Form Layout Guide

Version 1.01

The purpose of this document is to present to developers the recommended way to lay out controls within a form. Control descriptions are provided to help beginners understand the nature of the recommended controls and to give a better understanding on how to use them. Also included are actual layout guidelines that are the principles for good form usability and specific caveats that lend to having a consistent interface.

Author: Roberto Torres
roberto.torres@dridefault.com

Editor: Josef De Beer
josef.debeer@dridefault.com

Table of Contents

Controls

Base Content Controls

These controls have been defined by Silverlight and are the basis of all controls.

StackPanel

The `StackPanel` is the preferred base content control due to it being lightweight and easy to implement. This control is best used for single column display of fields or groups that are not expected to resize dynamically. For more info:
http://msdn.microsoft.com/en-us/library/system.windows.controls.stackpanel(v=vs.95).aspx

Grid

The `Grid` is the base content control that is used when elements need to be aligned or spaced evenly. For more info:
http://msdn.microsoft.com/en-us/library/system.windows.controls.grid(v=vs.95).aspx

Canvas

Do not use this control.

TDS Container Controls

These controls have been defined by DRI control developers to increase form implementation facility and consistent look & feel.

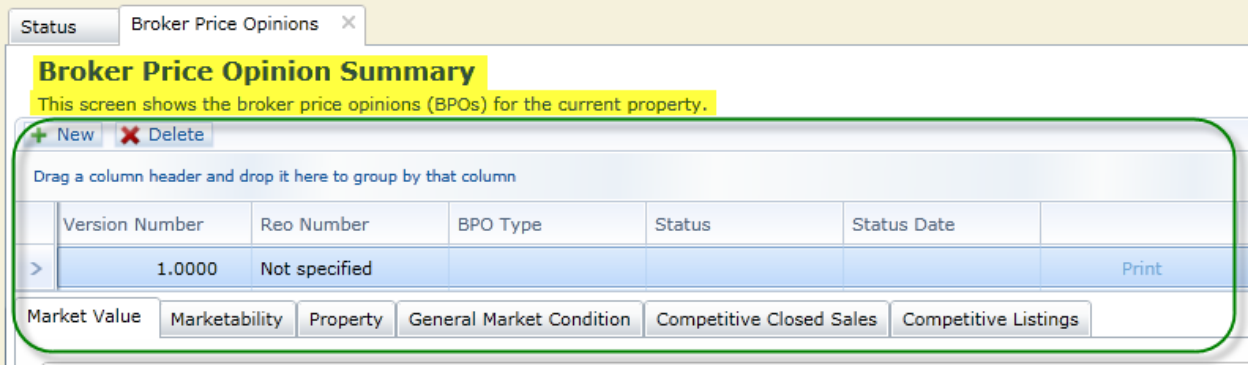
Form

The `Form` control should be the single encompassing content control for every screen. This control will contain a combination of other content controls that are listed below. The `FormTitle` and `FormDescription` properties should always be defined. `Forms` should primarily consist of `FormParts`. If there are many `DataFields` on a `Form`, consider making it a `FormPart`.

Sample (BrokerPriceOpinionList.xaml):

```
<controls:Form x:Class="Dri.Tds.UI.Forms.BrokerPriceOpinionList"
    xmlns="http://schemas.microsoft.com/client/2007"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:controls="clr-namespace:Dri.Tds.UI.Controls;assembly=Dri.Tds.UI.Controls"
    xmlns:formparts="clr-namespace:Dri.Tds.UI.FormParts;assembly=Dri.Tds.UI.FormParts"
    FormTitle="Broker Price Opinion Summary"
    SaveBarVisibility="Visible"
    FormDescription="This screen shows the broker price opinions (BPOs) for the current property."
    ContextObject="{Binding ContextObject, Mode=OneWay, ElementName=fpBrokerPriceOpinionList}">

    <formparts:BrokerPriceOpinionList x:Name="fpBrokerPriceOpinionList"/>
</controls:Form>
```



FormPart

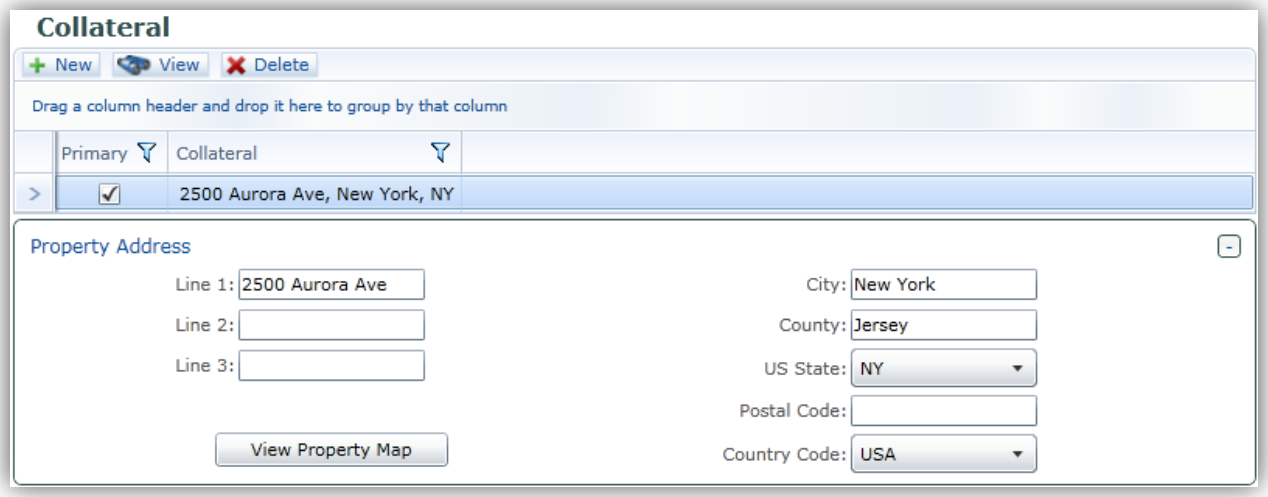
The `FormPart` control is a mini-version of the Form control. The `FormPart` is used for reusability. It should contain a subset of data that might be used elsewhere in the application.

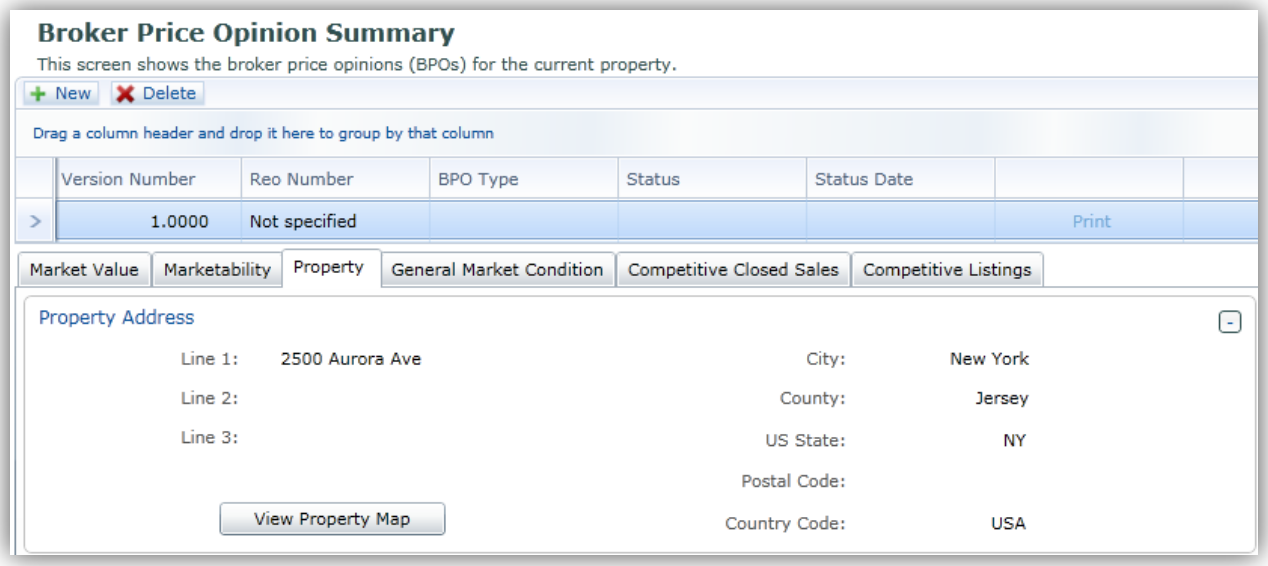
For example, `PropertyAddress.xaml` consists of Address Lines 1-3, City, State, etc. and is used in every screen that needs to display a property's address.

Sample (PropertyAddress.xaml):

```
<controls:FormPart x:Class="Dri.Tds.UI.FormParts.PropertyAddress"
    xmlns="http://schemas.microsoft.com/client/2007"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:controls="clr-namespace:Dri.Tds.UI.Controls;assembly=Dri.Tds.UI.Controls">
```

Result: PropertyAddress FormPart can is used in multiple forms.





DataFieldPanel

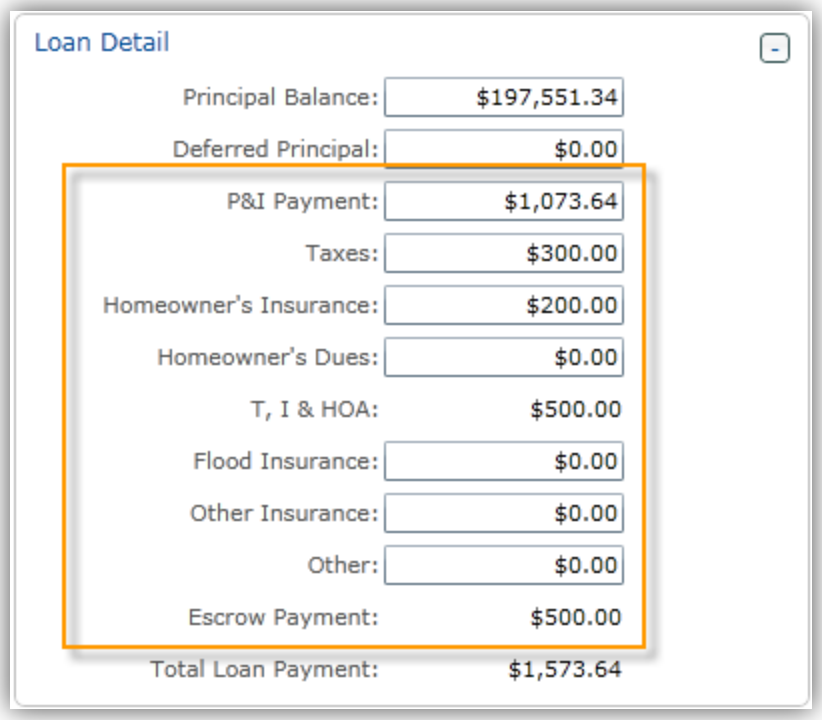
The `DataFieldPanel` can be used in place of the `StackPanel` control. This control is provided as a container for which you wish to apply the same properties to a set of `DataFields`:

- `DataObjectName`
- `FieldLabelAlignment`
- `FieldLabelPosition`
- `LabelWidth`

Sample (from PaymentDetail.xaml):

```
<controls:DataFieldPanel LabelWidth="175" FieldLabelAlignment="Right" Width="300">
    <controls:DataField DataFieldName="MonthlyPaymentPrincipalInterest" LabelText="P&amp;amp;I Payment" />
    <controls:DataField DataFieldName="MonthlyPaymentTax"  />
    …
```

Result: All labels are of equal length (invisible b/c label background is transparent) and right-aligned.



MasterDetailsGrid

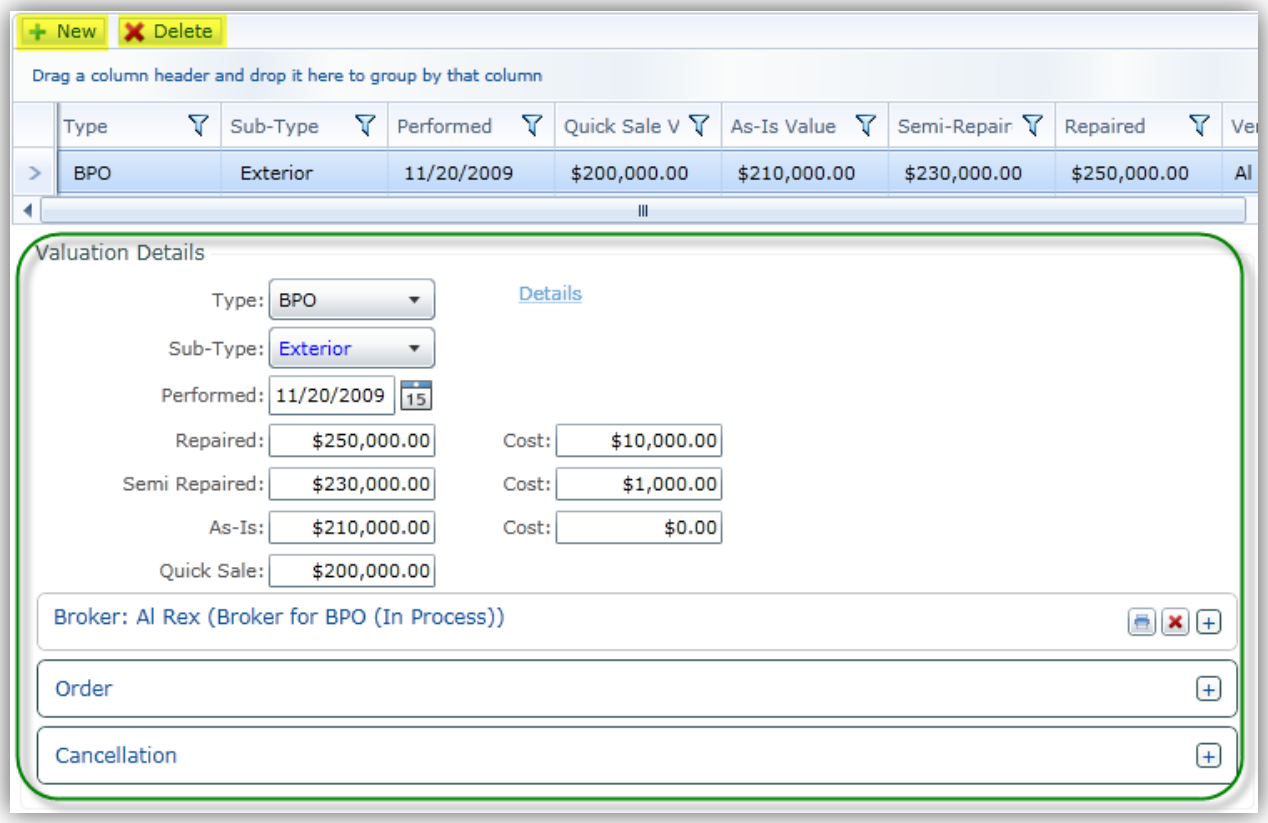
This control is essentially a `DataGrid` with expanded features. For most cases, this should be the control used when displaying multiple records. The versatility of this control can be attained by setting a couple of properties including, but not limited to:

- showing a detail panel below the grid associated to the selected row
- double-clicking a record to navigate the user to a different tab or even show a pop-up
- handling adding/deleting records

Sample (ValuationList.xaml):

```
<controls:MasterDetailsGrid x:Name="mdgValuationsList" DataObjectName="Valuations"
        BusinessObjectName="Valuation" Mode="Detail" DetailsControlName="ValuationDetails"
        IsReadOnly="True" CanAdd="True" CanDelete="True" HorizontalAlignment="Left" />
```

Result:



BorderGroupExpander

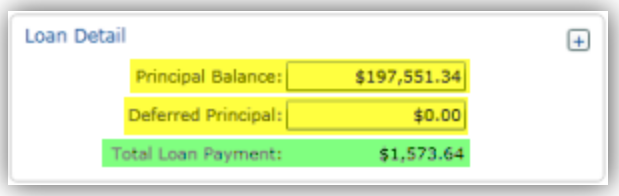
The `BorderGroupExpander` is the preferred control when trying to group information. This control allows for expanding and collapsing the contents to increase/reduce detail. It is also possible to show content when this control is collapsed by using the `BeforeCollapsibleContent` tag and the `AfterCollapsibleContent` tag. The `BeforeCollapsibleContent` tag that is used to show elements above the defined Content when the control is both collapsed and expanded. The `AfterCollapsibleContent` tag that is used to show elements below the defined Content when the control is both collapsed and expanded.

Sample (from PaymentDetail.xaml):

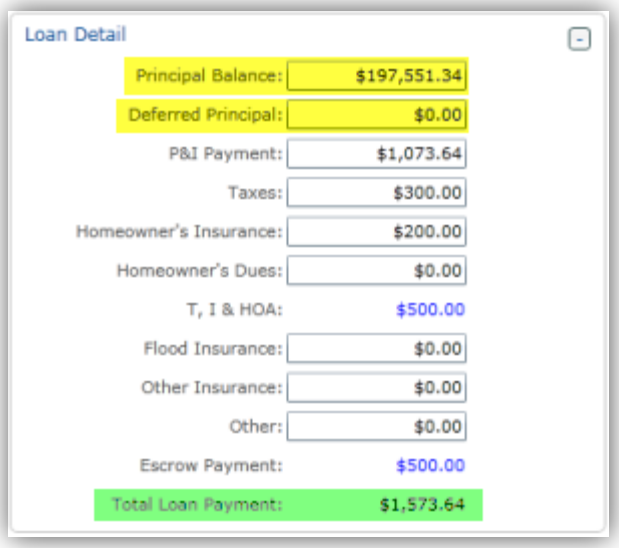
```
<controls:BorderGroupExpander.BeforeCollapsibleContent>
    <controls:DataFieldPanel LabelWidth="175" FieldLabelAlignment="Right" Width="300">
        <controls:DataField DataFieldName="PrincipalBalance" />
        <controls:DataField DataFieldName="DeferredPrincipal" DataObjectName="LoanData" />
    </controls:DataFieldPanel>
</controls:BorderGroupExpander.BeforeCollapsibleContent>
<controls:BorderGroupExpander.Content>
    <controls:DataFieldPanel LabelWidth="175" FieldLabelAlignment="Right" Width="300">
        <controls:DataField DataFieldName="MonthlyPaymentPrincipalInterest" LabelText="P&amp;amp;I Payment" />
        <controls:DataField DataFieldName="MonthlyPaymentTax"  />
        …
    </controls:DataFieldPanel>
</controls:BorderGroupExpander.Content>
<controls:BorderGroupExpander.AfterCollapsibleContent>
    <controls:DataFieldPanel LabelWidth="175" FieldLabelAlignment="Right" Width="300">
        <controls:DataField DataFieldName="MonthlyPayment" LabelText="Total Loan Payment" IsReadOnly="True"/>
    </controls:DataFieldPanel>
</controls:BorderGroupExpander.AfterCollapsibleContent>
```

Result:

Collapsed



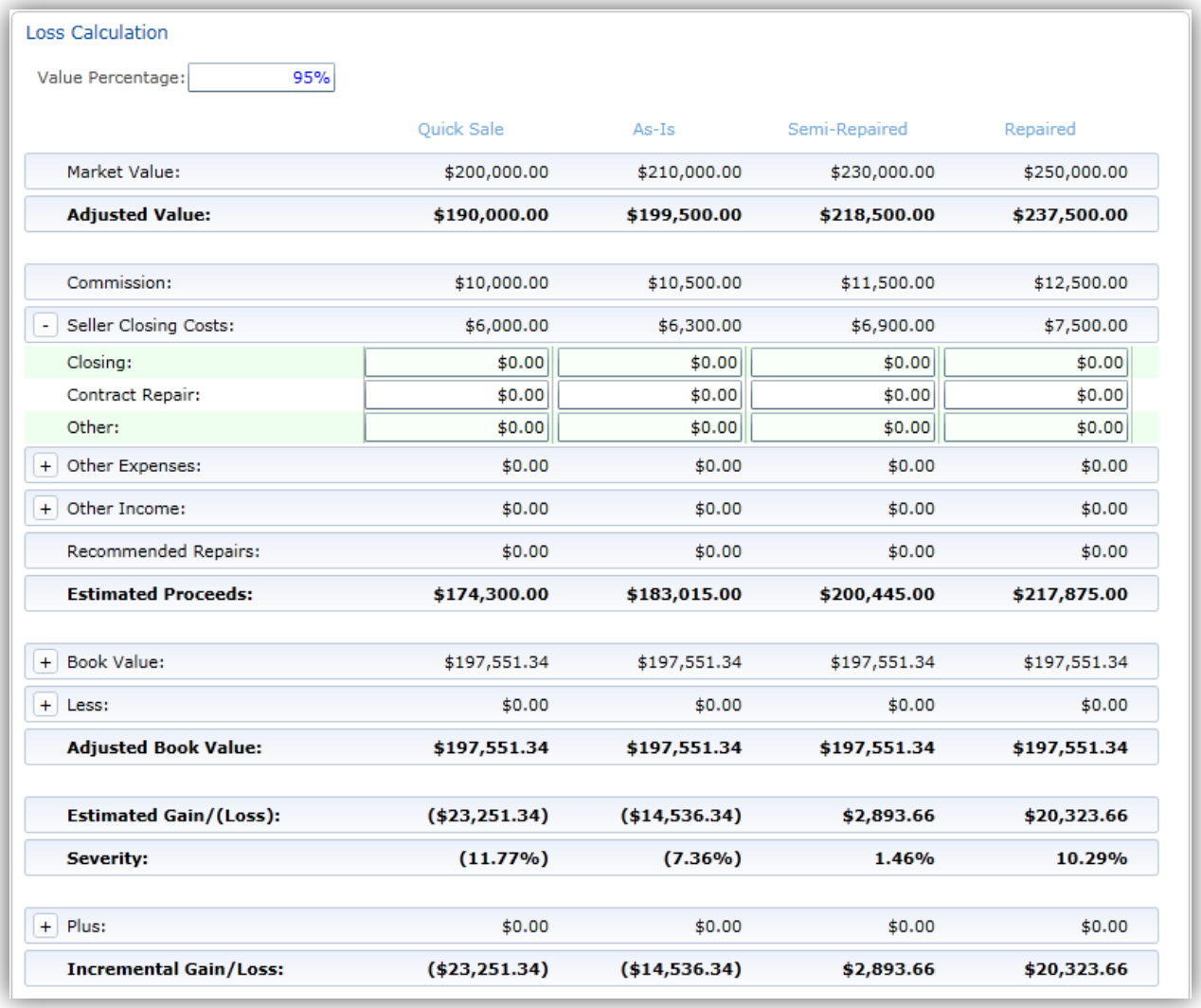
Expanded



ComparisonGrid

This control is used whenever columns of several objects need to be displayed and compared on the same form.

Sample (MarketStrategyDetailsComparisonGrid.xaml):



Loss Calculation

Value Percentage: 95%

	Quick Sale	As-Is	Semi-Repaired	Repaired
Market Value:	$200,000.00	$210,000.00	$230,000.00	$250,000.00
Adjusted Value:	**$190,000.00**	**$199,500.00**	**$218,500.00**	**$237,500.00**
Commission:	$10,000.00	$10,500.00	$11,500.00	$12,500.00
- Seller Closing Costs:	$6,000.00	$6,300.00	$6,900.00	$7,500.00
Closing:	$0.00	$0.00	$0.00	$0.00
Contract Repair:	$0.00	$0.00	$0.00	$0.00
Other:	$0.00	$0.00	$0.00	$0.00
+ Other Expenses:	$0.00	$0.00	$0.00	$0.00
+ Other Income:	$0.00	$0.00	$0.00	$0.00
Recommended Repairs:	$0.00	$0.00	$0.00	$0.00
Estimated Proceeds:	**$174,300.00**	**$183,015.00**	**$200,445.00**	**$217,875.00**
+ Book Value:	$197,551.34	$197,551.34	$197,551.34	$197,551.34
+ Less:	$0.00	$0.00	$0.00	$0.00
Adjusted Book Value:	**$197,551.34**	**$197,551.34**	**$197,551.34**	**$197,551.34**
Estimated Gain/(Loss):	**($23,251.34)**	**($14,536.34)**	**$2,893.66**	**$20,323.66**
Severity:	**(11.77%)**	**(7.36%)**	**1.46%**	**10.29%**
+ Plus:	$0.00	$0.00	$0.00	$0.00
Incremental Gain/Loss:	**($23,251.34)**	**($14,536.34)**	**$2,893.66**	**$20,323.66**

TDS Input Controls

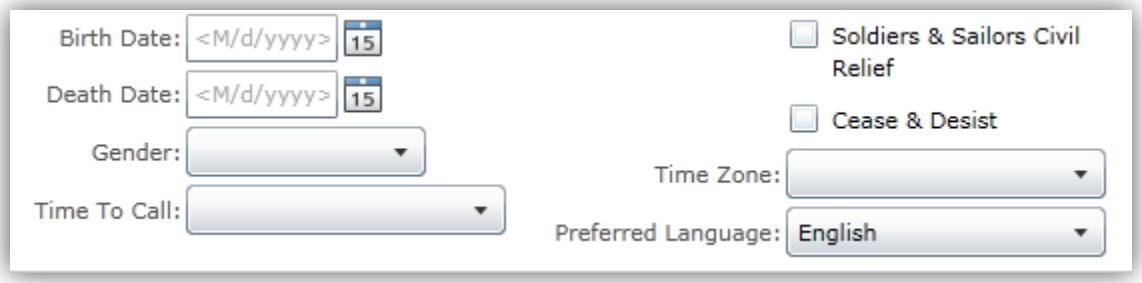
DataField

Most every field control in a `Form` should be a `DataField` control. This control automatically generates the label and input control based on its `DataFieldName` property by looking up the `MetaData` information, then by looking up the database type. For instance, if a field in the database is of type Date, the `DataField` control will create a `DateBox` control.

Sample (PersonalInformation.xaml):

```
<controls:DataFieldPanel Width="300" LabelWidth="130">
    <controls:DataField DataFieldName="BirthDate"/>
    <controls:DataField DataFieldName="DeathDate"/>
    <controls:DataField DataFieldName="Gender"/>
    <controls:DataField DataFieldName="PreferredTimeOfDayId" EditControlWidth="160"/>
</controls:DataFieldPanel>
<controls:DataFieldPanel Width="300">
    <controls:DataField DataFieldName="IsSoldiersAndSailors" EditControlWidth="160" LabelWidth="130"/>
    <controls:DataField DataFieldName="IsCeaseAndDesist" EditControlWidth="160" LabelWidth="130"/>
    <controls:DataField DataFieldName="TimeZoneId" EditControlWidth="160" LabelWidth="130"/>
    <controls:DataField DataFieldName="PreferredLanguageCode" EditControlWidth="160" LabelWidth="130"/>
</controls:DataFieldPanel>
```

Result: Simply providing the DataFieldNames, the DataField controls generate different input controls.



Buttons

When defining a new button, set the `MinWidth` to "**50**". If the button content expands beyond that width, it will automatically stretch horizontally to accommodate its contents. This will help button width consistency for buttons whose content is less than 50 wide. Button heights should generally not be modified.

Layout Guidelines

General

These guidelines are general principles for good form usability.

1. Prioritize data fields by importance.
2. Group related fields into the appropriate layout control.
3. Layout data groups from top to bottom in order of importance
4. Ensure your layout requires minimum user effort for the most vital data.
5. When a total is being calculated, line-up the related fields above the total field.
6. Always align related numerical fields in a column, right-justified.
7. If TDS Controls are used, margins are generally not needed as they have been styled to ease implementation. However, if margins are required for your specific case, the Margin around container elements should be “5”.

Specific

These guidelines point-out specific implementation caveats that should be addressed in order to maintain interface consistency.

Outer Container Width

The first base control defined immediately within the `Form` declaration must have `Width=“800”`. Every subsequent container control defined within this “outer container” should have `HorizontalAlignment=“Stretch”`. Besides ensuring that our minimum screen resolution is supported (1024x768) but also allows widescreen monitors the ability to pin the Bulletin Board when resolution is high enough.

Field Labels

Most of the time the `DataField` control should be used to display an input control with a label. When a different input control is required than what is automatically generated, it is recommended that you still use the `DataField` as a container for the new input control by defining the Content tag of the `DataField`. This ensures consistent label representation of the control. For other exceptions, make sure to use the `LabelTextColor` static-resource as the label’s `Foreground` (`Foreground="{StaticResource LabelTextColor}"`).

Data Field Columns

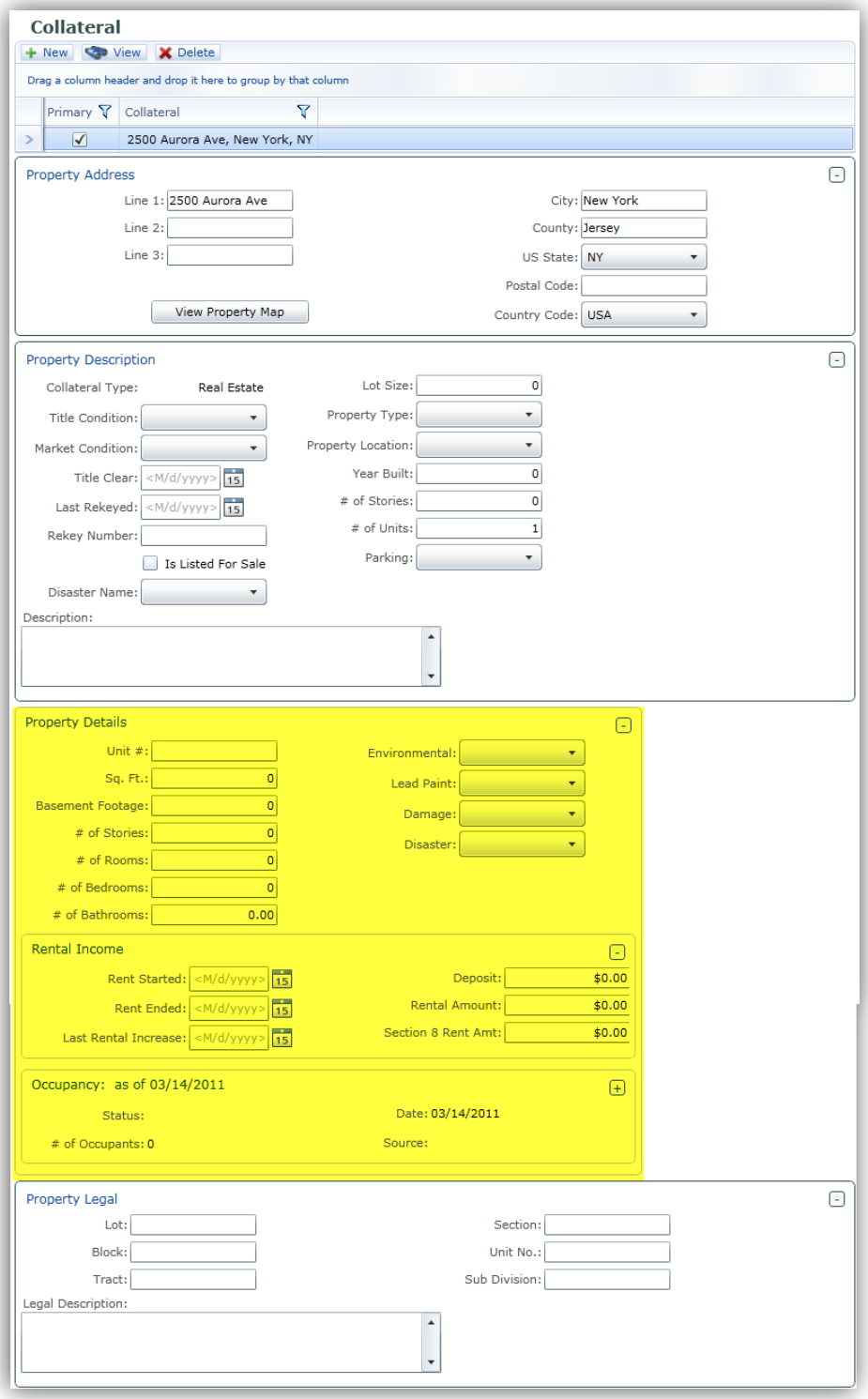
When multiple columns of `DataFields` are needed, they should be laid-out using `DataFieldPanels` and left-aligned.

FormPart

Never set the width of a `FormPart`. Never use the `DataObjectName`.

Example: Conforming to Standards

This section will go over the process of fixing a Form to conform to the standards outlined in this document. This fix will occur to the Collateral screen. The problem areas have been highlighted:



Expand Width of FormPart to Conform

Here we increase the width of the `FormPart` to match the rest of the `PropertyUnitDetails.xaml` `FormParts` in the Form.

XAML:

```
<controls:FormPart x:Class="Dri.Tds.UI.FormParts.PropertyUnitDetails"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:controls="clr-namespace:Dri.Tds.UI.Controls;assembly=Dri.Tds.UI.Controls"
    xmlns:formparts="clr-namespace:Dri.Tds.UI.FormParts;assembly=Dri.Tds.UI.FormParts"
    xmlns:converters="clr-namespace:Dri.Tds.UI.Controls.Converters;assembly=Dri.Tds.UI.Controls"
    HorizontalAlignment="Left">

    <controls:FormPart.Resources>
        <converters:VisibilityConverter x:Key="VisibilityConverter" />
    </controls:FormPart.Resources>
    <controls:BorderGroupExpander x:Name="bgeUnitDetails" Header="Unit Details"
        Summary="{Binding DisplayString}" HorizontalAlignment="Left" IsExpanded="True" HideButton="False">
        <StackPanel>
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition />
                    <ColumnDefinition />
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition />
                    <RowDefinition />
                    <RowDefinition />
                </Grid.RowDefinitions>
                    <controls:DataFieldPanel LabelWidth="120" MinWidth="150" MaxWidth="275">
                    <controls:DataField DataFieldName="UnitNo"/>
                    <controls:DataField DataFieldName="Size"/>
                    <controls:DataField DataFieldName="BasementSize"/>
                    <controls:DataField DataFieldName="NumberOfStories"/>
                    <controls:DataField DataFieldName="NumberOfRooms"/>
                    <controls:DataField DataFieldName="NumberOfBedRooms"/>
                    <controls:DataField DataFieldName="NumberOfBathRooms"/>
                </controls:DataFieldPanel>
                <controls:DataFieldPanel Grid.Column="1" LabelWidth="120" MinWidth="150" MaxWidth="275">
                    <controls:DataField DataFieldName="EnvironmentalProblemId"/>
                    <controls:DataField DataFieldName="LeadPaintProblemId"/>
                    <controls:DataField DataFieldName="PropertyDamageId"/>
                    <controls:DataField DataFieldName="DisasterId" LabelText="Disaster"/>
                </controls:DataFieldPanel>
                <formparts:PropertyCurrentRentalIncome Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="2"
                    Margin="0,5,0,0" Width="585"/>
                <StackPanel Grid.Row="2" Grid.Column="0" Grid.ColumnSpan="2"
                    Visibility="{Binding HasOccupancy, Converter={StaticResource VisibilityConverter}}">
                    <formparts:PropertyCurrentOccupancyPortal DataObjectName="CurrentOccupancy"
                        Margin="0,5,0,0" Width="585"/>
                </StackPanel>
            </Grid>
        </StackPanel>
    </controls:BorderGroupExpander>
</controls:FormPart>
```

1. Start by switching the `HorizontalAlignment` of the `BorderGroupExpander` from "*Left*" to "*Stretch*" for the container controls.
    a. *"Stretch"* allows the control to expand to the width of its parent container, which is essential for imbedded controls.
2. Replace the `Grid` control with a `StackPanel` having Orientation of "*Horizontal*".
3. You're done!
    a. Usually the step is to set the `Width` from the Parent control (in this case `Property.xaml`) but it was already set there.

XAML (final):

```
<controls:FormPart x:Class="Dri.Tds.UI.FormParts.PropertyUnitDetails"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:controls="clr-namespace:Dri.Tds.UI.Controls;assembly=Dri.Tds.UI.Controls"
    xmlns:formparts="clr-namespace:Dri.Tds.UI.FormParts;assembly=Dri.Tds.UI.FormParts"
    xmlns:converters="clr-namespace:Dri.Tds.UI.Controls.Converters;assembly=Dri.Tds.UI.Controls" >

    <controls:FormPart.Resources>
        <converters:VisibilityConverter x:Key="VisibilityConverter" />
    </controls:FormPart.Resources>

    <controls:BorderGroupExpander x:Name="bgeUnitDetails" Header="Unit Details"
        Summary="{Binding DisplayString}" HorizontalAlignment="Stretch" IsExpanded="True"
        HideButton="False">

        <StackPanel Orientation="Vertical">

            <StackPanel Orientation="Horizontal">
                <controls:DataFieldPanel LabelWidth="120" MinWidth="150" MaxWidth="275">
                    <controls:DataField DataFieldName="UnitNo"/>
                    <controls:DataField DataFieldName="Size"/>
                    <controls:DataField DataFieldName="BasementSize"/>
                    <controls:DataField DataFieldName="NumberOfStories"/>
                    <controls:DataField DataFieldName="NumberOfRooms"/>
                    <controls:DataField DataFieldName="NumberOfBedRooms"/>
                    <controls:DataField DataFieldName="NumberOfBathRooms"/>
                </controls:DataFieldPanel>

                <controls:DataFieldPanel LabelWidth="120" MinWidth="150" MaxWidth="275">
                    <controls:DataField DataFieldName="EnvironmentalProblemId"/>
                    <controls:DataField DataFieldName="LeadPaintProblemId"/>
                    <controls:DataField DataFieldName="PropertyDamageId"/>
                    <controls:DataField DataFieldName="DisasterId" LabelText="Disaster"/>
                </controls:DataFieldPanel>
            </StackPanel>

            <formparts:PropertyCurrentRentalIncome Margin="0,5,0,0" />

            <formparts:PropertyCurrentOccupancyPortal DataObjectName="CurrentOccupancy" Margin="0,5,0,0"
                Visibility="{Binding HasOccupancy, Converter={StaticResource VisibilityConverter}}" />
        </StackPanel>

    </controls:BorderGroupExpander>

</controls:FormPart>
```

Screenshot (final):