

现代操作系统应用开发实验报告

学号： 15331046

班级： 软工一班

姓名： 陈志扬

实验名称： HW5-Todos

一．参考资料

请在这里列出对本实验有帮助你所参考的资料或者网站。

参考资料：老师上课所用课件

参考网站：[创建自适应磁贴](#)里面的相关内容（本地磁贴等）；[借助本地通知使用通知队列](#)；[发送磁贴更新](#)；[共享数据](#)等

二．实验步骤

请在这里简要写下你的实验过程。

1. 在本次实验中，我在 HW3 的作业上来修改，即是说 HW3 作为 HW5 的 demo。
2. 由于一开始我觉得磁贴的内容可能得花费一些时间来学习，所以我先做了共享数据这一块。共享数据因为不要求共享动态图片，所以也比较简单，只需给定图片路径然后利用一些函数接口即可实现。具体实现过程在第四部分详述。

注册事件：

```
DataTransferManager.GetForCurrentView().DataRequested += OnShareDataRequested;
```

处理 share 后删除事件：

```
protected override void OnNavigatedFrom(NavigationEventArgs e)
{
    DataTransferManager.GetForCurrentView().DataRequested -= OnShareDataRequested;
}
```

3. 然后是磁贴的实现。根据 TA 和相关网站的指导，首先安装了 Notifications Visualize，根据里面的模板学习 xml，然后自己在项目文件根目录中新建文件 tile.xml，编写相关代码。
4. 接着，在 MainPage.xaml.cs 和 NewPage.xaml.cs 中编写 xml 加载函数 updateTile，可用于解析 xml，实现磁贴本地通知。
5. 最后，完善相关功能。例如，创建新的 Todo item，磁贴能够更新，并循环显示。

三 . 实验结果截图

请在这里把实验所得的运行结果截图。

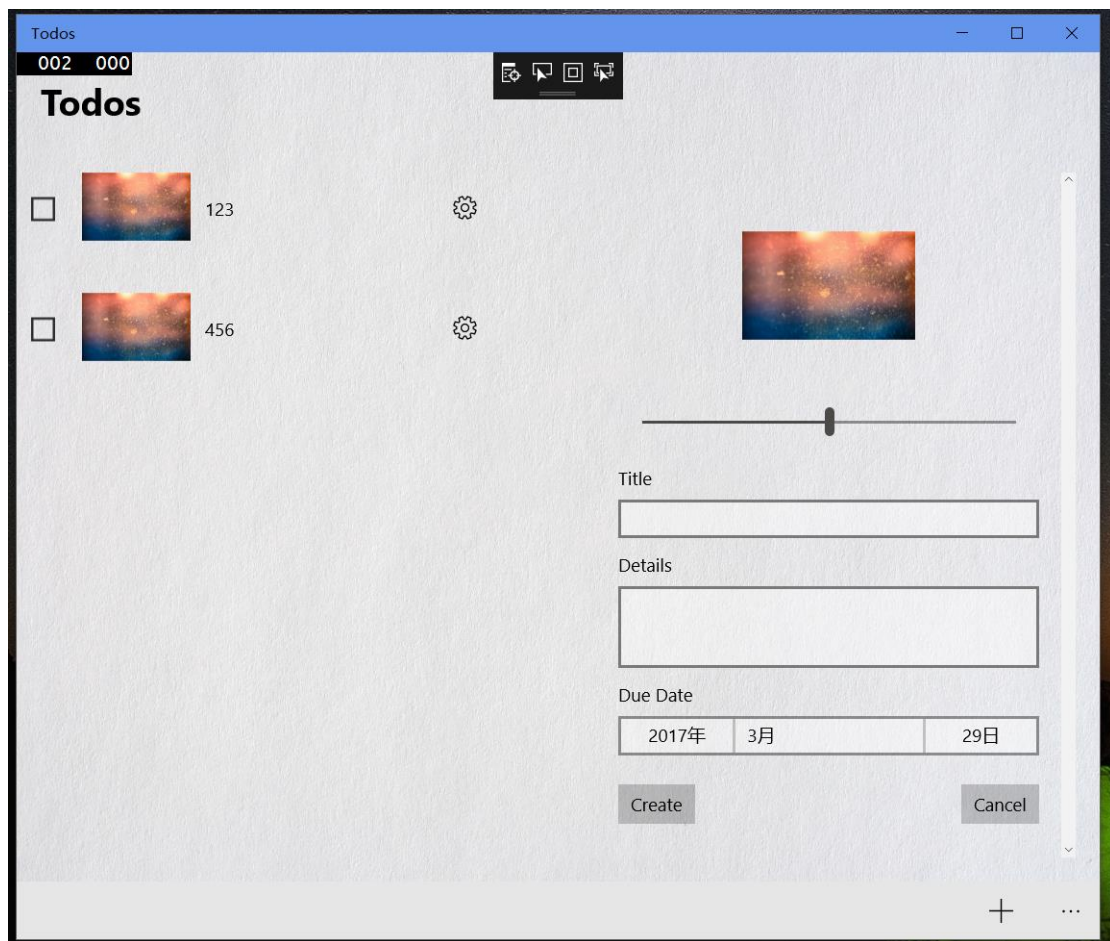
这是我添加的 StoreLogo，启动界面

Todos

- □ X



Todos
最近添加



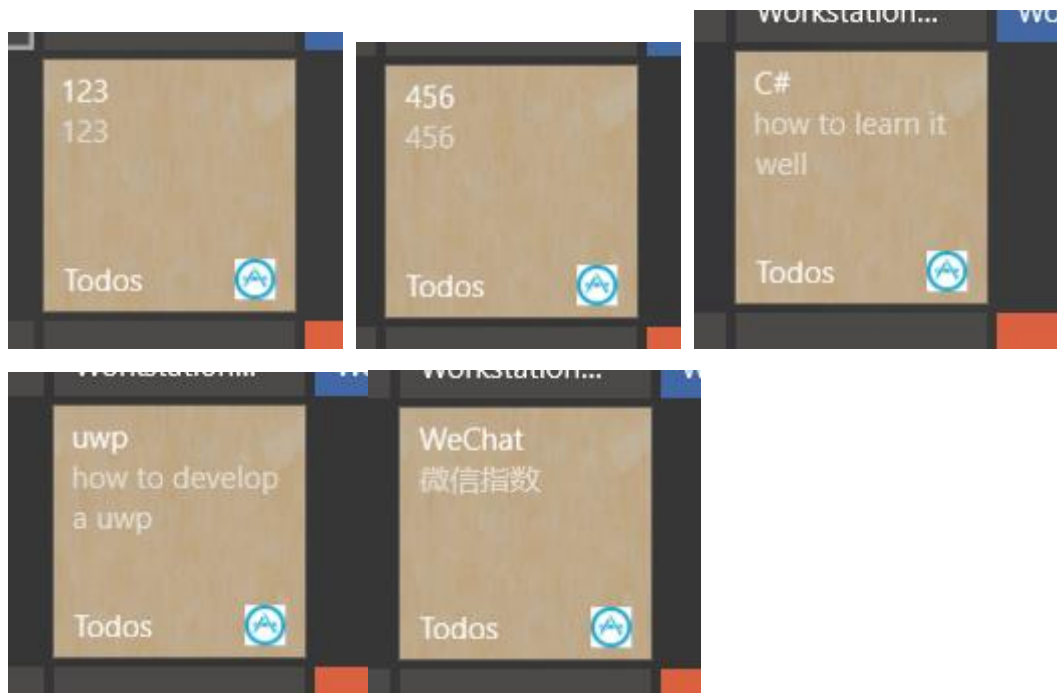
固定到开始屏幕，可看到如下的磁贴：



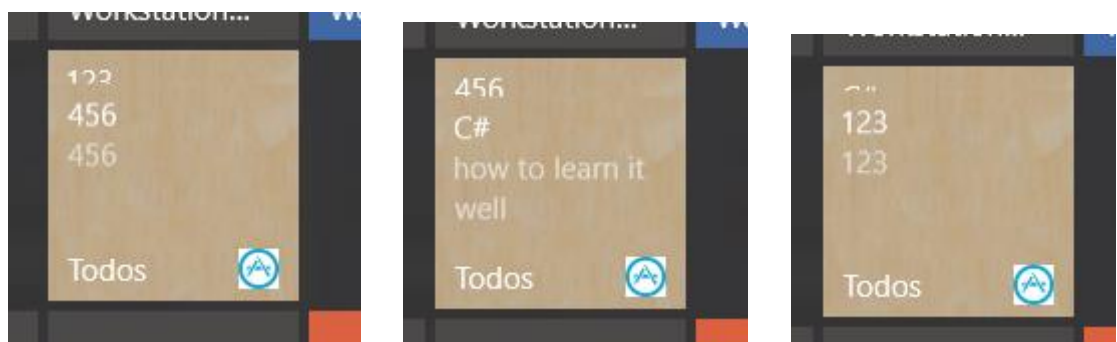
默认的只有小、中、宽三种大小，为其添加了“大”的尺寸大小

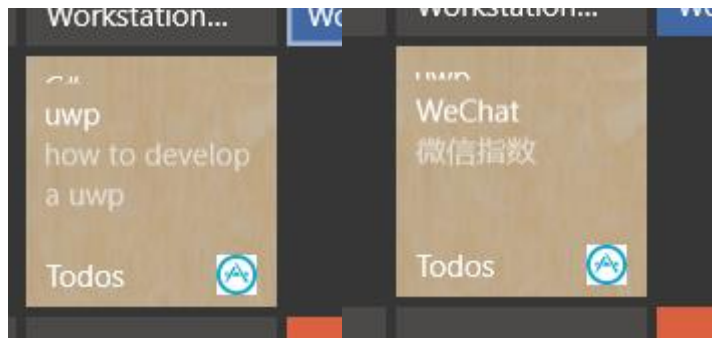


下面是创建的五個 **Todo Items**，按以下順序排列



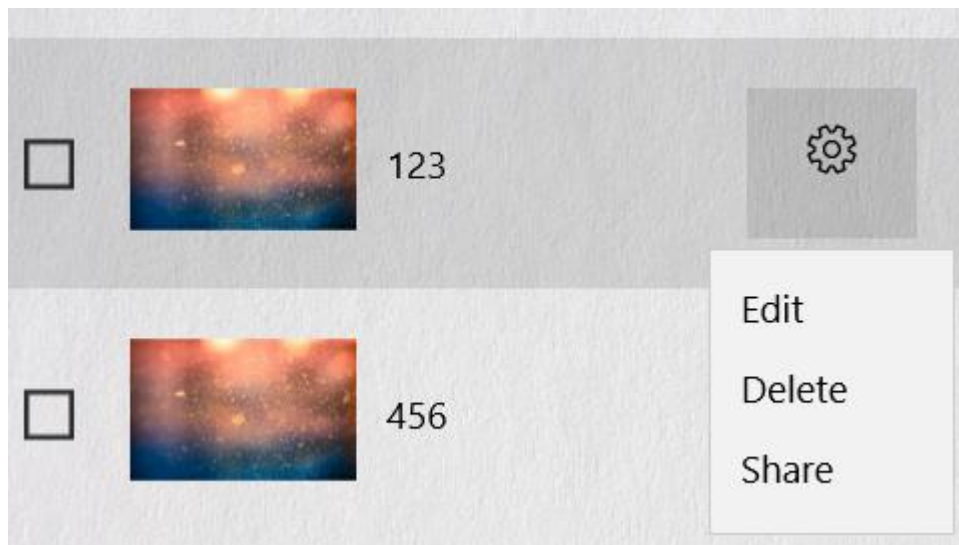
这里是体现轮询过程，五个 **Items** 按队列顺序依次展现在磁贴中：（截图比较难弄，但至少还是可以看出来变化状态的）



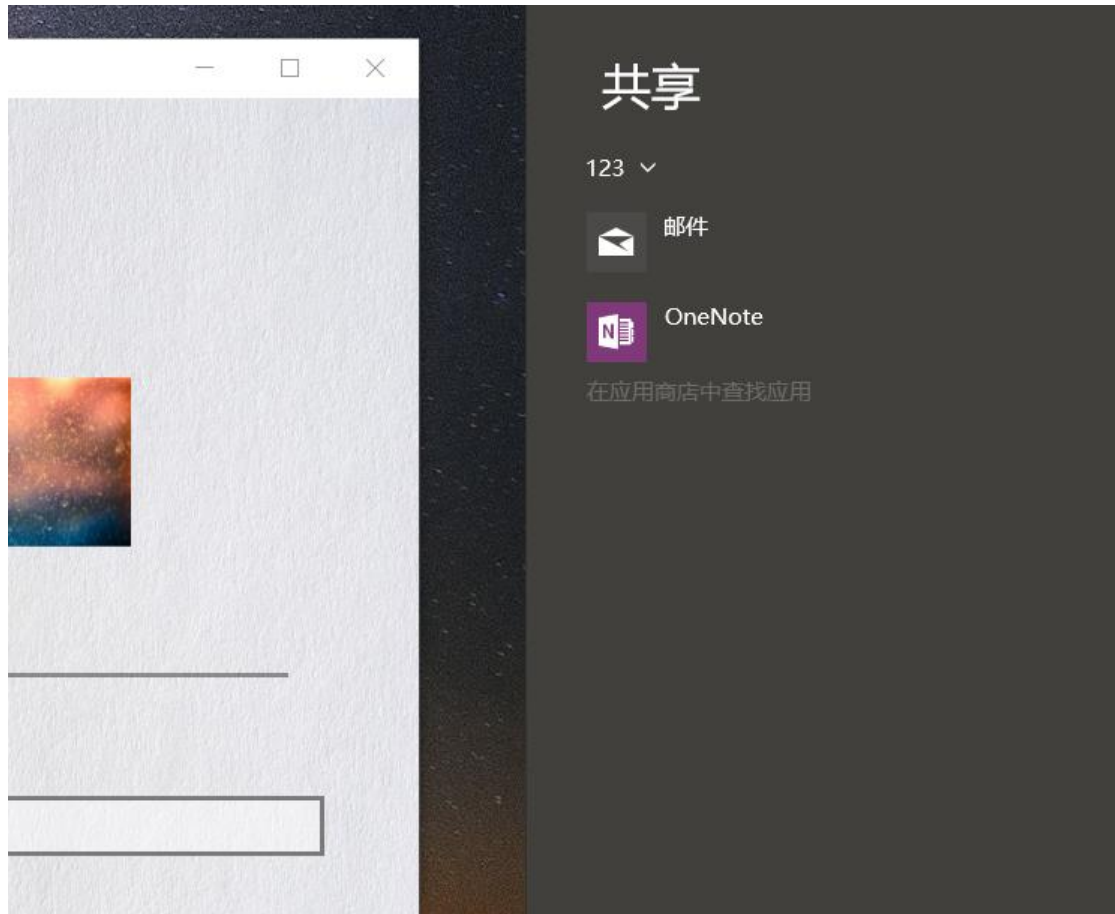


下面是共享数据的截图：

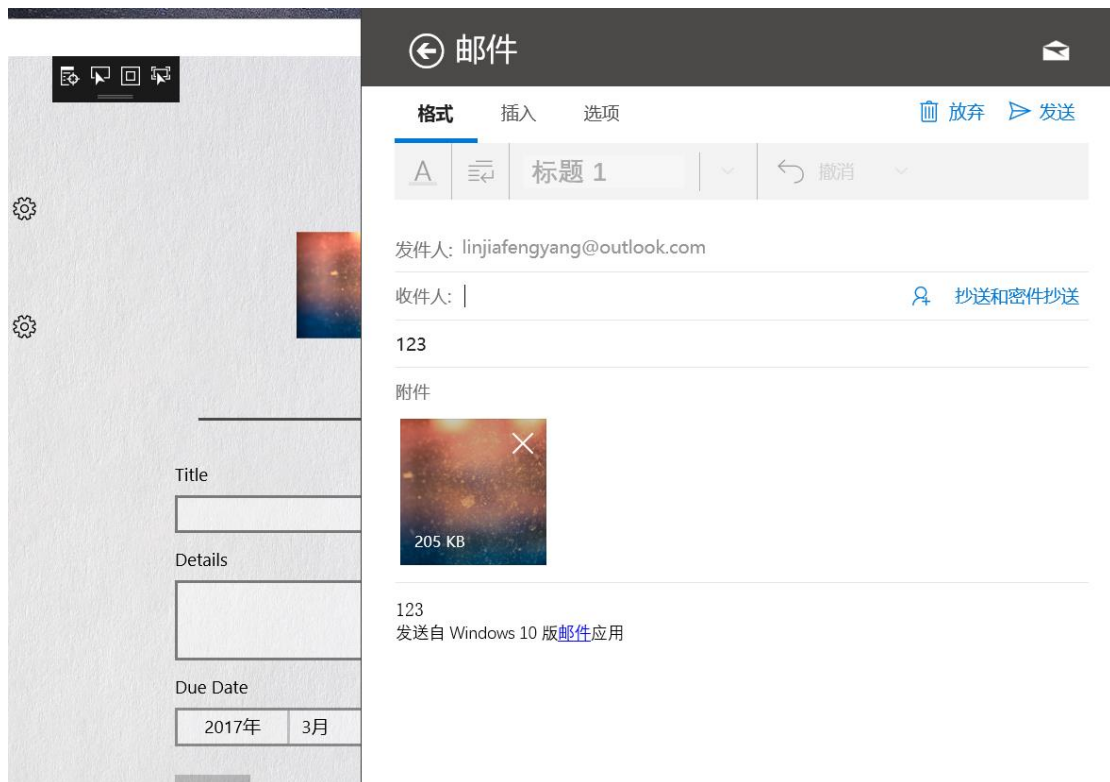
点击 Share：



在屏幕右侧弹出共享目标：由此也可见共享标题正确。

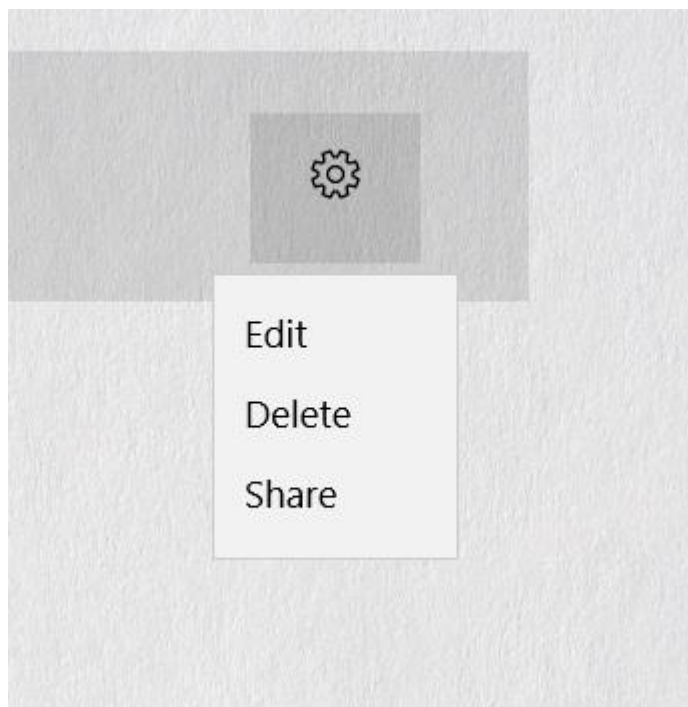


点击邮件应用作为共享目标，可见标题、文本内容、图片均有，说明已基本实现所要求的功能。

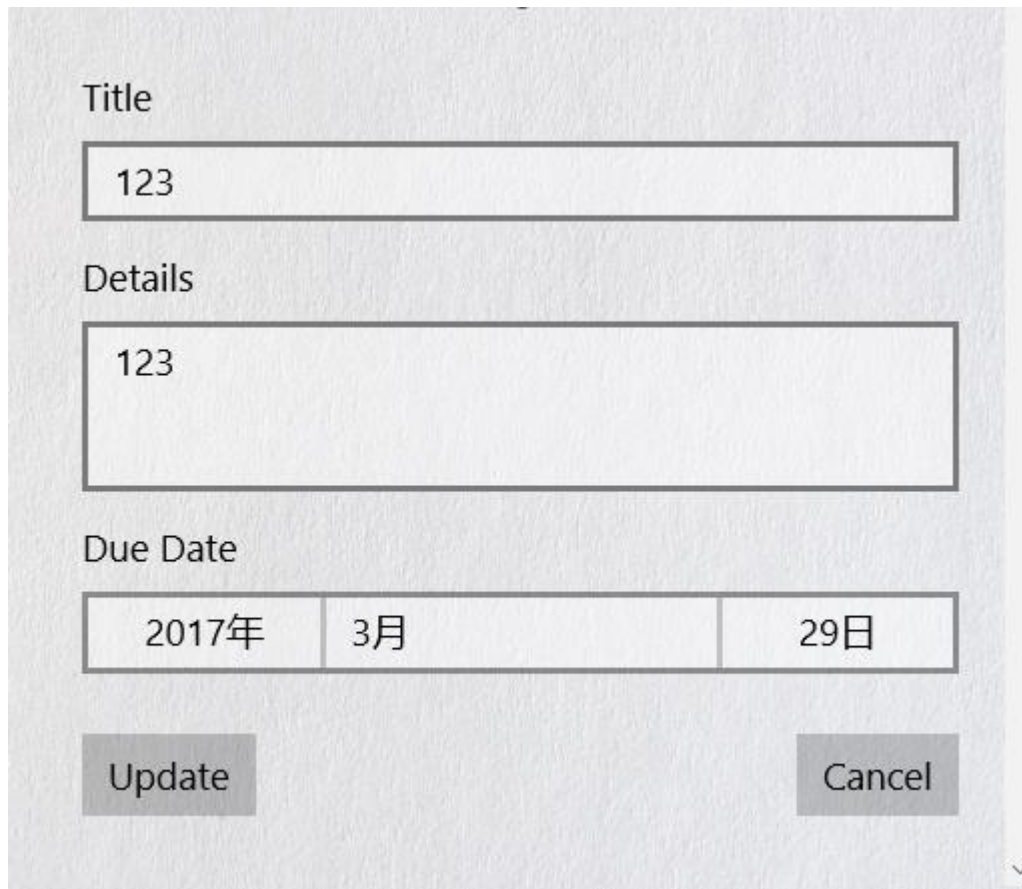


下面是我所做的补充功能，点击 **Setting** 按钮，可实现 **Edit**，**Delete** 功能，其中 **Edit** 对应 **Update** 功能(`UpdateButton_Clicked`)，**Delete** 对应直接删除功能(`DeleteButton_Clicked`)

点击 **Edit**:



宽屏下直接在右边修改，窄屏下跳转到 **NewPage** 修改，实现过程和 **HW3** 类似。



The image shows a light gray modal form for editing a todo item. It contains three input fields: 'Title' with the value '123', 'Details' with the value '123', and 'Due Date' with a date picker showing '2017年', '3月', and '29日'. At the bottom are two buttons: 'Update' and 'Cancel'.

Title

123

Details

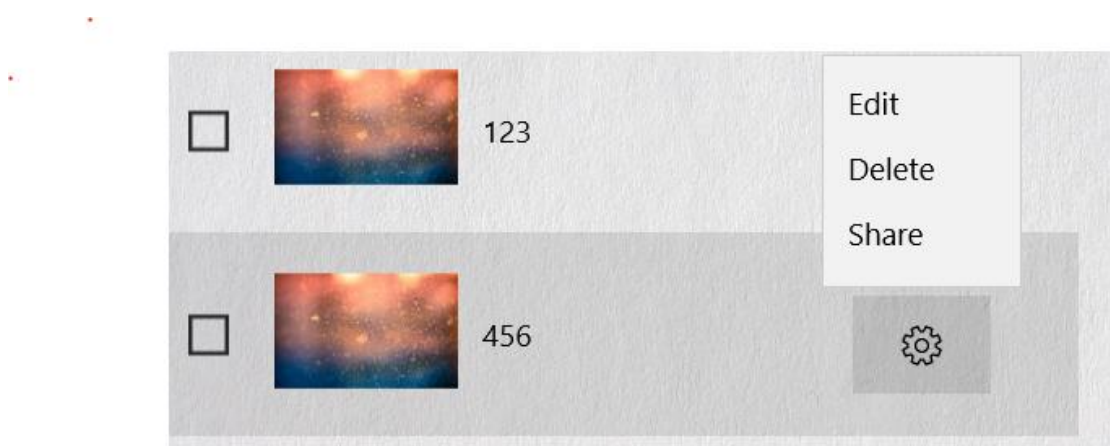
123

Due Date

2017年 3月 29日

Update Cancel

点击 **Delete**：直接删除掉对应的 **Todo Item**。





四．实验过程遇到的问题

请在这里写下你在实验过程中遇到的问题以及解决方案。

- ① 点击 Share，不知道如何得到该 Todo Item 的内容，以为之前的 Name.text 依然可以使用，但其实是不行的。其实需要获取 FrameworkElement 参与数据绑定时的数据上下文，将其转化为 ListViewItem。下面的代码可解决这个问题。

```
// 另一种方法
//dynamic ori = e.OriginalSource;
//ViewModel.SelectedItem = (Models.TODOItem)ori.DataContext;
var dataContext = (sender as FrameworkElement).DataContext;
var item = ToDoListView.ContainerFromItem(dataContext) as ListViewItem;
ViewModel.SelectedItem = (Models.TODOItem)(item.Content);
```

- ② 将本地图片作为共享数据时读取图片无效。可以从文件类 StorageFile 中的一个方法 GetFileFromApplicationUriAsync(Uri uri)获得文件流，这里

我漏了 ms-appx 导致一直出错。然后以输入和输出流的形式提供对数据流的随机访问，最后调用 `DataPackage.SetBitmap` 方法即可。具体代码如下：

```
private async void OnShareDataRequested(DataTransferManager sender, DataRequestedEventArgs args)
{
    DataRequest request = args.Request;
    request.Data.Properties.Title = ViewModel.SelectedItem.title;
    //request.Data.Properties.Description = Details.Text;
    request.Data.SetText(ViewModel.SelectedItem.description);

    var picture = await StorageFile.GetFileFromApplicationUriAsync(new Uri("ms-appx:///Assets/background.jpg"));
    RandomAccessStreamReference imageStreamRef = RandomAccessStreamReference.CreateFromFile(picture);
    request.Data.SetBitmap(imageStreamRef);
}
```

- ③ 动态磁贴的加载问题。找了很多文档，知道了 `LoadXml` 但是一直无效。仔细看了 `LoadXml` 的参数，原来是 `string` 类型，改成 `LoadXml(File.ReadAllText("tile.xml"))` 即可。
- ④ 接下来是使用通知队列。这个参考文档给的很详细，主要是在这个实验我写了个 `foreach` 的循环 将每个 `Todo Item` 的 `title`、`description` 数据逐一赋值到 `tile.xml` 文件中。具体代码如下：

```

private void updateTile()
{
    TileUpdateManager.CreateTileUpdaterForApplication().EnableNotificationQueue(true);
    TileUpdateManager.CreateTileUpdaterForApplication().Clear();
    XmlDocument xml = new XmlDocument();
    xml.LoadXml(File.ReadAllText("tile.xml"));
    foreach (var todo in ViewModel.AllItems)
    {
        XmlNodeList text = xml.GetElementsByTagName("text");
        for (int i = 0; i < text.Count; i++)
        {
            text[i].InnerText = todo.title;
            //i++;
            text[i].InnerText = todo.description;
        }
        TileNotification notification = new TileNotification(xml);
        TileUpdateManager.CreateTileUpdaterForApplication().Update(notification);
    }
}

```

⑤ 说说针对 Setting 按钮补充的一些功能。主要是点击 Edit 对应 Update ,可以实现更新 ,点击 Delete 可以直接删除该 Item ,然后再调用 updateTile 函数实现磁贴更新。具体代码如下：

```

// 点击Setting中的Delete, 直接删除该Item
private void DeleteSetting_Clicked(object sender, RoutedEventArgs e)
{
    var dataContext = (sender as FrameworkElement).DataContext;
    var item = ToDoListView.ContainerFromItem(dataContext) as ListViewItem;
    ViewModel.SelectedItem = (Models.TODOItem) (item.Content);

    DeleteButton_Clicked(sender, e);
    updateTile();
}
// 点击设置 即点击setting

```

五 . 思考与总结

请在这里写下你本次试验的心得体会以及所思所想。

这次实验主要是学习自适应磁贴和共享数据，主要学到下面几个函数接口的操作等等。

```

TileUpdateManager.CreateTileUpdaterForApplication().EnableNotificationQueue(true);

```

```

TileNotification notification = new TileNotification(xml);

```

```
TileUpdateManager.CreateTileUpdaterForApplication().Update(notification);

XMLDocument.LoadXml(File.ReadAllText("tile.xml"));

DataManager.GetForCurrentView().DataRequested

DataManager.ShowShareUI();
```

TA 说这次比较简单，就不给 demo 了，我只想说如果只是简单实现基本功能，确实不是很难，但是考虑到大磁贴显示多条 item 我还是没能解决问题，再者，如果共享要考虑到图片是绑定的图片，那难度又上升了。对 Setting 按钮的补充功能中，Edit 对应 update 虽然可以修改，但是不知道为什么磁贴更新不了（确切地说应该是时好时坏）。学到现在，现操的代码真的感觉迷，接下来还有期中项目的 uwp，真的要好好努力，付出更多精力来学习。

1. 实验报告提交格式为 pdf。
2. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。