# Lecture 7 – Navigation

Mr. Yousif Garabet Arshak
Computer Science Department
University of Zakho
yousif.arshak@uoz.edu.krd

# Outlines

- **Hierarchical Navigation**
- **TabbedPage**
- **FlyoutPage**

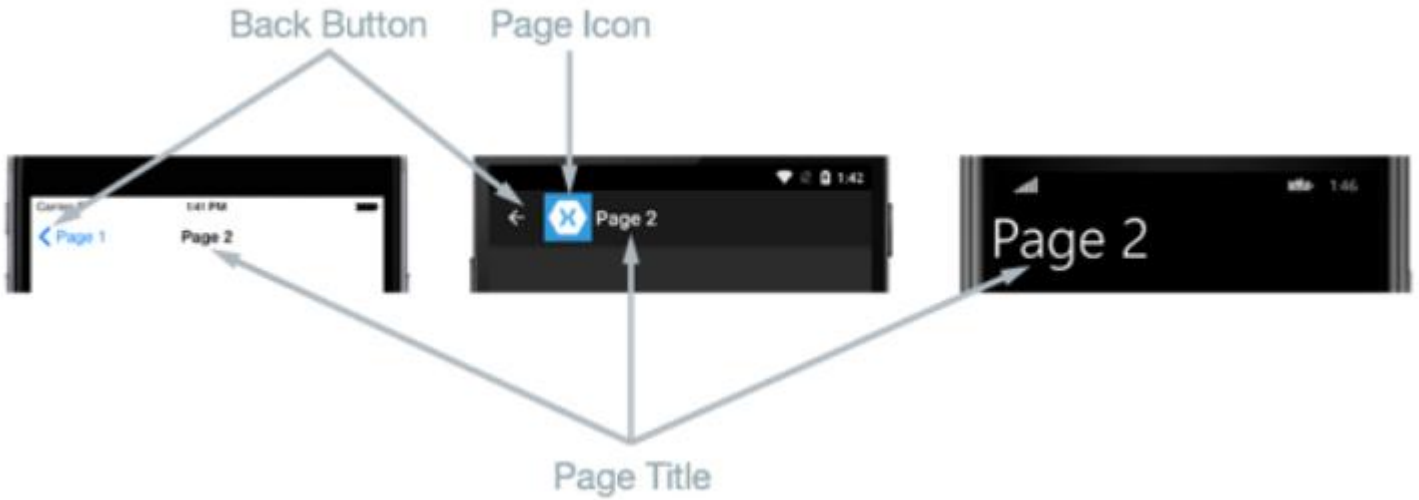
# Hierarchical Navigation

- The NavigationPage class provides a hierarchical navigation experience where the user is able to navigate through pages, forwards and backwards, as desired. The class implements navigation as a last-in, first-out (LIFO) stack of Page objects.
- To move from one page to another, an application will push a new page onto the navigation stack, where it will become the active page, as shown in the following diagram:

- To return back to the previous page, the application will pop the current page from the navigation stack, and the new topmost page becomes the active page, as shown in the following diagram:

# Performing Navigation

- In hierarchical navigation, the NavigationPage class is used to navigate through a stack of ContentPage objects. The following screenshots show the main components of the NavigationPage on each platform:



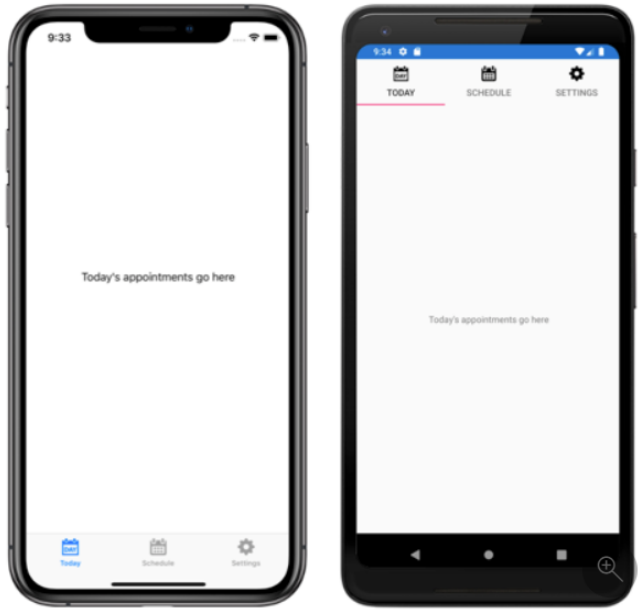
- The layout of a NavigationPage is dependent on the platform:
  - On iOS, a navigation bar is present at the top of the page that displays a title, and that has a Back button that returns to the previous page.
  - On Android, a navigation bar is present at the top of the page that displays a title, an icon, and a Back button that returns to the previous page. The icon is defined in the [Activity] attribute that decorates the MainActivity class in the Android platform-specific project.
  - On the Universal Windows Platform, a navigation bar is present at the top of the page that displays a title.
  - On all the platforms, the value of the Page.Title property will be displayed as the page title. In addition, the IconColor property can be set to a Color that's applied to the icon in the navigation bar.

**Note**

It's recommended that a NavigationPage should be populated with ContentPage instances only.

# TabbedPage

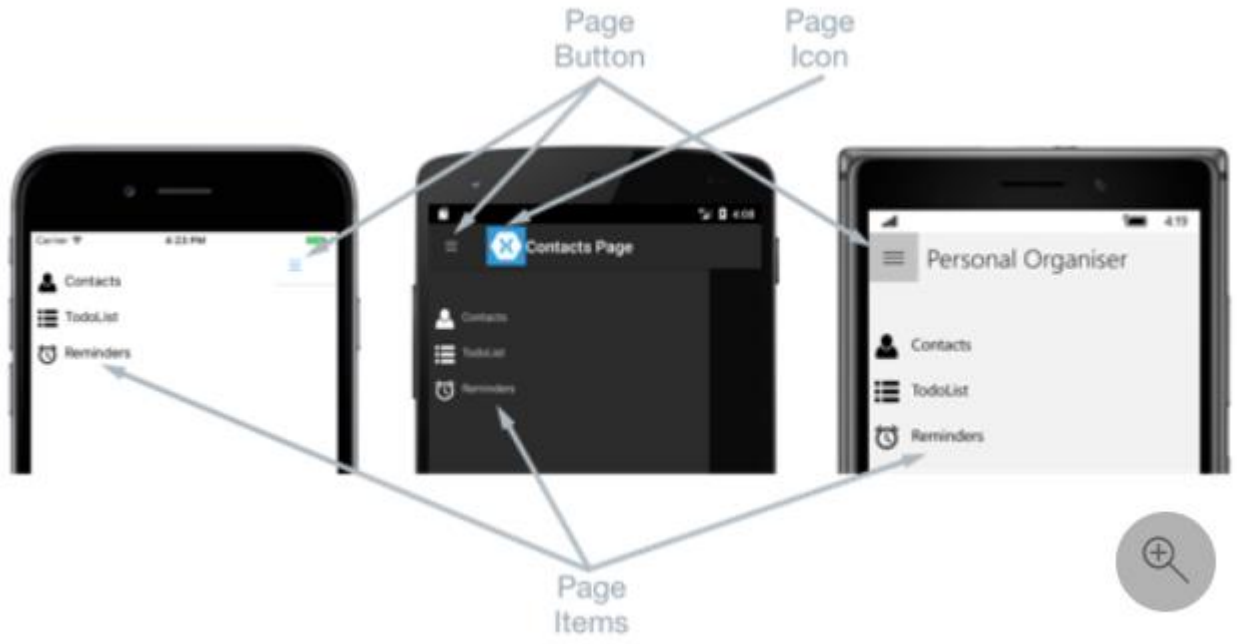
- The Xamarin.Forms TabbedPage consists of a list of tabs and a larger detail area, with each tab loading content into the detail area. The following screenshots show a TabbedPage on iOS and Android:



- On iOS, the list of tabs appears at the bottom of the screen, and the detail area is above. Each tab consists of a title and an icon, which should be a PNG file with an alpha channel. In portrait orientation, tab bar icons appear above tab titles. In landscape orientation, icons and titles appear side by side. In addition, a regular or compact tab bar may be displayed, depending on the device and orientation. If there are more than five tabs, a **More** tab will appear, which can be used to access the additional tabs.
- On Android, the list of tabs appears at the top of the screen, and the detail area is below. Each tab consists of a title and an icon, which should be a PNG file with an alpha channel. However, the tabs can be moved to the bottom of the screen with a platform-specific. If there are more than five tabs, and the tab list is at the bottom of the screen, a *More* tab will appear that can be used to access the additional tabs. For information about icon requirements, see Tabs on material.io and Support different pixel densities on developer.android.com.

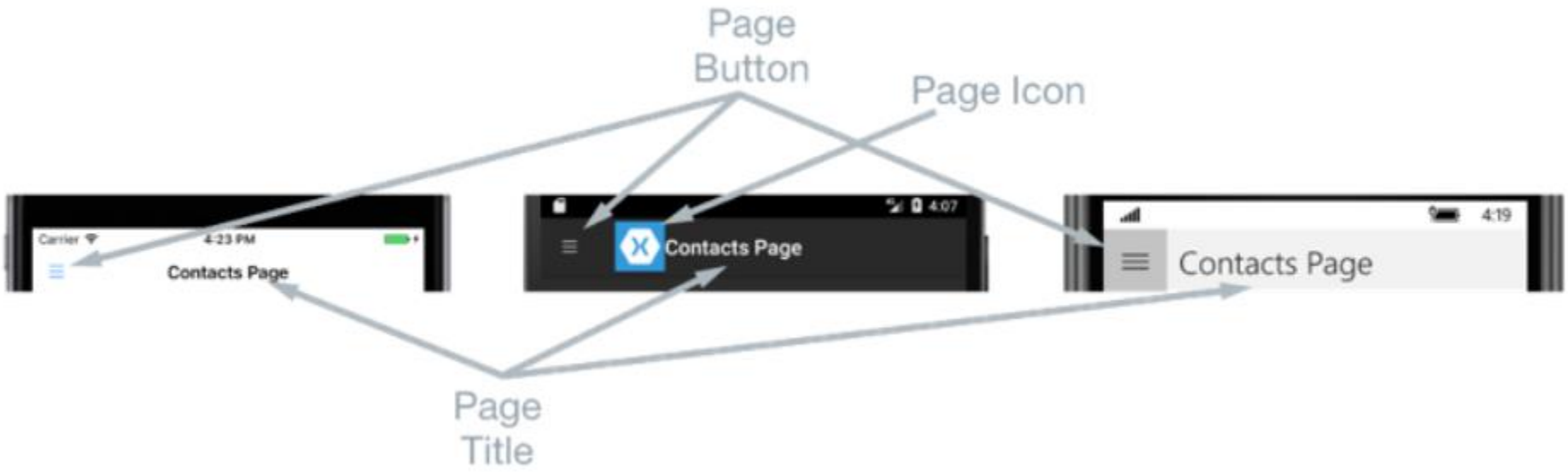
# FlyoutPage

- A flyout page typically displays a list of items, as shown in the following screenshots:



- The location of the list of items is identical on each platform, and selecting one of the items will navigate to the corresponding detail page. In addition, the flyout page also features a navigation bar that contains a button that can be used to navigate to the active detail page:
  - On iOS, the navigation bar is present at the top of the page and has a button that navigates to the detail page. In addition, the active detail page can be navigated to by swiping the flyout to the left.
  - On Android, the navigation bar is present at the top of the page and displays a title, an icon, and a button that navigates to the detail page. The icon is defined in the [Activity] attribute that decorates the MainActivity class in the Android platform-specific project. In addition, the active detail page can be navigated to by swiping the flyout page to the left, by tapping the detail page at the far right of the screen, and by tapping the Back button at the bottom of the screen.
  - On the Universal Windows Platform (UWP), the navigation bar is present at the top of the page and has a button that navigates to the detail page.

- A detail page displays data that corresponds to the item selected on the flyout page, and the main components of the detail page are shown in the following screenshots:



- The detail page contains a navigation bar, whose contents are platform-dependent:
  - On iOS, the navigation bar is present at the top of the page and displays a title, and has a button that returns to the flyout page, provided that the detail page instance is wrapped in the NavigationPage instance. In addition, the flyout page can be returned to by swiping the detail page to the right.
  - On Android, a navigation bar is present at the top of the page and displays a title, an icon, and a button that returns to the flyout page. The icon is defined in the [Activity] attribute that decorates the MainActivity class in the Android platform-specific project.
  - On UWP, the navigation bar is present at the top of the page and displays a title, and has a button that returns to the flyout page.

# Navigation behavior

The behavior of the navigation experience between flyout and detail pages is platform dependent:

- On iOS, the detail page slides to the right as the flyout page slides from the left, and the left part of the detail page is still visible.
- On Android, the detail and flyout pages are overlaid on each other.
- On UWP, the flyout page slides from the left over part of the detail page, provided that the FlyoutLayoutBehavior property is set to Popover.

# Any Questions?