General Product Storages Product Units

Id 0
 Description
 Type **Good**
 Price
 Observations

Insert **Cancel**

- En el nodo Theme->Transaction se pueden modificar las clases asociadas a estas acciones, modificando en Button_Confirm, Button_Check y Button_Cancel.
- Imagen de search:

Filters

Description

- En el Settings en el nodo Grid->WorkWithGrid y Grid->SubWorkWithGrid bajo la categoría Search Button, tienen la propiedad searchbutton caption y searchbuttonimage. En caso de que no se desee que sea un botón, pueden configurar el valor de la propiedad searchbuttonimage.

Parte III Customizacion de tema K2B:



Se recomienda para esta parte, como herramienta de apoyo utilizar el firebug, complemento de firefox que te permite dado un campo visualizar que clase del tema posee y hasta modificarlo en tiempo de ejecución.

Clases importantes:

- Grid/GridCustom: Se configura la visualización de la grilla de los trabajar con y subworkwith
- Grid/Grid_Transaction: Se configura la visualización de la grilla de las transacciones.
- Attribute/AttributeGrid: Visualización de los atributos dentro de la grilla
- Attribute/AttributeTrn: Visualización de los atributos dentro de la transacción
- Attribute/AttributeTabular: Visualización de atributos en vista tabular. (ver hijos: fixed data, etc)
- TextBlock->TextBlockTabular: Visualización de descripciones en vistas planas (ver hijos)

Parte IV Customizacion de objetos de Layout:

Para esto se recomienda seguir el punto 9.3.1

Parte V Customizacion de headers y master page:

Se puede modificar el objeto K2BWWMasterPageNuevo como el usuario desee pero teniendo en cuenta las siguientes consideraciones. Dado que en la master page está configurada la navegación de la aplicación y la inclusión de javascripts a los objetos tener en cuenta lo siguiente:

- En el evento start de la master page siempre incluir el K2BTools.js
`Form.JScriptSrc.Add("K2BTools.js")`
- En el Header incluir el componente de recentlinks o en caso de no desear visualizar los recent links agregar en el evento start el siguiente código
`Call(K2BStackManagement, &FormCaption, &includeInStack, &Stack)`



10 Seguridad

En esta versión se cuenta con una única API de invocación a seguridad. El conjunto de parámetros con los que se invoca a esta API es llamado actividad. Cada uno de los objetos generados por el pattern invocan a la API de seguridad, para controlar el acceso al objeto y visualización de controles de pantalla.

La idea es que el usuario del pattern implemente su API de seguridad, integrándola a la seguridad ya existente, implementando una nueva, o integrándose a alguna herramienta externa, por ejemplo GXPortal.

La configuración de seguridad es definida por transacción. Dentro de las propiedades de la transacción hay una nueva categoría cuyo nombre es K2BTools. Ahí se encuentra con la propiedad Security, en caso de estar en false, el pattern no generará la invocación a la API de seguridad.

Specified Date	05/12/2008 12:
<input checked="" type="checkbox"/> K2B Tools	
Security	True
EntitySecurityName	Empresa

El nombre entity security name, sirve para poder definir seguridad a nivel de un conjunto de transacciones con valor semántico común, en lugar de a traves de una transacción sola. El ejemplo claro es cuando tenemos la transacción de Producto, Producto Unidad, ProductoClase, etc, y sabemos que si tengo permisos para dar de alta un producto puedo dar de alta el resto y viceversa. Para esto existe la posibilidad de definir un agrupador de transacciones que agrupa todas las transacciones en un mismo nombre de seguridad. Modificando a mano esta propiedad se puede setear el nombre que el usuario desee. En próximas versiones se contará con un nuevo tipo de objeto Entidad, para darle un valor semántico adicional a la base de conocimiento. En caso de que la transacción este adentro de una entidad, la propiedad EntitySecurityName se actualizará con el nombre de la entidad. Como se verá más adelante, el nombre de la entidad será uno de los parámetros con los que se invocará a la API de seguridad, en caso de querer usar la entidad como nombre de seguridad , en la implementación de la API usar ese parámetro.

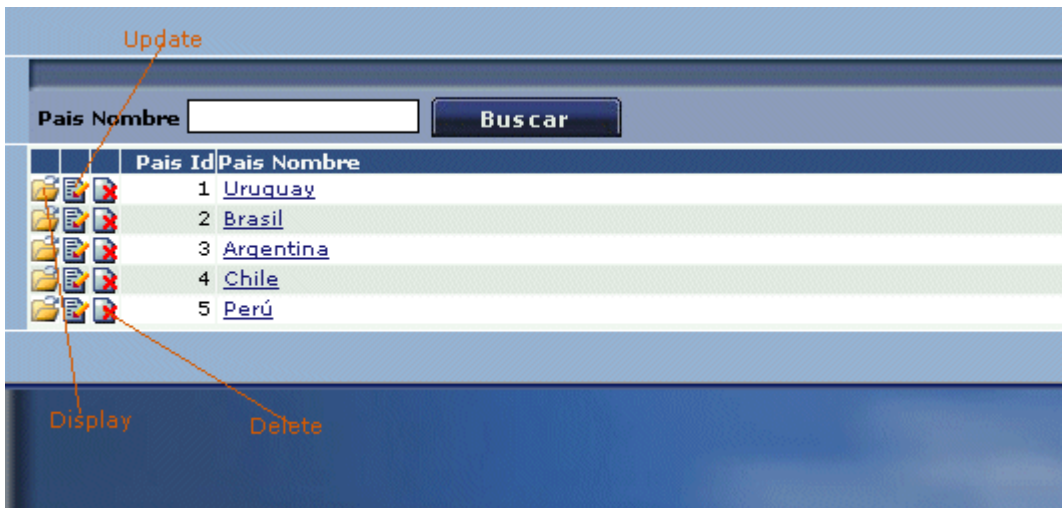
10.1 Actividades dentro de una instancia

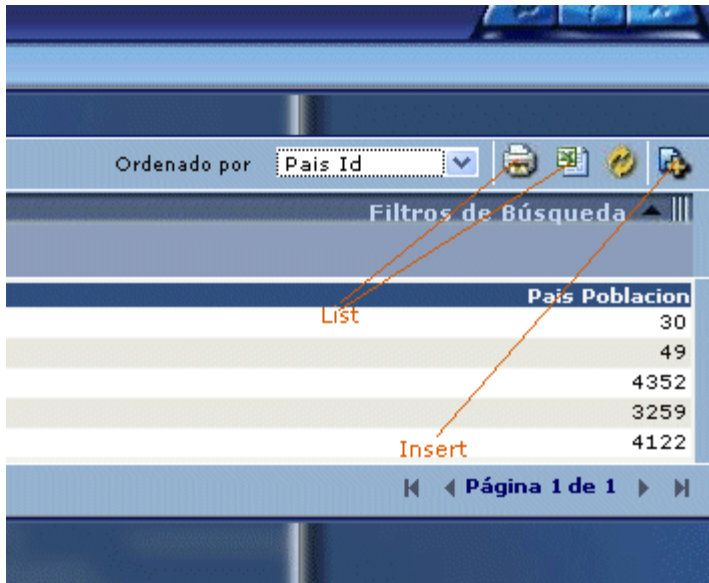
El Pattern K2BEntityServices se encarga de manejar el acceso a actividades relacionadas con operaciones en la transacción base de la instancia. Estas son:

- Posibilidad de dar de alta un registro. (Insert)
- Posibilidad de actualizar un registro. (Update)
- Posibilidad de eliminar un registro. (Delete)
- Posibilidad de visualizar un registro. (Display)
- Posibilidad de visualizar los registros relacionados.
- Posibilidad de ver un conjunto de registros en forma de listado. (List)

El Pattern K2BEntityServices maneja el acceso y la implementación de estas actividades. De esta forma podría chequear en forma conjunta todas estas actividades.

La idea es que para cada actividad que no está habilitada se inhabilitaría no solo el objeto que implementa esa actividad sino que también se inhabilita el control que desea acceder a ese objeto.





Dentro de las actividades es posible distinguir entre dos tipos de actividades.

Actividades de mantenimiento y actividades de visualización.

Las actividades de mantenimiento (Maintenance) están compuestas por:

- Insert
- Update
- Delete

Las actividades de visualización (Visualization) estarán compuestas por:

- Display
- List

Es posible en la implementación de seguridad agrupar las actividades de Insert, Update y Delete bajo una actividad de mantenimiento y las de Display y List en actividades de visualización.

Para realizar esta implementación, si el proc de seguridad recibe el tipo de actividad que se desea realizar(insert, update, delete) y la transacción, alcanzaría para implementar el mecanismo de seguridad presentado. De todas formas para permitir más flexibilidad y poder adaptarse a cualquier mecanismo de seguridad que sea necesario usar, la API de seguridad también recibirá el nombre de programa.

Además se podrá chequear la seguridad para acciones que no maneja el pattern, esto es acciones de usuario. La API de seguridad se describirá a continuación.



10.2 API de seguridad

Dentro de los objetos básicos del Pattern, en la carpeta User/Security se encuentra con un conjunto de procedimientos para controlar la seguridad.



K2BNotAuthorized es un web panel que se visualizará cuando el usuario no tiene permisos para realizar determinada actividad.

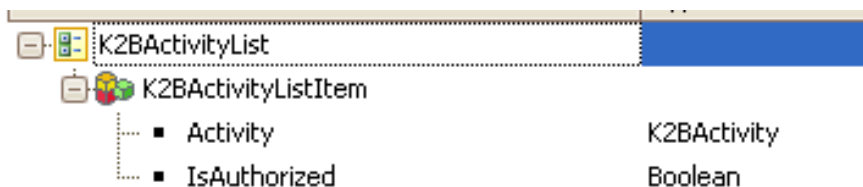
De los procedimientos que aparecen, todos son invocadores al procedimiento principal de seguridad que es K2BIsAuthorizedActivityList. Este es el procedimiento que el usuario deberá implementar para integrar la seguridad a su sistema. Por defecto el procedimiento está implementado para siempre devolver verdadero.

El procedimiento K2BIsAuthorizedActivityList recibe un sdt de tipo Collection con un conjunto de actividades y devuelve por cada una de estas verdadero o falso. La razón de que sea una colección es debido a que muchas veces se desea minimizar la invocación a la API de seguridad, pues esta puede insumir tiempo, dado que hemos encontrado implementaciones en las que por ejemplo invocan a algún web service.

Como ya se dijo el proc recibe el siguiente parámetro.

```
parm(InOut: &activityList);
```

El SDT K2BActivityList presenta la siguiente estructura.



Por cada actividad el proc setea si está autorizada a ser realizada o no. La actividad está representada por la siguiente estructura.



K2BActivity		K2 BActivity
EntityName	ObjectName	Entity Name
TransactionName	ObjectName	Transaction Name
PgmName	ObjectName	Pgm Name
StandardActivityType	K2BStandardActivi...	Standard Activity ...
UserActivityType	ObjectName	User Activity Type

En la seguridad se desea determinar si un usuario tiene permisos a realizar un tipo de actividad, que tiene una transacción asociada dentro de una entidad y que permite acceder o realizar determinada operación en un objeto.

Por ejemplo: Se desea saber si un usuario tiene permisos para realizar un insert en la transacción de paises donde se accede a la transacción País.

Tipos de actividad:

El tipo de actividad está diferenciada si es una actividad standard, o una actividad de usuario. Las actividades standards son Insert, Update, Delete, Display y List. Las actividades de usuario no son actividades standards y se diferencian a través de un nombre. El StandardActivityType está basado en un dominio enumerado que tiene los siguientes valores.

Name	Description	Value
Insert	Insert	Insert
Update	Update	Update
Delete	Delete	Delete
Maintenance	Maintenance	Mainten...
Visualization	Visualization	Visualiz...
Display	Display	Display
List	List	List
None	None	None

Buttons: Add, Remove, Edit, Ok, Cancel

En caso de que el valor sea None se entenderá que se trata de una actividad de usuario y en UserActivityType se colocará el nombre de la actividad de usuario.

Esta seguridad será controlada a nivel tanto de objeto como de acción.



Indicaremos ahora como se invoca a los programas de seguridad dentro de los objetos generados por el pattern para que se tenga una mayor idea de cómo funciona.

10.2.1 Invocación API en transacción

El Procedimiento K2BIsAuthorizedActivityName recibe como parámetros los campos del SDT, se encarga de armar el SDT agregarlo a la colección e invocar a la API de seguridad.

El Proc tiene los siguientes parámetros:

```
parm(In:&EntityName, In:&TransactionName, In:&StandardActivityType,
     In:&UserActivityType, In:&ProgramName, Out:&IsAuthorized);

Event Start
//+ K2BES.Security.Fixed
Do Case
  Case &Mode = K2BTrnMode.Insert
    &StandardActivityType = K2BStandardActivityType.Insert
  Case &Mode = K2BTrnMode.Update
    &StandardActivityType = K2BStandardActivityType.Update
  Case &Mode = K2BTrnMode.Delete
    &StandardActivityType = K2BStandardActivityType.Delete
  Case &Mode = K2BTrnMode.Display
    &StandardActivityType = K2BStandardActivityType.Display
EndCase
K2BIsAuthorizedActivityName.Call(!"Pais",!"Pais", &StandardActivityType, "", &PgmName, &IsAuthorized)
If &IsAuthorized = False
  K2BNotAuthorized.Call(!"Pais",!"Pais", &StandardActivityType, "", &PgmName)
EndIf
//-
```

Como se ve, se invoca a la transacción según la actividad que se desea realizar.

10.2.2 Invocación API objeto con grilla

El WorkWith así como cualquier objeto con grilla, implementa determinada actividad así como también brinda el acceso a otras. La actividad que implementa es la de listado. En caso de no estar autorizado a la realización del listado se invocará al objeto K2BNotAuthorized para informar del error.

Si se tiene acceso también hará invisible aquellos controles que desean acceder a actividades a las que el usuario no tiene acceso. De esta forma el WorkWith cargará la lista de actividades e invocará a la API de seguridad como se muestra a continuación. Primero carga las actividades.



```
-----  
    &ActivityList = new K2BActivityList()  
    &ActivityListItem = new K2BActivityList.K2BActivityListItem()  
    &ActivityListItem.Activity.EntityName = !"Pais"  
    &ActivityListItem.Activity.TransactionName = !"Pais"  
    &ActivityListItem.Activity.StandardActivityType = K2BStandardActivityType.List  
    &ActivityListItem.Activity.PgmName = &pgmName  
    &ActivityList.Add(&ActivityListItem)  
    &ActivityListItem = new K2BActivityList.K2BActivityListItem()  
    &ActivityListItem.Activity.StandardActivityType = K2BStandardActivityType.Display  
    &ActivityListItem.Activity.EntityName = !"Pais"  
    &ActivityListItem.Activity.TransactionName = !"Pais"  
    &ActivityListItem.Activity.PgmName = !"EntityManagerPais"  
    &ActivityList.Add(&ActivityListItem)  
    &ActivityListItem = new K2BActivityList.K2BActivityListItem()  
    &ActivityListItem.Activity.StandardActivityType = K2BStandardActivityType.Update  
    &ActivityListItem.Activity.EntityName = !"Pais"  
    &ActivityListItem.Activity.TransactionName = !"Pais"  
    &ActivityListItem.Activity.PgmName = !"EntityManagerPais"  
    &ActivityList.Add(&ActivityListItem)  
    &ActivityListItem = new K2BActivityList.K2BActivityListItem()  
    &ActivityListItem.Activity.StandardActivityType = K2BStandardActivityType.Delete  
    &ActivityListItem.Activity.EntityName = !"Pais"  
    &ActivityListItem.Activity.TransactionName = !"Pais"  
    &ActivityListItem.Activity.PgmName = !"EntityManagerPais"  
    &ActivityList.Add(&ActivityListItem)
```

Luego chequea la seguridad para la actividad de listado y en caso de no estar implementada se direccionará a el objeto k2bnotauthorized.

```
K2BIsAuthorizedActivityList.Call(&ActivityList)  
If not &ActivityList.Item(1).IsAuthorized  
    K2BNotAuthorized.Call(&ActivityList.Item(1).Activity  
EndIf
```

Luego por cada acción a la que no se tiene acceso de inhabilitará



```
&Display.FromImage(K2BActionDisplay)
&Display.TooltipText = "Abrir"
] If not &ActivityList.Item(2).IsAuthorized
    &Display.Visible = False
- EndIf
&Update.FromImage(K2BActionUpdate)
&Update.TooltipText = "Modificar"
] If not &ActivityList.Item(3).IsAuthorized
    &Update.Visible = False
- EndIf
&Delete.FromImage(K2BActionDelete)
&Delete.TooltipText = "Eliminar"
] If not &ActivityList.Item(4).IsAuthorized
    &Delete.Visible = False
- EndIf
..
```

En caso de existir acciones de usuario se cargará la acción con el standardactivitytype en None, y en UserActivityType se pasará el activityName de la acción como se muestra a continuación

```
//-
&ActivityList = new K2BActivityList()
&ActivityListItem = new K2BActivityList.K2BActivityListItem()
&ActivityListItem.Activity.StandardActivityType = K2BStandardActivityType.None
&ActivityListItem.Activity.UserActivityType = !"Activacion"
&ActivityListItem.Activity.EntityName = !"Proveedor"
&ActivityListItem.Activity.TransactionName = !"Proveedor"
&ActivityListItem.Activity.PgmName = !"PactivarProveedores"
&ActivityList.Add(&ActivityListItem)
```

10.2.3 Alta de actividades

Dado que el Pattern K2BEntityServices tiene el conocimiento de todas las actividades por las que chequea la seguridad, este genera un procedimiento genérico K2BLoadActivityList, que retorna una colección de actividades. El que utilice la API de carga de actividades podrá implementar un procedimiento para usando esta colección darlas de alta en la estructura correspondiente.

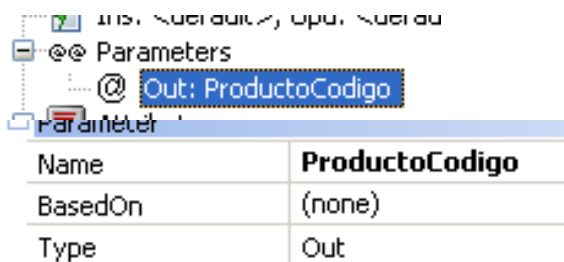
11 Pattern K2BPrompt

El pattern K2BPrompt es un pattern especializado en generar los prompts asociados a una transacción. La estructura es similar a la de los objetos WorkWith del Pattern K2BEntityServices, permitiendo la inclusión de formsections, acciones, eventos, etc. En los filtros es posible especificar grupos, columnas, etc. Toda esta funcionalidad está presente en el pattern K2BPrompt.

El manejo es similar salvo algunas diferencias que pasaremos a destacar, que están asociadas a propiedades particulares para el prompt.

Nodo Parameters:

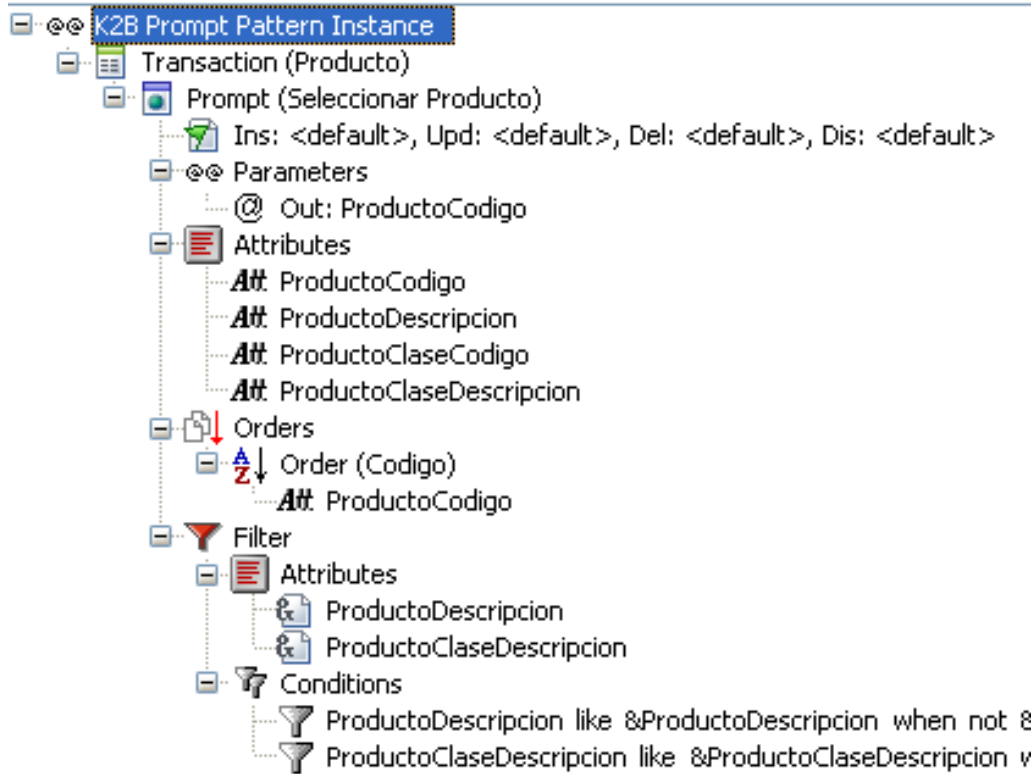
En el nodo parameters, aparece la propiedad type, donde es posible especificar si el objeto es solo de In, Out o InOut.



Name	ProductoCodigo
BasedOn	(none)
Type	Out

ReturnOnClick:

El pattern infiere por qué atributo realizar return on click. El atributo por el que realiza el return on click es si el atributo DA está en la grilla es ese atributo, sino será el primer atributo visible que aparezca en la grilla.



Para invocar al prompt se debe hacer uso de la regla prompt.

| `Prompt (PromptProducto, ProductoCodigo) ;`

Datos de Factura **AKB2345**

General

Codigo AKB2345
Fecha 09/12/08
Cliente ClientePrueba

Línea

Producto		
Cinta Adhesiva	↑	Cinta
CocaCola	↑	Coca
Agua Nativa	↑	Agua
	↑	

Filtros de búsqueda

Descripción
Producto Clase

Codigo	Descripción	Producto Clase
Aqua Nativa	Agua Nativa	Comestible
Aqua Salud	Agua Salud	Comestible
Aqua Sirte	Agua Sirte	Comestible
Aqua2	Agua2	Comestible
Cinta Adhesiva	Cinta Adhesiva	Papelería
CocaCola	CocaCola	Comestible
Coke	Coke	Comestible
Cubo Rubbik	Cubo Rubbik	Juguete
Fanta	Fanta	Comestible
Galletitas María	Galletitas María	Comestible

Cancel

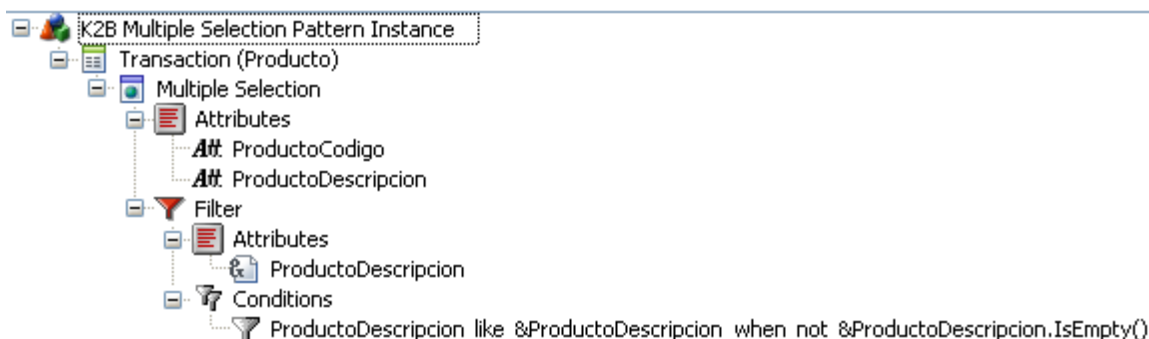
Aceptar **Cancelar**

12 Pattern K2BMultiple Selection

El Pattern Multiple Selection es un simple pattern que se encarga de generar el servicio de selección de atributos asociado a una entidad. Este objeto es un objeto básico que puede ser utilizado por el usuario en diversos contextos que requiera la invocación a ese panel.



La estructura de la instancia es simple:



El nodo atributos permite configurar cuáles son los atributos que se van a mostrar en ambas grillas, y el nodo filtros los filtros asociados a la grilla de los posibles ítems de la transacción a seleccionar. En base a esto se genera el panel y el sdt donde se almacenará en sesión los ítems seleccionados.

12.1 Uso del objeto generado por el pattern múltiple Selection:



El objeto generado por el pattern múltiple selection, recibe por parámetro una variable de tipo `&ExecutionContext`, que representa el contexto de ejecución del objeto. Esta clave es utilizada en sesión para almacenar el sdt de los ítems seleccionados.

```
K2BContextualSessionGet.Call(&ExecutionContext, !"SelectedItems",  
&XmlSelectedItems)
```

```
K2BContextualSessionSet.Call(&ExecutionContext, !"SelectedItems",  
&SelectedItems.ToXml())
```

```
K2BContextualSessionSet.Call(&ContextualContext, !"ProductoCodigoList",  
&ProductoCodigoList.ToXml())
```

Los métodos `K2BContextualSessionSet` sirven para almacenar dentro de la web session para ese contexto de ejecución la información de los ítems seleccionados. Para obtenerlos se utiliza el método `K2BContextualSessionGet` y esto es obtenido con el valor de clave `SelectedItems`.

También el objeto de múltiple selection almacena y lee en una variable de tipo collection la lista de ids de ítems seleccionados. Esto es a los efectos de poder filtrar la grilla de la izquierda por los elementos que ya fueron seleccionados.

Para invocar al objeto llamador, es posible:

- 1) Configurar el estado inicial de la grilla de ítems seleccionados.
- 2) Configurar la colección de la grilla de ítems seleccionados.
- 3) Leer los ítems seleccionados para realizar alguna acción.

Mostraremos un breve ejemplo de cómo configurar esto para el caso de un Nexó.

En este ejemplo se va a utilizar el multiple selection de productos para asociarle los productos seleccionados al proveedor.

En primer lugar el panel llamador va a recibir como parámetro el `ProveedorCodigo`. Lo primero es obtener un contexto de ejecución para el objeto multiple selection que vamos a crear en nuestro web panel.

Esto se hace por ejemplo tomando el link al propio objeto que estoy creando.

```
&ContextualContext = ProductoProveedorSelectionList.Link(&ProveedorCodigo)
```

El otro paso es cargar código para cargar la lista de ítems seleccionados y la colección, proveniente de los productos que ya se encuentran en el nexó.

```
If &HttpMethod.Method = K2BHttpMethod.Get
```

```
For Each
```

```
Where ProveedorCodigo = &ProveedorCodigo
```




```
&SelectedItem = new()  
&SelectedItem.ProductoCodigo = ProductoCodigo  
&SelectedItem.ProductoDescripcion = ProductoDescripcion  
&SelectedItems.Add(&SelectedItem)  
&ProductoCodigoList.Add(ProductoCodigo)  
EndFor  
&ContextualContext =  
ProductoProveedorSelectionList.Link(&ProveedorCodigo)  
K2BContextualSessionSet.Call(&ContextualContext,  
!"ProductoCodigoList", &ProductoCodigoList.ToXml())  
K2BContextualSessionSet.Call(&ContextualContext, !"SelectedItems",  
&SelectedItems.ToXml())  
EndIF
```

Finalmente podemos crear el componente

ComponenteMultipleSeleccion.Object =

ProductoMultipleSelection.Create(&ContextualContext)

Luego podemos crear acciones para dado el SDT dar de alta el nexos con los productos seleccionados.

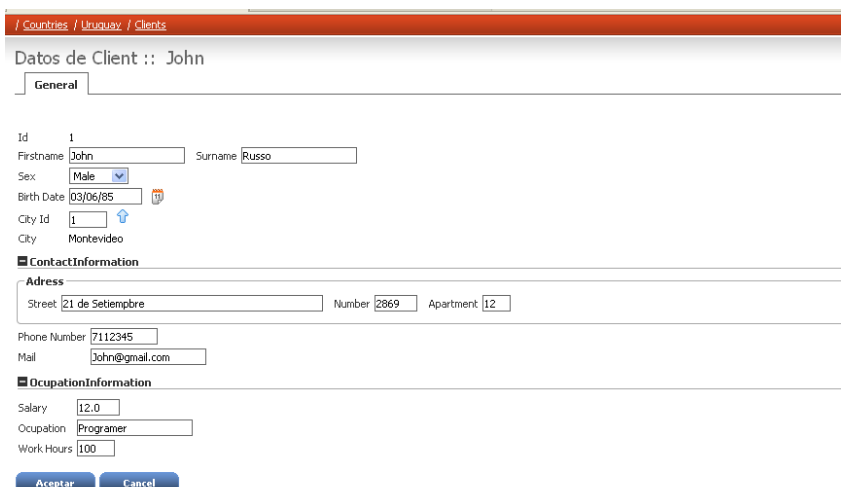
```
K2BContextualSessionGet.Call(&ExecutionContext, !"SelectedItems", &xml)  
&SelectedItems.FromXml(&Xml)
```

Se obtiene el SDT de la sesión y luego se realiza todo el proceso de dar de alta.

En un futuro se intentará que este pattern sea más automático la forma de utilizarse, integrándolo con el EntityServices, que este por ejemplo pueda detectar cuando se encuentra ante la presencia de un caso de nexos.

13 Pattern K2BTrnForm

El pattern K2BTrnForm permite mantener desde patterns el web form de la transacción. Además de mejor usabilidad, el usuario podrá generar interfaces más lindas y más fáciles de mantener. Están disponibles todas las funcionalidades que se tienen en las vistas planas, como por ejemplo agrupar atributos, colocar líneas separadoras, filas, columnas; definir atributos de resumen, la posibilidad de colapsar determinados atributos, etc.



The screenshot shows a web form titled "Datos de Client :: John" with a "General" tab selected. The form contains the following fields:

- Id:** 1
- Firstname:** John
- Surname:** Russo
- Sex:** Male (dropdown)
- Birth Date:** 03/06/85
- City Id:** 1
- City:** Montevideo
- Contact Information:**
 - Address:**
 - Street:** 21 de Setiembre
 - Number:** 2869
 - Apartment:** 12
 - Phone Number:** 7112345
 - Mail:** John@gmail.com
- Occupation Information:**
 - Salary:** 12.0
 - Occupation:** Programmer
 - Work Hours:** 100

At the bottom, there are "Aceptar" and "Cancel" buttons.

Para utilizar este patrón, se deberá seleccionar en la transacción el tab TrnForm. Ahí aparecerá la instancia por defecto. Su uso es como en el resto de los patterns.

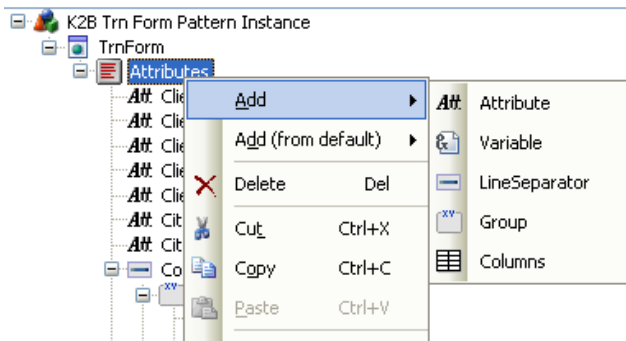


Utiliza las mismas propiedades que se utilizan en las vistas planas del pattern k2bentityservices:

- NoSkip, para posicionar atributos en la misma fila.
- Grupos, para agrupar atributos o variables.
- Columnas, para distribuir de mejor forma la pantalla.



- Lines Separator, para separar atributos.



Además para cada atributo es posible setear:

- Right Text y Left Text
- Context Help

Id 0

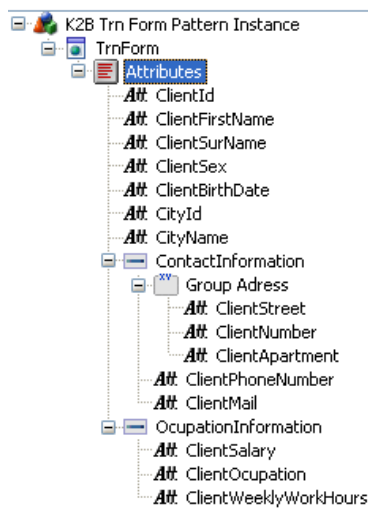
FirstName SurName

Sex

Birth Date

City

En las siguientes figuras se muestra un ejemplo de una instancia, y el web form generado.



/ Countries / Uruguay / Clients

Datos de Client :: John

General

Id 1

Firstname John Surname Russo

Sex Male

Birth Date 03/06/85

City Id 1

City Montevideo

Contact Information

Address

Street 21 de Septiembre Number 2869 Apartment 12

Phone Number 7112345

Mail john@gmail.com

Occupation Information

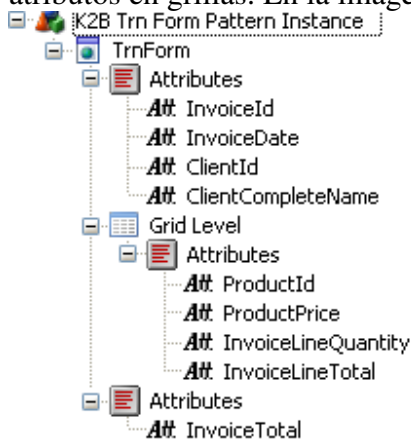
Salary 12.0

Occupation Programmer

Work Hours 100

Aceptar Cancel

Además de esto está la posibilidad de para las transacciones de dos niveles de agrupar los atributos en grillas. En la imagen de abajo se muestra una factura.



El nodo gridlevel representa una grilla asociada a un nivel.

Definition	
Name	Line
Description	Line

Es posible asociarle un name y un descriptions, que son por defecto el nombre del nivel y su descripción.

En el nodo attributes es posible setear mediante la propiedad del nodo form, DisplayAs, como visualizar esos atributos.

Form	
DisplayAs	<default>

Las posibilidades son Tabular que es la forma normal, o Summary, si se desea ver como resumen de una grilla. Para el cálculo de la instancia por defecto se toma, si el atributo viene a continuación de una grilla el valor summary, de lo contrario se toma Tabular.

General

Id 1
Date 03/11/09
Client 1 John

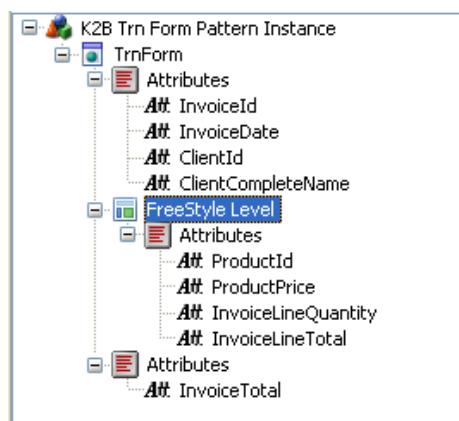
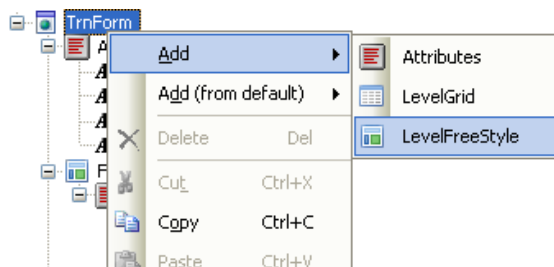
Line

Product Id	Price	Quantity	Total
1	2.50	2	5
2	22.00	2	44

Total 49

[Modificar](#) [Eliminar](#)



Es posible también en caso de que se requiera en lugar de generar una grilla, generar un Freestyle grid. Para esto se cuenta con el nodo FreeStyleGrid. Esto se hace como muestra la figura de abajo.




Ahora se muestra el mismo web form pero las líneas serán visualizadas dentro de un Freestyle grid.




General

Id 1
Date 03/11/09 
Client 1  John

Line

Product Id 1 
Price 2.50
Quantity 2
Total 5

Product Id 2 
Price 22.00
Quantity 2
Total 44

[New row]

Total 49

Aceptar

Cancelar