

Источник - Сергей Звездин: персональный блог (<http://blog.zvezdin.com/ru/250>)

## Использование UriMapper в приложениях WP7

---

Известно, что концепция приложений WP7 включает в себя страницы. Адреса страниц в приложении формируются исходя из физического положения страницы в проекте. В этой заметке я расскажу про объект UriMapper, который позволяет задать адрес страницы независимо от размещения страницы в проекте.

Представим, что в приложении имеется несколько страниц. Для перехода от одной странице к другой следует воспользоваться объектом NavigationService и указать адрес страницы, к которой мы хотим перейти:

```
1 | NavigationService.Navigate(new Uri("<strong>/SettingsPage.xaml</strong>", UriKind.Relative));
```

Адрес страницы формируется из ее названия и расположения внутри проекта. Так, например, если мы поместим эту страницу в подпапку "Pages", то адрес страницы изменится на **"/Pages/SettingsPage.xaml"**; при переносе страницы в другую сборку, адрес также будет изменен.

Эта проблема оказывается действительно неприятной, когда пересматривается структура папок проекта, а то и структура самого проекта. Самое неприятное в этой ситуации то, что при наличии какой-то более-менее сложной навигации можно просто упустить из виду некоторые вызовы NavigationService и допустить ошибку в приложении. Visual Studio и компилятор такие ошибки отловить не смогут. Как результат, при переходе по адресу несуществующей странице приложение просто завершится.

Для того, чтобы этого избежать можно использовать объект UriMapper, который позволяет определить адреса страниц независимо от их расположения в проекте. Фактически, мы задаем некоторый алиас для страницы и используем именно его для перехода на страницы. В этом случае при пересмотре структуры проекта нужно будет только сменить настройки для преобразования алиасов в реальные адреса страницы.

Для того, чтобы использовать UriMapper в своем приложении нужно выполнить следующие действия:

- 1) Создать и определить настройки объекта UriMapper. Обычно я это делаю в ресурсах приложения, в файле App.xaml. Естественно, это совсем необязательно.

```

1 <Application.Resources>
2   <phone:UriMapper x:Name="ApplicationUriMapper">
3     <phone:UriMapping Uri="/Home" MappedUri="/Views/MainPage.xaml"/>
4     <phone:UriMapping Uri="/Session/{ID}" MappedUri="/Views/Session.xaml?ID={ID}"/>
5     <phone:UriMapping Uri="/Sessions" MappedUri="/Views/SessionTracks.xaml"/>
6     <phone:UriMapping Uri="/About" MappedUri="/Views/About.xaml"/>
7   </phone:UriMapper>
8 </Application.Resources>

```

2) Указать приложению, что нужно использовать данный UriMapper. Для этого при инициализации приложения необходимо задать свойство UriMapper для объекта PhoneApplicationFrame. Понятно, что здесь можно сконструировать объект UriMapper, но поскольку он у нас уже определен в ресурсах, то возьмем его прямо оттуда:

```

1 private void InitializePhoneApplication()
2 {
3     if (phoneApplicationInitialized)
4         return;
5
6     RootFrame = new PhoneApplicationFrame();
7     RootFrame.UriMapper = (UriMapper)Resources["ApplicationUriMapper"];

```

3) Использовать в качестве адресов указанные алиасы:

```

1 NavigationService.Navigate(new Uri("<strong>/About</strong>", UriKind.Relative));

```

При определении адресов можно пользоваться шаблонами. Например, при определении UriMapper выше используется алиас "/Session/{ID}", значение параметра "ID" потом подставляется в конечный адрес. Аналогично можно использовать несколько параметров.

Используя такой несложный прием вы можете избежать проблем с путаницей адресов страниц в будущем, если потребуются изменить структуру страницы.