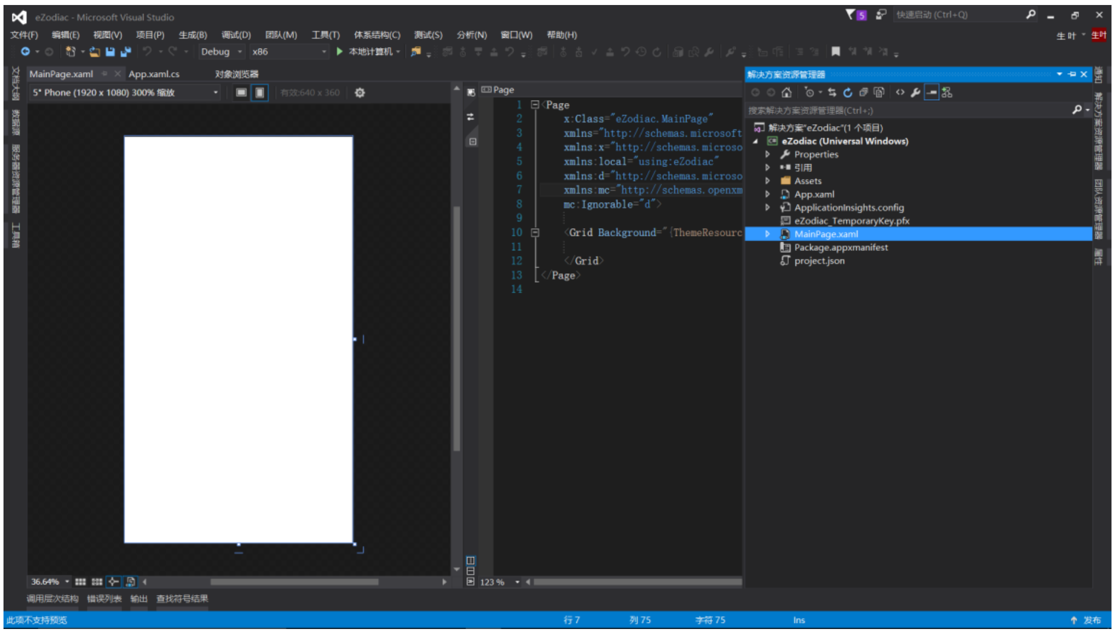
4.19

得知比赛信息，开始计划，准备编写天气查询软件，取名易星运（eZodiac），上网取得 API，新建项目，设计界面。

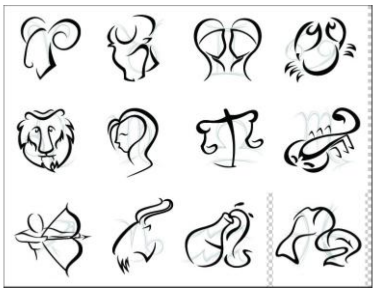


```
<Page
    x:Class="eZodiac.MainPage"
    xmlns="http://schemas.microsoft
    xmlns:x="http://schemas.microso
    xmlns:local="using:eZodiac"
    xmlns:d="http://schemas.microso
    xmlns:mc="http://schemas.openxm
    mc:Ignorable="d">

    <Grid Background="{ThemeResourc

    </Grid>
</Page>
```

eZodiac - Microsoft Visual Studio
MainPage.xaml
App.xaml.cs
5" Phone (1920 x 1080) 300% 缩放
解决方案资源管理器
解决方案"eZodiac"(1 个项目)
eZodiac (Universal Windows)
Properties
引用
Assets
App.xaml
ApplicationInsights.config
eZodiac_TemporaryKey.pfx
MainPage.xaml
Package.appxmanifest
project.json

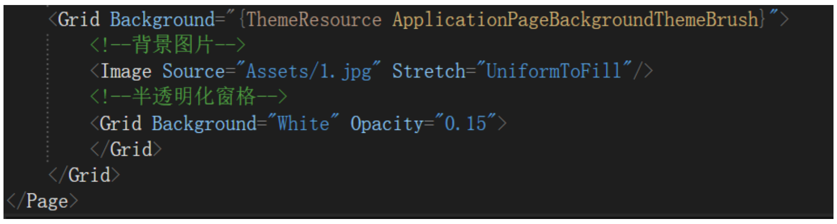
4.20

找图片素材，切图，做各种小图片、图标。



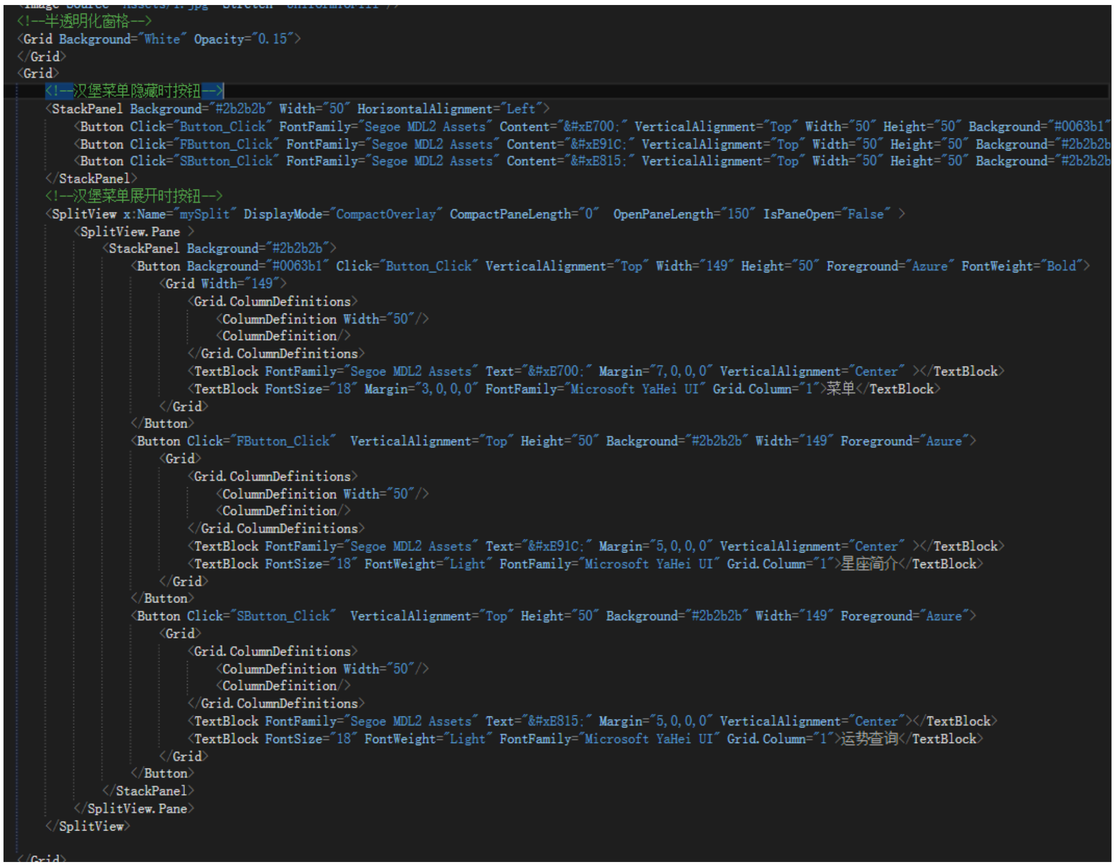
4.21

设置背景图片与半透明化 Grid



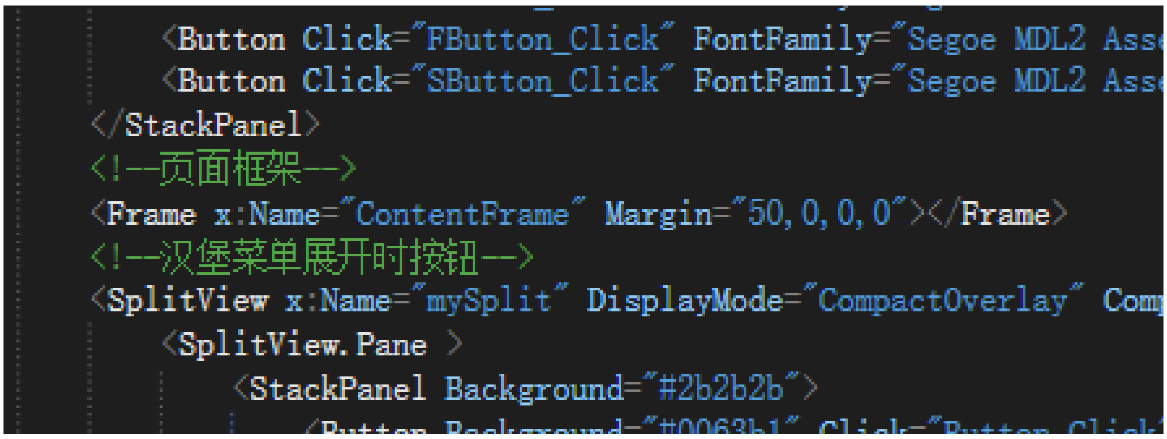
```
    <Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">
        <!--背景图片-->
        <Image Source="Assets/1.jpg" Stretch="UniformToFill"/>
        <!--半透明化窗格-->
        <Grid Background="White" Opacity="0.15">
        </Grid>
    </Grid>
</Page>
```

添加汉堡菜单



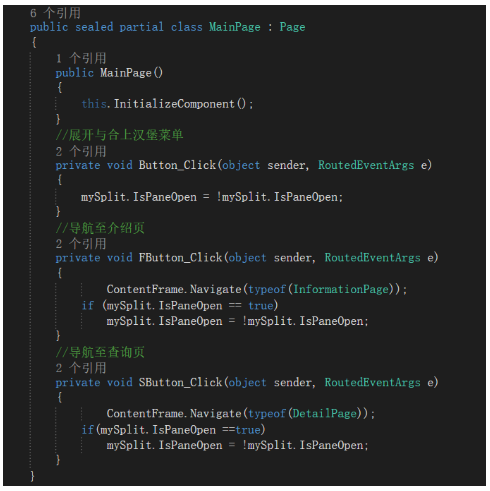
```
<!--半透明化窗格-->
<Grid Background="White" Opacity="0.15">
</Grid>
<Grid>
    <!--汉堡菜单隐藏时按钮-->
    <StackPanel Background="#2b2b2b" Width="50" HorizontalAlignment="Left">
        <Button Click="Button_Click" FontFamily="Segoe MDL2 Assets" Content="&#xE700;" VerticalAlignment="Top" Width="50" Height="50" Background="#0063b1"
        <Button Click="FButton_Click" FontFamily="Segoe MDL2 Assets" Content="&#xE91C;" VerticalAlignment="Top" Width="50" Height="50" Background="#2b2b2b
        <Button Click="SButton_Click" FontFamily="Segoe MDL2 Assets" Content="&#xE815;" VerticalAlignment="Top" Width="50" Height="50" Background="#2b2b2b
    </StackPanel>
    <!--汉堡菜单展开时按钮-->
    <SplitView x:Name="mySplit" DisplayMode="CompactOverlay" CompactPaneLength="0"  OpenPaneLength="150" IsPaneOpen="False" >
        <SplitView.Pane >
            <StackPanel Background="#2b2b2b">
                <Button Background="#0063b1" Click="Button_Click" VerticalAlignment="Top" Width="149" Height="50" Foreground="Azure" FontWeight="Bold">
                    <Grid Width="149">
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="50"/>
                            <ColumnDefinition/>
                        </Grid.ColumnDefinitions>
                        <TextBlock FontFamily="Segoe MDL2 Assets" Text="&#xE700;" Margin="7,0,0,0" VerticalAlignment="Center" ></TextBlock>
                        <TextBlock FontSize="18" Margin="3,0,0,0" FontFamily="Microsoft YaHei UI" Grid.Column="1">菜单</TextBlock>
                    </Grid>
                </Button>
                <Button Click="FButton_Click"  VerticalAlignment="Top" Height="50" Background="#2b2b2b" Width="149" Foreground="Azure">
                    <Grid>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="50"/>
                            <ColumnDefinition/>
                        </Grid.ColumnDefinitions>
                        <TextBlock FontFamily="Segoe MDL2 Assets" Text="&#xE91C;" Margin="5,0,0,0" VerticalAlignment="Center" ></TextBlock>
                        <TextBlock FontSize="18" FontWeight="Light" FontFamily="Microsoft YaHei UI" Grid.Column="1">星座简介</TextBlock>
                    </Grid>
                </Button>
                <Button Click="SButton_Click"  VerticalAlignment="Top" Height="50" Background="#2b2b2b" Width="149" Foreground="Azure">
                    <Grid>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="50"/>
                            <ColumnDefinition/>
                        </Grid.ColumnDefinitions>
                        <TextBlock FontFamily="Segoe MDL2 Assets" Text="&#xE815;" Margin="5,0,0,0" VerticalAlignment="Center"></TextBlock>
                        <TextBlock FontSize="18" FontWeight="Light" FontFamily="Microsoft YaHei UI" Grid.Column="1">运势查询</TextBlock>
                    </Grid>
                </Button>
            </StackPanel>
        </SplitView.Pane>
    </SplitView>

</Grid>
```

主页面添加 frame 用于导航新添加三个的页面



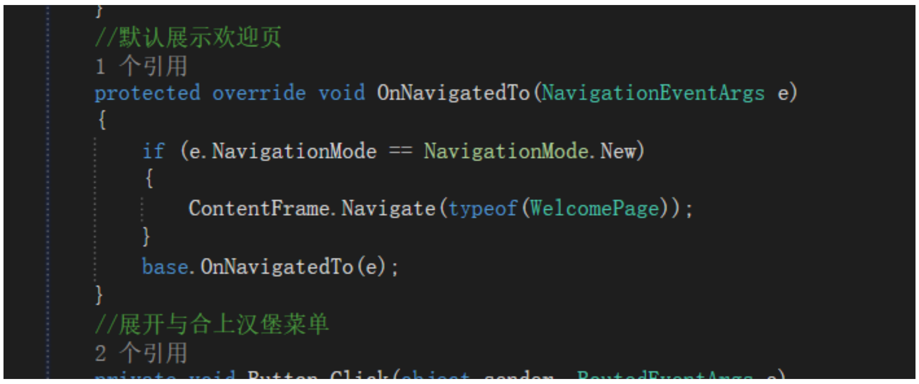
```
        <Button Click="FButton_Click" FontFamily="Segoe MDL2 Asse
        <Button Click="SButton_Click" FontFamily="Segoe MDL2 Asse
    </StackPanel>
    <!--页面框架-->
    <Frame x:Name="ContentFrame" Margin="50,0,0,0"></Frame>
    <!--汉堡菜单展开时按钮-->
    <SplitView x:Name="mySplit" DisplayMode="CompactOverlay" Comp
        <SplitView.Pane >
            <StackPanel Background="#2b2b2b">
```

实现各个按钮转跳页面功能



```
6 个引用
public sealed partial class MainPage : Page
{
    1 个引用
    public MainPage()
    {
        this.InitializeComponent();
    }
    //展开与合上汉堡菜单
    2 个引用
    private void Button_Click(object sender, RoutedEventArgs e)
    {
        mySplit.IsPaneOpen = !mySplit.IsPaneOpen;
    }
    //导航至介绍页
    2 个引用
    private void FButton_Click(object sender, RoutedEventArgs e)
    {
            ContentFrame.Navigate(typeof(InformationPage));
        if (mySplit.IsPaneOpen == true)
            mySplit.IsPaneOpen = !mySplit.IsPaneOpen;
    }
    //导航至查询页
    2 个引用
    private void SButton_Click(object sender, RoutedEventArgs e)
    {
            ContentFrame.Navigate(typeof(DetailPage));
        if(mySplit.IsPaneOpen ==true)
            mySplit.IsPaneOpen = !mySplit.IsPaneOpen;
    }
}
```

实现 frame 默认为欢迎页



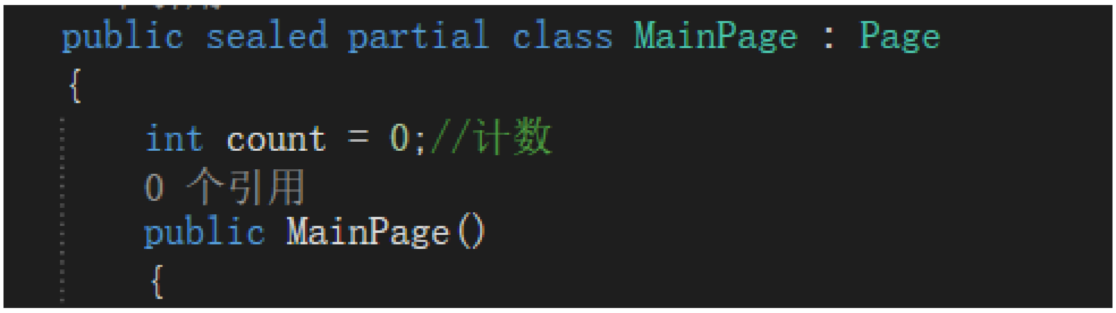
```
    }
    //默认展示欢迎页
    1 个引用
    protected override void OnNavigatedTo(NavigationEventArgs e)
    {
        if (e.NavigationMode == NavigationMode.New)
        {
            ContentFrame.Navigate(typeof(WelcomePage));
        }
        base.OnNavigatedTo(e);
    }
    //展开与合上汉堡菜单
    2 个引用
    private void Button_Click(object sender, RoutedEventArgs e)
```

4.22-4.23

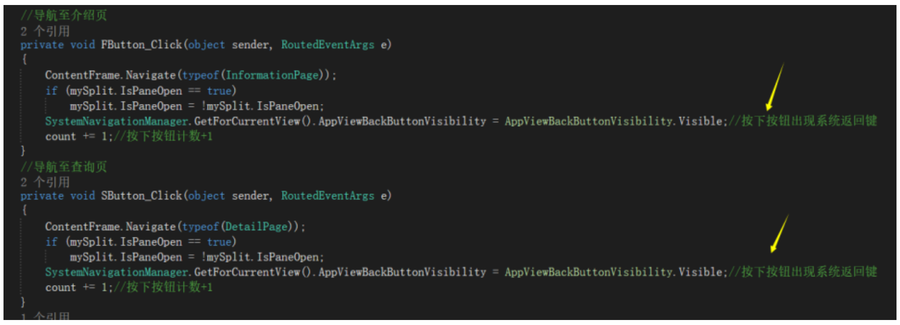
=_=||休整预习高数，准备期中考

4.24

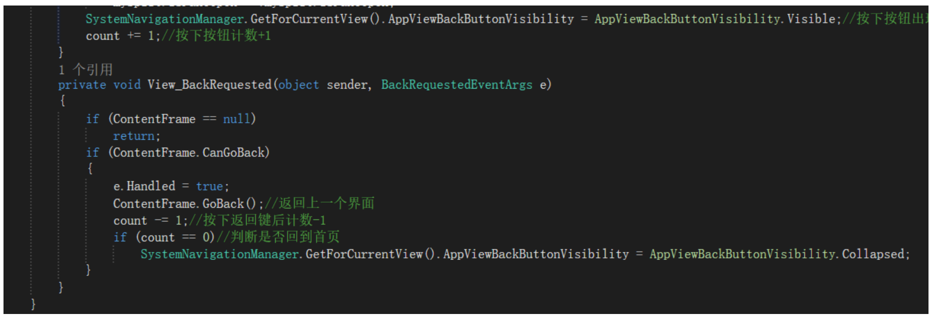
增加系统返回键，用于回退窗口，实现回退功能并增加逻辑判断，若为首页则返回键不出现



```
public sealed partial class MainPage : Page
{
    int count = 0;//计数
    0 个引用
    public MainPage()
    {
```



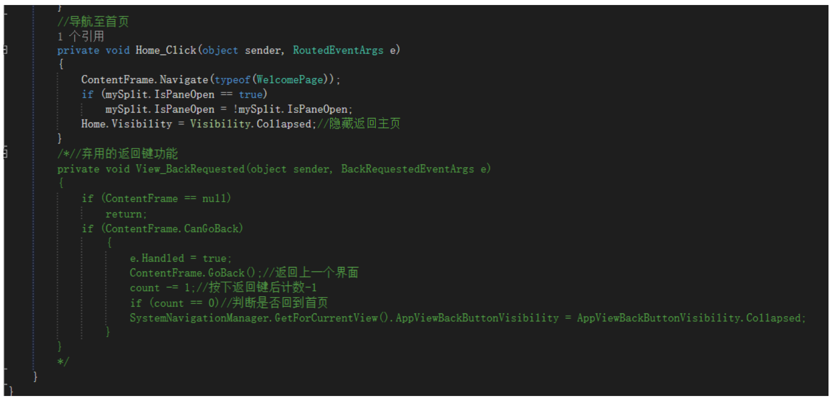
```
//导航至介绍页
2 个引用
private void FButton_Click(object sender, RoutedEventArgs e)
{
    ContentFrame.Navigate(typeof(InformationPage));
    if (mySplit.IsPaneOpen == true)
        mySplit.IsPaneOpen = !mySplit.IsPaneOpen;
    SystemNavigationManager.GetForCurrentView().AppViewBackButtonVisibility = AppViewBackButtonVisibility.Visible;//按下按钮出现系统返回键
    count += 1;//按下按钮计数+1
}
//导航至查询页
2 个引用
private void SButton_Click(object sender, RoutedEventArgs e)
{
    ContentFrame.Navigate(typeof(DetailPage));
    if (mySplit.IsPaneOpen == true)
        mySplit.IsPaneOpen = !mySplit.IsPaneOpen;
    SystemNavigationManager.GetForCurrentView().AppViewBackButtonVisibility = AppViewBackButtonVisibility.Visible;//按下按钮出现系统返回键
    count += 1;//按下按钮计数+1
}
1 个引用
```



```
            SystemNavigationManager.GetForCurrentView().AppViewBackButtonVisibility = AppViewBackButtonVisibility.Visible;//按下按钮出
            count += 1;//按下按钮计数+1
        }
        1 个引用
        private void View_BackRequested(object sender, BackRequestedEventArgs e)
        {
            if (ContentFrame == null)
                return;
            if (ContentFrame.CanGoBack)
            {
                e.Handled = true;
                ContentFrame.GoBack();//返回上一个界面
                count -= 1;//按下返回键后计数-1
                if (count == 0)//判断是否回到首页
                    SystemNavigationManager.GetForCurrentView().AppViewBackButtonVisibility = AppViewBackButtonVisibility.Collapsed;
            }
        }
    }
```

4.25

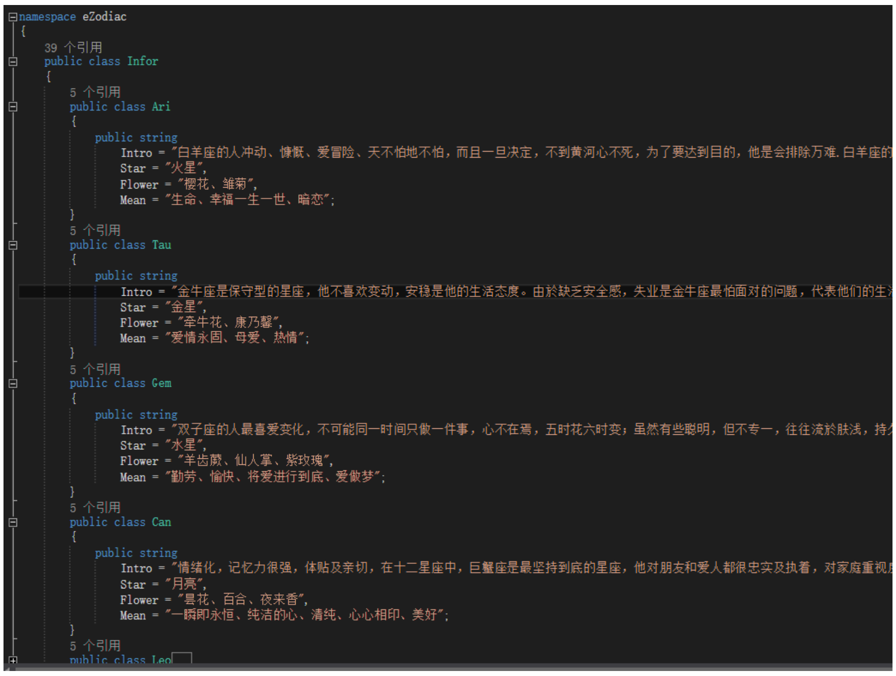
注释掉上个返回键方案，改为返回主页按钮



```
        }
        //导航至首页
        1 个引用
        private void Home_Click(object sender, RoutedEventArgs e)
        {
            ContentFrame.Navigate(typeof(WelcomePage));
            if (mySplit.IsPaneOpen == true)
                mySplit.IsPaneOpen = !mySplit.IsPaneOpen;
            Home.Visibility = Visibility.Collapsed;//隐藏返回主页
        }
        /*//弃用的返回键功能
        private void View_BackRequested(object sender, BackRequestedEventArgs e)
        {
            if (ContentFrame == null)
                return;
            if (ContentFrame.CanGoBack)
                {
                    e.Handled = true;
                    ContentFrame.GoBack();//返回上一个界面
                    count -= 1;//按下返回键后计数-1
                    if (count == 0)//判断是否回到首页
                    SystemNavigationManager.GetForCurrentView().AppViewBackButtonVisibility = AppViewBackButtonVisibility.Collapsed;
                }
        }
        */
    }
}
```

4.26

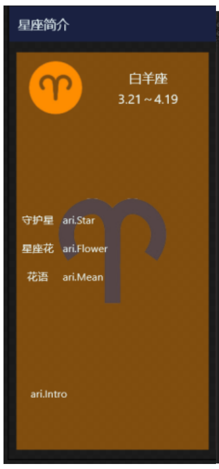
将星座信息写到类里。。。

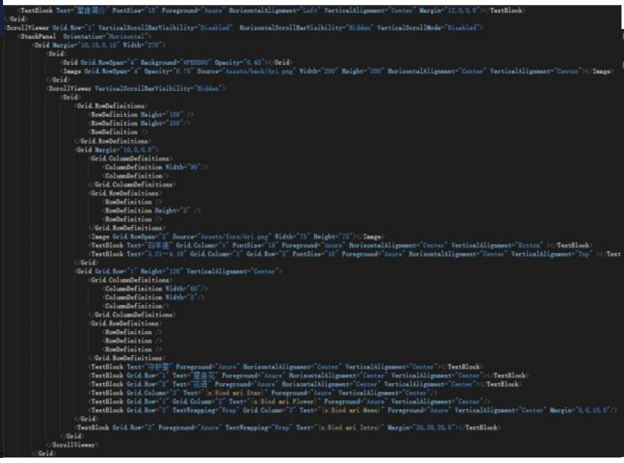


```
namespace eZodiac
{
    39 个引用
    public class Infor
    {
        5 个引用
        public class Ari
        {
            public string
                Intro = "白羊座的人冲动、慷慨、爱冒险、天不怕地不怕，而且一旦决定，不到黄河心不死，为了要达到目的，他是会排除万难.白羊座的
                Star = "火星",
                Flower = "樱花、雏菊",
                Mean = "生命、幸福一生一世、暗恋";
        }
        5 个引用
        public class Tau
        {
            public string
                Intro = "金牛座是保守型的星座，他不喜欢变动，安稳是他的生活态度。由於缺乏安全感，失业是金牛座最怕面对的问题，代表他们的生
                Star = "金星",
                Flower = "牵牛花、康乃馨",
                Mean = "爱情永固、母爱、热情";
        }
        5 个引用
        public class Gem
        {
            public string
                Intro = "双子座的人最喜爱变化，不可能同一时间只做一件事，心不在焉，五时花六时变；虽然有些聪明，但不专一，往往流於肤浅，持
                Star = "水星",
                Flower = "羊齿蕨、仙人掌、紫玫瑰",
                Mean = "勤劳、愉快、将爱进行到底、爱做梦";
        }
        5 个引用
        public class Can
        {
            public string
                Intro = "情绪化，记忆力很强，体贴及亲切，在十二星座中，巨蟹座是最坚持到底的星座，他对朋友和爱人都很忠实及执着，对家庭重视
                Star = "月亮",
                Flower = "昙花、百合、夜来香",
                Mean = "一瞬即永恒、纯洁的心、清纯、心心相印、美好";
        }
        5 个引用
        public class Leo
```

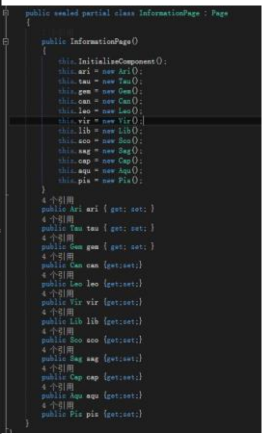
4.27-4.28

写星座介绍界面布局，书签式（可能文字布局不是很好看-_-||）数据用上了前一天写的类。。。实例化后绑定





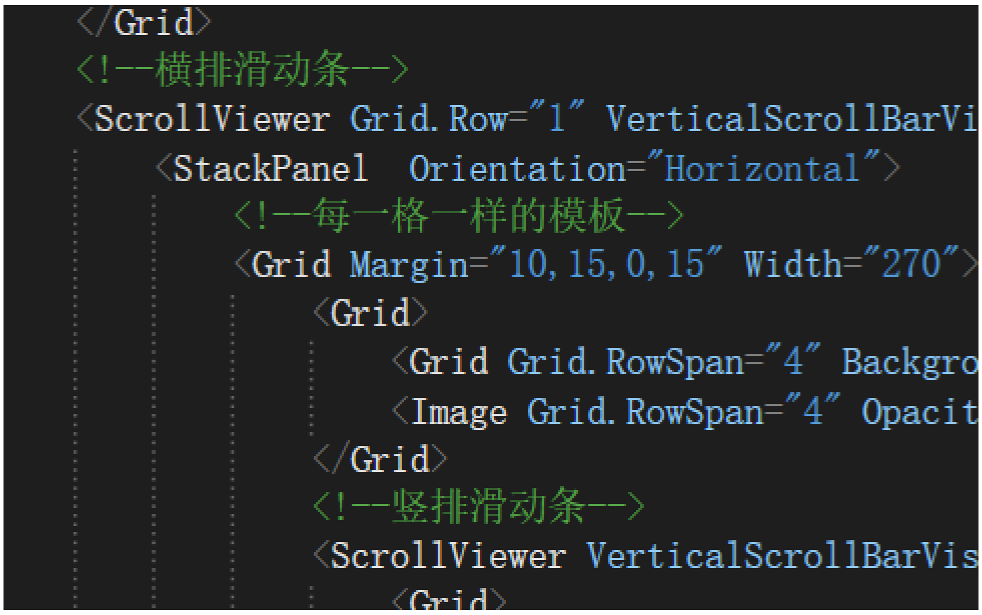
```
    <TextBlock Text="星座简介" FontSize="18" Foreground="Azure" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="12,0,0,0"></TextBlock>
</Grid>
<ScrollViewer Grid.Row="1" VerticalScrollBarVisibility="Disabled"  HorizontalScrollBarVisibility="Hidden" VerticalScrollMode="Disabled">
    <StackPanel  Orientation="Horizontal">
        <Grid Margin="10,15,0,15" Width="270">
            <Grid>
                <Grid Grid.RowSpan="4" Background="#F5B300" Opacity="0.45"></Grid>
                <Image Grid.RowSpan="4" Opacity="0.75" Source="Assets/back/Ari.png" Width="200" Height="200" HorizontalAlignment="Center" VerticalAlignment="Center"></Image>
            </Grid>
            <ScrollViewer VerticalScrollBarVisibility="Hidden">
                <Grid>
                    <Grid.RowDefinitions>
                        <RowDefinition Height="100" />
                        <RowDefinition Height="350"/>
                        <RowDefinition />
                    </Grid.RowDefinitions>
                    <Grid Margin="10,0,0,0">
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="90"/>
                            <ColumnDefinition/>
                        </Grid.ColumnDefinitions>
                        <Grid.RowDefinitions>
                            <RowDefinition />
                            <RowDefinition Height="5" />
                            <RowDefinition />
                        </Grid.RowDefinitions>
                        <Image Grid.RowSpan="3" Source="Assets/fore/Ari.png" Width="75" Height="75"></Image>
                        <TextBlock Text="白羊座" Grid.Column="1" FontSize="18" Foreground="Azure" HorizontalAlignment="Center" VerticalAlignment="Bottom" ></TextBlock>
                        <TextBlock Text="3.21~4.19" Grid.Column="1" Grid.Row="2" FontSize="18" Foreground="Azure" HorizontalAlignment="Center" VerticalAlignment="Top" ></Text
                    </Grid>
                    <Grid Grid.Row="1" Height="120" VerticalAlignment="Center">
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="60"/>
                            <ColumnDefinition Width="5"/>
                            <ColumnDefinition/>
                        </Grid.ColumnDefinitions>
                        <Grid.RowDefinitions>
                            <RowDefinition />
                            <RowDefinition />
                            <RowDefinition />
                        </Grid.RowDefinitions>
                        <TextBlock Text="守护星" Foreground="Azure" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>
                        <TextBlock Grid.Row="1" Text="星座花" Foreground="Azure" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>
                        <TextBlock Grid.Row="2" Text="花语" Foreground="Azure" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>
                        <TextBlock Grid.Column="2" Text="{x:Bind ari.Star}" Foreground="Azure" VerticalAlignment="Center"/>
                        <TextBlock Grid.Row="1" Grid.Column="2" Text="{x:Bind ari.Flower}" Foreground="Azure" VerticalAlignment="Center"/>
                        <TextBlock Grid.Row="2" TextWrapping="Wrap" Grid.Column="2" Text="{x:Bind ari.Mean}" Foreground="Azure" VerticalAlignment="Center" Margin="0,0,10,0"/>
                    </Grid>
                    <TextBlock Grid.Row="2" Foreground="Azure" TextWrapping="Wrap" Text="{x:Bind ari.Intro}" Margin="20,20,20,0"></TextBlock>
                </Grid>
            </ScrollViewer>
        </Grid>
```



```
public sealed partial class InformationPage : Page
{

    public InformationPage()
    {
        this.InitializeComponent();
        this.ari = new Ari();
        this.tau = new Tau();
        this.gem = new Gem();
        this.can = new Can();
        this.leo = new Leo();
        this.vir = new Vir();
        this.lib = new Lib();
        this.sco = new Sco();
        this.sag = new Sag();
        this.cap = new Cap();
        this.aqu = new Aqu();
        this.pis = new Pis();
    }
    4 个引用
    public Ari ari { get; set; }
    4 个引用
    public Tau tau { get; set; }
    4 个引用
    public Gem gem { get; set; }
    4 个引用
    public Can can {get;set;}
    4 个引用
    public Leo leo {get;set;}
    4 个引用
    public Vir vir {get;set;}
    4 个引用
    public Lib lib {get;set;}
    4 个引用
    public Sco sco {get;set;}
    4 个引用
    public Sag sag {get;set;}
    4 个引用
    public Cap cap {get;set;}
    4 个引用
    public Aqu aqu {get;set;}
    4 个引用
    public Pis pis {get;set;}
}
```

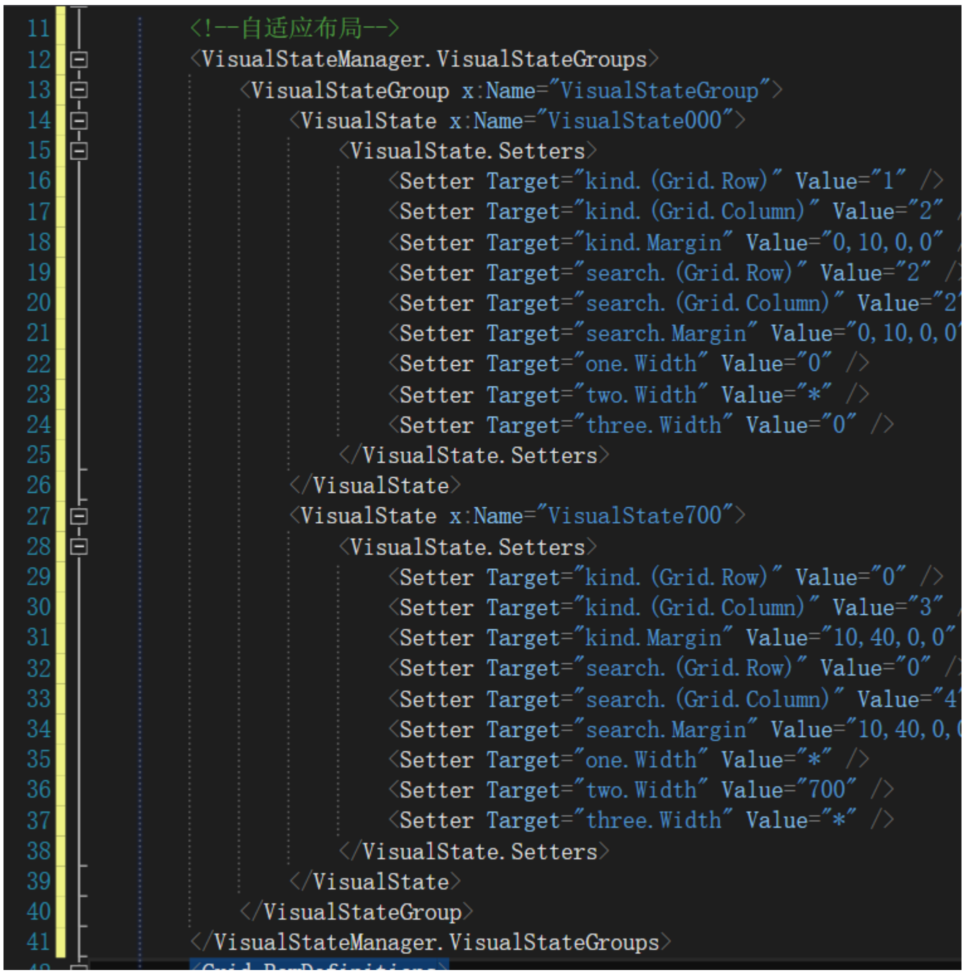
4.29-4.30

对前面一个页面添加注释



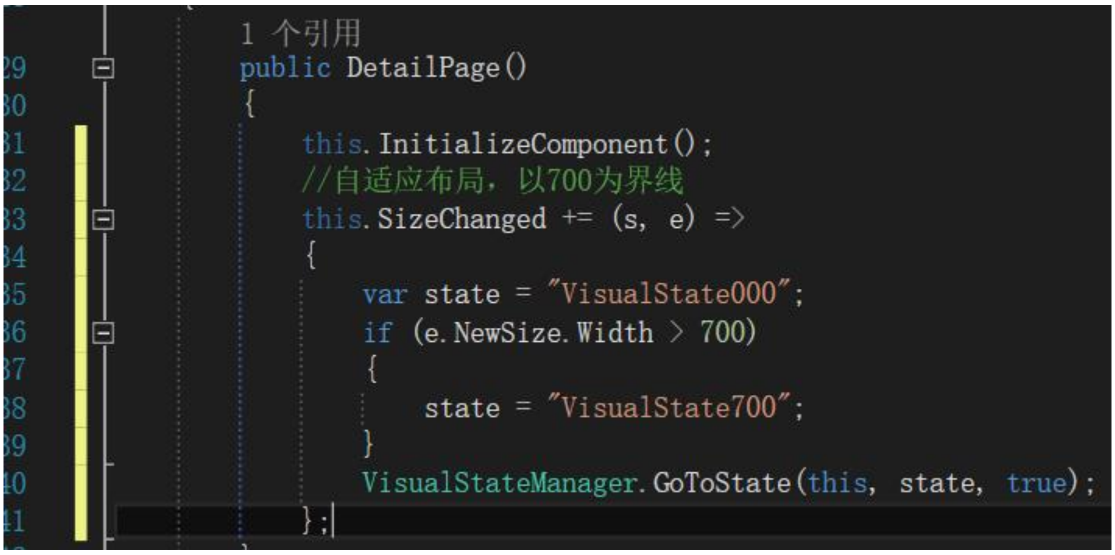
```
    </Grid>
    <!--横排滑动条-->
    <ScrollViewer Grid.Row="1" VerticalScrollBarVi
        <StackPanel  Orientation="Horizontal">
            <!--每一格一样的模板-->
            <Grid Margin="10,15,0,15" Width="270">
                <Grid>
                    <Grid Grid.RowSpan="4" Backgro
                    <Image Grid.RowSpan="4" Opacit
                </Grid>
                <!--竖排滑动条-->
                <ScrollViewer VerticalScrollBarVis
                    <Grid>
```

开始写查询页，首先是自适应布局



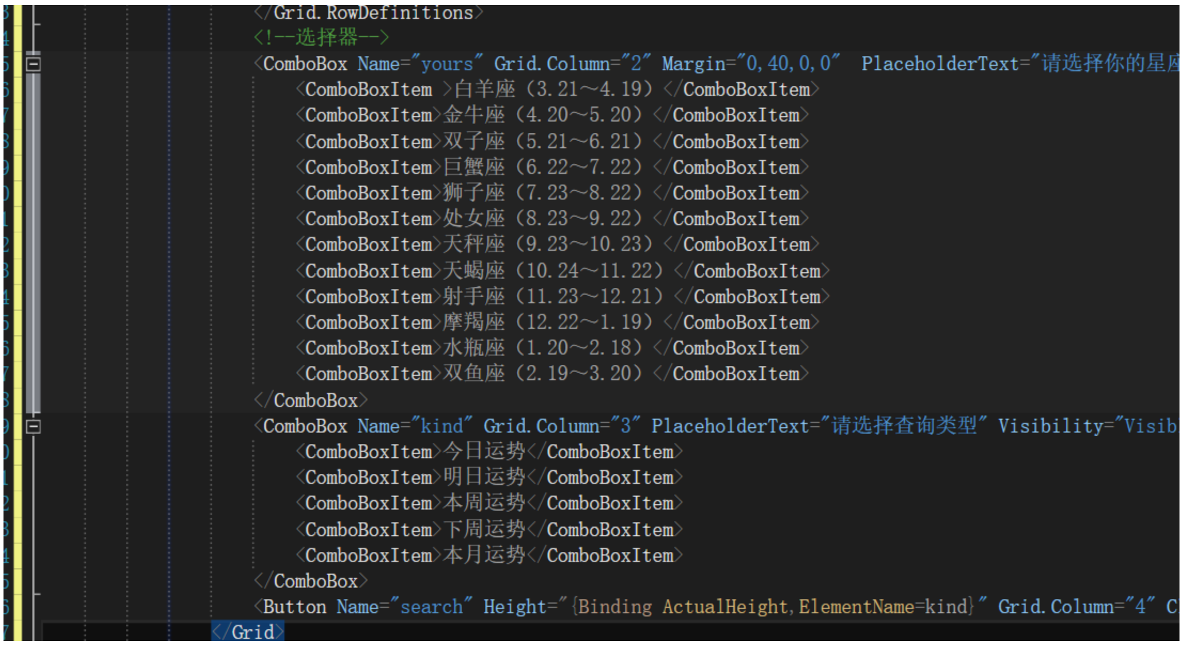
```
        <!--自适应布局-->
        <VisualStateManager.VisualStateGroups>
            <VisualStateGroup x:Name="VisualStateGroup">
                <VisualState x:Name="VisualState000">
                    <VisualState.Setters>
                        <Setter Target="kind.(Grid.Row)" Value="1" />
                        <Setter Target="kind.(Grid.Column)" Value="2"
                        <Setter Target="kind.Margin" Value="0,10,0,0"
                        <Setter Target="search.(Grid.Row)" Value="2" /
                        <Setter Target="search.(Grid.Column)" Value="2
                        <Setter Target="search.Margin" Value="0,10,0,0
                        <Setter Target="one.Width" Value="0" />
                        <Setter Target="two.Width" Value="*" />
                        <Setter Target="three.Width" Value="0" />
                    </VisualState.Setters>
                </VisualState>
                <VisualState x:Name="VisualState700">
                    <VisualState.Setters>
                        <Setter Target="kind.(Grid.Row)" Value="0" />
                        <Setter Target="kind.(Grid.Column)" Value="3"
                        <Setter Target="kind.Margin" Value="10,40,0,0"
                        <Setter Target="search.(Grid.Row)" Value="0" /
                        <Setter Target="search.(Grid.Column)" Value="4
                        <Setter Target="search.Margin" Value="10,40,0,
                        <Setter Target="one.Width" Value="*" />
                        <Setter Target="two.Width" Value="700" />
                        <Setter Target="three.Width" Value="*" />
                    </VisualState.Setters>
                </VisualState>
            </VisualStateGroup>
        </VisualStateManager.VisualStateGroups>
```

后台代码实现自适应



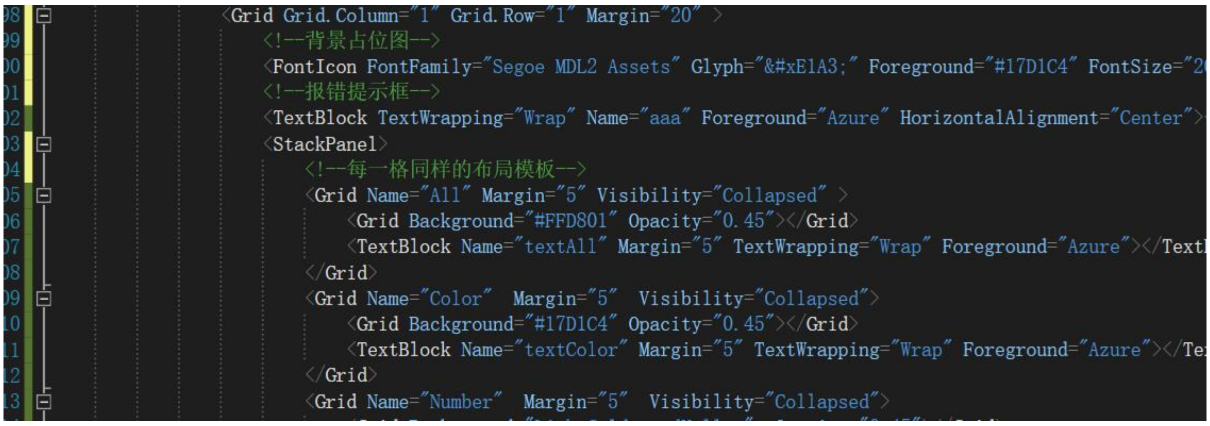
```
1 个引用
public DetailPage()
{
    this.InitializeComponent();
    //自适应布局，以700为界线
    this.SizeChanged += (s, e) =>
    {
        var state = "VisualState000";
        if (e.NewSize.Width > 700)
        {
            state = "VisualState700";
        }
        VisualStateManager.GoToState(this, state, true);
    };
```

添加选择器和查询按钮



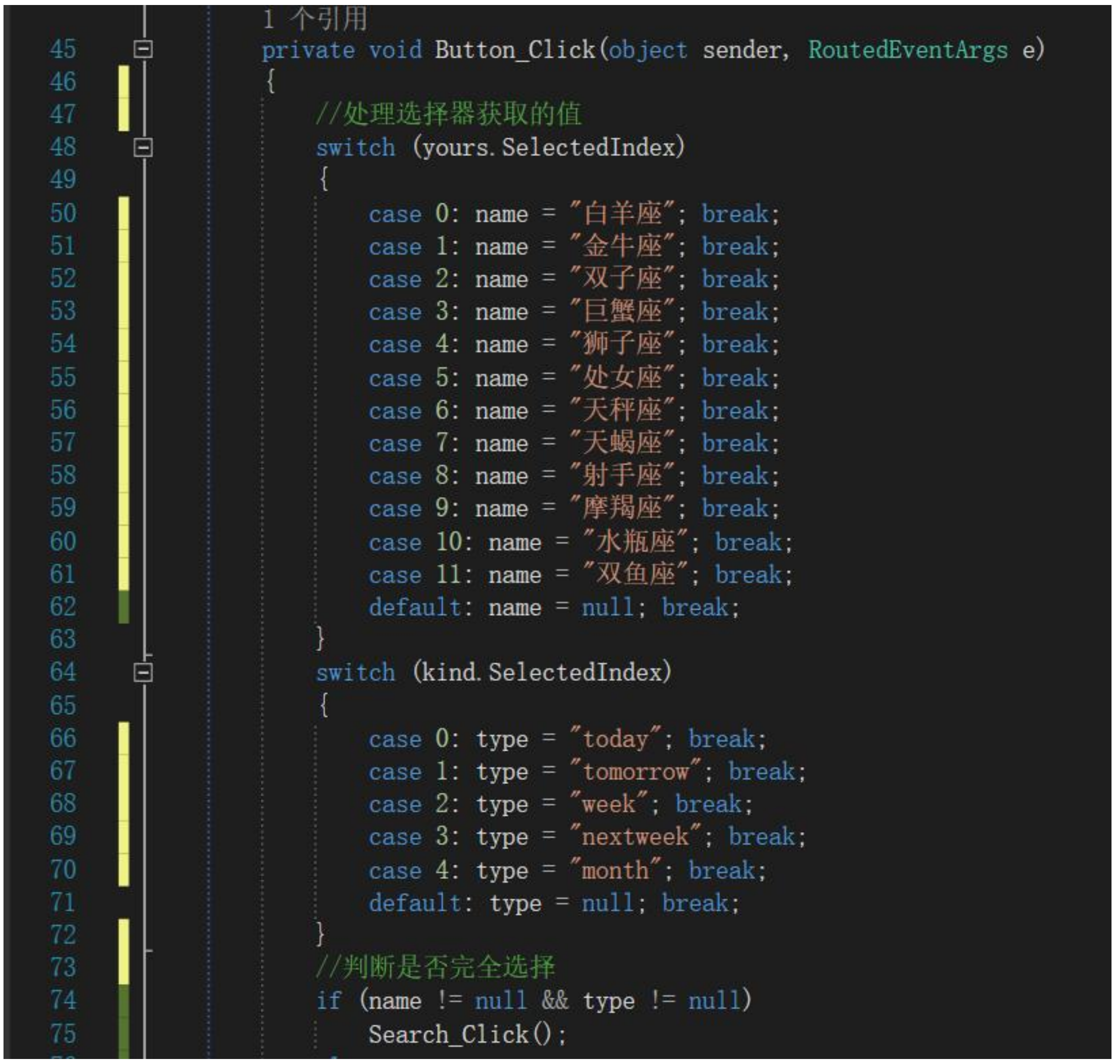
```
</Grid.RowDefinitions>
<!--选择器-->
<ComboBox Name="yours" Grid.Column="2" Margin="0,40,0,0"  PlaceholderText="请选择你的星座
    <ComboBoxItem >白羊座 (3.21~4.19) </ComboBoxItem>
    <ComboBoxItem>金牛座 (4.20~5.20) </ComboBoxItem>
    <ComboBoxItem>双子座 (5.21~6.21) </ComboBoxItem>
    <ComboBoxItem>巨蟹座 (6.22~7.22) </ComboBoxItem>
    <ComboBoxItem>狮子座 (7.23~8.22) </ComboBoxItem>
    <ComboBoxItem>处女座 (8.23~9.22) </ComboBoxItem>
    <ComboBoxItem>天秤座 (9.23~10.23) </ComboBoxItem>
    <ComboBoxItem>天蝎座 (10.24~11.22) </ComboBoxItem>
    <ComboBoxItem>射手座 (11.23~12.21) </ComboBoxItem>
    <ComboBoxItem>摩羯座 (12.22~1.19) </ComboBoxItem>
    <ComboBoxItem>水瓶座 (1.20~2.18) </ComboBoxItem>
    <ComboBoxItem>双鱼座 (2.19~3.20) </ComboBoxItem>
</ComboBox>
<ComboBox Name="kind" Grid.Column="3" PlaceholderText="请选择查询类型" Visibility="Visib
    <ComboBoxItem>今日运势</ComboBoxItem>
    <ComboBoxItem>明日运势</ComboBoxItem>
    <ComboBoxItem>本周运势</ComboBoxItem>
    <ComboBoxItem>下周运势</ComboBoxItem>
    <ComboBoxItem>本月运势</ComboBoxItem>
</ComboBox>
<Button Name="search" Height="{Binding ActualHeight,ElementName=kind}" Grid.Column="4" C
</Grid>
```

写背景占位图，报错提示框和每个模块



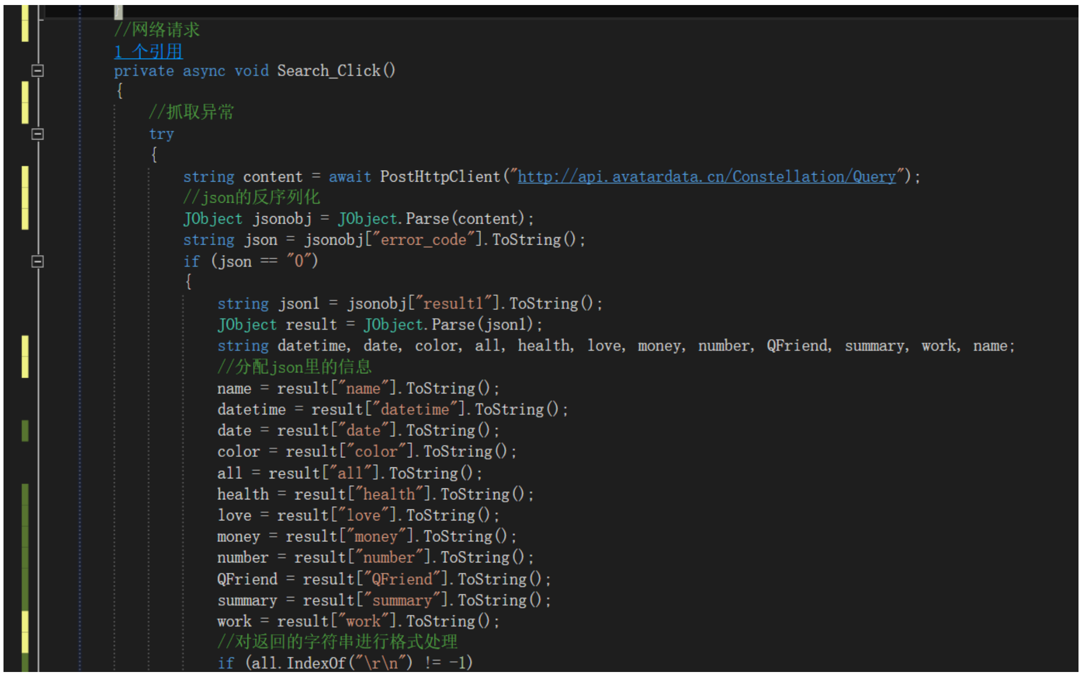
```
<Grid Grid.Column="1" Grid.Row="1" Margin="20" >
    <!--背景占位图-->
    <FontIcon FontFamily="Segoe MDL2 Assets" Glyph="&#xE1A3;" Foreground="#17D1C4" FontSize="2
    <!--报错提示框-->
    <TextBlock TextWrapping="Wrap" Name="aaa" Foreground="Azure" HorizontalAlignment="Center">
    <StackPanel>
        <!--每一格同样的布局模板-->
        <Grid Name="All" Margin="5" Visibility="Collapsed" >
            <Grid Background="#FFD801" Opacity="0.45"></Grid>
            <TextBlock Name="textAll" Margin="5" TextWrapping="Wrap" Foreground="Azure"></Text
        </Grid>
        <Grid Name="Color"  Margin="5"  Visibility="Collapsed">
            <Grid Background="#17D1C4" Opacity="0.45"></Grid>
            <TextBlock Name="textColor" Margin="5" TextWrapping="Wrap" Foreground="Azure"></Te
        </Grid>
        <Grid Name="Number"  Margin="5"  Visibility="Collapsed">
```

查询按钮的功能实现



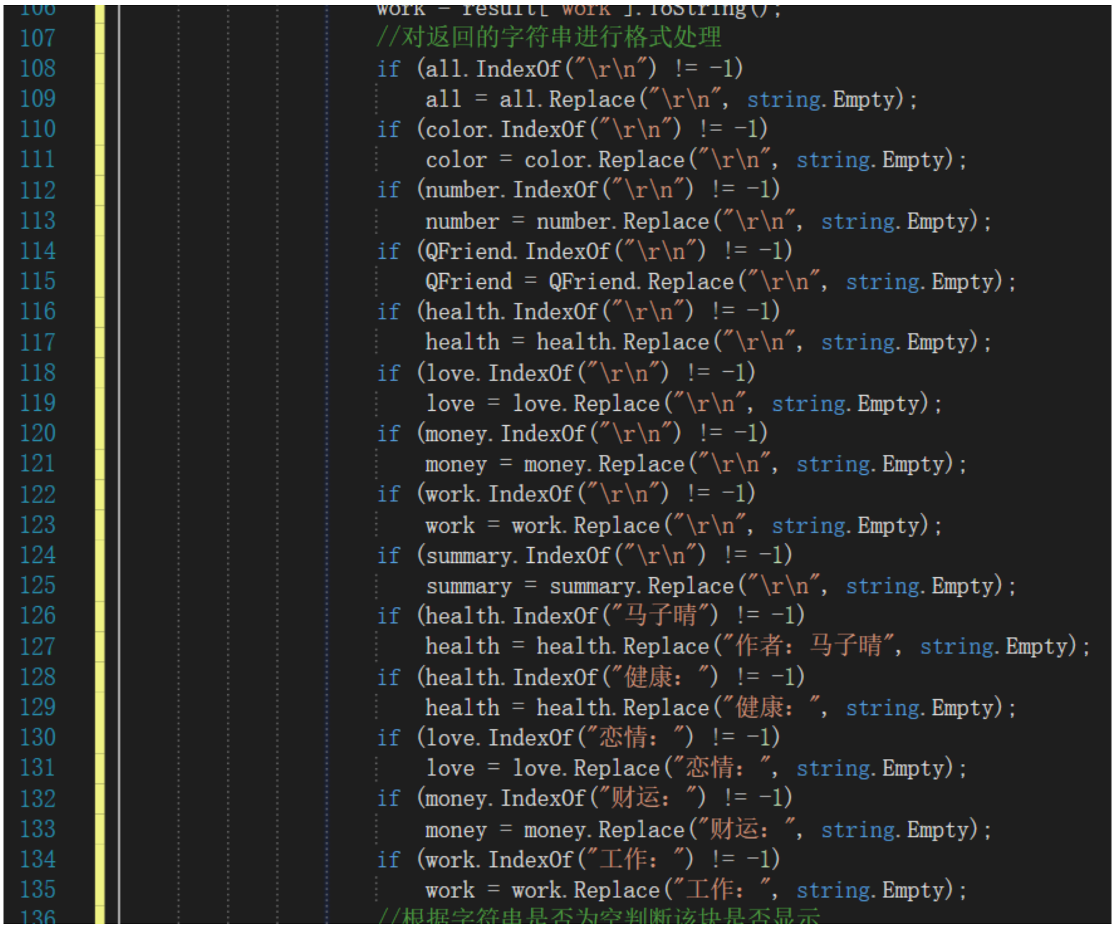
```
1 个引用
private void Button_Click(object sender, RoutedEventArgs e)
{
    //处理选择器获取的值
    switch (yours.SelectedIndex)
    {
        case 0: name = "白羊座"; break;
        case 1: name = "金牛座"; break;
        case 2: name = "双子座"; break;
        case 3: name = "巨蟹座"; break;
        case 4: name = "狮子座"; break;
        case 5: name = "处女座"; break;
        case 6: name = "天秤座"; break;
        case 7: name = "天蝎座"; break;
        case 8: name = "射手座"; break;
        case 9: name = "摩羯座"; break;
        case 10: name = "水瓶座"; break;
        case 11: name = "双鱼座"; break;
        default: name = null; break;
    }
    switch (kind.SelectedIndex)
    {
        case 0: type = "today"; break;
        case 1: type = "tomorrow"; break;
        case 2: type = "week"; break;
        case 3: type = "nextweek"; break;
        case 4: type = "month"; break;
        default: type = null; break;
    }
    //判断是否完全选择
    if (name != null && type != null)
        Search_Click();
```

网络请求，对获取到的 json 反序列化，分配字符串



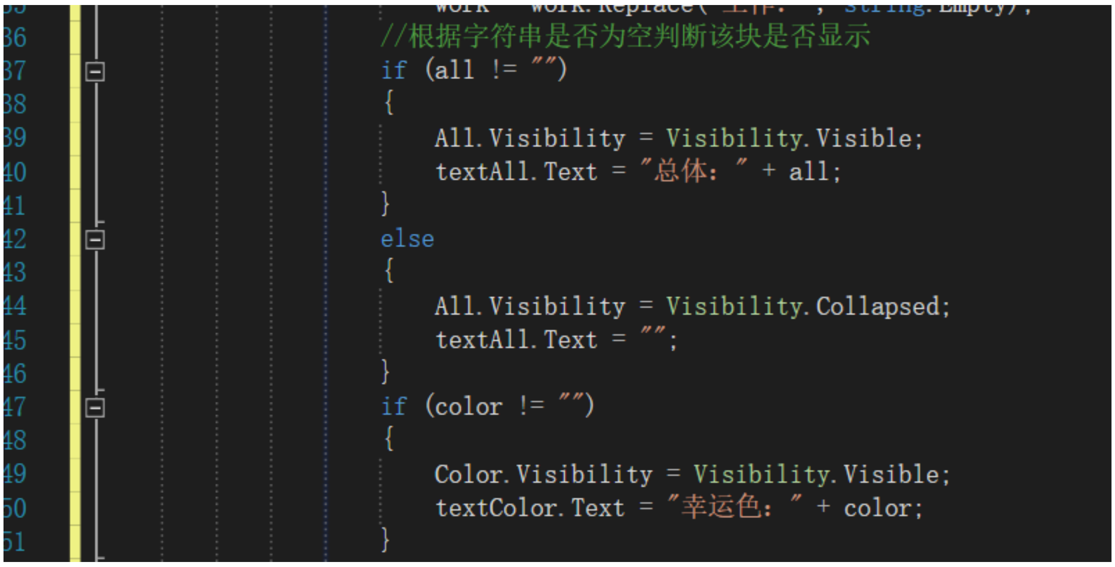
```
//网络请求
1 个引用
private async void Search_Click()
{
    //抓取异常
    try
    {
        string content = await PostHttpClient("http://api.avatardata.cn/Constellation/Query");
        //json的反序列化
        JObject jsonobj = JObject.Parse(content);
        string json = jsonobj["error_code"].ToString();
        if (json == "0")
        {
            string json1 = jsonobj["result1"].ToString();
            JObject result = JObject.Parse(json1);
            string datetime, date, color, all, health, love, money, number, QFriend, summary, work, name;
            //分配json里的信息
            name = result["name"].ToString();
            datetime = result["datetime"].ToString();
            date = result["date"].ToString();
            color = result["color"].ToString();
            all = result["all"].ToString();
            health = result["health"].ToString();
            love = result["love"].ToString();
            money = result["money"].ToString();
            number = result["number"].ToString();
            QFriend = result["QFriend"].ToString();
            summary = result["summary"].ToString();
            work = result["work"].ToString();
            //对返回的字符串进行格式处理
            if (all.IndexOf("\r\n") != -1)
```

处理字符串格式



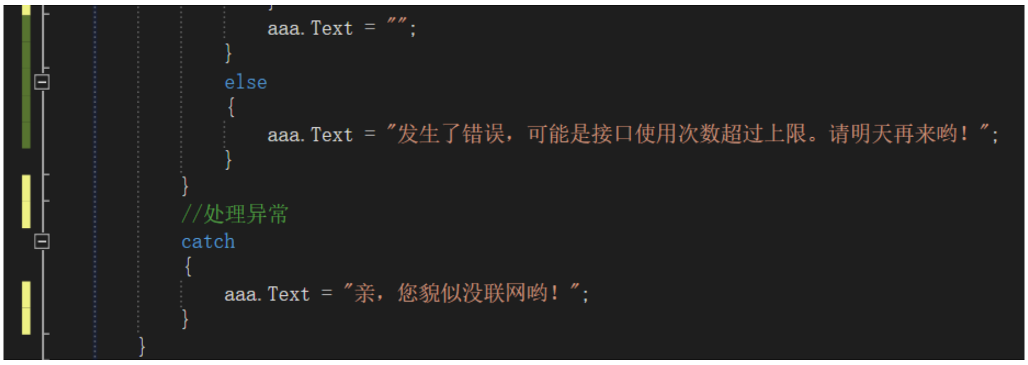
```
work = result["work"].ToString();
//对返回的字符串进行格式处理
if (all.IndexOf("\r\n") != -1)
    all = all.Replace("\r\n", string.Empty);
if (color.IndexOf("\r\n") != -1)
    color = color.Replace("\r\n", string.Empty);
if (number.IndexOf("\r\n") != -1)
    number = number.Replace("\r\n", string.Empty);
if (QFriend.IndexOf("\r\n") != -1)
    QFriend = QFriend.Replace("\r\n", string.Empty);
if (health.IndexOf("\r\n") != -1)
    health = health.Replace("\r\n", string.Empty);
if (love.IndexOf("\r\n") != -1)
    love = love.Replace("\r\n", string.Empty);
if (money.IndexOf("\r\n") != -1)
    money = money.Replace("\r\n", string.Empty);
if (work.IndexOf("\r\n") != -1)
    work = work.Replace("\r\n", string.Empty);
if (summary.IndexOf("\r\n") != -1)
    summary = summary.Replace("\r\n", string.Empty);
if (health.IndexOf("马子晴") != -1)
    health = health.Replace("作者: 马子晴", string.Empty);
if (health.IndexOf("健康: ") != -1)
    health = health.Replace("健康: ", string.Empty);
if (love.IndexOf("恋情: ") != -1)
    love = love.Replace("恋情: ", string.Empty);
if (money.IndexOf("财运: ") != -1)
    money = money.Replace("财运: ", string.Empty);
if (work.IndexOf("工作: ") != -1)
    work = work.Replace("工作: ", string.Empty);
//根据字符串是否为空判断该块是否显示
```

判断是否显示模块



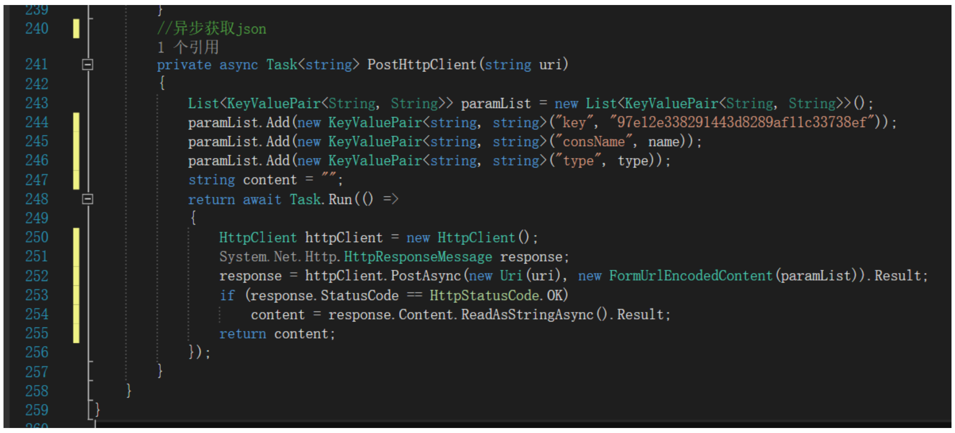
```
    work = work.Replace("工作: ", string.Empty);
//根据字符串是否为空判断该块是否显示
if (all != "")
{
    All.Visibility = Visibility.Visible;
    textAll.Text = "总体: " + all;
}
else
{
    All.Visibility = Visibility.Collapsed;
    textAll.Text = "";
}
if (color != "")
{
    Color.Visibility = Visibility.Visible;
    textColor.Text = "幸运色: " + color;
}
```

报错



```
                    aaa.Text = "";
                }
                else
                {
                    aaa.Text = "发生了错误，可能是接口使用次数超过上限。请明天再来哟！";
                }
            }
            //处理异常
            catch
            {
                aaa.Text = "亲，您貌似没联网哟！";
            }
        }
```

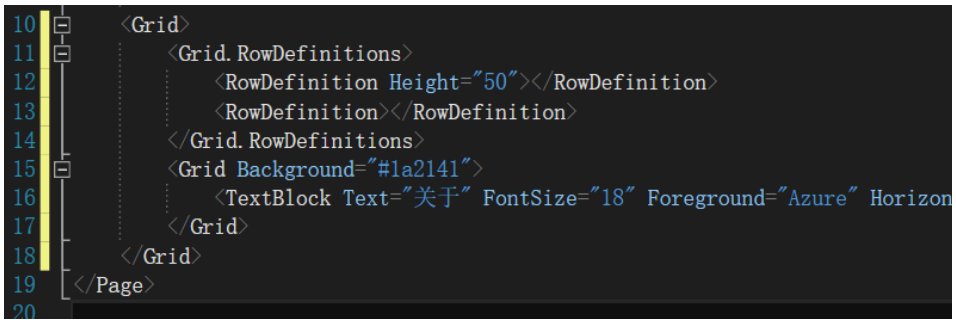
异步，获取 json



```
        }
        //异步获取json
        1 个引用
        private async Task<string> PostHttpClient(string uri)
        {
            List<KeyValuePair<String, String>> paramList = new List<KeyValuePair<String, String>>();
            paramList.Add(new KeyValuePair<string, string>("key", "97e12e338291443d8289af11c33738ef"));
            paramList.Add(new KeyValuePair<string, string>("consName", name));
            paramList.Add(new KeyValuePair<string, string>("type", type));
            string content = "";
            return await Task.Run(() =>
            {
                HttpClient httpClient = new HttpClient();
                System.Net.Http.HttpResponseMessage response;
                response = httpClient.PostAsync(new Uri(uri), new FormUrlEncodedContent(paramList)).Result;
                if (response.StatusCode == HttpStatusCode.OK)
                    content = response.Content.ReadAsStringAsync().Result;
                return content;
            });
        }
    }
}
```

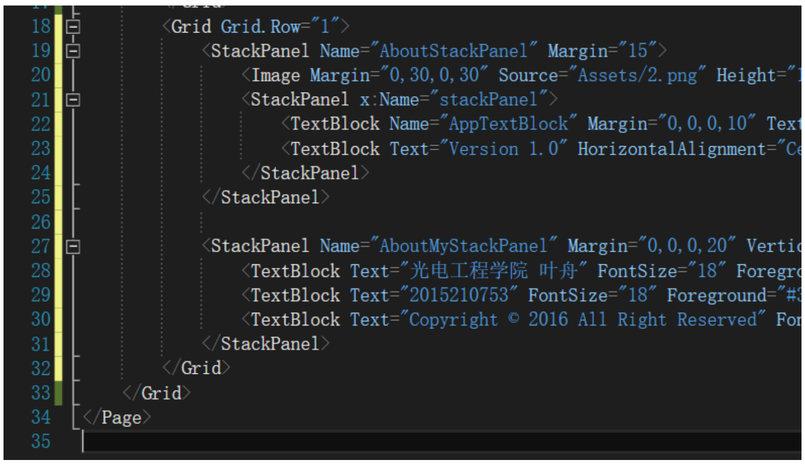
5.1 劳动节快乐 o(^▽^)o

添加空白页，参照前面的页面做相关信息



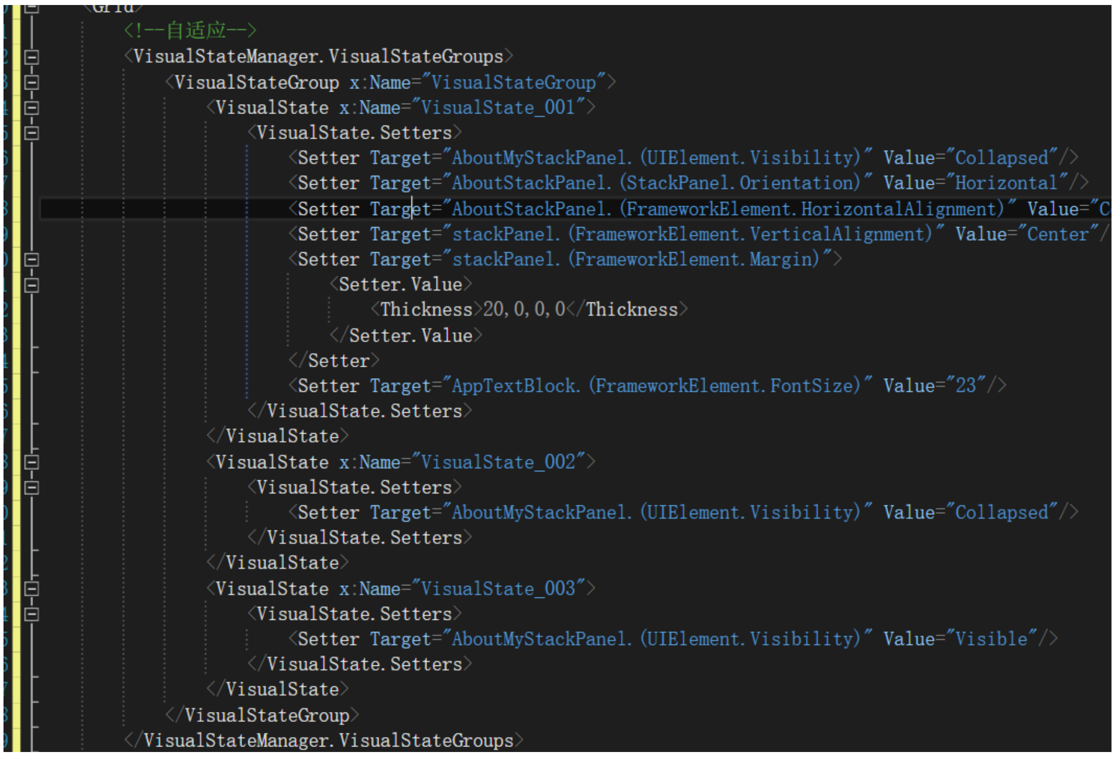
```
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="50"></RowDefinition>
            <RowDefinition></RowDefinition>
        </Grid.RowDefinitions>
        <Grid Background="#1a2141">
            <TextBlock Text="关于" FontSize="18" Foreground="Azure" Horizon
        </Grid>
    </Grid>
</Page>
```

添加相关信息



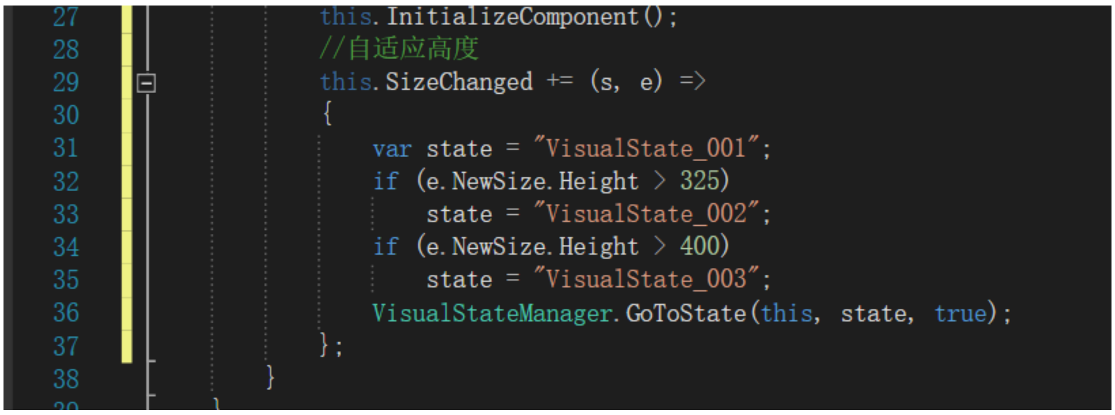
```
        <Grid Grid.Row="1">
            <StackPanel Name="AboutStackPanel" Margin="15">
                <Image Margin="0,30,0,30" Source="Assets/2.png" Height="1
                <StackPanel x:Name="stackPanel">
                    <TextBlock Name="AppTextBlock" Margin="0,0,0,10" Text
                    <TextBlock Text="Version 1.0" HorizontalAlignment="Ce
                </StackPanel>
            </StackPanel>

            <StackPanel Name="AboutMyStackPanel" Margin="0,0,0,20" Vertic
                <TextBlock Text="光电工程学院 叶舟" FontSize="18" Foregro
                <TextBlock Text="2015210753" FontSize="18" Foreground="#3
                <TextBlock Text="Copyright © 2016 All Right Reserved" For
            </StackPanel>
        </Grid>
    </Grid>
</Page>
```

添加自适应高度



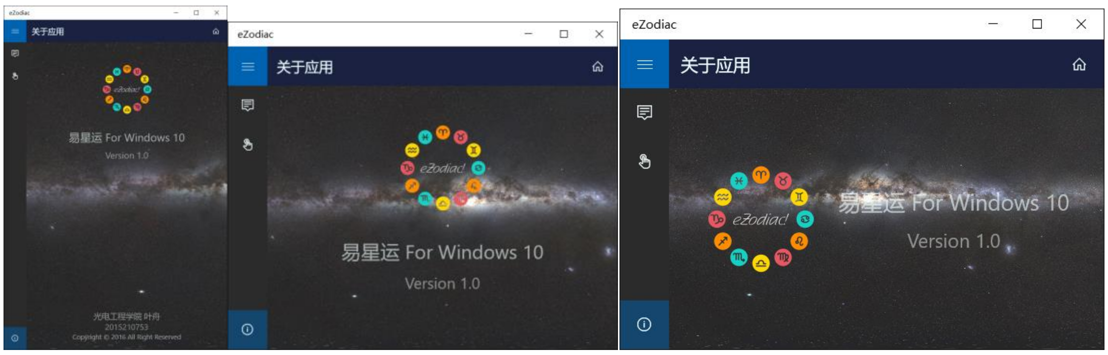
```
        <!--自适应-->
        <VisualStateManager.VisualStateGroups>
            <VisualStateGroup x:Name="VisualStateGroup">
                <VisualState x:Name="VisualState_001">
                    <VisualState.Setters>
                        <Setter Target="AboutMyStackPanel.(UIElement.Visibility)" Value="Collapsed"/>
                        <Setter Target="AboutStackPanel.(StackPanel.Orientation)" Value="Horizontal"/>
                        <Setter Target="AboutStackPanel.(FrameworkElement.HorizontalAlignment)" Value="C
                        <Setter Target="stackPanel.(FrameworkElement.VerticalAlignment)" Value="Center"/
                        <Setter Target="stackPanel.(FrameworkElement.Margin)">
                            <Setter.Value>
                                <Thickness>20,0,0,0</Thickness>
                            </Setter.Value>
                        </Setter>
                        <Setter Target="AppTextBlock.(FrameworkElement.FontSize)" Value="23"/>
                    </VisualState.Setters>
                </VisualState>
                <VisualState x:Name="VisualState_002">
                    <VisualState.Setters>
                        <Setter Target="AboutMyStackPanel.(UIElement.Visibility)" Value="Collapsed"/>
                    </VisualState.Setters>
                </VisualState>
                <VisualState x:Name="VisualState_003">
                    <VisualState.Setters>
                        <Setter Target="AboutMyStackPanel.(UIElement.Visibility)" Value="Visible"/>
                    </VisualState.Setters>
                </VisualState>
            </VisualStateGroup>
        </VisualStateManager.VisualStateGroups>
```

实现自适应

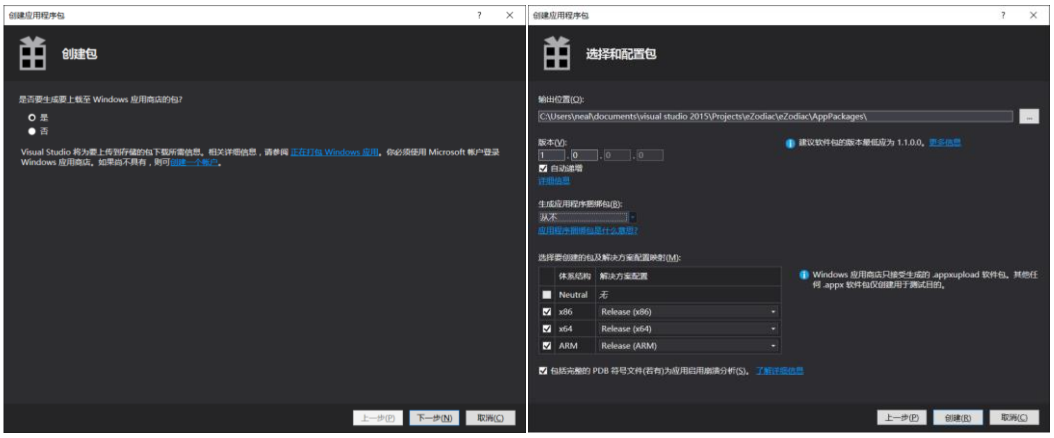


```
            this.InitializeComponent();
            //自适应高度
            this.SizeChanged += (s, e) =>
            {
                var state = "VisualState_001";
                if (e.NewSize.Height > 325)
                    state = "VisualState_002";
                if (e.NewSize.Height > 400)
                    state = "VisualState_003";
                VisualStateManager.GoToState(this, state, true);
            };
        }
```

自适应样式



打包


创建应用程序包

创建包

是否要生成要上载至 Windows 应用商店的包?

- 是
- 否

Visual Studio 将为要上传到存储的包下载所需信息。相关详细信息，请参阅 正在打包 Windows 应用。你必须使用 Microsoft 帐户登录 Windows 应用商店。如果尚不具有，则可创建一个帐户。

上一步(P) 下一步(N) 取消(C)

创建应用程序包

选择和配置包

输出位置(O):
C:\Users\neal\documents\visual studio 2015\Projects\eZodiac\eZodiac\AppPackages\

版本(V):
1 . 0 . 0 . 0
☑ 自动递增
详细信息

建议软件包的版本最低应为 1.1.0.0。更多信息

生成应用程序捆绑包(B):
从不
应用程序捆绑包是什么意思?

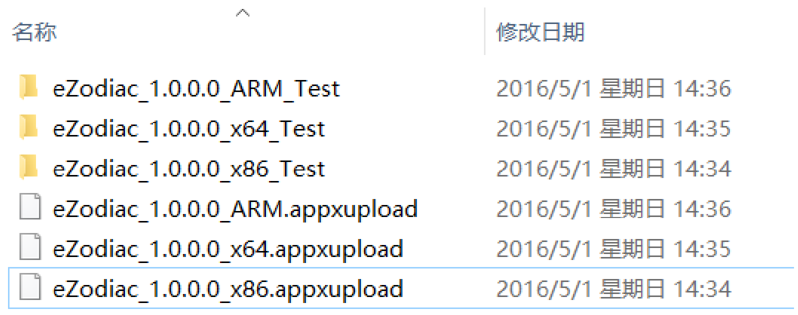
选择要创建的包及解决方案配置映射(M):

	体系结构	解决方案配置
☐	Neutral	无
☑	x86	Release (x86)
☑	x64	Release (x64)
☑	ARM	Release (ARM)

Windows 应用商店只接受生成的 .appxupload 软件包。其他任何 .appx 软件包仅创建用于测试目的。

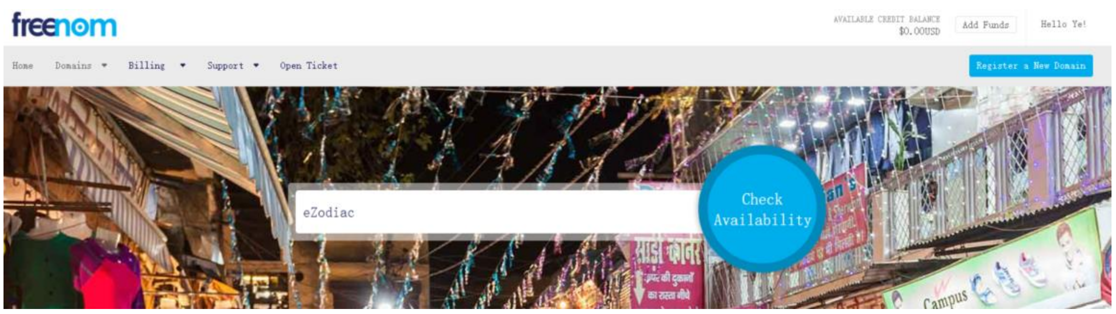
☑ 包括完整的 PDB 符号文件(若有)为应用启用崩溃分析(S)。了解详细信息

上一步(P) 创建(R) 取消(C)



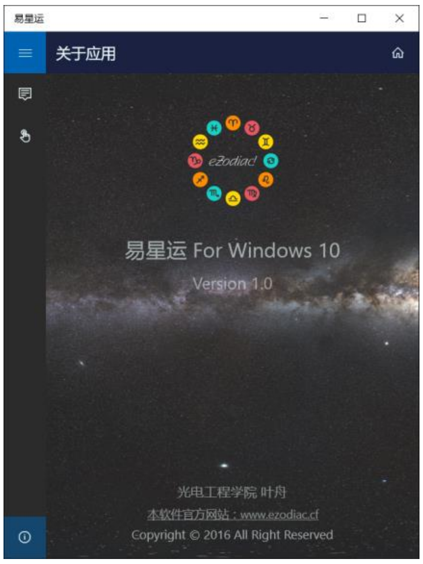
名称	修改日期
eZodiac_1.0.0.0_ARM_Test	2016/5/1 星期日 14:36
eZodiac_1.0.0.0_x64_Test	2016/5/1 星期日 14:35
eZodiac_1.0.0.0_x86_Test	2016/5/1 星期日 14:34
eZodiac_1.0.0.0_ARM.appxupload	2016/5/1 星期日 14:36
eZodiac_1.0.0.0_x64.appxupload	2016/5/1 星期日 14:35
eZodiac_1.0.0.0_x86.appxupload	2016/5/1 星期日 14:34

申请域名



前端界面的编写啊，DNS 解析啊，配置虚拟主机啥的就不写了。。。
网址为 http://www.eZodiac.cf/或者 http://eZodiac.cf/

最后在关于页加上了网址，重新打包了一遍



易星运 For Windows 10

Version 1.0

光电工程学院 叶舟

本软件官方网站：www.ezodiac.cf

Copyright © 2016 All Right Reserved

然后把代码传到 github，地址：https://github.com/nealbity/eZodiac.git

然后，准备打包提交啦~充实的五一节(/≧▽≦/)