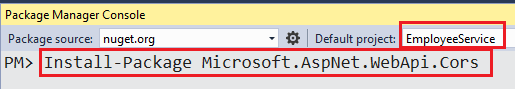
In this video we will discuss how to call an ASP.NET Web API service in a cross domain using jQuery ajax. In our [previous video](https://www.youtube.com/watch?v=WKZLVqwpoiE&list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b&index=14) we discussed how to do this using JSONP. In this video we will discuss how to **enable CORS** (Cross Origin Resource Sharing) which allows cross domain ajax calls. CORS support is released with ASP.NET Web API 2.   
  
This is continuation to [Part 14](https://www.youtube.com/watch?v=WKZLVqwpoiE&list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b&index=14). Please watch [Part 14](https://www.youtube.com/watch?v=WKZLVqwpoiE&list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b&index=14) from [ASP.NET Web API tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b)before proceeding with this video.  
  
  
  
Comment the following 2 lines of code in **Register**() method of **WebApiConfig** class in **WebApiConfig.cs** file in **App\_Start folder**. We added these lines in our previous video to make ASP.NET Web API Service return JSONP formatted data

//var jsonpFormatter = new JsonpMediaTypeFormatter(config.Formatters.JsonFormatter);

//config.Formatters.Insert(0, jsonpFormatter);

**To allow cross domain ajax calls by enabling CORS**  
**Step 1 :**Install **Microsoft.AspNet.WebApi.Cors** package. Execute the following command using NuGet Package Manager Console. Make sure to select "EmployeeService" project from "Default project" dropdown.  
Install-Package Microsoft.AspNet.WebApi.Cors   
  
   
  
  
  
**Step 2 :** Include the following 2 lines of code in **Register()**method of **WebApiConfig**class in **WebApiConfig.cs** file in **App\_Start**folder

EnableCorsAttribute cors = new EnableCorsAttribute("\*", "\*", "\*");

config.EnableCors();

**Step 3 :** In the ClientApplication, set the dataType option of the jQuery ajax function to json  
dataType: 'json'  
  
**Parameters of EnableCorsAttribute**

| **Parameter** | **Description** |
| --- | --- |
| origins | Comma-separated list of origins that are allowed to access the resource. For example "http://www.pragimtech.com,http://www.mywebsite.com" will only allow ajax calls from these 2 websites. All the others will be blocked. Use "\*" to allow all |
| headers | Comma-separated list of headers that are supported by the resource. For example "accept,content-type,origin" will only allow these 3 headers. Use "\*" to allow all. Use null or empty string to allow none |
| methods | Comma-separated list of methods that are supported by the resource. For example "GET,POST" only allows Get and Post and blocks the rest of the methods. Use "\*" to allow all. Use null or empty string to allow none |

The following 2 lines of code in Register() method of WebApiConfig.cs file in App\_Start folder, enables CORS globally for the entire application i.e for all controllers and action methods 

EnableCorsAttribute cors = new EnableCorsAttribute("\*", "\*", "\*");

config.EnableCors();

EnableCors attribute can be applied on a **specific controller** or **controller method.**  
  
If applied at a controller level then it is applicable for all methods in the controller. To apply it at the controller level  
  
1. There is no need to create an instance of EnableCorsAttribute in Register() method of WebApiConfig.cs file. Call the EnableCors() method without any parameter values.  
  
**config.EnableCors();**  
  
2. Apply the  EnableCorsAttribute on the controller class

[EnableCorsAttribute("\*", "\*", "\*")]

public class EmployeesController : ApiController

{

}

In the same manner, you can also apply it at a method level if you wish to do so.  
  
To disable CORS for a specific action apply [DisableCors] on that specific action  
  
When CORS is enabled, the browser sets the **origin**header of the request to the domain of the site making the request. The server sets **Access-Control-Allow-Origin**header in the response to either \* or the origin that made the request. \* indicates any site is allowed to make the request.