Windows 8 Store Development
Part I

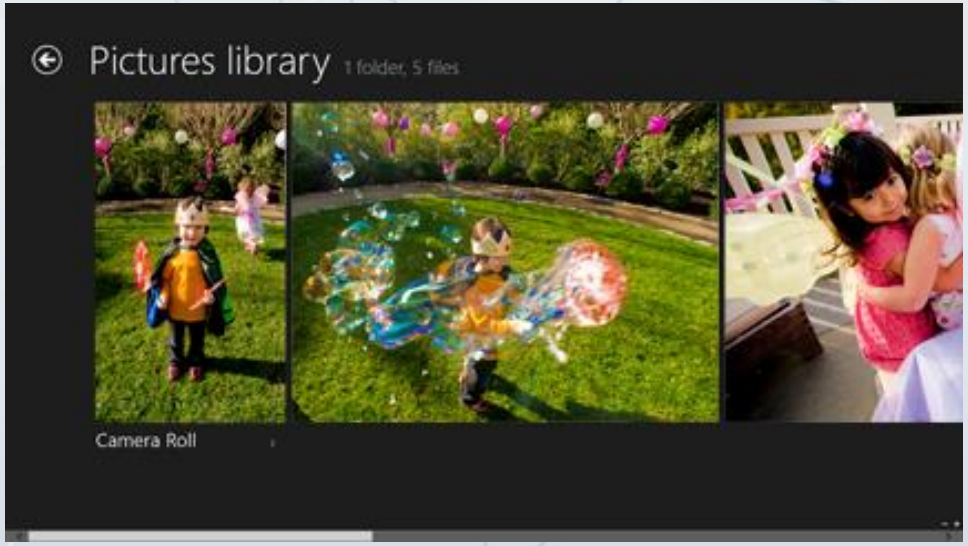
Tomer Shamam
Software Architect
CodeValue
http://www.codevalue.net
http://blogs.microsoft.co.il/blogs/tomershamam

Concept

CodeValue

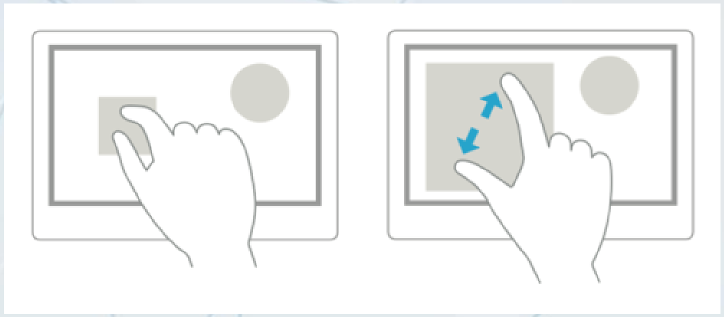
One window, multiple views

- A Windows Store app is a new type of application that runs on Windows 8 devices
- Unlike traditional desktop apps, a Windows Store app has a single, chrome-less window that fills the entire screen by default, so there are no distractions



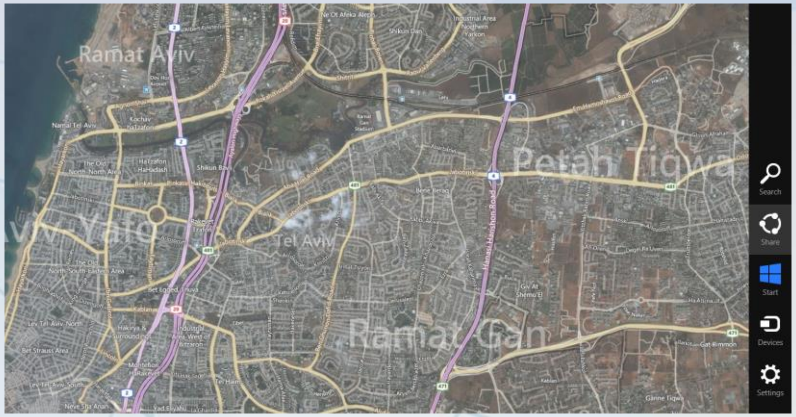
Layout, Views and Input

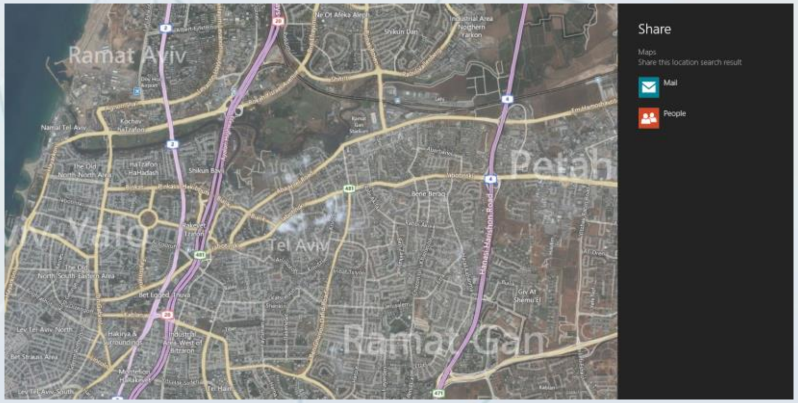
- A Windows Store app can support different layouts and views to create a fluid and harmonious experience across a variety of form factors and display sizes
- Windows Store apps work smoothly with a variety of input sources, including touch, pen, mouse, and keyboard input
- You can use a single set of events that work for all these input sources
- Windows Store apps get a set of default styles that ensure UI elements work well for touch scenarios



Windows Contracts

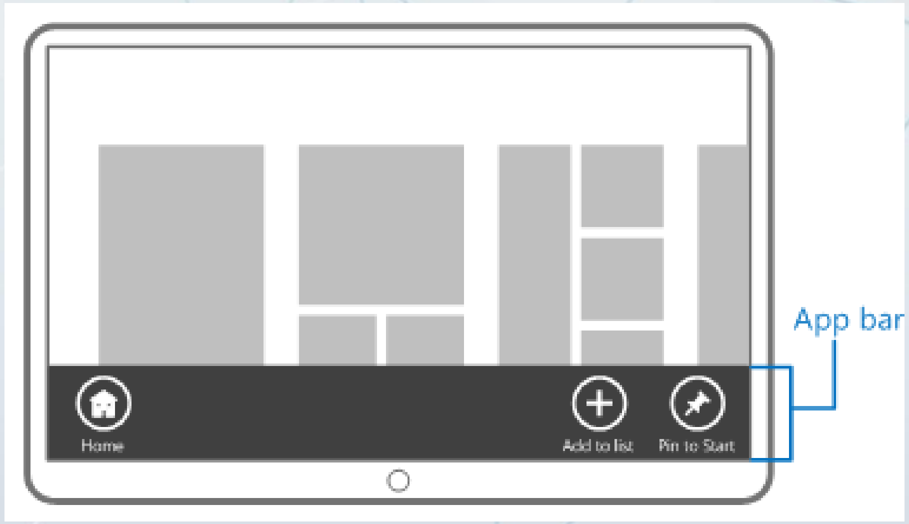
- App contracts are a way for users to seamlessly search across and share content between different apps
- They extend the usefulness of your app by eliminating the need to work with varying standards or app-specific APIs to access data stored or created by another app, all while keeping users in your branded experience
- You don't need to know anything about the target app other than its declared support for the target contract – it just works





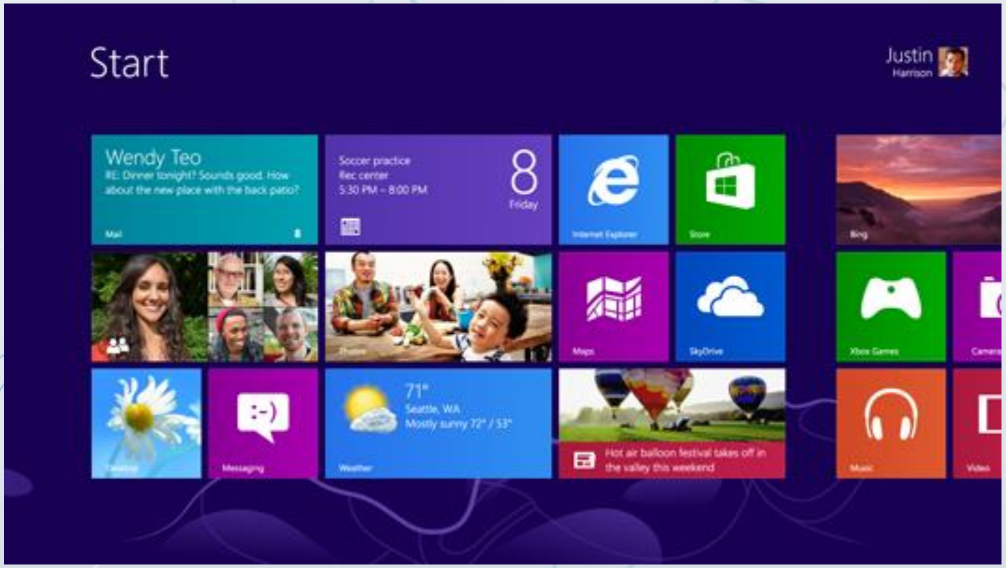
New controls and UI surfaces

- Windows Store apps provide several new controls that make it easier to create a great user experience
- Two of these controls are the app bar and the charms



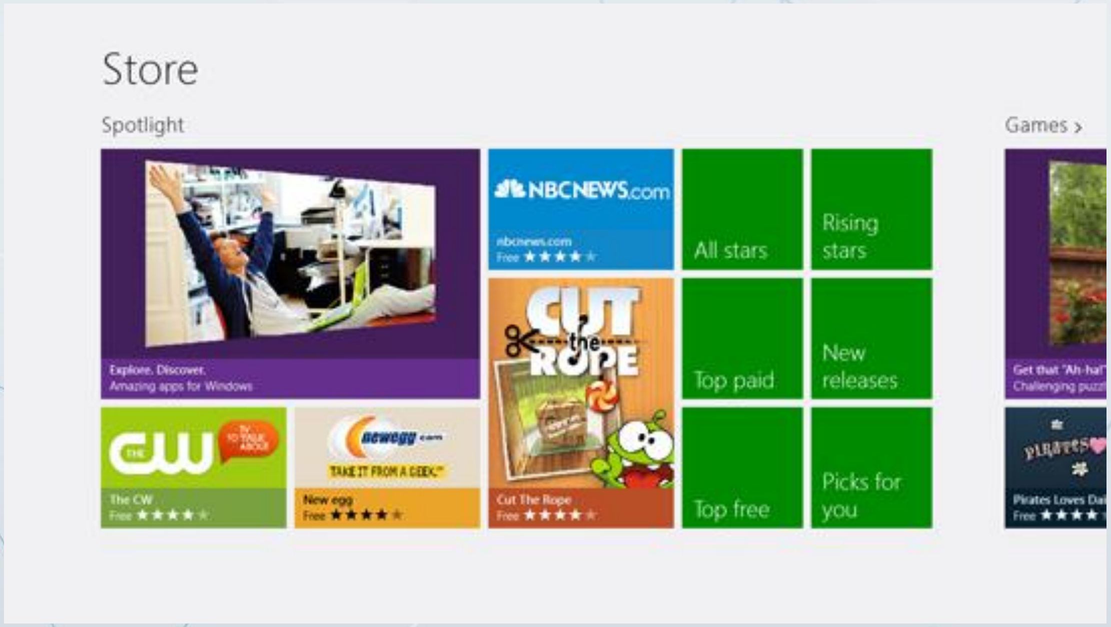
Live tiles instead of icons

- When the user installs your app, it shows up as a tile on the Start screen
- Touching or clicking the tile starts the app



The Windows Store

- The Windows Store makes your apps available to millions of customers around the world
- You write your app once, set the price in your local currency, and the Windows Store can make it available in the worldwide marketplace in 100+ languages
- The Windows Store makes it easy to distribute, update, and get paid for the apps that you develop



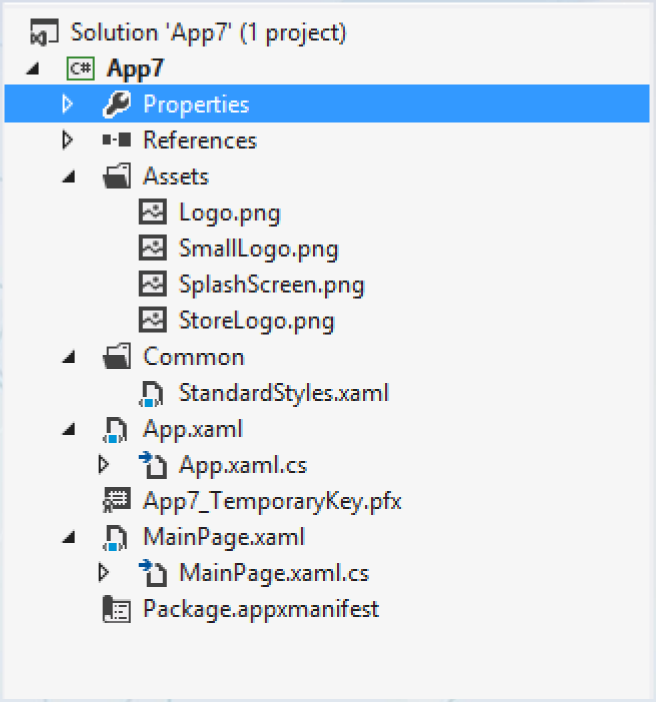
Development

Development Tools

- To develop Windows Store apps, you need to install **Windows 8** (x86 or x64) and **Visual Studio 2012**
  - You can download Visual Studio 2012 Express for free
- Get a developer license
  - Developer licenses are free
  - You can get as many as you need if you already have a Microsoft account
  - By default, developer licenses that you acquire by using a Microsoft account must be renewed every 30 days
  - The license is provided on a per-machine basis
  - After the developer license on your local machine expires, you won't be able to run uncertified apps

Application Structure

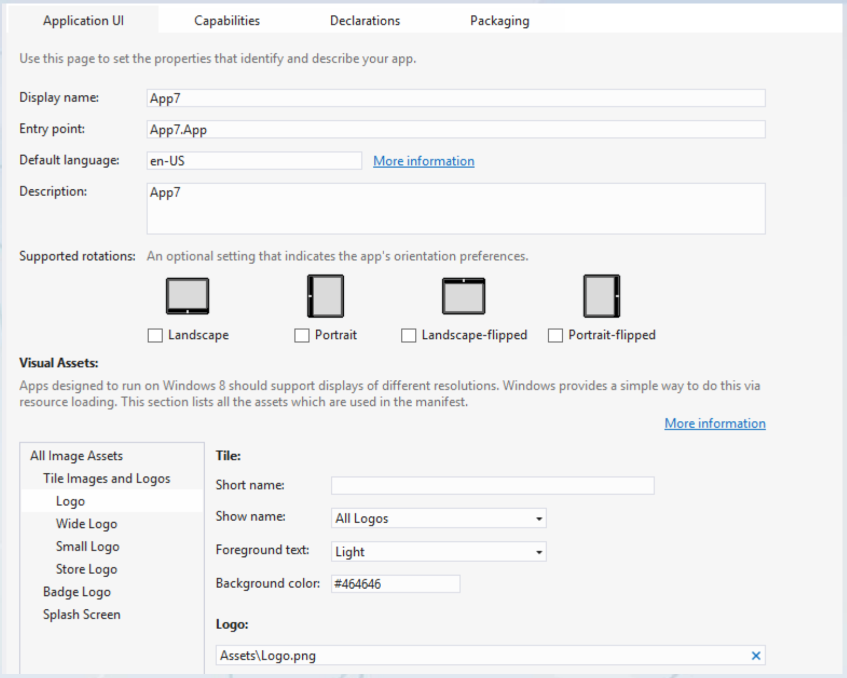
- A store C#/XAML app comprises:
  - **App.xaml** – XAML file contains application instance properties and shared XAML resources
  - **App.xaml.cs** – C# file contains definition for **App** class which represents the application
  - **MainPage.xaml** – XAML file describes the UI main page content
  - **MainPage.xaml.cs** – C# file contains interaction logic for the UI main page
  - **StandardStyles.cs** – XAML styles that simplify application development
  - **Package.appxmanifest** – XAML file describes app properties, capabilities, packaging info and more
  - **Assets** – list of logo images



```
Solution 'App7' (1 project)
  App7
    Properties
    References
    Assets
      Logo.png
      SmallLogo.png
      SplashScreen.png
      StoreLogo.png
    Common
      StandardStyles.xaml
    App.xaml
      App.xaml.cs
    App7_TemporaryKey.pfx
    MainPage.xaml
      MainPage.xaml.cs
    Package.appxmanifest
```

App manifest settings page

- Double clicking on **Package.appxmanifest** file, Visual Studio automatically opens the app manifest settings page
- You can also edit the app manifest manually opening it in XML editor
  - Right click on file, **Open With…**, **XML (Text) editor**
  - Some features can only be added directly from XML



Use this page to set the properties that identify and describe your app.

Display name:	App7
Entry point:	App7.App
Default language:	en-US [More information]
Description:	App7

Supported rotations: An optional setting that indicates the app's orientation preferences.

☐ Landscape ☐ Portrait ☐ Landscape-flipped ☐ Portrait-flipped

Visual Assets:

Apps designed to run on Windows 8 should support displays of different resolutions. Windows provides a simple way to do this via resource loading. This section lists all the assets which are used in the manifest.

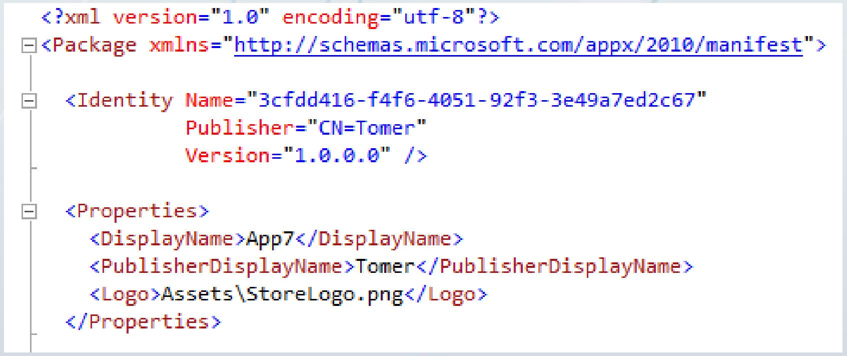
More information

Tile:

Short name:	
Show name:	All Logos
Foreground text:	Light
Background color:	#464646

Logo:

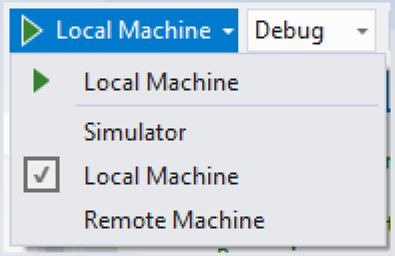
Assets\Logo.png



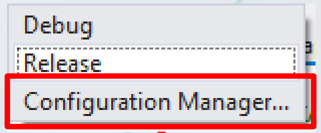
```
<?xml version="1.0" encoding="utf-8"?>
<Package xmlns="http://schemas.microsoft.com/appx/2010/manifest">

  <Identity Name="3cfdd416-f4f6-4051-92f3-3e49a7ed2c67"
            Publisher="CN=Tomer"
            Version="1.0.0.0" />

  <Properties>
    <DisplayName>App7</DisplayName>
    <PublisherDisplayName>Tomer</PublisherDisplayName>
    <Logo>Assets\StoreLogo.png</Logo>
  </Properties>
```

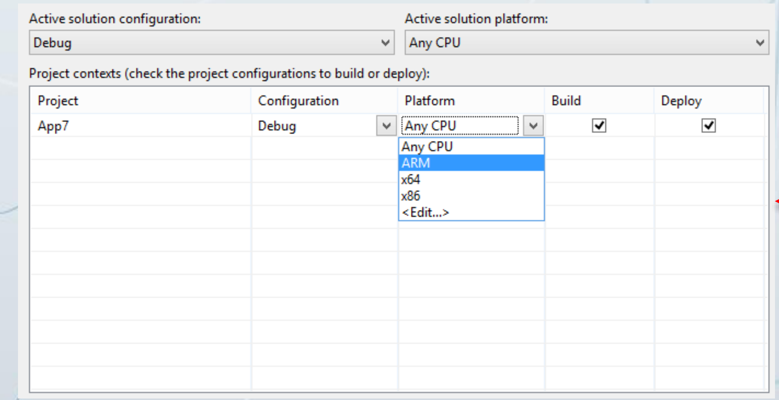
Deploying a Store app

- Select the deployment target and hit F5 (or Ctrl + F5)
  - **Local Machine** – deploy the app in the local machine. Best option with multiple monitors, will provide realistic results
  - **Simulator** – deploy the app in a Windows 8 simulator, simulating a new instance of current machine using RDP. Best for debugging Orientations, Different resolution, Touch gestures, Location and Screen captures
  - **Remote Machine** – deploy the app on a remote Windows 8 machine. Best for debugging an app on a real Tablet



- Select active configuration, and target platform: x86, x64, ARM





Active solution configuration: Debug

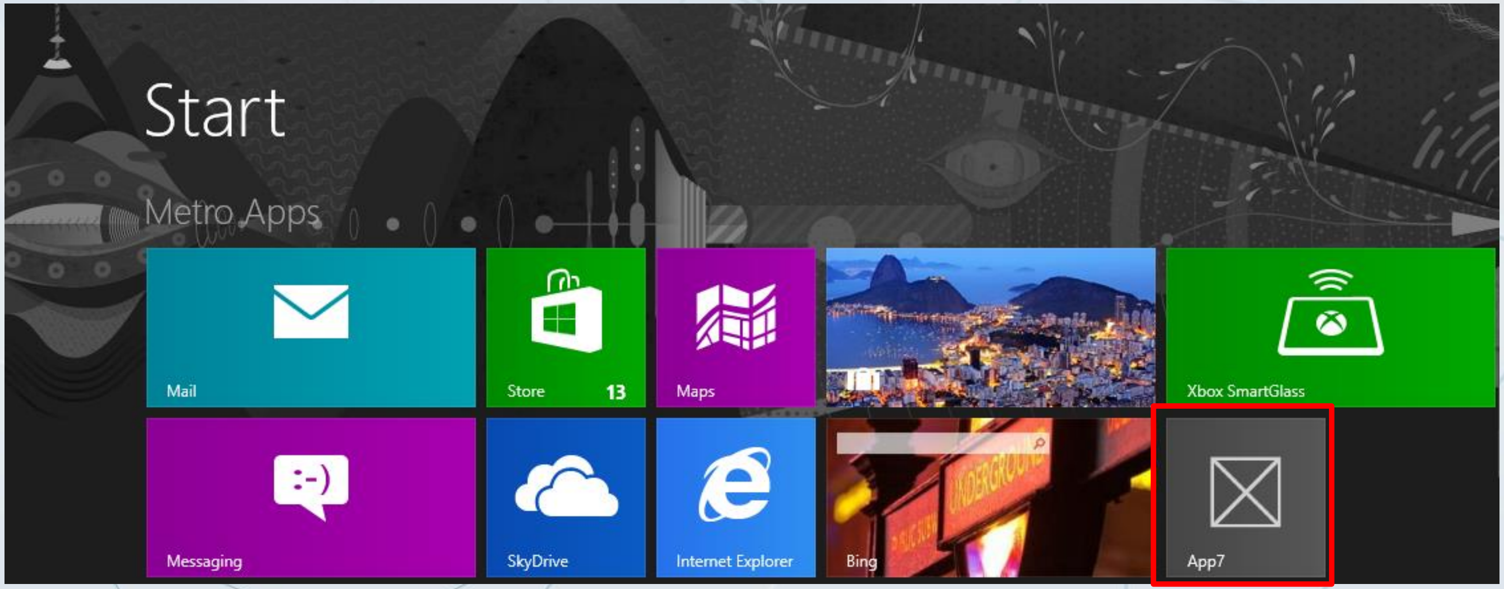
Active solution platform: Any CPU

Project contexts (check the project configurations to build or deploy):

Project	Configuration	Platform	Build	Deploy
App7	Debug	Any CPU	✓	✓

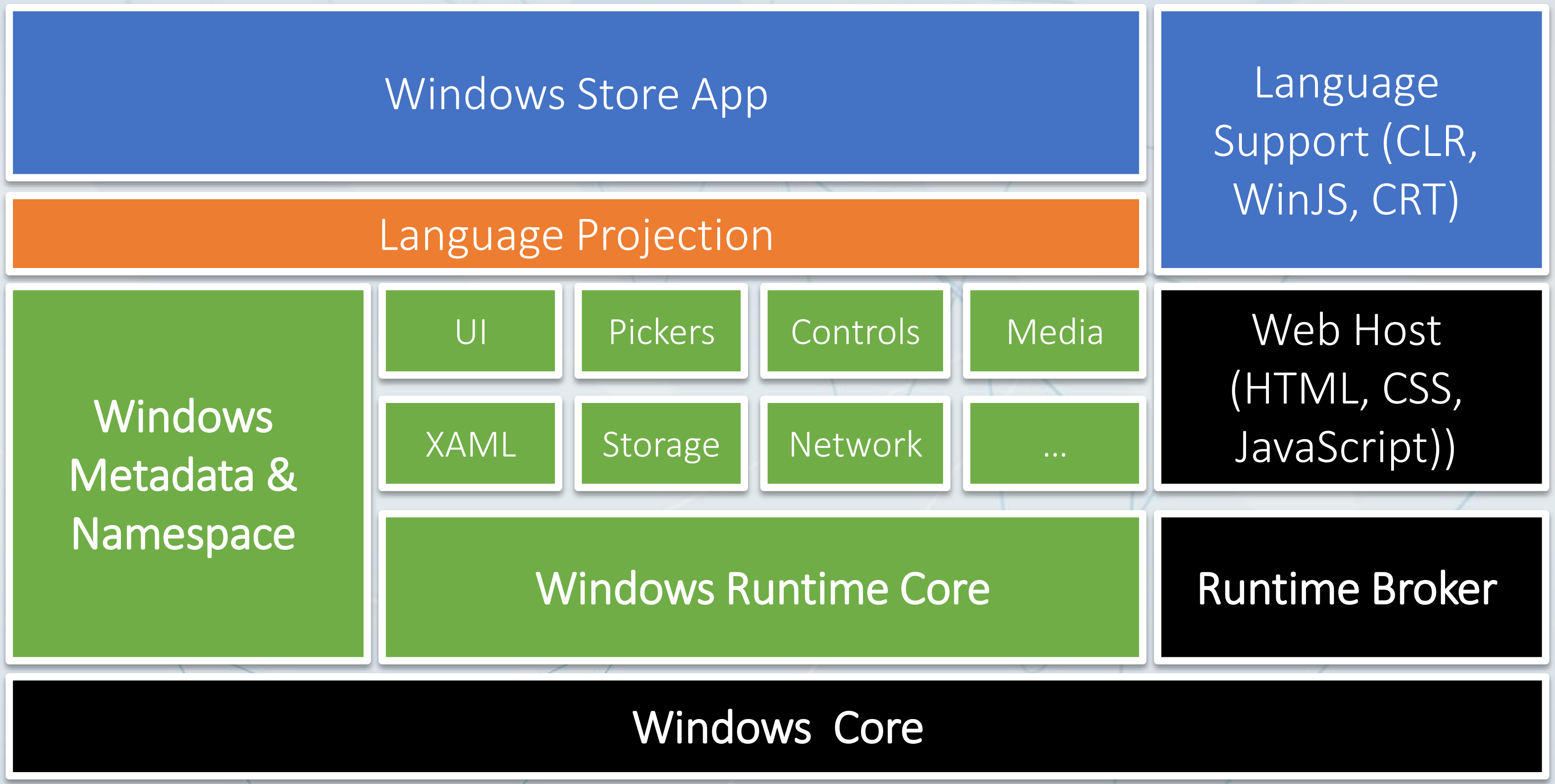
Running a Store app

- After a store app is successfully deployed, go to Start Screen and click the app Tile

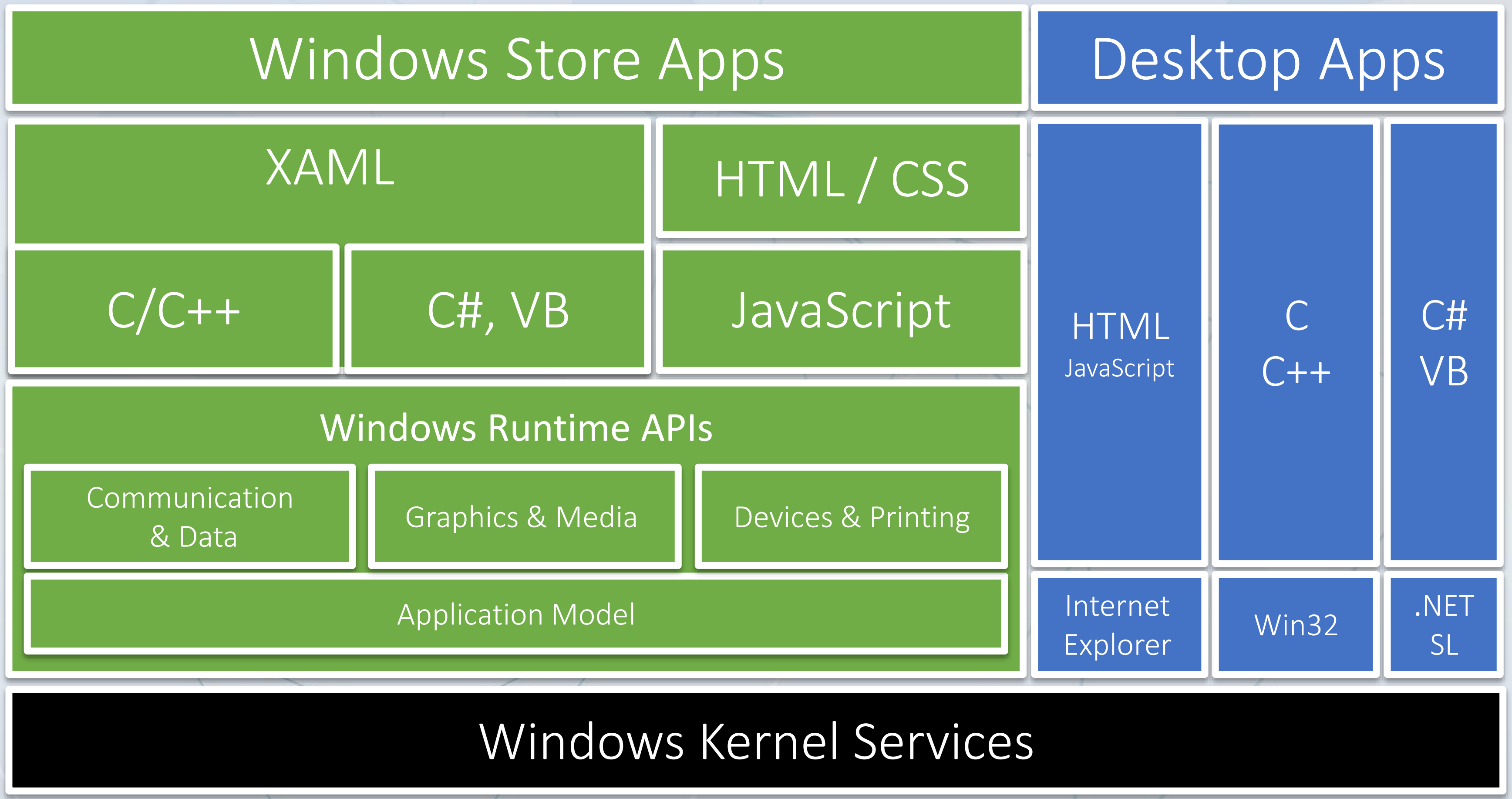


Architecture

Windows Runtime Architecture

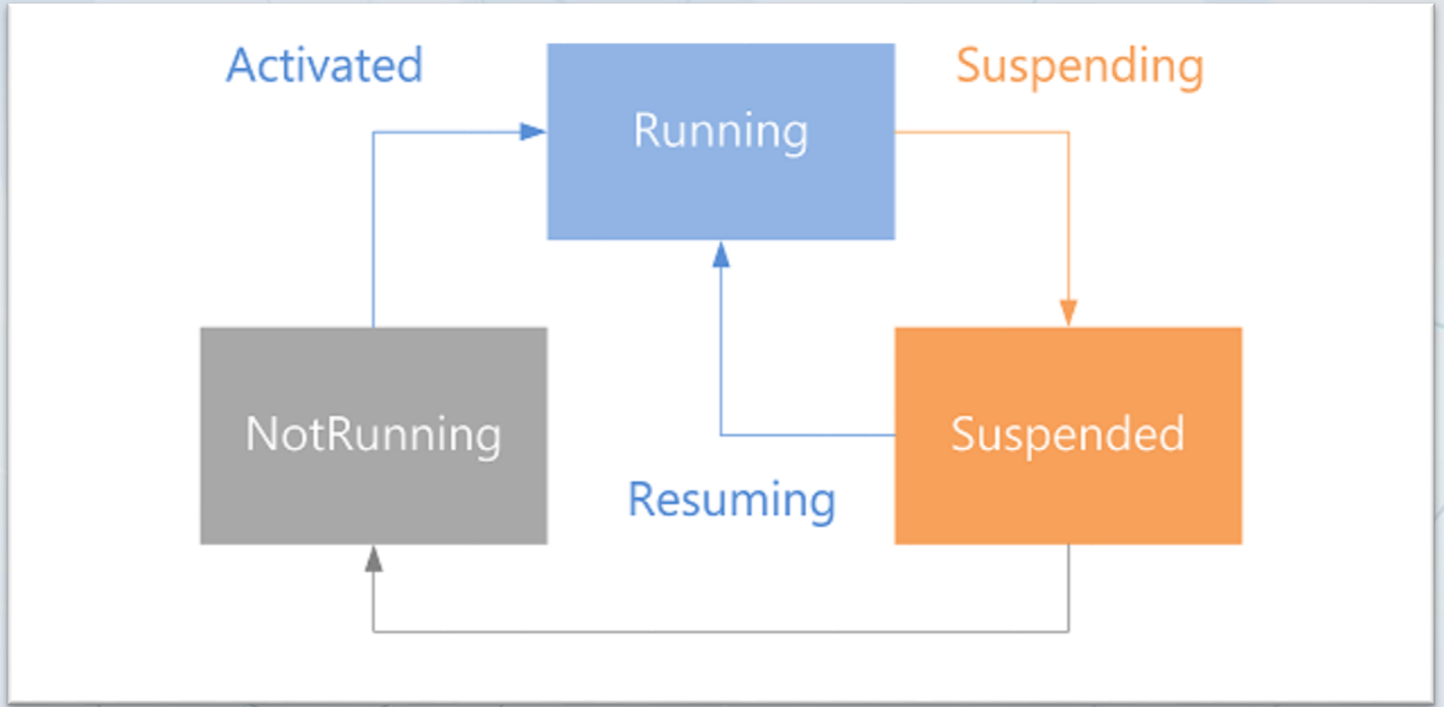


Windows 8 App Model



App Model

Application Execution State



Navigating Between in-app Pages

- Navigation between in-app pages is done using the **Frame**
- The **Frame** class provides methods, properties, and events to support navigation
- A **Page** has a property named Frame of type **Frame**
- Navigating out of page (for example directly from view-model) can be done by accessing the main frame element

Navigating to contact details page

```
private void ButtonContact_Click(object sender, RoutedEventArgs e)
{
    Frame.Navigate(typeof(ContactDetailsPage));
}
```

Handling Page Navigation Events

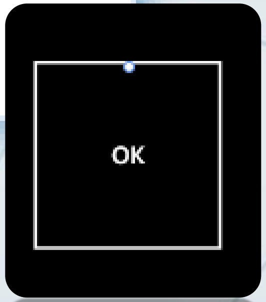
- Handling navigation events can be done in two ways:
  - Registering navigation events on the Frame, such as **Frame.Navigated**,
  - Overriding page's navigation virtual methods, such as **OnNavigatedTo**
- Handling navigation events in page level is very important for maintaining page state
- This will be discussed later in this module with much details

```
protected override void OnNavigatedTo(NavigationEventArgs e)
{
   base.OnNavigatedTo(e);
}
protected override void OnNavigatingFrom(NavigatingCancelEventArgs e)
{
    base.OnNavigatingFrom(e);
}
protected override void OnNavigatedFrom(NavigationEventArgs e)
{
    base.OnNavigatedFrom(e);
}
```

UI

XAML

```
<!--ContentPanel - place additional content here-->
<Grid x:Name="ContentPanel" Grid.Row="1"
Margin="12,0,12,0">
    <Button x:Name="buttonOk"
            Width="200"
            Height="200"
            Content="OK"
            Click="buttonOk_Click" />
</Grid>
```



Data Binding

Data model should be rendered

```
public class Lecturer
{
    public string Name { get; }
    public string Profile { get; }
    public string ImagePath { get; }
}
```

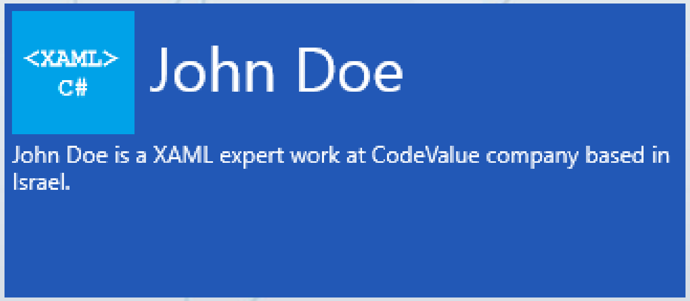
Set page’s binding-context with data model

```
public MainPage()
{
    DataContext = new Lecturer
    {
        Name = "John Doe",
        Profile = "John Doe is a XAML
expert work at CodeValue company based in
Israel.",
        ImagePath = "Images/XAML.png"
    };
}
```

Binding of UI elements properties to data model properties

```
<Grid Background="#FF2258B6">
    ...
    <TextBlock Text="{Binding Name}“ ... />
    <TextBlock Text="{Binding Profile}" ... />
    <Image Source="{Binding ImagePath}" ... />
</Grid>
```

Data rendering result



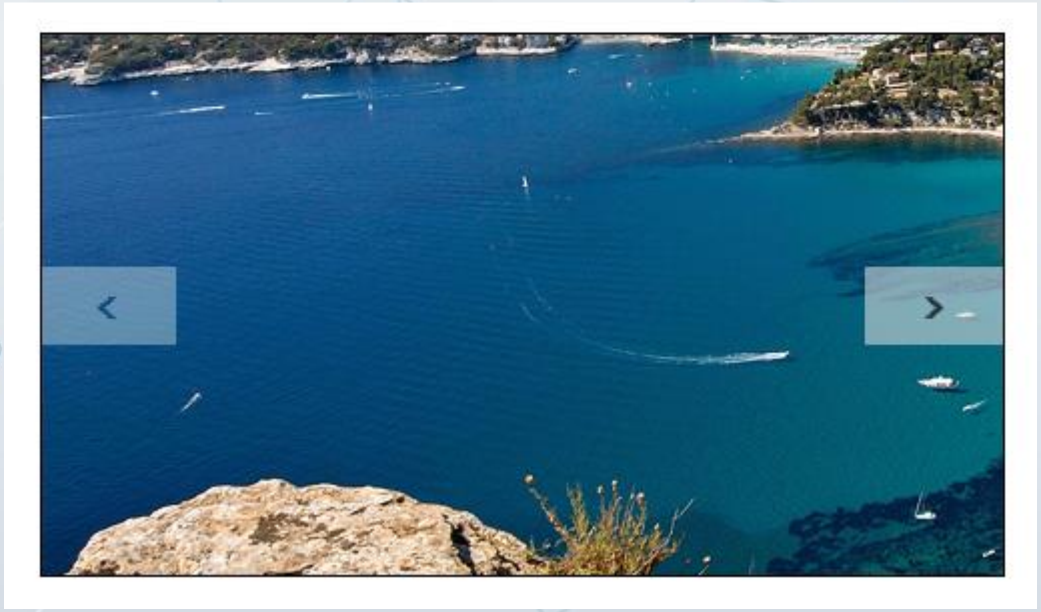
John Doe

John Doe is a XAML expert work at CodeValue company based in Israel.

Flip view

```
<FlipView x:Name="flipView1"
SelectionChanged="FlipView_SelectionChanged">
    <Image Source="Assets/Logo.png" />
    <Image Source="Assets/SplashScreen.png" />
    <Image Source="Assets/SmallLogo.png" />
</FlipView>
```

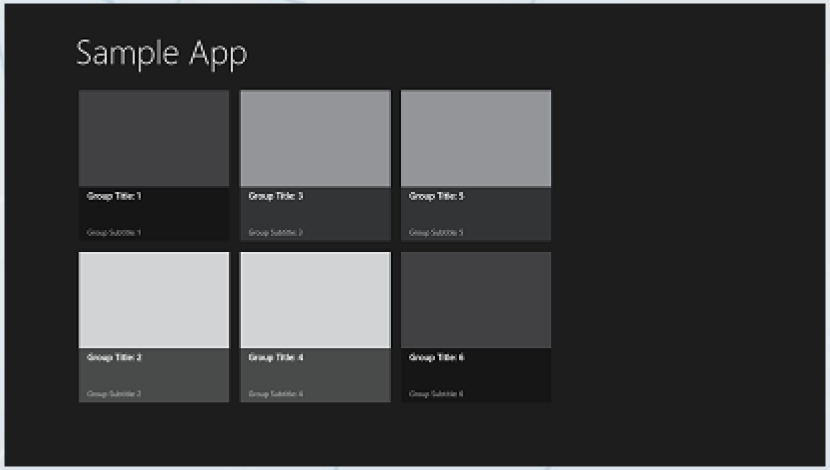
Flip view
A control that presents a collection of items that the user can flip through, one item at a time.



Grid view

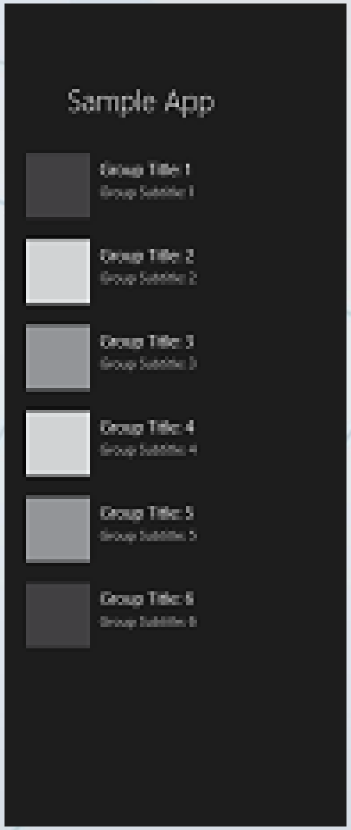
```
<GridView x:Name="gridView1"
SelectionChanged="GridView_SelectionChanged">
    <x:String>Item 1</x:String>
    <x:String>Item 2</x:String>
</GridView>
```

Grid view
A control that presents a collection of items in rows and columns that can scroll horizontally.



List view

```
 <ListView x:Name="listView1"
SelectionChanged="ListView_SelectionChanged">
           <x:String>Item 1</x:String>
           <x:String>Item 2</x:String>
 </ListView>
```

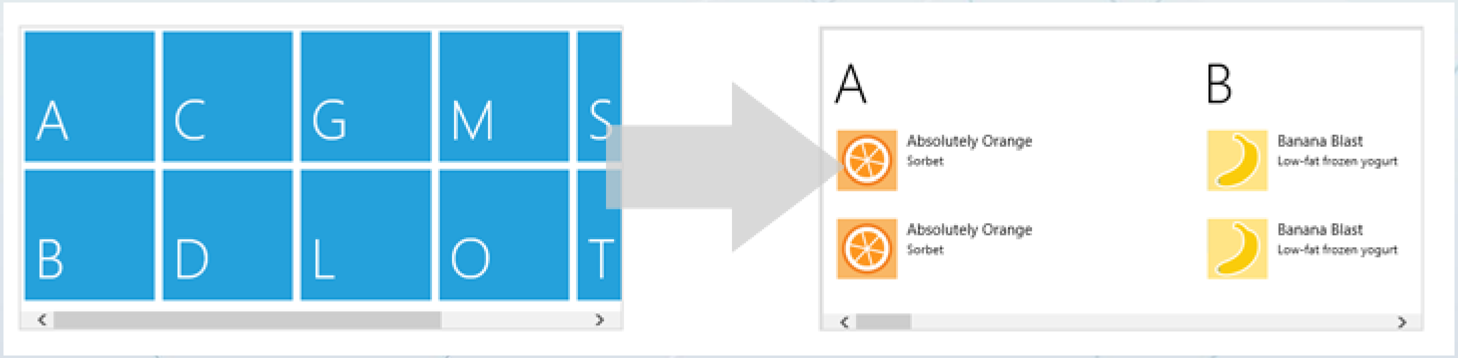


List view
A control that presents a collection of items in a list that can scroll vertically.

Sematic zoom

```
<SemanticZoom>
    <ZoomedInView>
        <GridView></GridView>
    </ZoomedInView>
    <ZoomedOutView>
        <GridView></GridView>
    </ZoomedOutView>
</SemanticZoom>
```

Semantic zoom
A container control that lets the user zoom between two views of a collection of items.



Summary

- Developing Windows Store apps, you must have Windows 8 and Visual Studio 2012
- Get a developer license!
- Choose your programming language: C#, VB, C++, JS
- Visual Studio 2012 provides everything you need in one place: Development, Debugging, Testing, Analyzing, Deployment
- Windows Store app comprises at least one App and one Page
- Configure app characteristics and capabilities using app manifest

Thank You

CodeValue