Soap Web Service Using APL64 Cross-Platform Component

Contents

Overview 2
SOAP Message Structure 2
SOAP Web Service Example 2
Download the Example Source Code 3
SOAP Web Service Visual Studio Project 3
Create the SoapWebService Project in Visual Studio 3
Configure the SoapWebService project 3
Supply the Additional Project Information 4
Install SoapCore NuGet package into the SoapWebService project 4
Remove Unnecessary Project Template Information 5
Create Project Folders 5
Create SoapService Class 6
Replace the code in the Program.cs code file 7
APL64 Cross-Platform Component (CPC) 9
Prepare an APL64 Workspace with an APL64 Public Function: 9
Test the APL64 Public Function 10
Use the APL64 Cross-platform Component Utility 10
Use the 'Load CPC Info' button 11
Update the 'CPC Workspace Path' and 'Cross Platform Component Target Folder' Entries 11
Update the Nuget Package Guid 11
Save the CPC Configuration File 12
Create the Cross-platform Component 12
Copy the APL64 CPC Nuget Package to a Local Nuget Publish Folder 13
Create the 'c:\ Local Nuget Publish Folder\' 13
In VS2022 Recognize the Local Nuget Publish Folder 14
Install APL64 CPC Nuget package into SoapWebService 15
Start the SoapWebService 15
Debug the solution in VS2022 to start the web service 15

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 1

In a web browser display the WSDL for the SOAP web service....16
Browser Client to Access Access SoapWebService....17
Add a new 'Console' project to the solution called SquareRootSoapRequest....17
Configure the Client project....17
Provide the Additional information for the Client project....18
Replace Program.cs code file of Client project with this code:....19
Make a Client Request to the SoapWebService....20
Request Square Root using HttpClient....21
More Information....26

Overview

- Create a SOAP web service with algorithmic support provided by an APL64 cross-platform component.
- Access the SOAP web service via a SOAP browser client.
- A SOAP web service uses the Simple Object Access Protocol to specify messages which can be transmitted between a web server and web server client. The SOAP protocol is inherently cross-platform because message content is rendered in XML format. XML format content is self-documenting since it is composed of descriptive tags and data enclosed within tag delimiters.

SOAP Message Structure

A SOAP message is an XML-format document containing these elements:

- Envelope identifying the XML document as a SOAP message
- Header containing 'header' information
- Body containing request or response information
- Fault containing error and status information

SOAP Web Service Example

This document provides instructions for creating a simple SOAP web service using the http protocol to transmit SOAP messages between a web client and web server.

- The web service is created in Microsoft Visual Studio using the C# programming language
- The web service can be hosted on .Net compatible platforms such as Windows and Linux
- The functionality of the web service is provided by an APL64 cross-platform component

© APLNext, LLC - All Rights Reserved.

Download the Example Source Code

Download the file: https://aplnow.com/APL64/UserDocumentation/APL64_CPC_IN_SOAP_SVC.zip

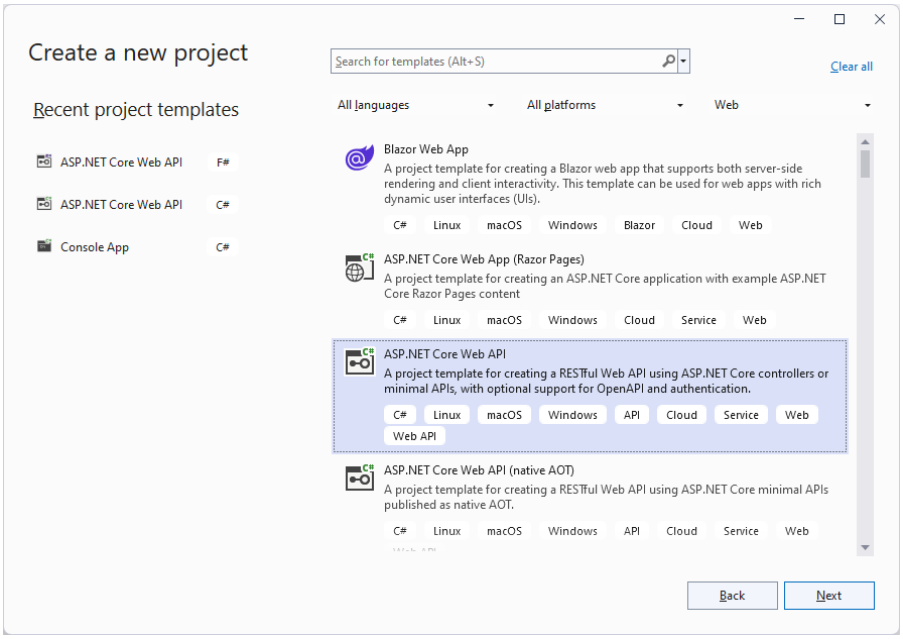
Extract the file to ‘c:\’

Follow the instructions in the remainder of this document to create the APL64 CPC and run the solution in Visual Studio.

SOAP Web Service Visual Studio Project

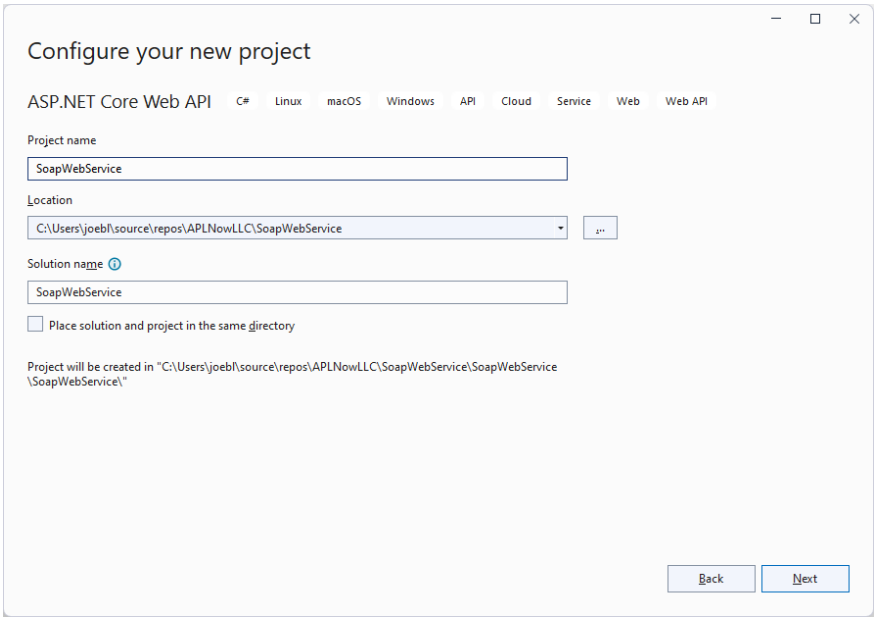
Create the SoapWebService Project in Visual Studio

Select the ASP.Net Core Web API project template for C#:

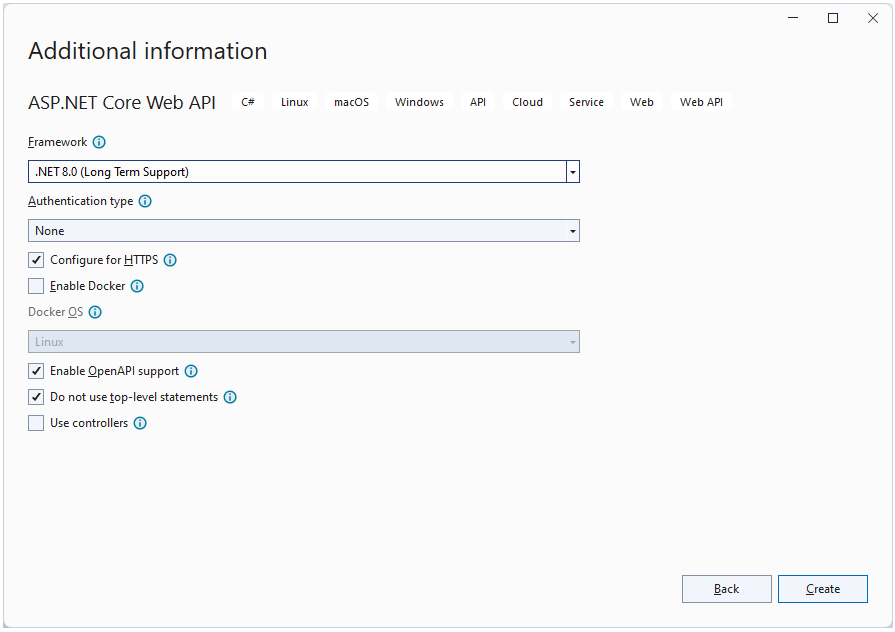


Configure the SoapWebService project

© APLNext, LLC - All Rights Reserved.

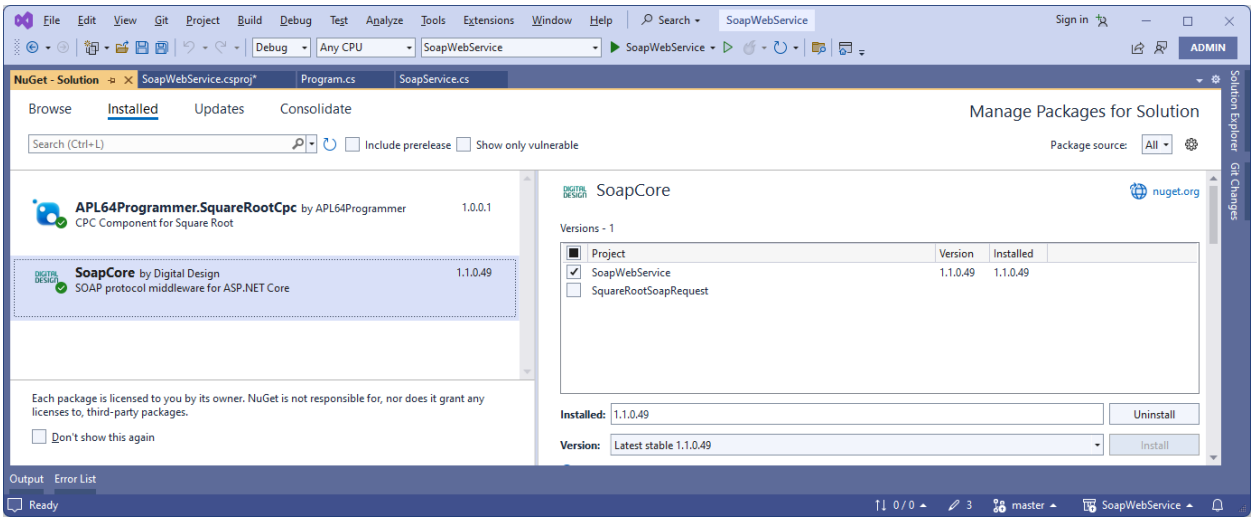


Supply the Additional Project Information



Install SoapCore NuGet package into the SoapWebService project

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 4



Remove Unnecessary Project Template Information

Close the 'ASP.Net Core' introductory page tab.

Delete the WeatherForecast.cs code file from the project

Delete the Properties folder from the project

Uninstall the Swashbuckle.AspNetCore Nuget package

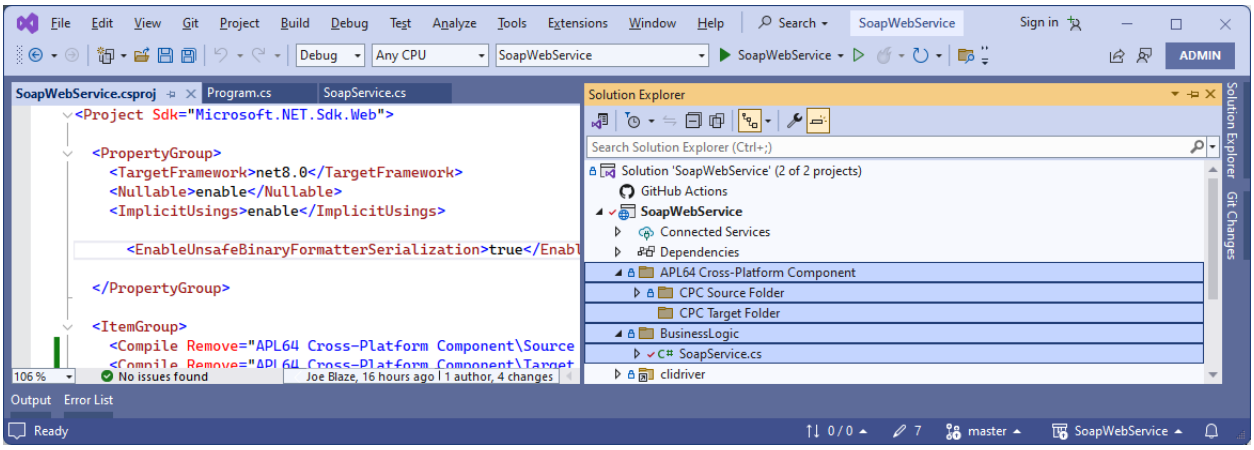
Uninstall the Microsoft.AspNetCore.OpenApi Nuget package.

Create Project Folders

'BusinessLogic' folder

'APL64 Cross-Platform Component' folder and sub-folders:

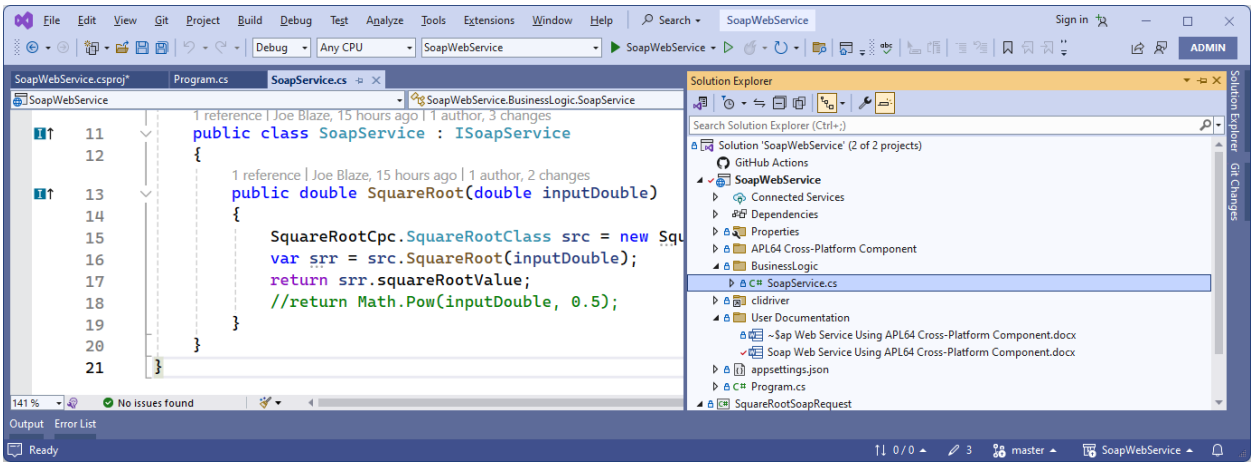
- 'CPC Source Folder'
- 'CPC Target Folder'



© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 5

Create SoapService Class

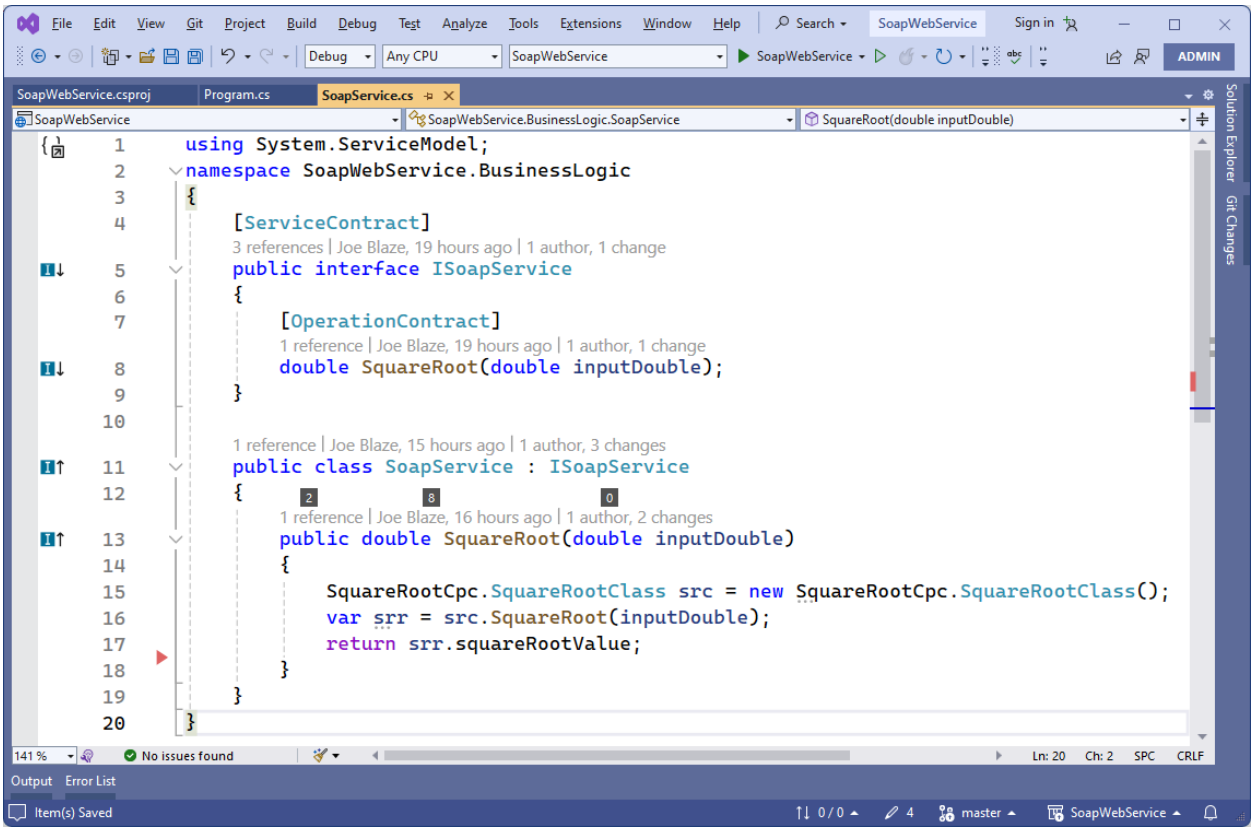
Create a SoapService.cs code file in the BusinessLogic folder:



Replace the code in the SoapService.cs code file with this code:

```
using System.ServiceModel;
namespace SoapWebService.BusinessLogic
{
  [ServiceContract]
  public interface ISoapService
  {
    [OperationContract]
    double SquareRoot(double inputDouble);
  }

  public class SoapService : ISoapService
  {
    public double SquareRoot(double inputDouble)
    {
      SquareRootCpc.SquareRootClass src = new SquareRootCpc.SquareRootClass();
      var srr = src.SquareRoot(inputDouble);
      return srr.squareRootValue;
    }
  }
}
```

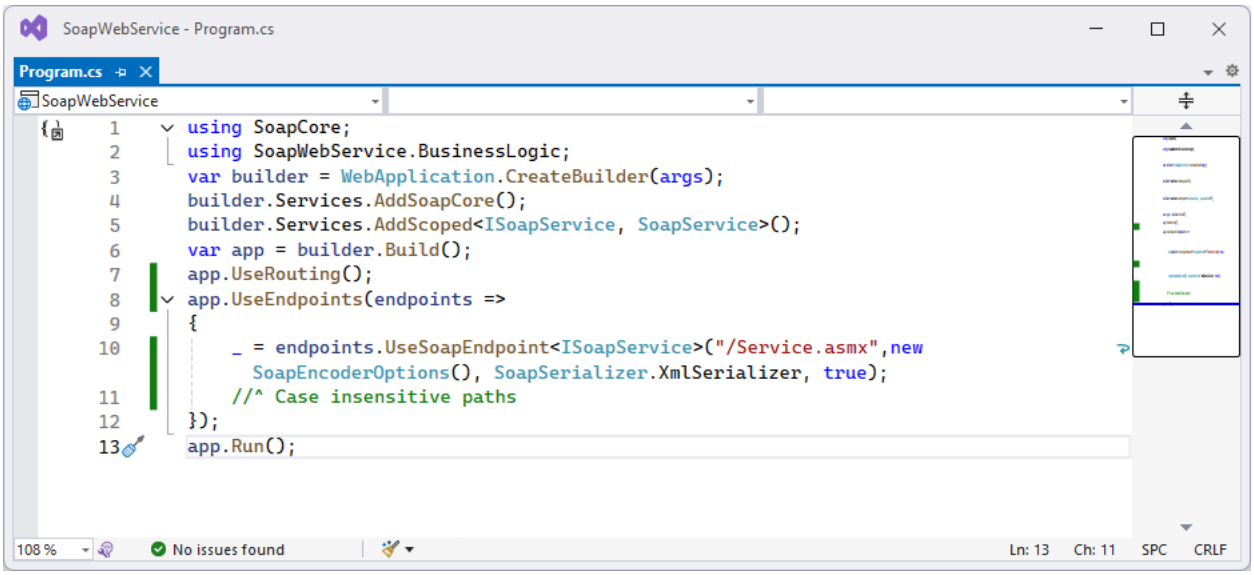
© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 6



Replace the code in the Program.cs code file

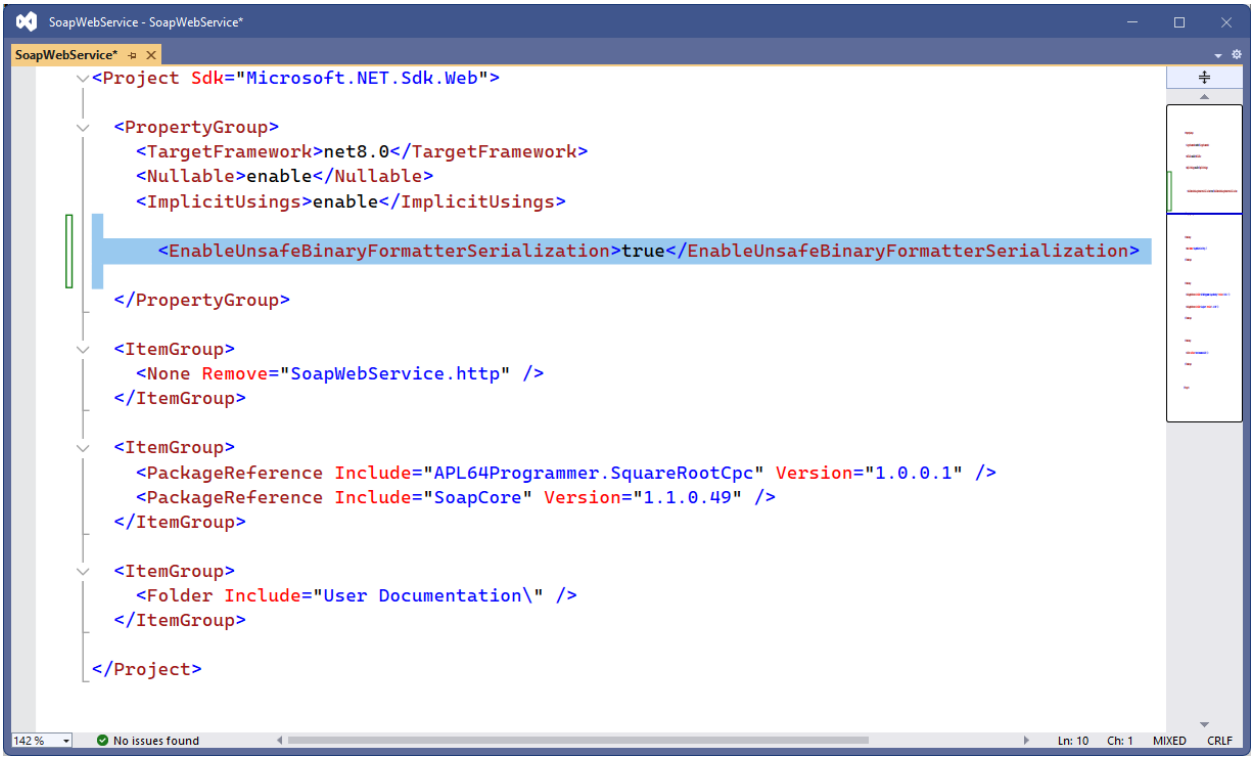
```
using SoapCore;
using SoapInNetCore.BusinessLogic;
var builder = WebApplication.CreateBuilder(args);
builder.Services.AddSoapCore();
builder.Services.AddScoped<ISoapService, SoapService>();
var app = builder.Build();
app.UseRouting();
app.UseEndpoints(endpoints =>
{
  _=endpoints.UseSoapEndpoint<ISoapService>("/Service.asmx",
new SoapEncoderOptions(), SoapSerializer.XmlSerializer, true);
});
//^ Case-insensitive paths
app.Run();
```

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 7



Add the 'binary formatter' override to the SoapWebService.proj project file, so that the APL64 CPC embedded workspace can be loaded into the project:

```
<EnableUnsafeBinaryFormatterSerialization>true</EnableUnsafeBinaryFormatterSerialization>
```



© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 8

APL64 Cross-Platform Component (CPC)

For purposes of this document the APL64 CPC implementation files are contained in the 'APL64 Cross-Platform Component' folder of the SoapWebService solution.

Prepare an APL64 Workspace with an APL64 Public Function:

An APL64 programmer-defined, public function will support the square root algorithm in the web service. The 'APL64 Cross-Platform Component\CPC Source Folder\SquareRootCPC.ws64' workspace contains the required public function.

The APL64 public function, SquareRoot, has a single right argument, inputDouble, specified as a Double numeric value and three result values:

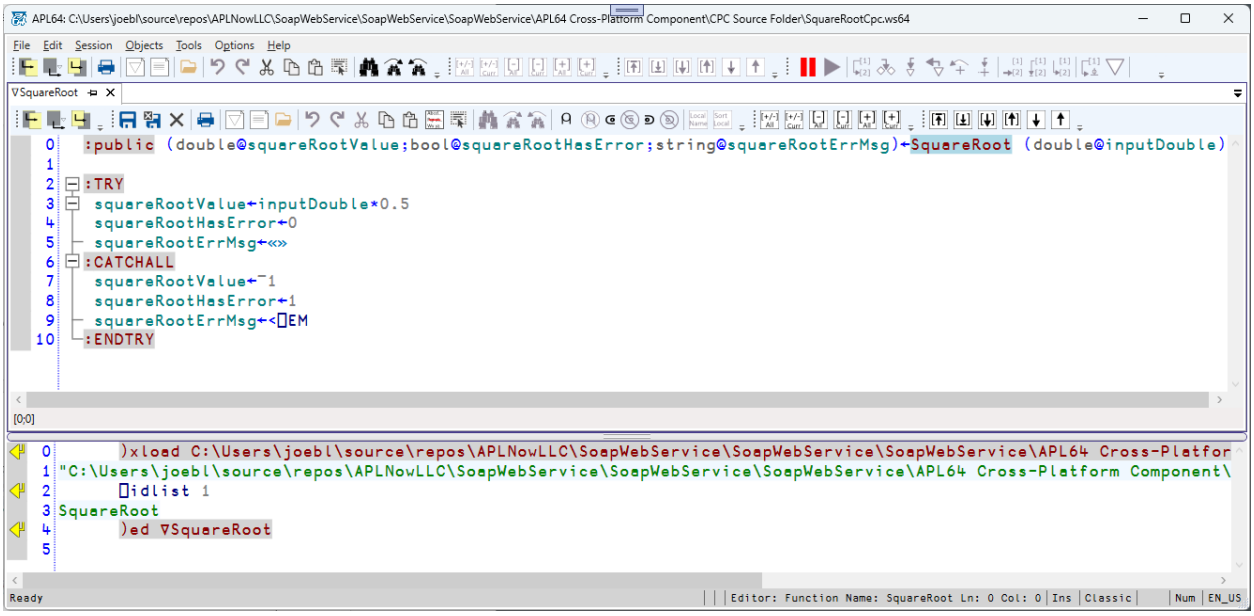
- squareRootValue, specified as a Double numeric value
- squareRootHasError, specifies as a Boolean value
- squareRootErrorMsg, specified as a String value

When the calculation of the squareRootValue occurs without an exception, the squareRootErrMsg is assigned an empty String, «». If the calculation of the squareRootValue results in an exception, the squareRootErrMsg is assigned the string value of the APL64 error message system variable, <⎕EM.

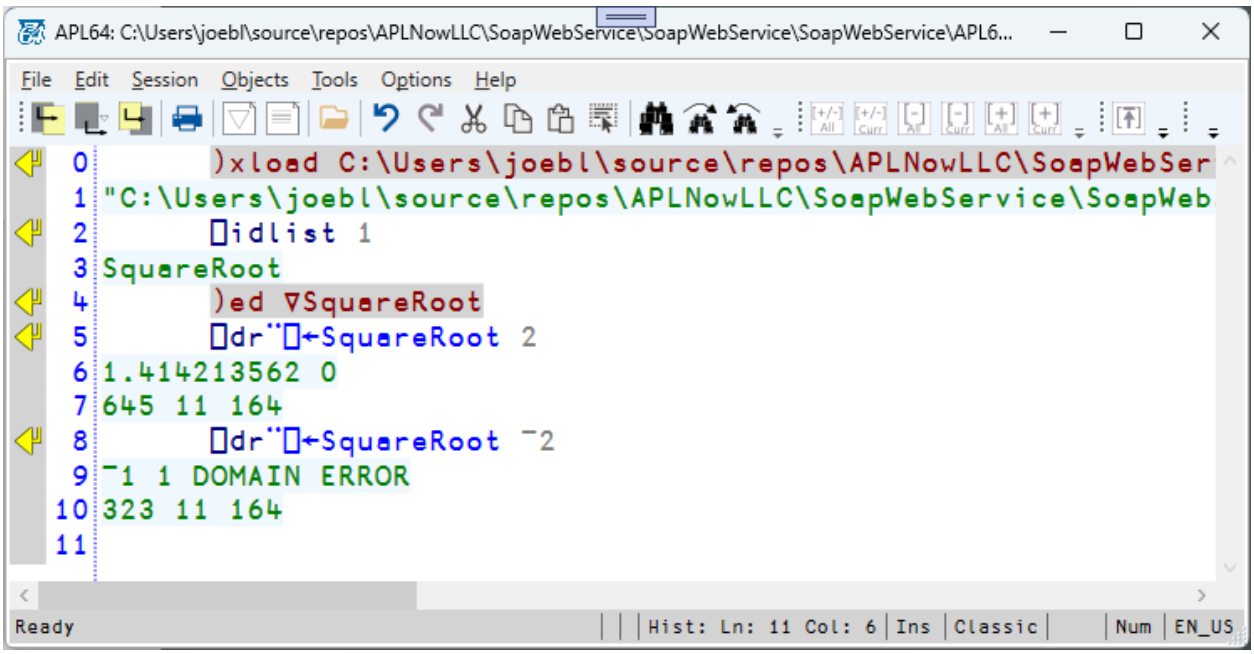
```
:public
(double@squareRootValue;bool@squareRootHasError;string@squareRootErrMsg)←SquareRoot
(double@inputDouble)

:TRY
 squareRootValue←inputDouble*0.5
 squareRootHasError←0
 squareRootErrMsg←«»
:CATCHALL
 squareRootValue←¯1
 squareRootHasError←1
 squareRootErrMsg←<⎕EM
:ENDTRY
```

In the APL64 developer version, load the 'APL64 Cross-Platform Component\CPC Source Folder\SquareRootCPC.ws64' workspace:

© APLNext, LLC - All Rights Reserved.



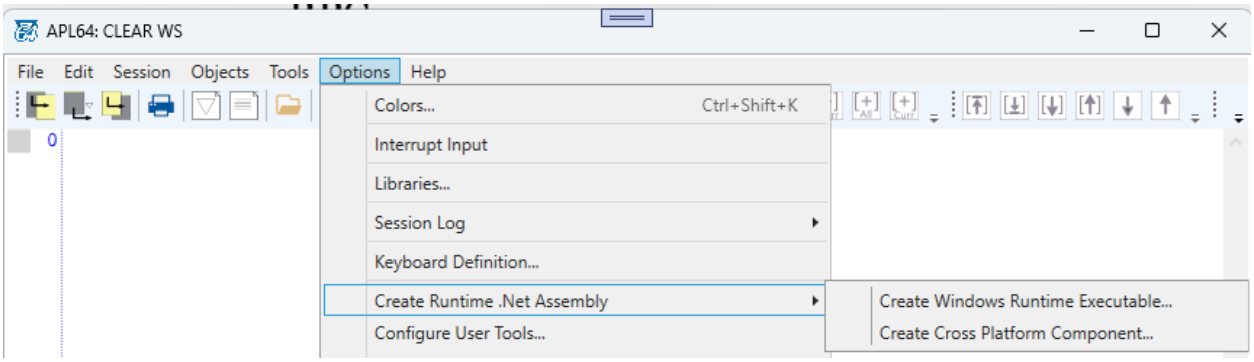
Test the APL64 Public Function



Use the APL64 Cross-platform Component Utility

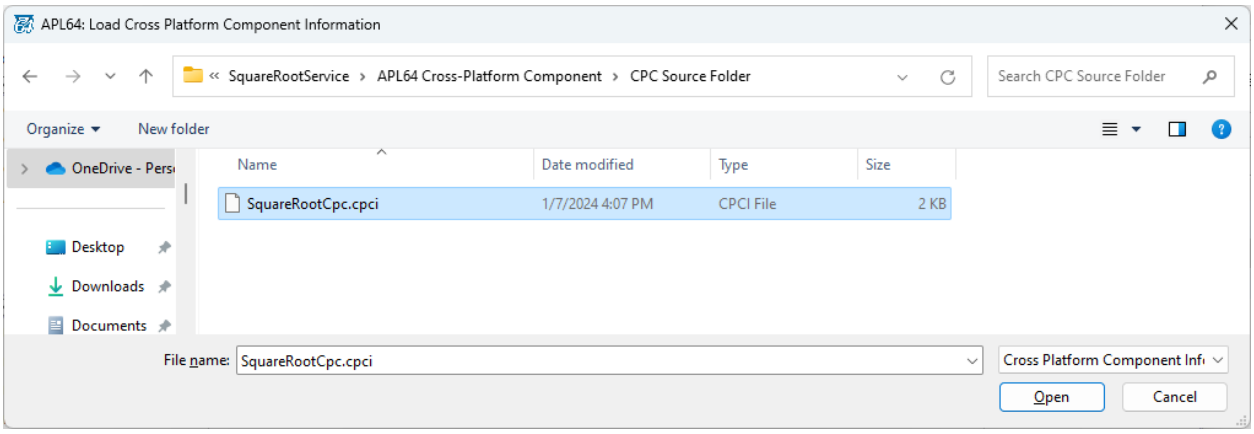
Use Options | Create Runtime .Net Assembly | Create Cross-Platform Component utility in the APL64 Developer version:

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 10



Use the 'Load CPC Info' button

Load the 'APL64 Cross-Platform Component\CPC Source Folder\SquareRootCpc.cpci' CPC configuration info file:



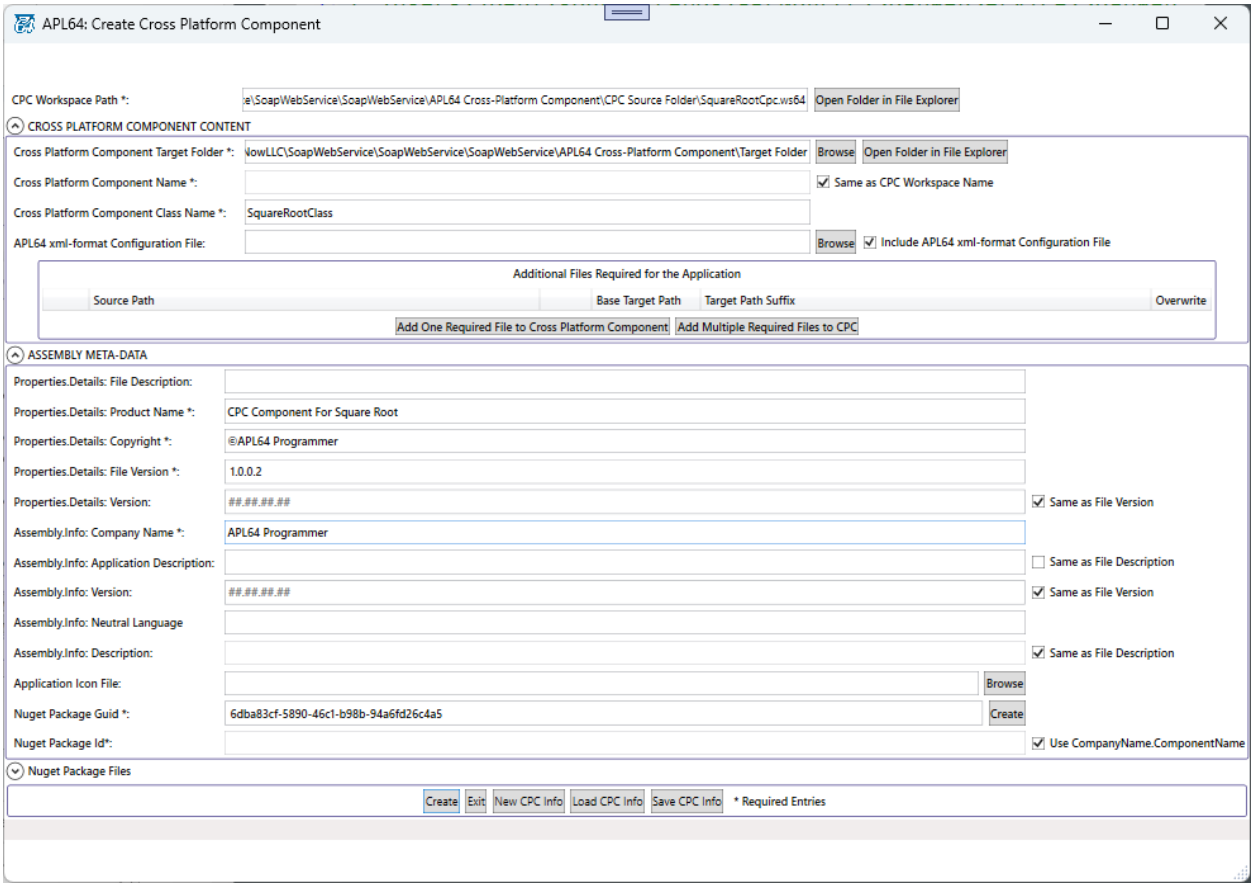
Update the 'CPC Workspace Path' and 'Cross Platform Component Target Folder' Entries

The entries should be modified as appropriate for the location of the SquareRoot service solution on the APL64 programmer's workstation.

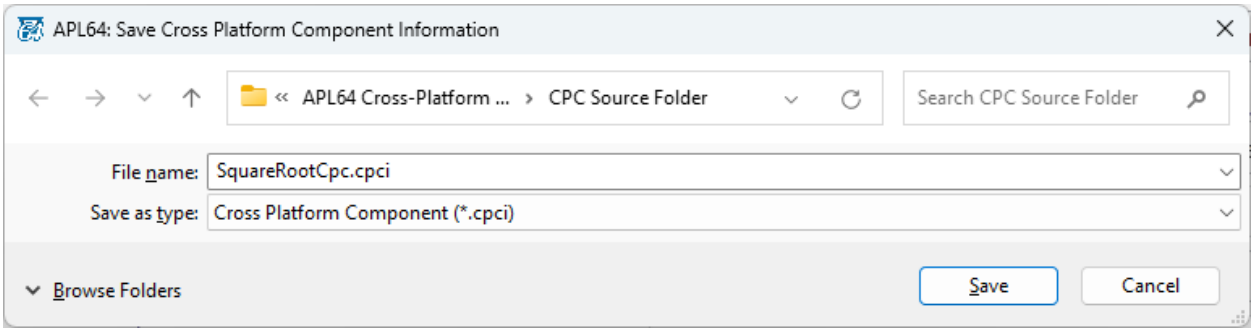
Update the Nuget Package Guid

Use the 'Create' button to the right of this entry in the dialog:

© APLNext, LLC - All Rights Reserved.



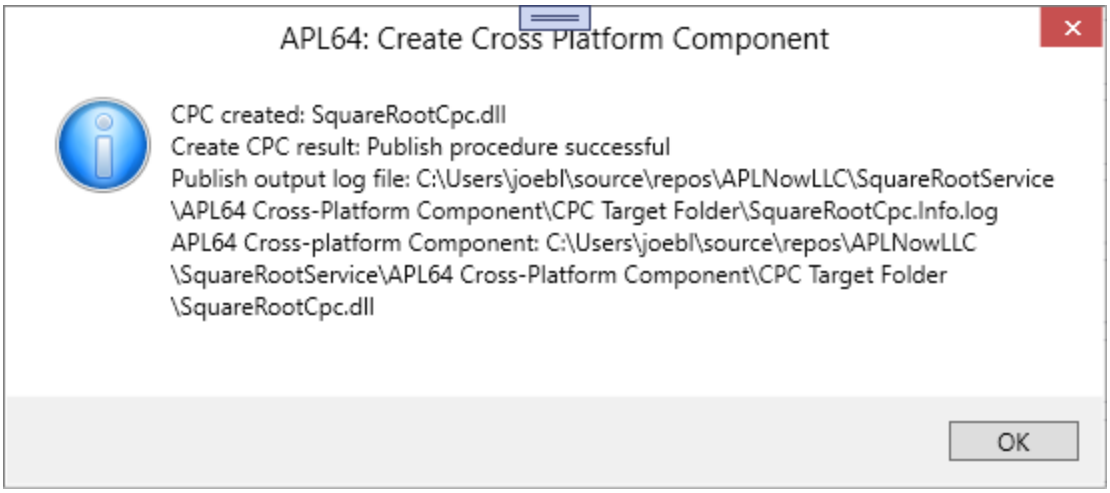
Save the CPC Configuration File



Create the Cross-platform Component

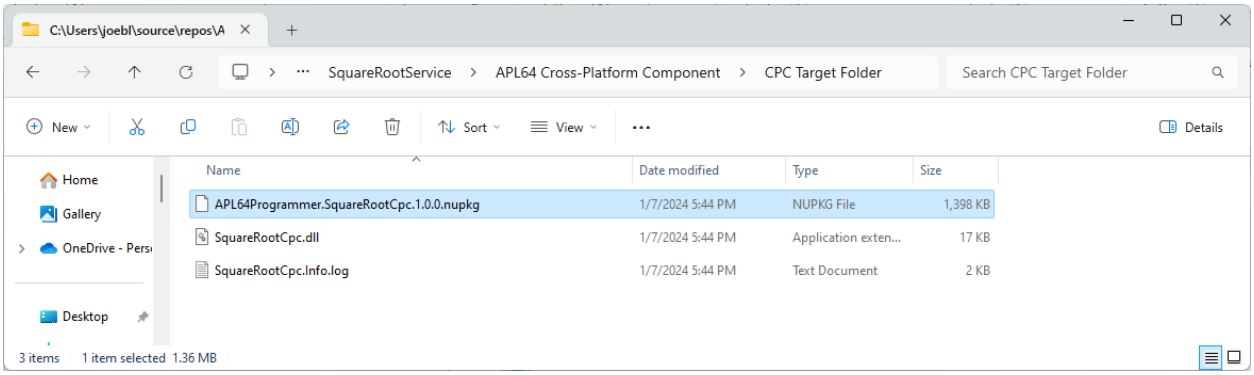
Click the Create button and observe the successful creation of the CPC:

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 12



Copy the APL64 CPC Nuget Package to a Local Nuget Publish Folder

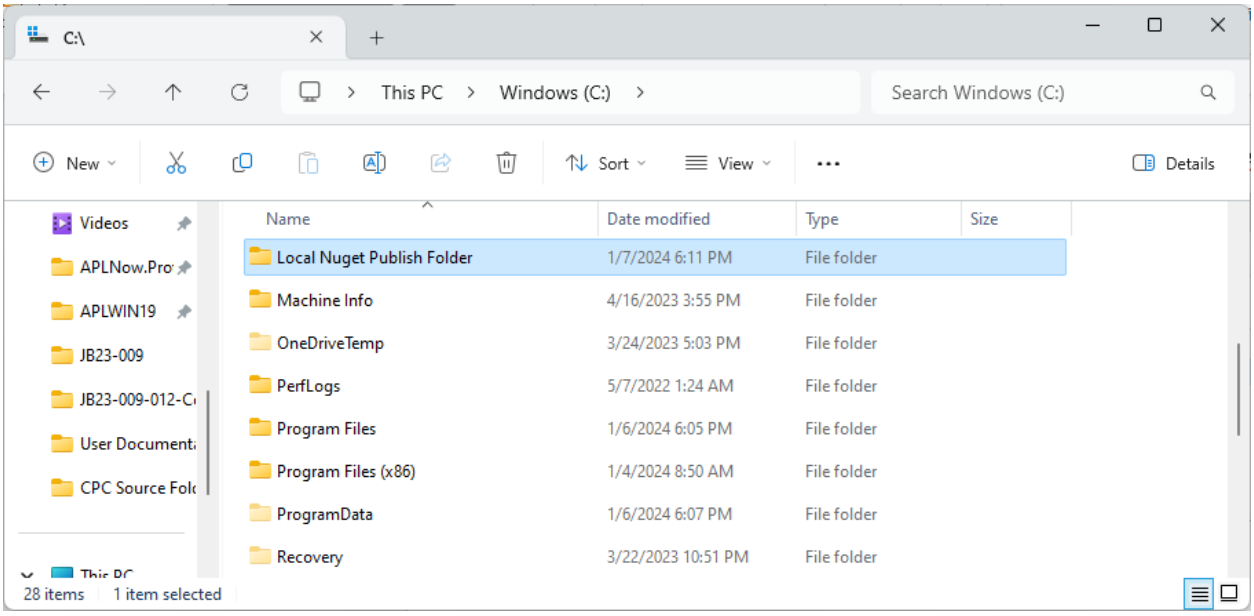
Copy the 'APL64Programmer.SquareRootCpc.1.0.0.nupkg' Nuget package from the CPC Target Folder:



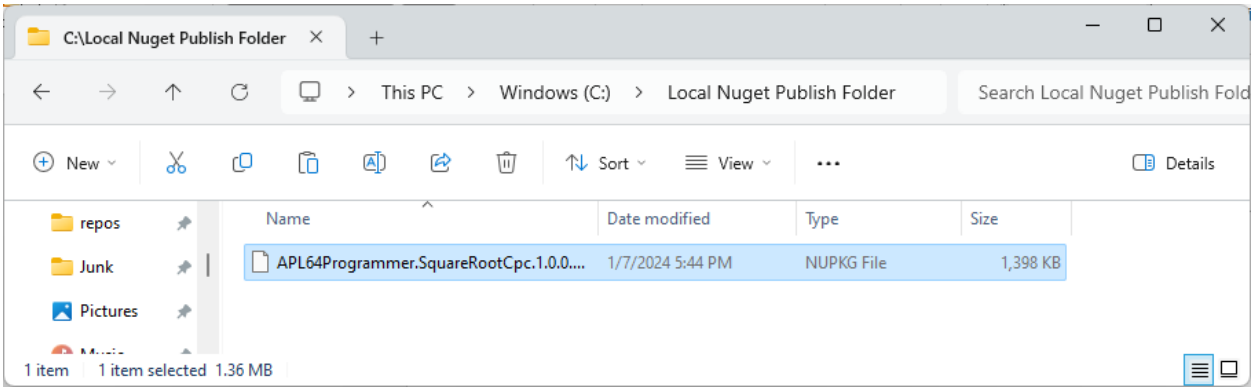
Create the 'c:\ Local Nuget Publish Folder\'

This folder should not be in the SoapWebService solution folder.

© APLNext, LLC - All Rights Reserved.



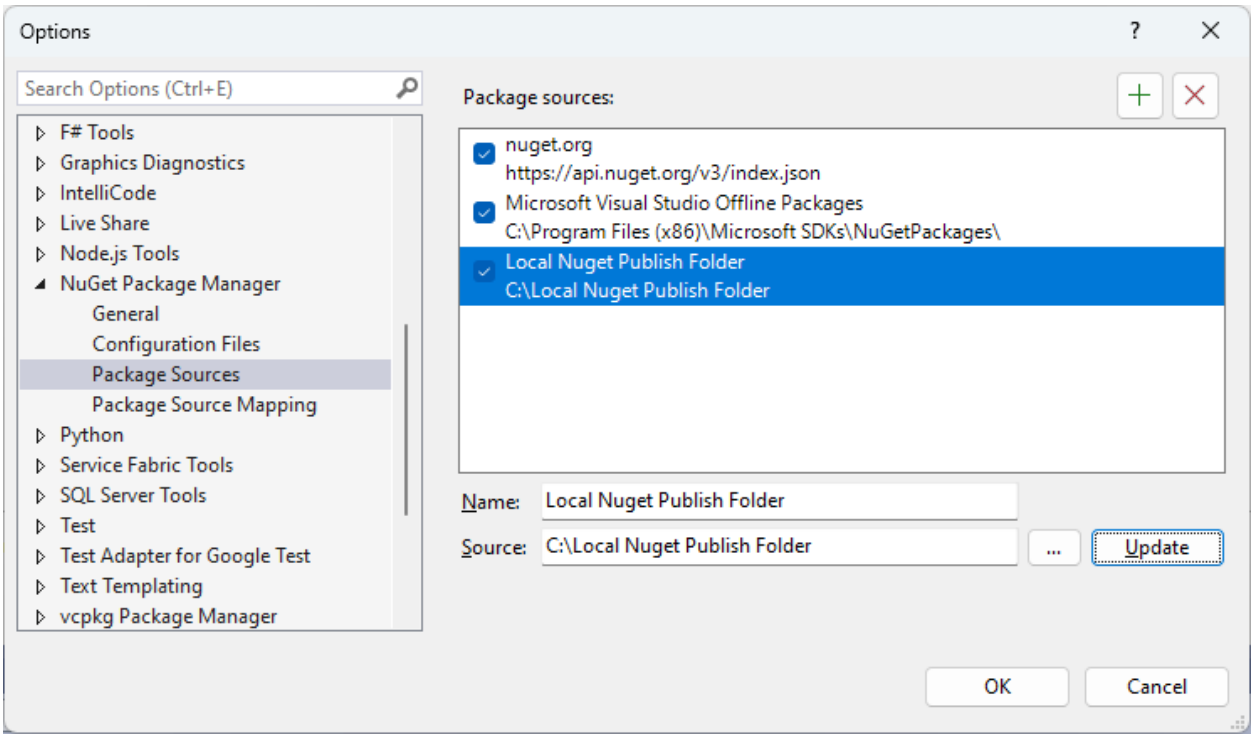
Paste the 'APL64Programmer.SquareRootCpc.1.0.0.nupkg' Nuget package to the 'c:\ Local Nuget Publish Folder\':



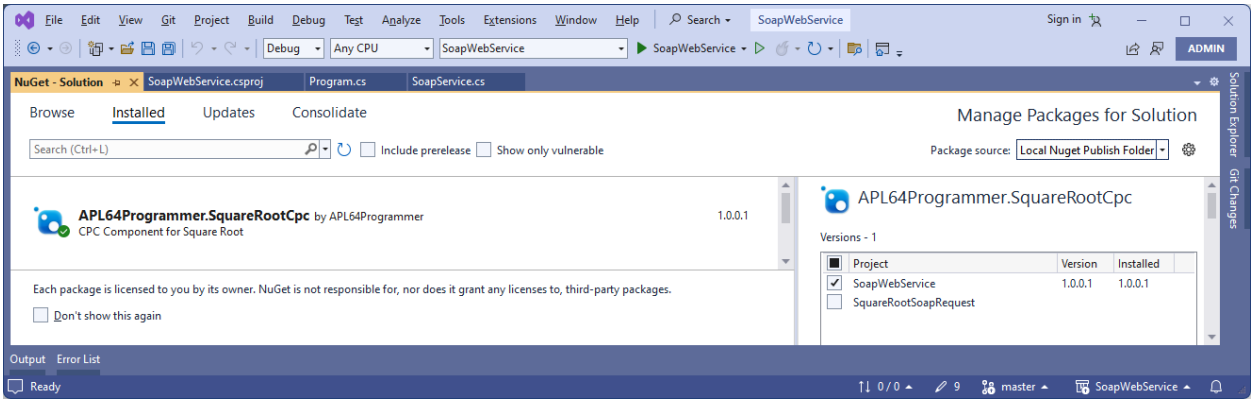
In VS2022 Recognize the Local Nuget Publish Folder

In Visual Studio use: Tools | Nuget Package Manager | Package Manager Settings | Package Sources | + (Add package source) to specify the Local Nuget Publish folder:

© APLNext, LLC - All Rights Reserved.



Install APL64 CPC Nuget package into SoapWebService



Start the SoapWebService

Debug the solution in VS2022 to start the web service

The URL and port for the http and https web transfer protocols are provided for use by a client of the web service.

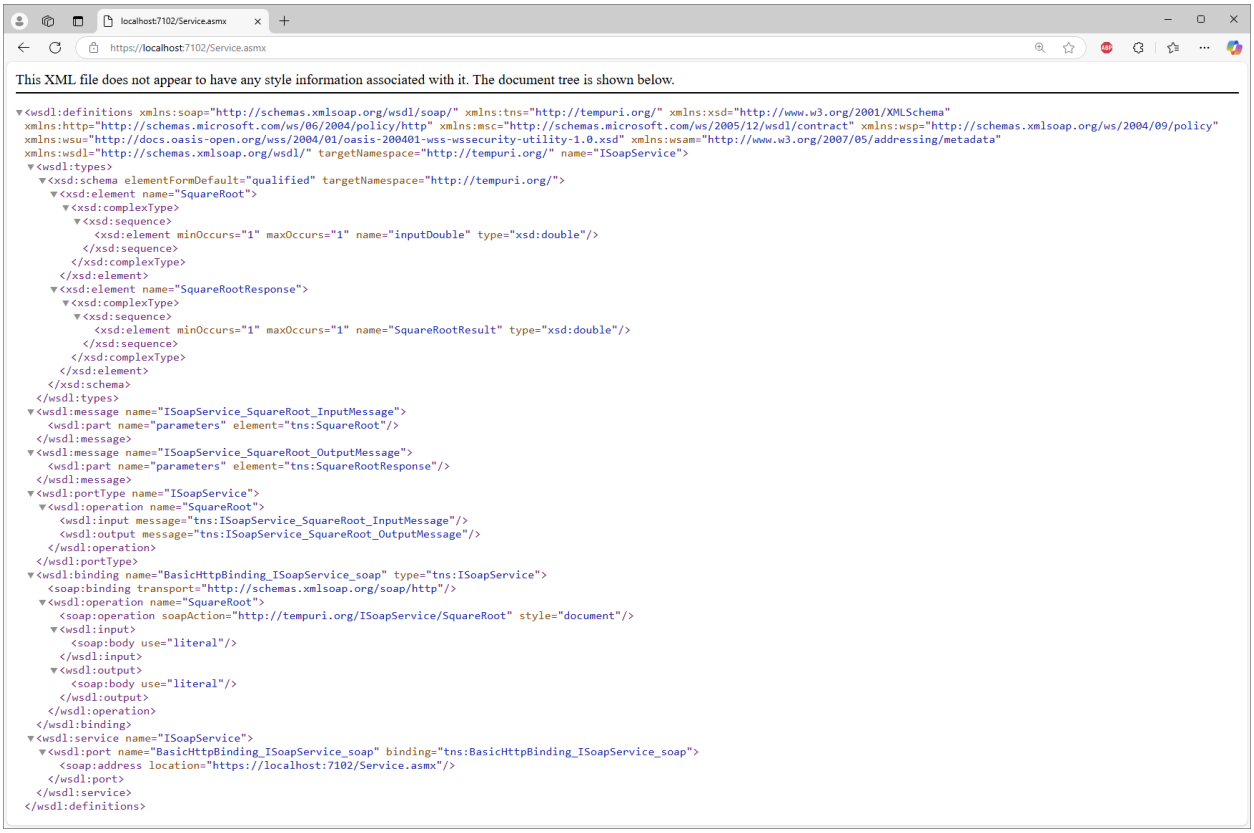
© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 15

The default web browser on the target workstation will open and display a 404 web exception, which is to be expected because the web service has no default web page.

In a web browser display the WSDL for the SOAP web service

The WSDL is a description of the web service which can be used by clients to access the web service. The WSDL uses the Web Service Description Language.

Using the web browser go to this url: https://localhost:7102/Service.asmx



© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 16

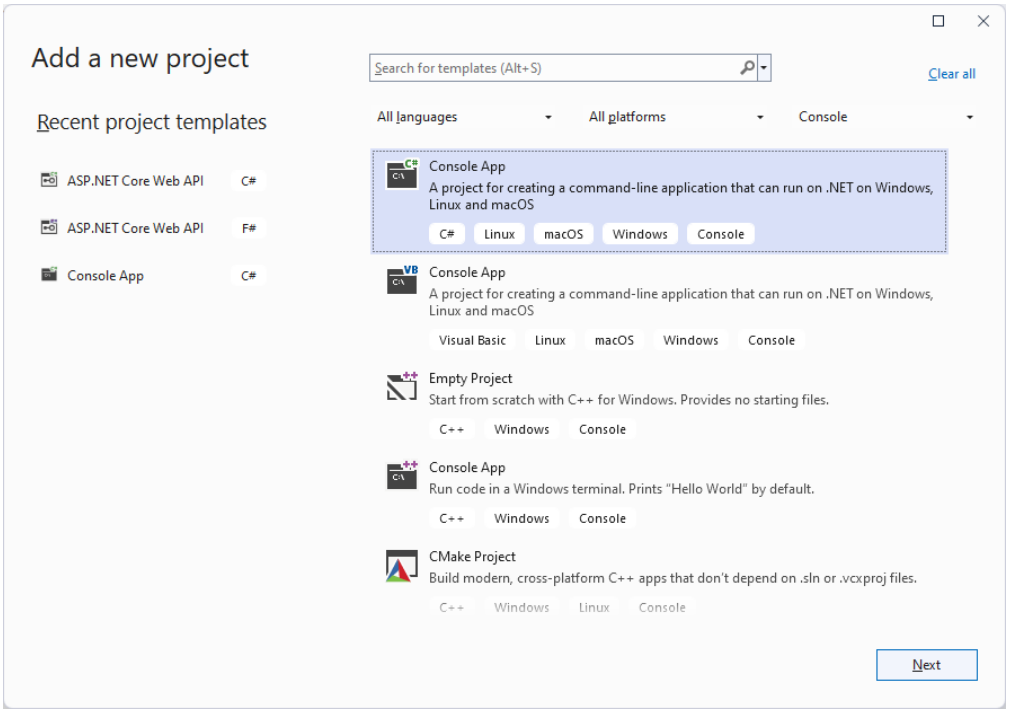
Browser Client to Access Access SoapWebService

In a production environment a client for the SoapWebService would be a server-to-server or end-use GUI application. For purposes of this example, a console project is used as the client.

Note: The SquareRootSoapRequest client described here is based on the Microsoft-deprecated 'WebRequest' technology, which has been replace by 'HttpClient' technology. The SquareRootSoapRequest client project was included in previous versions of this document.

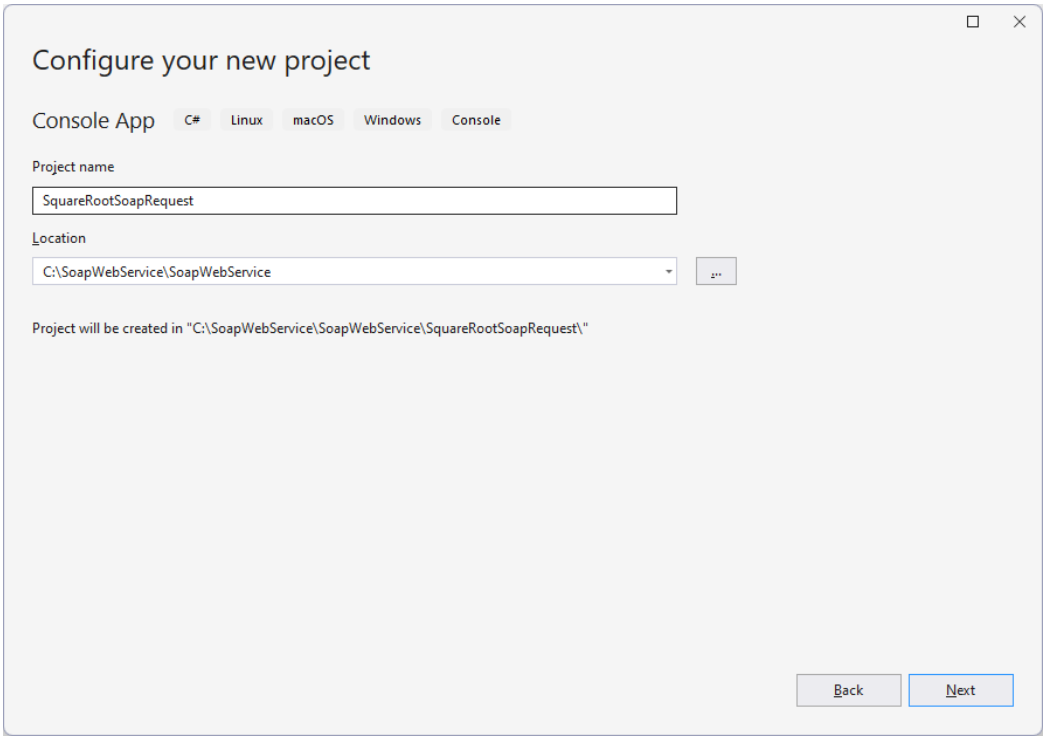
See below in this document for an up-to-date '[SoapServiceHttpClient]' project with the same functionality.

Add a new 'Console' project to the solution called SquareRootSoapRequest

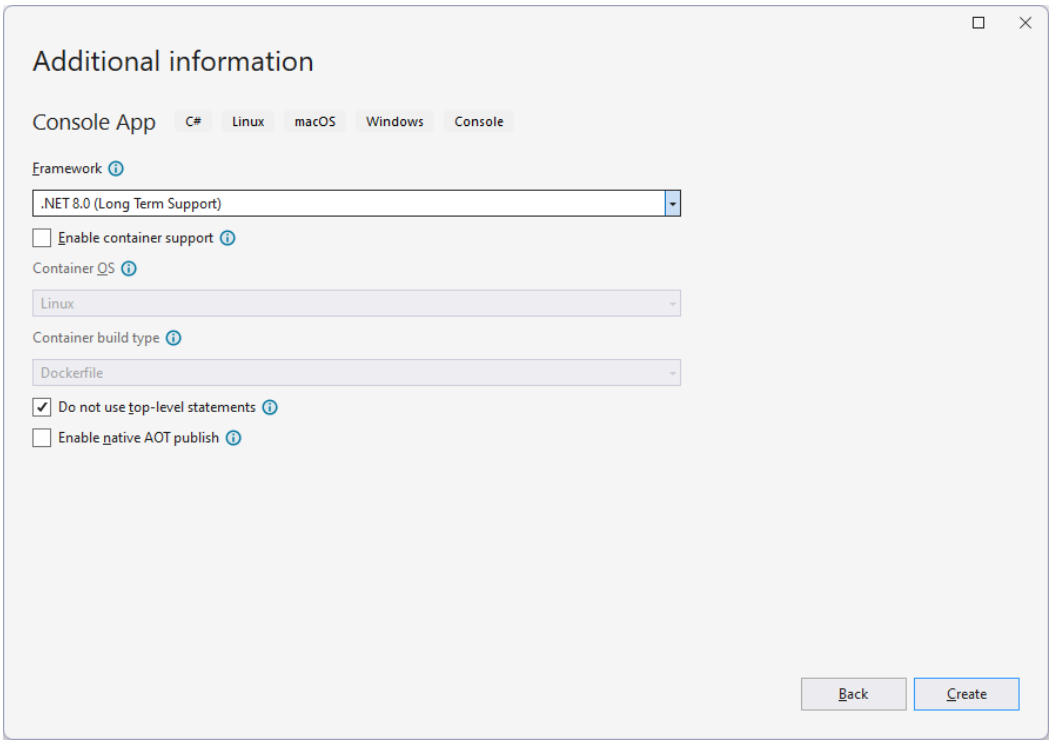


Configure the Client project

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 17



Provide the Additional information for the Client project



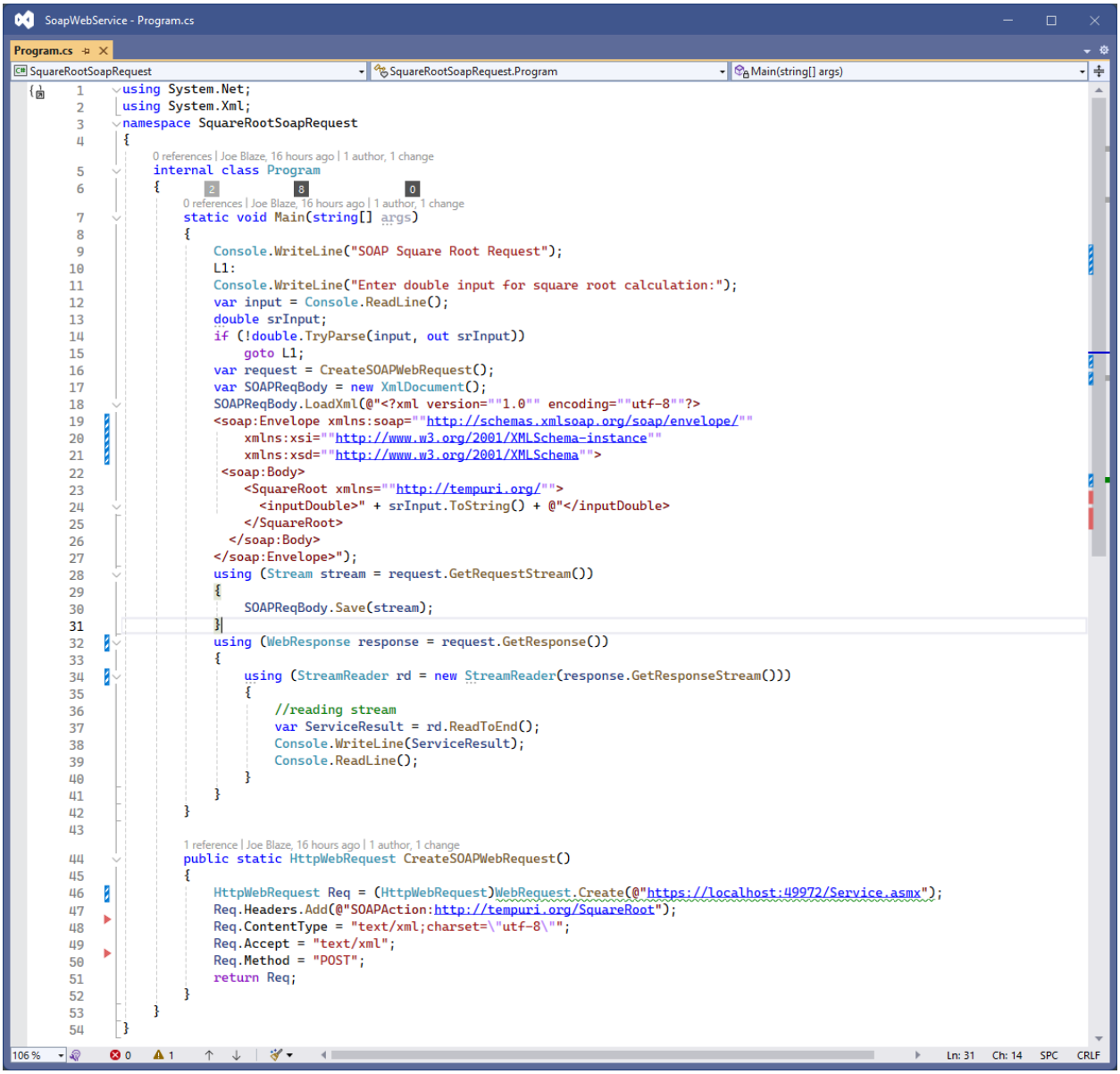
© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 18

Replace Program.cs code file of Client project with this code:

```
using System.Net;
using System.Xml;
namespace SquareRootSoapRequest
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("SOAP Square Root Request");
            L1:
            Console.WriteLine("Enter double input for square root calculation:");
            var input = Console.ReadLine();
            double srInput;
            if (!double.TryParse(input, out srInput))
                goto L1;
            var request = CreateSOAPWebRequest();
            var SOAPReqBody = new XmlDocument();
            SOAPReqBody.LoadXml(@"<?xml version=""1.0"" encoding=""utf-8""?>
            <soap:Envelope
xmlns:soap=""http://schemas.xmlsoap.org/soap/envelope/""
                xmlns:xsi=""http://www.w3.org/2001/XMLSchema-instance""
                xmlns:xsd=""http://www.w3.org/2001/XMLSchema"">
             <soap:Body>
                <SquareRoot xmlns=""http://tempuri.org/"">
                  <inputDouble>" + srInput.ToString() + @"</inputDouble>
                </SquareRoot>
              </soap:Body>
            </soap:Envelope>");
            using (Stream stream = request.GetRequestStream())
            {
                SOAPReqBody.Save(stream);
            }
            using (WebResponse response = request.GetResponse())
            {
                using (StreamReader rd = new
StreamReader(response.GetResponseStream()))
                {
                    //reading stream
                    var ServiceResult = rd.ReadToEnd();
                    Console.WriteLine(ServiceResult);
                    Console.ReadLine();
                }
            }
        }

        public static HttpWebRequest CreateSOAPWebRequest()
        {
            HttpWebRequest Req =
(HttpWebRequest)WebRequest.Create(@"https://localhost:7102/Service.asmx");
            Req.Headers.Add(@"SOAPAction:http://tempuri.org/SquareRoot");
            Req.ContentType = "text/xml;charset=\"utf-8\"";
            Req.Accept = "text/xml";
            Req.Method = "POST";
            return Req;
        }
    }
```

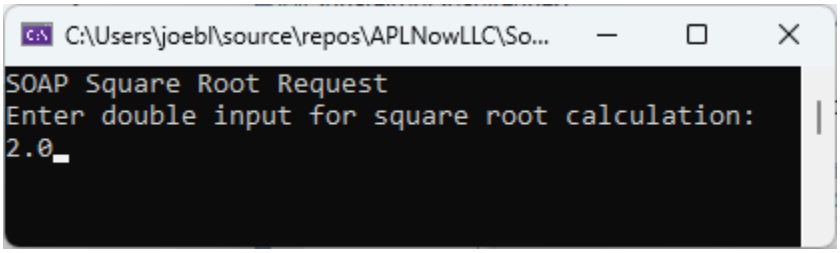
© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 19

```
}
```



Make a Client Request to the SoapWebService

With the SoapWebService running, debug the SquareRootSoapRequest client project in Visual Studio, enter the desired double input value and click the Enter/Return key to calculate its square root:

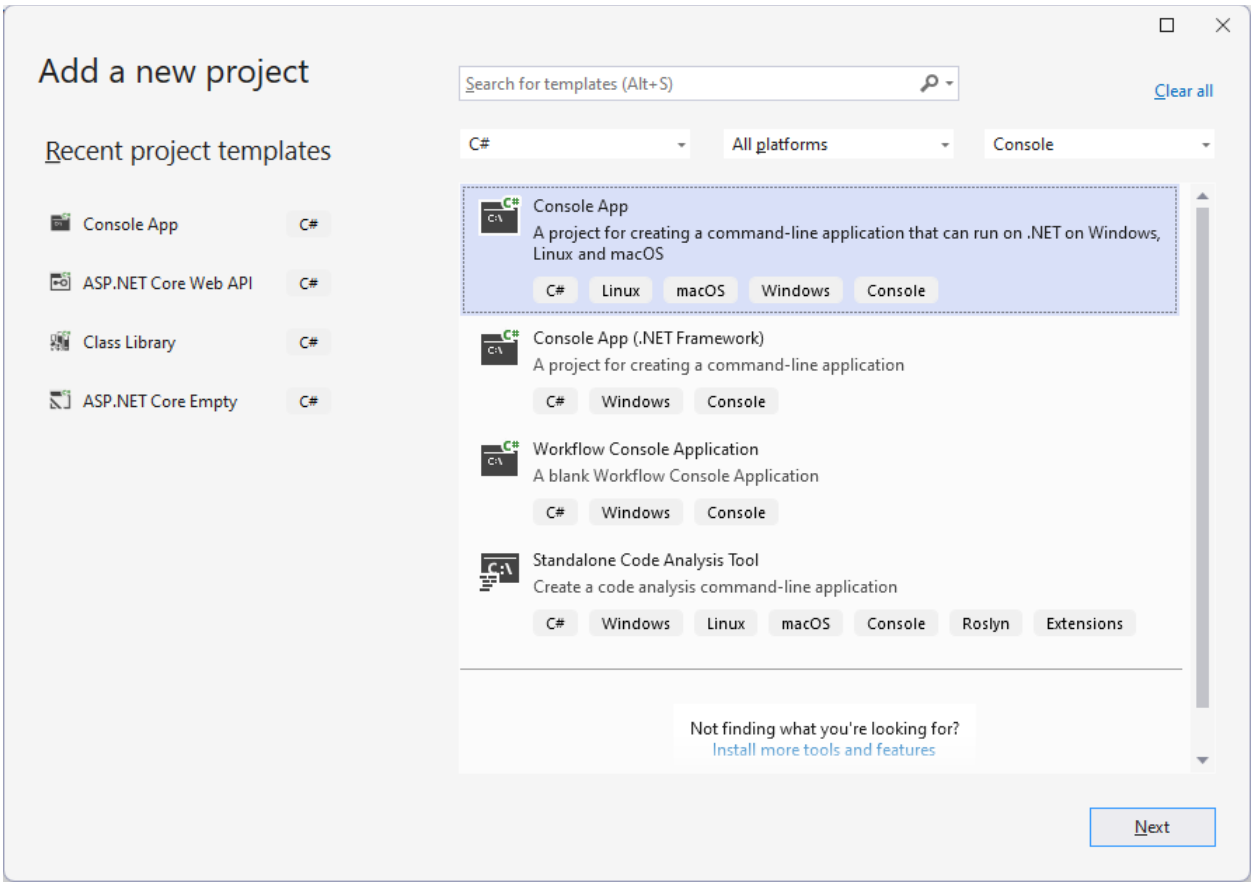


© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 20

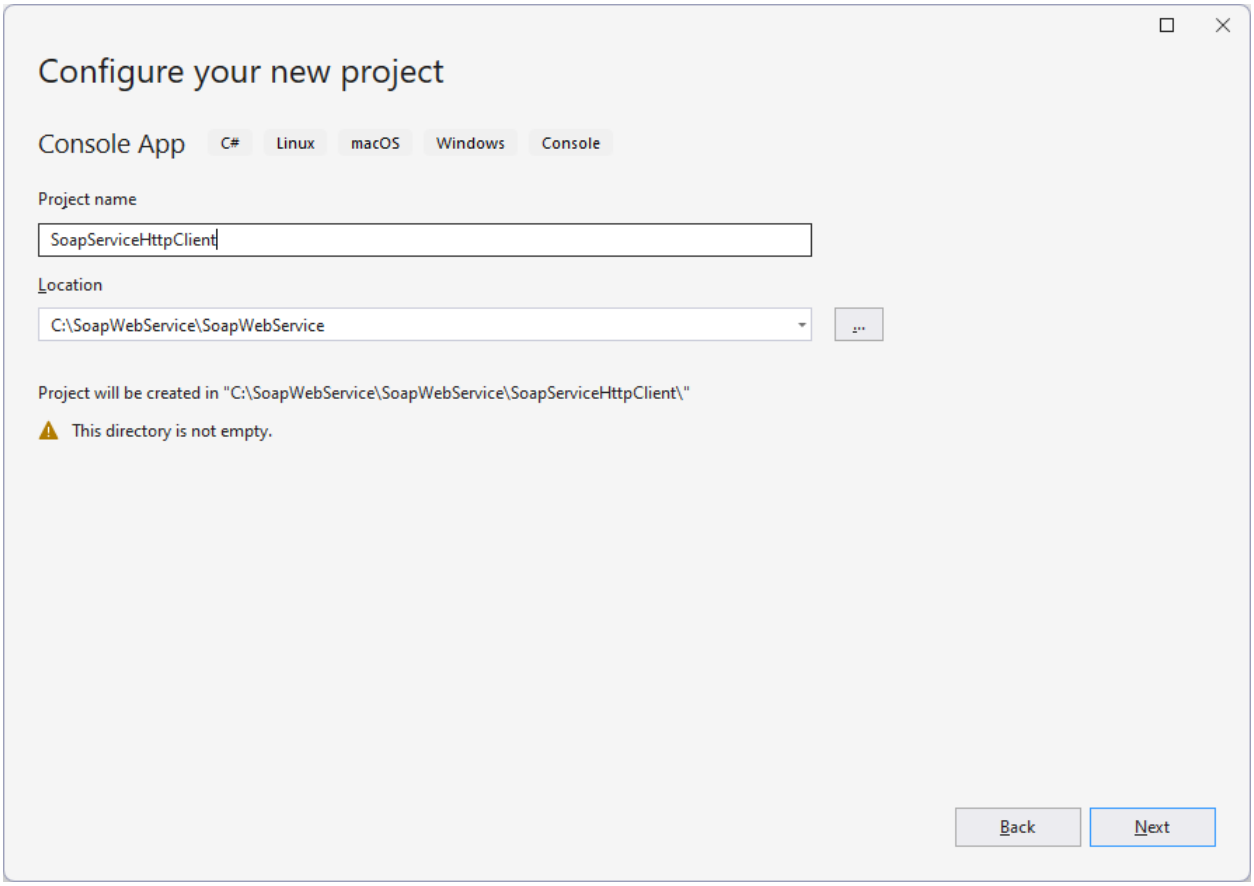
The resulting XML-format response, enclosed in a SOAP envelope in xml-format, which can be parsed by the client application as necessary:

Request Square Root using HttpClient

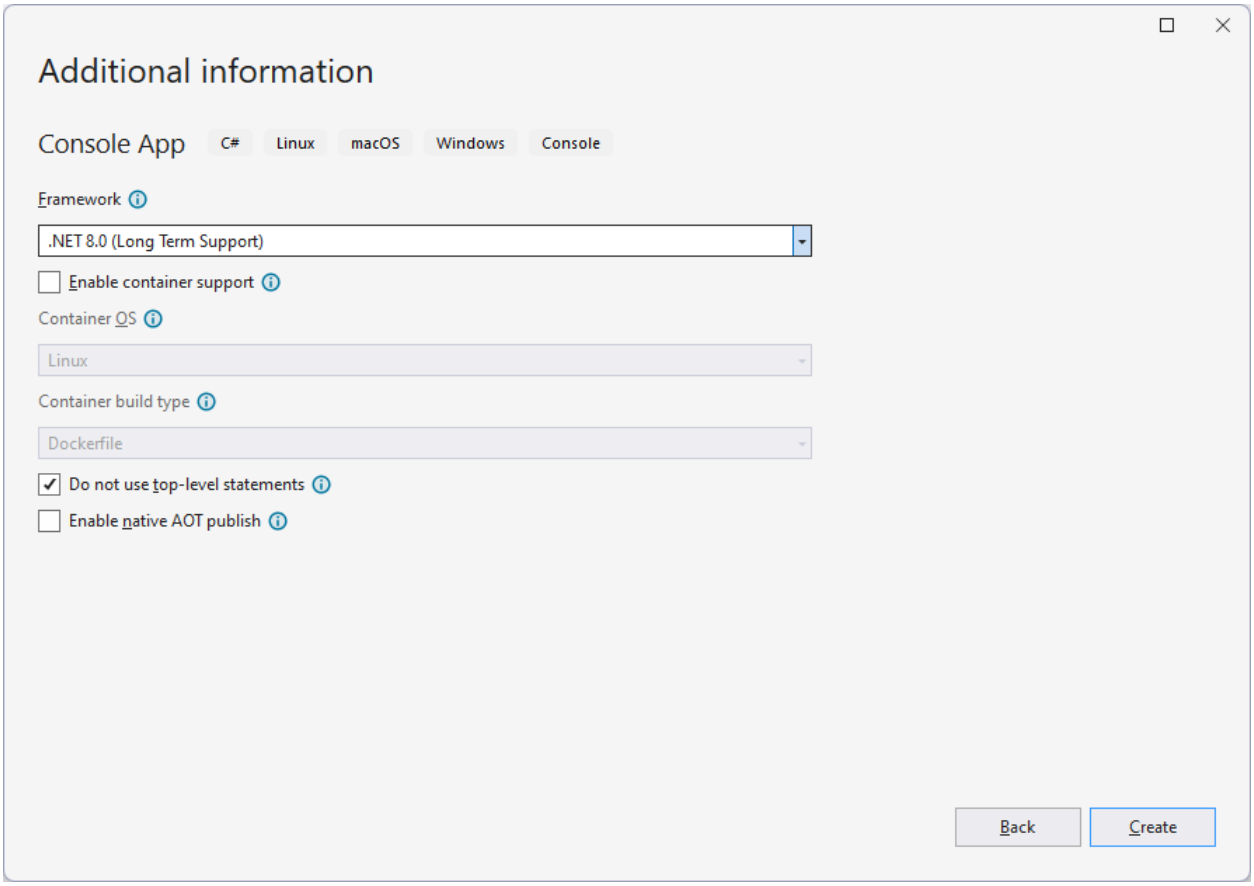
Add the new 'SoapServiceHttpClient' console project to the 'SoapWebService' solution.



© APLNext, LLC - All Rights Reserved.



© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 22



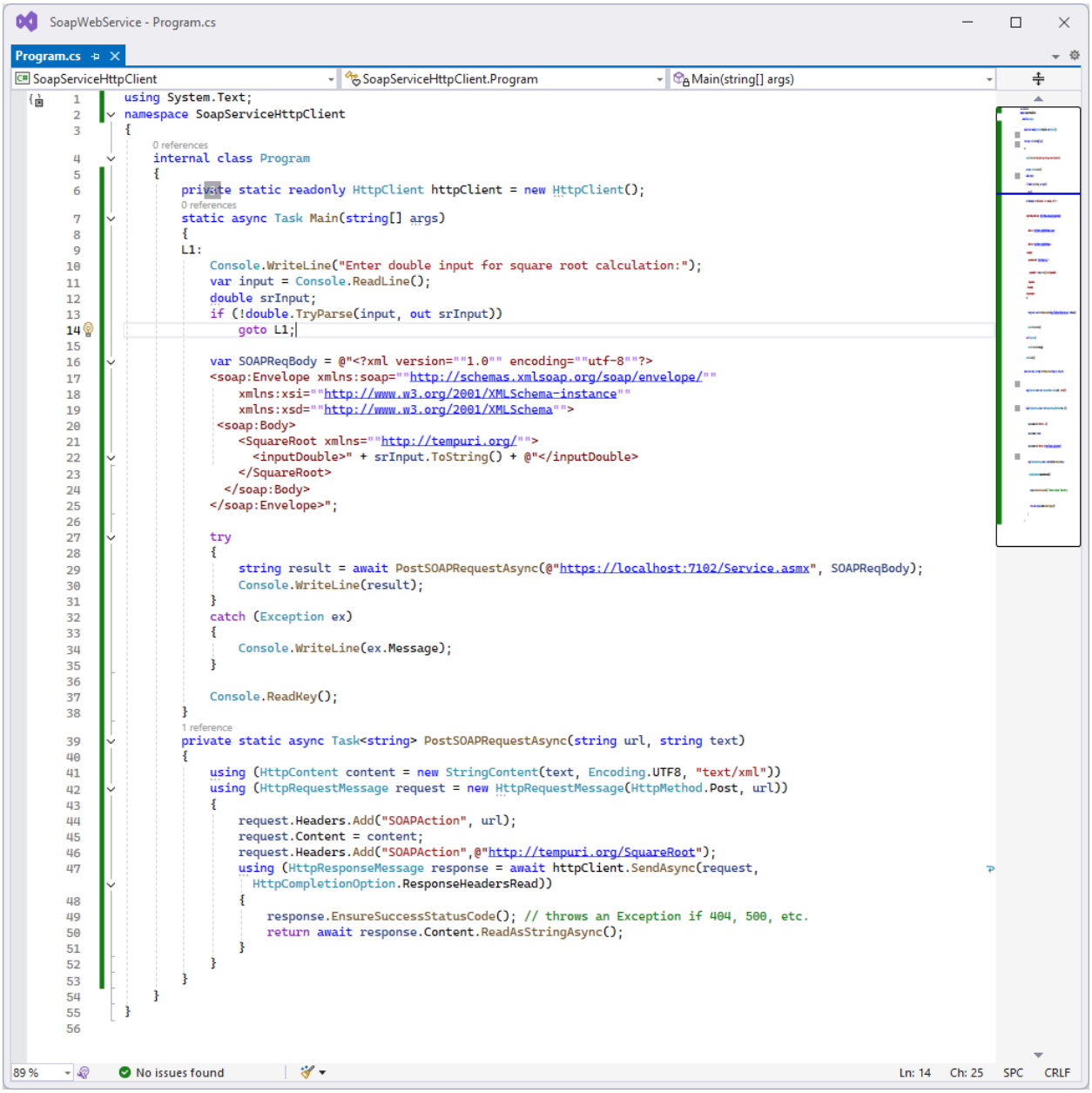
Replace the content of the Program.cs code file in the SoapServiceHttpClient project with this code:

```
using System.Text;
namespace SoapServiceHttpClient
{
   internal class Program
   {
      private static readonly HttpClient httpClient = new HttpClient();
      static async Task Main(string[] args)
      {
      L1:
         Console.WriteLine("Enter double input for square root calculation:");
         var input = Console.ReadLine();
         double srInput;
         if (!double.TryParse(input, out srInput))
            goto L1;

         var SOAPReqBody = @"<?xml version=""1.0"" encoding=""utf-8""?>
         <soap:Envelope xmlns:soap=""http://schemas.xmlsoap.org/soap/envelope/""
            xmlns:xsi=""http://www.w3.org/2001/XMLSchema-instance""
```

© APLNext, LLC - All Rights Reserved.

```
          xmlns:xsd=""http://www.w3.org/2001/XMLSchema"">
        <soap:Body>
          <SquareRoot xmlns=""http://tempuri.org/"">
           <inputDouble>" + srInput.ToString() + @"</inputDouble>
          </SquareRoot>
        </soap:Body>
       </soap:Envelope>";

       try
       {
         string result = await PostSOAPRequestAsync(@"https://localhost:7102/Service.asmx",
SOAPReqBody);
         Console.WriteLine(result);
       }
       catch (Exception ex)
       {
         Console.WriteLine(ex.Message);
       }

       Console.ReadKey();
    }
    private static async Task<string> PostSOAPRequestAsync(string url, string text)
    {
       using (HttpContent content = new StringContent(text, Encoding.UTF8, "text/xml"))
       using (HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Post, url))
       {
         request.Headers.Add("SOAPAction", url);
         request.Content = content;
         request.Headers.Add("SOAPAction",@"http://tempuri.org/SquareRoot");
         using (HttpResponseMessage response = await httpClient.SendAsync(request,
HttpCompletionOption.ResponseHeadersRead))
         {
           response.EnsureSuccessStatusCode(); // throws an Exception if 404, 500, etc.
           return await response.Content.ReadAsStringAsync();
         }
       }
    }
  }
}
```

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 24

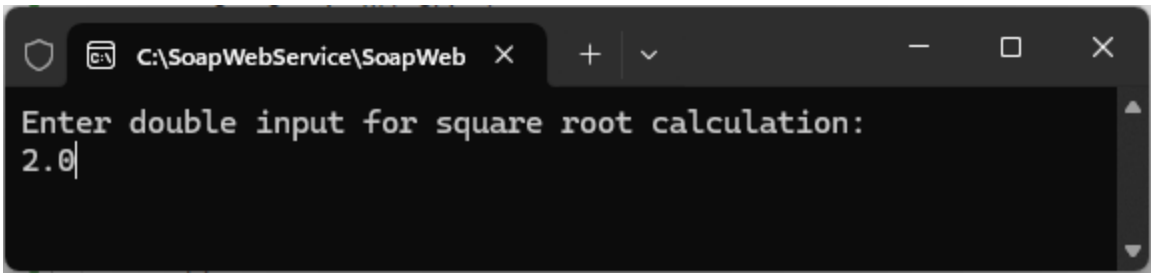


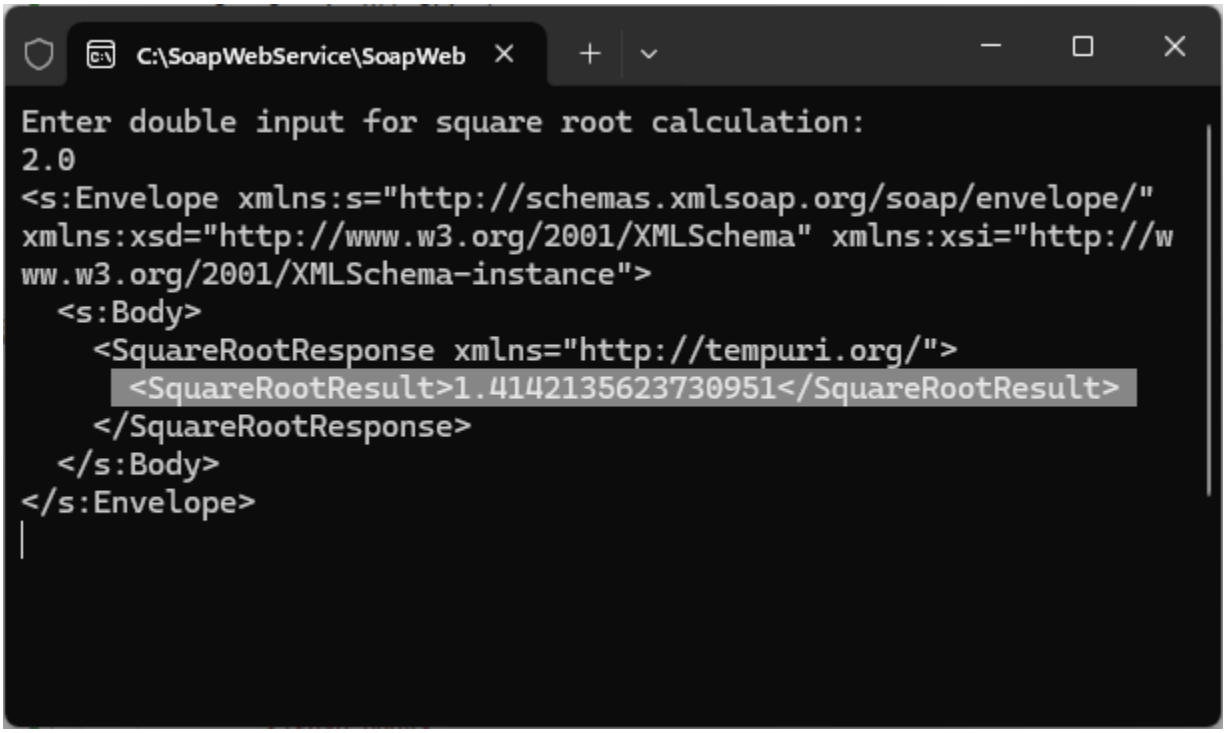
The source code in Program.cs:

- Requests user input of a double numeric value
- Creates xml-format text containing a SOAP envelope
- Adds xml-format SOAP body text with the request for a square root calculation
- Sends the SOAP envelope to the SoapWebService as a POST request
- Displays the result for the user

With the SoapWebService running, debug the SoapServiceHttpClient project, enter the desired double numeric input and click Enter/Return to see the SoapServiceHttpClient in action:

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 25





More Information

For technical information for APL64 subscribers contact support@apl2000.com

For customized consulting or licensing information contact sales@apl2000

© APLNext, LLC - All Rights Reserved. - 2026-07-21 UTC - Page: 26