ASP.NET Web API - 1.1

Summary: This document provides information about the extension providing ASP.NET Web API support for C#.

What's new?

See ASP.NET Web API - 1.1 - Release Notes for more information.

Description

This extension provides support for ASP.NET Web API. This extension will enable users to create links between server side APIs and client calls for HttpGet, httpPut, HttpPost, and HttpDelete methods.

In what situation should you install this extension?

If your .NET application contains ASP.NET Web API source code and you want to view these object types and their links with client side calls, then you should install this extension.

ASP.NET Web API support

The following frameworks are supported by this extension:

Version	Supported
Web API 2	✅
ASP.NET Core Web API	✅

Function Point, Quality and Sizing support

This extension provides the following support:

- **Function Points (transactions)**: a green tick indicates that OMG Function Point counting and Transaction Risk Index are supported
- **Quality and Sizing**: a green tick indicates that CAST can measure size and that a minimum set of Quality Rules exist

Function Points (transactions)	Quality and Sizing
✅	✅

CAST AIP compatibility

This extension is compatible with:

CAST AIP release	Supported	Languages
8.3.x	✅	C#
8.2.x	✅	C#
8.1.x	✅	C#
8.0.x	✅	C#
7.3.7 and all higher 7.3.x releases	✅	C#

Supported DBMS servers

This extension is compatible with the following DBMS servers:

DBMS	Supported
CSS/PostgreSQL	✅
Oracle	❌
Microsoft SQL Server	❌

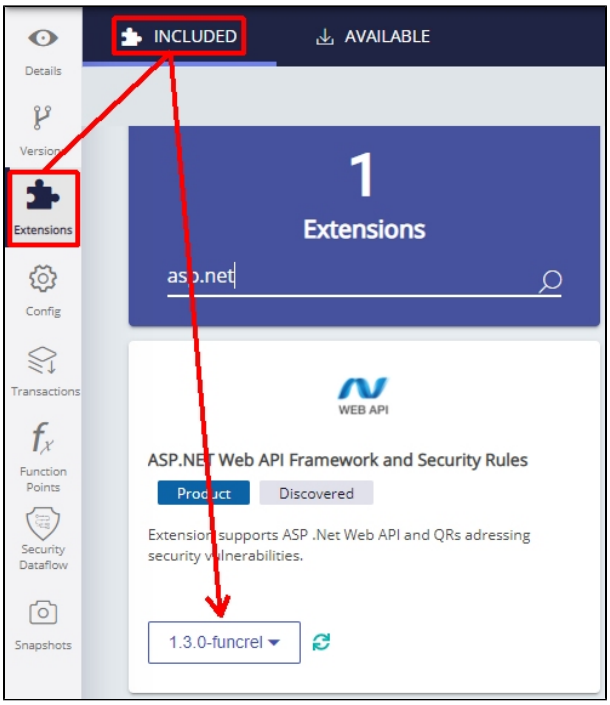
Prerequisites

✅	An installation of any compatible release of CAST AIP (see table above)

Download and installation instructions

A specific version of the ASP.NET Web API Framework extension is shipped with AIP Core. However, this release may not be the release you want to use, therefore you should check before beginning the analysis (i.e. by performing an **Advanced onboarding**) that the correct extension release is being used. You can see the list of shipped extensions for each release of AIP Core here: **Technology coverage changes in CAST AIP 8.3.x**.

If you need to change the release use the **Included** interface in AIP Console:

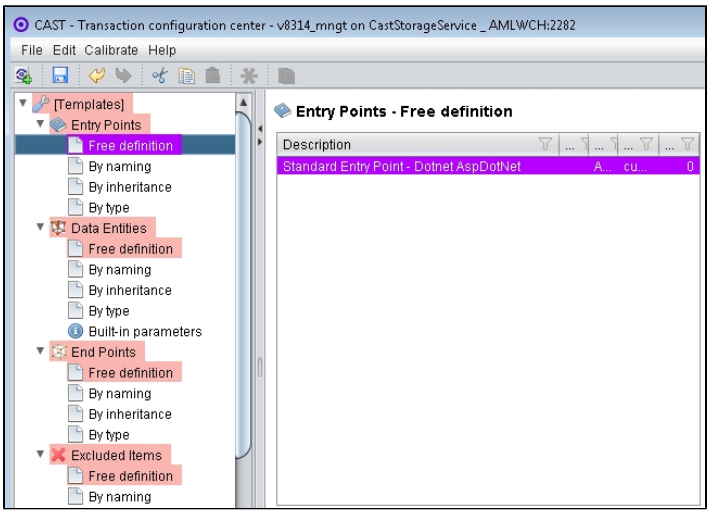


There is nothing further to do. Follow the instructions below to run a new analysis/snapshot to generate new results:

- **Advanced onboarding - run and validate the initial analysis**
- **Advanced onboarding - snapshot generation and validation**

CAST Transaction Configuration Center (TCC) configuration

If you are using the extension with **CAST AIP 8.3.x**, a set of ASP.NET WebAPI specific **items** are now **automatically imported** when the extension is installed. These items will be available in the CAST Transaction Configuration Center (click to enlarge):



Packaging, delivering and analyzing your source code

Once the extension is installed, no further configuration changes are required before you can package your source code and run an analysis. The process of packaging, delivering and analyzing your source code does not change in any way:

- **Package and deliver** your .NET application (that includes source code which uses **ASP.NET Web API**) in the exact same way as you always have.
- **Analyze** your delivered .NET application source code in the CAST Management Studio in the exact same way as you always have - the source code which uses **ASP.NET Web API** will be detected and handled correctly.

By default, (i.e. out of the box without the ASP.NET Web API extension installed) ASP.NET Web API object types are automatically “captured” by a default configuration provided by the HTML5/JavaScript extension (“Standard Entry Point - HTML5 AspDotNet”). After installation of the ASP.NET WebAPI extension you will find that the "Standard Entry Point - HTML5 AspDotNet" set no longer captures any objects. Instead the ASP.NET objects will be captured by the "Standard Entry Point - Dotnet AspDotNet" set provided in the ASP.NET WebAPI extension. Therefore you need to update TCC configuration if you are using the "Standard Entry Point - HTML5 AspDotNet" configuration in your sets and layers.

What results can you expect?

Once the analysis/snapshot generation has completed, you can view the results in the normal manner. The following **objects** and **links** will be displayed in CAST Enlighten:

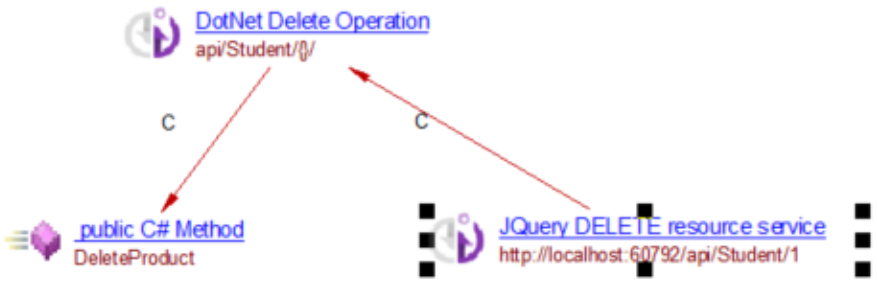
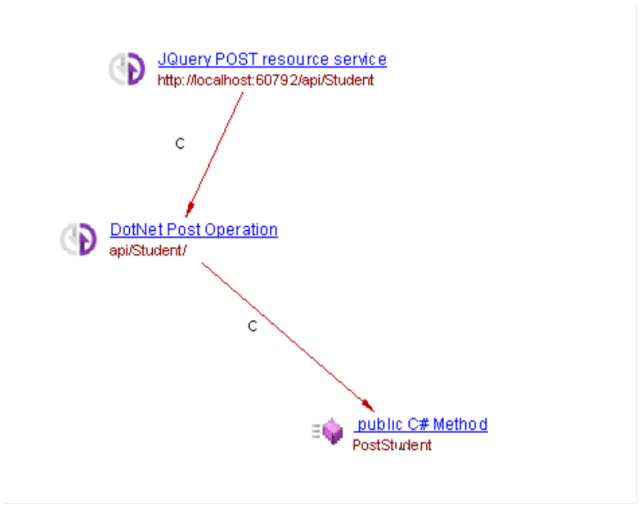
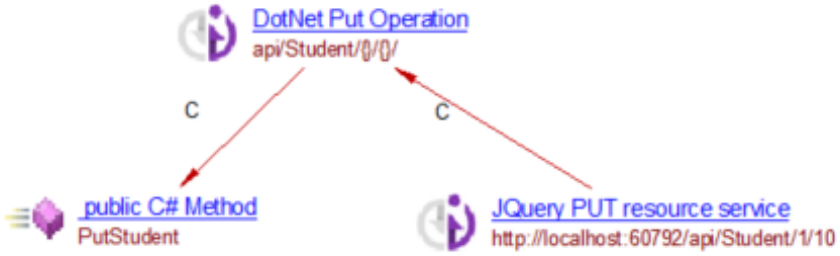
Objects

All objects are represented under the **Class browser** folders in CAST Enlighten:

Icon	Description
	.NET Get Operation
	.NET Delete Operation
	.NET Post Operation
	.NET Put Operation

Links

Source	Link type	Target	Example
Client call	Call	.NET Get Operation	

Client call	Call	.NET Delete Operation	
Client call	Call	.NET Post Operation	
Client call	Call	.NET Put Operation	

ASHX/ASMX support

WebHandle/ProcessRequest

In ashx/asmx file:

```
<%@ WebHandler Language="vb" CodeBehind="IISHandler1.vb" class="WebApplication1.IISHandler1" %>
```

In IISHandler1.vb:

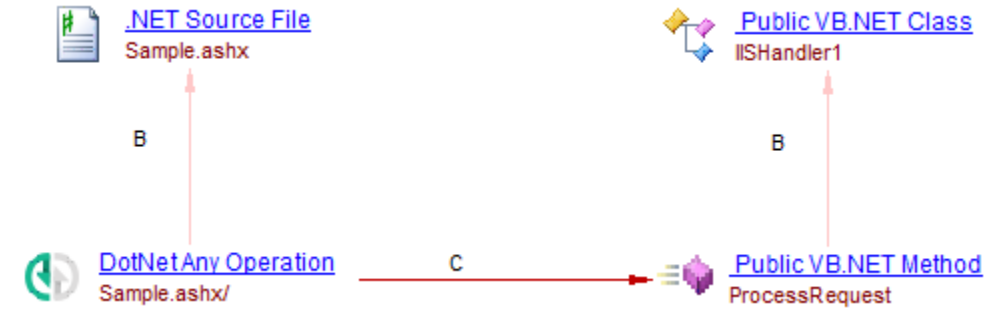
```
Imports System.Web
Public Class IISHandler1
    Implements IHttpHandler

    Public Sub ProcessRequest(ByVal context As HttpContext) Implements IHttpHandler.ProcessRequest

        ' Write your handler implementation here.

    End Sub
End Class
```

Will create an operation:



WebService/WebMethod

In asmx file:

```
<%@ WebService Language="vb" CodeBehind="WebService1.asmx.vb" class="WebApplication1.WebService1" %>
```

In vb file:

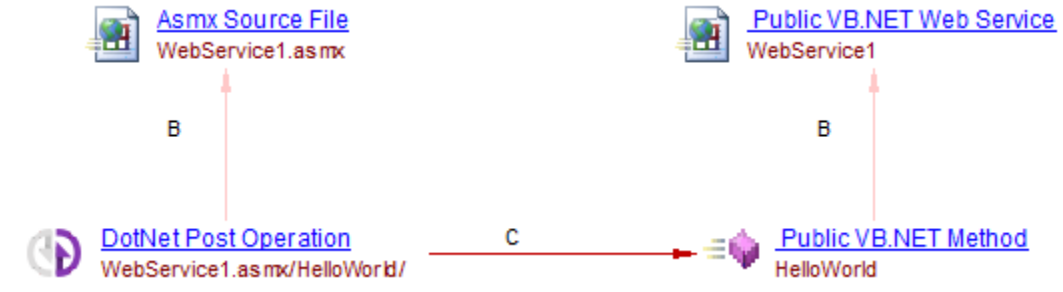
```
Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports System.ComponentModel

' To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
' <System.Web.Script.Services.ScriptService()> _
<System.Web.Services.WebService(Namespace:="http://tempuri.org/")> _
<System.Web.Services.WebServiceBinding(ConformsTo:=WsiProfiles.BasicProfile1_1)> _
<ToolboxItem(False)> _
Public Class WebService1
    Inherits System.Web.Services.WebService

    <WebMethod()> _
    Public Function HelloWorld() As String
       Return "Hello World"
    End Function

End Class
```

Will create an operation for each WebMethod annotated methods:



Rules

1.1.0-funcrel	https://technologies.castsoftware.com/rules?sec=srs_dotnetweb&ref=||1.1.0-funcrel
1.1.0-beta1	https://technologies.castsoftware.com/rules?sec=srs_dotnetweb&ref=||1.1.0-beta1
1.1.0-alpha3	https://technologies.castsoftware.com/rules?sec=srs_dotnetweb&ref=||1.1.0-alpha3
1.1.0-alpha2	https://technologies.castsoftware.com/rules?sec=srs_dotnetweb&ref=||1.1.0-alpha2
1.1.0-alpha1	https://technologies.castsoftware.com/rules?sec=srs_dotnetweb&ref=||1.1.0-alpha1

Limitations

In this section we list the most significant functional limitations that may affect the analysis of applications using ASP.NET Web API:

- Currently support only exists for the following objects:
  - HttpGet
  - HttpPut
  - HttpDelete
  - HttpPost
- URL parameters on HttpPost operation not detected
- RoutingPath with parameters in the middle instead of end not supported
- Combination of Route annotation and Http[Post|Get|Put|Delete] annotations creates more links than expected