**现代操作系统应用开发实验报告**

——刘亚辉 16340157

UWP第二周实验报告

1. 参考资料
2. <https://blogs.msdn.microsoft.com/tiles_and_toasts/2015/06/30/adaptive-tile-templates-schema-and-documentation/>（动态磁贴）
3. <https://docs.microsoft.com/en-us/uwp/api/windows.storage.applicationdatacompositevalue> (composite用法)
4. <https://blog.csdn.net/lindexi_gd/article/details/51242913> (XDocument生成动态磁贴、Toast)
5. <https://www.c-sharpcorner.com/article/live-tiles-in-universal-windows-apps/> (动态磁贴)
6. <https://github.com/oysteinkrog/SQLite.Net-PCL> (SQLite数据库)
7. <https://www.cnblogs.com/yanxiaodi/p/4941312.html> (SQLite.Net-PCL数据库使用)
8. <http://www.cnblogs.com/ms-uap/p/4798269.html> (SQLitePCL)
9. <http://www.cnblogs.com/h82258652/p/4802076.html> (SQLite.Net-PCL)
10. <http://www.sqliteexpert.com/> (SQLiteExpert可视化工具)
11. PPT、课件
12. 实验步骤

**·**需求及流程：

1. week-4：实现挂起并关闭时保存当前的状态

当应用启动时会调用onLaunched方法，并向该方法传递LaunchActivatedEventArgs参数，从LaunchActivatedEventArgs.PreviousExecutionState获取应用之前的状态，这将返回ApplicationExecutionState对象。

当应用暂停时，系统会尝试将应用及其数据保留在内存中。 但是，如果系统没有资源将应用保留在内存中，它将终止应用。 应用不会收到它们被终止的通知，所以只能**将应用数据保存在 OnSuspension 事件处理程序中**，或者**在 EnteredBackground 处理程序中以异步方式保存。**

当应用确定它在终止后被激活时，它应该加载它保存的应用程序数据，以使应用处于与其终止之前相同的状态。 当用户**切换回已终止的暂停应用时**，该应用应该在其 **OnLaunched 方法中还原其应用程序数据。** 当终止应用时系统不会通知应用，因此**在暂停应用之前，应用必须保存其应用程序数据**并**释放独占资源和文件句柄，并且当在终止后又激活应用时还原这些内容。**

1. week-5：

实现动态磁贴以及共享条目的功能。

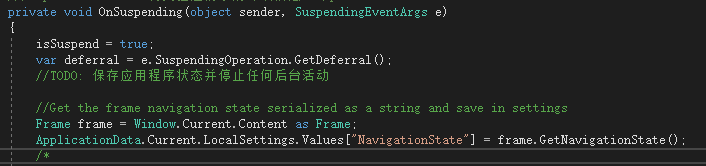
1. week-6：

利用SQLite数据库实现数据持久化。

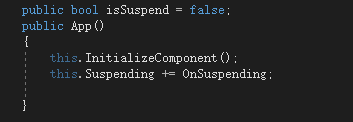
**·**详细实现过程：

1. 实现挂起关闭并恢复数据的功能：

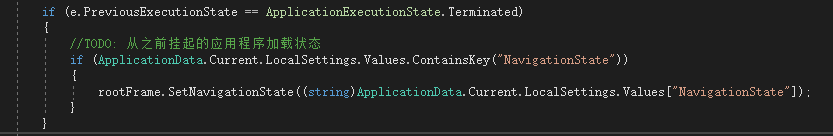
首先需要在App.xaml.cs中构建onSuspending函数，表示挂起过程中需要进程的一些操作，保存当前的状态：



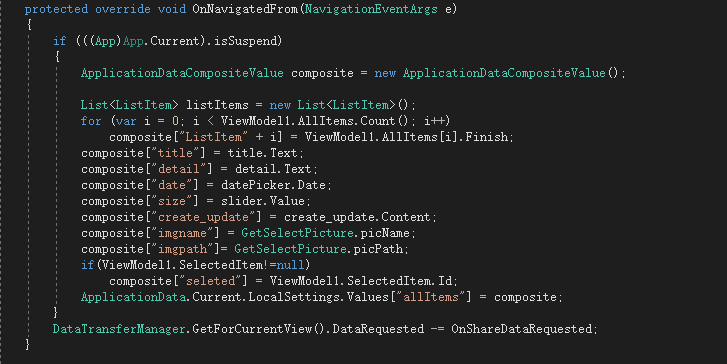
并进行事件委托：



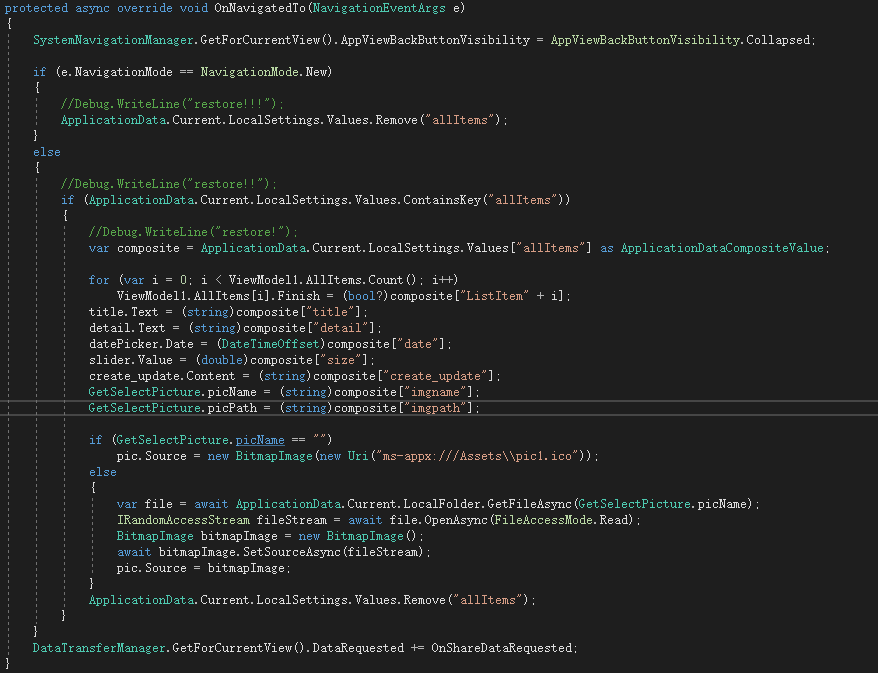
并在重新打开，OnLaunched函数中利用NavigationState实现加载到之前的几面：



之后利用MainPage和NewPage中的OnNavigatedFrom和OnNavigatedTo两个函数，来进行保存数据与恢复数据：



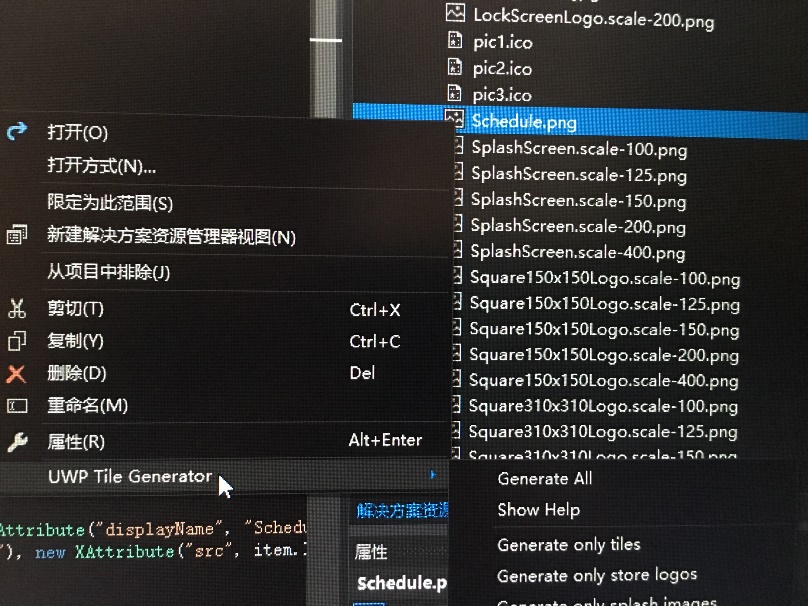
上图是MainPage中的挂起时保存数据的函数，通过ApplicationDataCompositeValue对象实例和键值对的方法来保存数据。其中图片的保存只是保存了path和name。（A**pplicationDataCompositeValue**：一组必须按原子方式序列化和反序列化的相关应用设置。 使用**复合设置**可轻松处理相互依赖的设置的原子更新。 系统会在并发访问和漫游时确保复合设置的完整性。 复合设置针对少量数据进行了优化，如果将它们用于大型数据集，性能可能很差。在此只有少量数据并未考虑性能情况。）



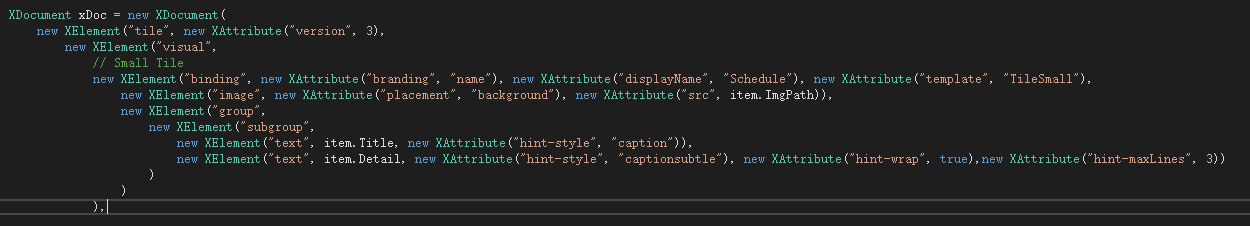
上图是MainPage中的恢复数据的函数，在该函数中，判断是否是挂起后再次打开，然后给之前界面上的元素进行赋值，图片利用ApplicationData.Current.LocalFolder.GetFileAsync函数来得到。在读取或编写本地应用数据前，必须**检索本地应用数据存储。** 若要检索本地应用数据存储，需要使用 **ApplicationData.Current.LocalSettings.Value[KEY] 方法**获取应用**作为 ApplicationDataContainer 对象（在此处为ApplicationDataCompositeValue对象）的本地设置。 使用 ApplicationData.Current.LocalFolder 属性**可以获取**StorageFolder 对象中的文件。**使用**ApplicationData.Current.LocalCacheFolder**属性来获取保存不包括在备份和还原中的文件的本地应用数据存储中的文件夹。

NewPage中处理方法也是如此。

1. 动态磁贴及共享条目写法：

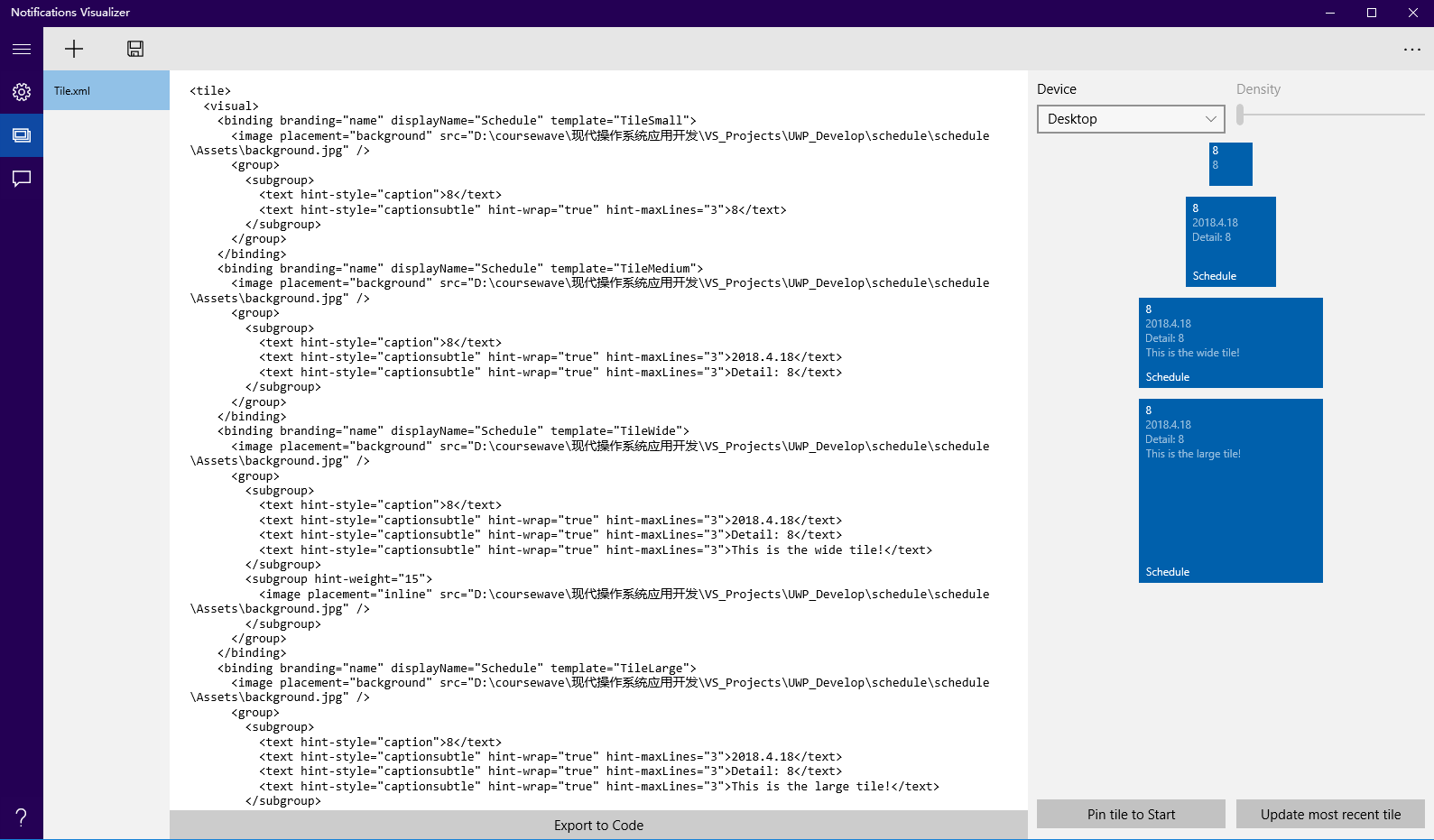
首先，利用下图UWPTileGenerator这款插件来生成所需的各种磁贴的图片，效果如下：  
  
 选择图片之后点击Generate All即可生成所需的各种图片。

之后开始书写生成动态磁贴的代码，这里我使用XDocument来实现xml的生成：

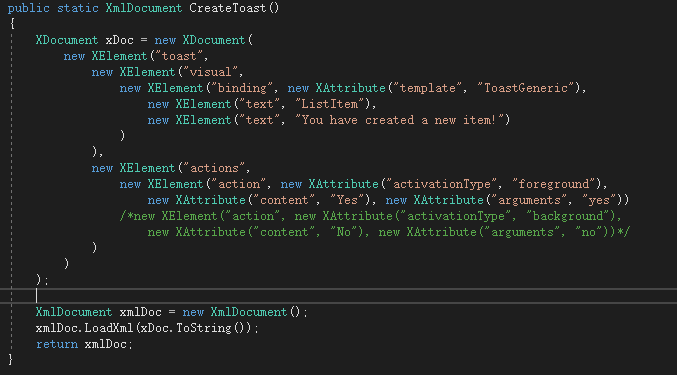


以上代码实现的是小磁贴的样式设计，通过XAttribute，XElement等嵌套使用，即可像xml文件那样利用C#实现后台动态磁贴的生成。其中visual表示可视化的元素设计，binding表示绑定该磁贴类型（TileSmall, TileMidium,TileWide或者TileLarge）；branding表示该磁贴的现实的标识，有三个属性（name, logo, nameAndlogo）；displayName可以向branding那样动态更改磁贴左下角的标识；hint-wrap表示当前文本是否适应当前磁贴宽度自动换行，true表示自动换行，false表示不换行，超过磁贴宽度不显示。

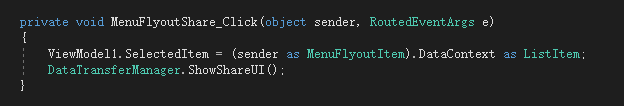
另外，也可以是用可视化工具Notifications Visualizer来帮助实现C#动态磁贴代码的生成：在左侧xml文本框内书写磁贴xml代码，样式效果如右图所示，之后点击Export to Code即可。



之后，实现Toast，方法与Tile相同，也可以利用可视化工具来生成C#代码。



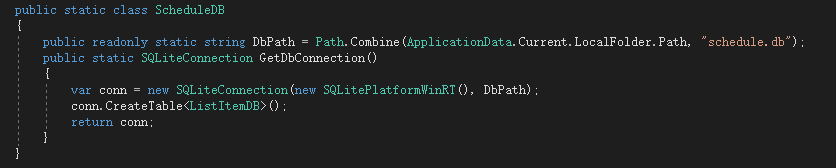
最后，实现条目共享功能：先在MenuFlyout中新加一个元素，作为share点击的按钮，并注册点击事件：



DataTransferManager是一个静态类，用来初始化共享条目，先调用Get GetForCurrentView方法来得到当前活动界面，然后给DataRequested对象增加事件监听器。在离开和进入MainPage界面中分别取消事件委托或者增加事件委托。DataTransferManager.GetForCurrentView().DataRequested -=/+= OnShareDataRequested;

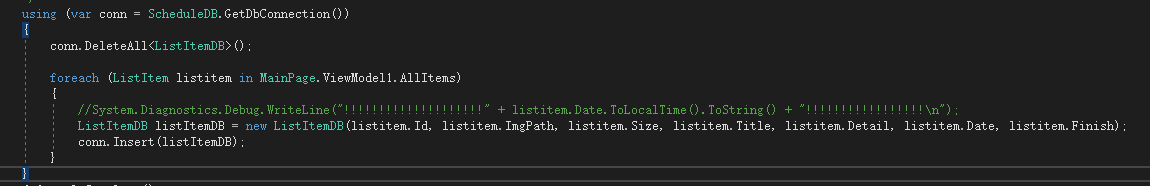
1. SQLite数据库使用：

首先，安装SQLite引用依赖和SQLite.Net-PCL API的包之后。新建ScheduleDB.cs文件，添加ScheduleDB类，在类内部生成新的数据库。

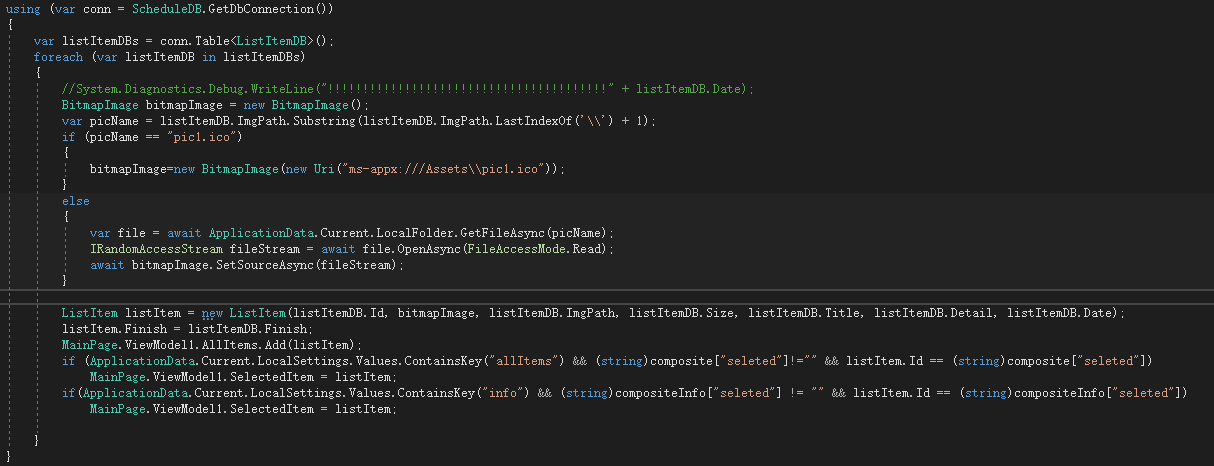


每次使用数据库进行增删改查操作是，调用SQLiteConnection方法，得到一个数据库的连接变量，上述方法，只有在第一次调用时才会生成一个新的表，之后调用便会忽略创建表的操作。在进行数据持久化的设计中，我在打开应用的时候，连接数据库来实现数据恢复，在关闭应用的时候实现数据保存。

数据保存如下：首先清空所有的数据，并且将当前ListItem中的数据在插入一遍：



数据读取如下：using的语法相当于一个块作用域，当前的打括号内的变量在执行完之后即被销毁：



1. 关键步骤截图

请在这里把实验所得的运行结果截图。

1. 亮点与改进（可选）

请在这里写下你完成的bonus和在基本需求之外做的工作。

1. 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. 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。

\*注： 此为报告内容的参考结构，可以用自己喜欢的排版

清理解决方案！

Pdf!

如有抄袭，0分处理！