

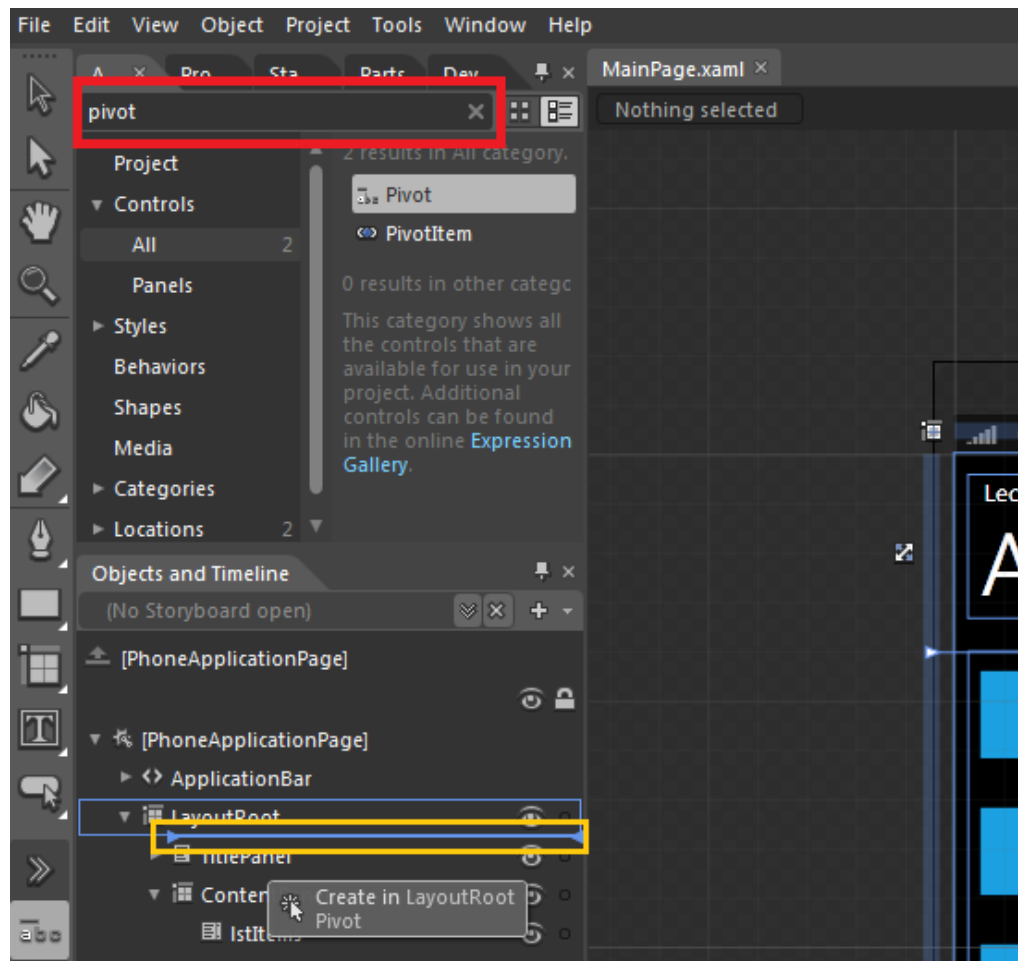
Hands on Lab: Almacenamiento local, launchers y choosers

Etapas 3

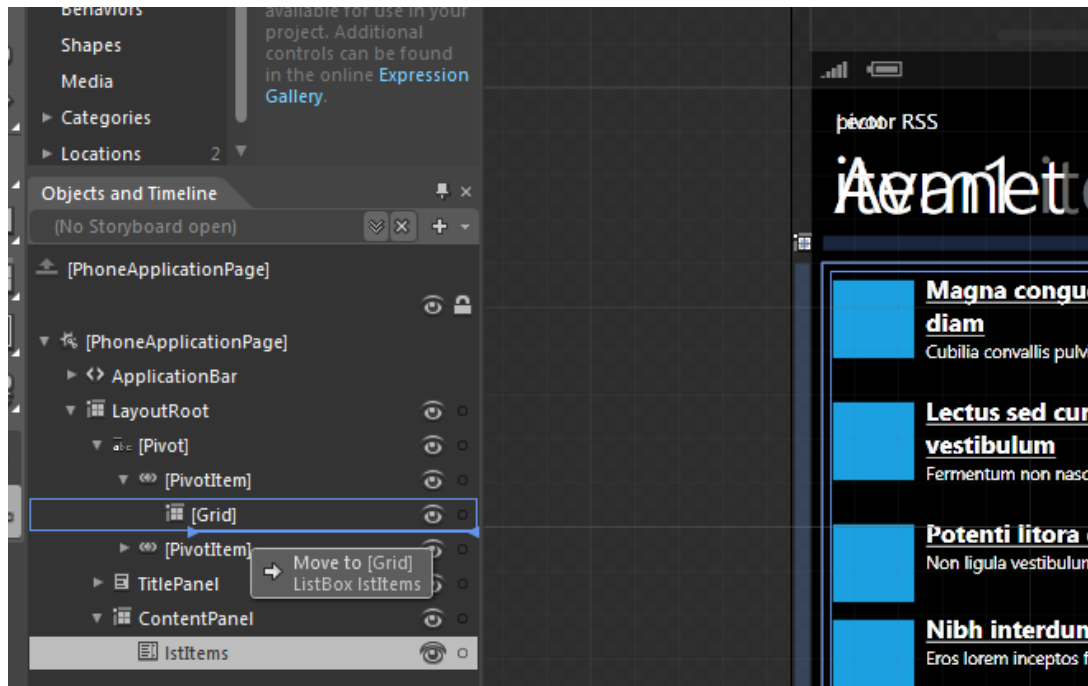
Esta etapa es para los participantes que hayan finalizado la **etapa 2**. El objetivo de este Hands On Lab es mejorar la aplicación lectora de RSS con funcionalidades complementarias ofrecidas en Windows Phone.

Enlazando datos

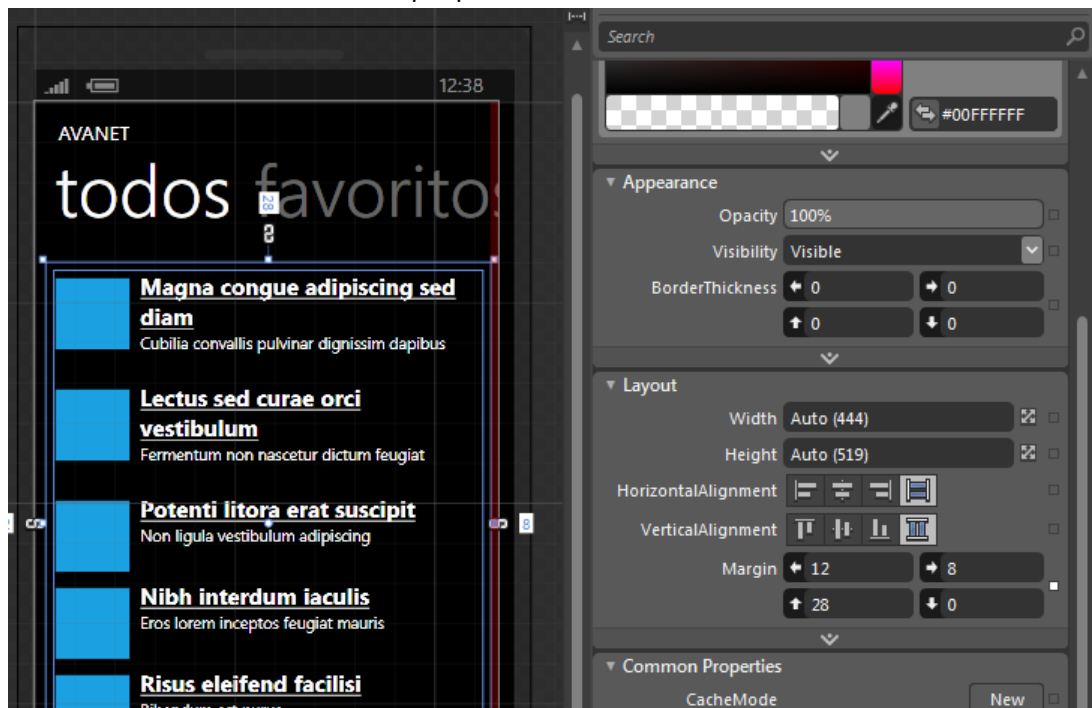
1. Retome el proyecto de la etapa 2 y ábralo en Expression Blend. Busque el control Pivot y agregue uno al LayoutRoot, verifique que el mensaje indique que se creará un nuevo Pivot.



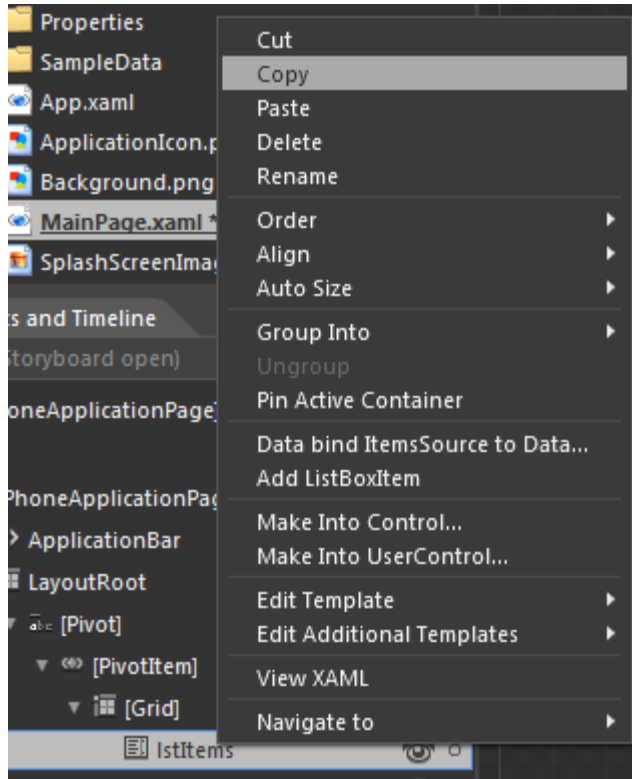
2. No se preocupe si los controles se desordenan. Expanda el contenido del **ContentPanel** y arrastre la lista de **Items** en el primer **Pivot Item**, dentro del **Grid**. Verifique que el mensaje que aparece indique que el control se moverá.



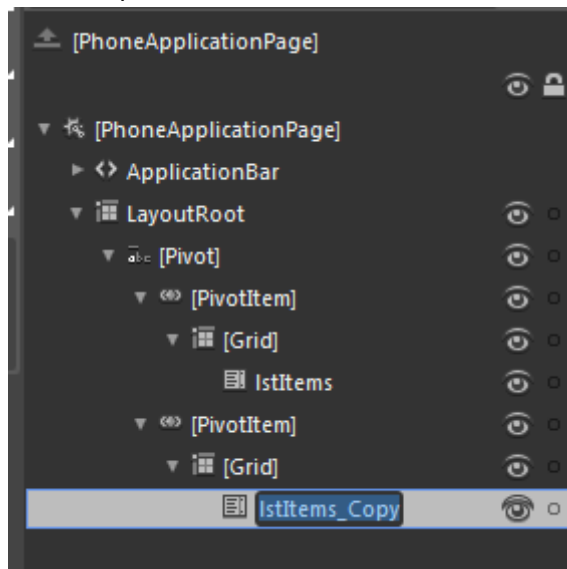
3. Elimine los elementos **TitlePanel** y **ContentPanel**. Reorganice los contenedores restantes el Pivot, el Grid y la Lista, además establezca los títulos del **Pivot** y los **Pivot Items** usando el área de propiedades.



4. En este punto la aplicación debe funcionar correctamente tal como lo hacia en la etapa 2. Copie la lista de ítems dando clic derecho sobre ella en la jerarquía.

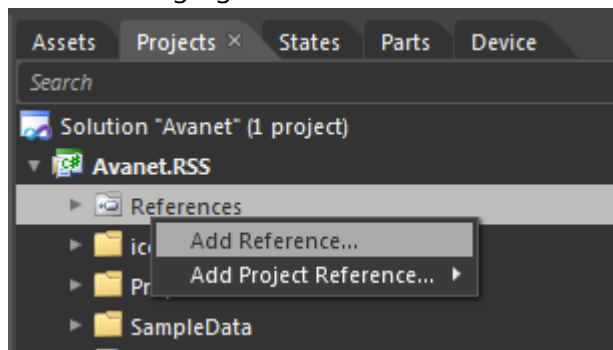


5. De la misma forma peque la lista sobre el **Grid** del segundo **Pivot Item** y cambie el nombre por **IstFavoritos**.

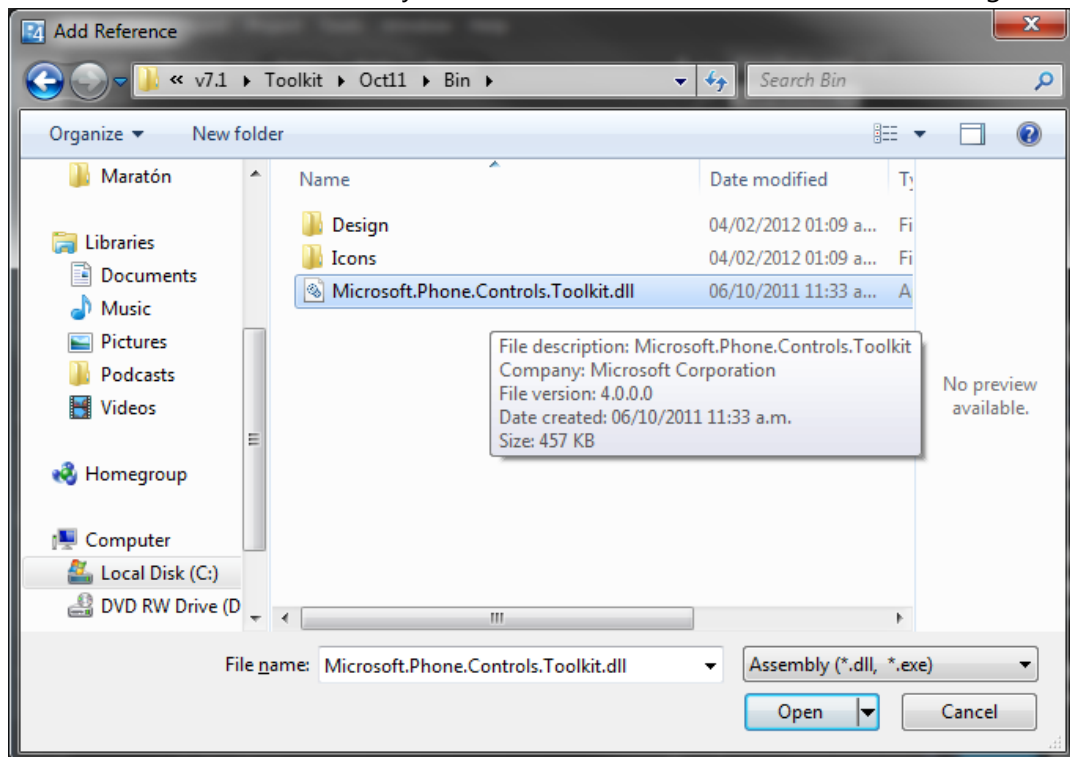


Trabajando con Menús Contextuales

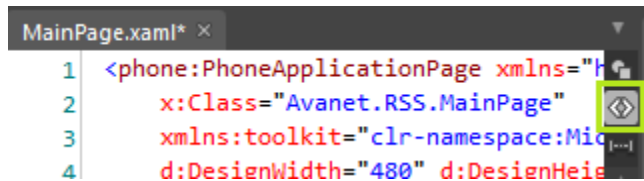
1. Ingrese a Codeplex y descargue el Silverlight Control Toolkit.
<http://silverlight.codeplex.com/>
2. Después de terminar la instalación en la pestaña de proyectos en la carpeta de referencias agregue una nueva referencia



3. Busque la ruta C:\Program Files (x86)\Microsoft SDKs\Windows Phone\v7.1\Toolkit\Oct11\Bin y añada el ensamblado del Toolkit de Silverlight



4. Cambie a la vista de marcado para editar el XAML



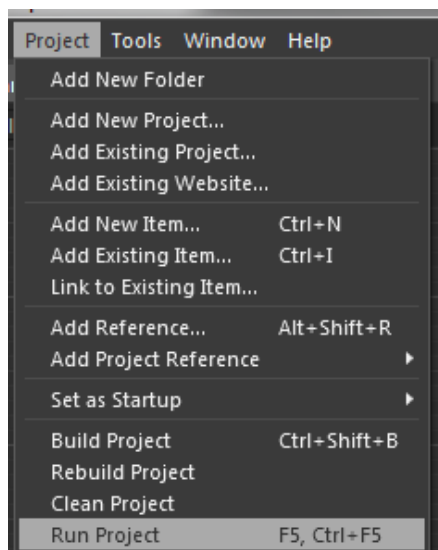
5. Agregue en la parte superior la referencia al espacio de nombres
xmlns:toolkit="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls.Toolkit"

```
1 <phone:PhoneApplicationPage xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
2   x:Class="Avanet.RSS.MainPage"
3   xmlns:toolkit="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls.Toolkit"
4   d:DesignWidth="480" d:DesignHeight="768"
5   mc:Ignorable="d"
6   SupportedOrientations="Portrait" Orientation="Portrait" />
```

6. Agregue un menú contextual a la plantilla del **SyndicationItem**

```
<DataTemplate x:Key="SyndicationItemTemplate">
  <StackPanel Orientation="Horizontal" Width="428" MinHeight="50" Margin="0,0,0,8">
    <toolkit:ContextMenuService.ContextMenu>
      <toolkit:ContextMenu>
        <toolkit:MenuItem Header="compartir enlace" />
        <toolkit:MenuItem Header="enviar por correo" />
        <toolkit:MenuItem Header="guardar como favorito" />
      </toolkit:ContextMenu>
    </toolkit:ContextMenuService.ContextMenu>
    <Rectangle Fill="{StaticResource PhoneAccentBrush}" Stroke="Black" Width="76" Margin="0,6,10,0" />
    <StackPanel Width="326">
      <HyperlinkButton Content="{Binding Title.Text}" FontWeight="Bold" NavigateUri="{Binding Link.Uri}" />
      <TextBlock Text="{Binding Summary.Text}" VerticalAlignment="Top" TextWrapping="Wrap" FontSize="12" />
    </StackPanel>
  </StackPanel>
</DataTemplate>
```

7. Si ejecuta la aplicación podrá ver las opciones aunque aun no tienen código asociado.



8. Debe poder ver algo como se muestra en la siguiente imagen



9. Debido a que las opciones que se ejecutan sobre los favoritos son diferentes debe hacer una copia de la plantilla de SyndicationItem. Llámela FavoriteItem y cambie en ella las opciones necesarias.

```
<DataTemplate x:Key="FavoriteTemplate">
    <StackPanel Orientation="Horizontal" Width="428" MinHeight="50" Margin="0,0,0,8">

        <toolkit:ContextMenuService.ContextMenu>
            <toolkit:ContextMenu>
                <toolkit:MenuItem Header="compartir enlace" />
                <toolkit:MenuItem Header="enviar por correo" />
                <toolkit:MenuItem Header="eliminar de favoritos" />
            </toolkit:ContextMenu>
        </toolkit:ContextMenuService.ContextMenu>

        <Rectangle Fill="{StaticResource PhoneAccentBrush}" Stroke="Black" Width="76" Ma
        <StackPanel Width="326">
            <HyperlinkButton Content="{Binding Title.Text}" FontWeight="Bold" NavigatUr
            <TextBlock Text="{Binding Summary.Text}" VerticalAlignment="Top" TextWrappin
        </StackPanel>
    </StackPanel>
</DataTemplate>
```

10. Además busque en lugar donde se declaran los listbox en la parte inferior y cambie el template asignado por el nuevo que creó.

```
<controls:Pivot Margin="0,0,16,16" Title="AVANET" Grid.RowSpan="2">
  <controls:PivotItem Header="todos" Margin="12,28,8,0">
    <Grid Margin="0,0,0,8">
      <ListBox x:Name="lstItems" ItemTemplate="{StaticResource SyndicationItemTemplate}" ItemsSource="{Binding Items}" />
    </Grid>
  </controls:PivotItem>
  <controls:PivotItem Header="favoritos">
    <Grid>
      <ListBox x:Name="lstFavoritos" ItemTemplate="{StaticResource FavoriteTemplate}" ItemsSource="{Binding Items}" />
    </Grid>
  </controls:PivotItem>
</controls:Pivot>
```

11. **Guarde en Expression Blend y regrese a Visual Studio.** Establezca el nombre de cada uno de los items del menu contextual usando **x:Name**. Recuerde establecer los nombres en las 2 plantillas y asignar a cada item un nombre diferente.

```
<toolkit:ContextMenuService.ContextMenu>
  <toolkit:ContextMenu>
    <toolkit:MenuItem x:Name="mnuCompartirEnlaceTodos" Header="compartir enlace" />
    <toolkit:MenuItem Header="enviar por correo" />
    <toolkit:MenuItem Header="guardar como favorito" />
  </toolkit:ContextMenu>
</toolkit:ContextMenuService.ContextMenu>
```

12. Luego usando el intellisense de Visual Studio cree los métodos asociados al evento clic para cada uno de los ítems de menú

```
<toolkit:ContextMenuService.ContextMenu>
  <toolkit:ContextMenu>
    <toolkit:MenuItem x:Name="mnuCompartirEnlaceTodos"
      Header="compartir enlace" Click="" />
  </toolkit:ContextMenu>
</toolkit:ContextMenuService.ContextMenu>
```

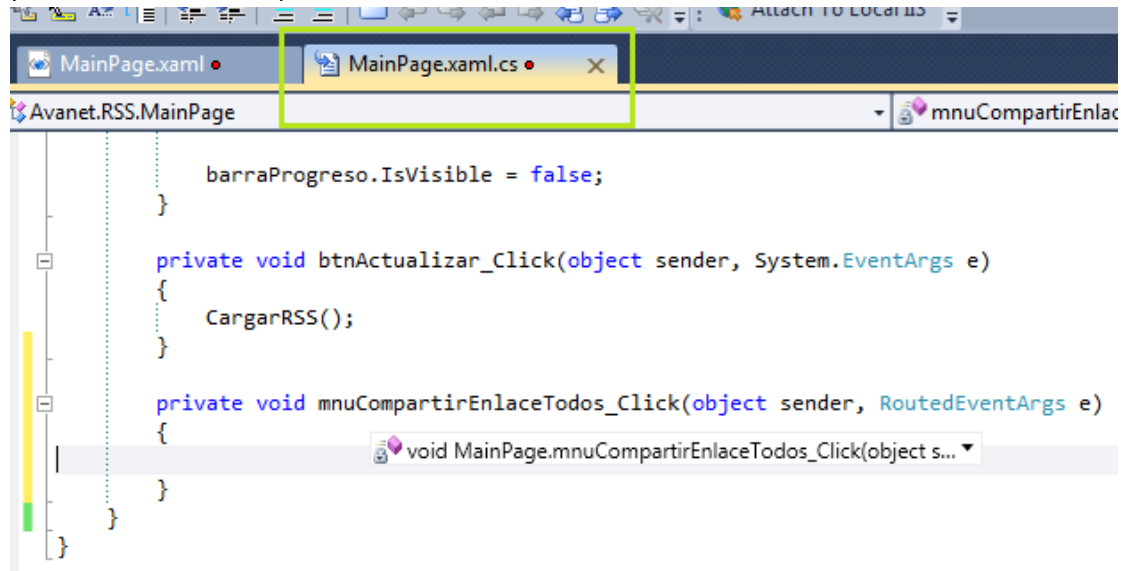
Bind event to a newly created method called 'mnuCompartirEnlaceTodos_Click'. Use 'Navigate to Event Handler' to navigate to the newly created method.

<New Event Handler>
MainPage_Loaded

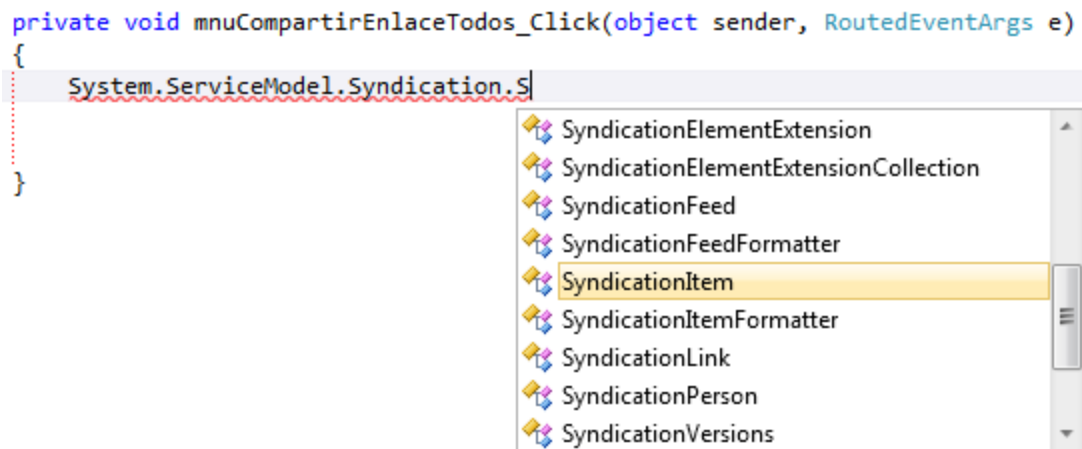
13. Observe que aparece el nombre según el nombre que asignó al control

```
<toolkit:ContextMenuService.ContextMenu>
  <toolkit:ContextMenu>
    <toolkit:MenuItem x:Name="mnuCompartirEnlaceTodos"
      Header="compartir enlace" Click="mnuCompartirEnlaceTodos_Click" />
    <toolkit:MenuItem Header="enviar por correo" />
    <toolkit:MenuItem Header="guardar como favorito" />
  </toolkit:ContextMenu>
</toolkit:ContextMenuService.ContextMenu>
```

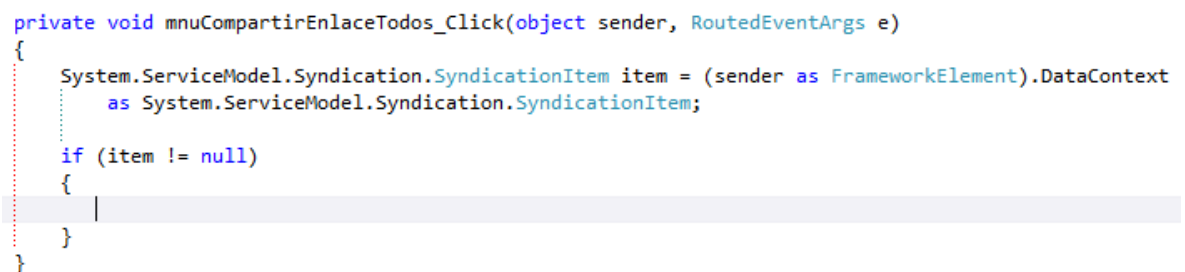
14. Además en el código de la página principal se habrá creado el método en el cual podrá construir las opciones asociadas al ítem de menú.



15. Ahora debe recuperar el ítem del RSS que se seleccionó para que apareciera el Menú Contextual. Para eso debe declararse un objeto del mismo tipo de dato que contiene la lista.



16. Recuerde verificar que efectivamente se ha recuperado el ítem para que el código no falle.

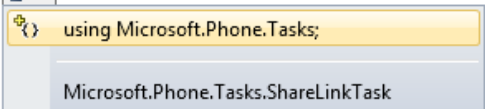


Aprendiendo sobre Launches (lanzadores) y Choosers (selectores)

1. Para implementar la opción de compartir el enlace se usará un **Launcher** o Lanzador que realiza esta acción. Podrá observar que el código no lo reconoce por lo tanto debe como en ocasiones anteriores referenciar el espacio de nombre al que pertenece.

```
private void mnuCompartirEnlaceTodos_Click(object sender, RoutedEventArgs e)
{
    System.ServiceModel.Syndication.SyndicationItem item = (sender as FrameworkElement).DataContext
        as System.ServiceModel.Syndication.SyndicationItem;

    if (item != null)
    {
        ShareLinkTask
    }
}
```



2. Luego de esto a través de unos sencillos pasos puede implementar la funcionalidad del lanzador

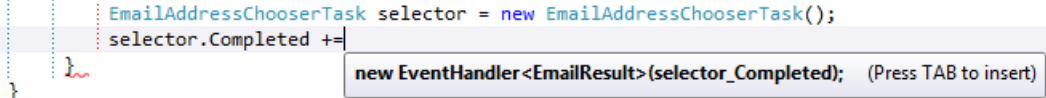
```
private void mnuCompartirEnlaceTodos_Click(object sender, RoutedEventArgs e)
{
    System.ServiceModel.Syndication.SyndicationItem item = (sender as FrameworkElement).DataContext
        as System.ServiceModel.Syndication.SyndicationItem;

    if (item != null)
    {
        //1. Instanciar el lanzador
        ShareLinkTask lanzador = new ShareLinkTask();
        //2. Configurar las propiedades
        lanzador.LinkUri = new Uri(item.Links[0].Uri.AbsoluteUri);
        lanzador.Message = string.Format(item.Summary.Text.Substring(0,50) + "...");
        lanzador.Title = "Post de @avanet " + item.Title.Text;
        //3. Mostrar el lanzador
        lanzador.Show();
    }
}
```

3. Para enviar correos necesita primero seleccionar el correo de la lista de contactos por lo tanto usaremos un **Chooser** o selector. Observe que en este debe suscribirse al evento que se da cuando el usuario termina la operación. Recuerde usar la técnica de dar doble TAB para que se cree el código asociado.

```
private void mnuEnviarCorreo_Click(object sender, RoutedEventArgs e)
{
    System.ServiceModel.Syndication.SyndicationItem item = (sender as FrameworkElement).DataContext
        as System.ServiceModel.Syndication.SyndicationItem;

    if (item != null)
    {
        EmailAddressChooserTask selector = new EmailAddressChooserTask();
        selector.Completed +=
    }
}
```



4. El código se generará vacío con un código que lanzará una excepción si no se implementa.

```
private void mnuEnviarCorreo_Click(object sender, RoutedEventArgs e)
{
    System.ServiceModel.Syndication.SyndicationItem item = (sender as FrameworkElement).DataContext
        as System.ServiceModel.Syndication.SyndicationItem;

    if (item != null)
    {
        EmailAddressChooserTask selector = new EmailAddressChooserTask();
        selector.Completed += new EventHandler<EmailResult>(selector_Completed);
    }
}

void selector_Completed(object sender, EmailResult e)
{
    throw new NotImplementedException();
}
```

5. Siguiendo una serie de sencillos pasos podrá ejecutar el selector. Observe que además declaramos una variable itemSeleccionado a nivel de la clase, para que este ítem esté disponible cuando la selección del contacto se complete.

```
public SyndicationItem itemSeleccionado { get; set; }

private void mnuEnviarCorreo_Click(object sender, RoutedEventArgs e)
{
    System.ServiceModel.Syndication.SyndicationItem item = (sender as FrameworkElement).DataContext
        as System.ServiceModel.Syndication.SyndicationItem;

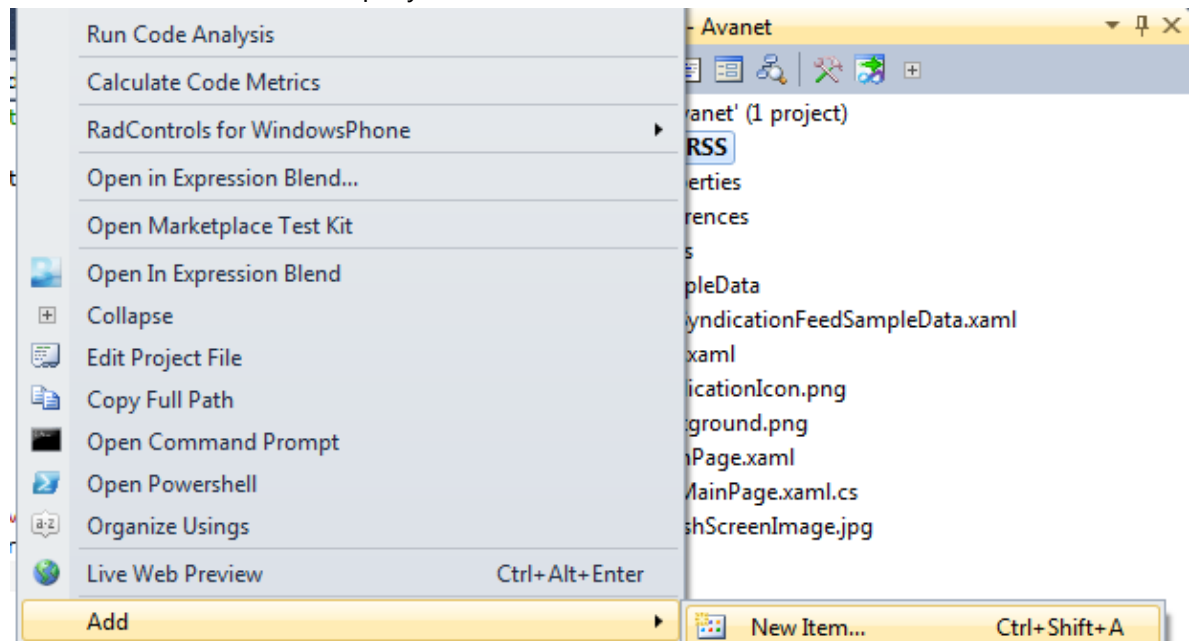
    if (item != null)
    {
        //1. Instanciamos el selector
        EmailAddressChooserTask selector = new EmailAddressChooserTask();
        //2. Completamos las propiedades si las tiene (En este caso no las tiene)
        //3. Nos suscribimos al evento Completed
        selector.Completed += new EventHandler<EmailResult>(selector_Completed);
        this.itemSeleccionado = item;
        //4. Mostramos el selector
        selector.Show();
    }
}
```

6. No olvide implementar el método que se ejecuta al momento de finalizar la selección comprobando el resultado de la operación.

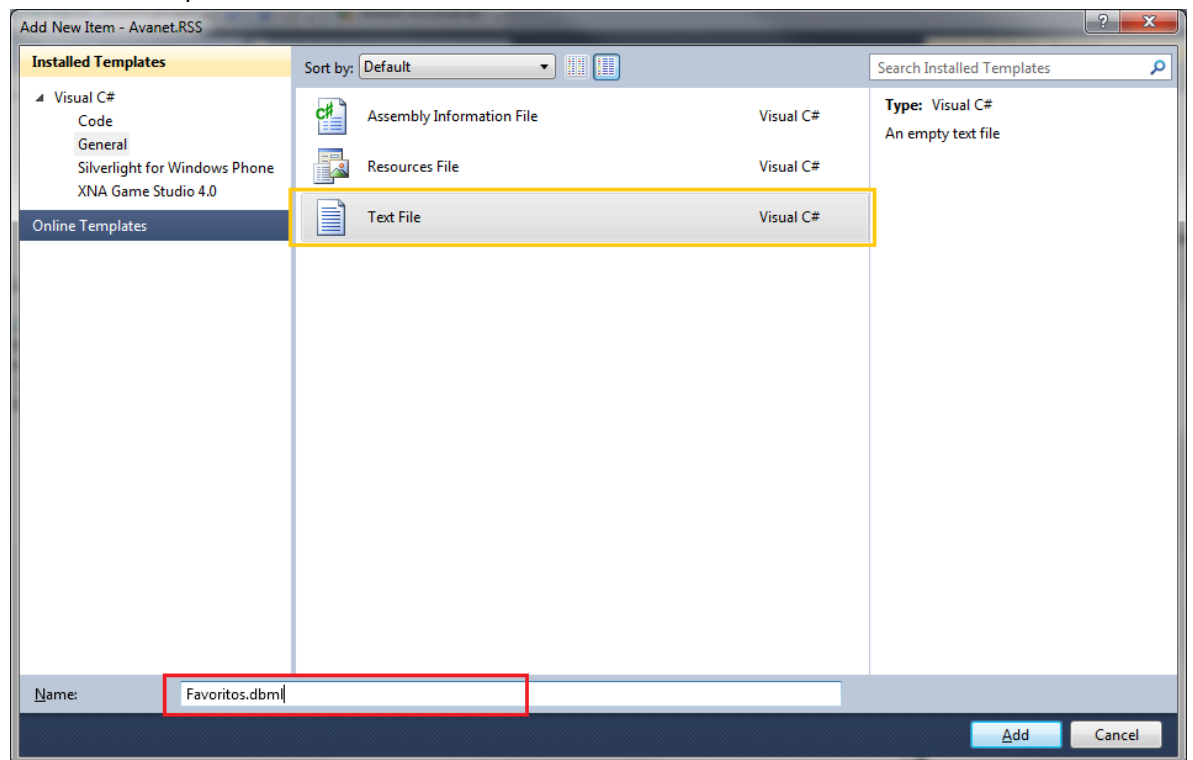
```
void selector_Completed(object sender, EmailResult e)
{
    if (e.TaskResult == TaskResult.OK)
    {
        EmailComposeTask lanzador = new EmailComposeTask();
        lanzador.To = e.Email;
        lanzador.Body = string.Format("Te comparto el post \"{0}\" que leí " +
            "a través del RSS de Avaneet\r\n Enlace: {1}",
            itemSeleccionado.Title.Text, itemSeleccionado.Links[0].Uri.AbsoluteUri);
        lanzador.Show();
    }
}
```

Trabajando con Enlaces de Datos y Bases de Datos

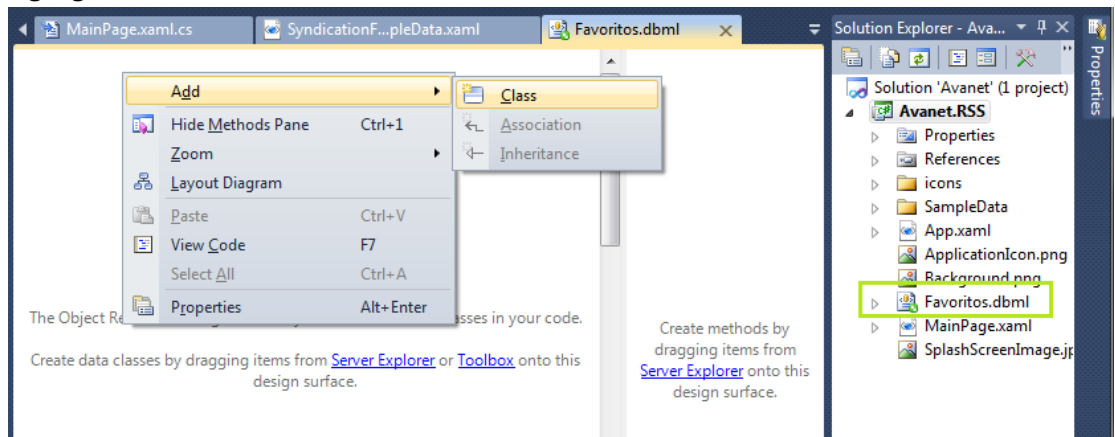
1. Añada un nuevo archivo al proyecto



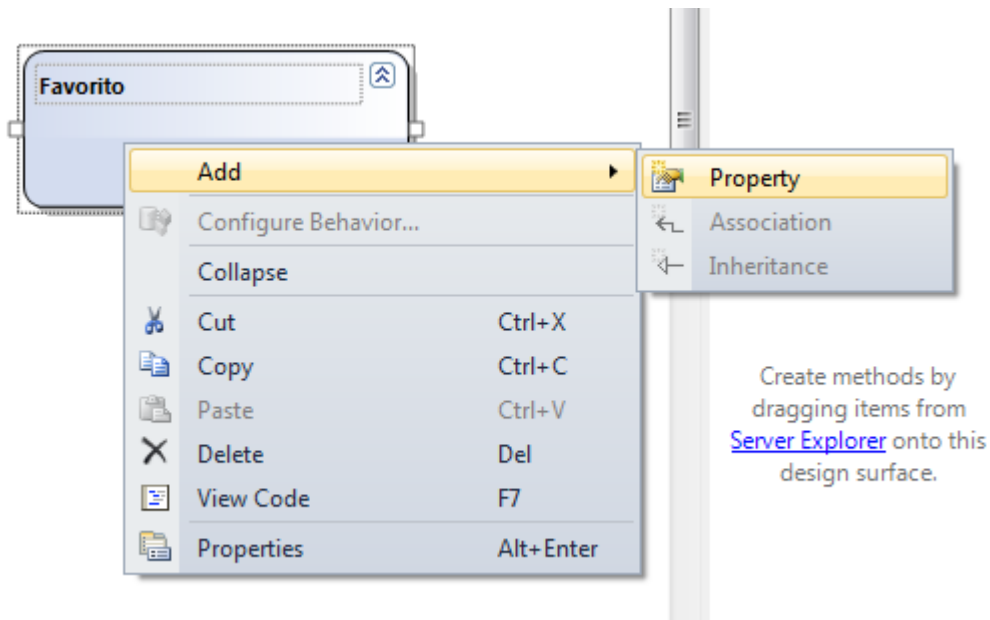
2. El archivo debe ser tipo TextFile sin embargo en esta ocasión debe cambiar la extensión txt por dbml. Establezca el nombre como **Favoritos.dbml**



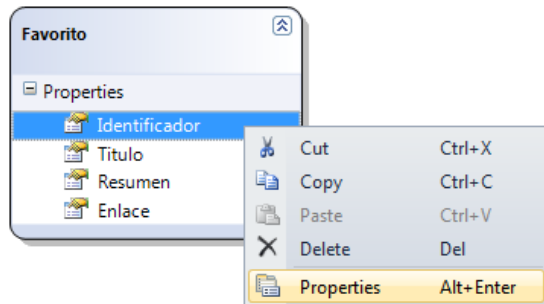
3. Observe que el archivo se creó correctamente, debe aparecer un diseñador. Agregue una clase nueva.



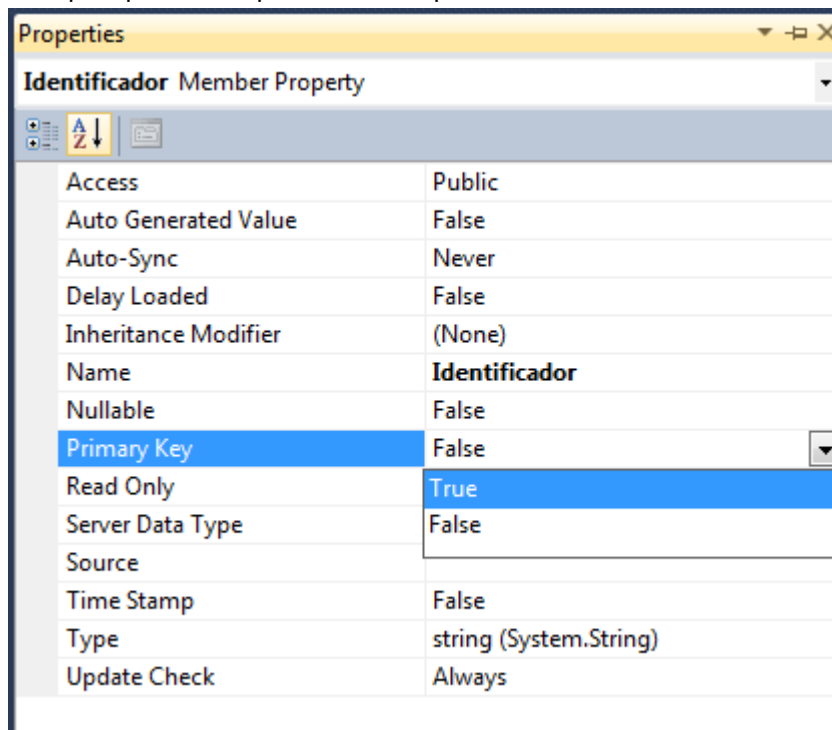
4. Estas clases se mapean directamente contra tablas en la base de datos. Cambie el nombre por defecto a Favorito y añada los campos que va a tener la tabla: Identificador, Titulo, Resumen, Enlace.



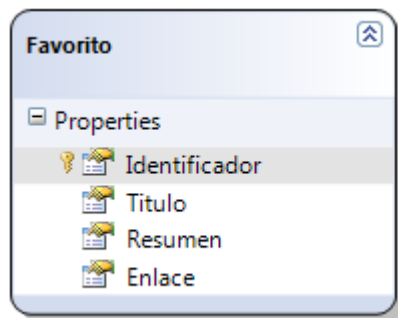
5. Seleccione el campo Identificador que será la clave primaria y vaya a las propiedades.



6. Indique que el campo será clave primaria.

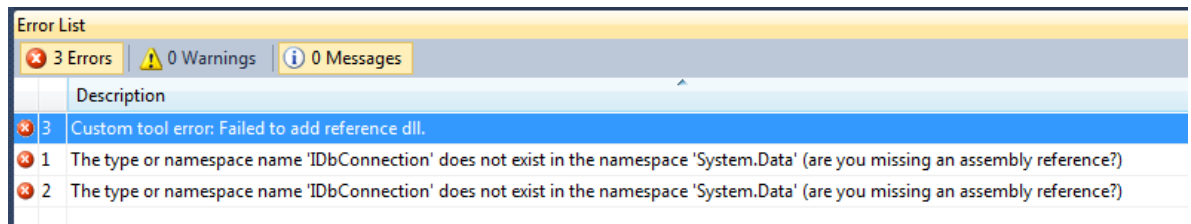


7. Verifique que el icono en forma de llave aparece al lado izquierdo del campo.



8. Compile la solución.

9. Podrá observar que tiene errores, ya que el soporte para partes del código que genera LINQ no es completo en Windows Phone. De doble clic sobre los errores para ver donde se generan.



10. Borre el código asociado a los errores y vuelva a compilar asegurándose que la compilación es exitosa.

```
public FavoritosDataContext(string connection) :  
    base(connection, mappingSource)  
{  
    OnCreated();  
}
```

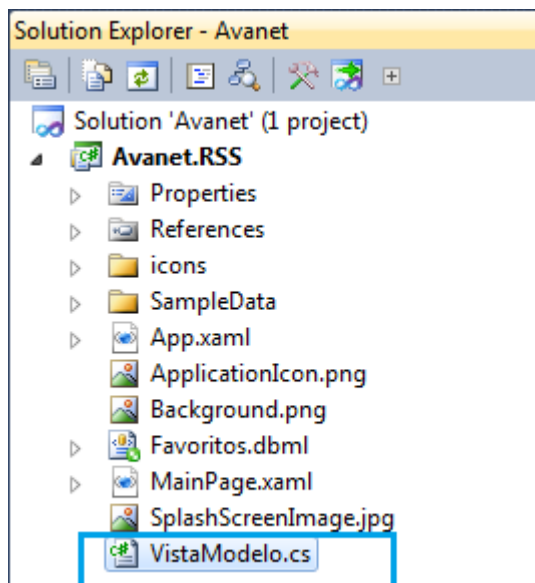
```
public FavoritosDataContext(System.Data.IDbConnection connection) :  
    base(connection, mappingSource)  
{  
    OnCreated();  
}
```

```
public FavoritosDataContext(string connection, System.Data.Linq.Mapping.MappingSource mappingSource) :  
    base(connection, mappingSource)  
{  
    OnCreated();  
}
```

```
public FavoritosDataContext(System.Data.IDbConnection connection, System.Data.Linq.Mapping.MappingSource mappingSource) :  
    base(connection, mappingSource)  
{  
    OnCreated();  
}
```

```
public System.Data.Linq.Table<Favorito> Favoritos  
{  
}
```

11. Ahora declare la clase en donde se cargarán los datos que son consultados desde la base de datos. Añada una nueva clase llamada **VistaModelo**



- 12.

13. Esta clase debe heredar desde **INotifyPropertyChanged** para poder enlazarse a la lista de favoritos e implementar la infraestructura necesaria para notificar los cambios a la interfaz. Guíese de la siguiente implementación para su clase.

```
namespace Avamet.RSS
{
    public class VistaModelo : INotifyPropertyChanged
    {
        private ObservableCollection<Favorito> favoritos = new ObservableCollection<Favorito>();

        public ObservableCollection<Favorito> Favoritos
        {
            get { return favoritos; }
            set
            {
                favoritos = value;
                OnPropertyChanged("Favoritos");
            }
        }

        public event PropertyChangedEventHandler PropertyChanged;

        protected virtual void OnPropertyChanged(string propName)
        {
            PropertyChangedEventHandler eventHandler = this.PropertyChanged;
            if (eventHandler != null)
            {
                eventHandler(this, new PropertyChangedEventArgs(propName));
            }
        }
    }
}
```

14. Diríjase a **App.xaml.cs** y declare dos propiedades una para la cadena de conexión a la base de datos y otra para mantener una instancia de la clase que creamos en el paso anterior para cargar los datos. Observe que son estáticas.

```
namespace Avamet.RSS
{
    public partial class App : Application
    {
        static string cadenaConexion = "isostore:/favoritos.sdf";

        public static string CadenaConexion
        {
            get { return App.cadenaConexion; }
        }

        private static VistaModelo vistaModelo = null;

        public static VistaModelo VistaModelo
        {
            get
            {
                if (vistaModelo == null)
                {
                    vistaModelo = new VistaModelo();
                }
                return vistaModelo;
            }
        }
    }
}
```

15. En el constructor del **MainPage**, asigne al contexto de la página la propiedad que se creó a nivel del contexto de la aplicación.

```
public partial class MainPage : PhoneApplicationPage
{
    ProgressIndicator barraProgreso;

    // Constructor
    public MainPage()
    {
        InitializeComponent();

        this.DataContext = App.VistaModelo;

        this.Loaded += new RoutedEventHandler(MainPage_Loaded);
    }
}
```

16. Además cuando se cargue la página ejecute dos métodos, uno que se encargue de revisar si la base de datos existe y en caso de no existir la cree a partir del archivo dbml y otro que cargue los ítems favoritos de la base de datos en caso de existir.

```
void MainPage_Loaded(object sender, RoutedEventArgs e)
{
    CargarRSS();
    VerificarBaseDatos();
    CargarFavoritos();
}

private void VerificarBaseDatos()
{
    using (FavoritosDataContext contexto = new FavoritosDataContext(App.CadenaConexion))
    {
        if (!contexto.DatabaseExists())
        {
            contexto.CreateDatabase();
        }
    }
}

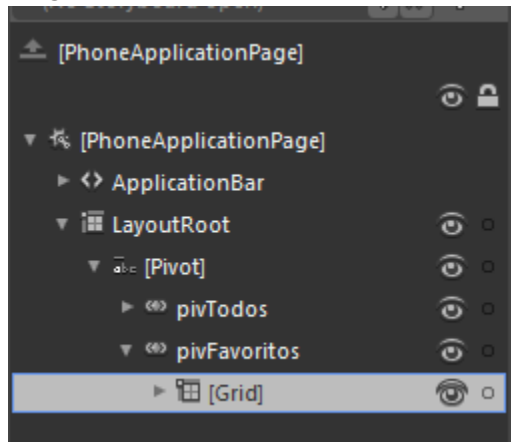
private void CargarFavoritos()
{
    using (FavoritosDataContext contexto = new FavoritosDataContext(App.CadenaConexion))
    {
        ObservableCollection<Favorito> resultado = new ObservableCollection<Favorito>();

        (from f in contexto.Favoritos select f).ToList().ForEach(x =>
            resultado.Add(x));

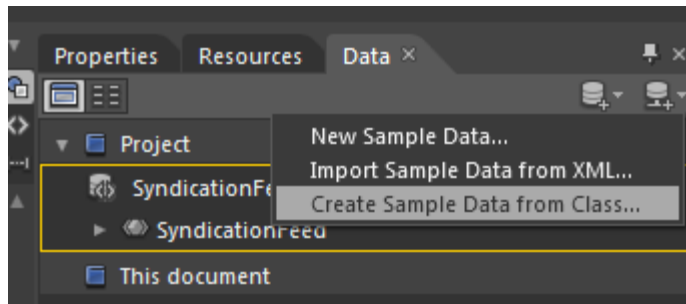
        App.VistaModelo.Favoritos = resultado;
    }
}
```

17. Compile y corrija errores si los tiene. Guarde los cambios y regrese a Expression Blend.

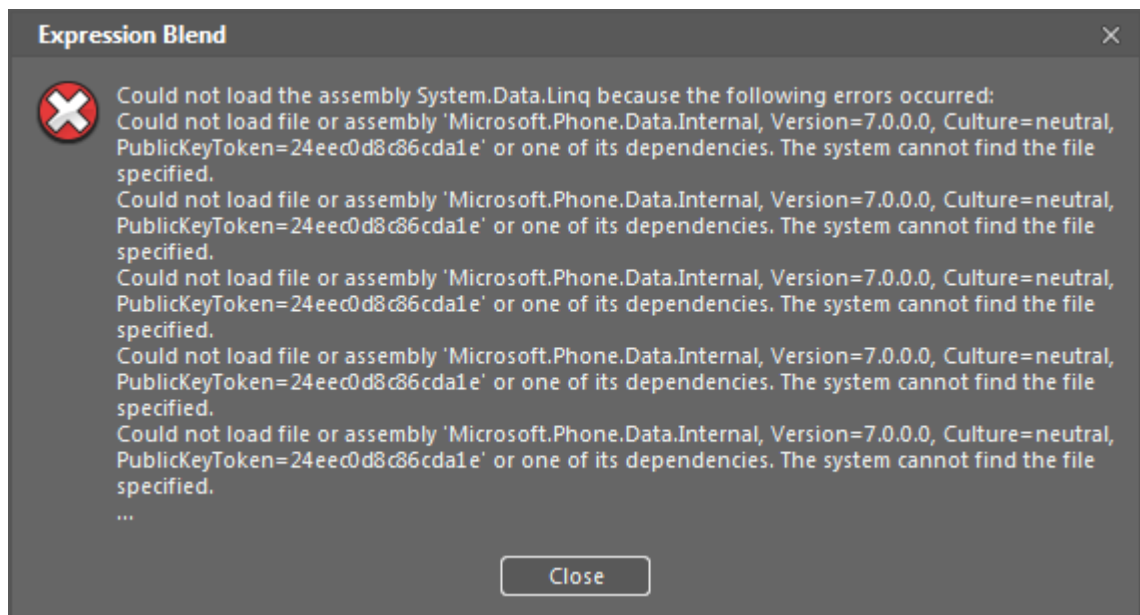
18. Asigne nombres a los 2 **Pivot Item**



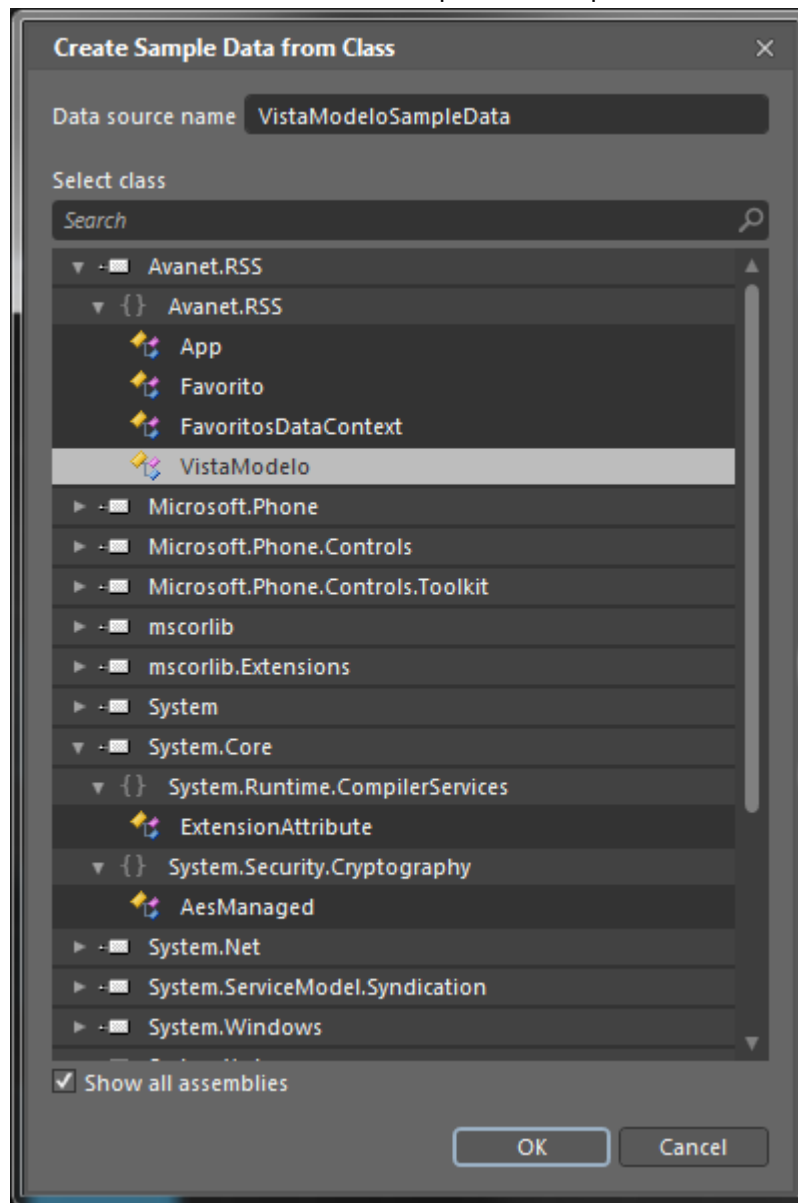
19. Cree un nuevo SampleData a partir de una clase



20. Ignore el error que aparece se debe a la parte que no se soporta de LINQ en Windows Phone



21. Seleccione la clase VistaModelo que creó en pasos anteriores.



22. Observe que aparece la estructura de la clase



23. **IMPORTANTE:** No haga enlace con el nuevo SampleData con el listbox arrastrándolo por que perderá el estilo de los ítems.

24. Verifique con atención los cambios que se necesitan. Remueva la asignación de **DataContext** que tiene el **LayoutRoot** y asigne a cada uno de los **PivotItem** el **DataContext** que corresponde

```
<!--LayoutRoot is the root grid where all page content is placed-->
<Grid x:Name="LayoutRoot" Background="Transparent">
    <Grid.RowDefinitions>
        <RowDefinition Height="Auto"/>
        <RowDefinition Height="*/>
    </Grid.RowDefinitions>

    <controls:Pivot Margin="0,0,16,16" Title="AVANET" Grid.RowSpan="2">
        <controls:PivotItem x:Name="pivTodos" Header="todos" Margin="12,28,8,0"
            d:DataContext="{d:DesignData /SampleData/SyndicationFeedSampleData.xaml}">
            <Grid Margin="0,0,0,8">
                <ListBox x:Name="lstItems" ItemTemplate="{StaticResource SyndicationItemTemplate}" ItemsSource="{Binding Items}" Margin="8"/>
            </Grid>
        </controls:PivotItem>
        <controls:PivotItem x:Name="pivFavoritos" Header="favoritos" d:DataContext="{d:DesignData /SampleData/VistaModeloSampleData.xaml}">
            <Grid>
                <ListBox x:Name="lstFavoritos" ItemTemplate="{StaticResource FavoriteTemplate}" ItemsSource="{Binding Favoritos}" Margin="8,8,4,16"/>
            </Grid>
        </controls:PivotItem>
    </controls:Pivot>

    <!--TitlePanel contains the name of the application and page title-->

    <!--ContentPanel - place additional content here-->
</Grid>
```

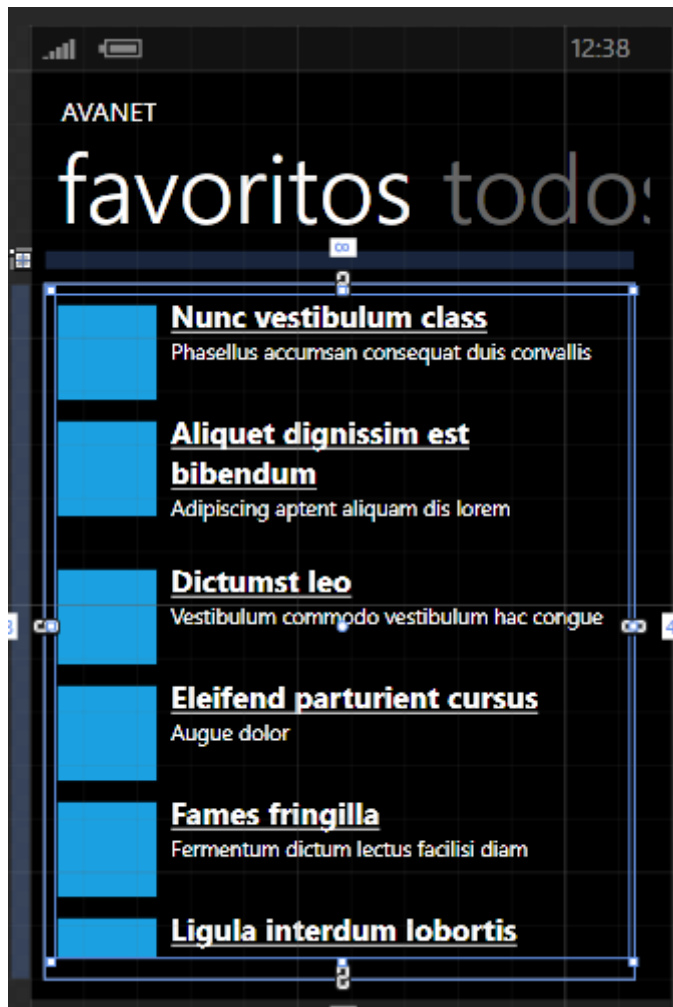
25. Diríjase a la plantilla de los favoritos que creó en pasos anteriores y edite con cuidado cada uno de los enlaces a los campos.

```
<DataTemplate x:Key="FavoriteTemplate">
    <StackPanel Orientation="Horizontal" Width="428" MinHeight="50" Margin="0,0,0,8">

        <toolkit:ContextMenuService.ContextMenu>
            <toolkit:ContextMenu>
                <toolkit:MenuItem x:Name="mnuCompartirEnlaceFavoritos"
                    Header="compartir enlace" Click="mnuCompartirEnlace_Click" />
                <toolkit:MenuItem x:Name="mnuEnviarCorreoFavoritos"
                    Header="enviar por correo" Click="mnuEnviarCorreo_Click" />
                <toolkit:MenuItem x:Name="mnuEliminarFavorito"
                    Header="eliminar favorito" Click="mnuEliminarFavorito_Click" />
                <toolkit:MenuItem x:Name="mnuAnclarInicioFavorito"
                    Header="anclar al inicio" Click="mnuAnclarInicioFavorito_Click"/>
            </toolkit:ContextMenu>
        </toolkit:ContextMenuService.ContextMenu>

        <Rectangle Fill="{StaticResource PhoneAccentBrush}" Stroke="Black" Width="76" Margin="0,6,10,0" Height="73" VerticalAlignment="Top"/>
        <StackPanel Width="326">
            <HyperlinkButton Content="{Binding Titulo}" FontWeight="Bold" NavigateUri="{Binding Enlace, Mode=OneWay}"
                Style="{StaticResource HyperlinkButtonStyle1}" TargetName="_blank"/>
            <TextBlock Text="{Binding Resumen}" VerticalAlignment="Top" TextWrapping="Wrap" FontSize="16" Height="42"/>
        </StackPanel>
    </StackPanel>
</DataTemplate>
```

26. Si realizó todos los pasos correctamente la **Lista de Favoritos** debe funcionar y tener ítems diferentes en diseño a la **Lista de Todos**.



27. Diríjase al método que carga el RSS y en la asignación del **DataContext** cámbielo por asignar el **DataContext** del **PivotItem** que corresponde.

```
private void CargarRSS()
{
    this.barraProgreso.IsVisible = true;

    WebClient cliente = new WebClient();
    cliente.DownloadStringAsync(new Uri("http://avanet.org/blog924rss.aspx"
        + "?Trick=" + DateTime.Now.ToShortDateString() + DateTime.Now.ToShortTimeString()));

    cliente.DownloadStringCompleted += new DownloadStringCompletedEventHandler(cliente_DownloadStringCompleted);
}

void cliente_DownloadStringCompleted(object sender, DownloadStringCompletedEventArgs e)
{
    XmlReader reader = XmlReader.Create(new StringReader(e.Result));
    SyndicationFeed feed = SyndicationFeed.Load(reader);
    this.pivTodos.DataContext = feed;

    barraProgreso.IsVisible = false;
}
```

28. Implemente la funcionalidad de **Guardar Favorito en la lista de Todos**

```
private void mnuGuardarFavorito_Click(object sender, RoutedEventArgs e)
{
    System.ServiceModel.Syndication.SyndicationItem item = (sender as FrameworkElement).DataContext
        as System.ServiceModel.Syndication.SyndicationItem;

    using (FavoritosDataContext contexto = new FavoritosDataContext(App.CadenaConexion))
    {
        Favorito favoritoConsultado = (from f in contexto.Favoritos
                                        where f.Identificador.Equals(item.Id)
                                        select f).SingleOrDefault();

        if (favoritoConsultado == null)
        {
            Favorito nuevoFavorito = new Favorito()
            {
                Identificador = item.Id,
                Enlace = item.Links[0].Uri.AbsoluteUri,
                Resumen = item.Summary.Text.Substring(50),
                Titulo = item.Title.Text
            };

            contexto.Favoritos.InsertOnSubmit(nuevoFavorito);
            contexto.SubmitChanges();
        }
        else
        {
            MessageBox.Show("El favorito ya existe");
        }
    }

    CargarFavoritos();
}
```

29. Implemente la funcionalidad de **Eliminar Favorito en la lista de Favoritos**

```
private void mnuEliminarFavorito_Click(object sender, RoutedEventArgs e)
{
    Favorito item = (sender as FrameworkElement).DataContext as Favorito;

    using (FavoritosDataContext contexto = new FavoritosDataContext(App.CadenaConexion))
    {
        Favorito favoritoConsultado = (from f in contexto.Favoritos
                                        where f.Identificador.Equals(item.Identificador)
                                        select f).SingleOrDefault();

        if (favoritoConsultado != null)
        {
            contexto.Favoritos.DeleteOnSubmit(favoritoConsultado);
            contexto.SubmitChanges();
        }
    }

    CargarFavoritos();
}
```

Añadiendo Live Tiles o Mosaicos Vivos

1. Su último reto es añadir una opción adicional al Menú Contextual de la lista de Favoritos, que haga que se cree un Live Tile. Para ello en el código asociado a la nueva opción debe crear el nuevo Tile usando el siguiente código:

```
private void mnuAnclarInicioFavorito_Click(object sender, RoutedEventArgs e)
{
    Favorito item = (sender as FrameworkElement).DataContext
        as Favorito;

    StandardTileData nuevoMosaico = new StandardTileData
    {
        BackgroundImage = new Uri("Background.png", UriKind.Relative),
        Title = "Favorito Avonet",
        BackContent = item.Titulo,
    };

    ShellTile.Create(new Uri("/MainPage.xaml?enlace=" + item.Enlace, UriKind.Relative), nuevoMosaico);
}
```

2. Además debe **sobrecribir** el método **OnNavigatedTo** creándolo en el **MainPage**, de tal forma que este consulte si se está accediendo desde un Tile y ejecute un lanzador que abra el enlace que se ancló como favorito.

```
protected override void OnNavigatedTo(System.Windows.Navigation.NavigationEventArgs e)
{
    base.OnNavigatedTo(e);

    if (NavigationContext.QueryString.ContainsKey("enlace"))
    {
        WebBrowserTask lanzador = new WebBrowserTask();
        lanzador.Uri = new Uri(NavigationContext.QueryString["enlace"].ToString(), UriKind.Absolute);
        lanzador.Show();
    }
}
```

IMPORTANTE: Todas las funcionalidades explicadas aquí son generales para todos los participantes, recuerde personalizar su aplicación y hacer control de errores de su código. Si su aplicación se diferencia de las demás estará más cerca de ganar el reto.