Resources & Internationalization

Outline

- Resource hierarchy
- Binary resources
- Internationalization

Reusable Resources

- Templates
- Graphics
- Complex brushes
- Any object

```
<Page.Resources>
  <SolidColorBrush x:Key="PluralsightGreen"
                   Color="#63b941" />
</Page.Resources>

...
  <GeometryDrawing
     Brush="{StaticResource PluralsightGreen}">
...
```

Defining Resources

```
<Page.Resources>

  <Color x:Key="PsGreen">#63b941</Color>

  <SolidColorBrush x:Key="PsGreenBrush"
    Color="{StaticResource PsGreen}" />

  <ControlTemplate x:Key="bt" TargetType="{x:Type Button}">
    <Ellipse Fill="Red" />
  </ControlTemplate>

  <ObjectDataProvider x:Key="ds" />

  <sys:String x:Key="hw">Hello, world</sys:String>

  <x:Array x:Key="foobar" Type="{x:Type sys:String}">
    <sys:String>Foo</sys:String>
    <sys:String>Bar</sys:String>
  </x:Array>

  <Button x:Key="btn" /> <!-- Danger! -->

</Page.Resources>
```

ResourceDictionary

- **Resources property:**
  - FrameworkElement
  - FrameworkContentElement
  - Application
  - Style
  - FrameworkTemplate

```
public ResourceDictionary Resources
{ get; set; }
```

- **Implements IDictionary**
  - Key and value both of type Object

- **Parses each item on demand**

Resource References

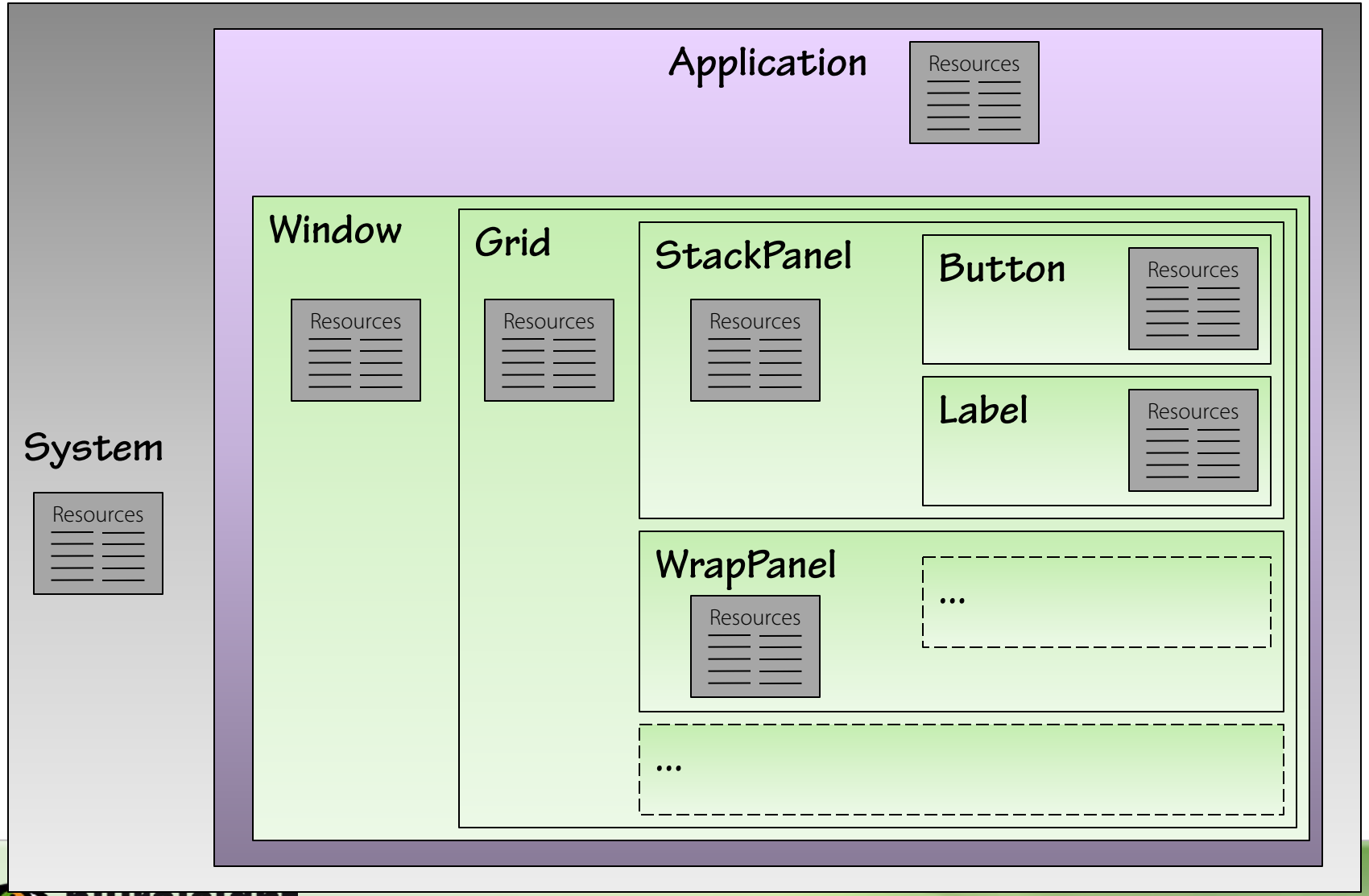
- **Markup extensions**
  - StaticResource
  - DynamicResource

```
<SomeElement Prop="{StaticResource Name}"/>

<SomeElement Prop="{DynamicResource Name}"/>
```

- **Object reference**
  - Not copy

```
// StaticResource:
elem.Prop = FindResource("Name");

// DynamicResource:
elem.SetResourceReference(
  SomeElement.PropProperty,
  "Name");
```

Resource Hierarchy



Alternate Reference Syntax

```
<MyElem>
  <MyElem.Prop>
    <StaticResource ResourceKey="Name" />
  </MyElem.Prop>

  <MyElem.OtherProp>
    <DynamicResource ResourceKey="Name" />
  </MyElem.OtherProp>

  <StaticResource ResourceKey="Name" />

</MyElem>
```

Implicit Resource Usage

- DataTemplates
- Styles

```
<Page.Resources>
  <DataTemplate DataType="{x:Type m:MyData}">
    <Ellipse Fill="Red" />
  </DataTemplate>

  <Style TargetType="{x:Type Button}">
    ...
  </Style>

  <!-- ...or... -->

  <Style x:Key="{x:Type Button}">
    ...
  </Style>
</Page.Resources>
```

Merging Resources

```
public class ResourceDictionary
{
  ...
  public Collection<ResourceDictionary> MergedDictionaries
  { get; }
  ...
}
```

```
<Application.Resources>
  <ResourceDictionary>
    <ResourceDictionary.MergedDictionaries>
      <ResourceDictionary Source="ResourceDictionary1.xaml"/>
      <ResourceDictionary Source="ResourceDictionary2.xaml"/>
    </ResourceDictionary.MergedDictionaries>

    <SolidColorBrush x:Key="myRed" Color="Red" Opacity="0.5"/>

  </ResourceDictionary>
</Application.Resources>
```

Binary Resources

- External (e.g. URL)
- 'Loose'
- Embedded

Preferred

Build Action	Resource Type
Content	Loose
Resource	ResourceManager resource
Embedded Resource	Assembly Manifest Resource
<PropertyGroup> <Win32Resource>...	Win32 resource

Example: ImageSource

- **Absolute URL**

```
<Image Source="http://www.pluralsight.com/images/logo.gif" />
```

- **Relative URL**
  - For Resource or Content

```
<Image Source="Flower.jpg" />
```

```
<Image Source="Images\Flower.jpg" />
```

Application Class Resource Methods

- **GetResourceStream**
  - ResourceManager style embedded resource
- **GetContentStream**
  - Loose resources
- **GetRemoteStream**
  - Non-local resources
- **LoadComponent**
  - Load XAML or BAML embedded resource as object

Themes

- **Compiled Xaml resources in Themes folder:**
  - generic.xaml
  - Aero.NormalColor.xaml
  - Luna.NormalColor.xaml
  - Luna.Homestead.xaml
  - Luna.Metallic.xaml
  - Classic.xaml
  - Royale.NormalColor.xaml

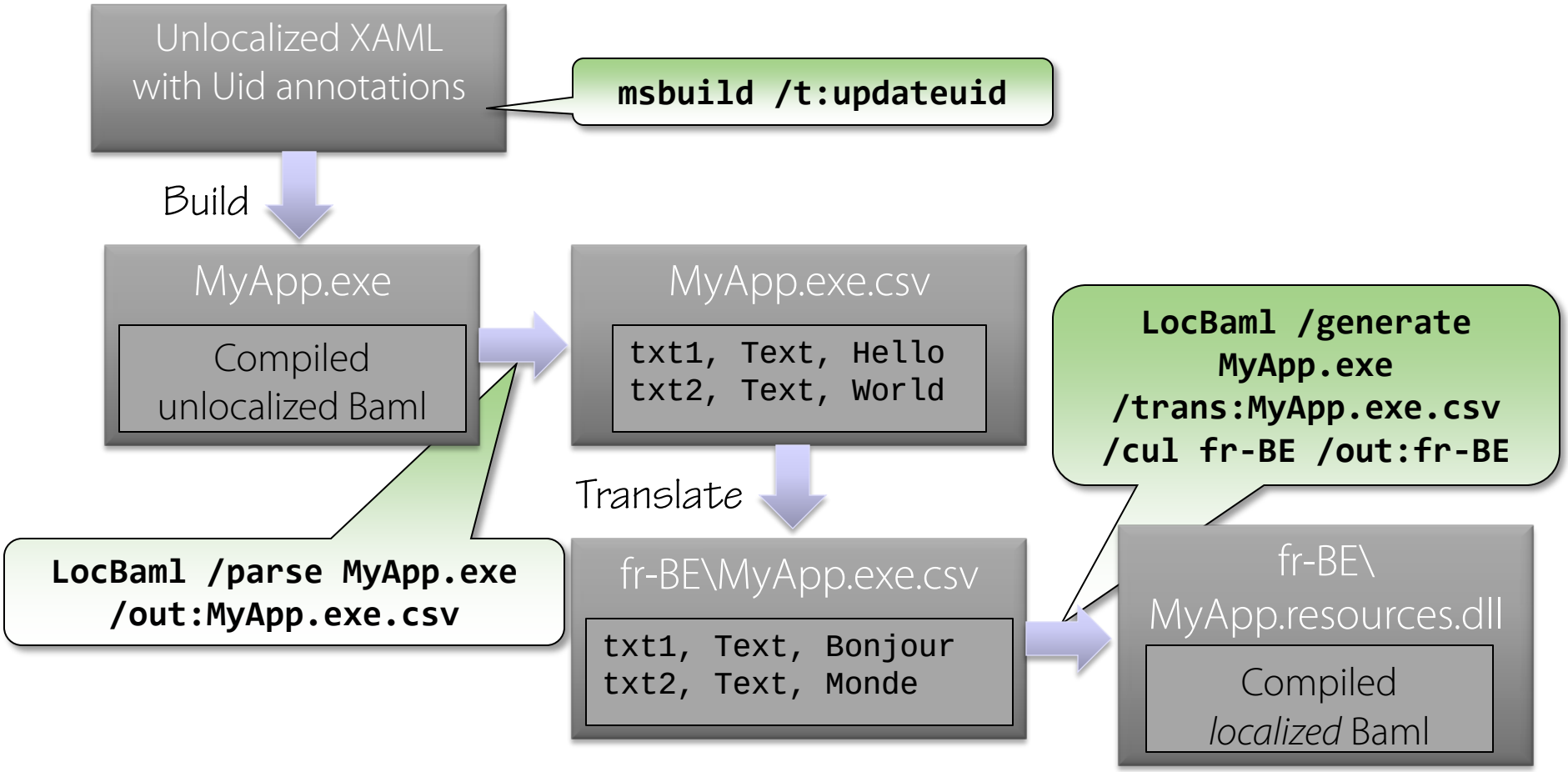
Internationalization

- ResourceManager and satellite assemblies
- WPF automatic layout can simplify process

XAML and Localization

- **Usually one XAML file per language**
- **Localization means creating multiple variants each XAML UI**
  - Variants share codebehind
  - Structure constant across variants
- **Generation of variants can be part of build process**
  - Allows changes to original XAML after localization

Localization Workflow



LocBaml

- **Some assembly required**
- **Source for LocBaml provided in SDK**
  - Proper tool likely in future
  - Tool not supported, but based on supported APIs

Xaml, XML, and Language

- **Supported encodings for Xaml:**
  - ASCII
  - UTF-8 (default)
  - UTF-16

- **xml:lang attribute supported**
  - Inherited by nested elements
  - Influences ideograph shapes & fonts for far eastern languages
  - Controls spell checking in text edit controls

Summary

- Resource hierarchy
- Binary resources
- Internationalization