Using Resources and Styling Controls

Kevin Dockx
http://blog.kevindockx.com/
@KevinDockx

pluralsight
hardcore dev and IT training

Introduction to Resources

Any object can be defined as a resource, effectively making it a reusable asset

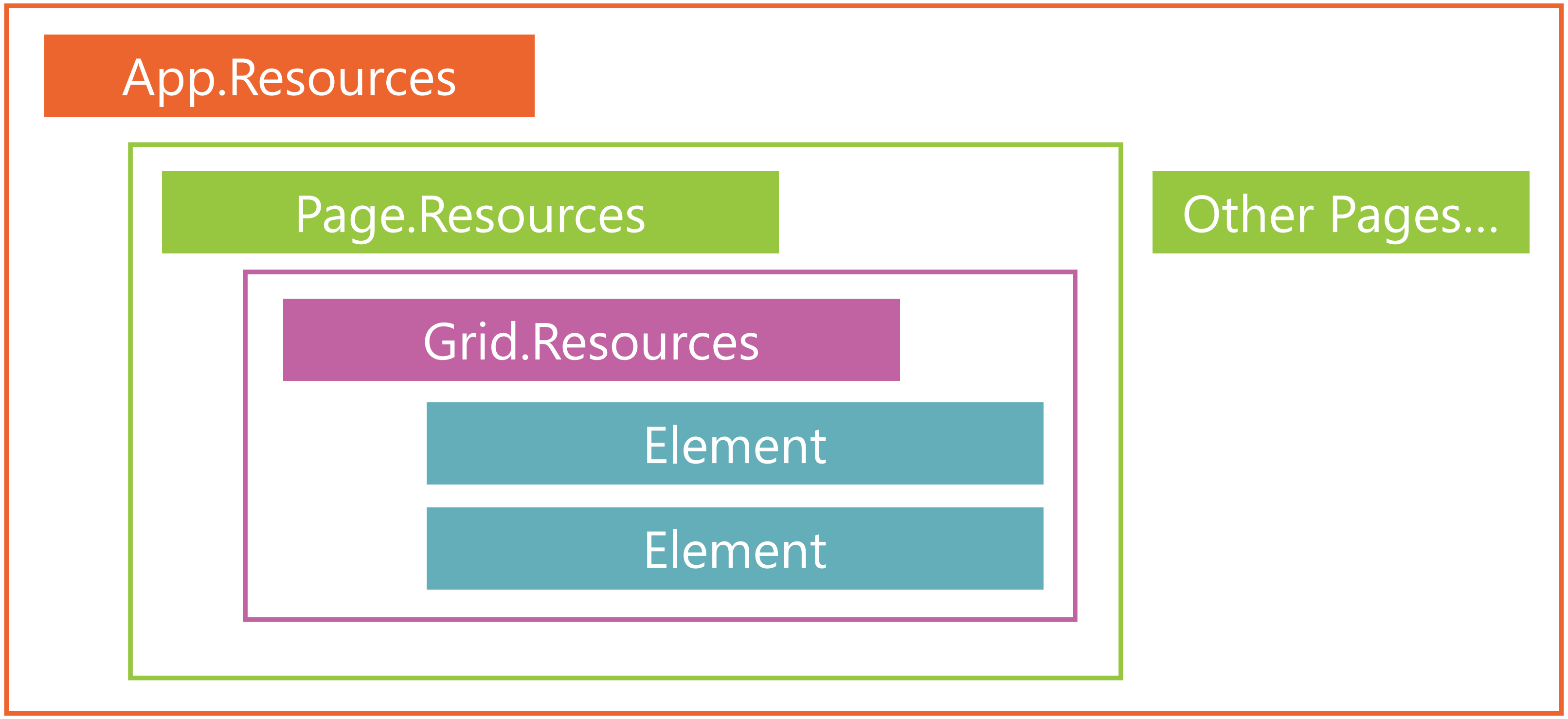
- **Often-used are (Data)Templates, Brushes and Styles**

On resources…

- They should be given a **key** (x:Key) to access them
- They are defined in a **Resource Dictionary**
- On Framework(Content)Element, the **Resources** property (of type ResourceDictionary) is defined
- The **StaticResource** markup extension is used to access the resource

Resource Scope

The scope of a resource depends on the resource dictionary it's defined in



Resource Dictionaries

More often than not, when you hear someone refer to a Resource Dictionary, they mean through separate files

Using separate files offers advantages

- Better **separation**
- More **control**
- More **reuse**

.xaml document starting with ResourceDictionary tag

Scope & usage

- Add the dictionary to the **Resources** tag where you want it scoped
- … or merge multiple resource dictionaries through **ResourceDictionary.MergedDictionaries** syntax

Styles

A Style is a convenient way to apply a set of property values to more than one element

On styles

- Styles are defined in a **Resource Dictionary**
- Through **TargetType**, we specify the type of target for the Style (eg: TextBlock, Button, …)
- Every Style must have a key (**x:Key**)
- Styles are assigned through **StaticResource**

Implicit styling

- Implicit styling allows us to specify a style that will be **applied to all elements of a specific type**
- Through **TargetType**, we specify the type of target for the Style (eg: TextBlock, Button, …)
- We **don't set a key**: this is implicit
- We **no longer have to assign the Style** explicitly

Style Inheritance

When most property values in different style sets are the same, we can inherit a base style to base the more specific styles on

Inheritance

- Define a base Style, and give it a **key**
- Define an inheriting Style, and set the **BasedOn** property to that Style through **StaticResource**
- Inheritance depth is not limited, but multiple inheritance isn't allowed

Summary

- **Working with Resources**
- **Resource Scoping and Resource Dictionaries**
- **Styles**
  - Implicit styling
  - Style inheritance