How to rename the default namespace in Windows Phone app project

This article explains how you can change the namespace in a Windows Phone app project.

Overview

Visual Studio creates phone projects with a default namespace matching the app's name. While sometimes that is adequate, often you would prefer to add your company name into the namespace or group the namespace name with some other project.

The error message you'll get if the namespaces in .cs and .xaml don't match doesn't directly hint to the problem:

```
D:\Sources\...\PanoramaApp1\PanoramaApp1\App.xaml.cs(11,18,11,27): error CS0234: The type or namespace name
'Resources' does not exist in the namespace 'MyFirstApp' (are you missing an assembly reference?)
```

Even worse is the situation if the startup object is not correctly set: the app compiles and deploys correctly, but closes immediately after starting. There is no obvious cause when debugging!

Solution

If you're just changing the namespace "name" (without adding a parent namespace) then:

1. Change the namespace in an arbitrary code-behind file, say **MainPage.xaml.cs**. The hot tip that appears after the change proposes to rename the namespace in the whole project. Accept!
2. Verify the startup object. Normally you would have to select the newly named `App` class here, say `My.FooBar.App`. It should be selectable in the combo box.

Note that the above order is important! If you first change the default namespace, you have to change all **.cs** and all **.xaml** files manually:

```
namespace MyFirstApp
```

```
<phone:PhoneApplicationPage
    x:Class="MyFirstApp.MainPage"
```

Change to:

```
namespace MyLastApp
```

```
<phone:PhoneApplicationPage
    x:Class="MyLastApp.MainPage"
```

If you have a parent namespace then there are some additional "intermediate" steps:

1. First you change the namespace in an arbitrary code-behind file, say **MainPage.xaml.cs**. The hot tip that appears after the change proposes to rename the namespace in the whole project. Accept!
2. Beware! If you also added a parent namespace (like the "My" in `My.FooBar`), the change affects only the last part of the name! In this case you have to check all .cs and .xaml files for the correct naming.
3. Open the project's property page and see the default namespace entry changed. This has to be changed here manually to include the parent.
4. The last step is to verify the startup object. Normally you would have to select the newly named `App` class here, say `My.FooBar.App`. It should be selectable in the combo box.

Summary

Visual Studio should automate the rename/re-factoring job, but since it doesn't, the above steps provide a clear process for manually handling renaming the namespace.