**@CrossOrigin**

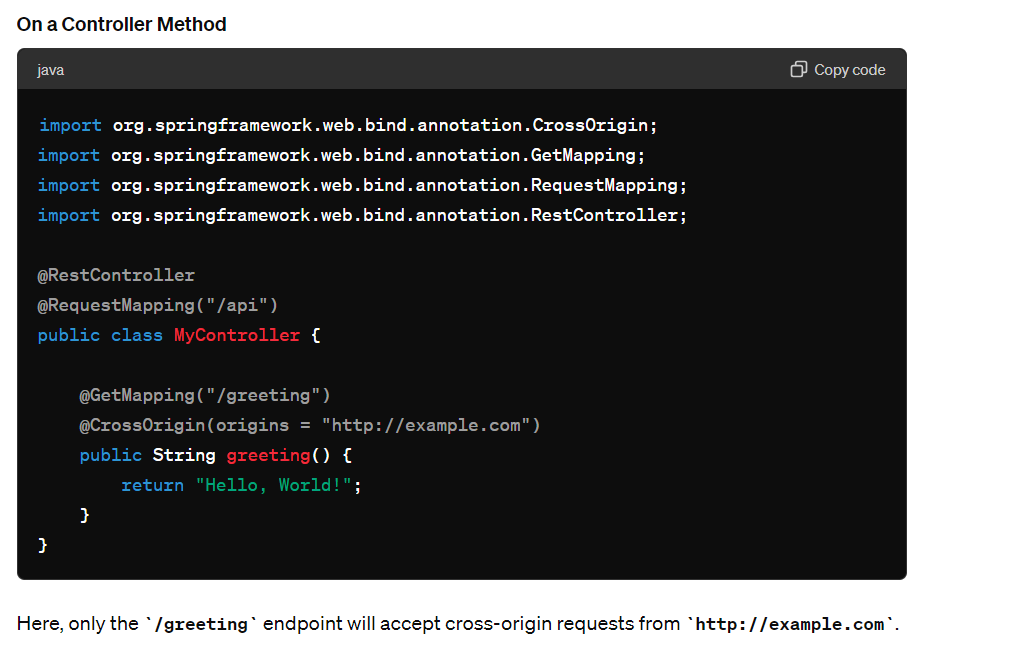
The **@CrossOrigin** annotation in Spring Framework is used to handle Cross-Origin Resource Sharing (CORS) at the controller level. CORS is a security feature implemented by browsers to prevent web applications from making requests to domains other than the one that served the web page. However, there are legitimate use cases where cross-origin requests are necessary, and **@CrossOrigin** allows you to specify the rules under which these requests are allowed.

**Usage**

The **@CrossOrigin** annotation can be applied to methods or entire classes in a Spring controller. Here’s how you can use it:

****

In this example, all endpoints in the **MyController** class will accept cross-origin requests from [**http://example.com**](http://example.com).

****

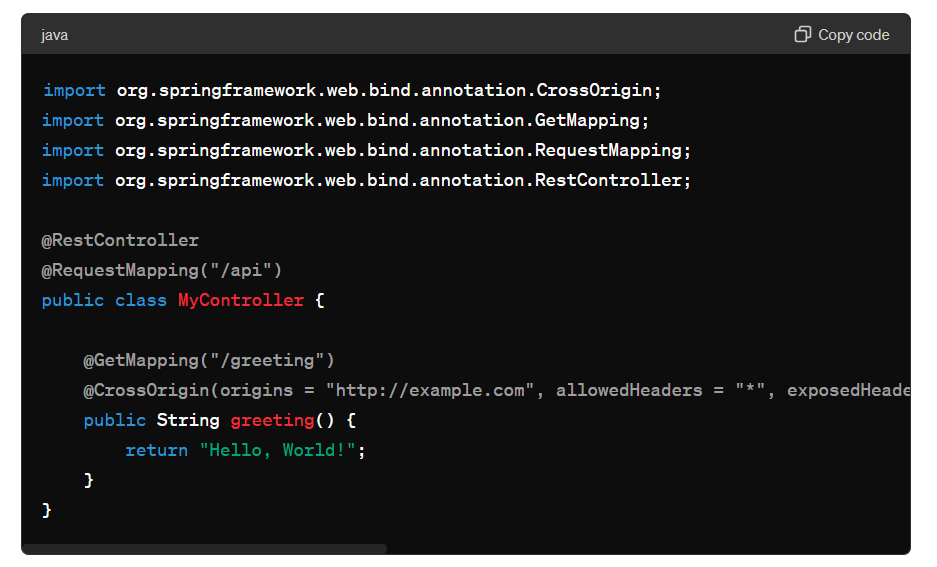
**Configuration Options**

The **@CrossOrigin** annotation has several attributes that you can use to fine-tune its behavior:

* **origins**: Specifies the allowed origins (e.g., **"http://example.com"**). Use **\*** to allow any origin.
* **allowedHeaders**: Specifies the headers that can be used during the actual request.
* **exposedHeaders**: Specifies the response headers that can be exposed to the client.
* **methods**: Specifies the HTTP methods allowed when accessing the resource (e.g., **RequestMethod.GET**, **RequestMethod.POST**).
* **maxAge**: Indicates how long the results of a preflight request can be cached.
* **allowCredentials**: Indicates whether user credentials are supported.

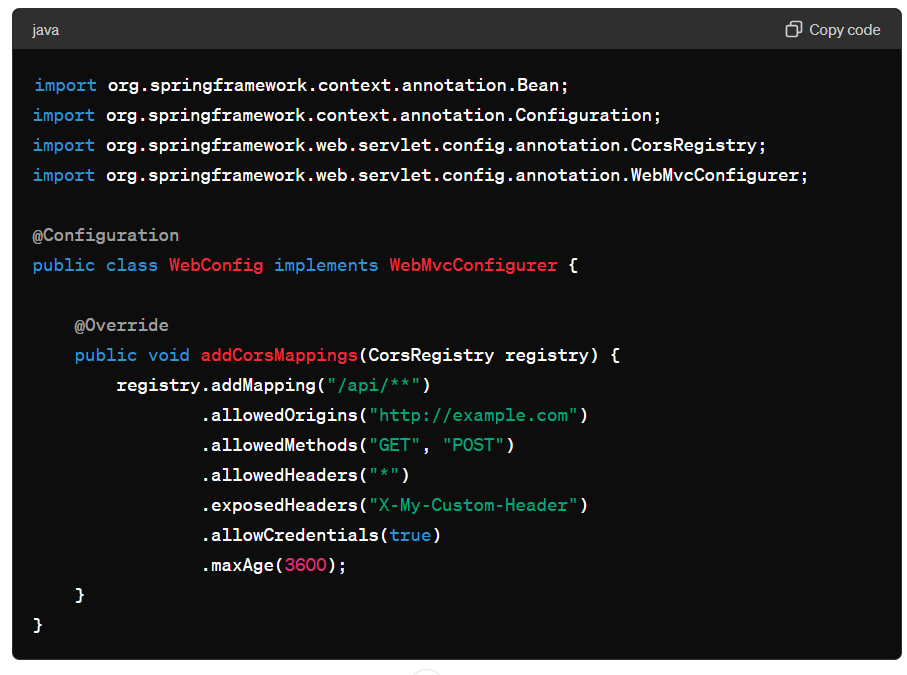
Example with more attributes:

@CrossOrigin(origins = "http://example.com", allowedHeaders = "\*", exposedHeaders = "X-My-Custom-Header", methods = {RequestMethod.GET, RequestMethod.POST}, maxAge = 3600, allowCredentials = "true")



### Global CORS Configuration

While the **@CrossOrigin** annotation is useful for fine-grained control, you might want to configure CORS globally for all controllers. This can be done in a Spring configuration class:

****

In this configuration, all endpoints matching **/api/\*\*** will accept cross-origin requests from **http://example.com**.

### Conclusion

The **@CrossOrigin** annotation in Spring allows for precise control over CORS at both the class and method levels. For broader use cases, global configuration can be set up to manage CORS rules across the application. This flexibility ensures that you can handle cross-origin requests securely and according to your application’s needs.

**Caching :**

Caching is a technique used to store copies of data or computations in a location that can be accessed more quickly than the original source. The primary goal of caching is to improve the speed and efficiency of data retrieval.