ASP.NET WEB API 2.1

Enable Cross-Origin Requests (CORS)

Content

- Introduction
- What is "same origin"?
- Enabling CORS in middleware
- Enabling CORS in Controller
- Disable CORS
- CORS policy options
- Example

Introduction

Cross Origin Resource Sharing (CORS) is a W3C standard that allows a server to relax the same-origin policy. Using CORS, a server can explicitly allow some cross-origin requests while rejecting others. CORS is safer and more flexible than earlier techniques such as JSONP

What is "same origin"?

- Two URLs have the same origin if they have identical schemes, hosts, and ports. (RFC 6454)
- These two URLs have the same origin:

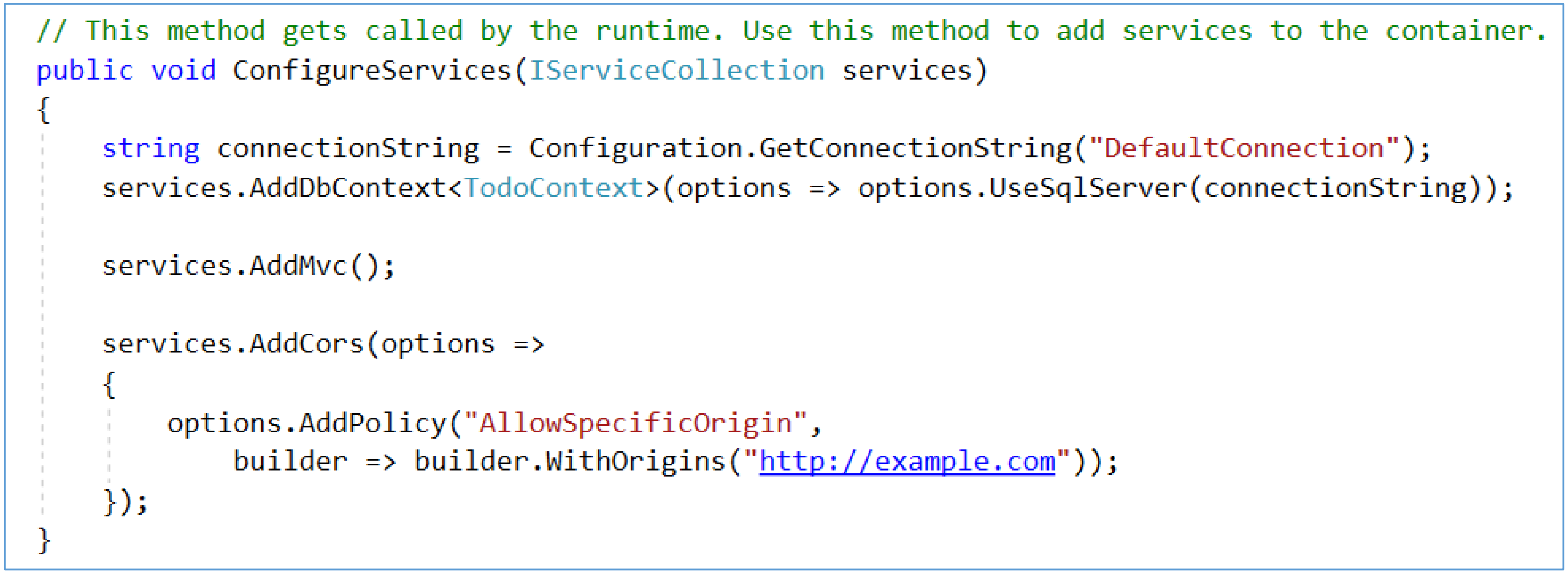
  http://example.com/foo.html
  http://example.com/bar.html

- These URLs have different origins than the previous two:

  http://example.net - Different domain
  http://www.example.com/foo.html - Different subdomain
  https://example.com/foo.html - Different scheme
  http://example.com:9000/foo.html - Different port

Enabling CORS in middleware

- Modify Startup.ConfigureServices

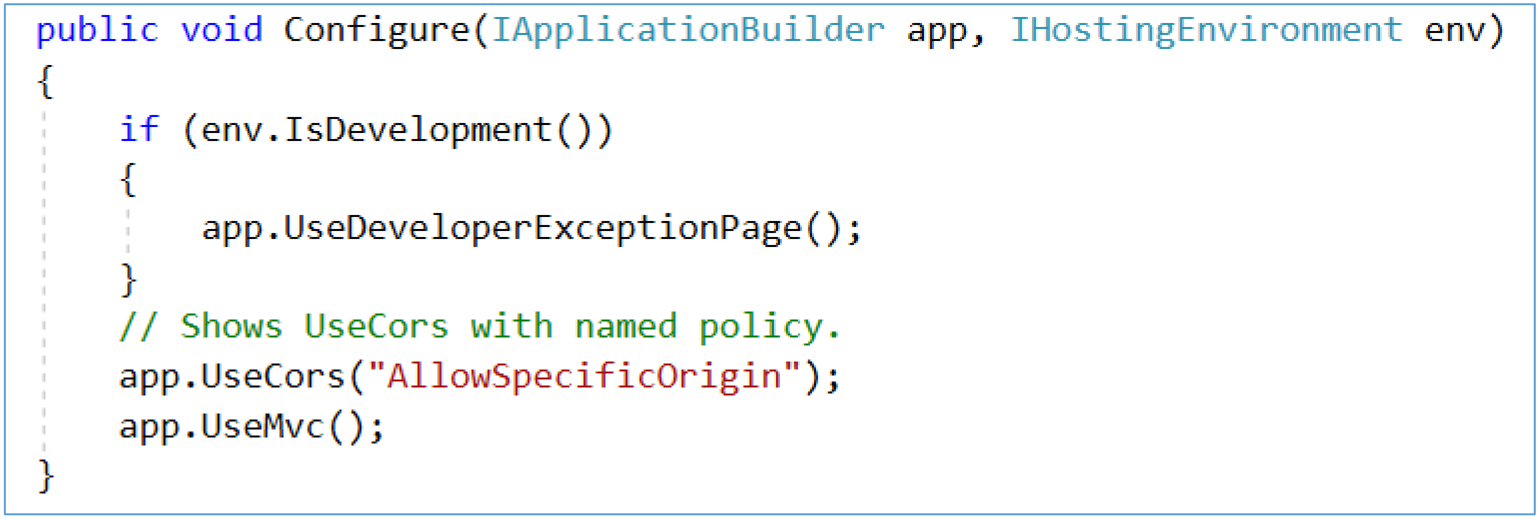


```
// This method gets called by the runtime. Use this method to add services to the container.
public void ConfigureServices(IServiceCollection services)
{
    string connectionString = Configuration.GetConnectionString("DefaultConnection");
    services.AddDbContext<TodoContext>(options => options.UseSqlServer(connectionString));

    services.AddMvc();

    services.AddCors(options =>
    {
        options.AddPolicy("AllowSpecificOrigin",
            builder => builder.WithOrigins("http://example.com"));
    });
}
```

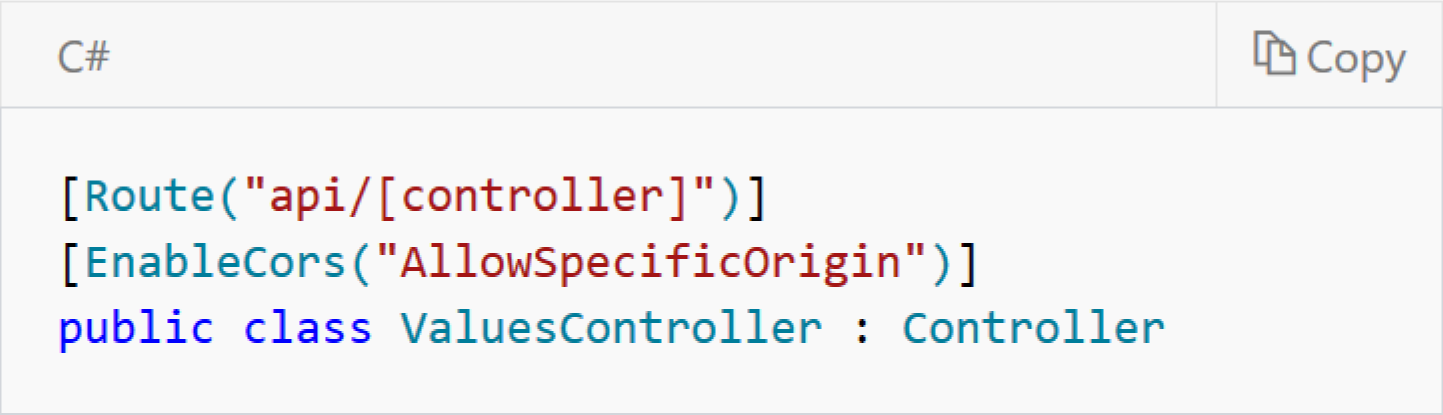
- Modify Startup.Configure



```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    // Shows UseCors with named policy.
    app.UseCors("AllowSpecificOrigin");
    app.UseMvc();
}
```

Enabling CORS in Controller

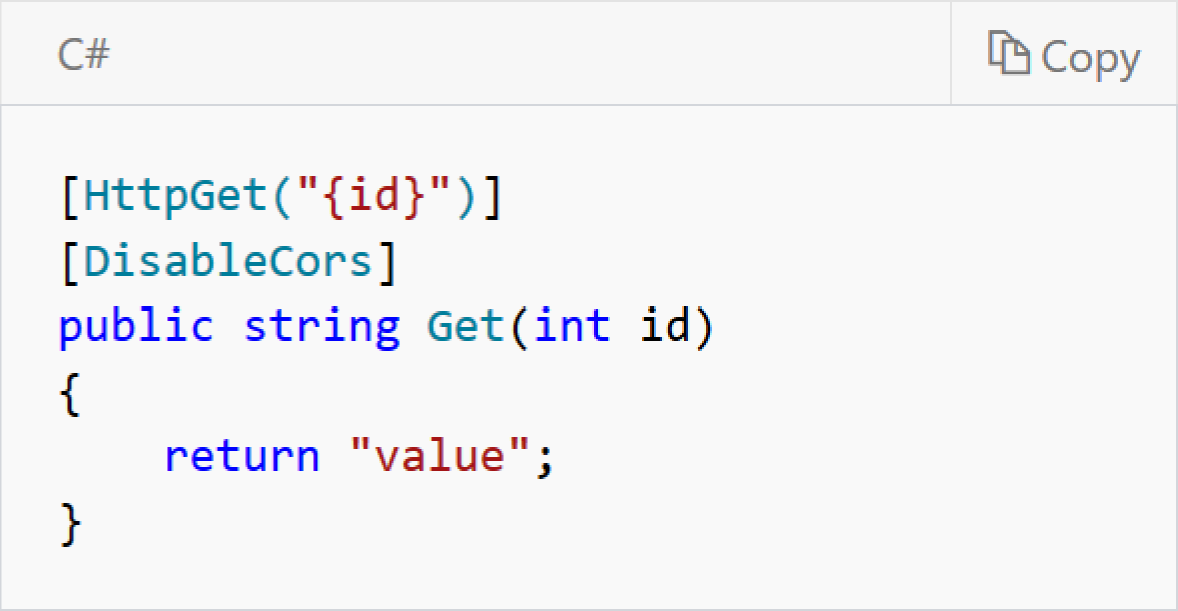
To specify the CORS policy for a specific controller add the [EnableCors] attribute to the controller class. Specify the policy name.



```
[Route("api/[controller]")]
[EnableCors("AllowSpecificOrigin")]
public class ValuesController : Controller
```

Disable CORS

To disable CORS for a controller or action, use the [DisableCors] attribute.

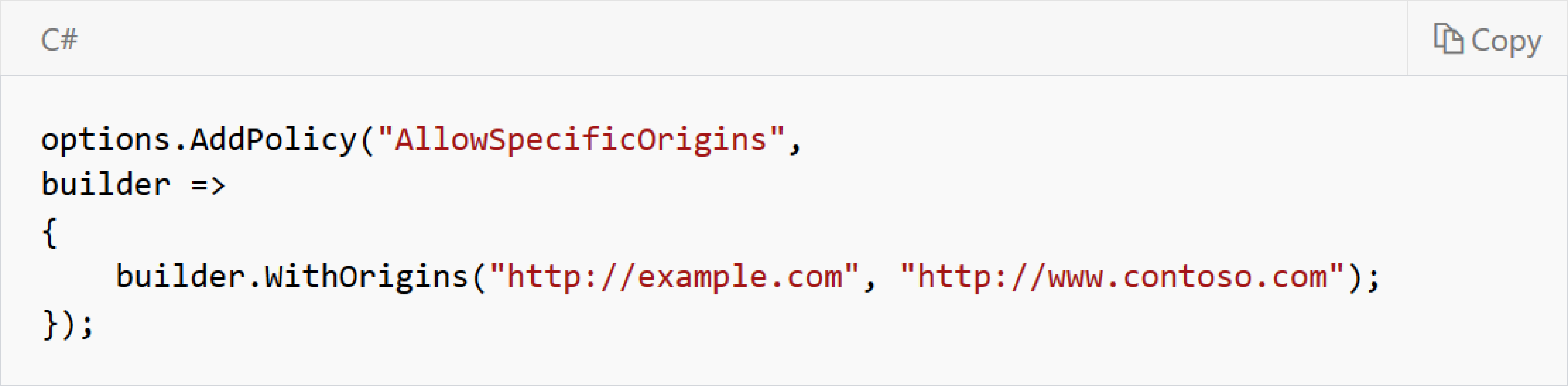


```csharp
[HttpGet("{id}")]
[DisableCors]
public string Get(int id)
{
    return "value";
}
```

CORS policy options

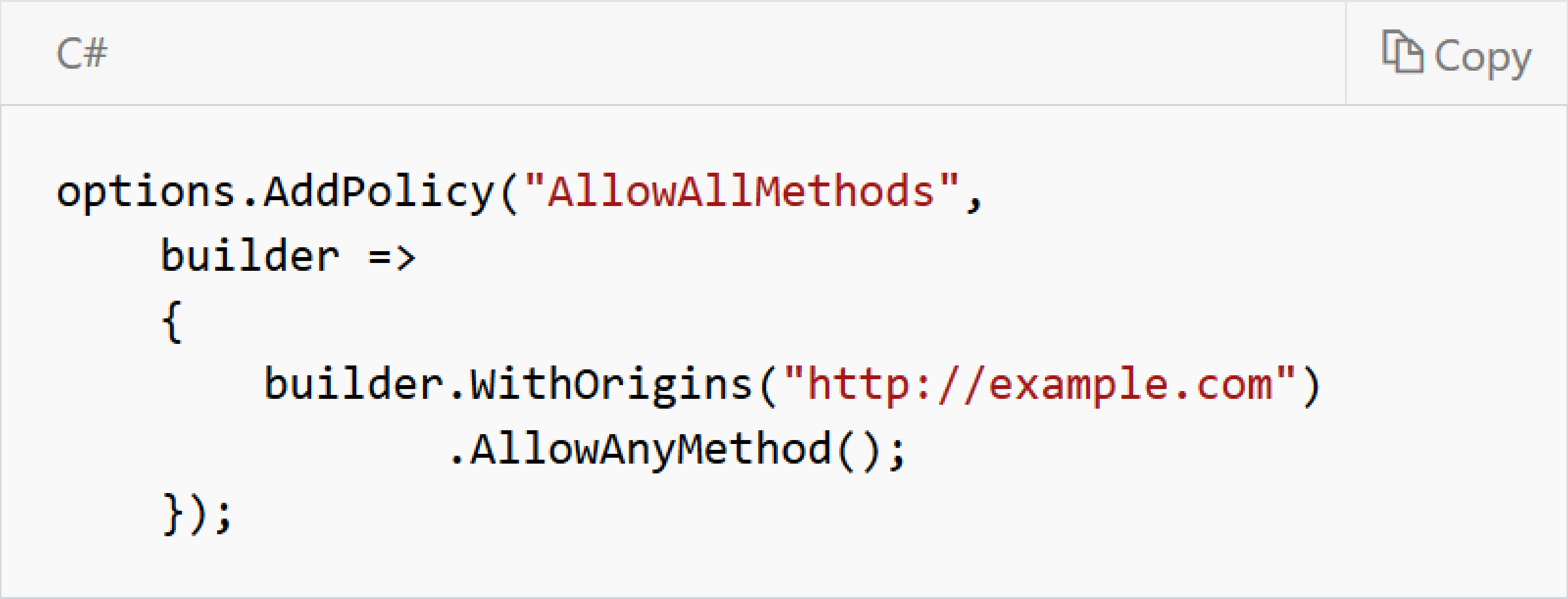
- Set the allowed origins
- Set the allowed HTTP methods
- Set the allowed request headers
- Set the exposed response headers
- Credentials in cross-origin requests
- Set the preflight expiration time

Set the allowed origins



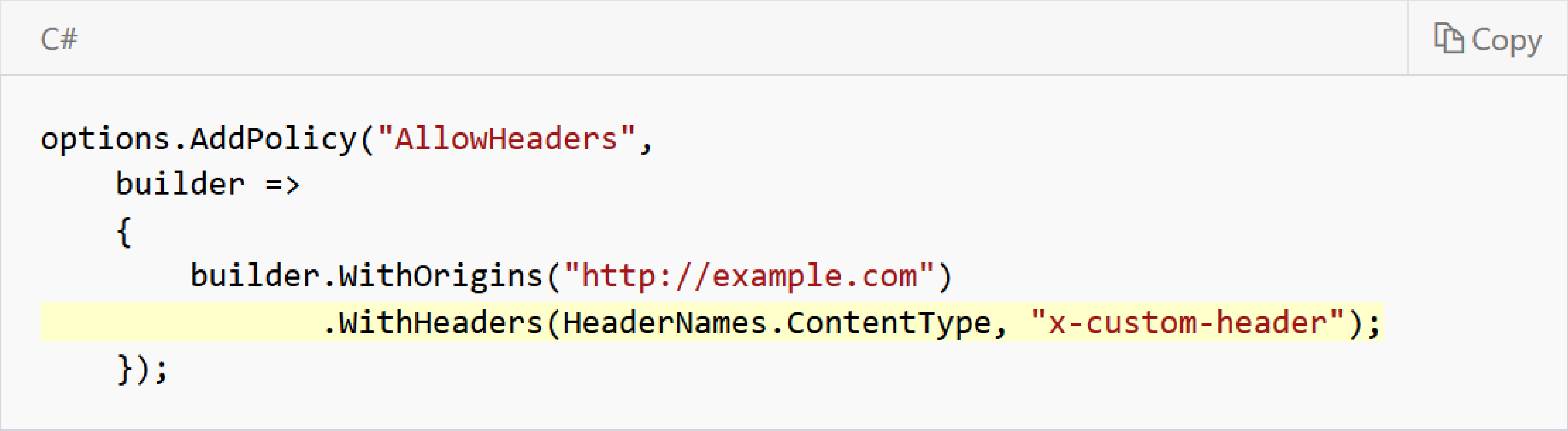
```
options.AddPolicy("AllowSpecificOrigins",
builder =>
{
    builder.WithOrigins("http://example.com", "http://www.contoso.com");
});
```

Set the allowed HTTP methods

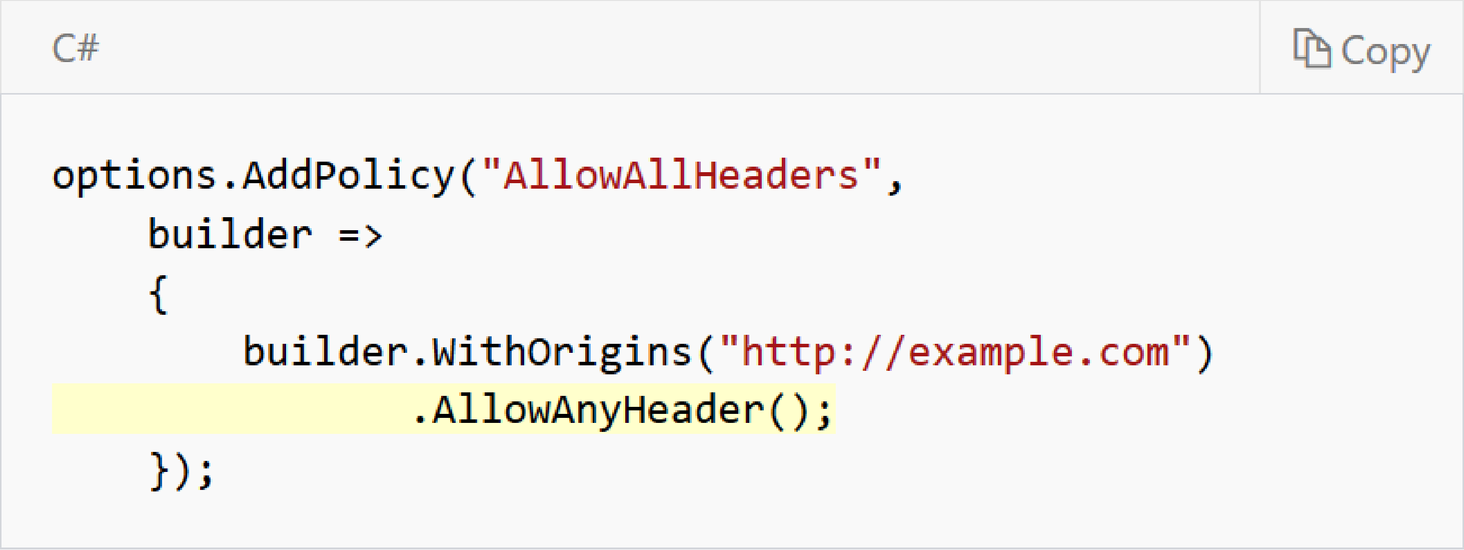


```
options.AddPolicy("AllowAllMethods",
    builder =>
    {
        builder.WithOrigins("http://example.com")
               .AllowAnyMethod();
    });
```

Set the allowed request headers



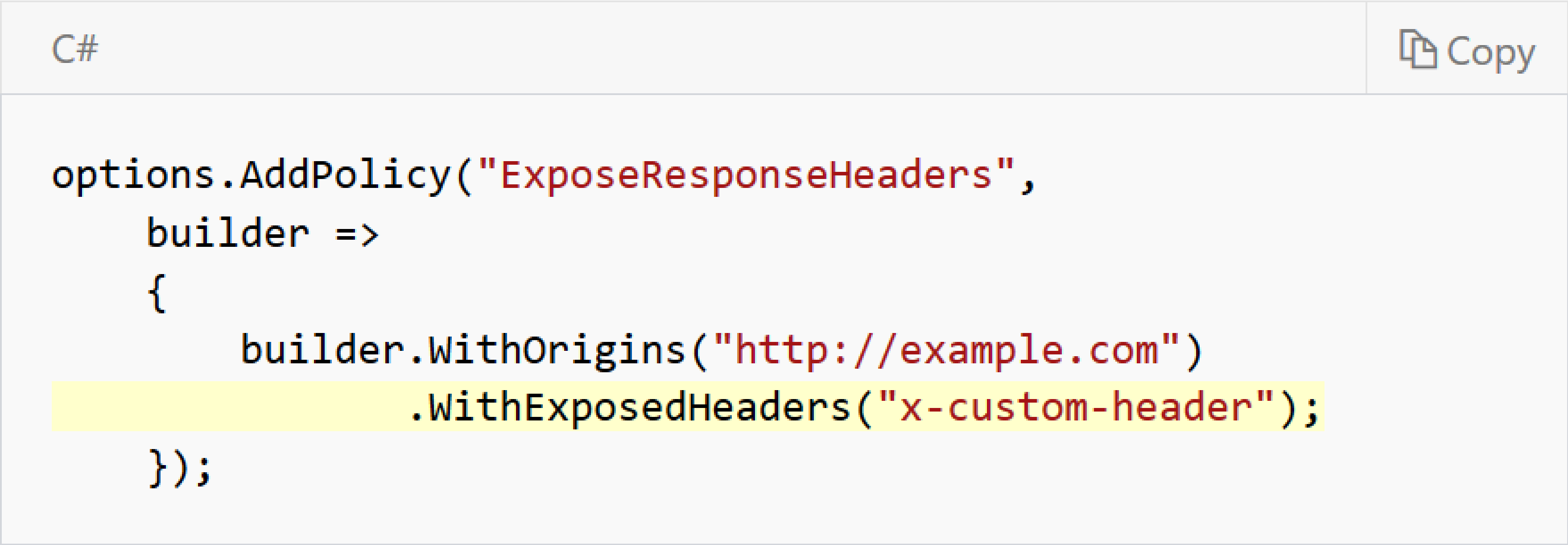
```
options.AddPolicy("AllowHeaders",
    builder =>
    {
        builder.WithOrigins("http://example.com")
               .WithHeaders(HeaderNames.ContentType, "x-custom-header");
    });
```



```
options.AddPolicy("AllowAllHeaders",
    builder =>
    {
        builder.WithOrigins("http://example.com")
               .AllowAnyHeader();
    });
```

Set the exposed response headers

- Cache-Control
- Content-Language
- Content-Type
- Expires
- Last-Modified
- Pragma



```
options.AddPolicy("ExposeResponseHeaders",
    builder =>
    {
        builder.WithOrigins("http://example.com")
               .WithExposedHeaders("x-custom-header");
    });
```

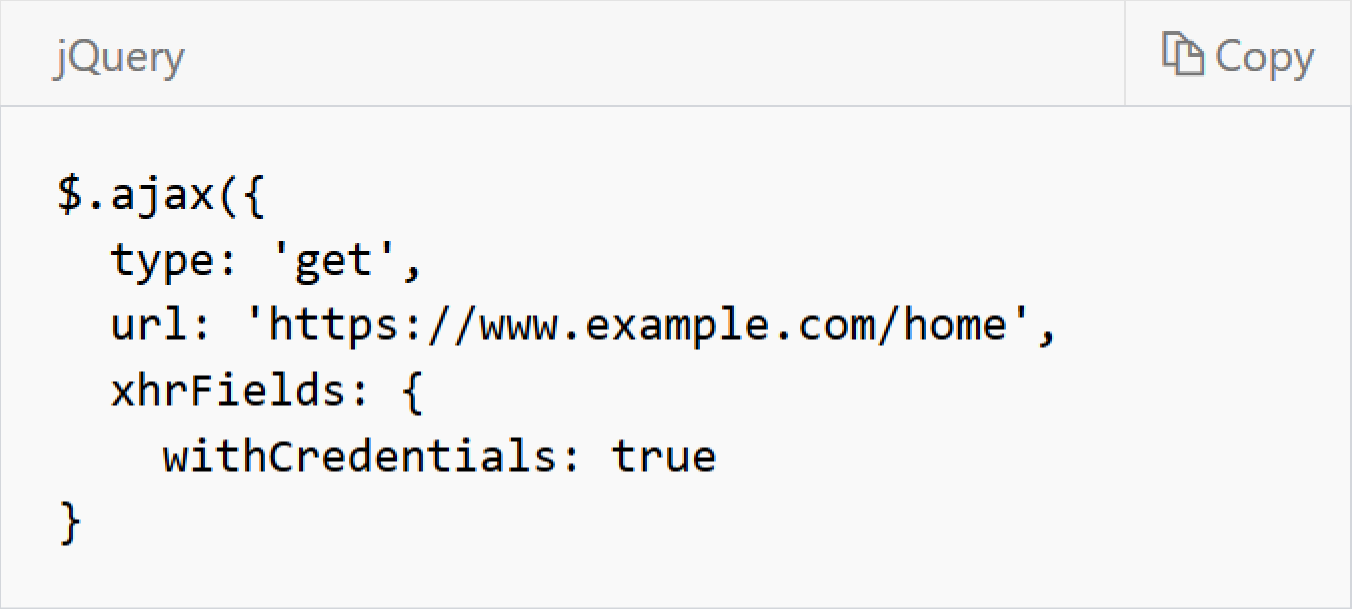
Credentials in cross-origin requests

Credentials require special handling in a CORS request.

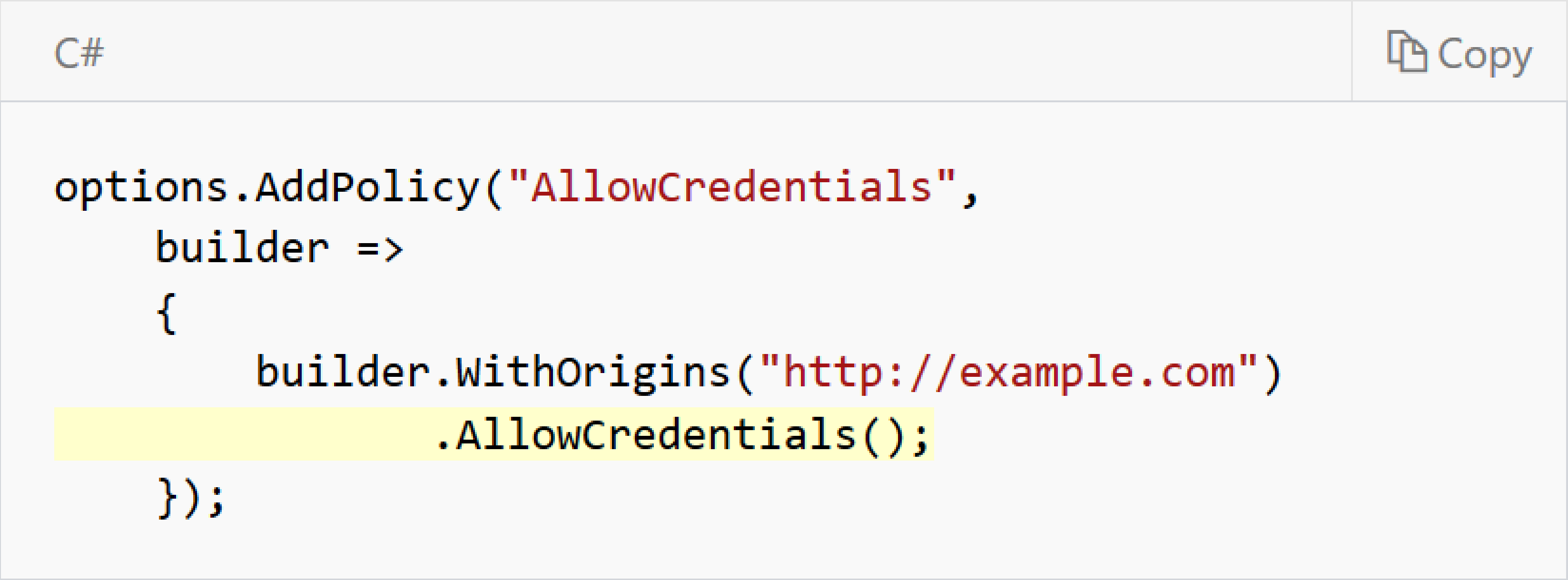
By default, the browser doesn't send credentials with a cross-origin request.

Credentials include cookies and HTTP authentication schemes.

To send credentials with a cross-origin request, the client must set XMLHttpRequest.withCredentials to true.



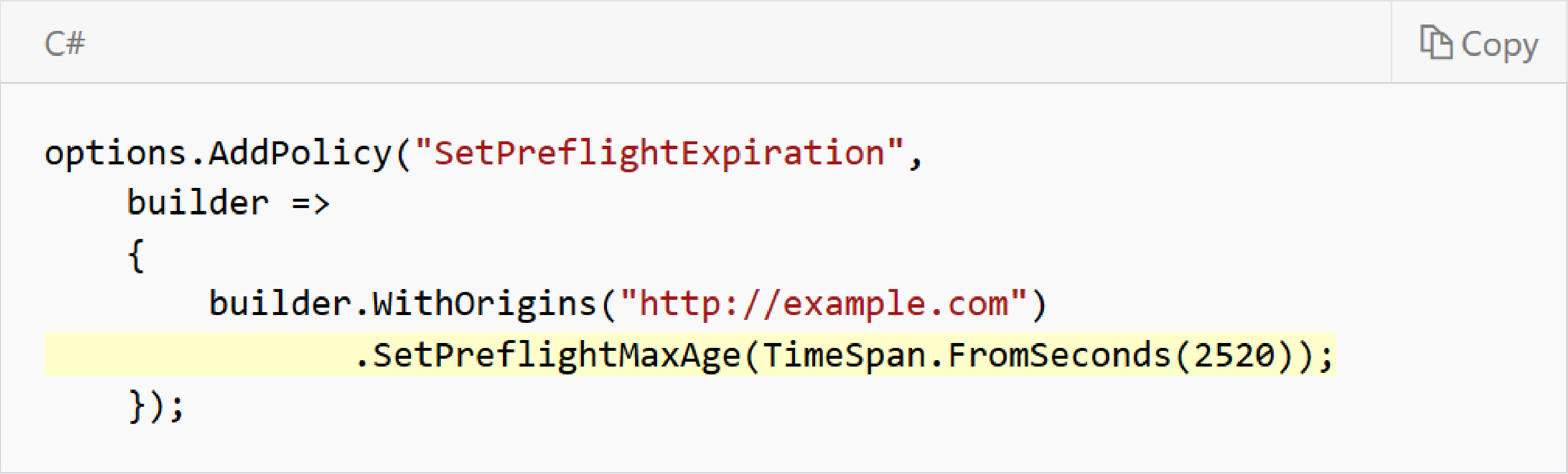
```
$.ajax({
  type: 'get',
  url: 'https://www.example.com/home',
  xhrFields: {
    withCredentials: true
}
```



```
options.AddPolicy("AllowCredentials",
    builder =>
    {
        builder.WithOrigins("http://example.com")
               .AllowCredentials();
    });
```

Set the preflight expiration time

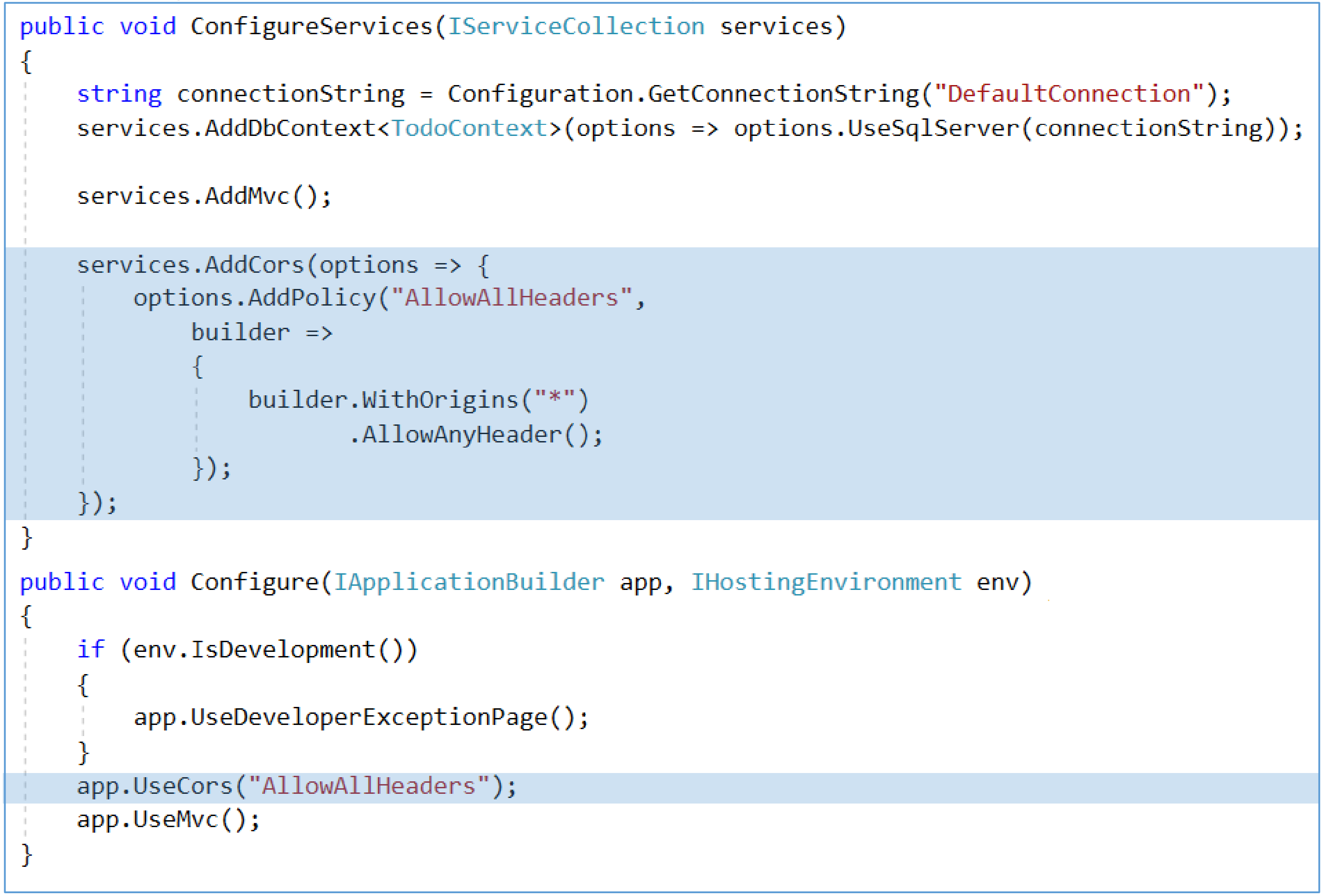
The Access-Control-Max-Age header specifies how long the response to the preflight request can be cached



```
options.AddPolicy("SetPreflightExpiration",
    builder =>
    {
        builder.WithOrigins("http://example.com")
               .SetPreflightMaxAge(TimeSpan.FromSeconds(2520));
    });
```

Example

- Modify content in Startup.cs



```
public void ConfigureServices(IServiceCollection services)
{
    string connectionString = Configuration.GetConnectionString("DefaultConnection");
    services.AddDbContext<TodoContext>(options => options.UseSqlServer(connectionString));

    services.AddMvc();

    services.AddCors(options => {
        options.AddPolicy("AllowAllHeaders",
            builder =>
            {
                builder.WithOrigins("*")
                       .AllowAnyHeader();
            });
    });
}
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    app.UseCors("AllowAllHeaders");
    app.UseMvc();
}
```

THE END