

第93期-学习WebAPI响应数据的格式

2020年3月17日 15:17

ASP.NET Core MVC 支持设置响应数据的格式。 可以使用特定格式返回数据或响应客户端请求。

上行标头类型: Content-Type, 下行标头类型: Accept

特定于格式的操作结果

一些操作结果类型特定于特殊格式, 例如 JsonResult 和 ContentResult。 操作可以返回使用特定格式设置格式的结果, 而不考虑客户端首选项。

内容协商机制

当客户端指定 Accept 标头时, 会发生内容协商, 默认格式是 JSON。 若将返回的对象为 null, 将返回 204 No Content 响应。

未找到对应的输入格式化程序, 将返回 406 Not Acceptable 状态码。

未找到对应的输出格式化程序, 将使用第一个可用的格式化程序, 否则异常。

如果未指定 Accept 标头, 将使用第一个可以处理对象的格式化程序, 不执行任何协商。 服务器将决定要返回的格式。

浏览器请求API接口, 将忽略 Accept 标头, 默认 JSON 约定, 强制采用浏览器标头:

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddControllers(options =>
    {
        options.RespectBrowserAcceptHeader = true; // false by default
    });
}
```

配置格式化程序

格式化程序库一般使用 Nuget 包承载, 安装添加以下代码。

```
Install-Package Microsoft.AspNetCore.Mvc.Formatters.Json
Install-Package Microsoft.AspNetCore.Mvc.Formatters.Xml
```

```
public void ConfigureServices(IServiceCollection services)
```

```
{
    services.AddControllers().AddXmlSerializerFormatters();
}
```

配置基于 System.Text.Json 的格式化程序

```
services.AddControllers().AddJsonOptions(options =>
{
    // Use the default property (Pascal) casing.
    options.SerializerOptions.PropertyNamingPolicy = null;

    // Configure a custom converter.
    options.SerializerOptions.Converters.Add(new MyCustomJsonConverter());
});

public IActionResult Get()
{
    return Json(model, new JsonSerializerOptions
    {
        WriteIndented = true
    });
}
```

添加基于 Newtonsoft.Json 的 JSON 格式支持

Install-Package Microsoft.AspNetCore.Mvc.NewtonsoftJson

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddControllers().AddNewtonsoftJson();
}
```

可使用下面代码配置基于 Newtonsoft.Json 的序列化器。

```
services.AddControllers().AddNewtonsoftJson(options =>
{
    // Use the default property (Pascal) casing
    options.SerializerSettings.ContractResolver = new
DefaultContractResolver();

    // Configure a custom converter
    options.SerializerOptions.Converters.Add(new MyCustomJsonConverter());
});

public IActionResult Get()
{
    return Json(model, new JsonSerializerSettings
    {
        options.Formatting = Formatting.Indented,
```

```
    });  
}
```

强制指定响应格式

应用 [Produces] 筛选器，以限制响应格式。如同大多筛选器，[Produces] 可以在操作层面、控制器层面或全局范围内应用：

```
[ApiController]  
[Route("[controller]")]  
[Produces("application/json")]  
public class WeatherForecastController : ControllerBase  
{  
}
```

内置特殊格式化程序

Microsoft.AspNetCore.Mvc.Formatters Namespace

```
public void ConfigureServices(IServiceCollection services)  
{  
    services.AddControllers(options =>  
    {  
        // requires using Microsoft.AspNetCore.Mvc.Formatters;  
        options.OutputFormatters.RemoveType<StringOutputFormatter>();  
        options.OutputFormatters.RemoveType<HttpNoContentOutputFormatter>  
    });  
}
```

响应格式 URL 映射

```
[Route("api/[controller]")]  
[ApiController]  
[FormatFilter]  
public class ProductsController : ControllerBase  
{  
    [HttpGet("{id}. {format?}")]  
    public Product Get(int id)  
    {  
    }  
}
```

自定义格式化程序

<https://github.com/aspnet/Entropy/blob/master/samples/Mvc.Formatters/TextPlainInputFormatter.cs>