Extending and Configuring Razor

Alex Wolf

www.alexwolfthoughts.com

To-Do List

- Getting Started with Custom Tag Helpers
- Demo: A Simple Custom Tag Helper
- Demo: Custom Elements with Tag Helpers
- Demo: Building Custom HTML Helpers
- A Closer Look at View Engines
- Demo: Customizing View Location Search Patterns
- Demo: Touring the View Rendering Process

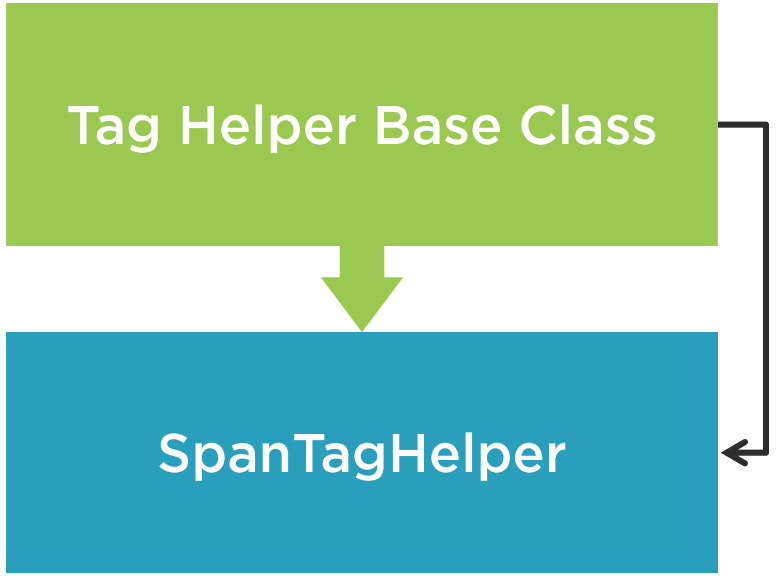
Getting Started with Custom Tag Helpers

```
<div class="form-group">
    <span last-updated="Today">
</div>


<div class="form-group">
    <progressbar />
</div>
```

- Tag Helper applied to an existing span element with an additional data attribute

- Entirely custom HTML element Tag Helper

Tag Helper Inheritance



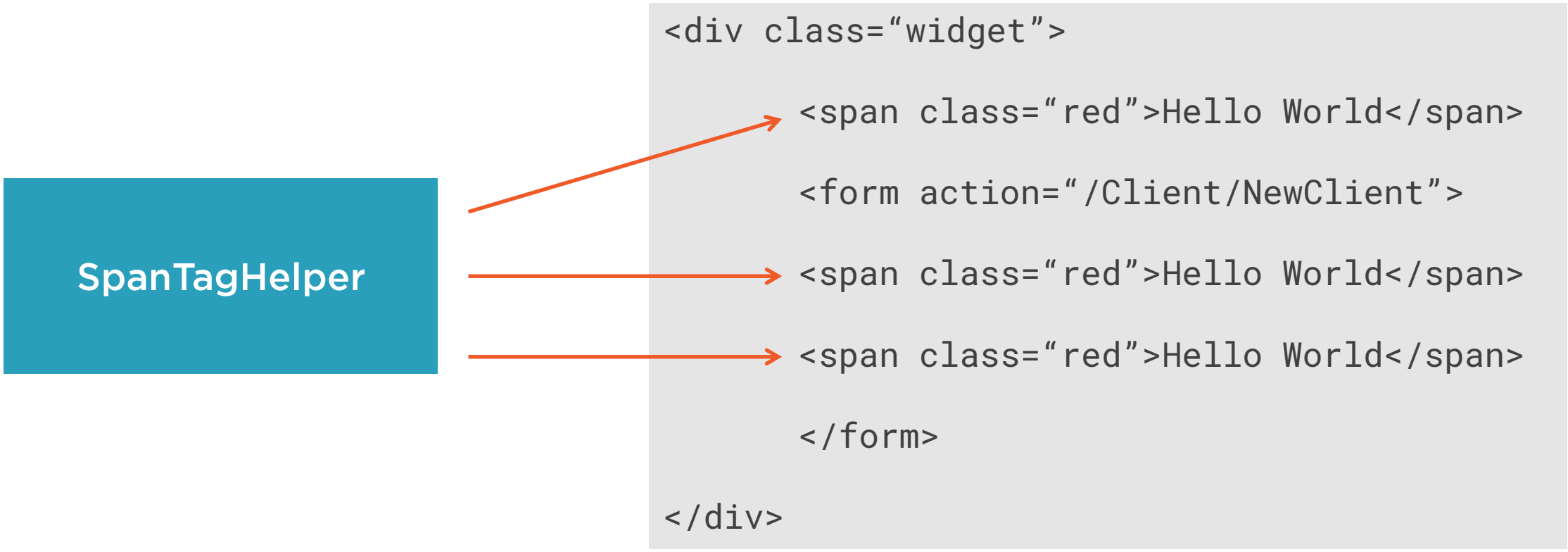
Framework Integration

Helper Properties & Methods

```
public class SpanTagHelper : TagHelper
{


    public string LastUpdated
    { get; set;}


    public override void Process
    (TagHelperContext context,
    TagHelperOutput output)
    {
        // Tag Helper logic here
    }

}
```

- Conventional naming and inheritance
- Property that will map to an HTML attribute for passing in data
- Process method of the Tag Helper to execute logic

Understanding Tag Helper Scope



```
[HtmlTargetElement("span", ParentTag = "form")]
public class RedSpanTagHelper : TagHelper
{
    // Class contents
}
```

Controlling Tag Helper Scope

The HtmlTargetElement attribute allows you to influence Tag Helper scope through various parameters

Influencing Tag Helper Scope

[HtmlTargetElement
(ParentTag = “Form”)]
SpanTagHelper

```
<div class=“widget”>
        <span>Hello World</span>
        <form action=“/Client/NewClient”>
        <span class=“red”>Hello World</span>
        <span class=“red”>Hello World</span>
        </form>
</div>
```

```
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
@addTagHelper *, OrderingApplication
```

Registering Tag Helpers

Tag Helpers must be registered in the ViewImports.cshtml file

Visual Studio provides intellisense for custom Tag Helpers

Demo

Building a custom Tag Helper

Registering custom Tag Helpers and using them inside Views

Demo

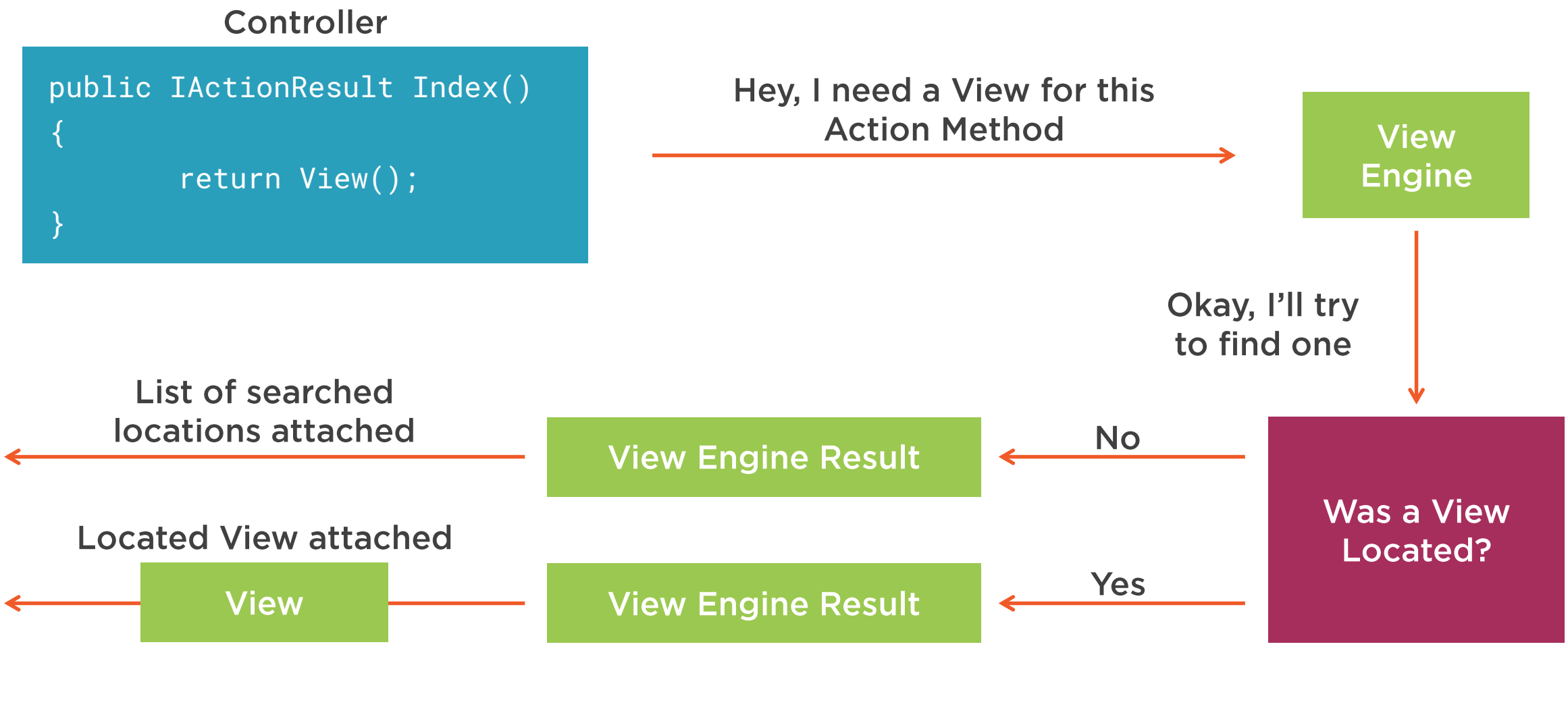
Creating a custom HTML Element Tag Helper

Demo

Building Custom HTML Helpers

A Closer Look at View Engines

View Engine Components



```
public interface IViewEngine
{
    ViewEngineResult FindView(ActionContext context, string viewName,
    bool isMainPage);

    ViewEngineResult GetView(string executingFilePath, string viewPath,
    bool isMainPage);
}
```

The IViewEngine Interface

Every View Engine in MVC must implement this interface

Primary responsibility is to locate Views

```
public class ViewEngineResult

{

    public IEnumerable<string>
    SearchedLocations { get; }

    public bool Success { get; }

    public IView View { get; }

    public string ViewName { get; }



    public static ViewEngineResult Found
    (string viewName, IView view);

    public static ViewEngineResult  NotFound
    (string viewName, IEnumerable<string>
    searchedLocations);

}
```

- Properties used to communicate the search results

- Attaches the located View

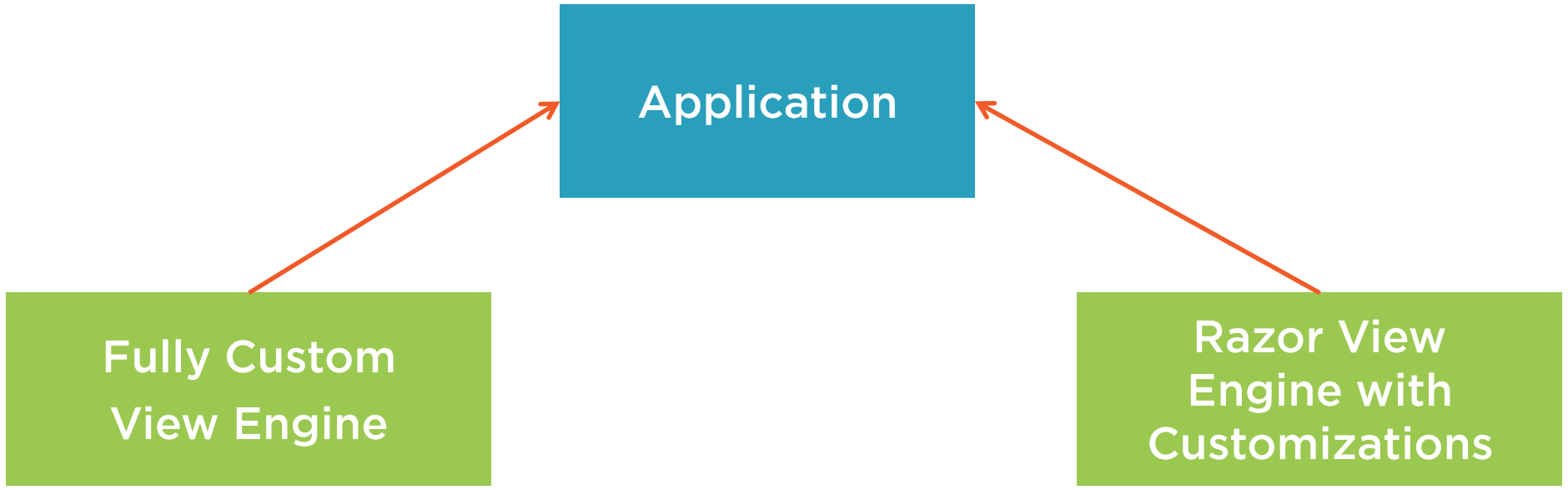
- Attaches the list of searched locations

```
public interface IView
{
    string Path { get; }
    Task RenderAsync(ViewContext context);
}
```

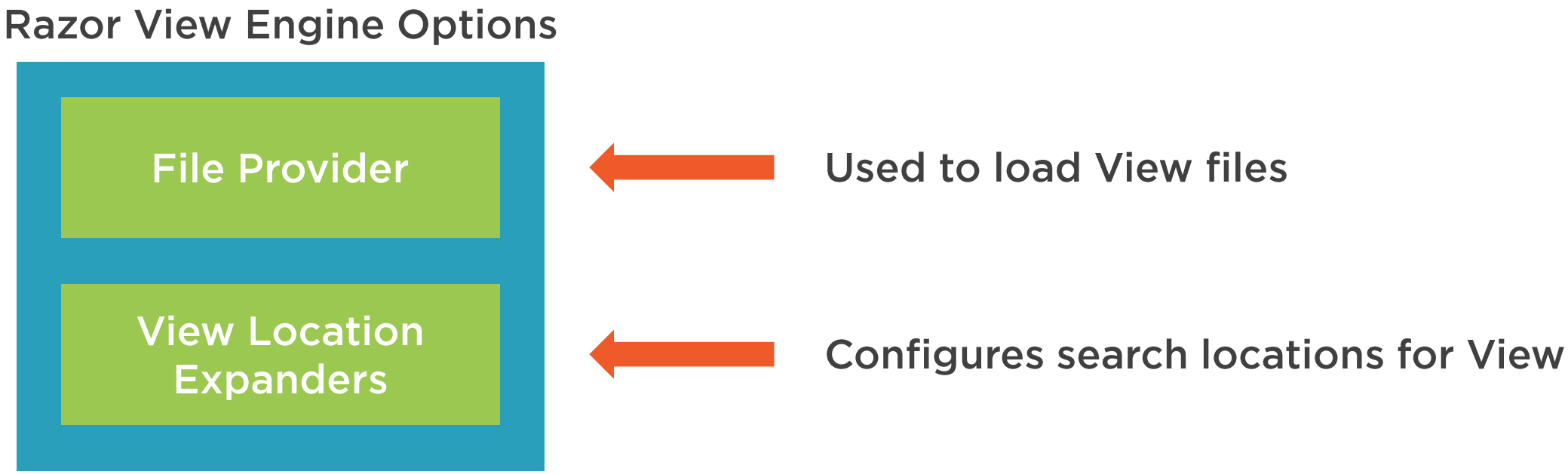
The IView Interface

Must be implemented by a View to render markup or some type of response

Razor Customization Scenarios



Razor Configuration Options



Demo

Influencing the Razor View Engine search locations

Demo

Touring a C# class generated from a Razor View

Summary

Custom Tag Helpers can transform and extend the rendering of HTML elements

Custom Tag Helpers can modify existing HTML elements, or create entirely new elements

Custom HTML Helpers can be created through extension methods

The Razor View Engine search locations for views can be customized

Razor Views are compiled into C# classes that write their contents to the response