Telerik®
Develop experiences

ASP.NET Web API

Telerik Software Academy
http://academy.telerik.com

Table of Contents

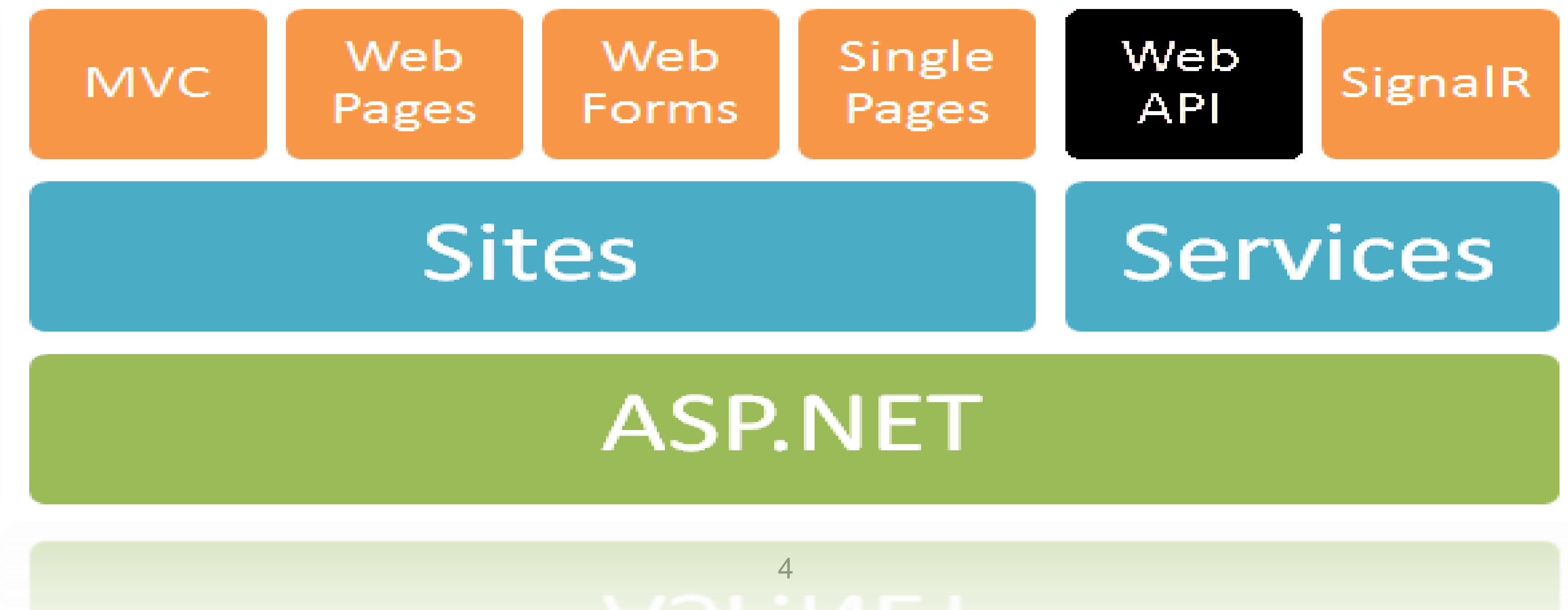
Telerik

What is ASP.NET Web API?

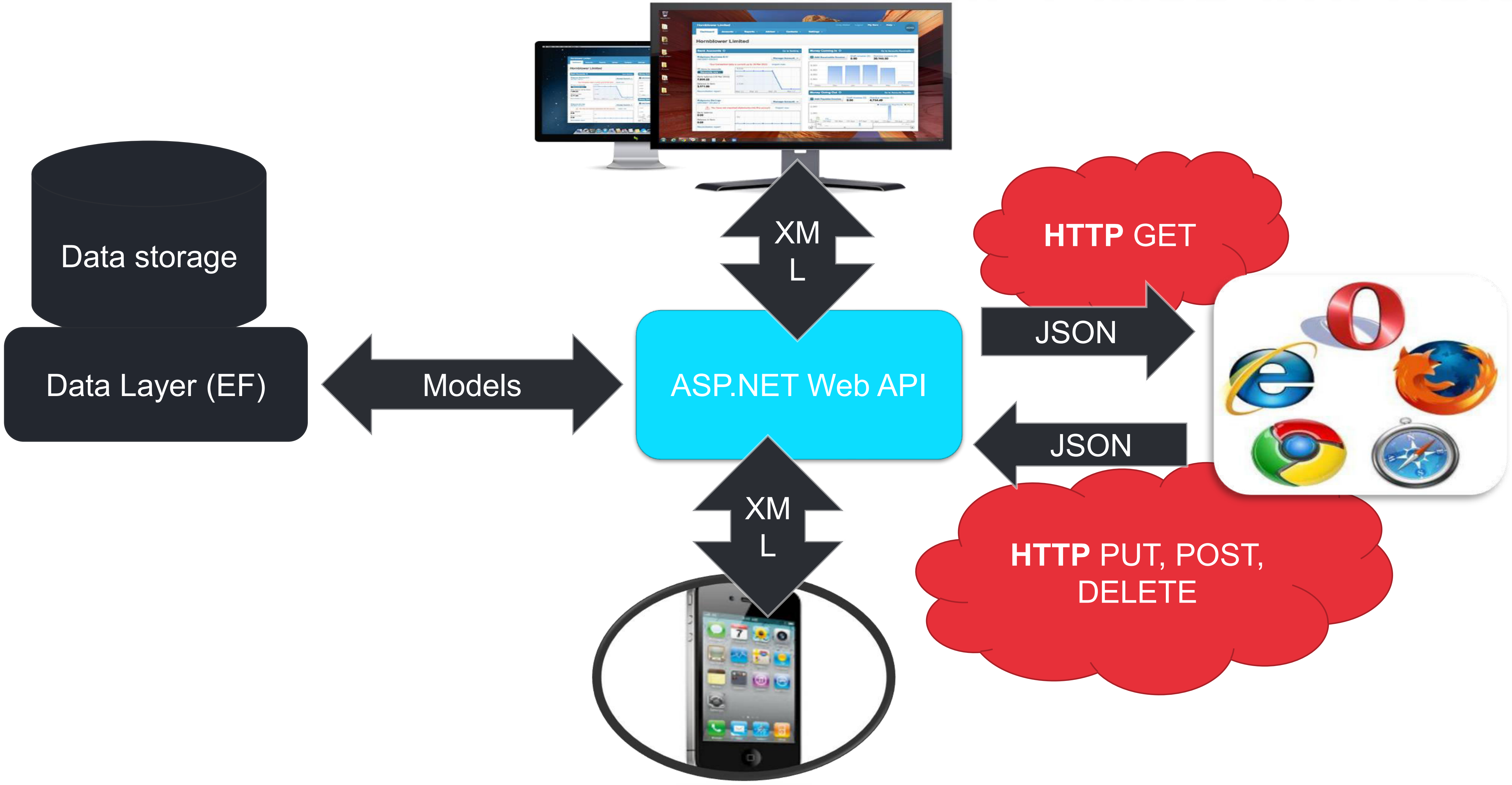
ASP.NET

ASP.NET Web API

- Framework that makes it easy to build HTTP services for browsers and mobile devices
- Platform for building RESTful applications on the .NET Framework using ASP.NET stack



ASP.NET Web API Role



Web API Features

- Modern HTTP programming model
  - Access to strongly typed HTTP object model
  - HttpClient API – same programming model
- Content negotiation
  - Client and server work together to determine the right format for data
  - Provide default support for JSON, XML and Form URL-encoded formats
  - We can add own formats and change content negotiation strategy

Telerik

Web API Features (2)

- Query composition
  - Support automatic paging and sorting
  - Support querying via the OData URL conventions when we return IQueryable<T>
- Model binding and validation
  - Combine HTTP data in POCO models
  - Data validation via attributes
  - Supports the same model binding and validation infrastructure as ASP.NET MVC

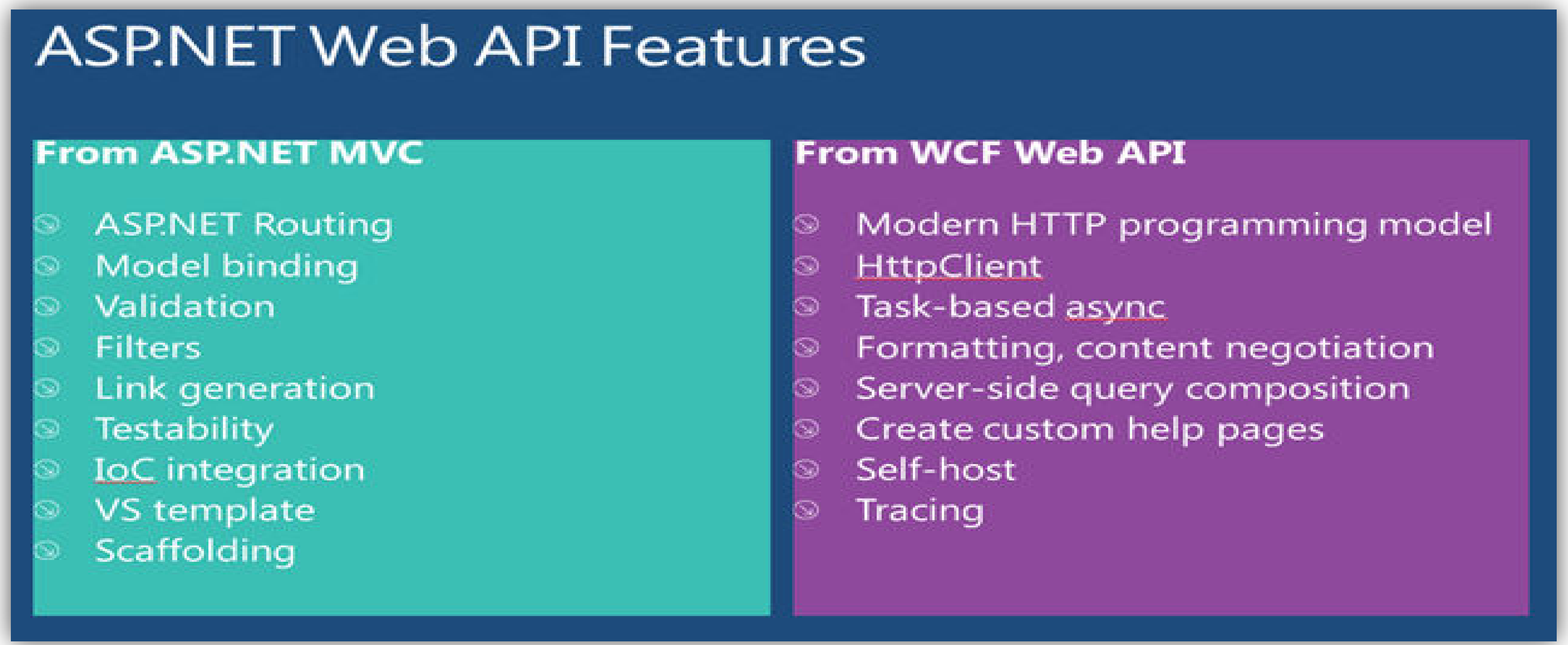
Web API Features (3)

- Routes (mapping between URIs and code)
  - Full set of routing capabilities supported within ASP.NET (MVC)
- Filters
  - Easily decorates Web API with additional validation (authorization, CORS, etc.)
- Testability
- IoC and dependency injection support
- Flexible hosting (IIS, Azure, self-hosting)

Telerik

Web API Features (4)

- Visual Studio IDE (+templates and scaffolding)
- Reuse of C# knowledge (+task-based async)
- Custom help pages, tracing, etc.



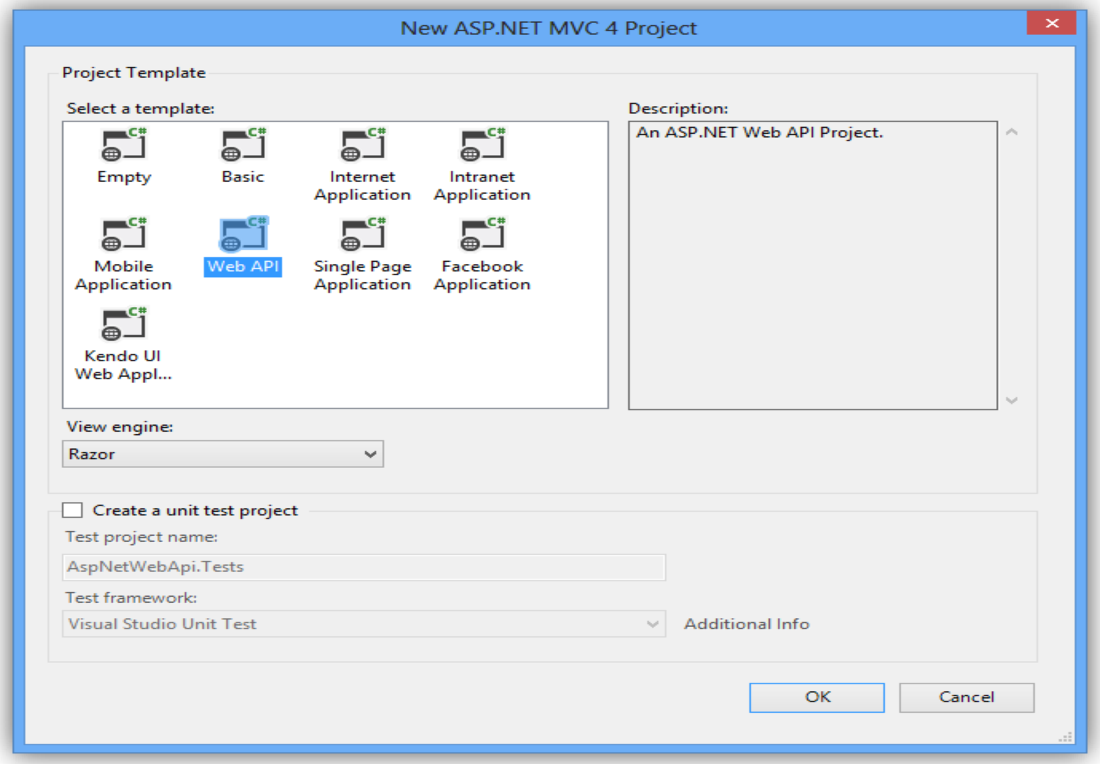
ASP.NET Web API Features

From ASP.NET MVC	From WCF Web API
ASP.NET Routing	Modern HTTP programming model
Model binding	HttpClient
Validation	Task-based async
Filters	Formatting, content negotiation
Link generation	Server-side query composition
Testability	Create custom help pages
IoC integration	Self-host
VS template	Tracing
Scaffolding	

ASP.NET Web API 2

- Attribute routing
- OData improvements: $select, $expand, $batch, $value and improved extensibility
- Request batching
- Portable ASP.NET Web API Client
- Improved testability
- CORS (Cross-origin resource sharing)
- Authentication filters
- OWIN support and integration (owin.org)

Telerik


New ASP.NET MVC 4 Project

Project Template

Select a template:

Description:

An ASP.NET Web API Project.

View engine: Razor

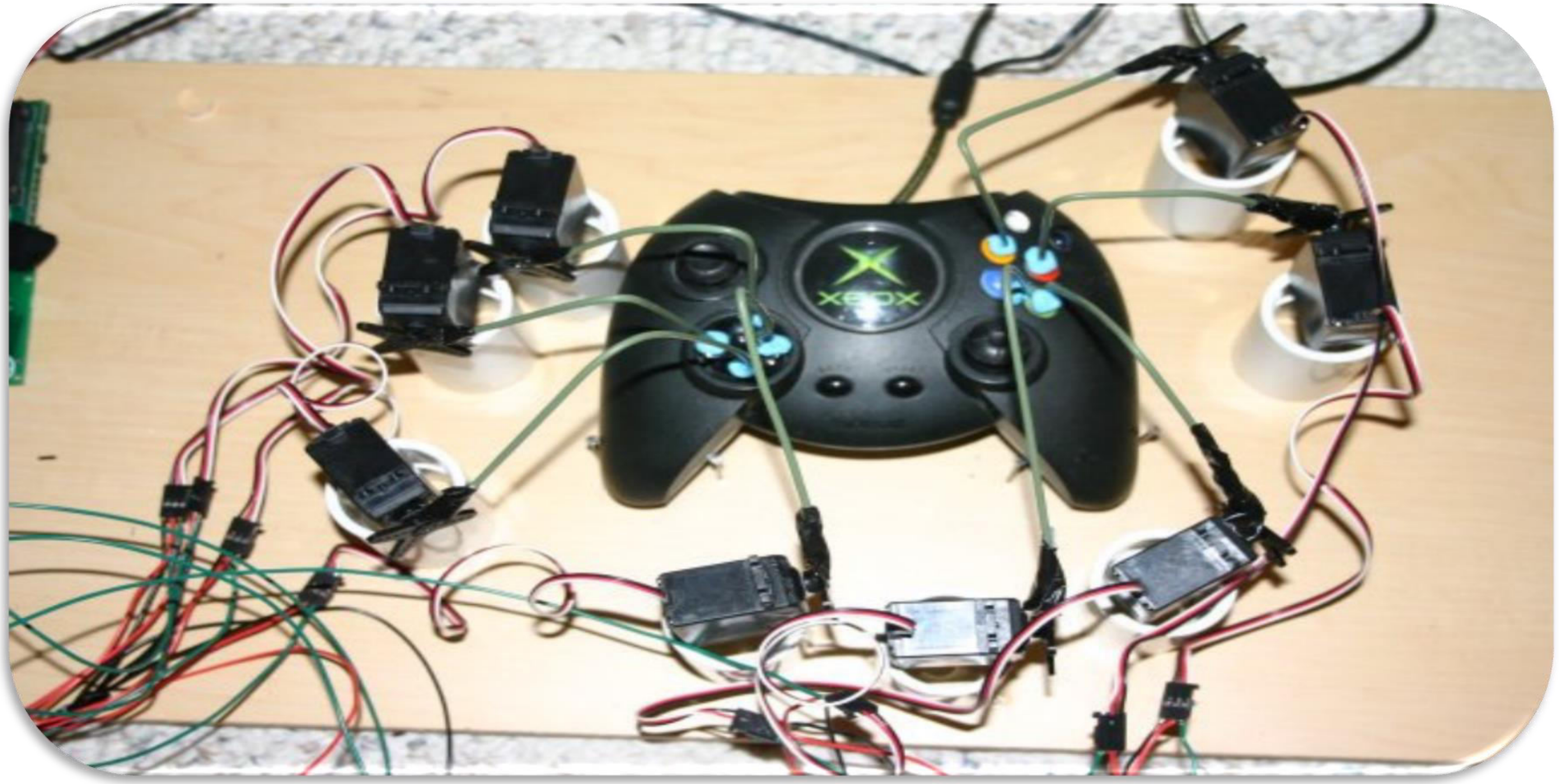
☐ Create a unit test project

Test project name: AspNetWebApi.Tests

Test framework: Visual Studio Unit Test

Demo: Creating ASP.NET Web API Project

Web API Controllers

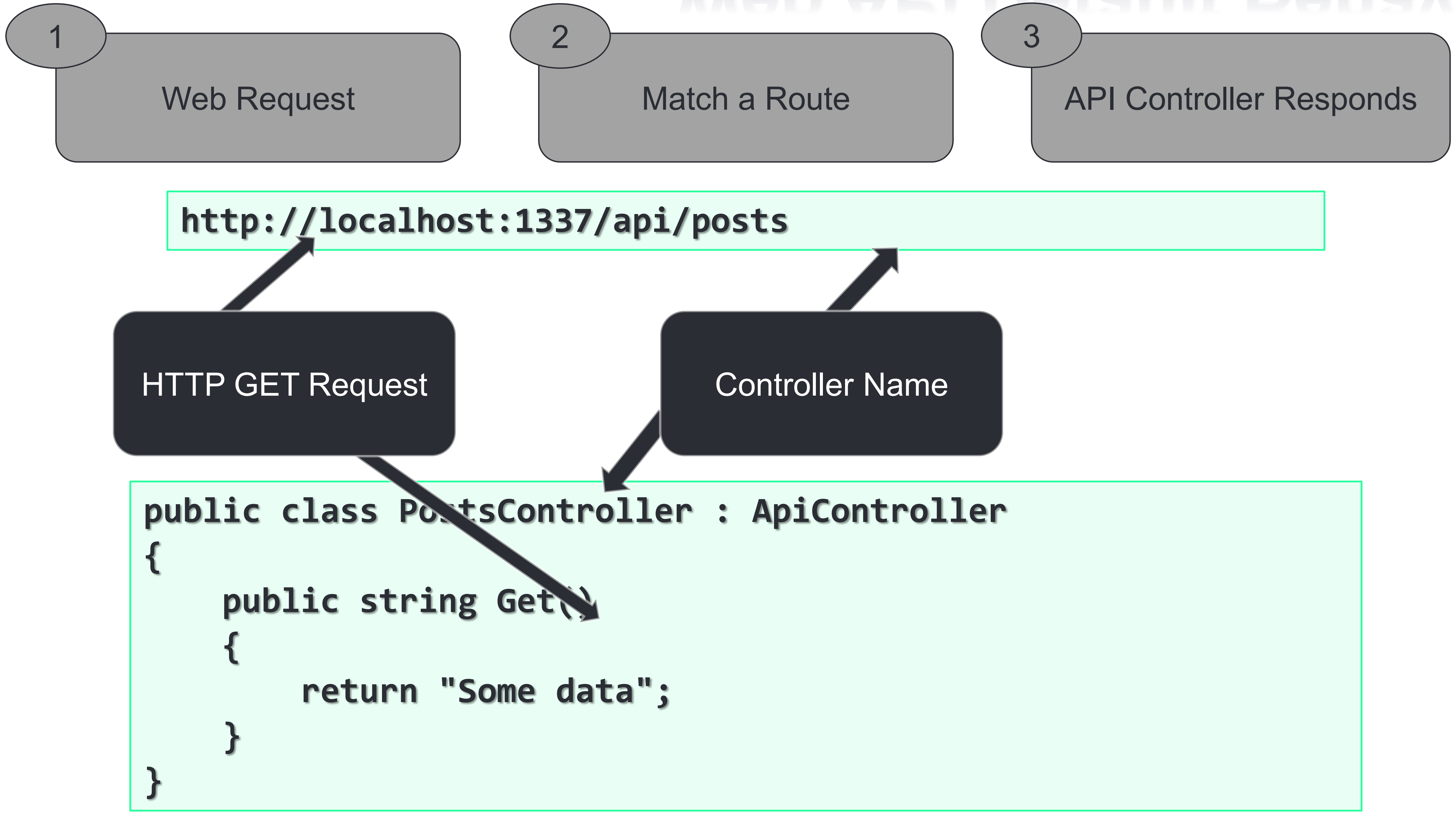


Web API Controllers

- A *controller* is an object that handles HTTP requests
  - All API controllers derive from ApiController
- By default ASP.NET Web API will map HTTP requests to specific methods called actions

Action	HTTP method	Relative URI	Method
Get a list of all posts	GET	/api/posts	Get()
Get a post by ID	GET	/api/posts/*id*	Get(int id)
Create a new post	POST	/api/posts	Post(PostModel value)
Update a post	PUT	/api/posts/*id*	Put(int id, PostModel value)
Delete a post	DELETE	/api/posts/*id*	Delete(int id)
Get a post by category	GET	/api/posts?category=*category*	Get(string category)

Web API Default Behavior



```
http://localhost:1337/api/posts
```

```
public class PostsController : ApiController
{
    public string Get()
    {
        return "Some data";
    }
}
```

Routing

- Routing is how ASP.NET Web API matches a URI to a controller and an action
- Web APIs support the full set of routing capabilities from ASP.NET (MVC)
  - Route parameters
  - Constraints (using regular expressions)
  - Extensible with own conventions
  - Attribute routing is available in version 2

Default Route

- Web API also provides smart conventions by default
  - We can create classes that implement Web APIs without having to explicitly write code
  - HTTP Verb is mapped to an action name

```
http://localhost:1337/api/posts
```

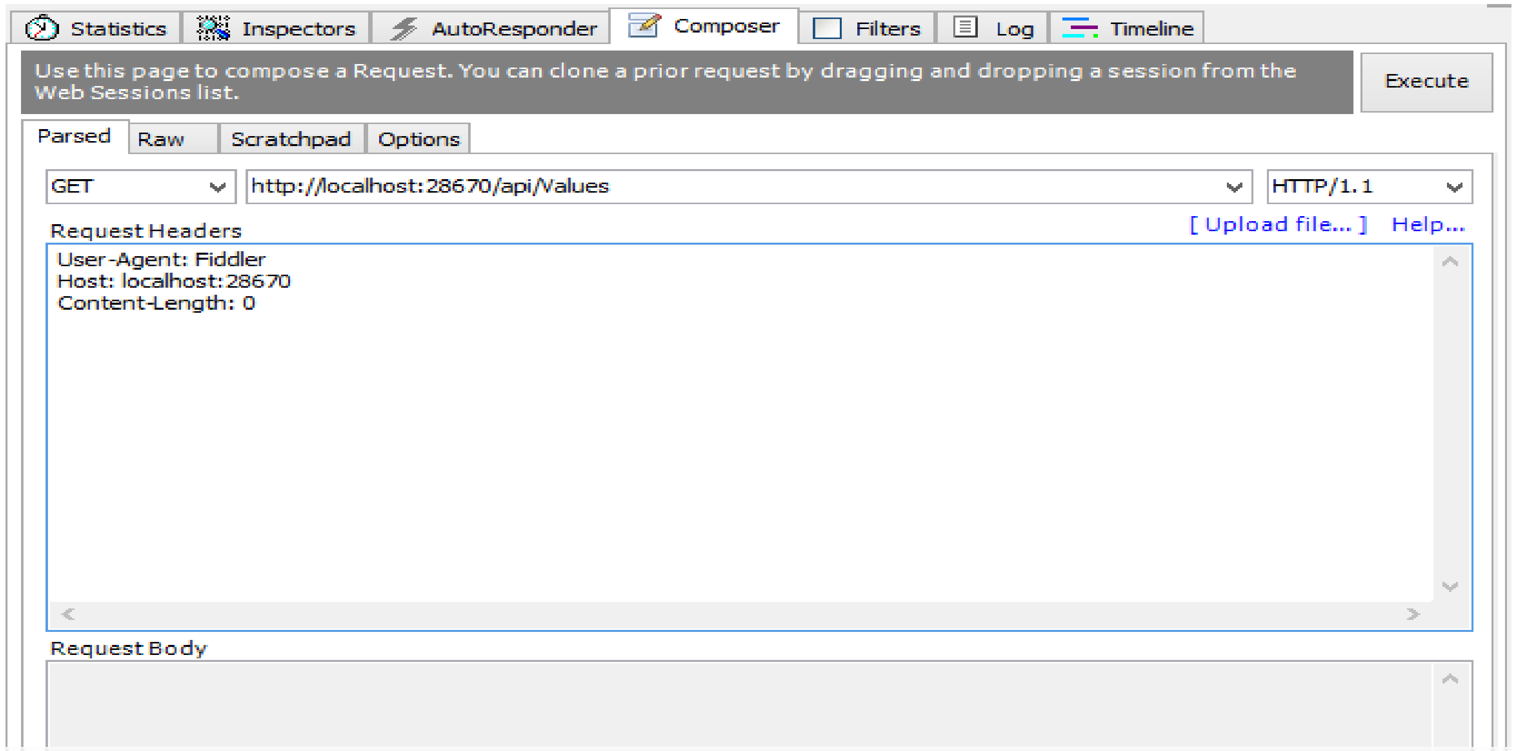
```
routes.MapHtpRoute(name: "DefaultApi",
    routeTemplate: "api/{controller}/{id}",
    defaults: new { id = RoutesParameter.Optional });
```

Telerik

Model Binding & Formatters

- By default the Web API will bind incoming data to POCO (CLR) types
  - Will look in body, header and query string
  - ASP.NET MVC has similar model binder
- MediaTypeFormatters are used to bind both input and output
  - Mapped to content types
- Validation attributes can also be used
- To go down further into the HTTP (set headers, etc.) we can use HttpRequestMessage and HttpResponseMessage

Telerik



Use this page to compose a Request. You can clone a prior request by dragging and dropping a session from the Web Sessions list.

GET http://localhost:28670/api/Values HTTP/1.1

Request Headers

```
User-Agent: Fiddler
Host: localhost:28670
Content-Length: 0
```

Request Body

Demo: Create API Controller

Return Different HTTP Code

- By default when everything is OK, we return HTTP status code 200
- Sometimes we need to return error

```
public HttpResponseMessage Get(int id)
{
   if (dataExists)
   {
      return Request.CreateResponse(
         HttpStatusCode.OK, data);
   }
   else
   {
      return Request.CreateErrorResponse(
         HttpStatusCode.NotFound, "Item not found!");
   }
}
```

OData Query Syntax

- OData (http://odata.org) is a open specification written by Microsoft
  - Provide a standard query syntax on resources
- Implemented by WCF Data Services
- ASP.NET Web API includes automatic support for this syntax
  - Return IQueryable<T> instead of IEnumerable<T>

Telerik

OData Query Syntax

- To enable OData queries uncomment "config.EnableQuerySupport();" line
- Then we can make OData queries like: "http://localhost/Posts?$top=2&$skip=2"

Option	Description
$filter	Filters the results, based on a Boolean condition.
$inlinecount	Tells the server to include the total count of matching entities in the response. (Useful for server-side paging.)
$orderby	Sorts the results.
$skip	Skips the first n results.
$top	Returns only the first n the results.

Web API Clients

HttpClient Model

- HttpClient is a modern HTTP client for .NET
  - Flexible and extensible API for accessing HTTP
- Has the same programming model as the ASP.NET Web API server side
  - HttpRequestMessage / HttpResponseMessage
- Uses Task pattern from .NET 4.0
  - Can use async and await keywords in .NET 4.5
- Installs with ASP.NET MVC 4
  - Can be retrieved via NuGet

Telerik

HttpClient Example

```
var client = new HttpClient {
    BaseAddress = new Uri("http://localhost:28670/") };
client.DefaultRequestHeaders.Accept.Add(new
    MediaTypeWithQualityHeaderValue("application/json"));
HttpResponseMessage response =
    client.GetAsync("api/posts").Result;
if (response.IsSuccessStatusCode)
{
    var products = response.Content
        .ReadAsAsync<IEnumerable<Post>>().Result;
    foreach (var p in products)
    {
        Console.WriteLine("{0,4} {1,-20} {2}",
            p.Id, p.Title, p.CreatedOn);
    }
}
else
    Console.WriteLine("{0} ({1})",
        (int)response.StatusCode, response.ReasonPhrase);
```

Consuming Web API from JS

- Web APIs can be consumed using JavaScript via HTTP AJAX request
  - Example with jQuery:

```
<ol id="posts"></ol>
<script>
   $.ajax({
      url: '/api/posts',
      success: function (posts) {
         var list = $('#posts');
         for (var i = 0; i < posts.length; i++) {
            var post = posts[i];
            list.append('<li>' + post.title + '</li>');
         }
      }
   });
</script>
```

Should be encoded

ASP.NET Web API

http://academy.telerik.com