

APL64: CPC in a JavaScript Server

Contents

Overview	1
Download the Project	2
Prerequisites	2
Create the APL64 CPC.....	2
The APL64 public function.....	3
Complete the CPC creation utility dialog.....	4
The JavaScript project	6
Web server launcher	7
JavaScript web server	7
Node.js JavaScript wrapper	9
.Net Bridge Process	10
Package Configuration File	14
Project File Structure	14
Use the JavaScript Project	15
Start the web server	15
Try a quick example	16
Enter user input and get results	17
Test the Exception Handling	18

Overview

An APL64 CPC is a cross-platform Nuget package which exposes one or more APL64 programmer-developed public functions as .Net methods.

This example illustrates how .Net methods can be used from html/JavaScript web pages.

This document outlines the creation of an APL64 CPC and a JavaScript web page which uses the APL64 CPC. The reader of this document is assumed to have a significant understanding of APL64 CPC creations, JavaScript, html web pages and web servers.

Download the Project

This document is not step-by-step recipe.

Download the entire project [here](#). After downloading the zip-format file, expand it to c:\CpcJs to match the references in this document. To run the project, follow the instructions in this document which require creating the APL64 CPC and referencing it in the .Net Bridge Process project.

Prerequisites

Components which must be installed on the development workstation:

- APL64 Developer version
- .NET SDK (1)
- Node.js (2)

Components which must be installed on the server workstation:

- .NET SDK (1)
- Node.js (2)

Components which must be installed on the client workstation:

- Up-to-date Internet browser

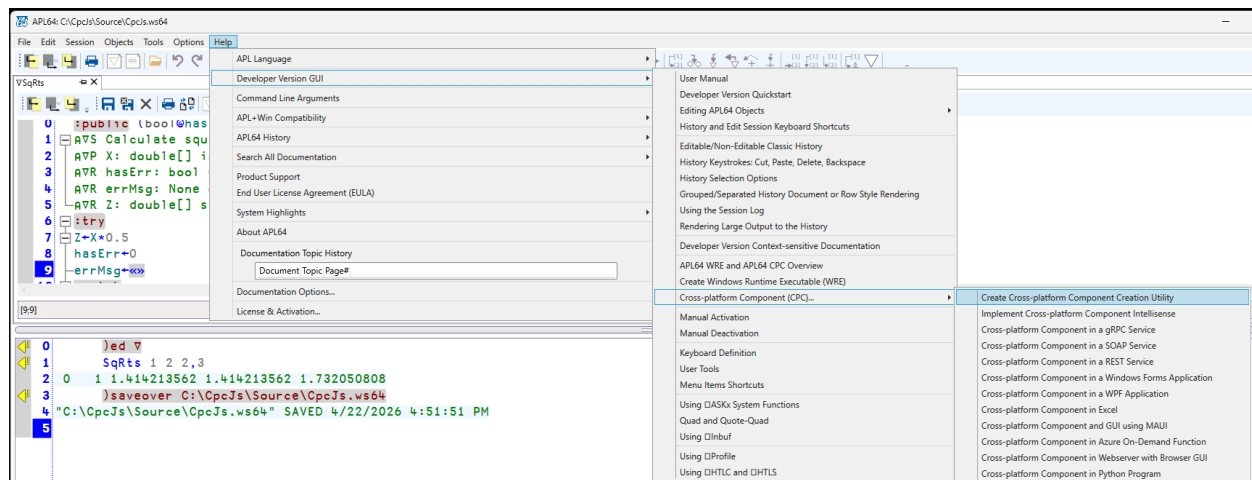
(1) .Net SDK is a no-cost component available [here](#). Use 2⇨ sysinit to determine the APL64-compatible .Net SDK version.

(2) Node.js, v24.14 or later, is a no-cost component available [here](#).

Create the APL64 CPC

For details on the APL64 CPC creation utility and workflow, see the user documentation in the APL64 developer version menu item:

Help | Developer Version GUI | Cross-platform Component (CPC)... | ...Creation Utility

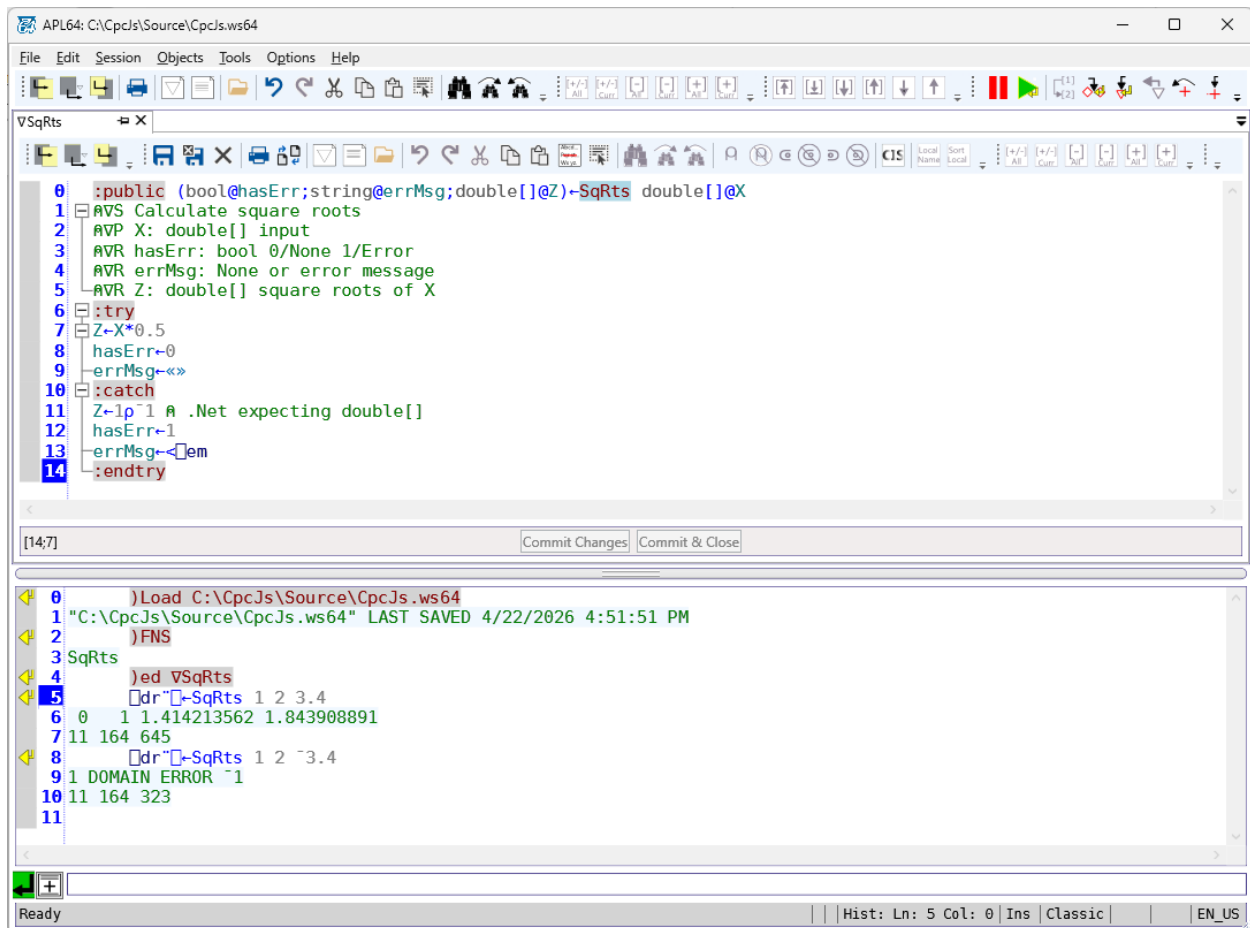


The APL64 public function

The SqRts public function is saved in the workspace: C:\CpcJs\Source\CpcJs.ws64.

The public function includes the Intellisense scaffold (public comments ⌈VS, ⌈VP, ⌈VR), so that the APL64 CPC will provide Intellisense documentation for the end user of the APL64 CPC in the .Net environment.

The public function can be tested in the APL64 developer version environment:



Complete the CPC creation utility dialog

The C:\CpcJs\Target folder will contain the APL64 CPC.

The completed CPC creation dialog:

APL64: Create Cross Platform Component

CPC Workspace Path *: C:\CpcJs\Source\CpcJs.ws64

CROSS PLATFORM COMPONENT CONTENT

Cross Platform Component Target Folder *: C:\CpcJs\Target Browse Open Folder in File Explorer

Cross Platform Component Name *: CpcJs ☒ Same as CPC Workspace Name

Cross Platform Component Class Name *: CpcJsClass

APL64 xml-format Configuration File: Browse ☐ Include APL64 xml-format Configuration File

Additional Files Required for the Application

Source Path	Base Target Path	Target Path Suffix	Overwrite
Add One Required File Add Multiple Required Files Add Folder of Required Files Remove All Required Files			

ASSEMBLY META-DATA

Properties.Details: File Description: Calculate Square Roots of double[]

Properties.Details: Product Name *: CpcJs

Properties.Details: Copyright *: ©APLNext LLC

Properties.Details: File Version *: 0.0.0.1

Properties.Details: Version: 0.0.0.1 ☒ Same as File Version

Assembly.Info: Company Name *: APLNextLLC

Assembly.Info: Application Description: Calculate Square Roots of double[] ☐ Same as File Description

Assembly.Info: Version: 0.0.0.1 ☒ Same as File Version

Assembly.Info: Neutral Language: EN-US

Assembly.Info: Description: Calculate Square Roots of double[] ☒ Same as File Description

Application Icon File: Browse

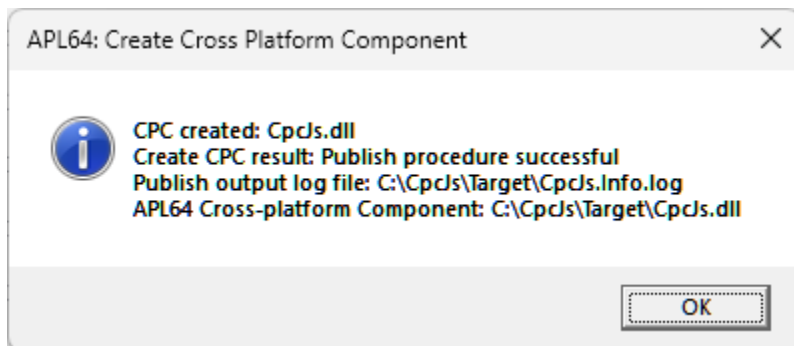
Nuget Package Guid *: 44d19de4-5fe6-4267-899b-c14aa7875dde Create

Nuget Package Id*: APLNextLLC.CpcJs ☒ Use CompanyName.ComponentName

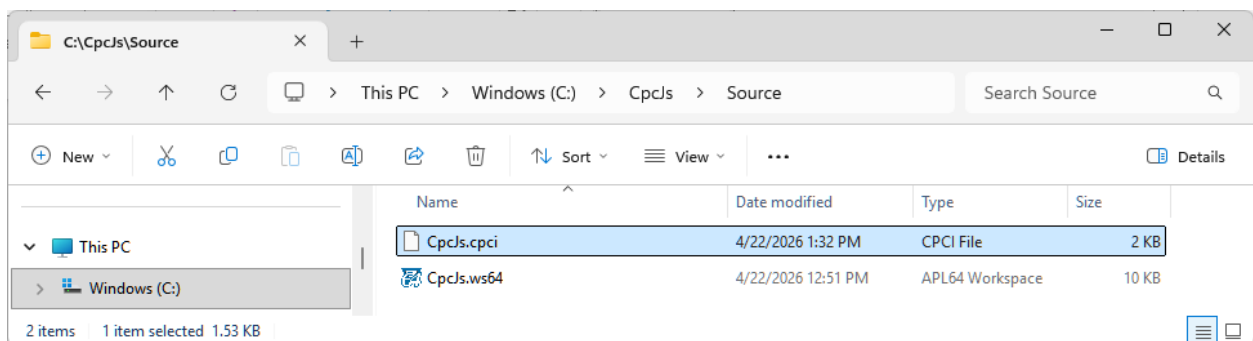
Nuget Package Files

Create Exit New CPC Info Load CPC Info Save CPC Info * Required Entries

Clicking the 'Create' button will create the APL64 CPC:

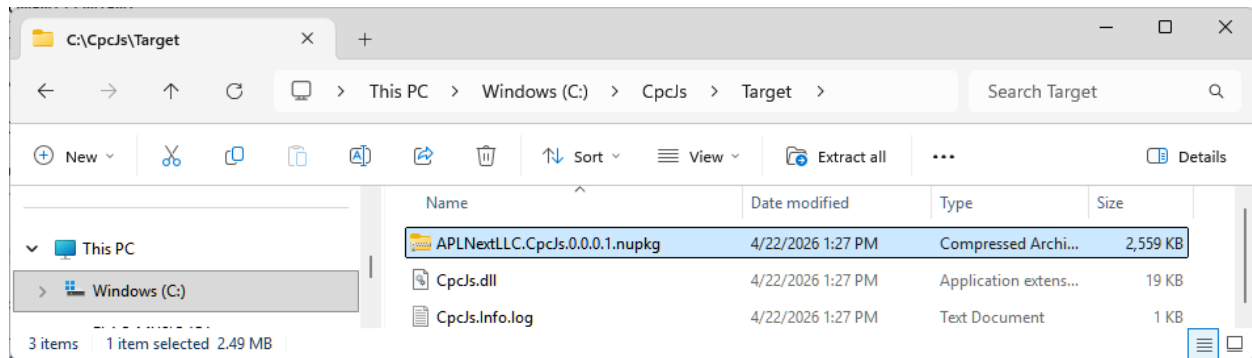


The information in the CPC creation dialog, CpcJs.cpci, is saved for future use in the C:\CpcJs\Source folder

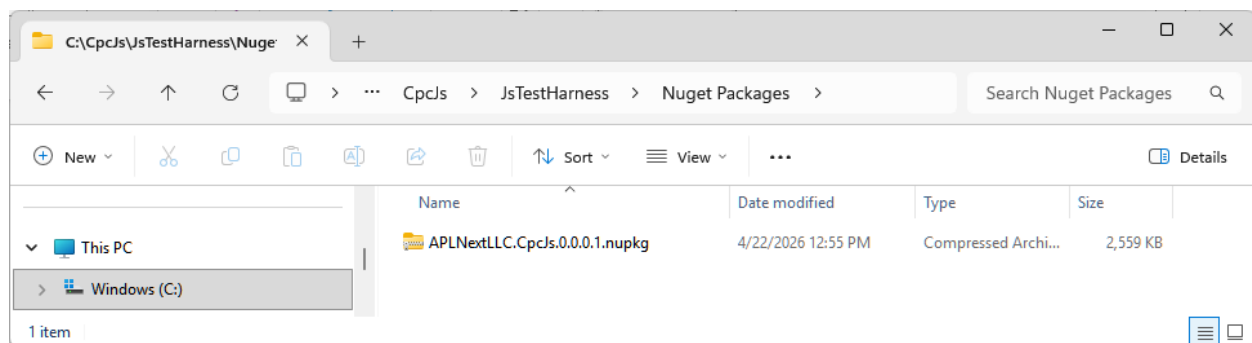


The C:\CpcJs\JsTestHarness\Nuget Packages folder contains the published APL64 CPC. In a production environment the CPC can be delivered to the end user or published on a Nuget package publish website.

To avoid including extraneous files in the APL64 CPC, the CPC in the target folder is copied from the target folder:



to the c:\CpcJs\JsTestHarness\Nuget Packages folder.



The JavaScript project

This document does not provide a step-by-step recipe to create the JavaScript application. Download the full project to see the project files referenced in this document.

Overall process:

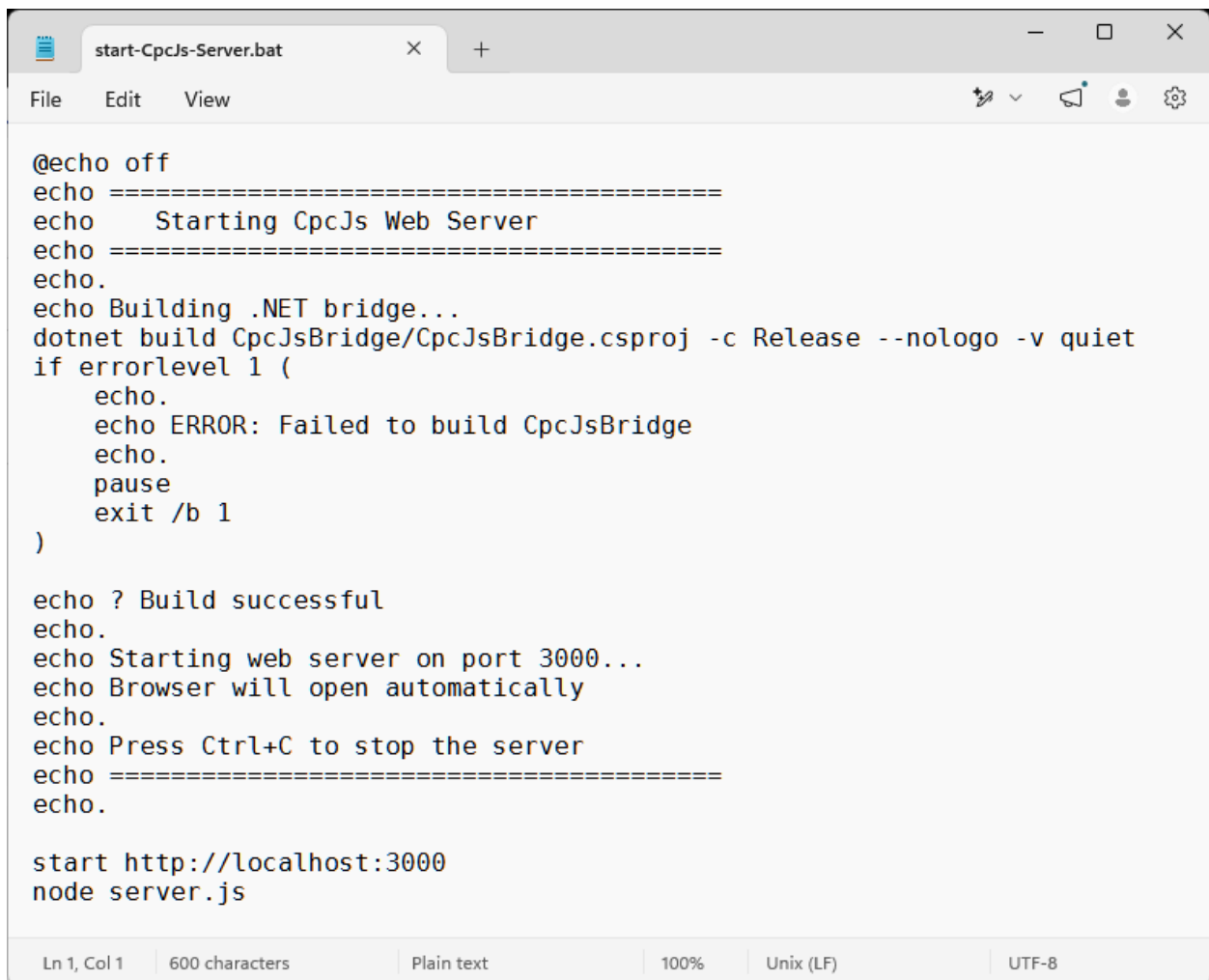
- The Start-CpcJs-Server.bat file is run to start the JavaScript server and open the default browser on the client workstation
- When the server.js starts, the index.html page is available to a client to access the server
- When the client inputs a double[] vector and requests server processing, the Node.js wrapper receives the input, asynchronously calls the .Net CpcJsBridge project, and waits for the result

- The CpcJsBridge project uses the APL64 CPC Nuget package SqRts() public function to calculate the square roots based on the client input.
- The results are passed to the client reversing the above flow of information.

Web server launcher

The C:\CpcJs\JsTestHarness\start-CpcJs-Server.bat file is run to start the server

- Builds the CpcJsBridge .Net project
- Starts the Node.js web server
- Opens the default browser to <http://localhost:3000>.



```
@echo off
echo =====
echo   Starting CpcJs Web Server
echo =====
echo.
echo Building .NET bridge...
dotnet build CpcJsBridge/CpcJsBridge.csproj -c Release --nologo -v quiet
if errorlevel 1 (
    echo.
    echo ERROR: Failed to build CpcJsBridge
    echo.
    pause
    exit /b 1
)

echo ? Build successful
echo.
echo Starting web server on port 3000...
echo Browser will open automatically
echo.
echo Press Ctrl+C to stop the server
echo =====
echo.

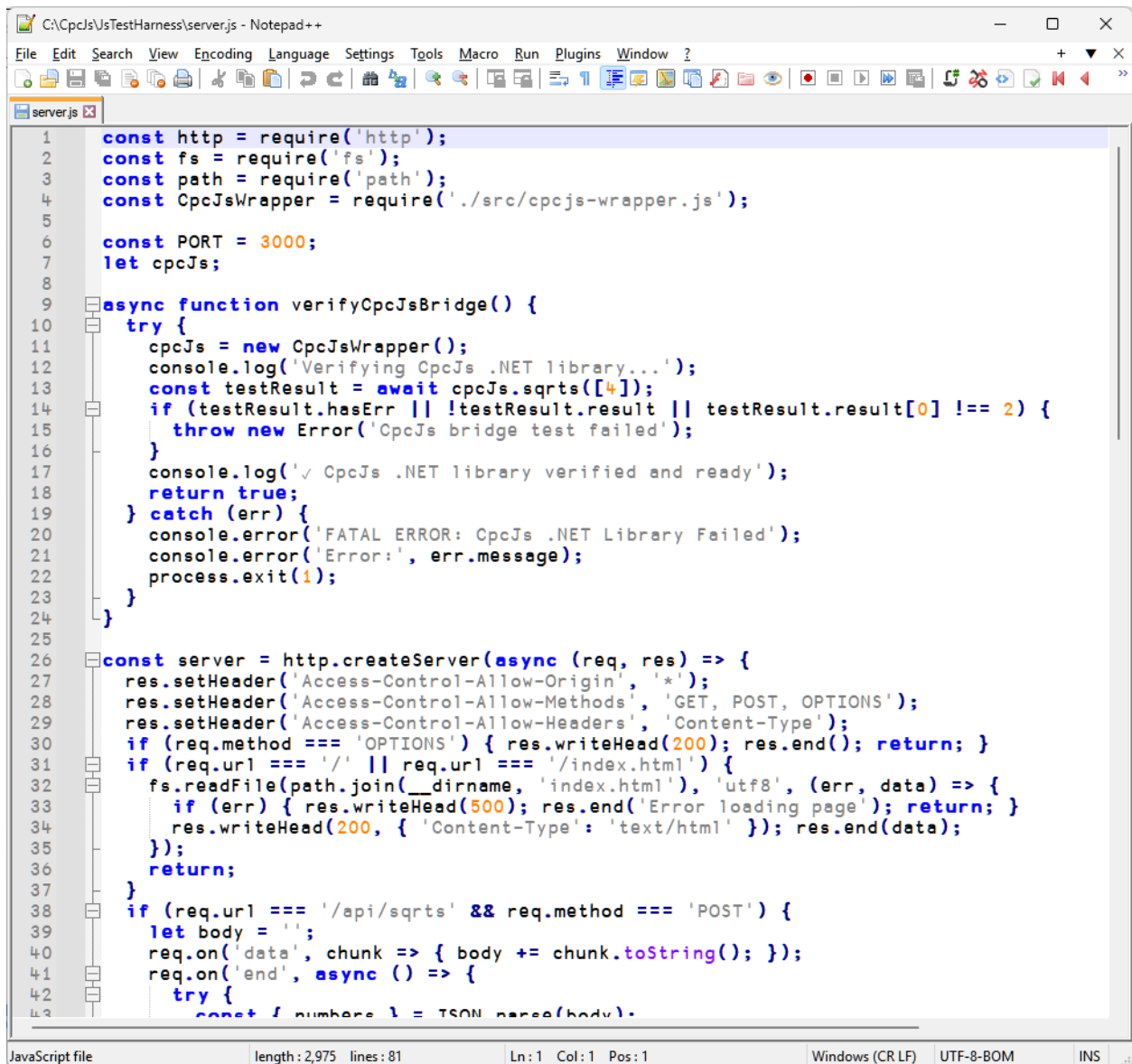
start http://localhost:3000
node server.js
```

JavaScript web server

The JavaScript web page runs in a web server. For testing purposes, a JavaScript web server is created on the workstation using port 3000. In a production environment the application would be supported in a production web server.

The C:\CpcJs\JsTestHarness\server.js file:

- Starts an HTTP (Node.js) server on port **3000**
- Verifies the CpcJsLibrary on server start up
- Serves the index.html web page as the default (home) page
- Exposes the POST endpoint /api/sqrts for requesting square roots of user input in the form {numbers: [1.23, 2, ...]}
- Calls **cpcJs.sqrts(numbers)** and returns the result JSON
- Handles Cross-origin Resource Sharing (CORS) between the JavaScript domain and the .Net domainHtml JavaScript web page

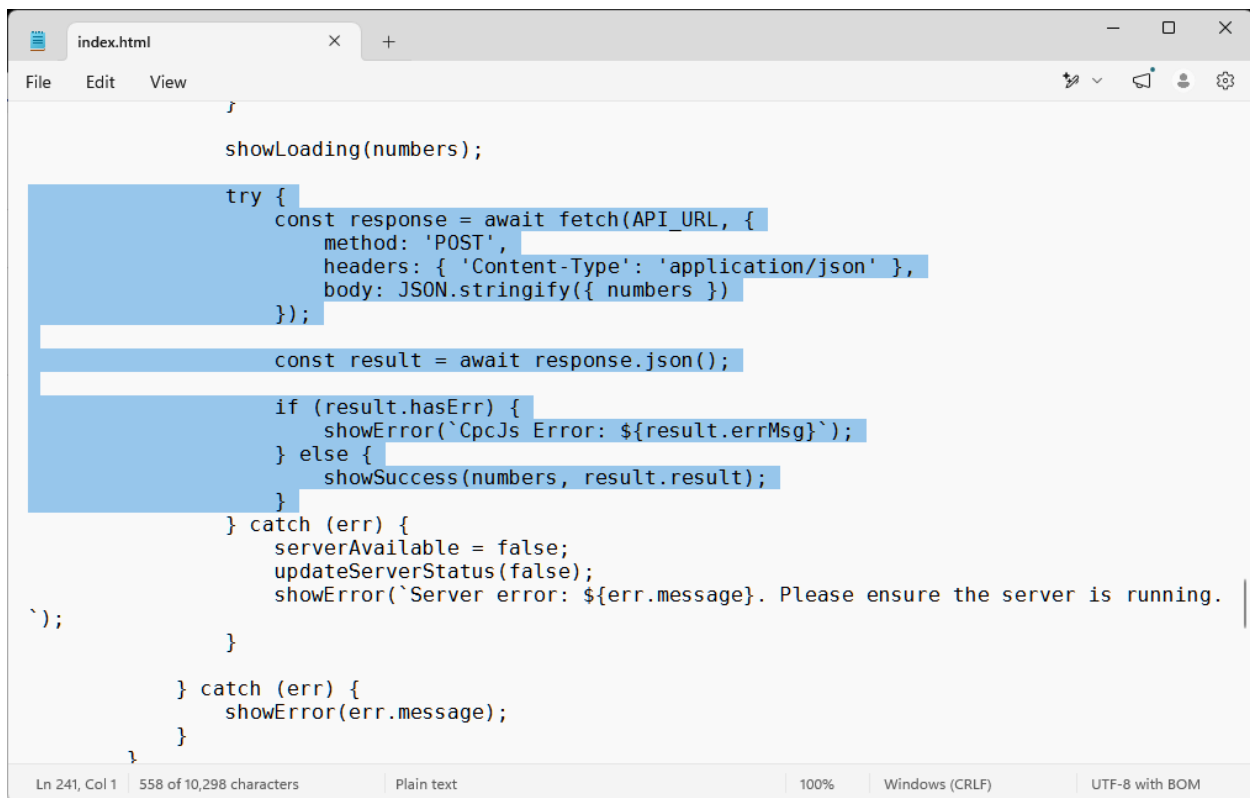


```
1 const http = require('http');
2 const fs = require('fs');
3 const path = require('path');
4 const CpcJsWrapper = require('./src/cpcjs-wrapper.js');
5
6 const PORT = 3000;
7 let cpcJs;
8
9 async function verifyCpcJsBridge() {
10   try {
11     cpcJs = new CpcJsWrapper();
12     console.log('Verifying CpcJs .NET library...');
13     const testResult = await cpcJs.sqrts([4]);
14     if (testResult.hasErr || !testResult.result || testResult.result[0] !== 2) {
15       throw new Error('CpcJs bridge test failed');
16     }
17     console.log('✓ CpcJs .NET library verified and ready');
18     return true;
19   } catch (err) {
20     console.error('FATAL ERROR: CpcJs .NET Library Failed!');
21     console.error('Error:', err.message);
22     process.exit(1);
23   }
24 }
25
26 const server = http.createServer(async (req, res) => {
27   res.setHeader('Access-Control-Allow-Origin', '*');
28   res.setHeader('Access-Control-Allow-Methods', 'GET, POST, OPTIONS');
29   res.setHeader('Access-Control-Allow-Headers', 'Content-Type');
30   if (req.method === 'OPTIONS') { res.writeHead(200); res.end(); return; }
31   if (req.url === '/' || req.url === '/index.html') {
32     fs.readFile(path.join(__dirname, 'index.html'), 'utf8', (err, data) => {
33       if (err) { res.writeHead(500); res.end('Error loading page'); return; }
34       res.writeHead(200, { 'Content-Type': 'text/html' }); res.end(data);
35     });
36     return;
37   }
38   if (req.url === '/api/sqrts' && req.method === 'POST') {
39     let body = '';
40     req.on('data', chunk => { body += chunk.toString(); });
41     req.on('end', async () => {
42       try {
43         const { numbers } = JSON.parse(body);
```

The C:\CpcJs\JsTestHarness\index.html file:

- Presents the main page of the application to a client browser
- Provides for user input of comma-separated, double[] vector
- Shows real-time server status
- The asynchronous function calculateSqRts() in the web page submits the user input to the Node.js wrapper in the web server
- Waits for the web server to respond with the result
- When results from the web server, the html page displays them for the client

This section of the Html web page accesses the JavaScript server when the user clicks the ‘Calculate Square Roots’ button on the page:



```

showLoading(numbers);

try {
    const response = await fetch(API_URL, {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ numbers })
    });

    const result = await response.json();

    if (result.hasErr) {
        showError(`CpcJs Error: ${result.errMsg}`);
    } else {
        showSuccess(numbers, result.result);
    }
} catch (err) {
    serverAvailable = false;
    updateServerStatus(false);
    showError(`Server error: ${err.message}. Please ensure the server is running.`);
}

} catch (err) {
    showError(err.message);
}

```

Node.js JavaScript wrapper

The JavaScript wrapper: C:\CpcJs\JsTestHarness\src\cpcjs-wrapper.js file:

- Spawns the .Net ‘bridge’ project as a child process
- Handles json-serialization and deserialization of input and output data
- Passes the prepared input data to the .Net ‘bridge’ process asynchronously
- Waits for the result .Net ‘bridge’ process results
- Returns the result to the JavaScript web page

```
1  const { spawn } = require('child_process');
2  const path = require('path');
3
4  /**
5   * CpcJs Wrapper - JavaScript interface to CpcJs .NET library
6   */
7  class CpcJsWrapper {
8      constructor() {
9          this.bridgePath = path.join(__dirname, '..', 'CpcJsBridge', 'bin', 'Release', 'net10.0', 'CpcJsBridge.exe');
10     }
11
12     /**
13      * Calculate square roots of an array of numbers
14      * @param {number[]} numbers - Array of numbers to calculate square roots for
15      * @returns {Promise<{hasErr: boolean, errMsg: string, result: number[]}>}
16      */
17     async sqrts(numbers) {
18         return new Promise((resolve, reject) => {
19             // Validate input
20             if (!Array.isArray(numbers)) {
21                 reject(new Error('Input must be an array of numbers'));
22                 return;
23             }
24
25             if (!numbers.every(n => typeof n === 'number')) {
26                 reject(new Error('All elements must be numbers'));
27                 return;
28             }
29
30             // Convert input to JSON
31             const inputJson = JSON.stringify(numbers);
32
33             // Spawn the .NET bridge process
34             const process = spawn(this.bridgePath, [inputJson]);
35
36             let stdout = '';
37             let stderr = '';
38
39             process.stdout.on('data', (data) => {
40                 stdout += data.toString();
41             });
42         });
43     }
44 }
```

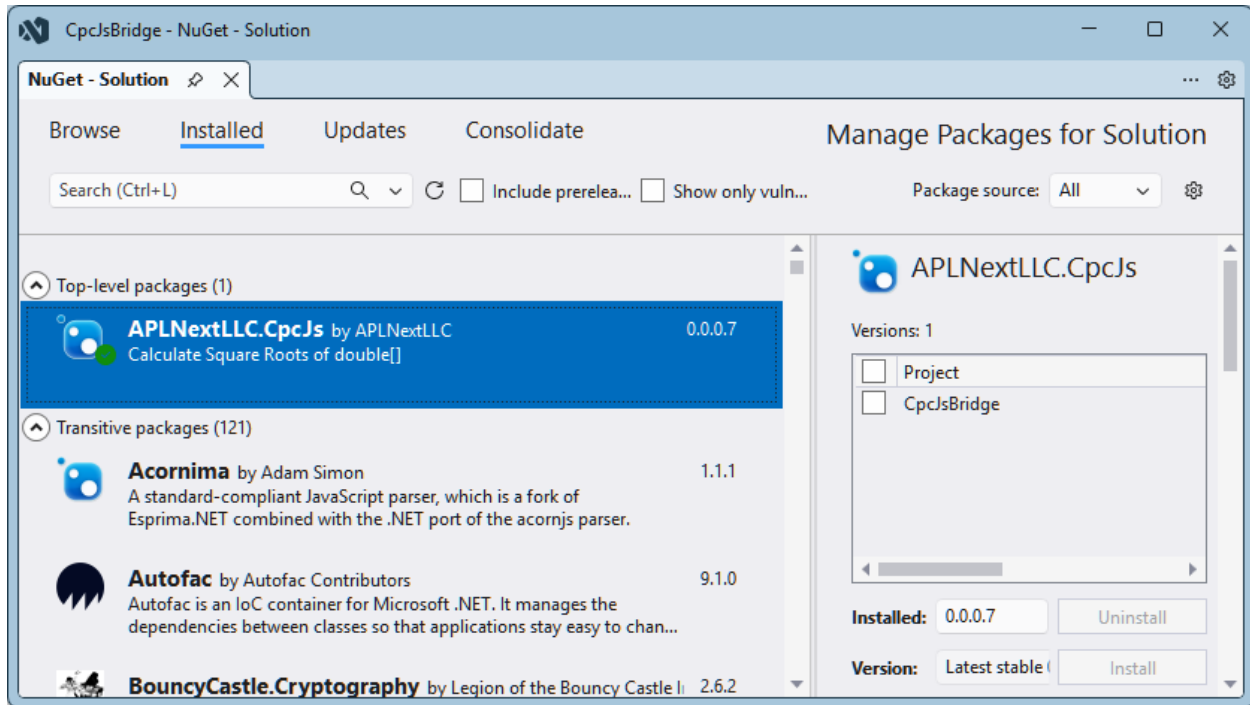
.Net Bridge Process

The .Net bridge process

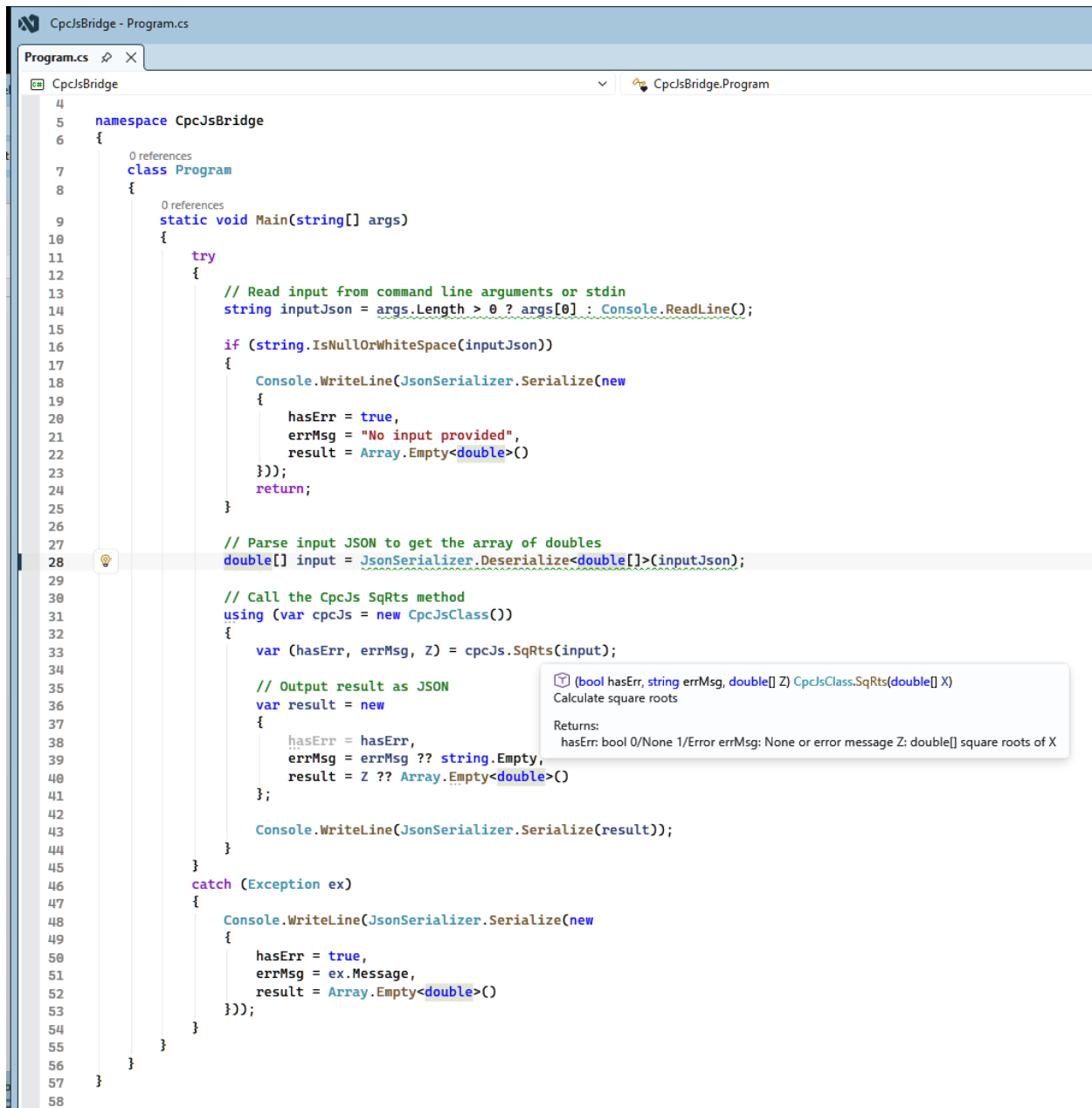
The .Net bridge process: C:\CpcJs\JsTestHarness\CpcJsBridge\Program.cs:

- Is a .Net console application
- Has a reference to the APL64 CPC Nuget package
- Accepts json-format client input via a command line argument (stdin)
- Converts the json-format input to .Net double[] data
- Calls the APL64 public CpcJsClass.SqRts() function to calculate the square roots
- Returns the result to the JavaScript wrapper

The .Net Bridge process references the APL64 Nuget package:



The Intellisense scaffold in the SqRts public function in the APL64 CPC provides Intellisense documentation for the end user of the APL64 CPC used in a .Net application like the .Net Bridge process:

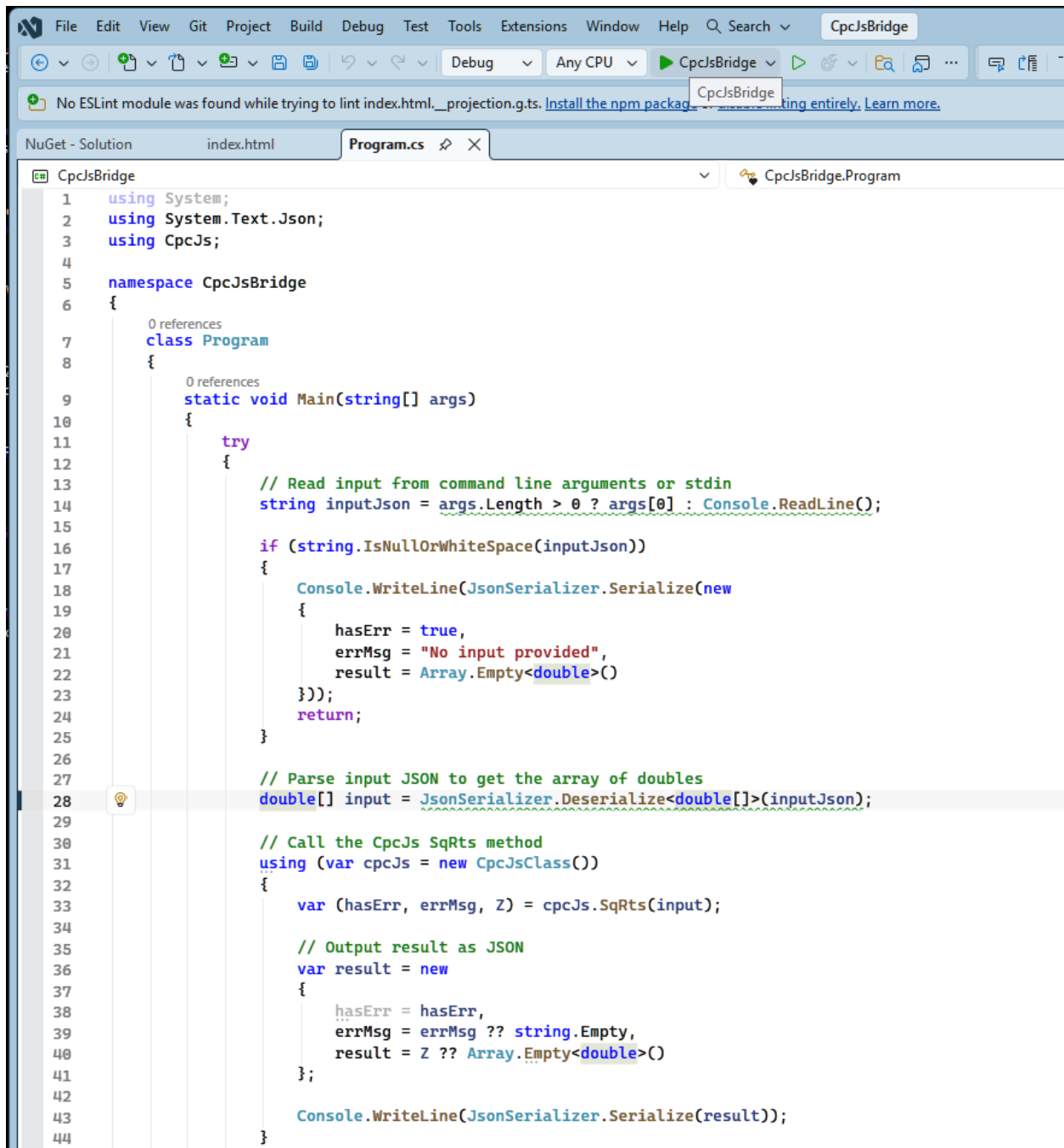


```
4
5 namespace CpcJsBridge
6 {
7     0 references
8     class Program
9     {
10         0 references
11         static void Main(string[] args)
12         {
13             try
14             {
15                 // Read input from command line arguments or stdin
16                 string inputJson = args.Length > 0 ? args[0] : Console.ReadLine();
17
18                 if (string.IsNullOrEmpty(inputJson))
19                 {
20                     Console.WriteLine(JsonSerializer.Serialize(new
21                     {
22                         hasErr = true,
23                         errMsg = "No input provided",
24                         result = Array.Empty<double>()
25                     }));
26                     return;
27                 }
28
29                 // Parse input JSON to get the array of doubles
30                 double[] input = JsonSerializer.Deserialize<double[]>(inputJson);
31
32                 // Call the CpcJs Sqrt method
33                 using (var cpcJs = new CpcJsClass())
34                 {
35                     var (hasErr, errMsg, Z) = cpcJs.Sqrt(input);
36
37                     // Output result as JSON
38                     var result = new
39                     {
40                         hasErr = hasErr,
41                         errMsg = errMsg ?? string.Empty,
42                         result = Z ?? Array.Empty<double>()
43                     };
44
45                     Console.WriteLine(JsonSerializer.Serialize(result));
46                 }
47             }
48             catch (Exception ex)
49             {
50                 Console.WriteLine(JsonSerializer.Serialize(new
51                 {
52                     hasErr = true,
53                     errMsg = ex.Message,
54                     result = Array.Empty<double>()
55                 }));
56             }
57         }
58     }
59 }
```

(bool hasErr, string errMsg, double[] Z) CpcJsClass.Sqrt(double[] X)
Calculate square roots
Returns:
hasErr: bool 0/None 1/Error errMsg: None or error message Z: double[] square roots of X

The .Net Bridge Process can be tested independently of the JavaScript wrapper.

- Debug the CpcJsBridge console project to present the Console
- Enter valid json text into the Console and click Enter/Return
- The three-element result (bool hasErr, string errMsg, double[] result) will be displayed in the Console



```
1 using System;
2 using System.Text.Json;
3 using CpcJs;
4
5 namespace CpcJsBridge
6 {
7     0 references
8     class Program
9     {
10         0 references
11         static void Main(string[] args)
12         {
13             try
14             {
15                 // Read input from command line arguments or stdin
16                 string inputJson = args.Length > 0 ? args[0] : Console.ReadLine();
17
18                 if (string.IsNullOrEmpty(inputJson))
19                 {
20                     Console.WriteLine(JsonSerializer.Serialize(new
21                     {
22                         hasErr = true,
23                         errMsg = "No input provided",
24                         result = Array.Empty<double>()
25                     }));
26                     return;
27                 }
28
29                 // Parse input JSON to get the array of doubles
30                 double[] input = JsonSerializer.Deserialize<double[]>(inputJson);
31
32                 // Call the CpcJs.Sqrt method
33                 using (var cpcJs = new CpcJsClass())
34                 {
35                     var (hasErr, errMsg, Z) = cpcJs.Sqrt(input);
36
37                     // Output result as JSON
38                     var result = new
39                     {
40                         hasErr = hasErr,
41                         errMsg = errMsg ?? string.Empty,
42                         result = Z ?? Array.Empty<double>()
43                     };
44
45                     Console.WriteLine(JsonSerializer.Serialize(result));
46                 }
47             }
48             catch { }
49         }
50     }
51 }
```

Enter [4] to get the square root of 4:

```
Microsoft Visual Studio Debug Console
[4]
{"hasErr":false,"errMsg":"","result":[2]}

C:\CpcJs\JsTestHarness\CpcJsBridge\bin\Debug\net10.0\CpcJsBridge.exe (process 20012) exited with code 0 (0x0)
.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Enter [-1] to test the exception handling:

```
Microsoft Visual Studio Debug Console
[-1]
{"hasErr":true,"errMsg":"DOMAIN ERROR","result":[-1]}

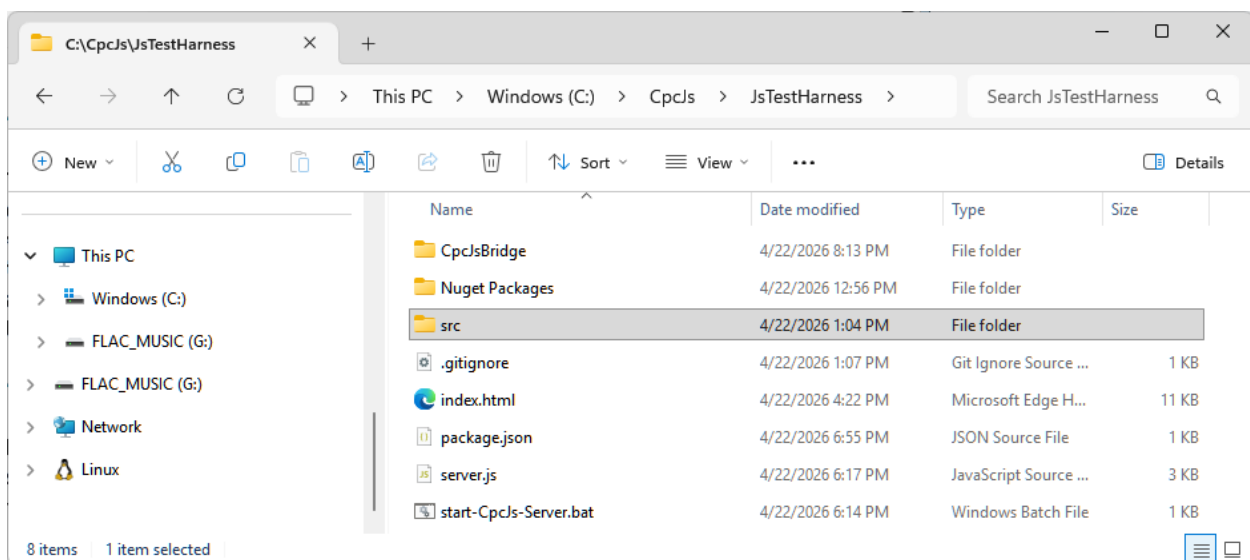
C:\CpcJs\JsTestHarness\CpcJsBridge\bin\Debug\net10.0\CpcJsBridge.exe (process 44784) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Package Configuration File

The C:\CpcJs\JsTestHarness\package.json file:

- Is the configuration file for the JavaScript project
- Defines the 'start: node server.js' script
- Includes the project metadata: name, version, description

Project File Structure



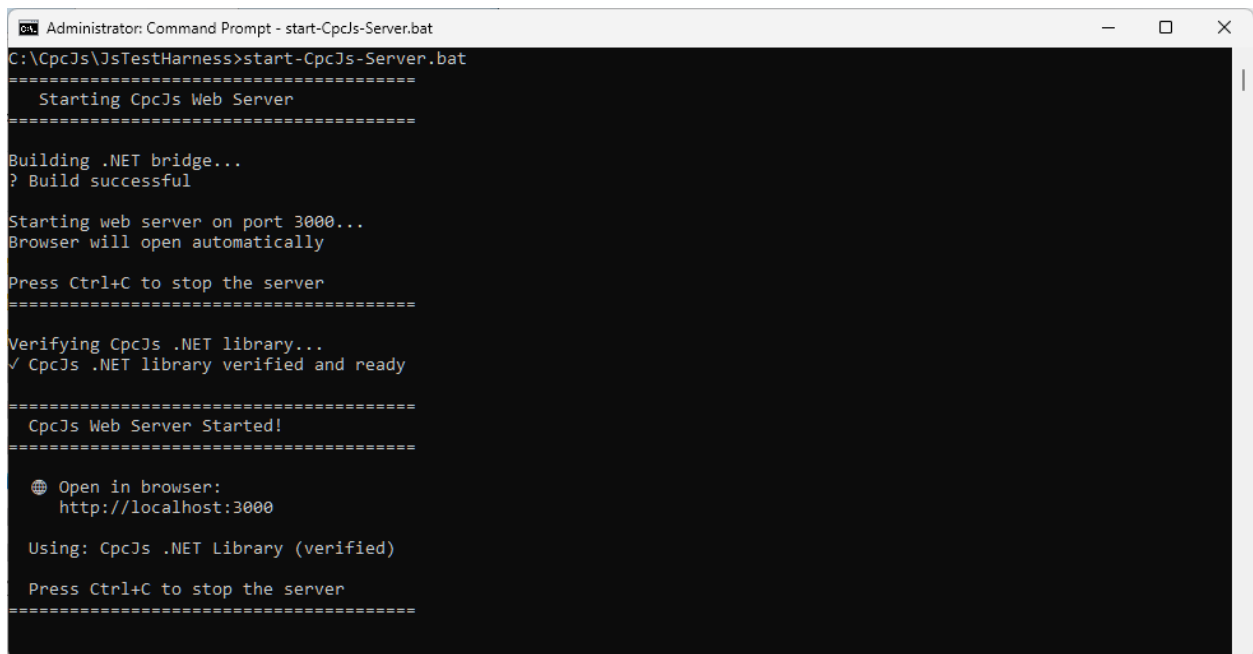
Use the JavaScript Project

Start the web server

Using a Command Prompt as administrator, run these commands:

- `cd C:\CpcJs\JsTestHarness`
- `start-CpcJs-Server.bat`

The web server will start, and the home page will be presented in the client's default browser at the localhost:3000 ([CpcJs Test Harness](http://localhost:3000)) url:



```
Administrator: Command Prompt - start-CpcJs-Server.bat
C:\CpcJs\JsTestHarness>start-CpcJs-Server.bat
=====
Starting CpcJs Web Server
=====

Building .NET bridge...
? Build successful

Starting web server on port 3000...
Browser will open automatically

Press Ctrl+C to stop the server
=====

Verifying CpcJs .NET library...
✓ CpcJs .NET library verified and ready

=====
CpcJs Web Server Started!
=====

🌐 Open in browser:
http://localhost:3000

Using: CpcJs .NET Library (verified)

Press Ctrl+C to stop the server
=====
```



CpcJs Test Harness - SqRts Method

✓ **Connected:** Using actual CpcJs .NET library.

Quick Examples

Click to load sample data:

Perfect Squares

Decimals

Large Array

Zero

Single Value

Enter Numbers (comma-separated):

e.g., 4, 9, 16, 25

Calculate Square Roots

Clear

Try a quick example

- Click the 'Decimals' button
- Click the 'Calculate Square Roots' button



CpcJs Test Harness - SqRts Method

✓ **Connected:** Using actual CpcJs .NET library.

Quick Examples

Click to load sample data:

Perfect Squares

Decimals

Large Array

Zero

Single Value

Enter Numbers (comma-separated):

2.25, 6.25, 12.25

Calculate Square Roots

Clear

✓ Results

Input	Square Root
2.25	1.500000
6.25	2.500000
12.25	3.500000

✓ **Source:** CpcJs.SqRts() .NET library

Enter user input and get results

- Enter some comma-separated numbers
- Click the 'Calculate Square Roots' button



CpcJs Test Harness - SqRts Method

✓ **Connected:** Using actual CpcJs .NET library.

Quick Examples

Click to load sample data:

Perfect Squares

Decimals

Large Array

Zero

Single Value

Enter Numbers (comma-separated):

1,2,2.5,9,11.11,144

Calculate Square Roots

Clear

✓ Results

Input	Square Root
1	1.000000
2	1.414214
2.5	1.581139
9	3.000000
11.11	3.333167
144	12.000000

✓ **Source:** CpcJs.SqRts() .NET library

Test the Exception Handling

The exception message is provided by the public function in the APL64 CPC.

Enter -2 and click Calculate Square Roots:



CpcJs Test Harness - SqRts Method

✓ **Connected:** Using actual CpcJs .NET library.

Quick Examples

Click to load sample data:

Perfect Squares

Decimals

Large Array

Zero

Single Value

Enter Numbers (comma-separated):

-2

Calculate Square Roots

Clear

✗ **Error**

CpcJs Error: DOMAIN ERROR