现代操作系统应用开发实验报告

学号： 15331248　　**班级　：** 上午班

姓名： 潘承远　　**实验名称：** Todos(HomeWorks3)

一．参考资料

- **《数据绑定 PPT》**：通过老师给的 PPT 大致了解数据绑定的概念。
- **《05-AdaptiveCode》**：了解了自适应 UI 的一些代码例子
- https://docs.microsoft.com/zh-cn/windows/uwp/data-binding/data-binding-quickstart

  这个网站让我更加深入地了解数据绑定
- https://msdn.microsoft.com/zh-cn/library/windows/apps/system.windows.uielement.visibility.aspx

  这个网站让我了解了 UIElement.Visibility 属性和它地属性值，让我知道它的属性只有 Visible 和 Collapsed，让我能够完成自适性 UI 时对不同 UI Element 的显示控制。
- https://msdn.microsoft.com/en-us/zhcn/library/system.windows.visualstategroup(v=vs.95).aspx

  这个网站让我了解了 VisualStateGroup 类
- https://msdn.microsoft.com/zh-cn/library/system.windows.setter(v=vs.110).aspx

  这个网站让我了解了 Setter 类

二．实验步骤

- 新建一个项目，根据题目要求，参考教程以及 demo 进行 UI 的设计；
- 实现“为 ListView 设置 VisualStateGroup，使得窗口宽度小于 600 时，image 不显示”。我新建立了一个 ListView，这个 ListView 没有图片，只有 CheckBox、Line 和 TextBlock。
- 实现“在宽屏显示两列的情况下，点击 ADD Button 不会跳转到 EditToDos 页面”。实现的代码如下：

```
private void AddAppBarButton_Click(object sender, RoutedEventArgs e)
{
    ViewModel.SelectedItem = null;
    if (InlineToDoItemViewGrid.Visibility == Visibility.Collapsed)
        Frame.Navigate(typeof(NewPage), ViewModel);
}
```

- 完成“勾选 CheckBox，Line 才会出现”，这是我花时间最多的地方，花了有半天的时间。因为 ListView 里面全部是 item 模板，所以需要用到数据绑定。我采用的方法是将 CheckBox 的 IsChecked 属性绑定 TodoItem 的 completed，并且为双向绑定，Line 的 Opacity 属性也绑定 TodoItem 的 completedLine，并且建立一个 CheckBox 的 Click 事件，使得点击 CheckBox 时刷新界面，让 Line 的 Opacity 知道属性发生了变化。因为勾选 CheckBox 后，如果不刷新的话，Line 是没有即时被通知到 TodoItem 的属性发生了变化的。实现的代码如下：

```
<CheckBox Grid.Column="0" VerticalAlignment="Center" Height="32" Width="32"  IsChecked="{Binding completed, Mode=TwoWay}" Clic
<Image Grid.Column="1" Source="Assets/background.jpg" Height="90" Width="90" Margin="0,3,12,7"/>
<TextBlock Text="{x:Bind title }" Grid.Column="2" VerticalAlignment="Center" Foreground="Black" FontWeight="Normal" FontSize="
<Line Opacity ="{Binding completedLine}" x:Name="Line_Main" Grid.Column="2" Stretch="Fill" Stroke="Black" StrokeThickness="1"
<AppBarButton Grid.Column="3" Icon="Setting" IsCompact="True" VerticalAlignment="Center">
```

```
private void CheckBox_bit(object sender, RoutedEventArgs e)
{
    Frame.Navigate(typeof(MainPage), ViewModel);  // 刷新界面
}
```

●完成“实现 Update,Delete 函数”，代码如下

```
public void RemoveTodoItem(Models.TodoItem id)
{
    this.allItems.Remove(id);  //  Delete一个Item
    this.selectedItem = null;
}

public void UpdateTodoItem(string time, string titl, string descriptio)
{
    this.selectedItem.title = titl;  // Update一个Item
    this.selectedItem.description = descriptio;
    this.SelectedItem.SetTime(time);
    this.selectedItem = null;
}
```

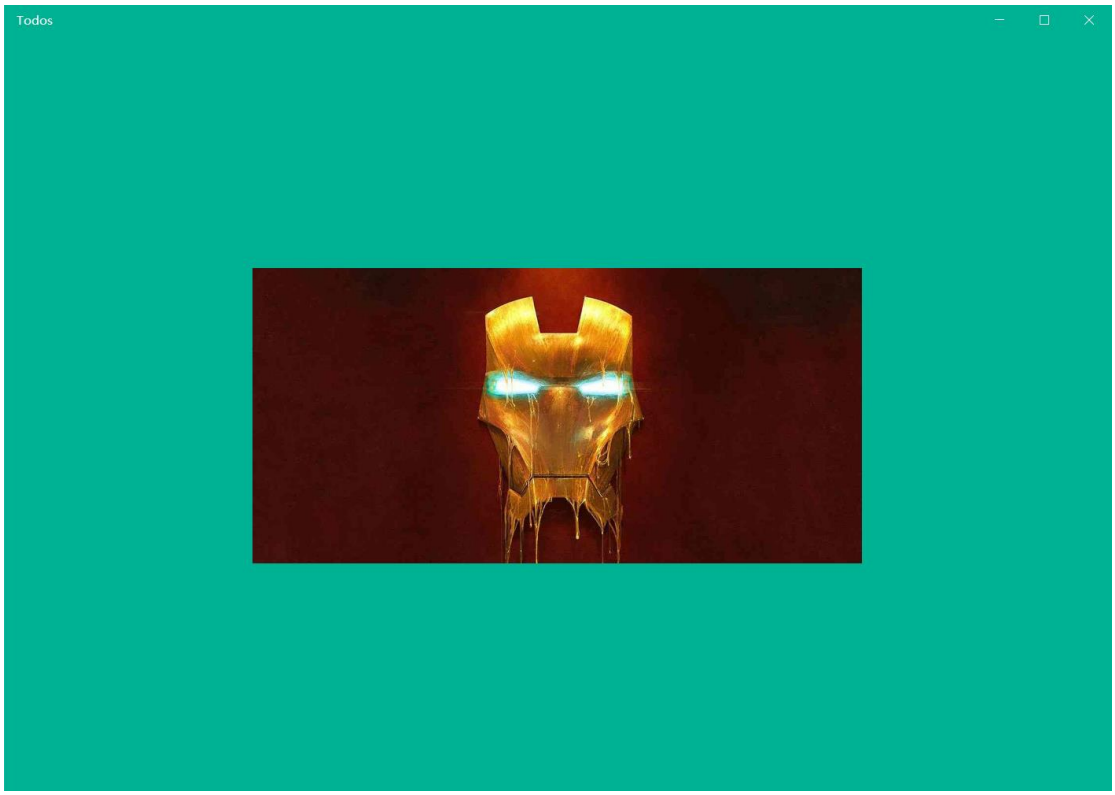
●完成“在窄屏显示一列的情况下删除或添加后要跳转到 Mainpage 界面”只需在对应函数添加下面的代码：

```
Frame.Navigate(typeof(MainPage), ViewModel);
```

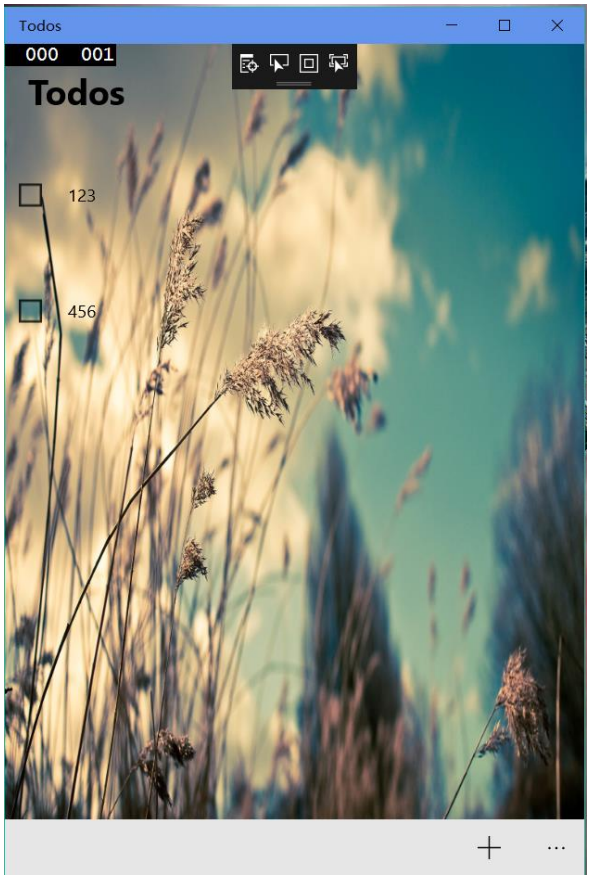
●进行了美化，添加了操作界面背景和程序初始化的图片。

三．实验结果截图

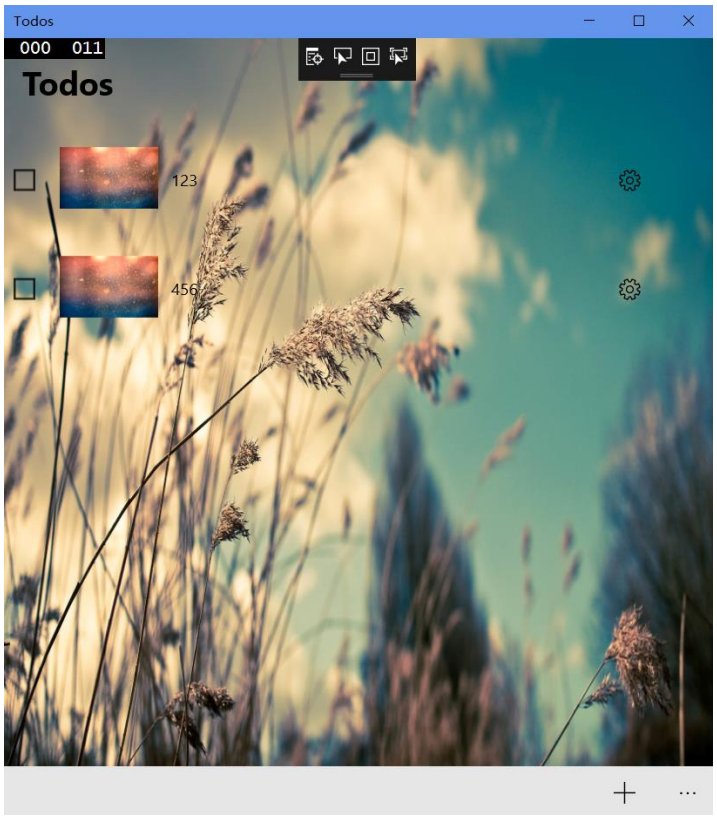
1. 程序的初始屏幕



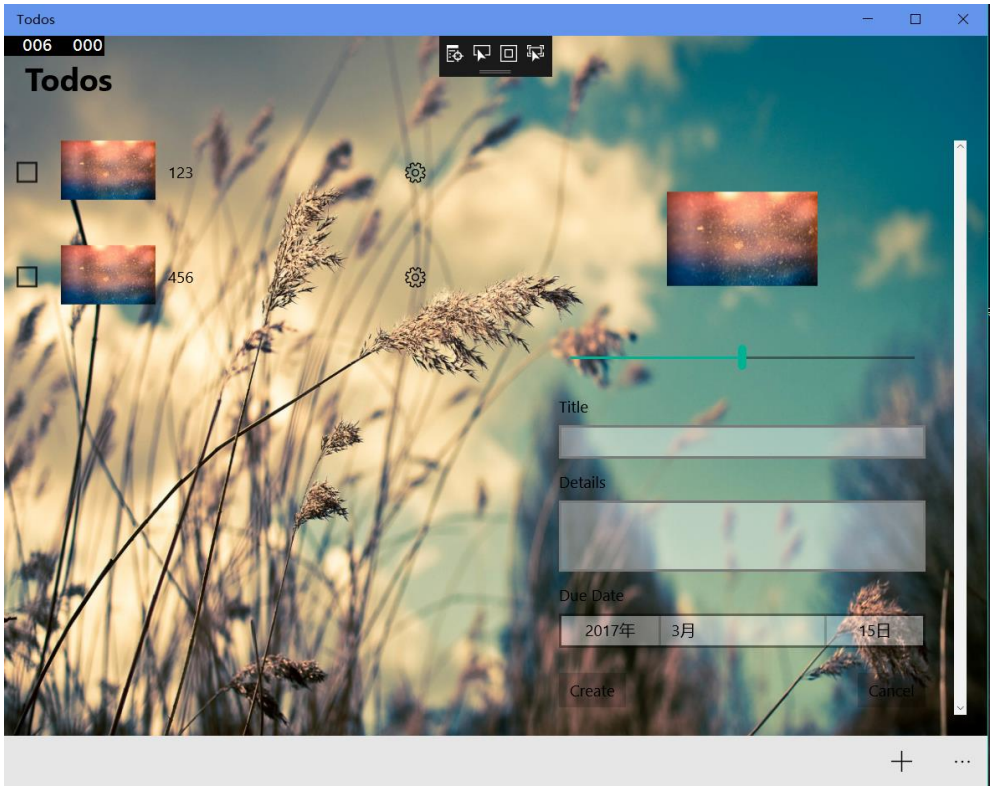
2. 当窗口宽度小于 600 时，Image 不显示



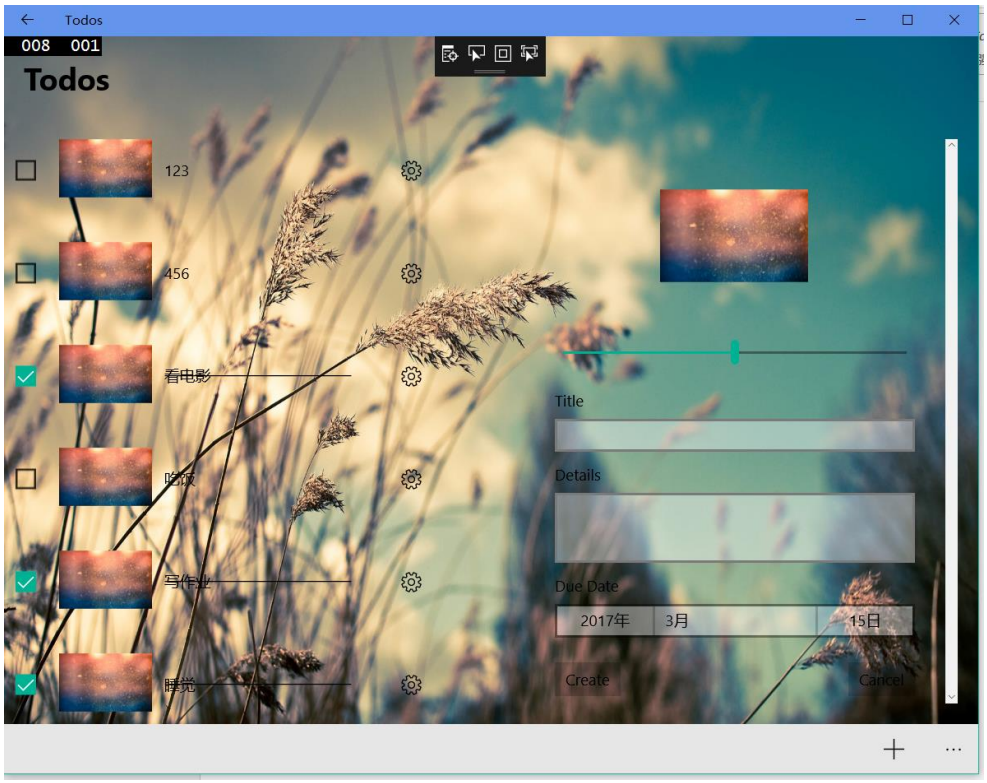
2. 当窗口宽度大于 600 小于 800 时， 只显示一列，此时会有一个 AppBarButton 在 Line 旁边，用于修改和删除 Item



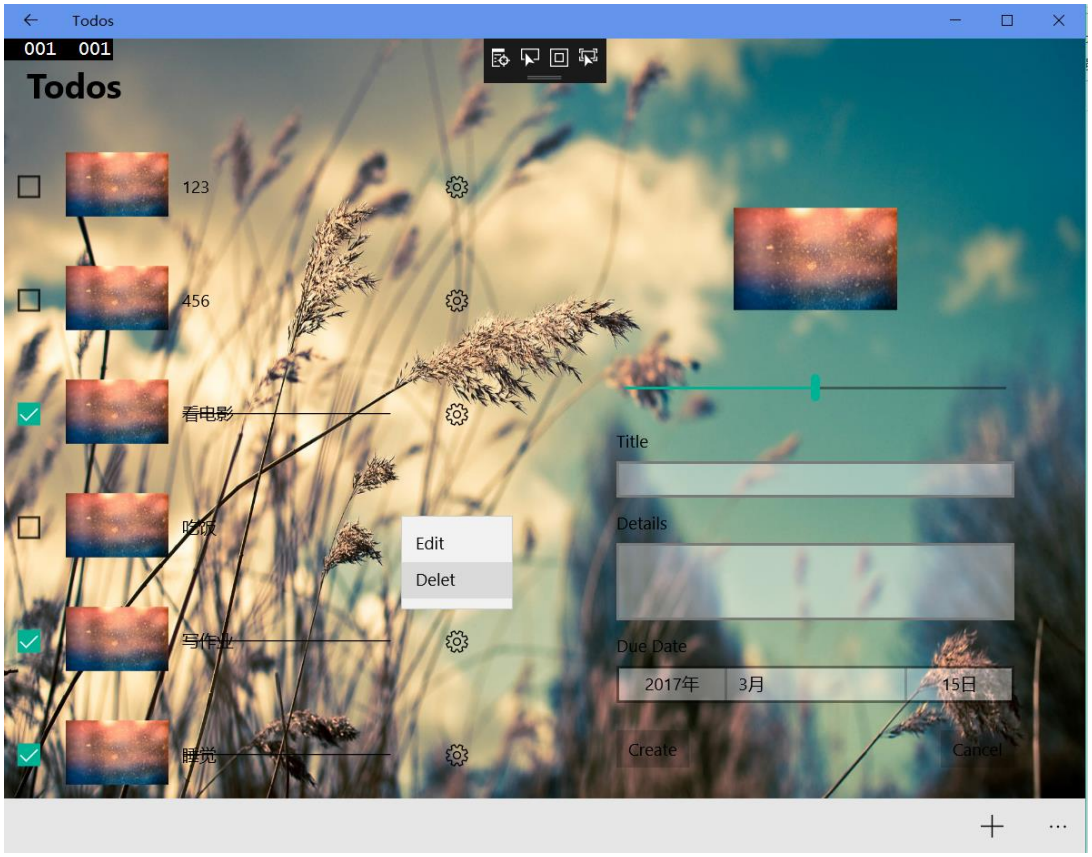
3. 当窗口宽度大于 800 时， 显示两列，此时点击右下角的“+”号是不能跳转到 NewPage 的，只有在一列的情况下可以跳转



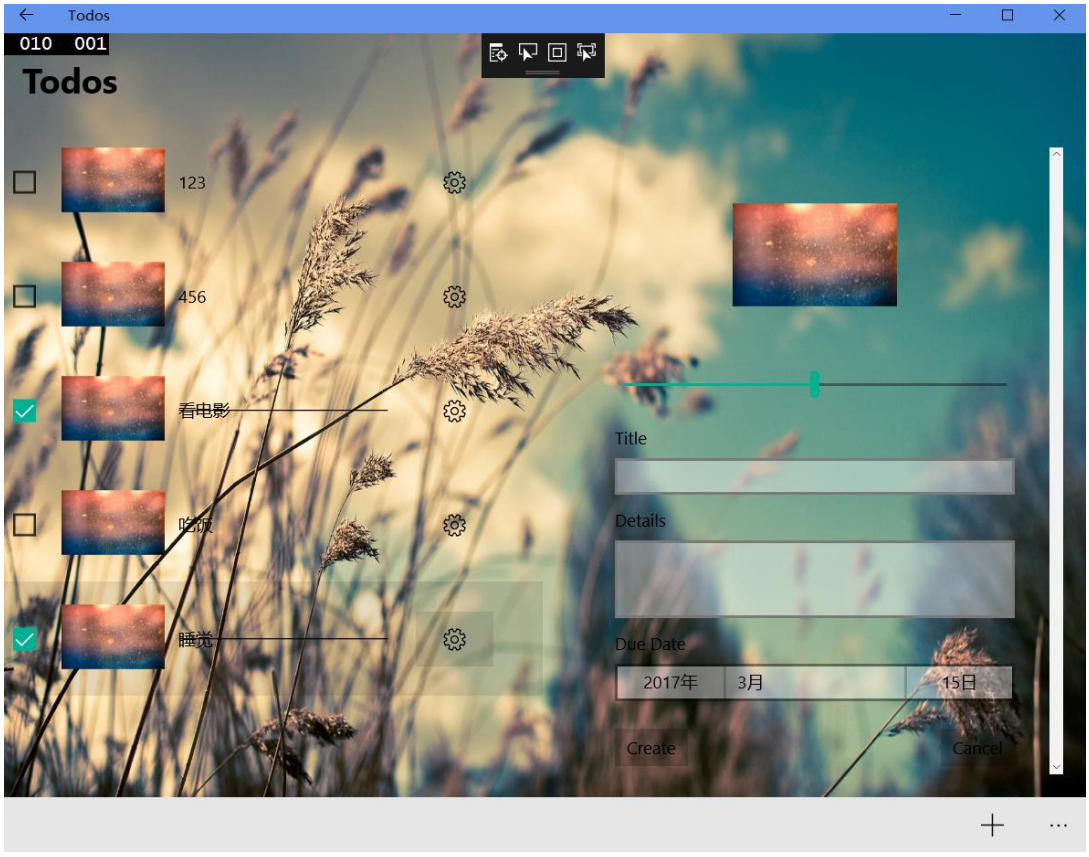
4. 当窗口宽度大于 800 时，在 MainPage 创建另外 4 个待办事项，并且勾选三项，因为我把 Line 与 TodoItem 类对象的属性绑定在一起，所以只要对象（待办事项）没有被删除，Line 就不会随着界面跳转而消失，而是会根据待办事项有没有完成来决定是否要显示。



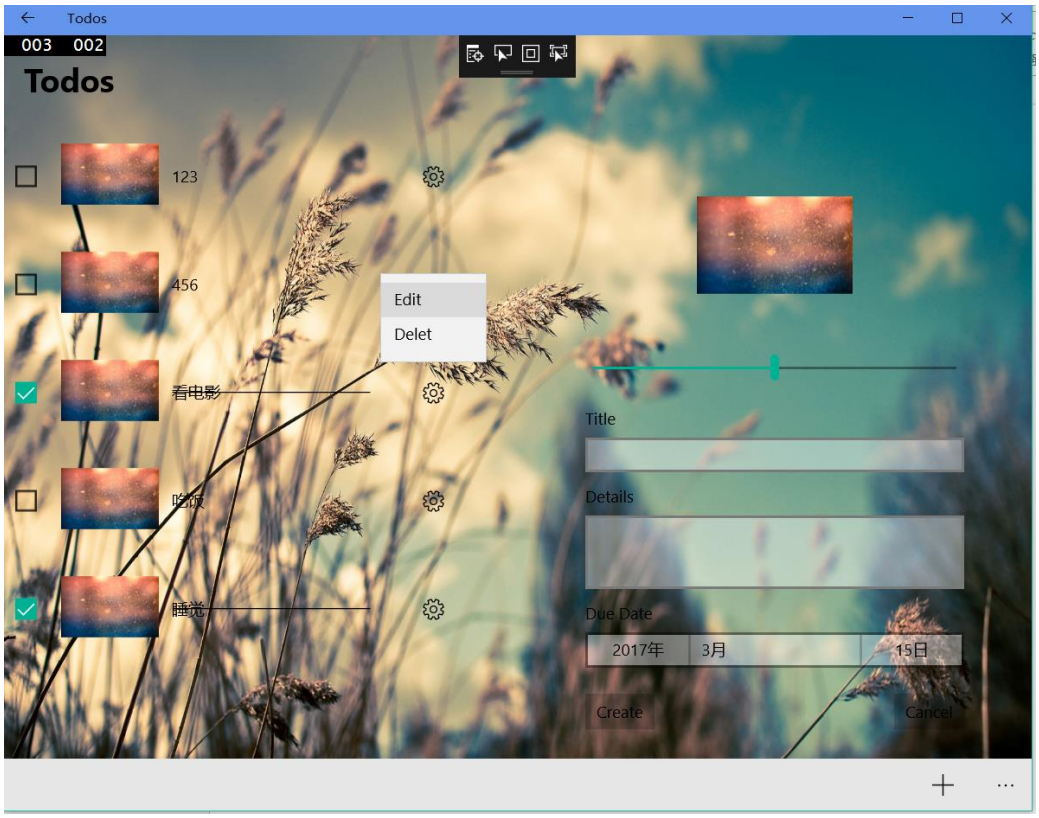
5. 点击“写作业”这条待办事项的 AppBarButton，对它进行删除



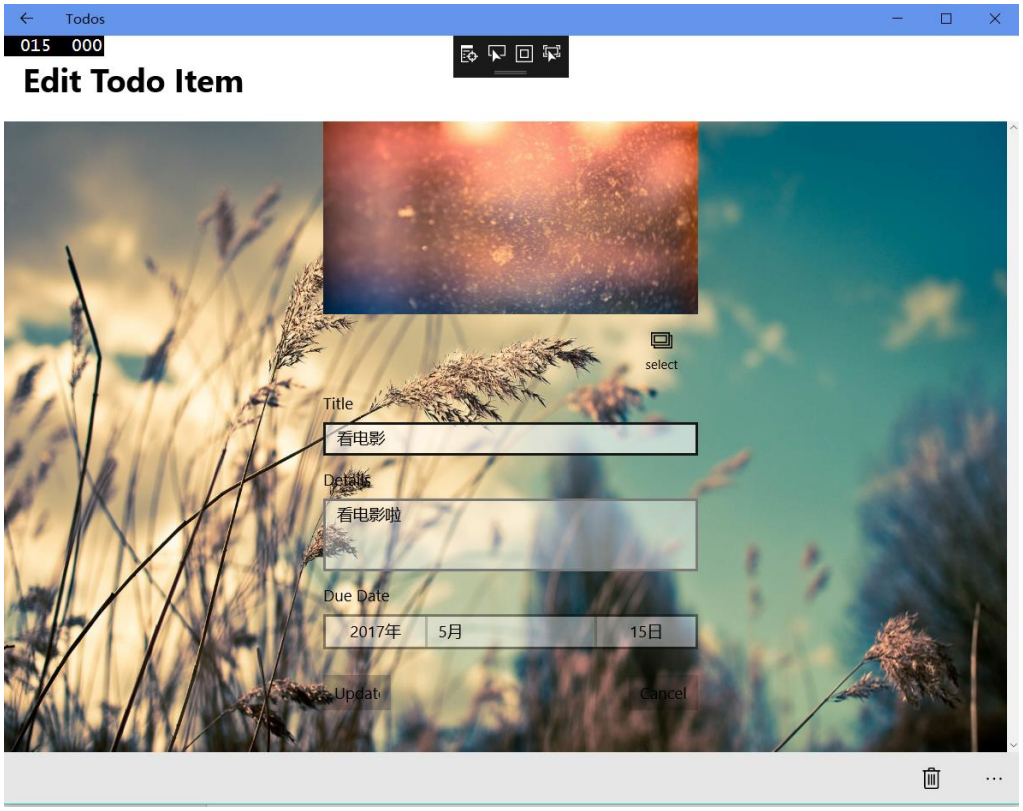
6. 将“写作业”这条待办事项删除后



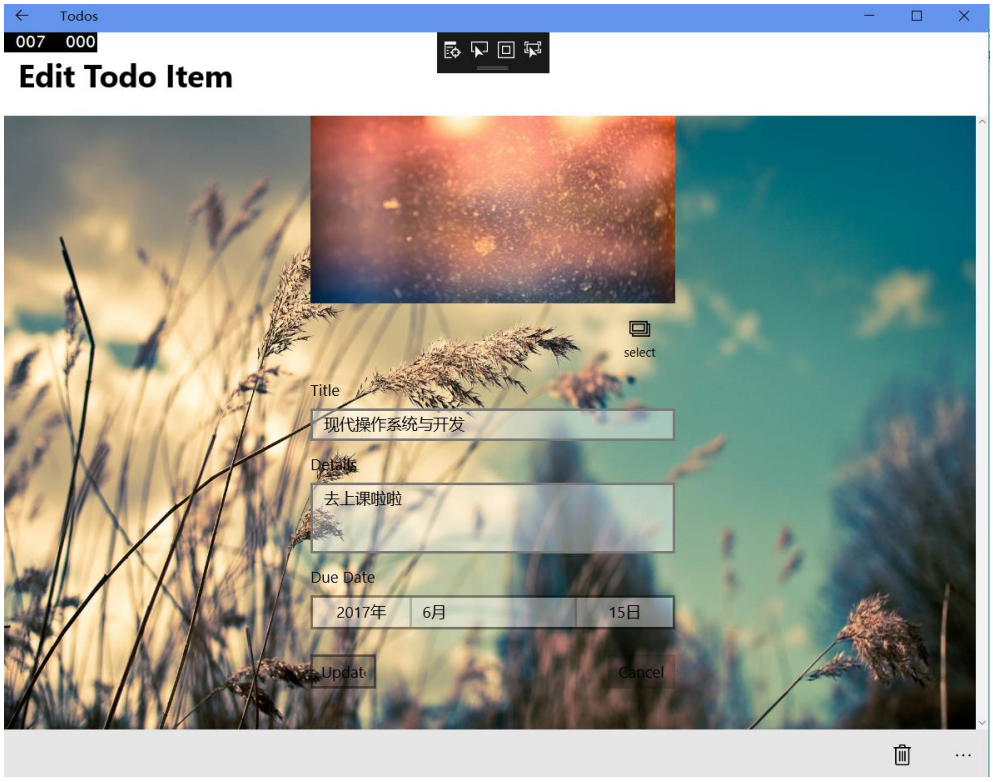
7. 点击“看电影”这条待办事项的 AppBarButton，对它进行 Edit



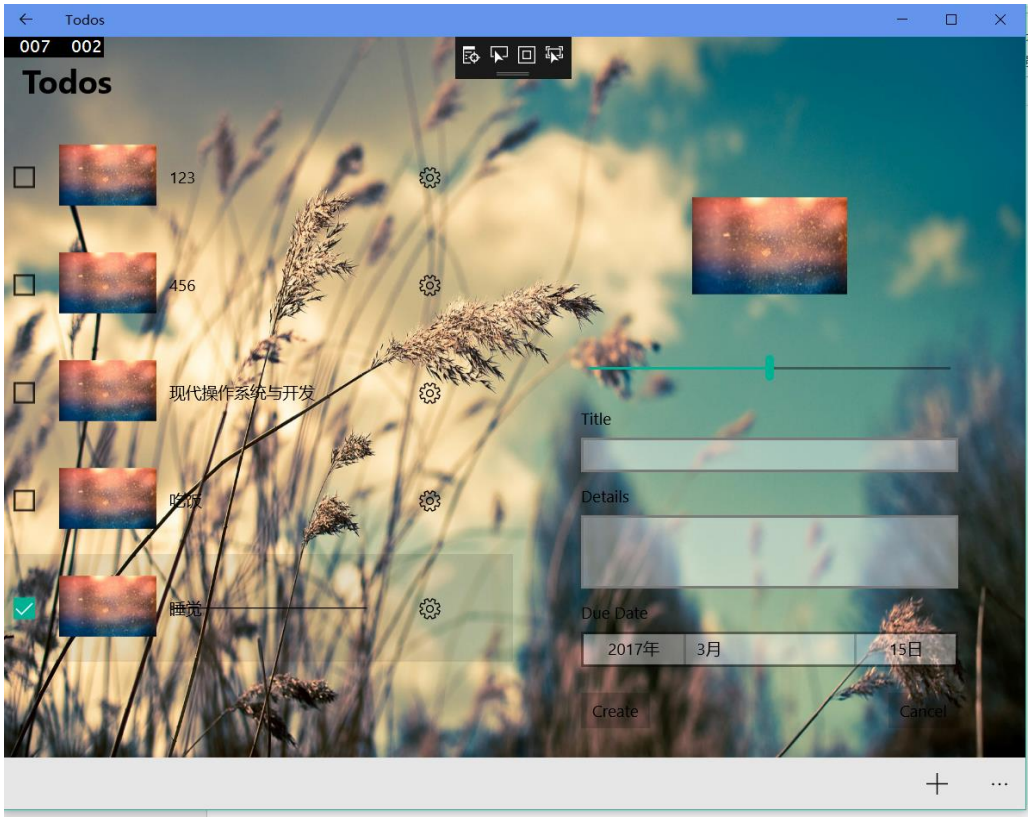
8. 跳转进入“看电影”这条待办事项的修改界面



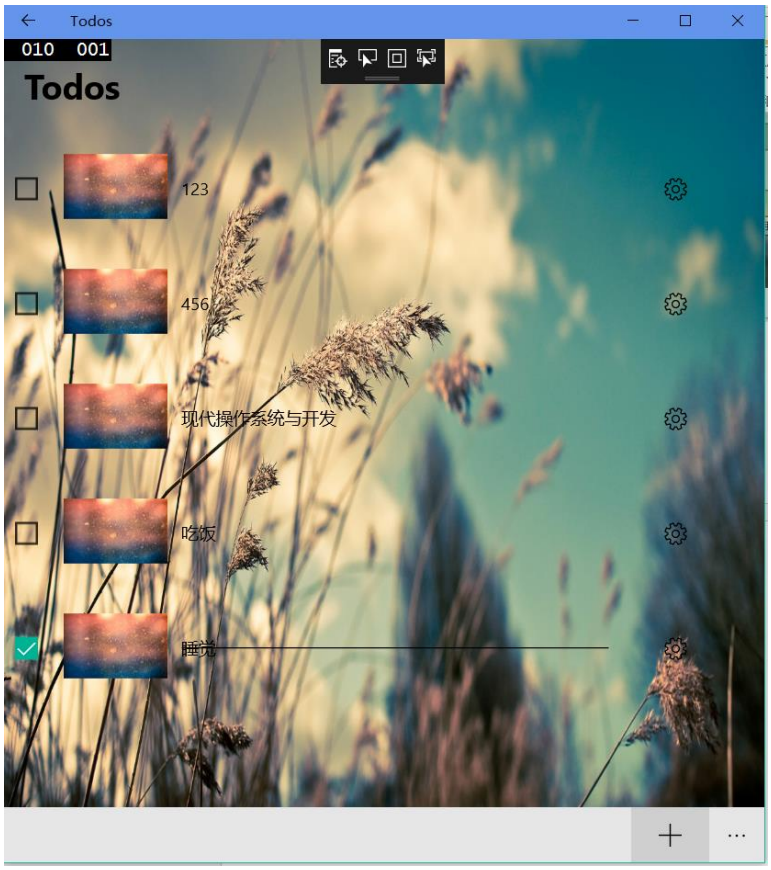
9. 将“看电影”这条待办事项修改成如下所示



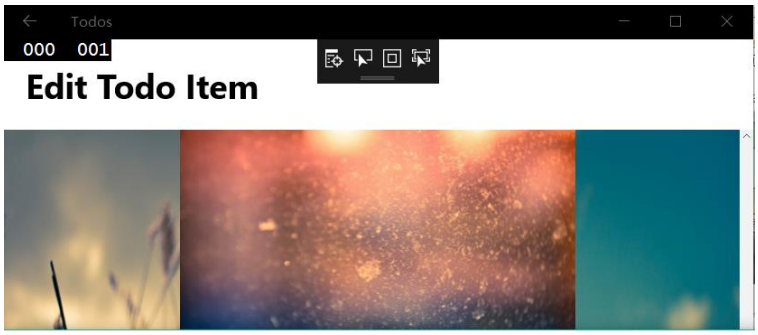
10. 点击“Update”建，并跳回 Main Page，因为是更新了待办事项，所以将它的 completed 设置为 false，此时 Line 不显示



11. 在只显示一列的情况下点击右下角的“+”建



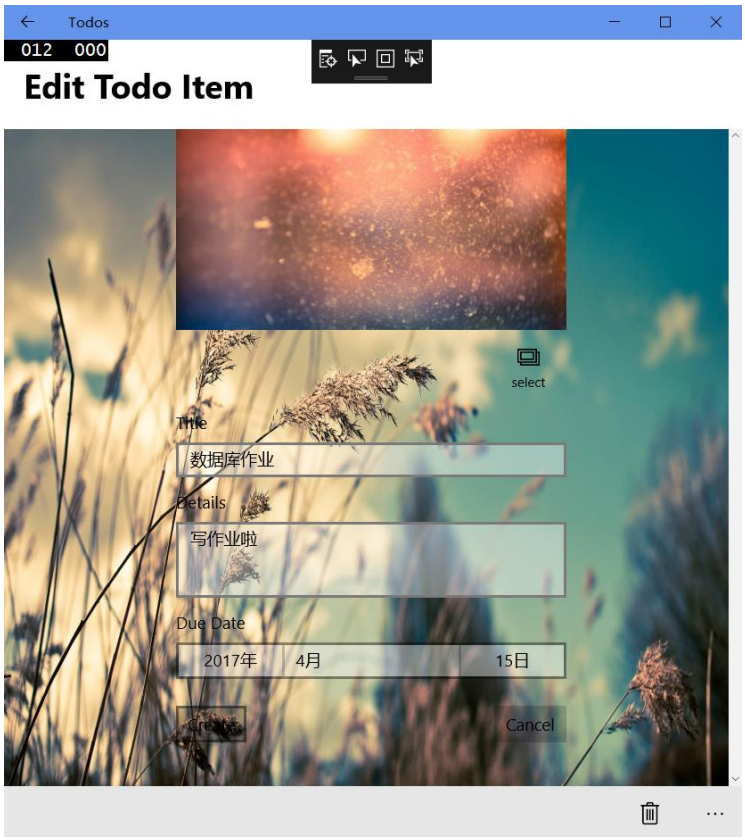
12. 进入 Todos 的创建界面



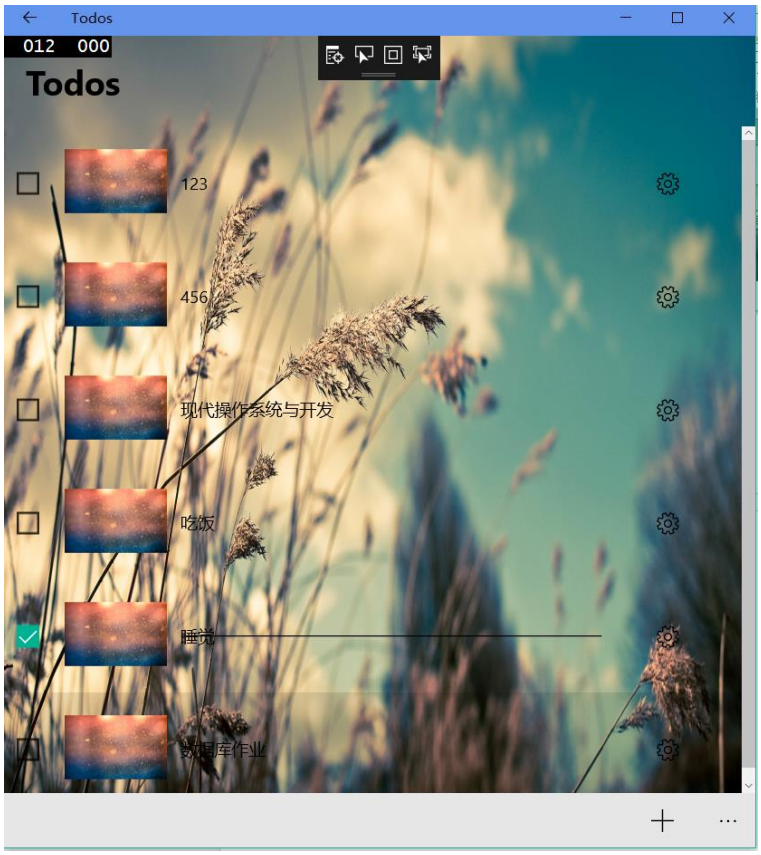
欢迎来到Todos的创建界面！

关闭(C)

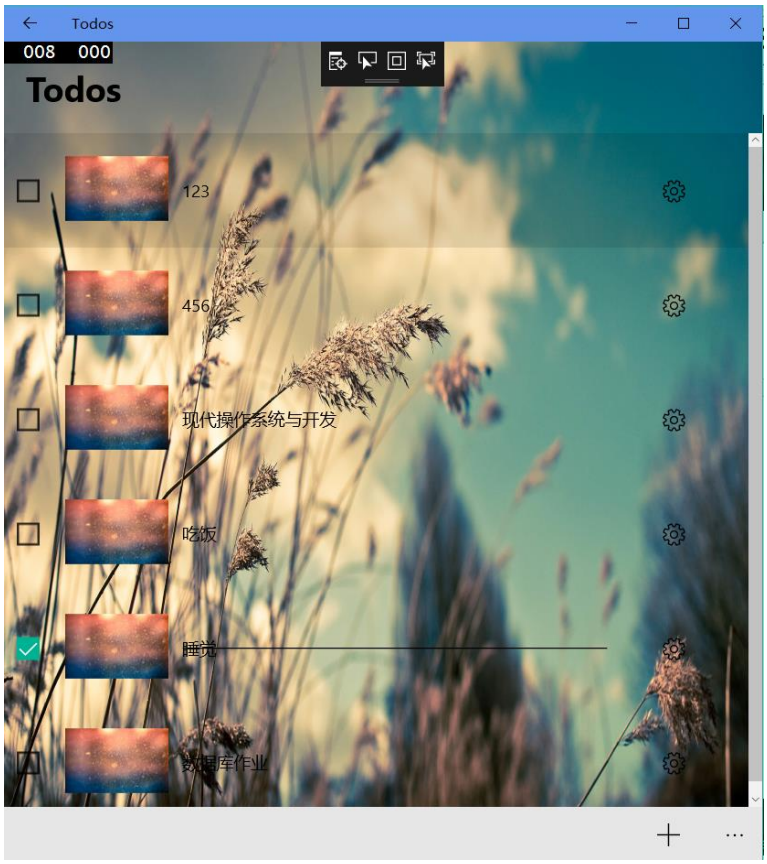
13. 创建如下所示的新的待办事项



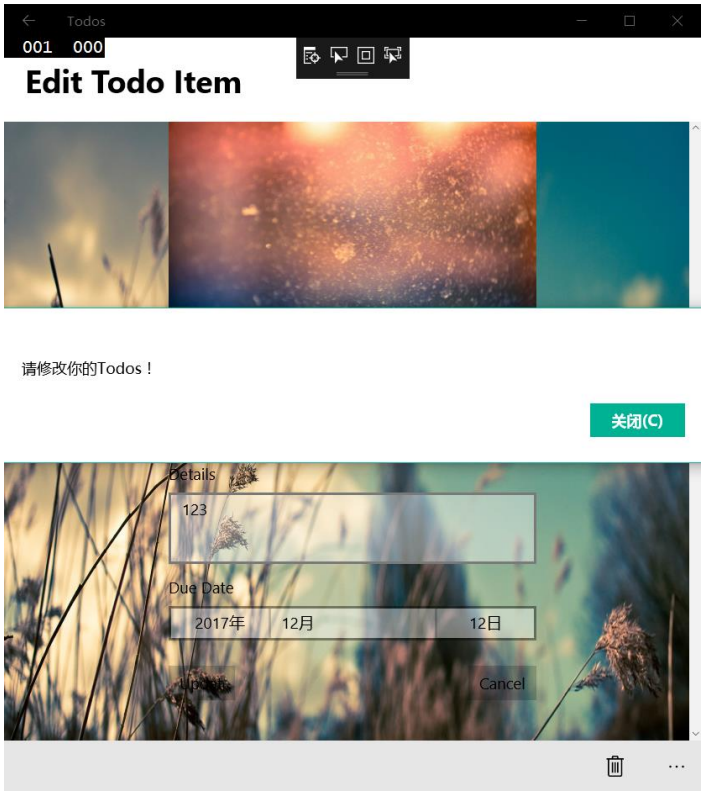
14. 点击“Create”键后跳转到 MainPage，ListView 的最后一行显示了刚才创建的待办事项



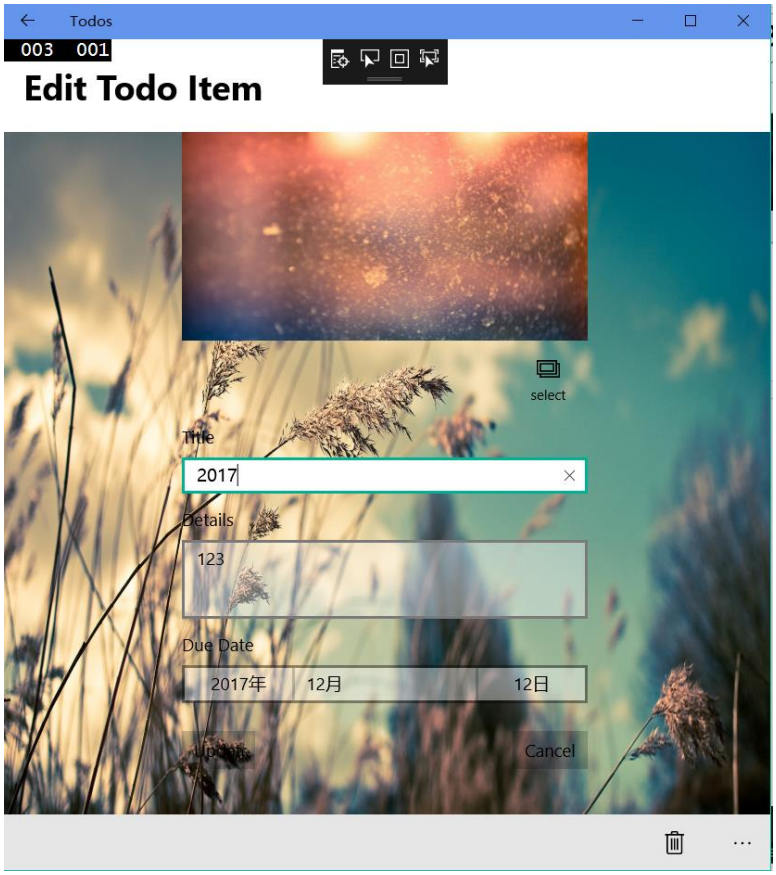
15. 点击“123”这条待办事项，进入 NewPage 对它经行修改



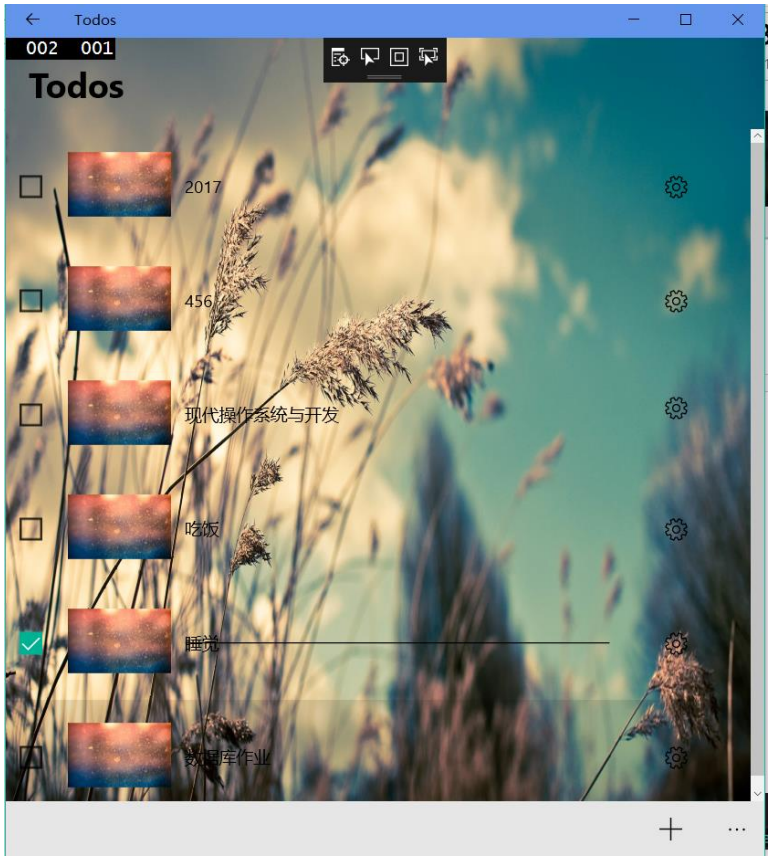
16. 进入 Todos 的修改界面



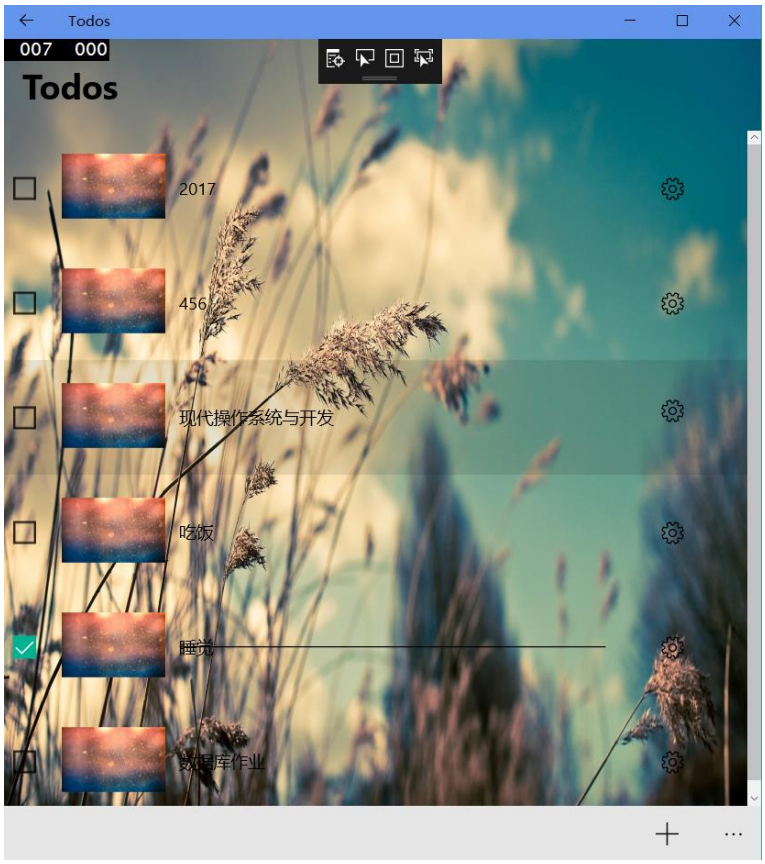
17. 将 Title 设置为 “2017”



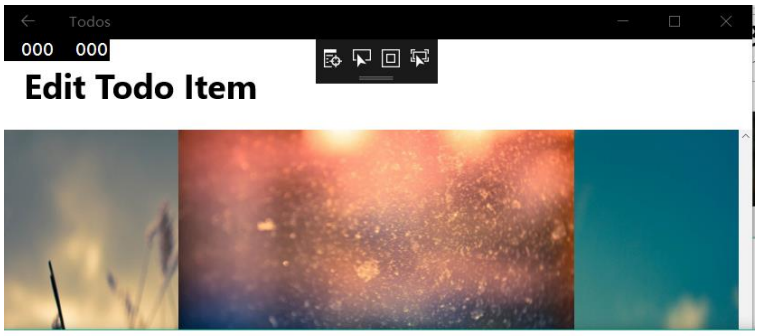
18. 点击 “Create”，跳转到 MainPage 界面



19. 点击“现代操作系统与开发”这条待办事项（创建的时候少打了“应用”两个字……）



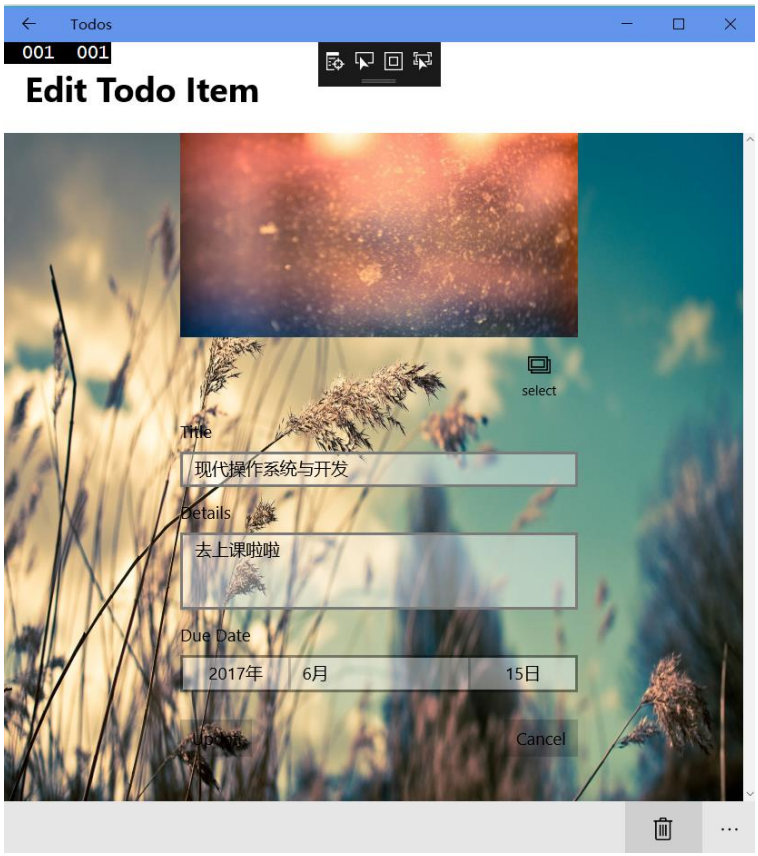
20. 进入 Todos 修改界面



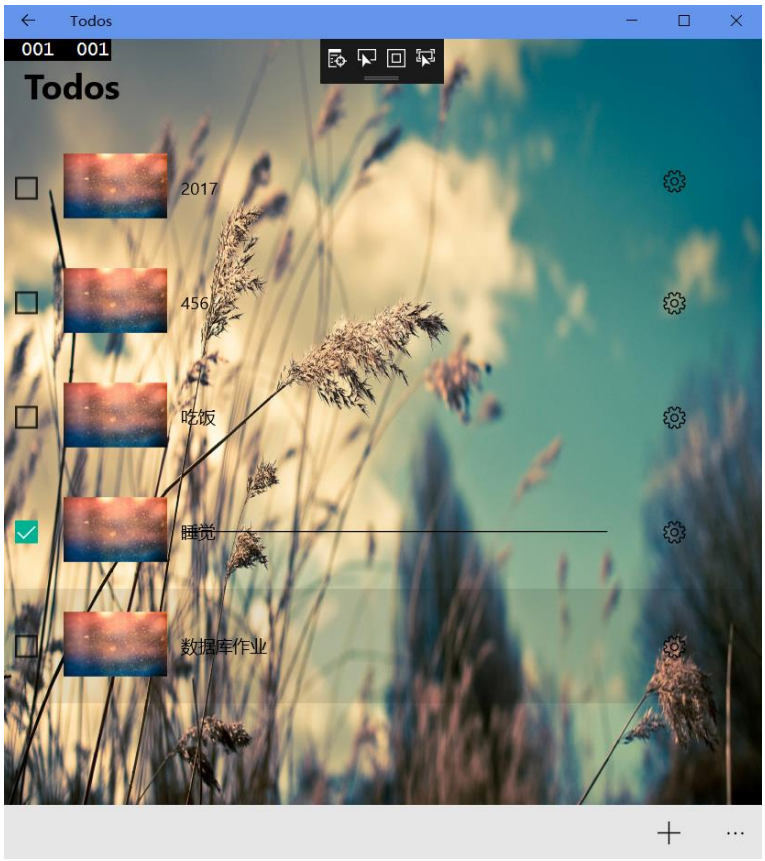
请修改你的Todos！

关闭(C)

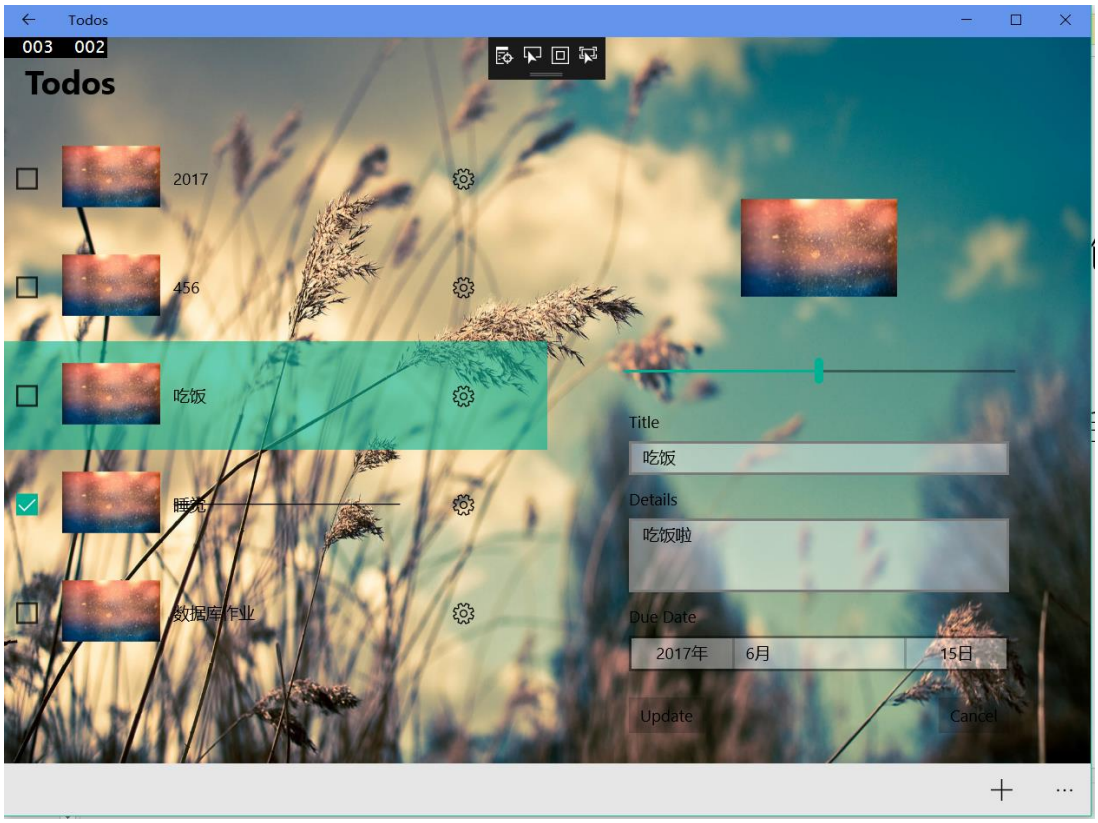
21.点击右下角的"删除"键（有点像垃圾桶），对 Item 进行删除



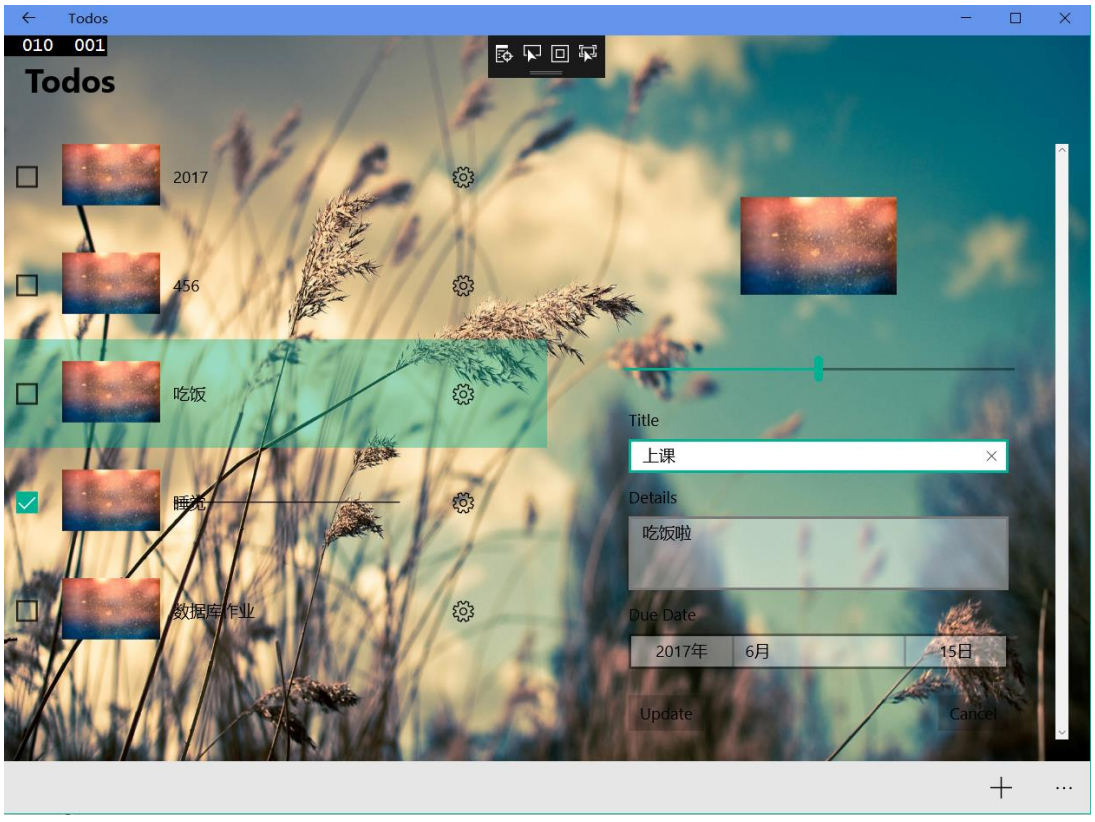
22.删除后自动回到 MainPage， Item 已经删除



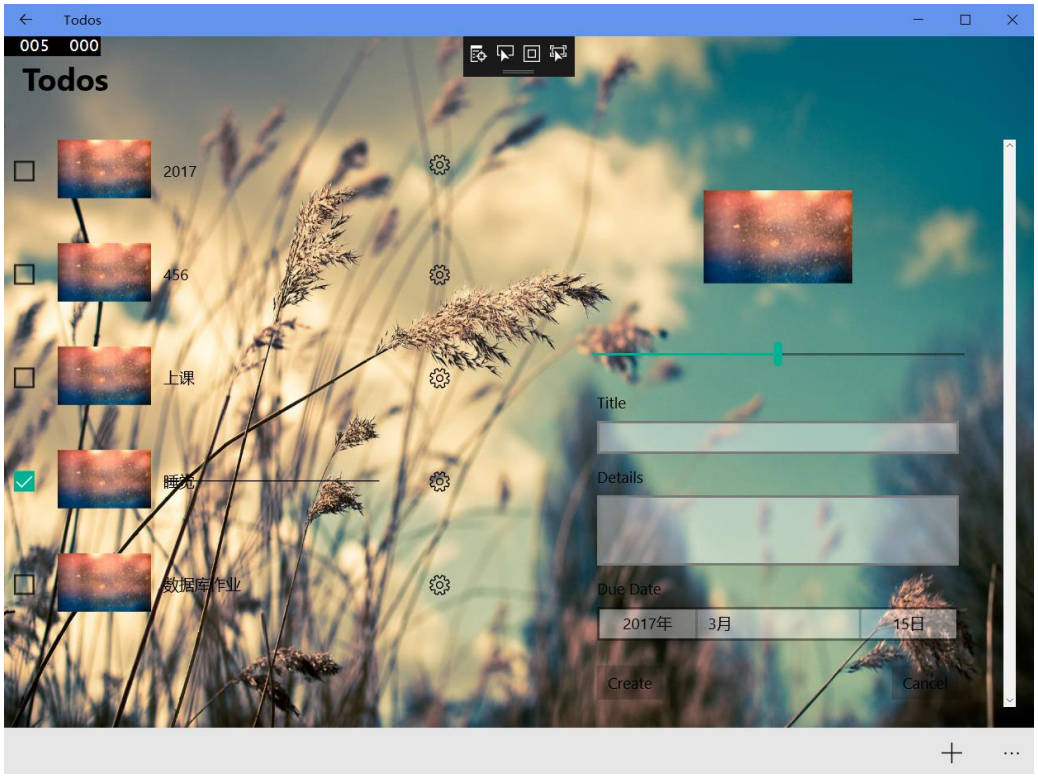
23. 在宽屏的情况下随便点击一个项目，右边就会显示出改项目的详细信息（这里点击的是“吃饭”）



24. 对“吃饭”这个代办事项的 Title 进行修改，修改为“上课”



25.点击“Update”键后左边的 ListView 刷新，显示最新的待办事项 列



注：上周完成的功能在此就不做展示。此外宽屏时右边的图片还会随着滑动条的值而变大变小。

四．实验过程遇到的问题

- 我遇到最大的问题就是将 CheckBox 勾选后怎么将 Line 实时地显示出来，这个问题在上周的作业中很容易解决，就是为 CheckBox 和 Line 分别命名，然后在后台代码写一个事件----“当勾选 CheckBox 时，Line 的 Opacity 属性发生变化”。但是在这周的项目中不能这样做，因为 ListView 里面的项目是用模板做出来的，只能通过数据绑定实现相应的功能。于是自己在网上看了很多有关数据绑定的知识，最后想到将 CheckBox 的 IsChecked 属性与 TotoItem 的 completed 绑定起来，并且是双向绑定，这样当 CheckBox 勾选时就会一同将对应的 TodoItem 的 completed 改变，而 Line 的 Opacity 属性要与 TodoItem 的 completedLine 绑定（当 completed 为 false 时 completedLine 返回 0，当 completed 为 true 时 completedLine 返回 1，这样就不用写 converter 了），但是这时候又遇到了新的问题，就是当我勾选 CheckBox 时，虽然对应的 TodoItem 的 completed 属性改变了，但是没有即时通知 Line，这是我想到用 INotifyPropertyChanged 接口，但是看了很多教程后，自己试验了很多遍发现直接刷新界面来得更加简单直接，于是我在 CheckBox 得属性中加了 Click，并且只要点击 CheckBox 界面就会刷新一次，这样 Line 就能即时根据 TodoItem 变换 Opacity 的值了。
- 此外在自适性界面功能的完成也遇到了一些问题，但是在微软开发人员官网查了很多资料后最终还是写了出来。我新建立了一个新的 ListView，这个 ListView 没有图片，只有 CheckBox、Line 和 TextBlock。代码如下

```
<ListView IsItemClickEnabled="True" x:Name="ToDoListViewBegin" ItemClick="TodoItem_ItemClicked" ItemsSource="{x:Bind ViewModel.AllItems}">
    <ListView.ItemTemplate>
        <DataTemplate x:DataType="md:TodoItem">
            <Grid Height="100">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="42"/>
                    <ColumnDefinition Width="Auto"/>
                    <ColumnDefinition Width="*"/>
                    <ColumnDefinition Width="100"/>
                </Grid.ColumnDefinitions>
                <CheckBox Grid.Column="0" VerticalAlignment="Center" Height="32" Width="32"  IsChecked="{Binding completed, Mode=TwoWay}" Click="CheckBox_bit"/>
                <TextBlock Text="{x:Bind title }" Grid.Column="2" VerticalAlignment="Center" Foreground="Black" FontWeight="Normal" FontSize="15" LineHeight="20" TextWrapping="
                <Line Opacity ="{Binding completedLine}" x:Name="Line_Main" Grid.Column="2" Stretch="Fill" Stroke="Black" StrokeThickness="1" X1="1" VerticalAlignment="Center" 
            </Grid>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
```

五．思考与总结

在做题目的过程中学到了很多东西，进一步了解到了自适应界面的概念和数据绑定的概念，也明显感受到这次的作业量比前面两次的作业量要大，希望自己在接下来的课程里学到更多东西。