عنوان: **آشنایی با WPF قسمت چهارم: کنترل ها**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۴/۰۲/۱۹ ۱:۲۰
آدرس: www.dotnettips.info
گروه‌ها: WPF

WPF همانند Windows Form شامل ابزارها یا کنترل‌های داخلی است که می‌توانند در تهیه‌ی یک برنامه بسیار کارآمد باشند. در این بخش به بررسی تعدادی از این کنترل‌ها می‌پردازیم و مابقی آن‌ها را در قسمت‌های آینده بررسی خواهیم کرد. در این نوشتار سعی بر این است که یک فرم ساده را با آن ایجاد کرده و مورد استفاده قرار دهیم.
این فرم دارای اطلاعاتی شامل : نام، جنسیت ، زمینه‌های کاری، کشور، تاریخ تولد و تصویر می‌باشد.

TextBlock
همان Label قدیمی خودمان است که برای نمایش متون کاربر دارد. متن داخل آن بین دو تگ قرار می‌گیرد و یا از خاصیت Text آن کمک گرفته خواهد شد. حتما از خاصیت Width و height آن برای مقداردهی کمک بگیرید، زیرا در غیر آن صورت کل Container خود را خواهد پوشاند. در صورتی که متنی در مکان خود جا نشود می‌توان از دو ویژگی استفاده کرد. آن را برش داد یا به خطوط بعدی شکست. برای حذف یا برش باقی مانده متن می‌توان از خصوصیت TextTrimming استفاده کرد که سه مقدار می‌گیرد:

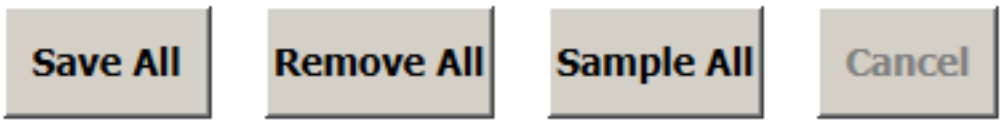
None مقدار پیش فرض

CharacterEllipsis با نزدیک شدن به آخر پهنای کار از ... استفاده می‌نماید. در صورتی که لیستی یا مورد مشابهی دارید میتواند بسیار کاربردی باشد.

WordEllipsis این گزینه هم مانند مورد بالاست با این تفاوت که سعی دارد تا آنجا که ممکن است خود را به آخرین حرف کلمه برساند تا شکستگی در وسط کلمه اتفاق نیفتد و آخرین کلمه کامل دیده شود و بعد ... قرار بگیرد؛ هر چند در تست‌های خودم تفاوتی مشاهده نکردم.

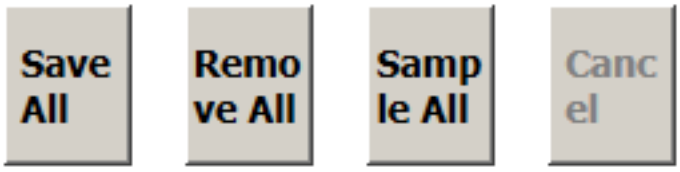
گزینه TextWrapping جهت شکستن یک خط به خطوط است؛ موقعی که متن شما به انتهای صفحه می‌رسد، این ویژگی باعث می‌شود متن به بیرون از پنجره نرفته و یک خط به سمت پایین حرکت کند. این گزینه سه مقدار را دارد:

تصویر زیر حالت اصلی نمایش بدون نیاز به Wrap شدن است:

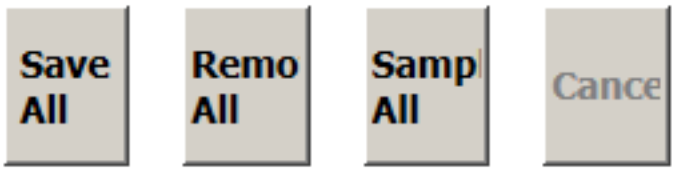


None : مقدار پیش فرض که خصوصیت Wrap را به همراه ندارد.

Wrap : فعال سازی ویژگی TextWrapping

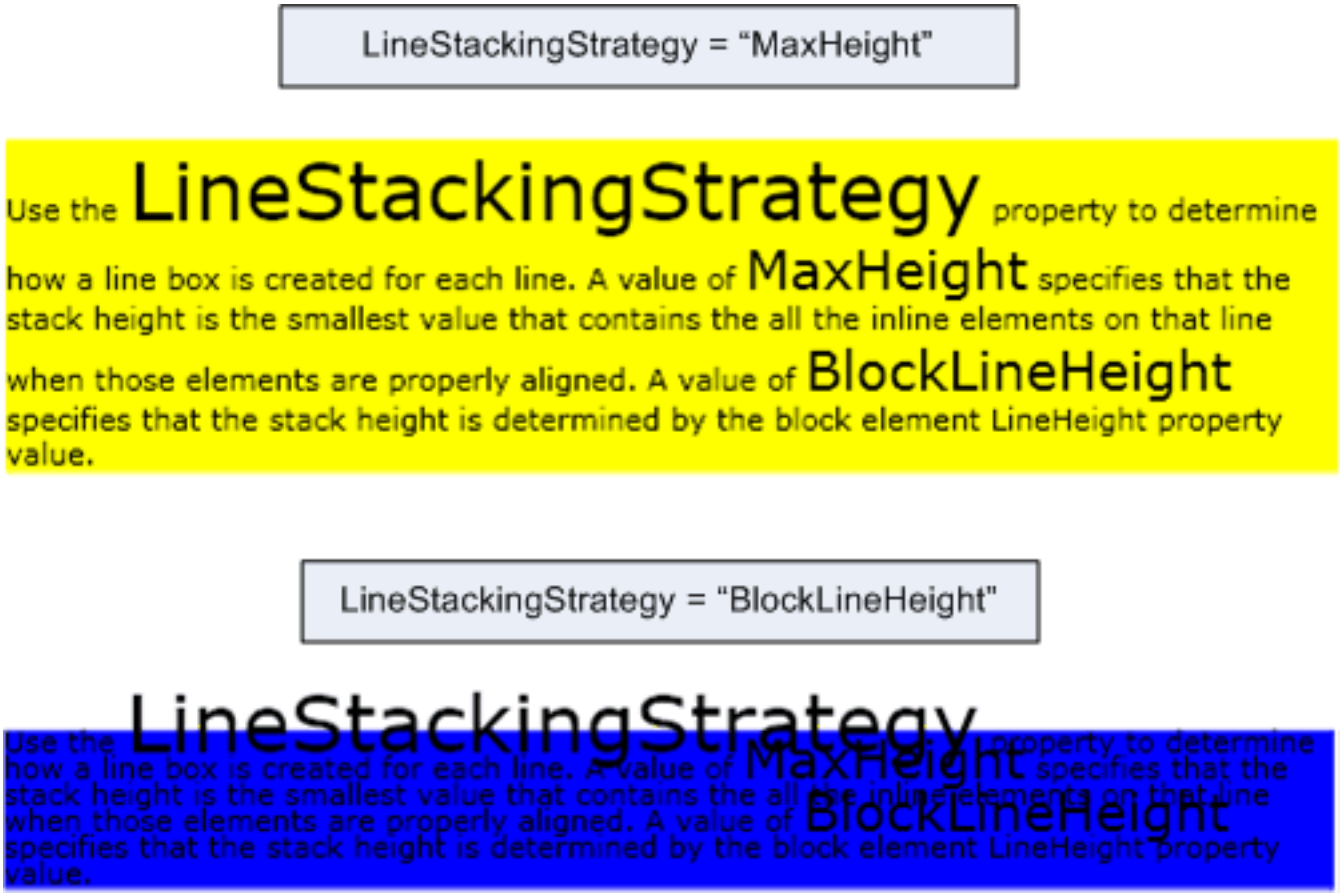


WrapWithOverflow : فرق این گزینه با گزینه بالا در این است که ، گزینه بالا در هر موقعیتی که پیش بیاید عمل خط شکن را روی عبارت یا حتی آن کلمه انجام می‌دهد. ولی در این گزینه فرصت خط شکنی مثل بالا فراهم نیست و اگر روی کلمه‌ای خط شکنی رخ دهد، مابقی آن کلمه از ناحیه‌ی خودش خارج شده و از کلمه‌ی بعدی، خط شکنی صورت می‌گیرد.



خصوصیت **LineStackingStrategy** :

این خصوصیت فاصله‌ی بین خطوط را با استفاده از یک واحد منطقی dp مشخص می‌کند. هر چند دو گزینه دیگر هم دارد که دو تصویر زیر را در این صفحه به شما نمایش می‌دهد:



برای ساخت فرم از یک گرید با سه ستون و 6 سطر استفاده می‌کنم.

```
<Grid Margin="5">
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="Auto"></ColumnDefinition>
        <ColumnDefinition Width="*"></ColumnDefinition>
            <ColumnDefinition Width="Auto"></ColumnDefinition>
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
            <RowDefinition Height="Auto"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>
            <RowDefinition Height="Auto"></RowDefinition>

        </Grid.RowDefinitions>
</Grid>
```

در ستون اول نام فیلدهای مورد نظر را می‌نویسیم و در ستون دوم هم کنترل‌های مد نظر هر فیلد را قرار خواهیم داد. در صورتی که دوست دارید کار از راست به چپ پشتیبانی کند از گزینه OverflowDirection در تگ پنجره Window استفاده نمایید.

در داخل گرید بعد از تعریف سطر و ستون، همانطور که قبلا توضیح دادیم کنترل‌های TextBlock را اضافه می‌کنیم:

```
<TextBlock Grid.Column="0"  Grid.Row="0" VerticalAlignment="Center" HorizontalAlignment="Left"
>Name</TextBlock>
        <TextBlock Grid.Column="0"  Grid.Row="1" VerticalAlignment="Center" HorizontalAlignment="Left"
>Gender</TextBlock>
        <TextBlock Grid.Column="0"  Grid.Row="2" VerticalAlignment="Center" HorizontalAlignment="Left"
>Field Of Work</TextBlock>
        <TextBlock Grid.Column="0"  Grid.Row="3" VerticalAlignment="Center" HorizontalAlignment="Left"
>Country</TextBlock>
        <TextBlock Grid.Column="0"  Grid.Row="4" VerticalAlignment="Center" HorizontalAlignment="Left"
>Birth Date</TextBlock>
```

```
<TextBox Grid.Row="0" Grid.Column="1" Name="Txtname" HorizontalAlignment="Left" Margin="5" Width="200"
></TextBox>
```

برای فیلد نام، از کنترل TextBox استفاده کردم که با محدود کردن Width آن اندازه ثابت به آن دادم. در صورتی که width ذکر نشود یا به Auto ذکر شود، در صورتی که متنی که کاربر تایپ می‌کند، بیش از اندازه تعیین شده کنترل Textbox باشد، کنترل هم همراه متن بزرگتر خواهد شد و تا پایان محدوده سلولی اش در گرید کش خواهد آمد.

Buttons

برای فیلد جنسیت Gender هم از RadioButton کمک گرفتم که با استفاده از خاصیت GroupName می‌توان دسته‌ای از این کنترل‌ها را با هم مرتبط ساخت تا با انتخاب یک آیتم جدید از همان گروه، آیتم قبلی که انتخاب شده بود از حالت انتخاب خارج شده و آیتم جدیدی انتخاب شود. از خاصیت IsChecked می‌توان برای انتخاب یک آیتم بهره برد.

به صورت کلی دکمه‌ها به چند دسته زیر تقسیم می‌شوند:

Button

ToggleButton

CheckBox

RadioButton

که همگی این عناصر از کلاسی به نام ButtonBase مشتق شده اند. کد زیر RadioButtonها را به صورت عمودی چینش کرده است:

```
<StackPanel Orientation="Vertical" Grid.Row="1" Grid.Column="1" Margin="10">
            <RadioButton GroupName="Gender" Name="RdoMale" IsChecked="True" >Male</RadioButton>
            <RadioButton GroupName="Gender" Name="RdoFemale" Margin="0 5 0 0" >Female</RadioButton>
        </StackPanel>
```

برای فیلد زمینه کاری ، لیست کشورها و تاریخ تولد از کدهای زیر کمک گرفتم:

```
<StackPanel Orientation="Horizontal" Grid.Row="2" Grid.Column="1" Margin="10">
            <CheckBox Name="ChkActor" >Actor/Actress</CheckBox>
            <CheckBox Name="ChkDirector" >Director</CheckBox>
            <CheckBox Name="ChkProducer" >Producer</CheckBox>
        </StackPanel>

        <ListBox Grid.Row="3" Grid.Column="1" Margin="10"  Height="80">
        <ListBoxItem>
                <TextBlock>UnitedStates</TextBlock>
            </ListBoxItem>
            <ListBoxItem>
                <TextBlock >UK</TextBlock>
            </ListBoxItem>
            <ListBoxItem>
                <TextBlock >France</TextBlock>
            </ListBoxItem>
            <ListBoxItem>
                <TextBlock >Japan</TextBlock>
            </ListBoxItem>
        </ListBox>

        <Calendar Grid.Row="4" Grid.Column="1" HorizontalAlignment="Left" Margin="10"></Calendar>
```

برای لیست کشورها می‌توان از یک ListBox یا ComboBox استفاده کرده که هر آیتم داخل آنها در یک تگ ListBoxItem یا ComboBoxItem قرار می‌گیرد. اگر از حالت ListBox استفاده می‌کنید، در صورتی که آیتم‌ها از ارتفاع لیست بیشتر شود به طور خودکار یک Scrollbar برای آن‌ها در نظر گرفته خواهد شد و نیازی نیست که آن را دستی اضافه کنید.

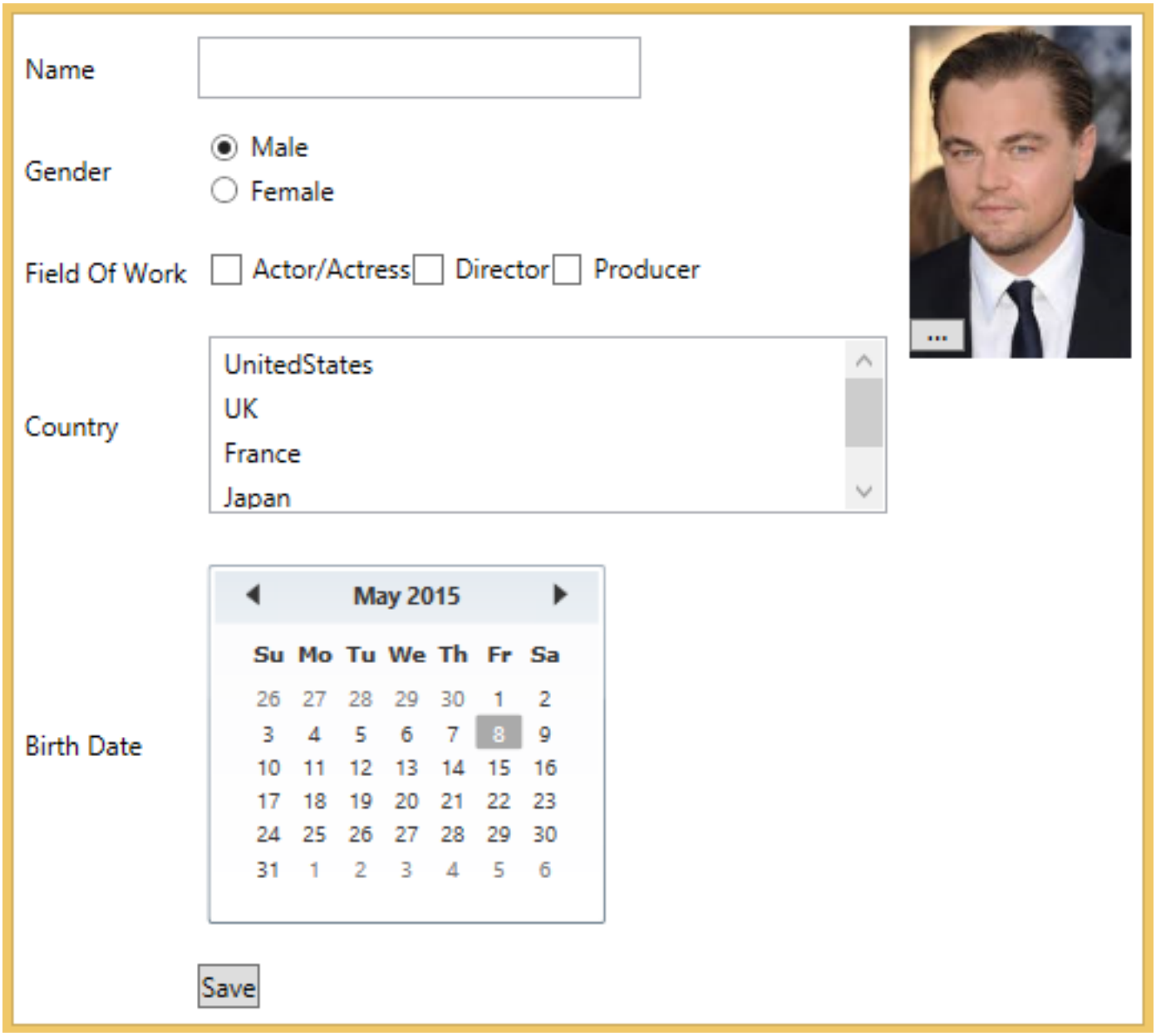
برای تصویر شخص، قصد دارم آن را در گوشه‌ی سمت راست و بالا قرار دهم. برای همین محل ستون آن را ستون سوم یا

اندیس دوم انتخاب کرده و از آنجا که این عکس حالت پرسنلی دارد، می‌تواند چند سطر را به خود اختصاص دهد که با کمک خاصیت Rowspan، چهار سطر، کنترل را ادامه دادم. برای ستون‌ها هم می‌توان از خاصیت ColumnSpan استفاده کرد. همچنین دوست دارم یک دکمه هم روی تصویر در گوشه‌ی سمت چپ و پایین قرار داده که کاربر با انتخاب آن به انتخاب عکس یا تغییر آن بپردازد. برای همین از یک پنل گرید استفاده کردم و کنترل دکمه را روی تصویر قرار دادم. همپوشانی کنترل‌ها در اینجا صورت گرفته است.

```
<Grid Grid.Row="0" Grid.Column="2" Grid.RowSpan="4">
            <Grid.RowDefinitions>
                <RowDefinition Height="*"></RowDefinition>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="*"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Image HorizontalAlignment="Right" Source="man.jpg" Stretch="UniformToFill"
VerticalAlignment="Top" Width="100" Height="150"></Image>
            <Button Width="25" Height="15"    Padding="0"  HorizontalAlignment="Left"
VerticalAlignment="Bottom" Margin="0,0,0,83">
                 <TextBlock VerticalAlignment="Center" Margin="0 -7 0 0">...</TextBlock>
            </Button>
        </Grid>
```

خاصیت **Stretch** کنترل Image در بالا، نحوه‌ی نمایش تصویر را نشان می‌دهد که چهار مقدار دارد:
None : تصویر، اندازه‌ی اصلی خود را حفظ کرده و هر مقدار آن که در کنترل جا شود، نمایش می‌یابد و بسته به سایز تصویر ممکن است گوشه هایی از تصویر نمایش نیابد.
Fill : تصویر را داخل کنترل به زور جا داده تا پهنا و ارتفاع عکس، هم اندازه کنترل می‌شود.
Uniform : تصویر بزرگ را با در نظر گرفتن نسبت پهنا و ارتفاع تصویر، با یکدیگر در کنترل جا می‌دهد.
UniformToFill : تصویر، کل کنترل را می‌گیرد ولی نسبت پهنا و عرض را حفظ کرده ولی قسمت هایی از تصویر در کنترل دیده نمی‌شود.

همانطور که قبلا هم گفتیم، خود کنترل دکمه شامل زیر کنترل‌هایی می‌شود که یکی از آن‌ها TextBlock است و از طریق خصوصیت *.TextBlock دیگر خصوصیات آن قابل تنظیم است و البته برای خصوصی سازی بیشتر هم می‌توان یک TextBlock را به صورت Nested یعنی داخل تگ Button تعریف کنید که ما همین کار را کرده ایم.
فرم نهایی ما به صورت زیر است:



در صورتی که دوست دارید جهت ListBox را از عمودی به افقی تغییر دهید می‌توانید از پنل‌های Stack یا Wrap استفاده کنید که تعریف آن به شکل زیر است:

```
<ListBox>
    <ListBox.ItemsPanel>
        <ItemsPanelTemplate>
            <VirtualizingStackPanel Orientation="Horizontal" />
        </ItemsPanelTemplate>
    </ListBox.ItemsPanel>
</ListBox>
```

بدین صورت ListBox به شکل زیر تغییر می‌یابد:

و در صورتی که می‌خواهید Scroll حذف شود و از Wrap استفاده کنید، کد را به شکل زیر تعریف کنید. فراموش نکنید که اسکرول افقی را غیرفعال کنید؛ وگرنه نتیجه کار به شکل بالا خواهد بود.

```
<ListBox ScrollViewer.HorizontalScrollBarVisibility="Disabled">
    <ListBox.ItemsPanel>
        <ItemsPanelTemplate>
            <WrapPanel />
        </ItemsPanelTemplate>
    </ListBox.ItemsPanel>
</ListBox>
```

نتیجه:

Calendar

تقویم یکی دیگر از کنترل‌های موجود است که شامل خصوصیات زیر است:

DisplayDate : تاریخ پیش فرض و اولیه تقویم را مشخص می‌کند؛ در صورتی که ذکر نشود تاریخ جاری درج می‌شود.

```
<Calendar DisplayDate="01.01.2010" />
```

از خصوصیات دیگر در این زمینه می‌توان به *DisplayDateStart* و *DisplayDateEnd* اشاره کرد که محدوده‌ی نمایش تاریخ تقویم را مشخص می‌کند. کد زیر تنها تاریخ‌های روز اول ماه ابتدای سال 2015، تا روز اول ماه پنجم 2015 را نمایش می‌دهد:

```
<Calendar DisplayDateStart="01.01.2015" DisplayDateEnd="05.01.2015" />
```

SelectionMode : نحوه‌ی انتخاب تاریخ را مشخص می‌کند:

SingleDate : فقط یک تاریخ قابل انتخاب است.

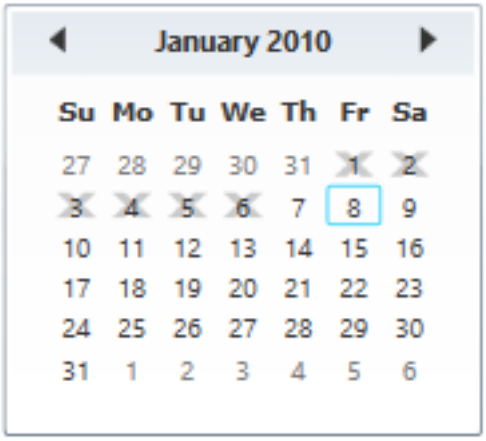
SingleRange : می‌توانید از یک تاریخ تا تاریخ دیگر را انتخاب کنید. ولی نمی‌توانید مجددا چند انتخاب دیگر را در جای جای تقویم داشته باشید. مثلا از تاریخ 5 آپریل تا 10 آپریل را انتخاب کرده‌اید؛ ولی دیگر نمی‌توانید تاریخ 15 آپریل یا محدوده‌ی 15 آپریل تا 20 آپریل را انتخاب کنید. چون تنها قادر به انتخاب یک رنج یا محدوده تاریخی هستید.

MultipleRanges : بر خلاف گزینه‌ی بالایی هر محدوده تاریخی قابل انتخاب است.

```
<Calendar SelectionMode="MultipleRange" />
```

نکته بعدی در مورد غیرفعال کردن بعضی از تاریخ هاست که شما قصد ندارید به کاربر اجازه دهید آن‌ها را انتخاب کند. برای مثال تاریخ‌های 1 آپریل تا 10 آپریل را از دسترس خارج کنید. برای همین از خصوصیت BlackoutDates استفاده می‌کنیم که نحوه‌ی تعریف آن به شرح زیر است که در این کد دو محدوده‌ی تاریخی غیر فعال شده اند:

```
<Calendar>
    <Calendar.BlackoutDates>
        <CalendarDateRange Start="01/01/2010" End="01/06/2010" />
        <CalendarDateRange Start="05/01/2010" End="05/03/2010" />
    </Calendar.BlackoutDates>
</Calendar>
```



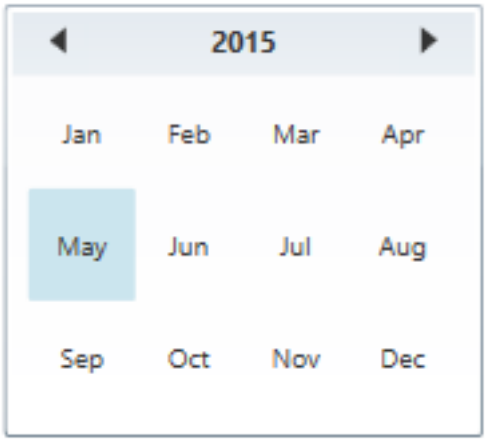
DisplayMode : به طور پیش فرض، تقویم ماه‌ها را نشان می‌دهد ولی میتوانید آن را توسط این خصوصیت روی سال یا دهه و ماه هم تنظیم کنید.

با انتخاب سال *Year* ، تقویم ماه‌های یک سال را نمایش می‌دهد.

با انتخاب دهه *Decades* سال‌های یک دهه‌ی تعیین شده را نشان می‌دهد و با انتخاب ماه Month روزهای هر ماه در آن نمایش داده می‌شود.

در هنگام انتخاب این گزینه، به داخل تقویم نگاه نکنید، بلکه به سر تیتر آن نگاه کنید.

```
<Calendar DisplayMode="Year" />
```


2015
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec