

FORTRA

beSOURCE
5.2

**Enterprise Quick Start
Guide**

Copyright Terms and Conditions

Copyright © Fortra, LLC and its group of companies. All trademarks and registered trademarks are the property of their respective owners.

The content in this document is protected by the Copyright Laws of the United States of America and other countries worldwide. The unauthorized use and/or duplication of this material without express and written permission from Fortra is strictly prohibited. Excerpts and links may be used, provided that full and clear credit is given to Fortra with appropriate and specific direction to the original content.

202307260331

Table of Contents

Preparations	1
Running servers in Windows	1
Running servers in Linux	2
Running the Admin Console	4
Running the beSOURCE Client	6
Adding users	6
Analyzing Java Source Files	11
Overview	11
Analyzing Java sample source files	11
Viewing Analysis Results	17
Analyzing C Source Files	28
Analyzing C sample source files	28
Viewing analysis results	32
Analyzing C++ Source Files	35
Analyzing C++ sample source files	35
Viewing analysis results	39
Generating Reports	42
Setting up the Report Authority Setting	42
Generating a report	43
Analyzing Source Files in beSOURCE Developer	45
Running beSOURCE Developer	45

Local mode analysis in beSOURCE Developer 46

Server mode analysis in beSOURCE Developer 52

Analyzing Source Files in the Eclipse Plugin 59

 Setting up the Eclipse plugin 59

 Creating a Java project 61

 Local mode analysis in the Eclipse plugin 63

Analyzing Source Files in the IntelliJ IDEA Plugin 69

 Setting up the IntelliJ or Android Studio plugin 69

 Local mode analysis in the IntelliJ or Android Studio plugin 75

 Server mode analysis in IntelliJ or Android Studio plugin 77

Filtering Out False Alarms 80

 Filtering false alarms in beSOURCE Client 80

 Filtering false alarms in beSOURCE Developer 82

 Filtering false alarms in Eclipse plugin 83

 Active Filtering between beSOURCE Server and developer PC 85

 Automatic filtering in analysis engine 86

Integrating with Jenkins 91

 Preparing for Jenkins 91

 Installing the Jenkins plugin 92

 Installing the Jenkins plugin 93

 Connecting Jenkins plugin to beSOURCE Server 94

 Running beSOURCE Jenkins Plug-in 95

Integrating with Git 97

Setting for Git integration	97
Analyzing Java source files from Git in beSOURCE Server	98
Analyzing Java source files from Git in beSOURCE Developer	101
Integrating with SVN	104
Analyzing source files from SVN in beSOURCE Server	104
Analyzing source files from SVN in beSOURCE Developer	107
Integrating with TFS	110
Analyzing source files from TFS in beSOURCE Server	110
Analyzing source files from TFS in beSOURCE Developer	112
Using a User-defined Header	115
Overview	115
Finding a parsing error in case of missing macro and fixing it	117
Finding a parsing error in case of missing class declaration and fixing it	122
Finding a parsing error in case of missing function declaration and fixing it	124
Setting user defined header in beSOURCE Developer	127
Automate Importing and Analyzing Source Files	132
Scheduler of project	132
Making your own scheduler job	135
Using the SSL (HTTPS) protocol	143
Dashboard	148
Viewing the dashboard	148
Making a new dashboard	150
Web User Interface	153

Appendix **157**

 How to get debugging information for technical support 157

 How to switch JDK 159

Preparations

NOTE: Before using this guide, make sure you have already installed beSOURCE Enterprise. For more information about beSOURCE Enterprise installation, please refer to the beSOURCE Enterprise Installation Guide.

Running servers in Windows

NOTE: The beSOURCE Analysis Server and View Server must be run by the beSOURCE administrator.

Analysis Server

To run the beSOURCE Analysis Server in Windows, use one of two methods:

- From the Windows start menu, select **beSOURCE > Start {Name} Server**.
- In Windows, open **Command Prompt**, go to the **CE_HOME** directory, and then execute the **startServer.cmd** file. In the command window, verify the Analysis Server has been successfully executed.

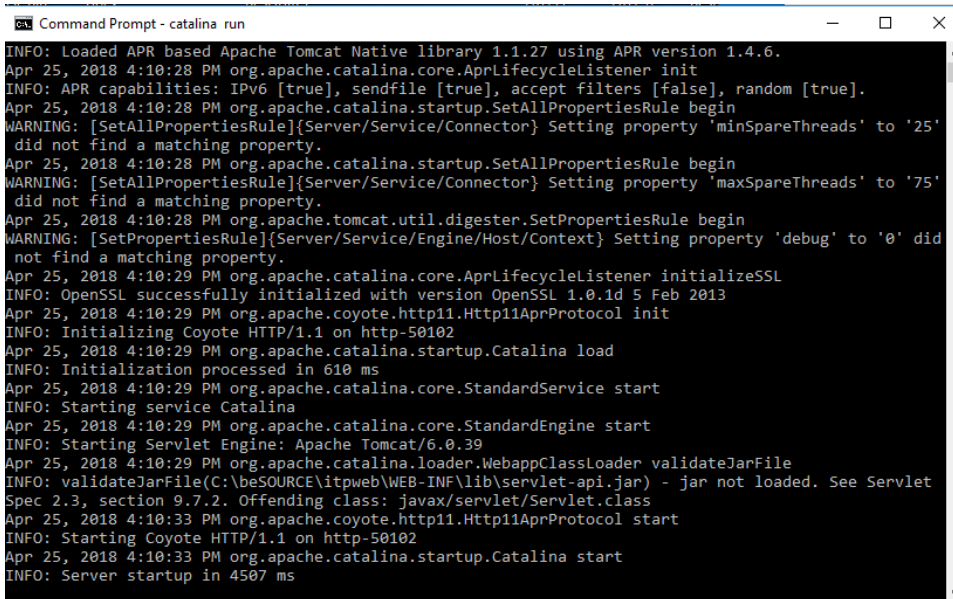
```

Start CE Server
**[13:55:27:053] 20180425000004 <DEBUG> w.processData-2 ms.
**[13:55:27:053] 20180425000004 <DEBUG> addJobSrcLog-0 ms.
**[13:55:27:053] 20180425000004 <INFO> N[510002] /testcases.h [ 449 ms Elapsed]
**GMC [Wed Apr 25 13:55:27 KST 2018]DEBUG : [MessageWriter-127.0.0.1:50103]
[13:55:27:451] CE <INFO> Registered JobServer-[20180425000004] /127.0.0.1:2528
**[13:55:28:365] 20180425000004 <TRACE : 5000> doPostTask-
**[13:55:28:365] 20180425000004 <INFO> update code-prism zip file-C:/beSOURCE/analyzer/SRC/src/filter/20180425000010.zip
(4)
**[13:55:28:365] 20180425000004 <INFO> make new code-prism zip file-C:/beSOURCE/analyzer/SRC/src/filter/20180425000010.zip
**[13:55:38:624] 20180425000004 <INFO> JOB-JOB IS ENDED. [20180425000004:1001:1000:20180425000008] C_LIB
**[13:55:38:624] 20180425000004 <INFO> ##### Object is -JobInfo : 0 : 1001 : 0 : 0/0/0({}{})
**[13:55:38:624] 20180425000004 <DEBUG> ChangeMiner-JOB ( 01 )-Job Thread Terminated.....
**[13:55:38:624] 20180425000004 <DEBUG> ChangeMiner-JOB ( 01 )-The Job Process was destroyed.
[13:55:39:572] CE <DEBUG> SESSION-/127.0.0.1 is closed.
[13:55:39:572] CE <TRACE : 5000> closeSession-
[13:55:39:588] CE <INFO> doJobProcess-END [20180425000004:1001:1000:20180425000008] C_LIB
[13:55:39:588] CE <INFO> JOB-JOB IS ENDED. [20180425000004:1001:1000:20180425000008] C_LIB
[13:55:39:588] 20180425000004 <INFO> SUMMARY-+++++
[13:55:39:588] 20180425000004 <INFO> SUMMARY-ID : 20180425000008
[13:55:39:588] 20180425000004 <INFO> SUMMARY-STEP : Collecting
[13:55:39:588] 20180425000004 <INFO> SUMMARY-NAME : C_LIB
[13:55:39:588] 20180425000004 <DEBUG> set up biz-true
[13:55:39:588] 20180425000004 <DEBUG> SET BIZ SRC-executed
[13:55:39:588] 20180425000004 <INFO> SUMMARY-Elapsed time is 0 min 15 sec.
[13:55:39:588] 20180425000004 <INFO> SUMMARY-+++++
[16:10:00:635] CE <DEBUG> SESSION-/127.0.0.1 is closed.
[16:10:00:635] CE <TRACE : 5000> closeSession-
[16:10:31:471] CE <DEBUG> Connected- from client : 127.0.0.1
  
```

View Server

To run the beSOURCE View Server in Windows, use one of two methods:

- From the Windows start menu, select **beSOURCE > Start View Server**.
- In Windows, open **Command Prompt**, go to the **vw_SERVER/bin** directory, and then execute the **startup.bat** file. In the command window, verify the View Server has been successfully executed.



```

Command Prompt - catalina run
INFO: Loaded APR based Apache Tomcat Native library 1.1.27 using APR version 1.4.6.
Apr 25, 2018 4:10:28 PM org.apache.catalina.core.AprLifecycleListener init
INFO: APR capabilities: IPv6 [true], sendfile [true], accept filters [false], random [true].
Apr 25, 2018 4:10:28 PM org.apache.catalina.startup.SetAllPropertiesRule begin
WARNING: [SetAllPropertiesRule]{Server/Service/Connector} Setting property 'minSpareThreads' to '25'
did not find a matching property.
Apr 25, 2018 4:10:28 PM org.apache.catalina.startup.SetAllPropertiesRule begin
WARNING: [SetAllPropertiesRule]{Server/Service/Connector} Setting property 'maxSpareThreads' to '75'
did not find a matching property.
Apr 25, 2018 4:10:28 PM org.apache.tomcat.util.digester.SetPropertiesRule begin
WARNING: [SetPropertiesRule]{Server/Service/Engine/Host/Context} Setting property 'debug' to '0' did
not find a matching property.
Apr 25, 2018 4:10:29 PM org.apache.catalina.core.AprLifecycleListener initializeSSL
INFO: OpenSSL successfully initialized with version OpenSSL 1.0.1d 5 Feb 2013
Apr 25, 2018 4:10:29 PM org.apache.coyote.http11.Http11AprProtocol init
INFO: Initializing Coyote HTTP/1.1 on http-50102
Apr 25, 2018 4:10:29 PM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 610 ms
Apr 25, 2018 4:10:29 PM org.apache.catalina.core.StandardService start
INFO: Starting service Catalina
Apr 25, 2018 4:10:29 PM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.39
Apr 25, 2018 4:10:29 PM org.apache.catalina.loader.WebappClassLoader validateJarFile
INFO: validateJarFile(C:\beSOURCE\itpweb\WEB-INF\lib\servlet-api.jar) - jar not loaded. See Servlet
Spec 2.3, section 9.7.2. Offending class: javax/servlet/Servlet.class
Apr 25, 2018 4:10:33 PM org.apache.coyote.http11.Http11AprProtocol start
INFO: Starting Coyote HTTP/1.1 on http-50102
Apr 25, 2018 4:10:33 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 4507 ms

```

If you want to create a detailed log of beSOURCE View Server in a specific log file, do the following:

1. In Windows, open **Command Prompt**.
2. Open the web application server's bin directory (for example, **CD C:\beSOURCE\WAS\bin**).
3. Run the View Server with following command: **catalina run > ../logs/catalina.out**. The **catalina.out** log file will be created in the **C:\beSOURCE\WAS\logs** folder.

Running servers in Linux

Analysis Server

To run the beSOURCE Analysis Server in Linux, do the following:

1. Go to the **CE_HOME** directory.
2. Execute **startServer.sh** file (for example, **/home/ci/besource/analyzer/server/CE] sh. /startServer.sh**)
3. Open the **nohup.out** file and verify if the Analysis Server is running.


```

For example, /home/ci/besource/analyzer/server/CE] tail -f
nohup.out

/home/ci/besource/analyzer/server/CE] tail -f nohup.out
[18:36:40:745] cm80 <INFO> Start beSOURCE Analyzer Server-
[18:36:40:745] cm80 <DEBUG> Thread for Log-Started
[18:36:40:745] cm80 <INFO> * CM -$Revision: 1.23 $
[18:36:40:745] cm80 <INFO> * vm-name -Java HotSpot(TM) Server
VM
[18:36:41:697] License <INFO> Load license file-
/besource/analyzer/license/license.xml
[18:36:42:080] License <INFO> Check Server License-
[18:36:42:115] License <INFO> ** -ok [18:36:42:937] cm80
<DEBUG> initialize Resource-
[18:36:43:001] cm80 <DEBUG> beSOURCE-Monitor-Started
[18:36:43:001] CMAalyzer <INFO> beSOURCE -Monitor-Thread
started ....
[18:36:49:953] cm80 <DEBUG> Database Information-oracle : 0 : 0
[18:36:50:757] cm80 <DEBUG> beSOURCE-Execute ( 01 )-Started
[18:36:50:760] cm80 <DEBUG> beSOURCE-Execute ( 02 )-Started
[18:36:50:788] cm80 <DEBUG> beSOURCE-JOB ( 01 )-Started
[18:36:50:817] CMScheduler <DEBUG> beSOURCE-Schedule-Started
[18:36:50:843] CMAalyzer <INFO> beSOURCE-Scheduler-Helper-
Started.....
[18:36:50:843] cm80 <DEBUG> Thread for Listen-Started
[18:36:50:844] cm80 <INFO> Listen-Open Port : 50103

```

View Server

To run the beSOURCE View Server in Linux, do the following:

1. Go to the **`vw_SERVER/bin`** directory.
2. Execute the **`startup.sh`** file (for example, `/besource/WAS/bin] sh ./startup.sh`).
3. Open the **`catalina.out`** file in the **`vw_SERVER/logs`** directory and verify if the View Server is running.

```

2017. 9. 10. PM 6:55:0
org.apache.coyote.http11.Http11BaseProtocol init Info:
Initializing Coyote HTTP/1.1 on http-50202
...
2017. 9. 10. PM 6:55:31

```

```
org.apache.coyote.http11.Http11BaseProtocol start Info:
Starting Coyote HTTP/1.1 on http-50102
2017. 9. 10. PM 6:55:32 org.apache.jk.common.ChannelSocket init
Info: JK: ajp13 listening on /0.0.0.0:9009
2017. 9. 10. PM 6:55:32 org.apache.jk.server.JkMain start Info:
Jk running ID=0 time=1/270 config=null
2017. 9. 10. PM 6:55:32
org.apache.catalina.storeconfig.StoreLoader load Info: Find
registry server-registry.xml at classpath resource
2017. 9. 10. PM 6:55:32 org.apache.catalina.startup.Catalina
start Info: Server startup in 31895 ms
```

Running the Admin Console

After installing the beSOURCE Admin Console, you can open it with the desktop shortcut icon. For information on installing the Admin Console, refer to the beSOURCE Enterprise Edition Installation Guide.

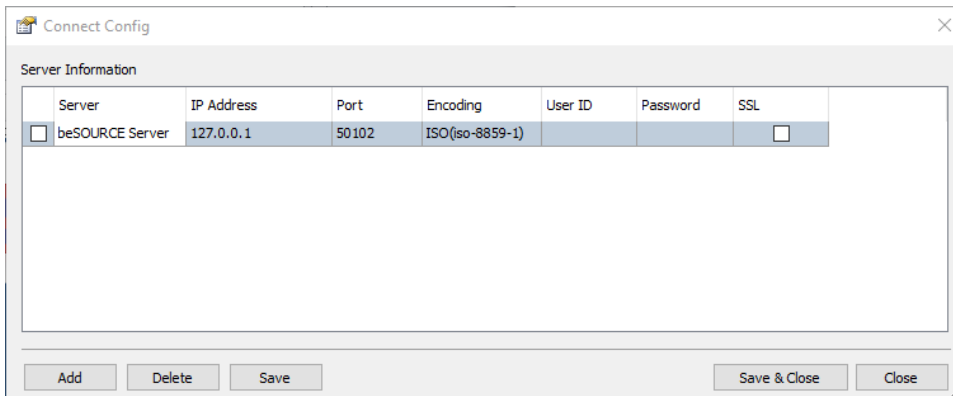


To run the Admin Console, do the following:

1. Double-click the **beSOURCE Management** shortcut icon on your desktop.
2. On the log in window, select **Config**.



3. Select **Add**.
4. In the **Server** box, enter your server's name.
5. In the **IP Address** box, enter the IP address of your beSOURCE View Server.
6. In the **Port** box, enter the port number of your beSOURCE View Server.
7. Select **Save** or **Save & Close**.



8. On the log in window, enter the administrator's **User ID** and **Password**. The default user ID is **besource** and the password is **besource**.
9. Select **Log In**.



Running the beSOURCE Client

After installing the beSOURCE Client, you can open it with its desktop shortcut icon.



Adding users

Before you can add users, you must create two user groups.

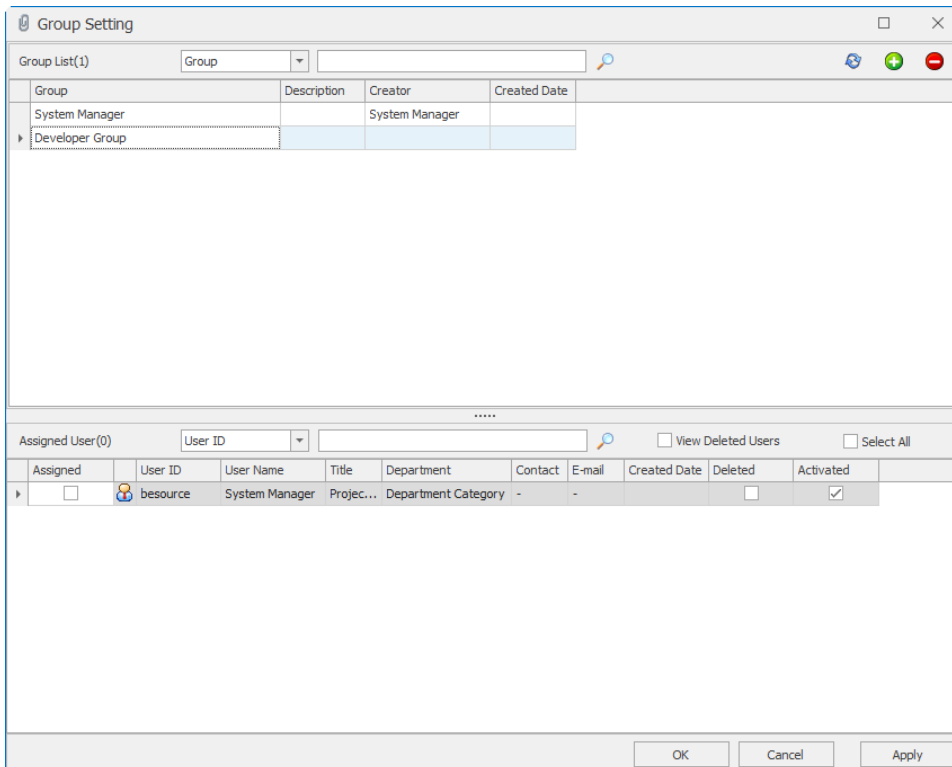
Creating user groups

To create user groups, do the following:

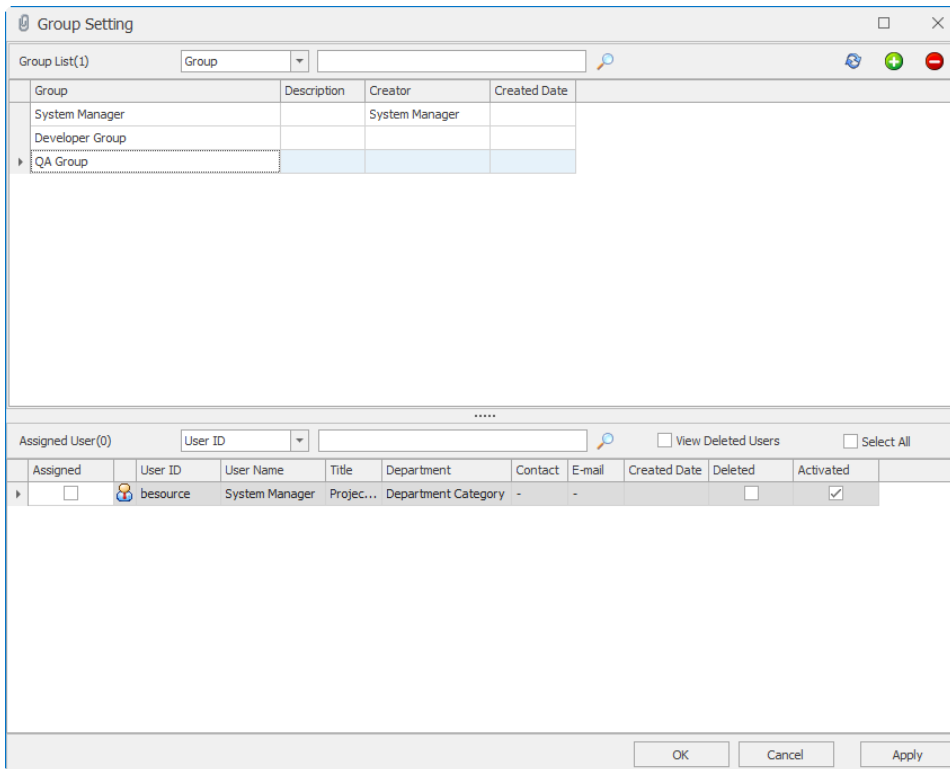
1. Select **User/Permission > Group Setting**.
2. Select the **Insert** icon (+) in the upper-right corner.
3. In the **Group** box, enter **Developer Group**.

NOTE: The System Manager group has all rights for beSOURCE administration. Do not remove the default group.

4. Select **Apply**.
5. In the dialog that appears, select **OK**.



6. Select the **Insert** icon (+) in the upper-right corner.
7. In the **Group** box, enter **QA Group**.



8. Select **Apply**.
9. In the dialog that appears, select **OK**.

Adding users

Now that you have created user groups, you can add users. To add users, do the following:

1. Select **User/Permission > User Setting**.
2. Select the **Insert** icon (+) in the upper-right corner.
3. In the following boxes, enter these values:
 - a. **User ID - besourcedev**
 - b. **User Name** - Your user name
 - c. **Password** - Your password
 - d. **Confirm Password** - Your password
 - e. **Title - Developer**

4. In the lower **Assigned Group** panel, select **Developer Group**.

The screenshot shows the 'User Setting' dialog box. It has two main sections: 'User List(1)' and 'Assigned Group(1)'. The 'User List(1)' section contains a table with columns: User ID, User Name, Password, Confirm Passw..., Title, Department, Contact, E-mail, Allow Duplicate Login, Authorized IP, and API Key. The 'Assigned Group(1)' section contains a table with columns: Assigned and Group.

User ID	User Name	Password	Confirm Passw...	Title	Department	Contact	E-mail	Allow Duplicate Login	Authorized IP	API Key
besource	System Manager			Project Manager	Department Cat...	-	-	<input checked="" type="checkbox"/>		
besourcedev	Kay	*****	*****	Developer	Department Cat...			<input checked="" type="checkbox"/>		

Assigned	Group
<input checked="" type="checkbox"/>	Developer Group
<input type="checkbox"/>	QA Group
<input type="checkbox"/>	System Manager

5. Select **Apply**.

6. In the dialog that appears, select **OK**.

7. Select the **Insert** icon (+) in the upper-right corner.

8. In the following boxes, enter these values:

- **User ID** - besourceqa
- **User Name** - Your user name
- **Password** - Your password
- **Confirm Password** - Your password
- **Title** - Engineer

9. In the lower **Assigned Group** panel, select **QA Group**.

User ID	User Name	Password	Confirm Passw...	Title	Department	Contact	E-mail	Allow Duplicate Login	Authorized IP	API Key
besource	System Manager			Project Manager	Department Cat...	-	-	<input checked="" type="checkbox"/>		
besourcedev	Kay	*****	*****	Developer	Department Cat...			<input checked="" type="checkbox"/>		
besourceqa	Maria	*****	*****	Engineer	Department Cat...			<input checked="" type="checkbox"/>		

Assigned	Group
<input type="checkbox"/>	Developer Group
<input checked="" type="checkbox"/>	QA Group
<input type="checkbox"/>	System Manager

10. Select **Apply**.

11. In the dialog that appears, select **OK**.

To change the user and password management policy, select **Tools > Options > User ID/Password Management Rules**. You can set a variety of options like below.

User ID/Password Management Rules

User ID Management Rules

- Minimum/Maximum length: 5 ~ 36
- Not Allowing Number
- Not Allowing Special Character

Password Management Rules

- Minimum/Maximum length: 5 ~ 50
- Not Allowing Same Password as ID
- Include At Least 1 Alphabet
- Include At Least 1 Lowercase Character
- Include At Least 1 Uppercase Character
- Include At Least 1 Numeric Character
- Include At Least 1 Special Character


Analyzing Java Source Files

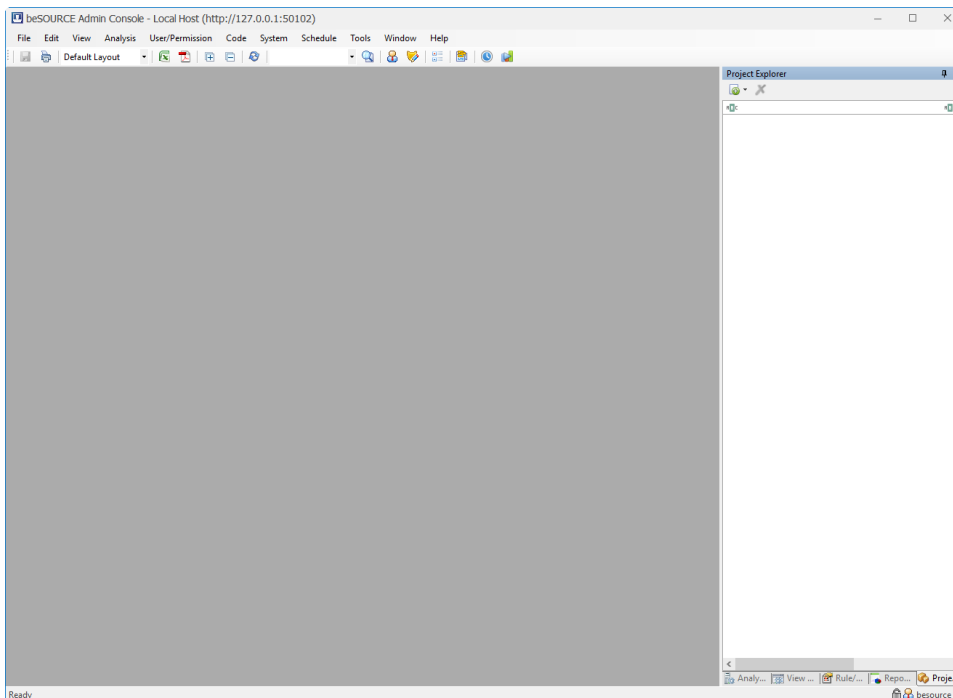
Overview

This chapter how to analyze Java sample source files on the server and view their results.

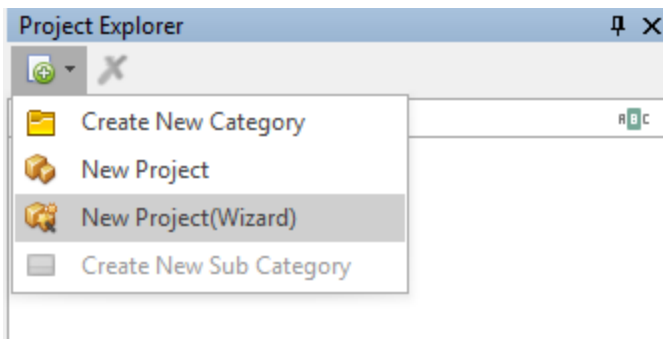
Analyzing Java sample source files

To analyze Java sample source files,

1. Open and log in to the **beSOURCE Admin Console**.
2. Under **Project Explorer**, select the **Add Project Items** () icon.

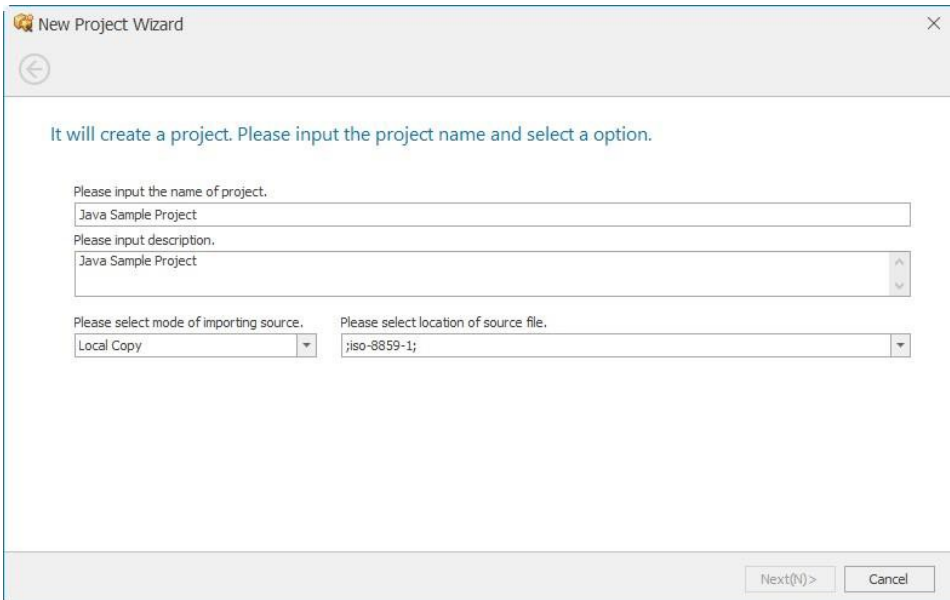


3. Select **New Project(Wizard)**. The Project Wizard opens.

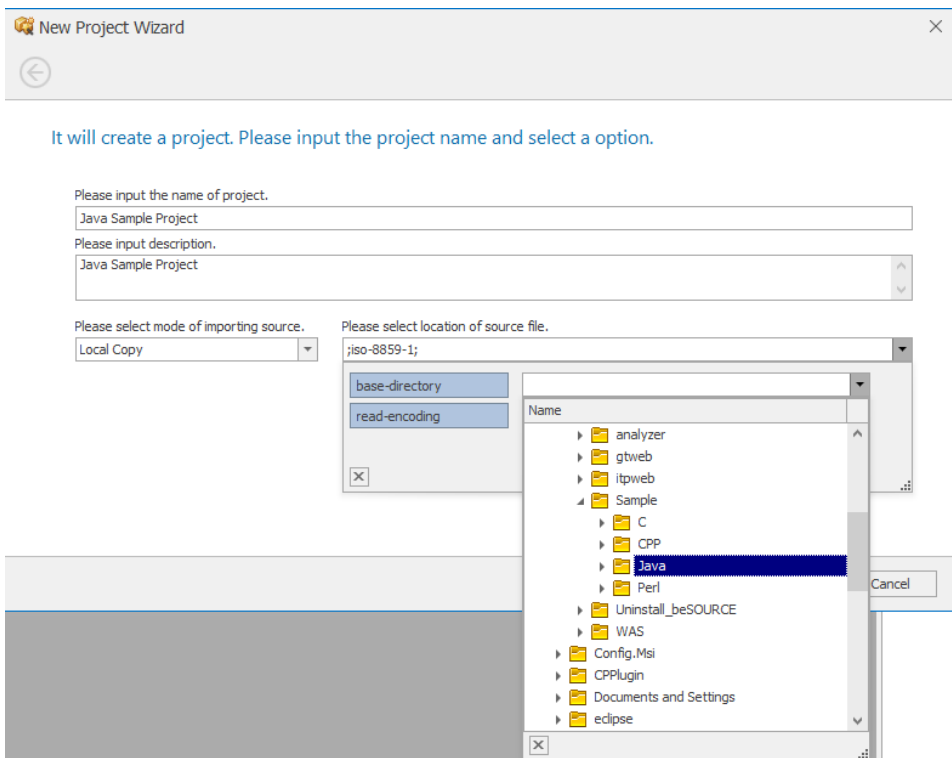


4. In the **Please input the name of the project** box, enter a name.
5. In the **Please input description** box, enter a description for the project.

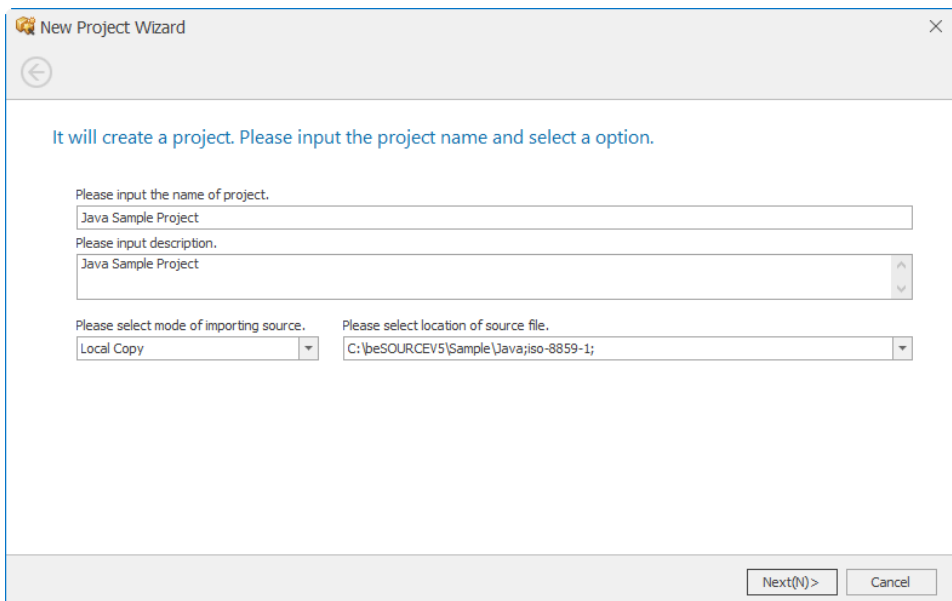
NOTE: The **Local Copy** option in the **Importing mode** box refers to the beSOURCE server computer's local hard disk.



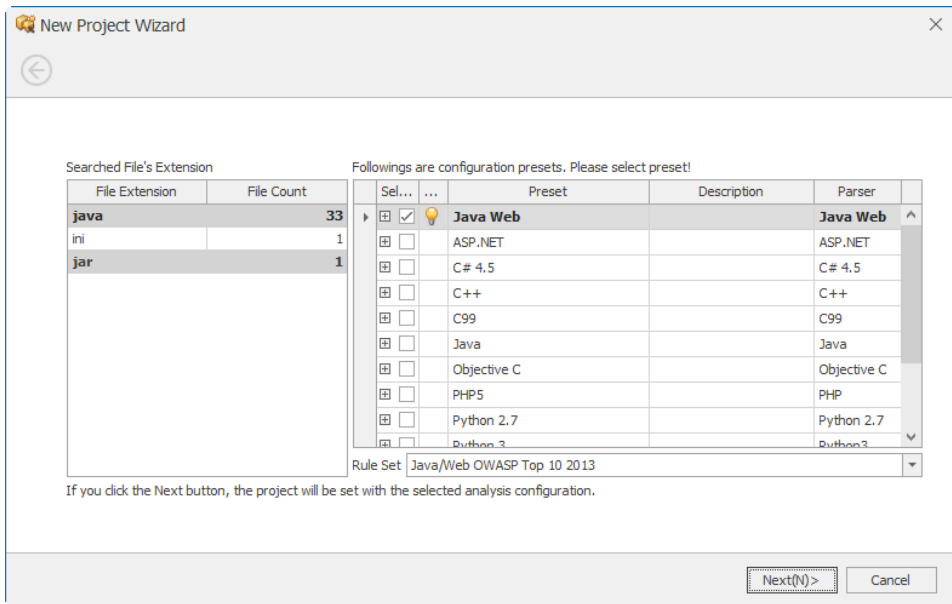
6. Select the **Please select location of source file** box.
7. Select the **base-directory** box.
8. Move to your beSOURCE Server (Enterprise Edition) installation folder (for example, C:\beSOURCE\).
9. Select and double-click the **sample\Java** folder.



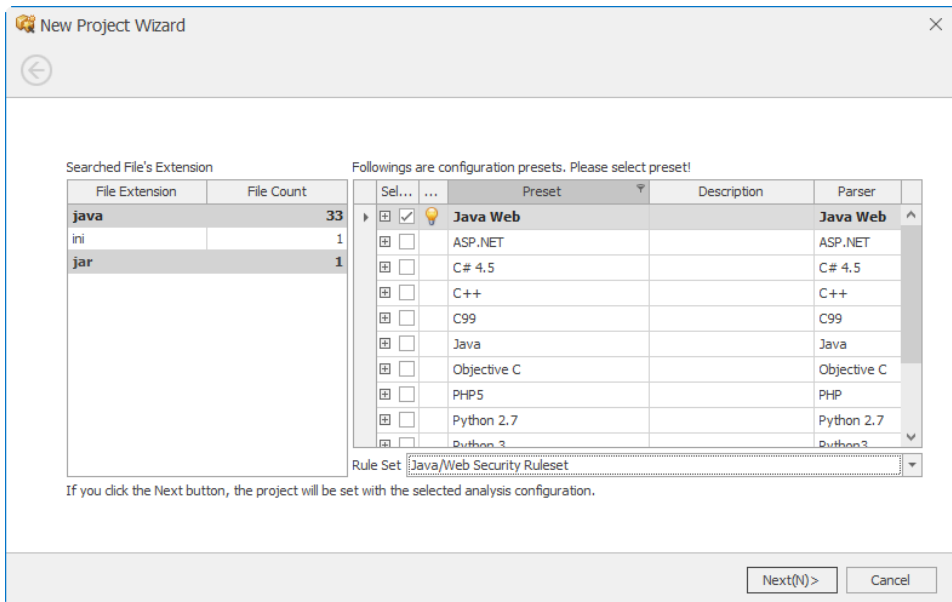
10. Select **Next**.



- The Project Wizard scans the source files and recommends a preset configuration. You can change the configuration preset depending on your project language. For example, if your source files are just java files without any XML, JS, JSP and so on, you can select Java preset instead of Java Web preset.

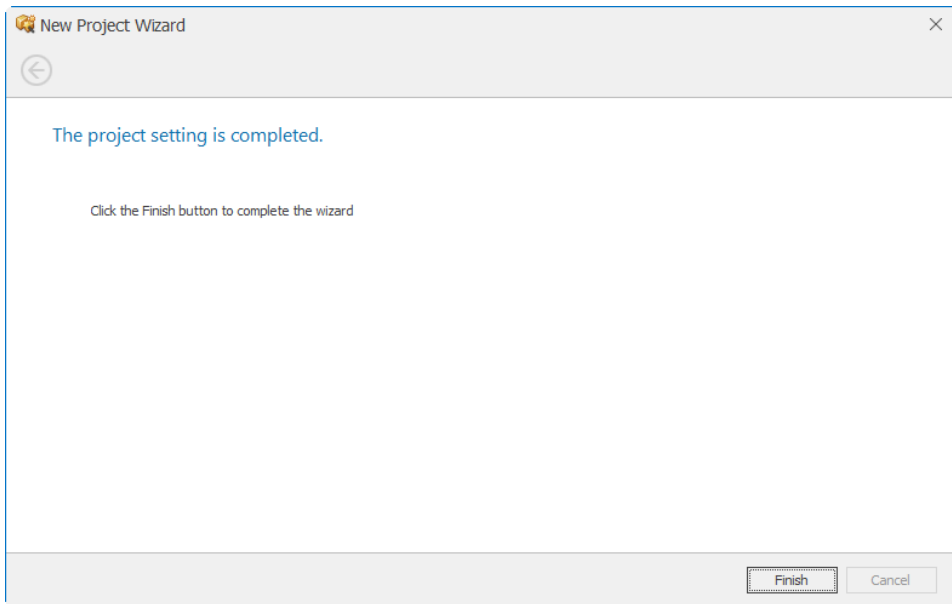


12. In the **Rule Set** box, select a rule set.

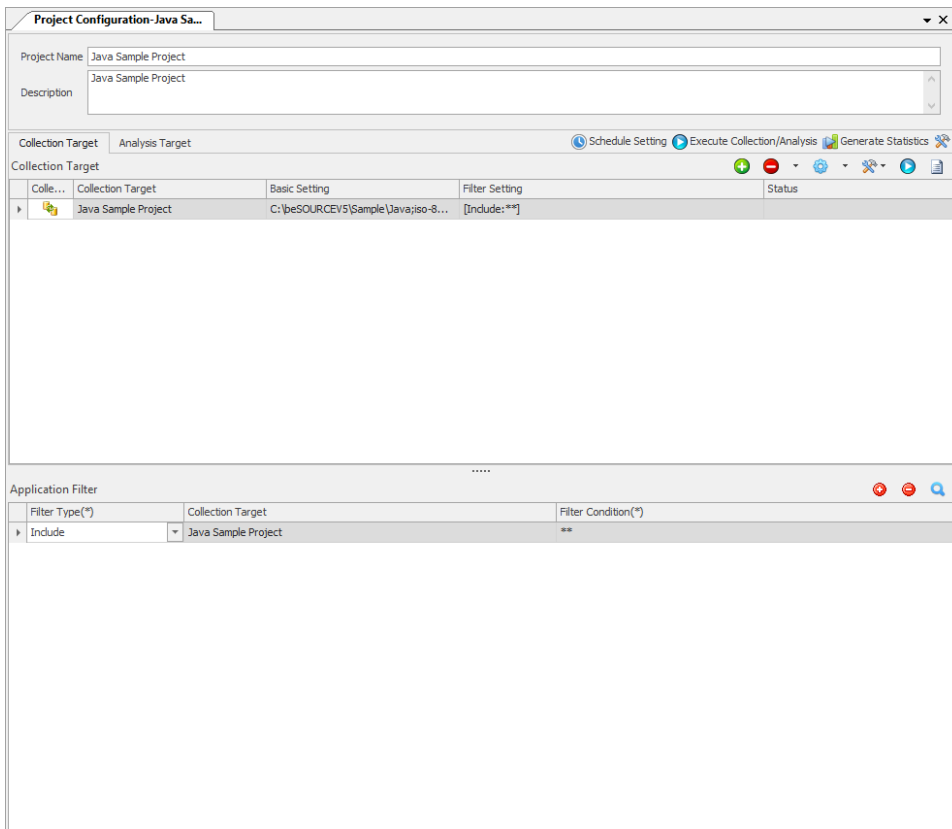


13. Select **Next**.

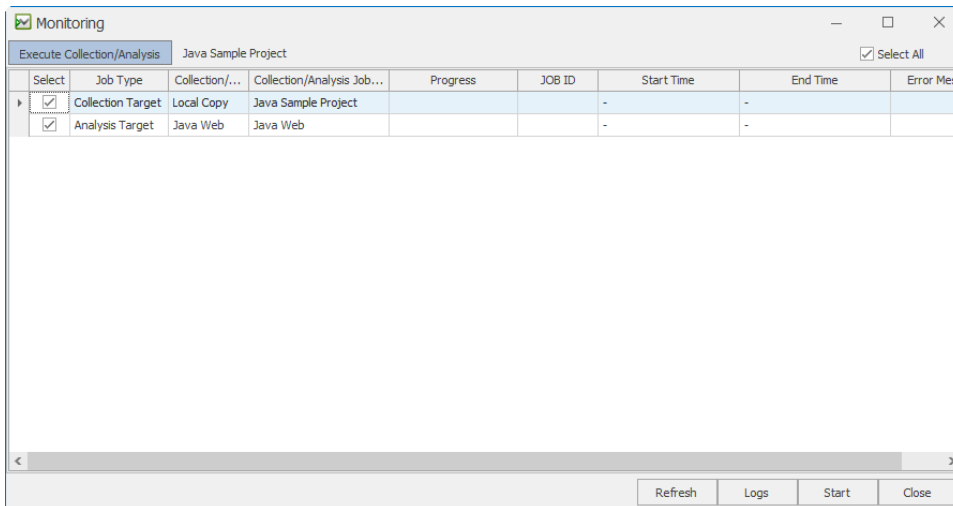
14. Select **Finish**. The Project Configuration window opens.



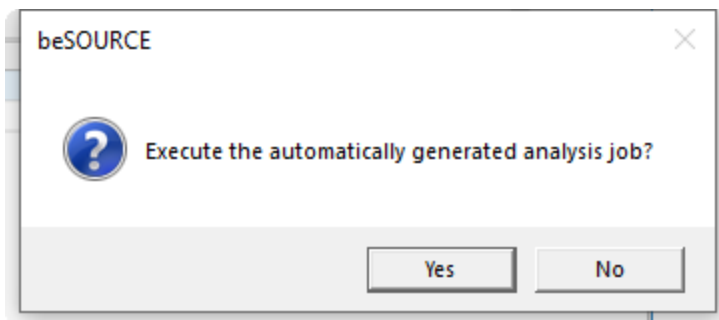
15. Select **Execute Collection/Analysis** ( Execute Collection/Analysis).



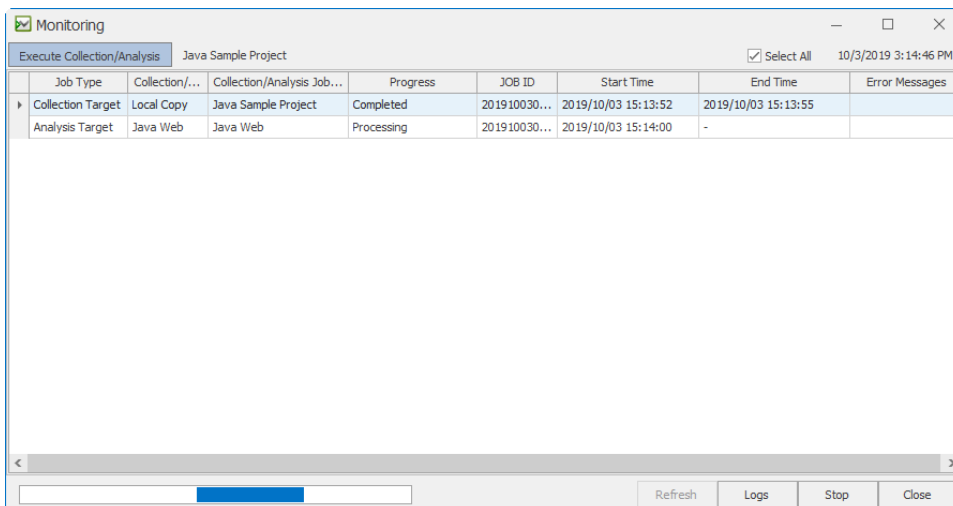
16. On the **Monitoring** window, select **Start**.



17. On the dialog, select **Yes**.



18. The Monitoring window shows the progress of importing and analyzing source files. You can safely close the Monitoring window once the collection and analysis jobs are begin.

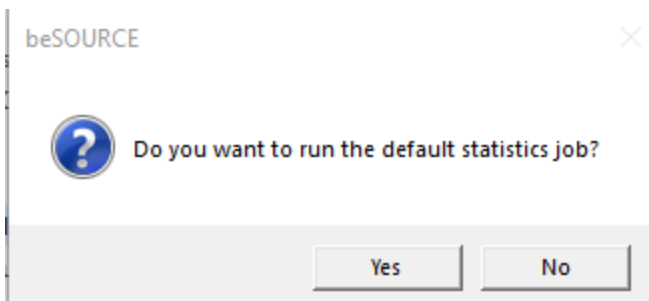


19. To see detail job logs, select **Logs**. The Monitoring window will close and the View Job Log window will open. Select **View** to refresh the window.

NOTE: The analysis job will ~10 minutes depending on your server's performance.

Status	Type	Job Target	Message	Structure Analysis			USA			Metric			Security
				All	Success	Fail	All	Success	Fail	All	Success	Fail	
Processing	Manual Job	Java Web		33	33	0	0	0	0	0	0	0	0

NOTE: If you didn't close the monitoring window, a dialog will appear asking if you want to "run default statistics job." Selecting Yes starts a job on the beSOURCE Server that will generate default statistics for dashboard and reports.



Viewing Analysis Results

Using Application Explorer

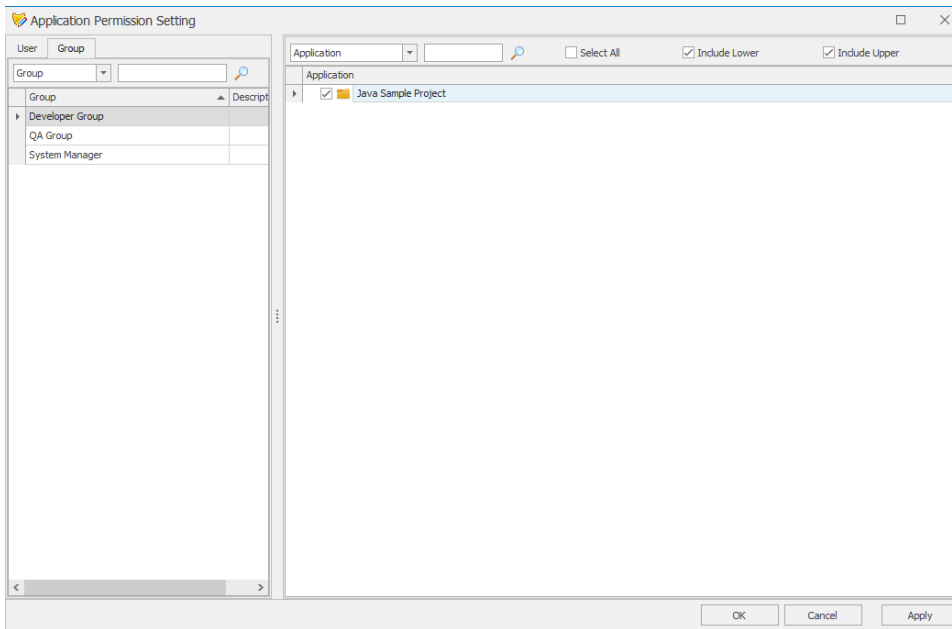
Users can view the analysis inspection results on the beSOURCE Server using beSOURCE Client.


To view the inspection results, do the following:

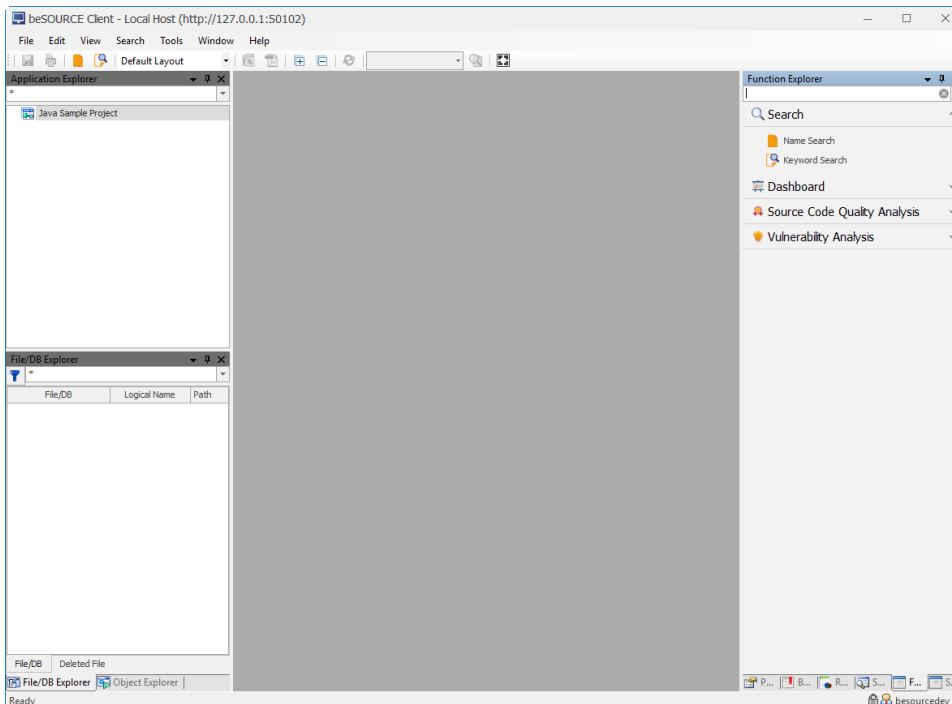
1. Open **beSOURCE Client**.
2. Log in as a beSOURCE developer user. You will not see a project or application in the Application Explorer. The administrator will need to grant access permission to the user.



3. Open the **beSOURCE Admin Console** again.
4. Select **User/Permission > Application Permission Setting**.
5. Select the **Group** tab.
6. Under **Group**, select the **Developer Group**.
7. Under **Application**, select **Java Sample Project**.
8. Select **Apply**.
9. Select **OK**.

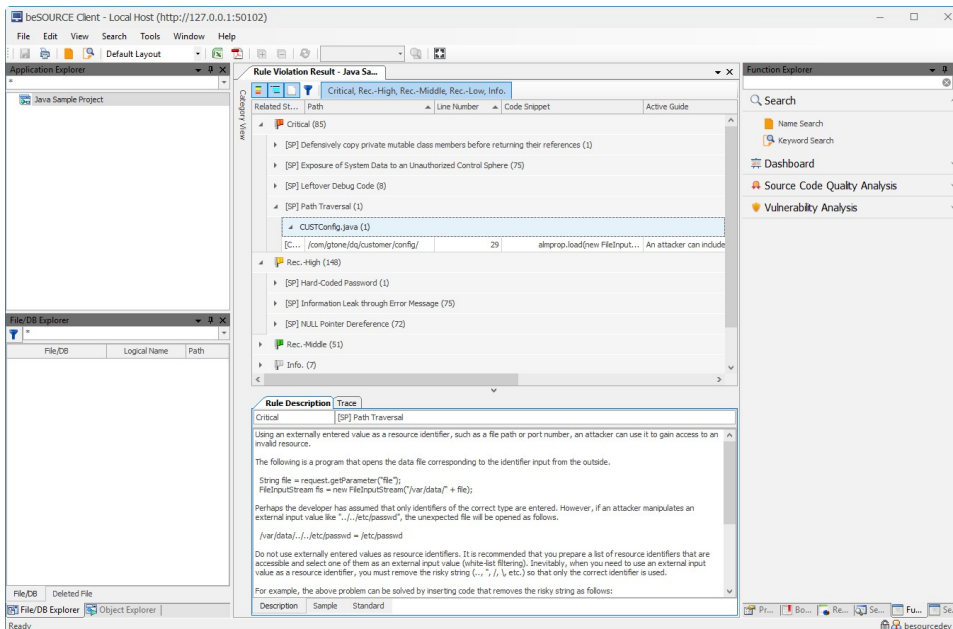


10. Open the **beSOURCE Client** again.
11. Select **View > Application Explorer**.
12. Select the **Refresh** () icon in the top tool bar. The developer user can now see the Java Sample Project.

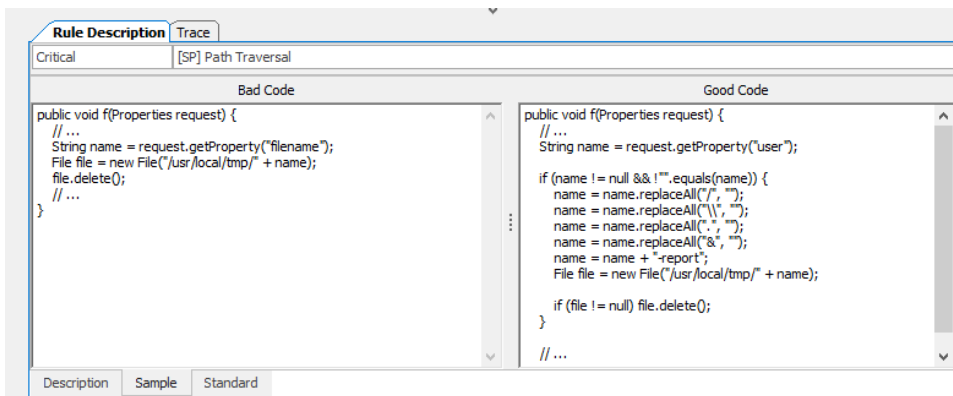


13. Select the **Java Sample Project**.
14. Select **Code Quality/Security Inspection > Show Rule Violations**. The Rule Violation Result window appears.

15. Select a violation from the list (for example, Path Traversal). The Rule Description panel will display the description of the rule.



16. Under the **Rule Description** tab, select the **Sample** tab in the lower area to view good code and bad code examples.



17. Select the Standard tab in the lower area to view the related international standard about the violation. For example, the Path Traversal is identified with CWE (Common Weakness Enumeration) ID 22.

Rule Description		Trace	
Critical	[SP] Path Traversal		
Related Standards	No	Content	Reference
CWE	22	Improper Limitation of a Pathname to a Restricted Directory (Path Traver...	https://cwe.mitre.org/data/definitions/22.html
CWE	99	Improper Control of Resource Identifiers (Resource Injection)	https://cwe.mitre.org/data/definitions/99.html
CWE/SANS Top 25 2011	13	Improper Limitation of a Pathname to a Restricted Directory (Path Traver...	http://cwe.mitre.org/top25/archive/2011/2011...
OWASP Top 10 2004	A6	Injection Flaws	https://www.owasp.org/index.php/A6_2004_1...
OWASP Top 10 2007	A2	Injection Flaws	https://www.owasp.org/index.php/Top_10_20...
OWASP Top 10 2007	A4	Insecure Direct Object Reference	https://www.owasp.org/index.php/Top_10_20...
OWASP Top 10 2010	A1	Injection	https://www.owasp.org/index.php/Top_10_20...
OWASP Top 10 2010	A4	Insecure Direct Object References	https://www.owasp.org/index.php/Top_10_20...
OWASP Top 10 2013	A1	Injection	https://www.owasp.org/index.php/Top_10_20...
OWASP Top 10 2013	A4	Insecure Direct Object References	https://www.owasp.org/index.php/Top_10_20...
OWASP Top 10 2017	A1	Injection	https://www.owasp.org/index.php/Top_10-201...
OWASP Top 10 2017	A4	Insecure Direct Object References	https://www.owasp.org/index.php/Top_10-201...
Description	Sample	Standard	

18. Double-click the rule violation **Path Traversal: CUSTConfig.java Line Number 29**.

19. The Source Viewer will appear and highlight the corresponding code line.

NOTE: Positioning your pointer on the rule name will display its description.

20. Select the **Trace Info.** link for the Path Traversal violation. The flow **Trace** tab will appear.

NOTE:

- Double-clicking a code line will display the corresponding code.
- The last code line in the list is a violation point and the first code line in the list is a starting point of method call sequences. For example, line number 29 of `CUSTConfig.java` in the above sample is a violation point. Line number 120 of `SysUtil.java` in the above sample is the starting point of method call sequences.

The screenshot shows a window titled "Rule Detail Information" with a "Trace" tab selected. It contains a table with columns: Name, Path, Line, Column, and Code Snippet. The table lists several code snippets from `SysUtil.java` and `CUSTConfig.java`. The snippet at line 29 of `CUSTConfig.java` is highlighted in yellow, indicating a violation point. Below the table, there is a description of the rule violation and a code snippet showing the replacement of a dangerous string.

Name	Path	Line	Column	Code Snippet
Trace Info. Group : 1				
SysUtil.java	/com/fgtone/dq/customer/uttl/	120	55	while ((line = reader.readLine()) != null)
The reader.readLine() value is a string that an attacker can manipulate. Do not make paths with these strings. Attackers can attempt to manipulate strings to create a path manipulation attack that creates the path they want. If you need to configure the path with this value, remove the dangerous string as follows:				
<pre>while ((line = reader.readLine().replaceAll("\\\\", "")) != null)</pre>				
SysUtil.java	/com/fgtone/dq/customer/uttl/	120	33	while ((line = reader.readLine()) != null)
SysUtil.java	/com/fgtone/dq/customer/uttl/	121	43	sb.append(line + sep);
SysUtil.java	/com/fgtone/dq/customer/uttl/	121	43	sb.append(line + sep);
SysUtil.java	/com/fgtone/dq/customer/uttl/	121	42	sb.append(line + sep);
SysUtil.java	/com/fgtone/dq/customer/uttl/	123	43	return sb.toString().trim();
SysUtil.java	/com/fgtone/dq/customer/uttl/	123	50	return sb.toString().trim();
SysUtil.java	/com/fgtone/dq/customer/uttl/	80	50	envval = getValue(ps.getInputStream());
SysUtil.java	/com/fgtone/dq/customer/uttl/	80	33	envval = getValue(ps.getInputStream());
SysUtil.java	/com/fgtone/dq/customer/uttl/	106	32	return envval;
SysUtil.java	/com/fgtone/dq/customer/uttl/	57	49	String envvalue = getEnv(envname);
SysUtil.java	/com/fgtone/dq/customer/uttl/	57	32	String envvalue = getEnv(envname);
SysUtil.java	/com/fgtone/dq/customer/uttl/	58	32	return envvalue + path.substring(index);
SysUtil.java	/com/fgtone/dq/customer/uttl/	58	32	return envvalue + path.substring(index);
CUSTConfig.java	/com/fgtone/dq/customer/config/	29	85	almprop.load(new FileInputStream(SysUtil.getRealPath(Constants.ALARM...

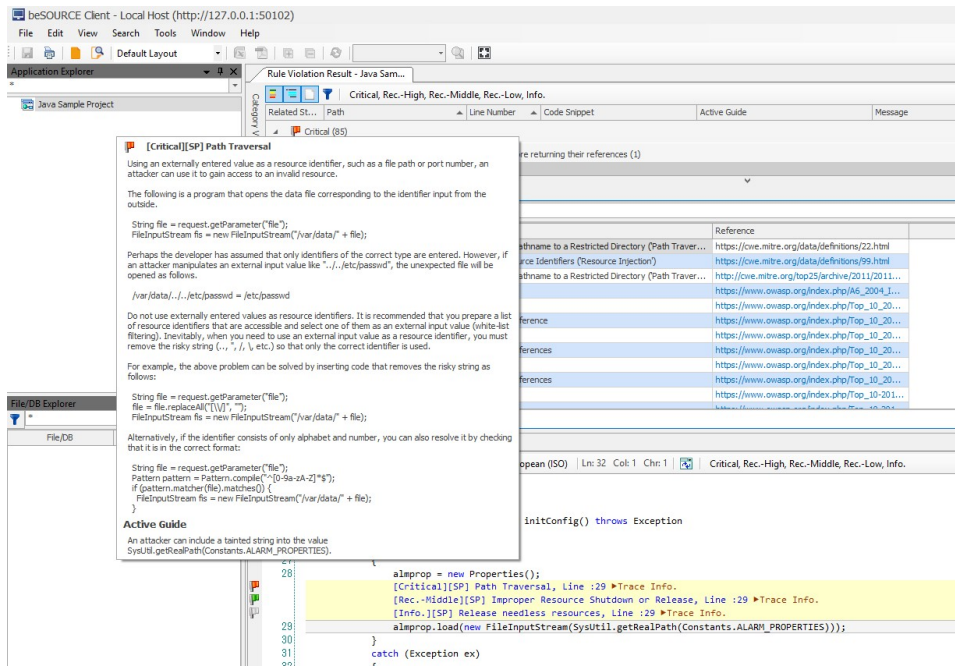
An attacker can include a tainted string into the value SysUtil.getRealPath(Constants.ALARM_PROPERTIES).

21. Check the code correction guide which is based on actual source code and is an Active Guide feature.

The screenshot shows the "Rule Detail Information" dialog box with the "Trace" tab selected. The active guide is displayed in a separate window, showing the code snippet and the replacement string.

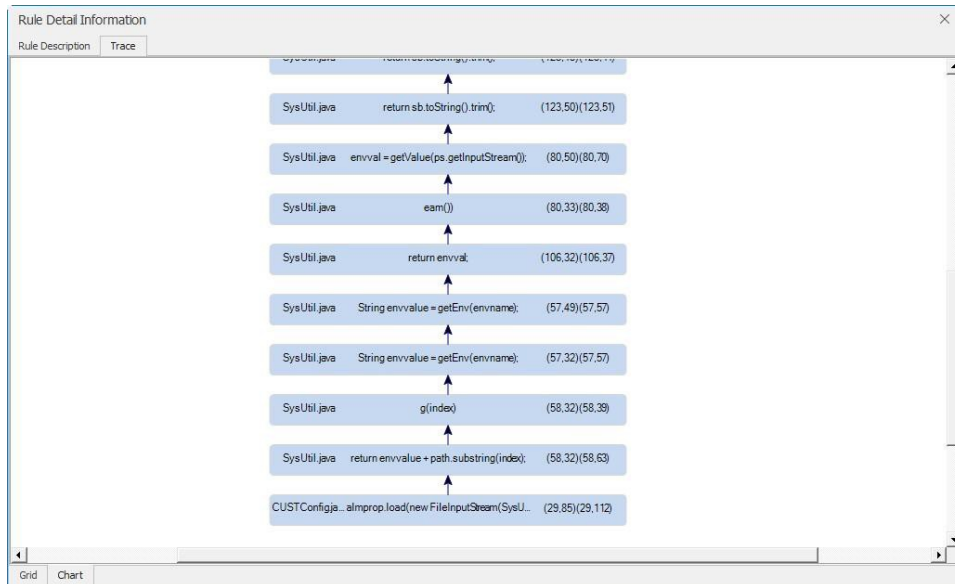
Name	Path	Line	Column	Code Snippet
Trace Info. Group : 1				
SysUtil.java	/com/fgtone/dq/customer/uttl/	120	55	while ((line = reader.readLine()) != null)
The reader.readLine() value is a string that an attacker can manipulate. Do not make paths with these strings. Attackers can attempt to manipulate strings to create a path manipulation attack that creates the path they want. If you need to configure the path with this value, remove the dangerous string as follows:				
<pre>while ((line = reader.readLine().replaceAll("\\\\", "")) != null)</pre>				

NOTE: Positioning the point on a rule violation will display its description and active guide.



22. Under the Trace tab, select the **Chart** tab in the lower area. The flow trace information will display in a diagram format.

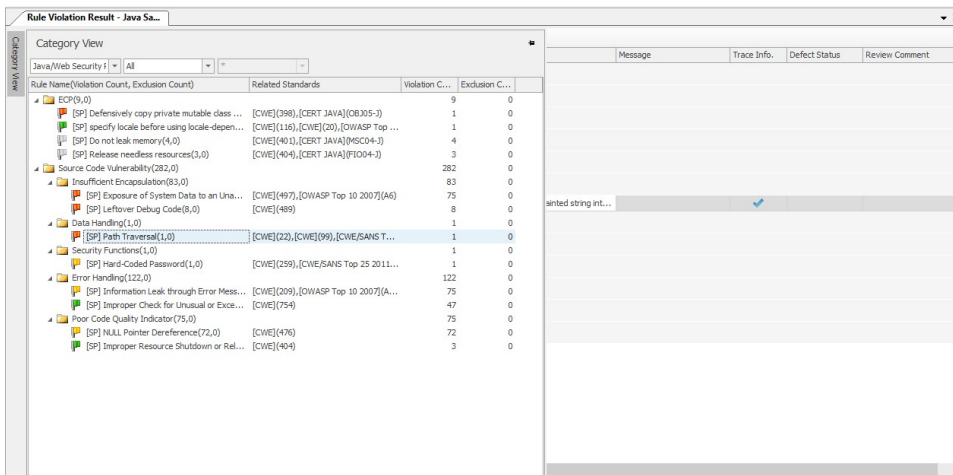
NOTE: The last code line in the chart is a violation point and the first code line in the chart is a starting point of method call sequences. When you double-click each code line, the Source Viewer will display the corresponding code.



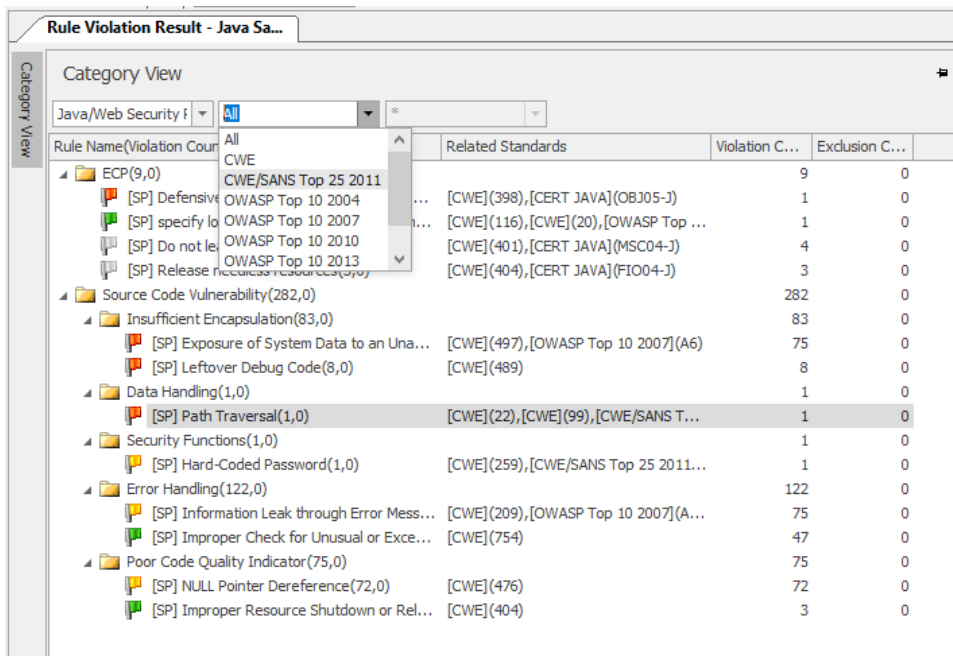
23. Close the **Rule Detail Information** window.

24. Close the **Source Viewers**.

25. Select the **Category View** tab in the upper-left corner. A sliding window appears.

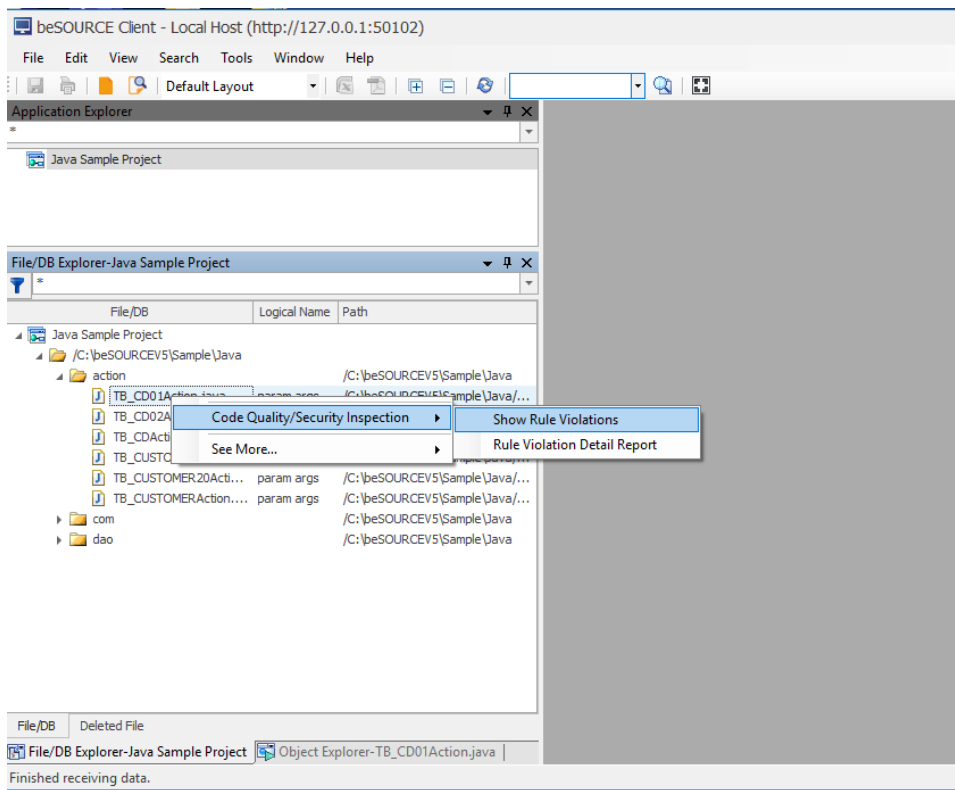


26. You can select a secure coding standard from the middle box. When you select a standard, only violations belonging to it will appear. Dimmed violations are not applicable.



27. Select any window other than Category View to hide it.

NOTE: If you double-click a project in the Application Explorer, the File/DB Explorer shows the list of source files. By using the shortcut menu of a source file in the File/DB Explorer, you can see the analysis results. When you select a source file in the File/DB Explorer and open the Object Explorer, it will show the parsing tree of the selected source file.

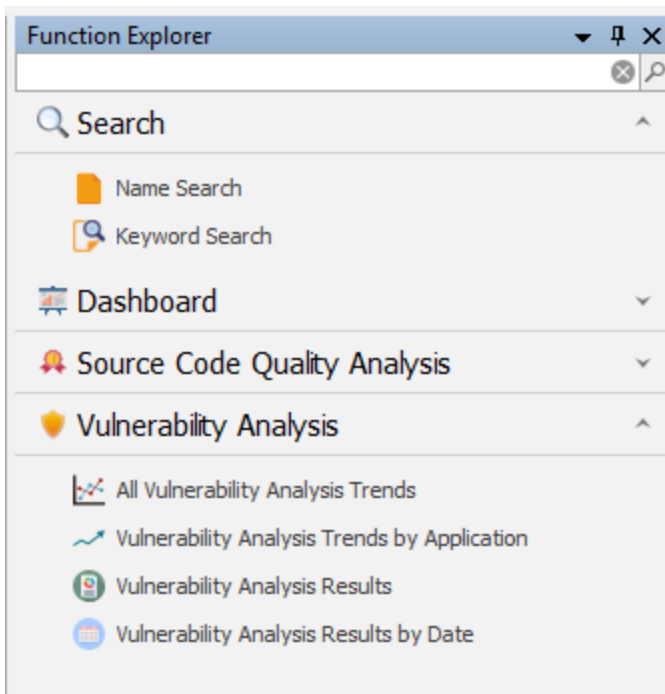


Using Function Explorer

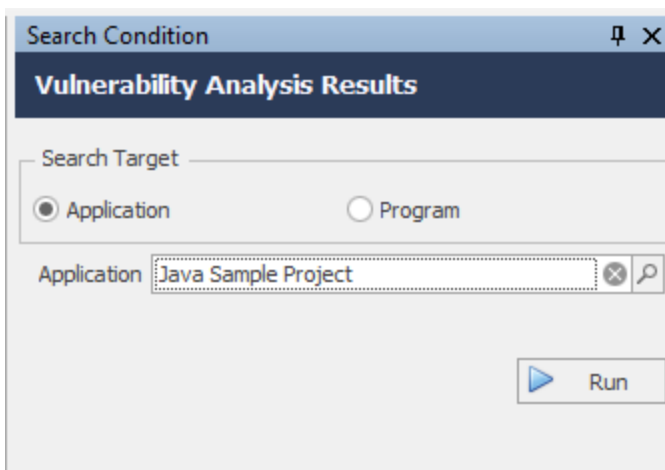
The Function Explorer provides another way to see server-side analysis results with the beSOURCE Client.

To view results with the Function Explorer, do the following:

1. Select **View > Function Explorer**.
2. Select **Vulnerability Analysis > Vulnerability Analysis Results**.



3. In the **Search Condition** window, select a project.
4. Select **Run**.



5. Any project rule violations will appear.

NOTE: You can adjust the UI layout by dragging and dropping each tab.

The screenshot displays the beSOURCE Client interface with the following components:

- Application Explorer:** Shows a tree view of the project structure, including 'Java Sample Project'.
- Function Explorer:** Provides search options like 'Name Search', 'Keyword Search', and 'Dashboard'.
- Vulnerability Analysis Results:** The main pane shows a list of findings categorized by severity (Critical, Rec.-High, Rec.-Middle, Rec.-Low, Info). The selected finding is:
 - Severity:** Critical (85)
 - Rule:** [SP] Defensively copy private mutable class members before returning their references (1)
 - Path:** DAO.java (1)
 - Code Snippet:**

```

return com;

```
- Summary Panel:** Provides an overview of the analysis:
 - Application Name: Java Sample Project
 - File/DB Object Name: TS_CDAction.java
 - Logical Name: param arg
 - Path: C:\beSOURCE\DEV\Sample\Java\actor
 - Last Collection/Analysis: 10/5/2019
 - Object Type: Java Source File
 - Analysis Succeeded: Y
 - Number of Lines: 18
- Rule Description:** A detailed view of the selected rule, explaining that returning references to internal mutable members can compromise security and recommending defensive copying.


To generate reports, see [Generating Reports on page 42](#).

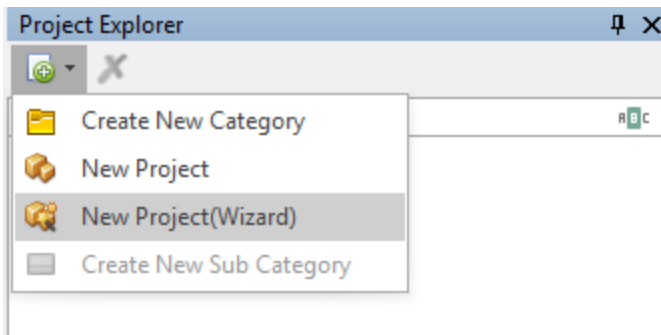
Analyzing C Source Files

This chapter describes how to analyze C sample source files on the server and view their results.

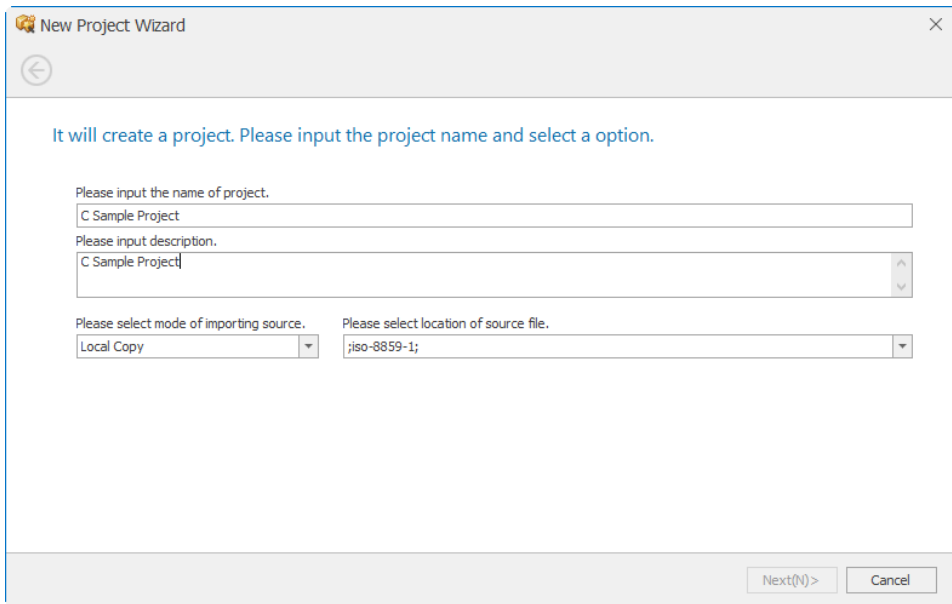
Analyzing C sample source files

To analyze C sample source files, do the following:

1. Open and log in to the **beSOURCE Admin Console**.
2. Select **View > Project Explorer**.
3. Select the **Add Project Items** () icon in the Project Explorer (without selecting any project).
4. Select **New Project(Wizard)**.



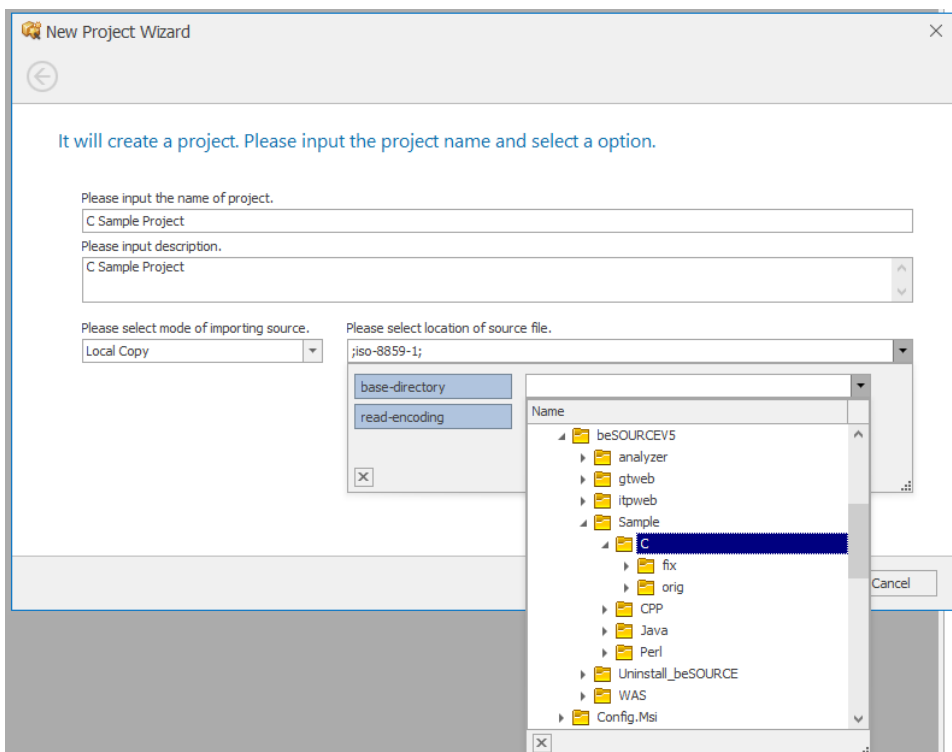
5. In the **Please input the name of the project** box, enter a name.
6. In the **Please input description** box, enter a description for the project.



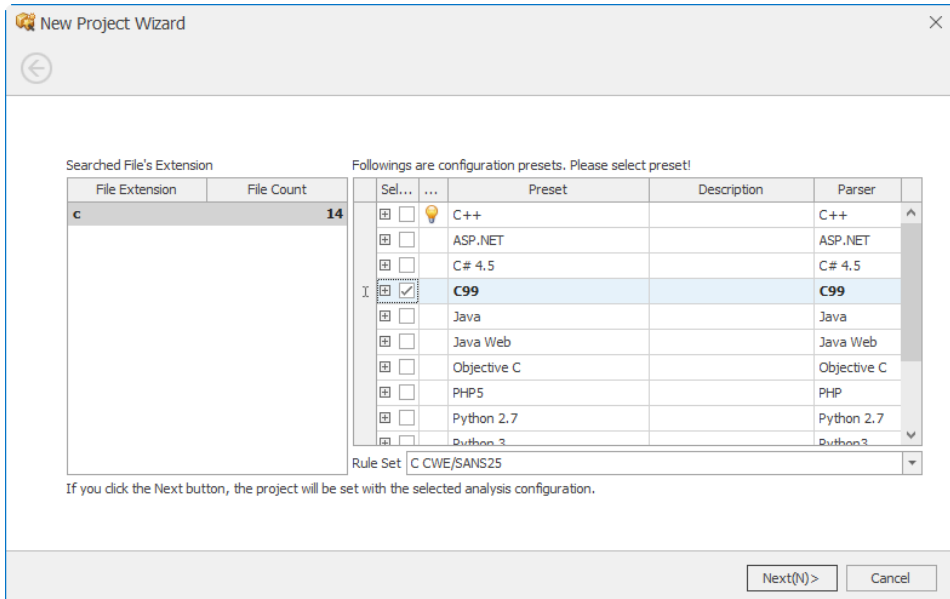
7. Select the **Please select location of source file** box, and then select the **base-directory** list box.

8. **NOTE:** The **Local Copy** option in the **Please select mode of importing source** box refers to the beSOURCE server's hard drive.

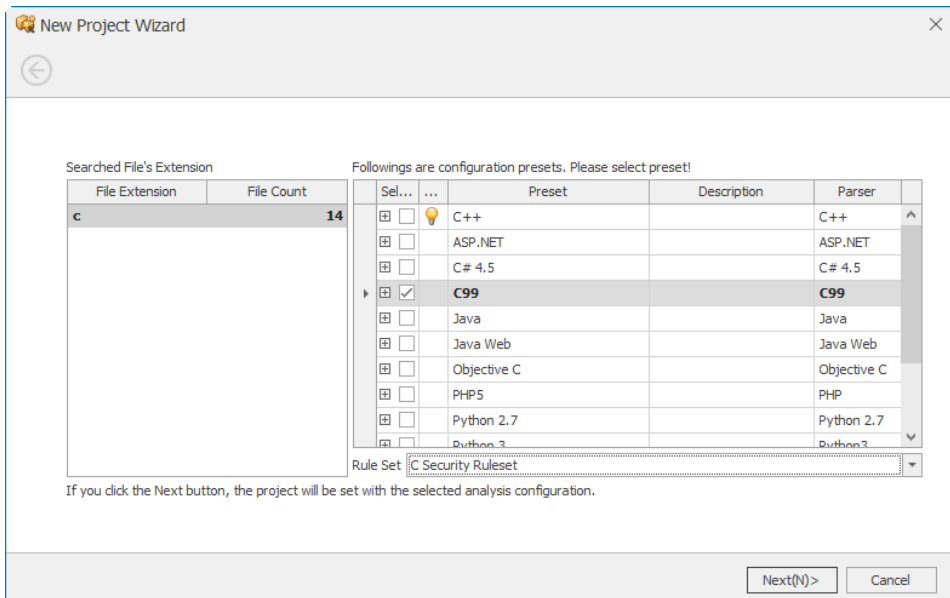
9. Select the **Sample\C** folder located within your beSOURCE Server installation directory (for example, `C : \beSOURCE\Sample\C`).




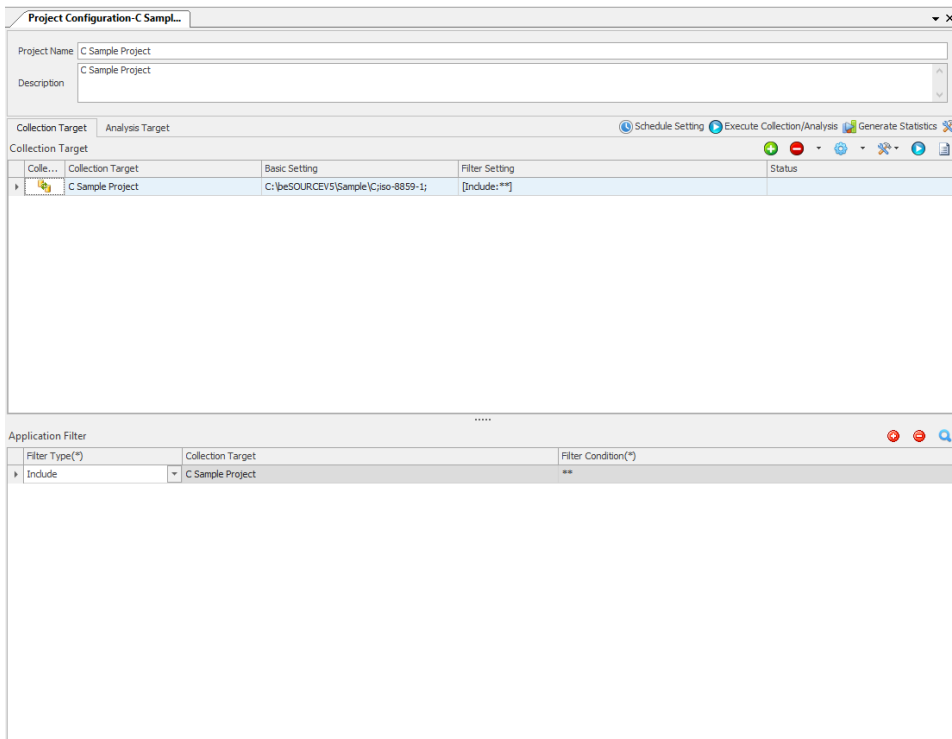
10. Select **Next**.
11. The Project Wizard will scan the source files and then recommend a preset of configuration. Select the **C99** preset (C language standard).



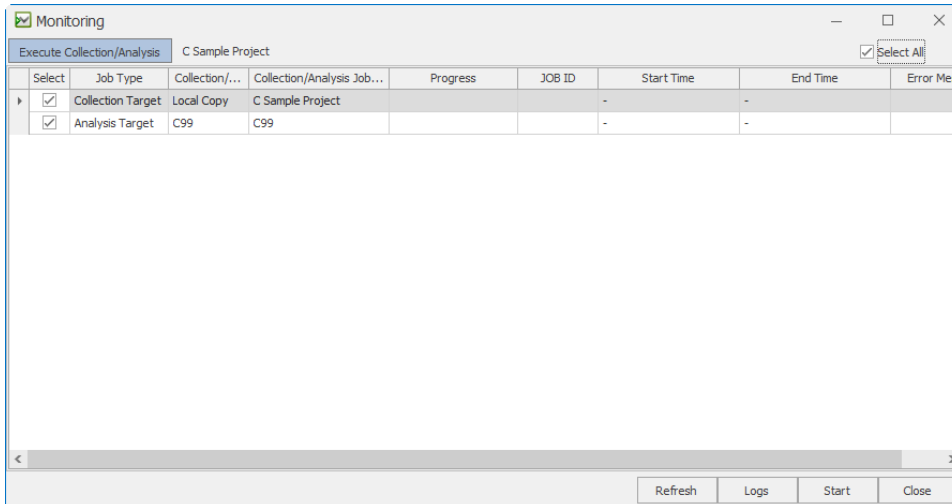
12. In the **Rule Set** box, select the desired rule set.



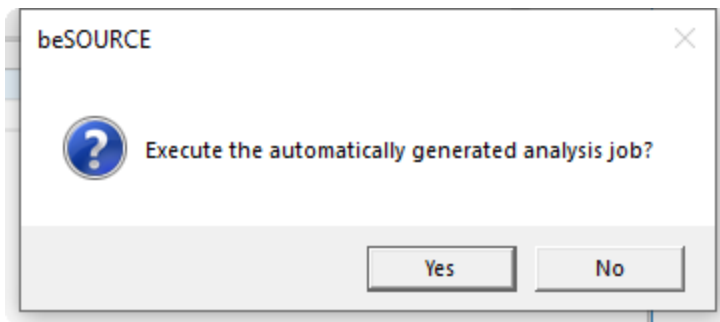
13. Select **Next**.
14. Select **Finish**. The Project Configuration window opens.
15. Select the **Execute Collection/Analysis** () icon. The Monitoring window opens.



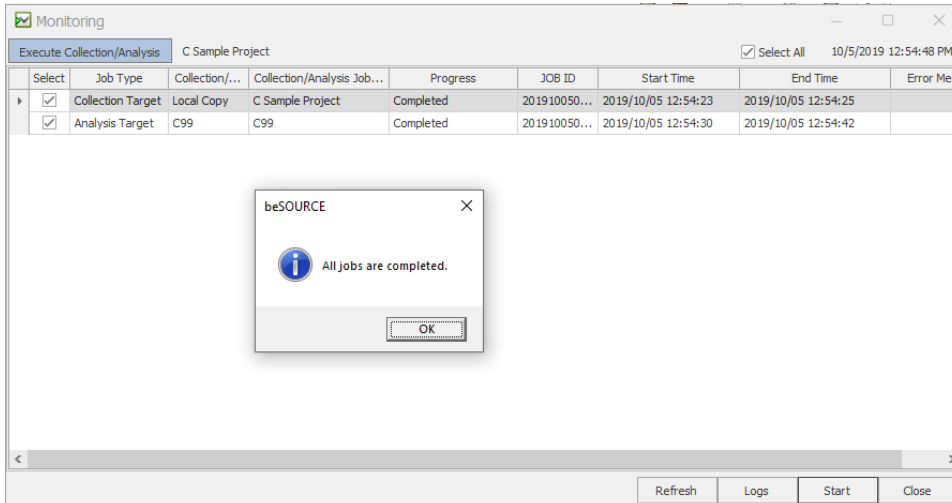
16. Select **Start**.



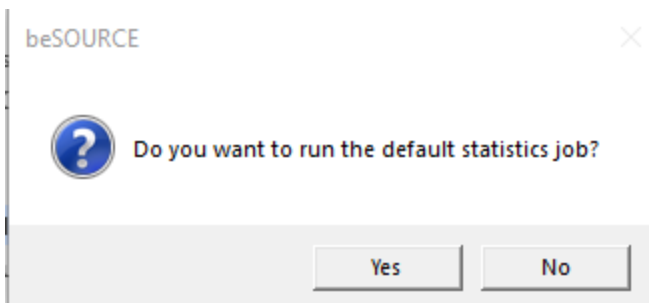
17. On the dialog that appears, select **Yes**.



18. The Monitoring window will display progress as the sources files are imported and analyzed. To view detailed job logs, select **Logs**. The Monitoring window will close and the View Job Log window will open. Select **View** to refresh the window. For more information on the View Job Log window, press F1 to open online help.
19. Once the jobs are complete, select **OK** on the dialog.



20. On the default statistics dialog that appears, select **Yes**.



21. On the next dialog, select **OK**.
22. Close the Monitoring window.


Viewing analysis results

Users can view analysis results on the beSOURCE Server using the beSOURCE Client.

To view the analysis results, do the following:

1. Open the **beSOURCE Admin Console**.
2. Select **User/Permission > Project Permission Setting**.
3. Select the **Group** tab, and then select **Developer Group** in the left panel.
4. Select **C Sample Application** in the right panel.

5. Select **Apply**.
6. Select **OK**.
7. Open the **beSOURCE Client**, and then log in as a developer-based user.
8. Select the **Project Explorer** to view the C Sample Project.

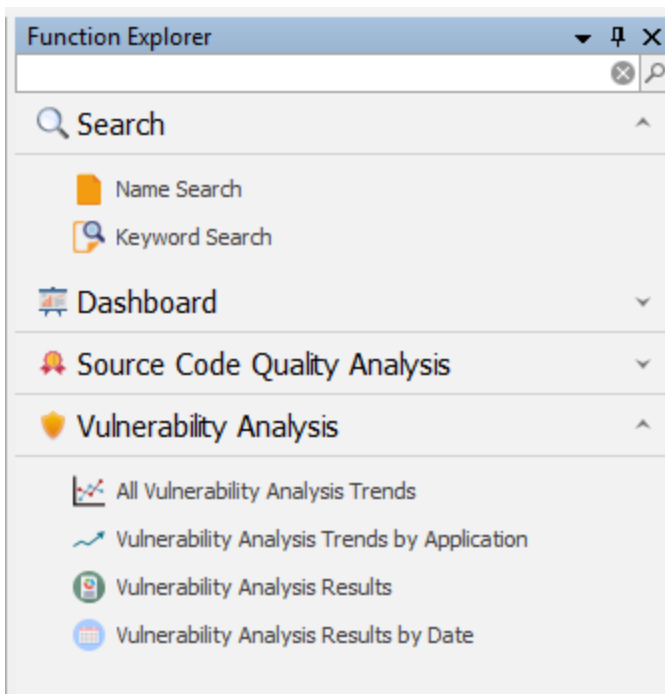
NOTE: Select the **Refresh** () icon if the C Sample Project does not appear. The default system administrator (besource) can view any projects without access permission.

Using the Function Explorer to view analysis results

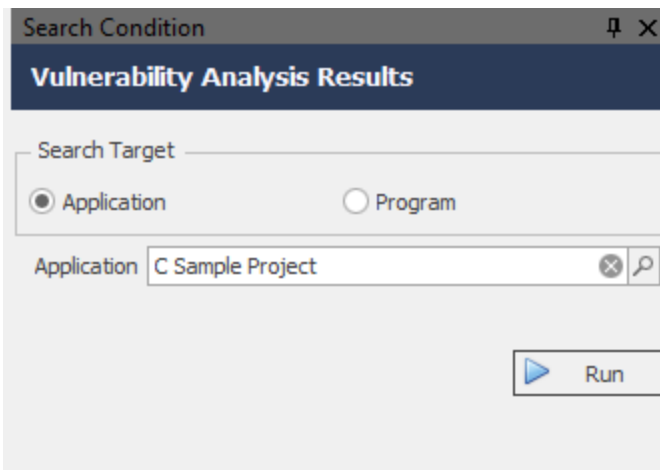
You can now use the Function Explorer to view the server-side analysis results.

To view server-side analysis results with the Function Explorer, do the following:

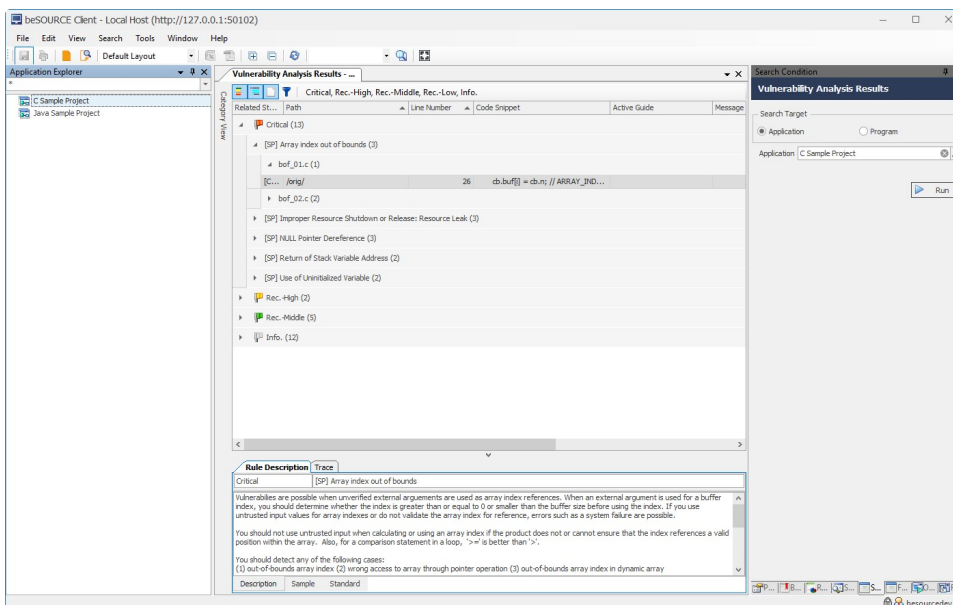
1. Select **View > Function Explorer**.
2. Select **Vulnerability Analysis > Vulnerability Analysis Results**.



3. In the **Search Condition** window, select the **C Sample Project**.
4. Select **Run**.



5. Any rule violations in the project will appear.




To generate reports, see [Generating Reports on page 42](#).

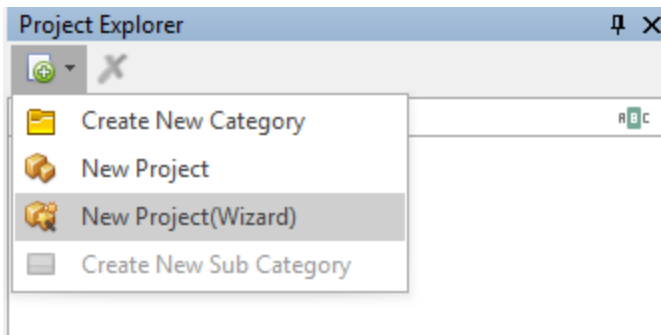
Analyzing C++ Source Files

This chapter describes how to analyze C++ sample source files on the server and view their results.

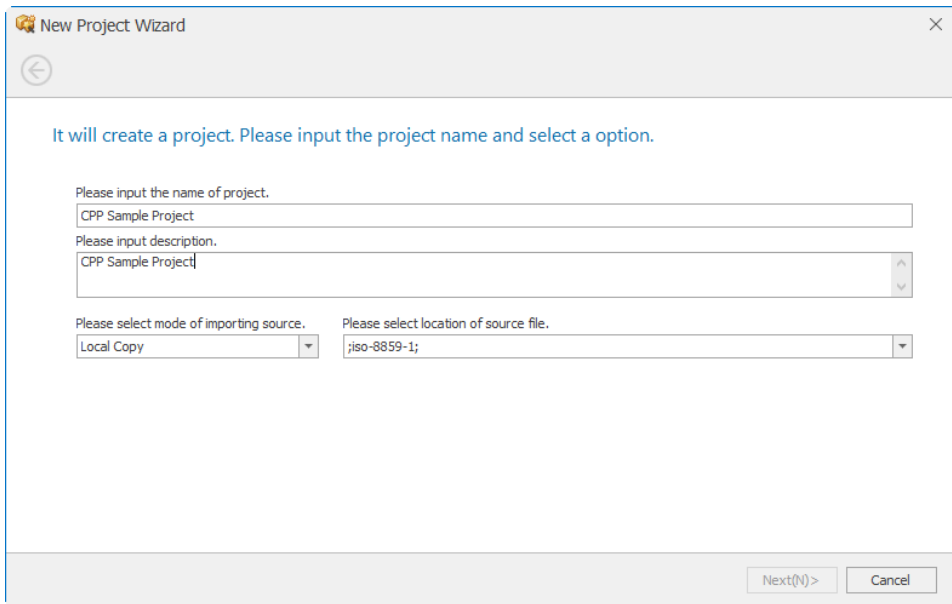
Analyzing C++ sample source files

To analyze C++ sample source files, do the following:

1. Open and log in to the **beSOURCE Admin Console**.
2. Select **View > Project Explorer**.
3. Select the **Add Project Items** () icon in the Project Explorer (without selecting any project).
4. Select **New Project(Wizard)**.



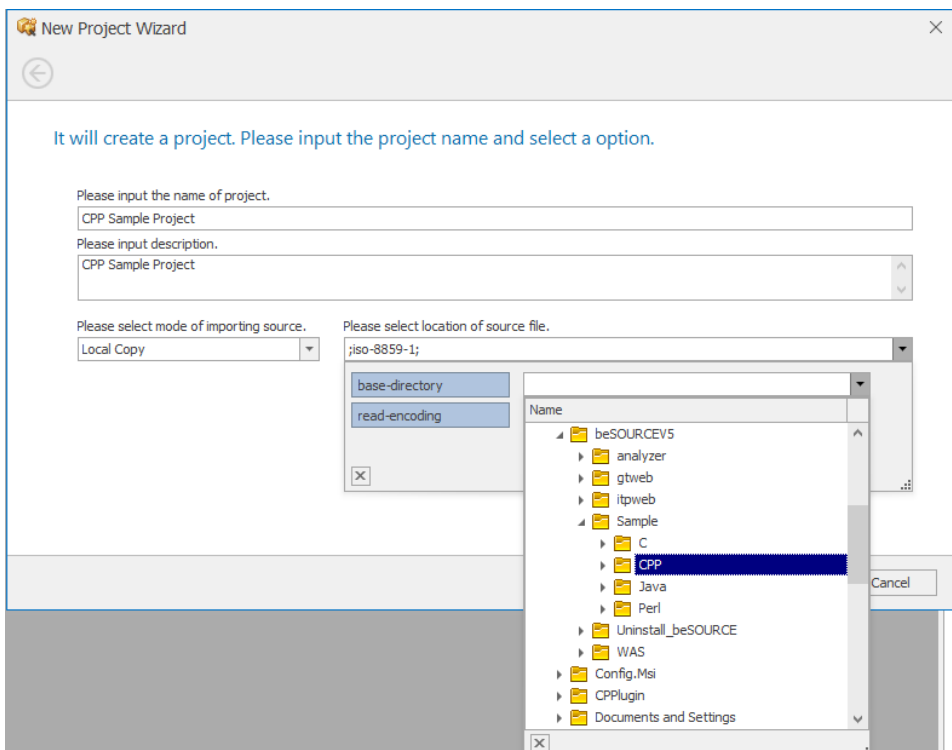
5. In the **Please input the name of the project** box, enter a name.
6. In the **Please input description** box, enter a description for the project.



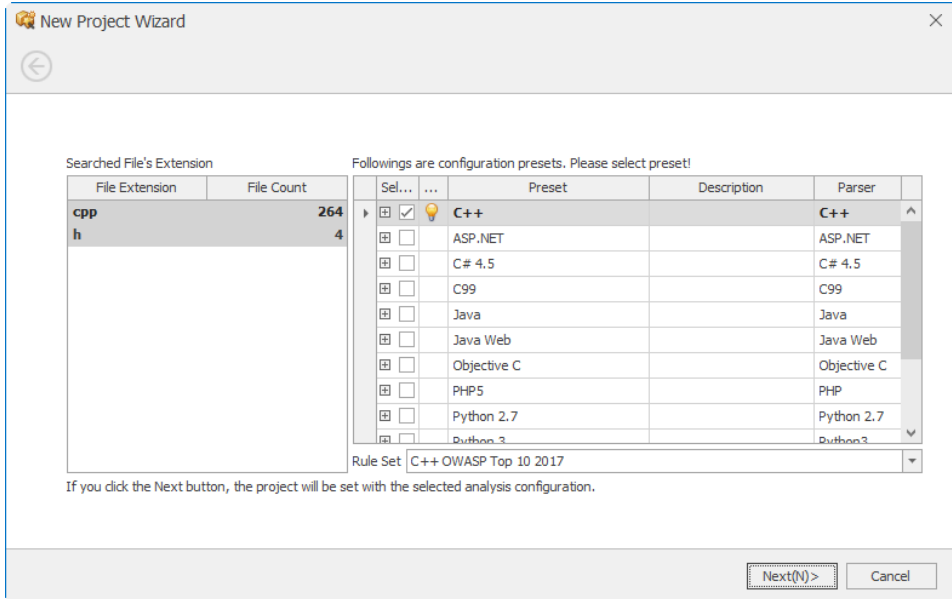
7. Select the **Please select location of source file** box, and then select the **base-directory** list box.

8. **NOTE:** The **Local Copy** option in the **Please select mode of importing source** box refers to the beSOURCE server's hard drive.

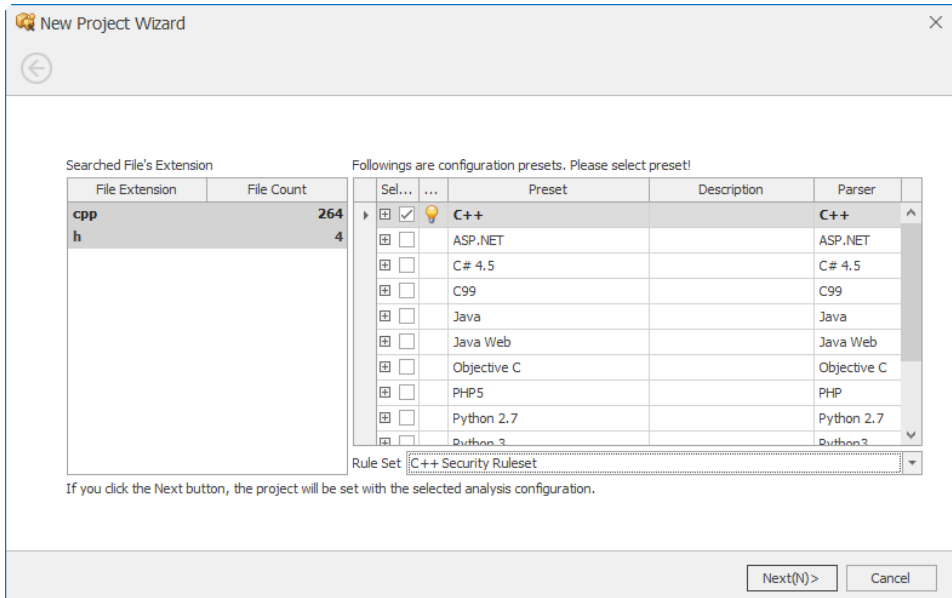
9. Select the **Sample\CPP** folder located within your beSOURCE Server installation directory (for example, `C : \beSOURCE\Sample\C`).




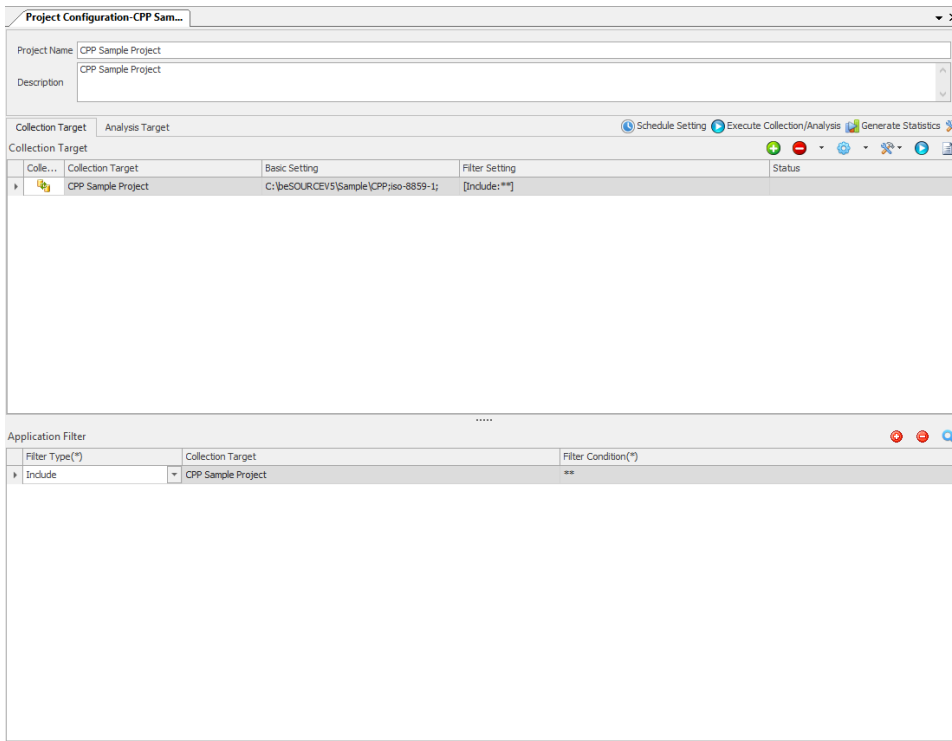
10. Select **Next**.
11. The Project Wizard will scan the source files and then recommend a preset of configuration. Select the **C++** preset.



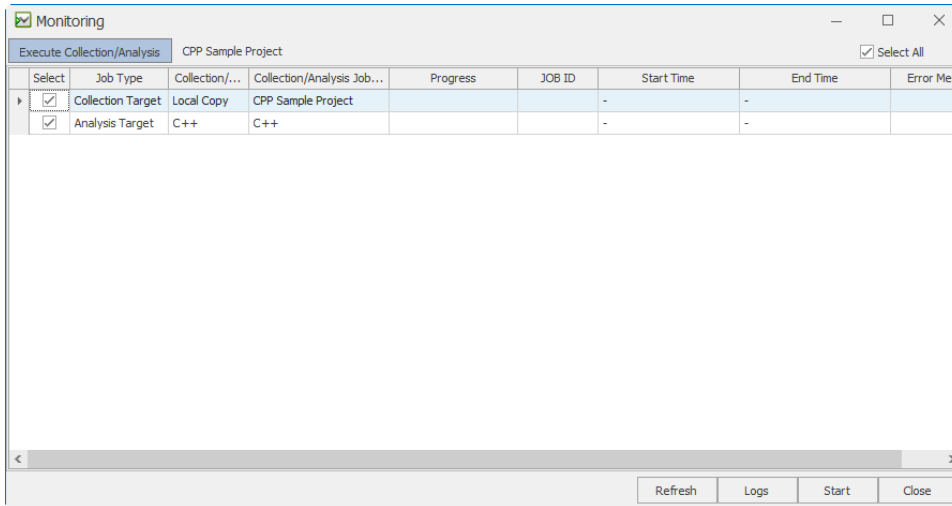
12. In the **Rule Set** box, select the desired rule set.



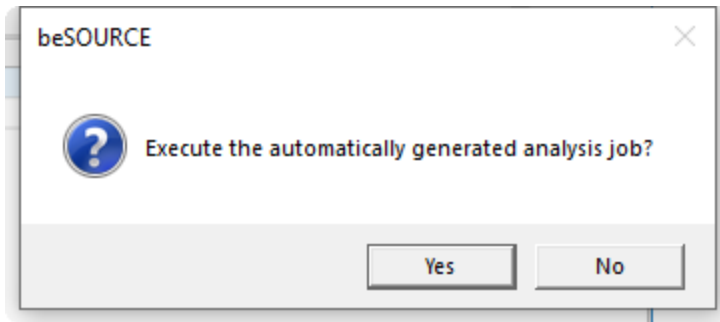
13. Select **Next**.
14. Select **Finish**. The Project Configuration window opens.
15. Select the **Execute Collection/Analysis** () icon. The Monitoring window opens.



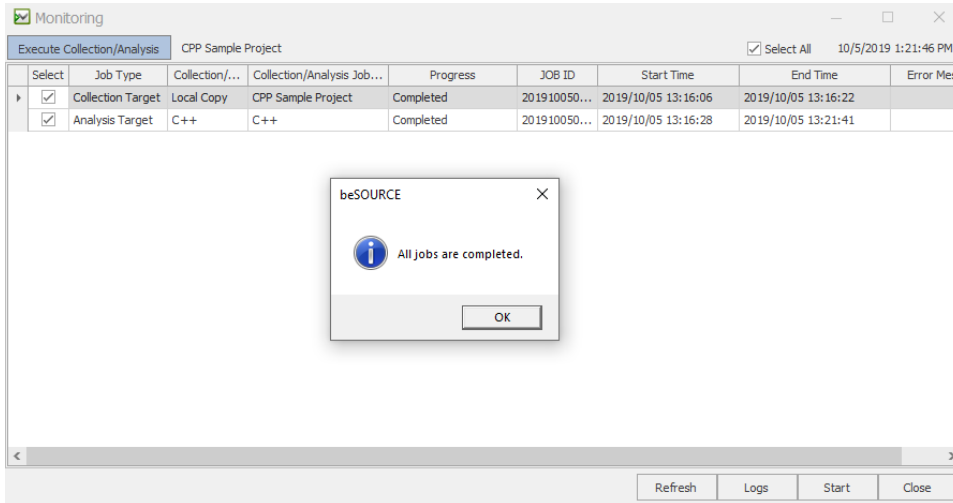
16. Select **Start**.



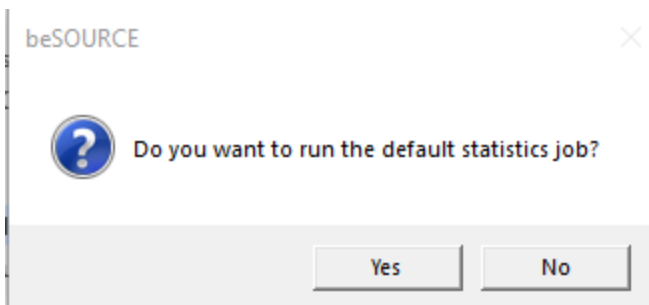
17. On the dialog that appears, select **Yes**.



18. The Monitoring window will display progress as the sources files are imported and analyzed. To view detailed job logs, select **Logs**. The Monitoring window will close and the View Job Log window will open. Select **View** to refresh the window. For more information on the View Job Log window, press F1 to open online help.
19. Once the jobs are complete, select **OK** on the dialog.



20. On the default statistics dialog that appears, select **Yes**.



21. On the next dialog, select **OK**.
22. Close the Monitoring window.


Viewing analysis results

Users can view analysis results on the beSOURCE Server using the beSOURCE Client.

To view the analysis results, do the following:

1. Open the **beSOURCE Admin Console**.
2. Select **User/Permission > Project Permission Setting**.
3. Select the **Group** tab, and then select **Developer Group** in the left panel.
4. Select **CPP Sample Application** in the right panel.

5. Select **Apply**.
6. Select **OK**.
7. Open the **beSOURCE Client**, and then log in as a developer-based user.
8. Select the **Project Explorer** to view the CPP Sample Project.

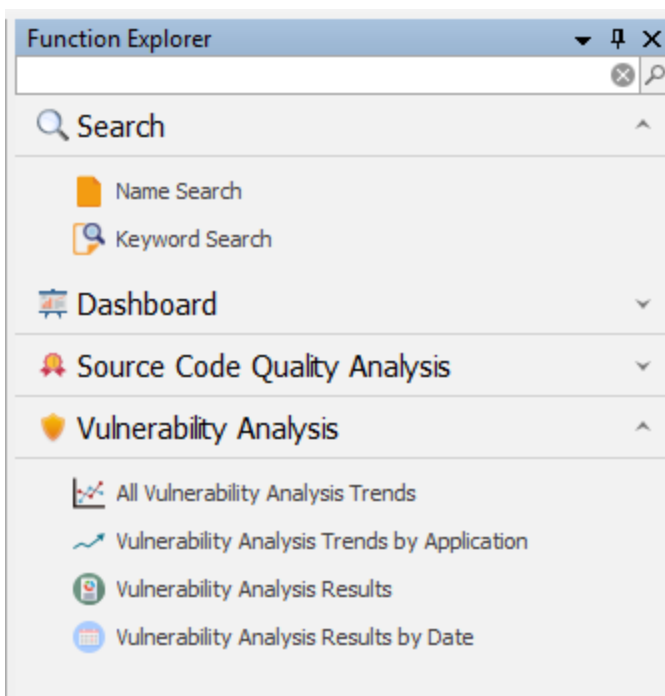
NOTE: Select the **Refresh** () icon if the C Sample Project does not appear. The default system administrator (besource) can view any projects without access permission.

Using the Function Explorer to view analysis results

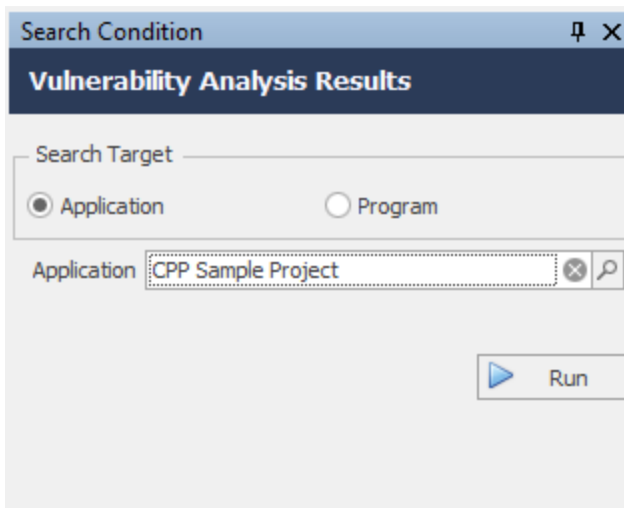
You can now use the Function Explorer to view the server-side analysis results.

To view server-side analysis results with the Function Explorer, do the following:

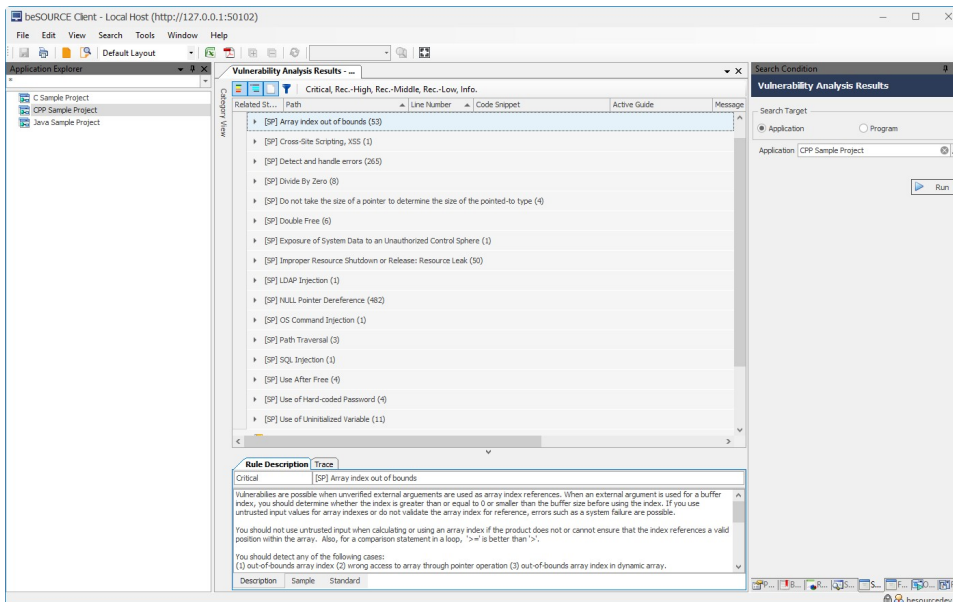
1. Select **View > Function Explorer**.
2. Select **Vulnerability Analysis > Vulnerability Analysis Results**.



3. In the **Search Condition** window, select the **C Sample Project**.
4. Select **Run**.



5. Any rule violations in the project will appear.



To generate reports, see [Generating Reports on page 42](#).

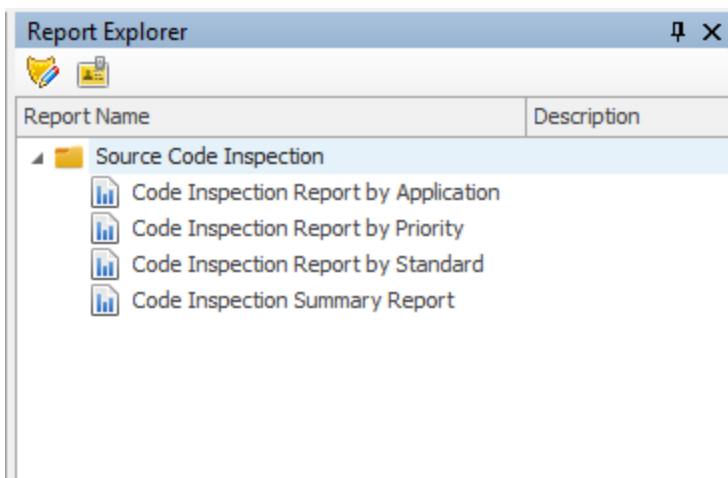
Generating Reports


The beSOURCE Client generates several types of reports. However, the administrator must first setup report authority for users or user group.

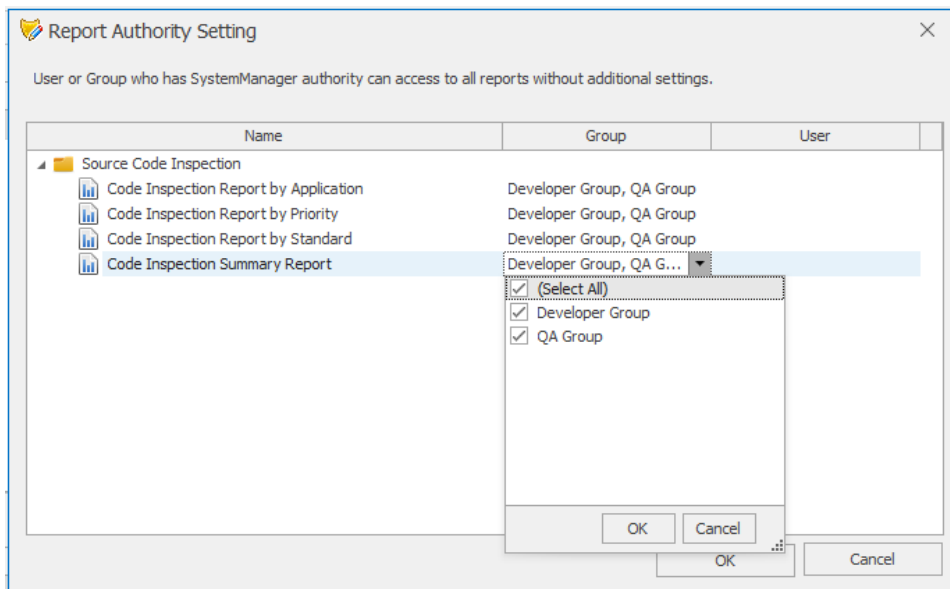
Setting up the Report Authority Setting

To set up the Report Authority Setting, do the following:

1. Open the **beSOURCE Admin Console**.
2. Select **View > Report Explorer**. The Report Explorer will appear.




3. In the **Report Explorer** window, select the **Report Authority Setting** () icon.
4. In the **Report Authority Setting** window, set a group or user for each report.

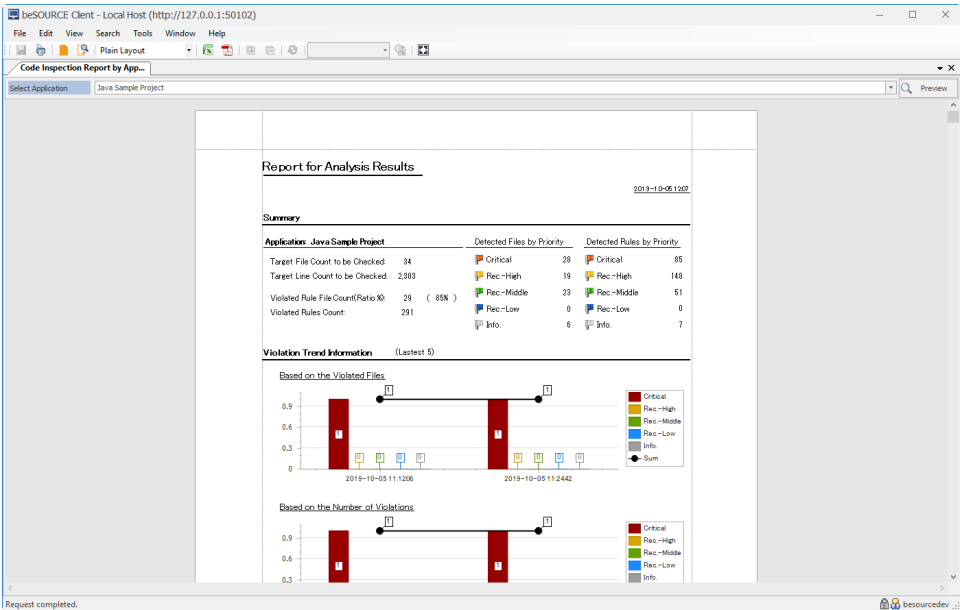


5. Select **OK**.

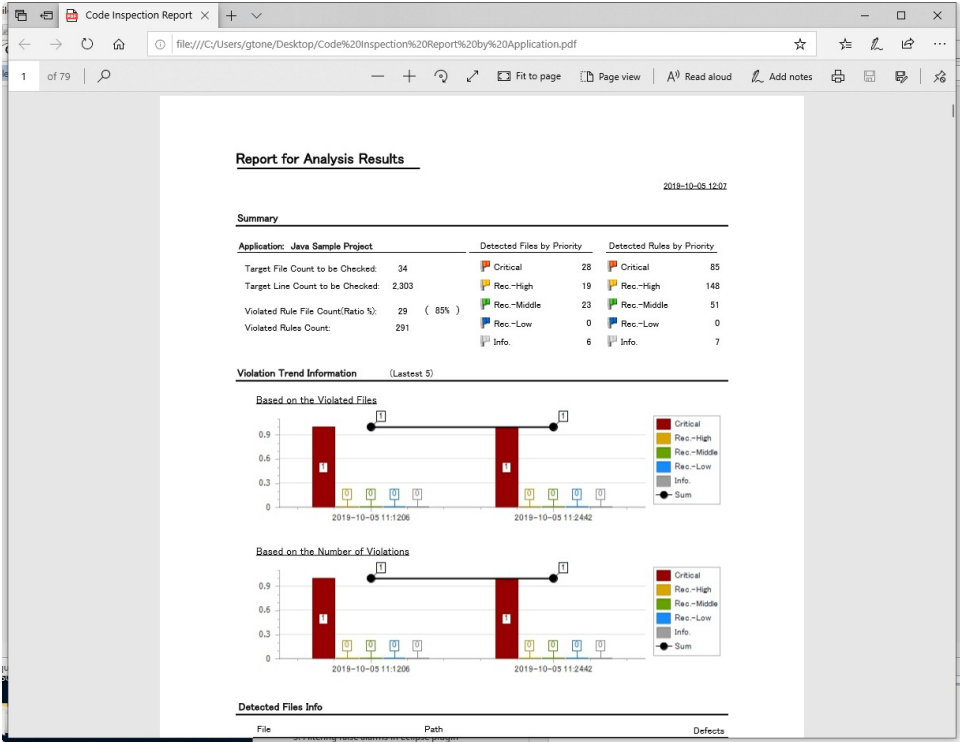
Generating a report

To generate a report, do the following:

1. Open the **beSOURCE Client**.
2. Log in as a developer.
3. Select **View > Report Explorer**.
4. Double-click a report (for example, Code Inspection Report by Application).
5. On the toolbar, select the **View Full Screen** () icon. Select the icon again to restore the UI layout.
6. Select a project or application name.
7. Select **Preview**. The report will generate.



8. Select **File > Export > PDF**.
9. Enter a file name.
10. Select **Save**. Open the newly created PDF to view the report.



Analyzing Source Files in beSOURCE Developer

This chapter describes how to analyze sample source files on the developer computer using beSOURCE Developer.

You can analyze source files using either of the following two modes in beSOURCE Developer:

- Local - Uses a local rule set to scan source files on the local computer.
- Server - Uses server-side setting information such as rule sets and libraries by synchronizing with the beSOURCE Server.

NOTE: It is recommended to copy sample source files from the beSOURCE Server to your local computer. The sample file directory on the server is [beSOURCE_Installation_Directory] \Sample.

Running beSOURCE Developer

After installing beSTORM Developer, you can open it by using the desktop shortcut icon. For information on installing beSOURCE Developer, refer to the beSOURCE Developer Edition Installation Guide.

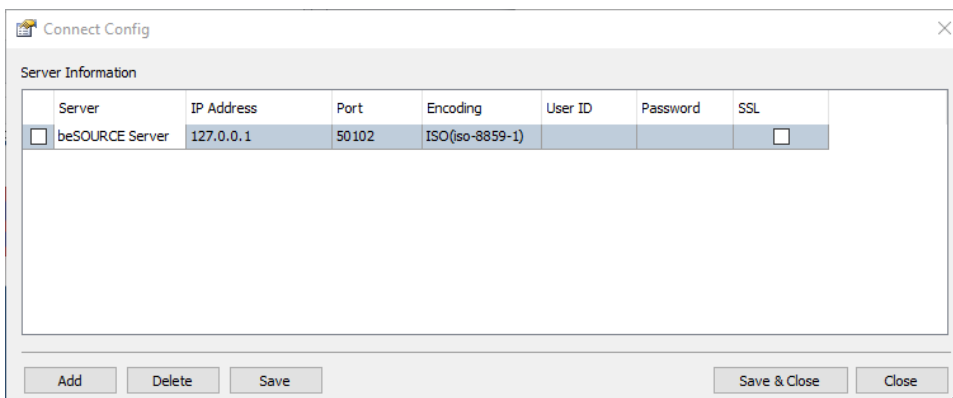


To open and log in to beSOURCE Developer, do the following:

1. Double-click beSOURCE Developer desktop shortcut icon.
2. On the log in window, select **Config**.



3. Select **Add**.
4. In the **Server**, **IP Address**, and **Port** boxes, enter the beSOURCE View Server's information.
5. Select **Save** or **Save & Close**.

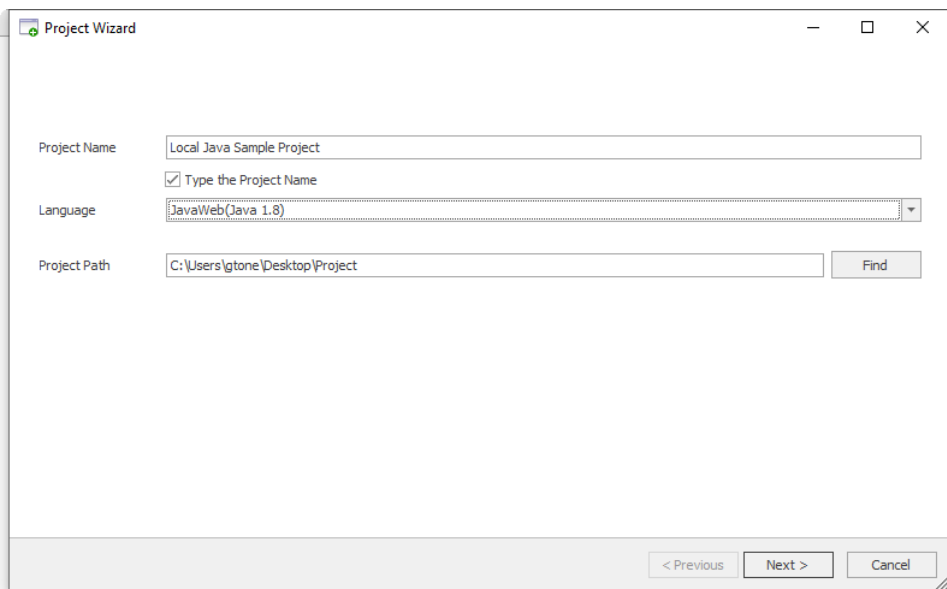


6. Back on the log in window, do the following:
 - a. In the **Server** box, select the server you created in step 5.
 - b. In the **User ID** box, enter a developer's user ID. For example, besourcedev that you created in the previous chapter.
 - c. In the **Password** box, enter the user's password., and enter a developer's **User ID** and **Password**.
7. Select **Log In**. You will see your connected server information in the title bar of beSOURCE Developer.

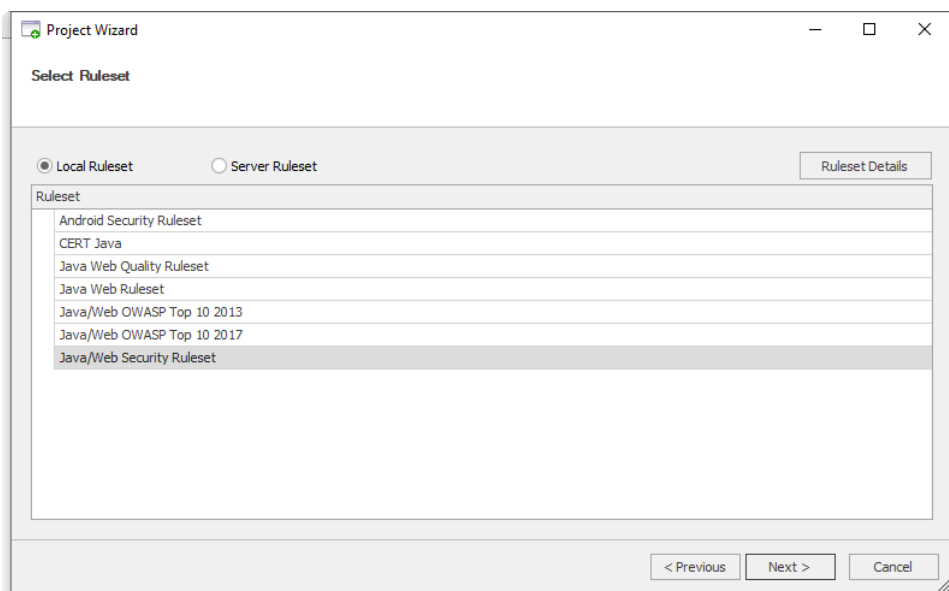
Local mode analysis in beSOURCE Developer

To analyze Java sample source files in beSOURCE Developer, do the following:

1. Select **File > New Project**, or select **New Project** on the Start Page. The Project Wizard opens.
2. On the first page of the wizard, do the following:
 - a. Select **Type the Project Name**.
 - b. In the **Project Name** box, enter a name.
 - c. In the **Language** box, select **JavaWeb(Java 1.8)**.

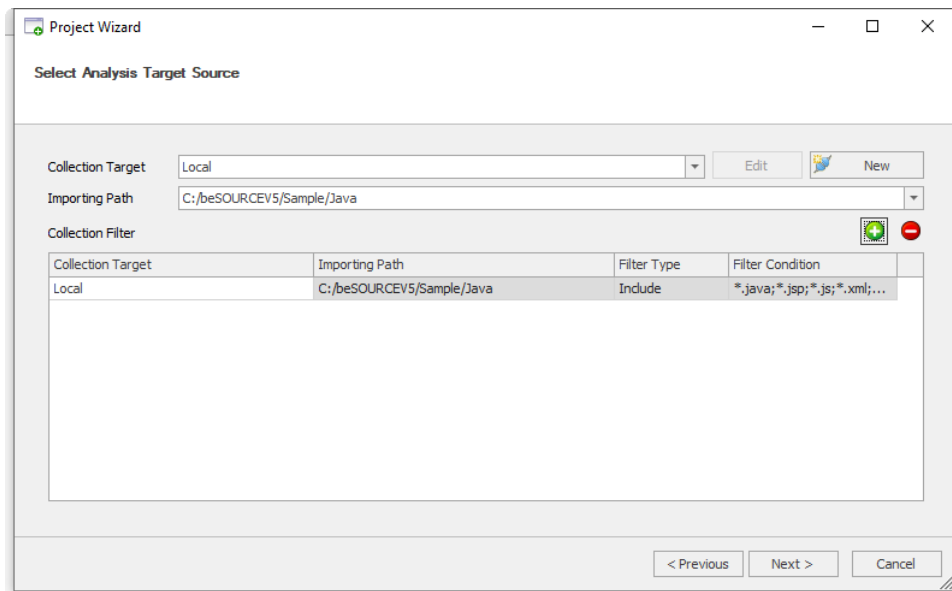


3. Select **Next**.
4. On the **Select Ruleset** page, select **Local Ruleset**, and then select a rule set.



5. Select **Next**.

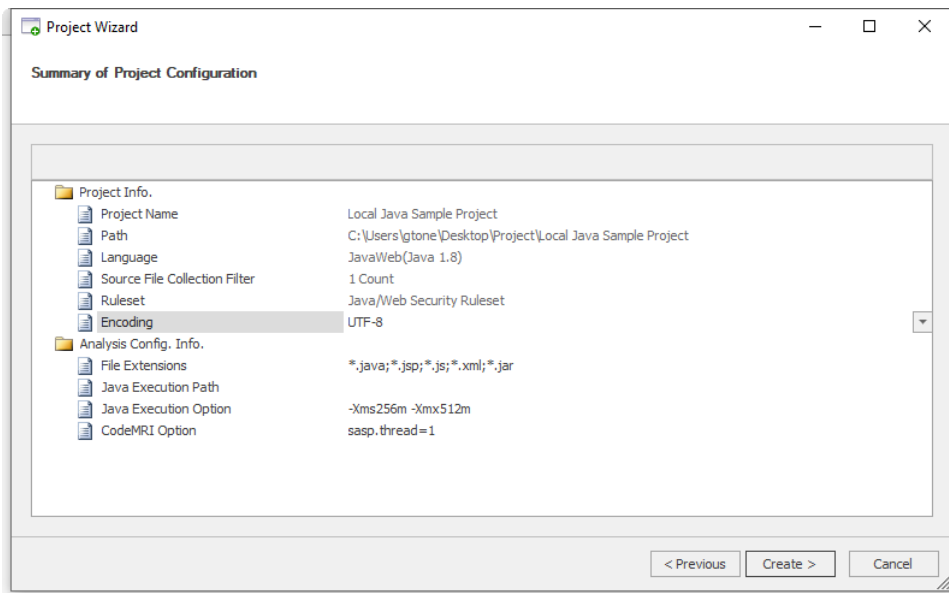
6. On the **Select Analysis Target Source** page, do the following:
 1. In the **Importing Path** box, enter or select the file path to your sample source files.
 2. Select **Add (+)**.



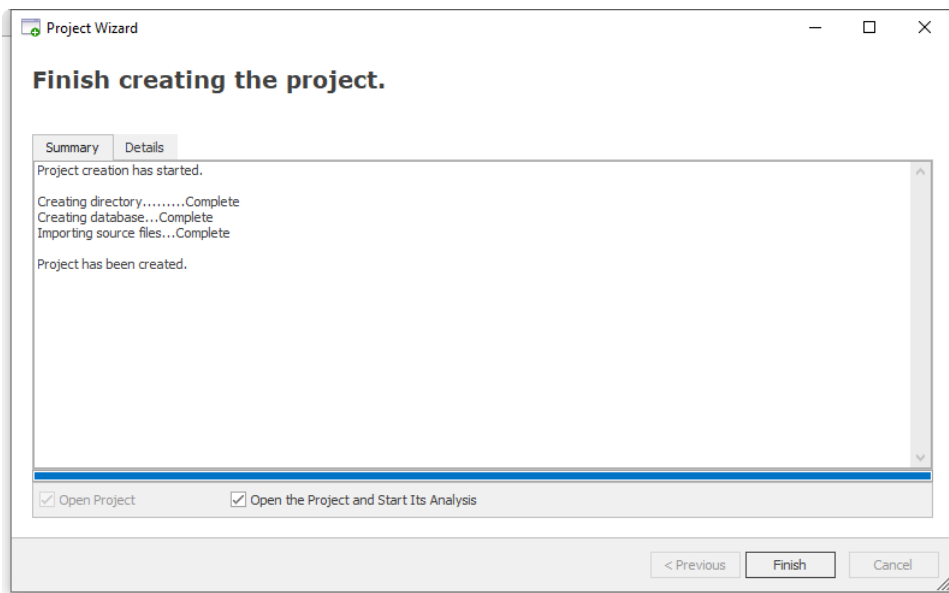
7. Select **Next**.
8. On the **Summary of Project Configuration** page, review the summary of your project, and then select **Create**.

NOTE:

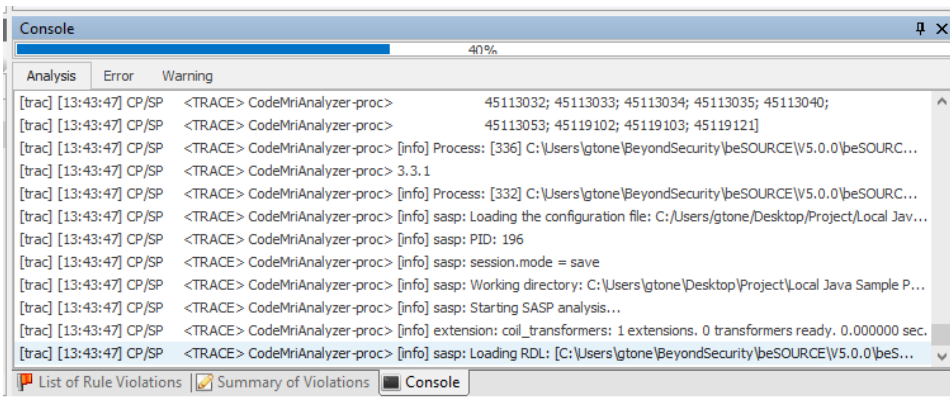
- If your computer has enough RAM, you can adjust the max heap memory size like below: Java Execution Option: -Xms256m -Xmx2048m.
- If you want to multi-threaded analysis, you can adjust the thread count like below: CodeMRI Option: sasp.thread=3 The max thread count must be your physical CPU core - 1.



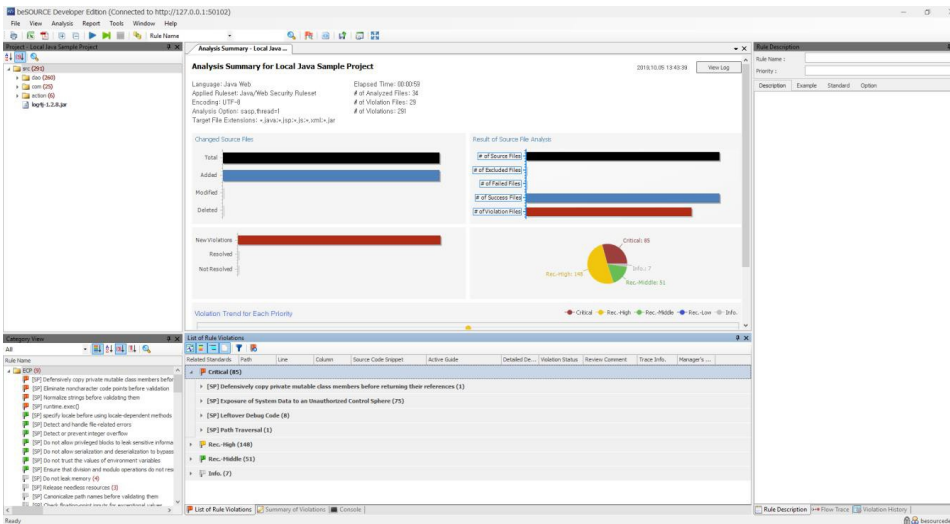
9. On the **Finish creating the project page**, select **Open the Project and Start Its Analysis**, and then select **Finish**.



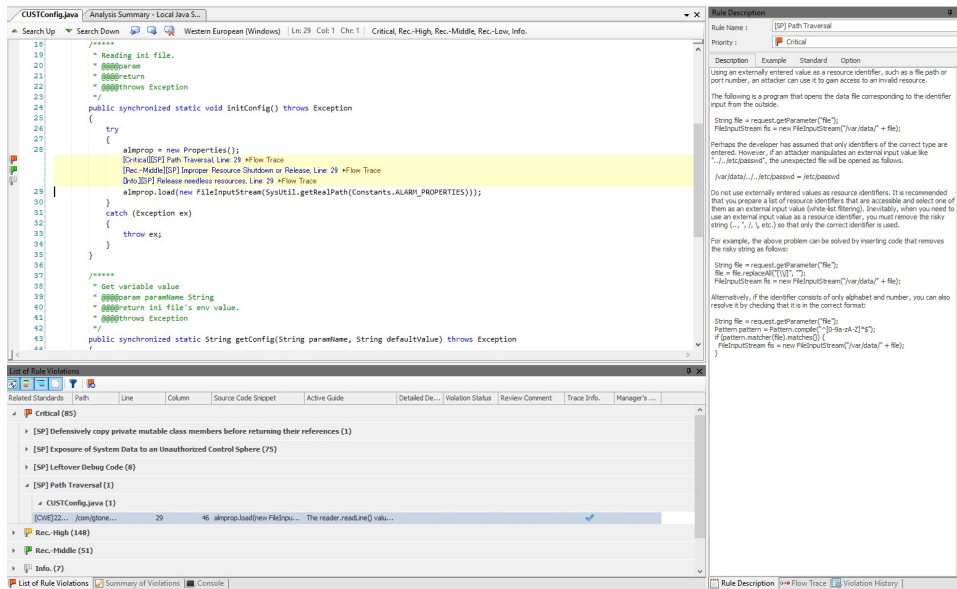
10. The Console window will show the analysis' progress.



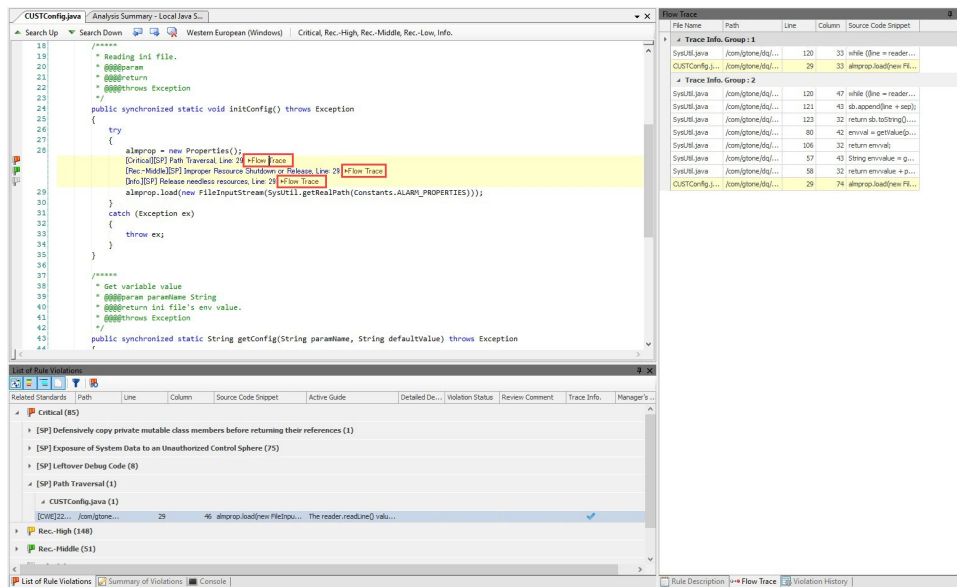
11. On the dialog that appears, select **OK** to open the project.
12. The **List of Rule Violations** tab shows rule violations.



13. Double-click a rule violation to open the Source Viewer. You can refer to description of the rule, code examples, and related secure coding standards in the right panel.



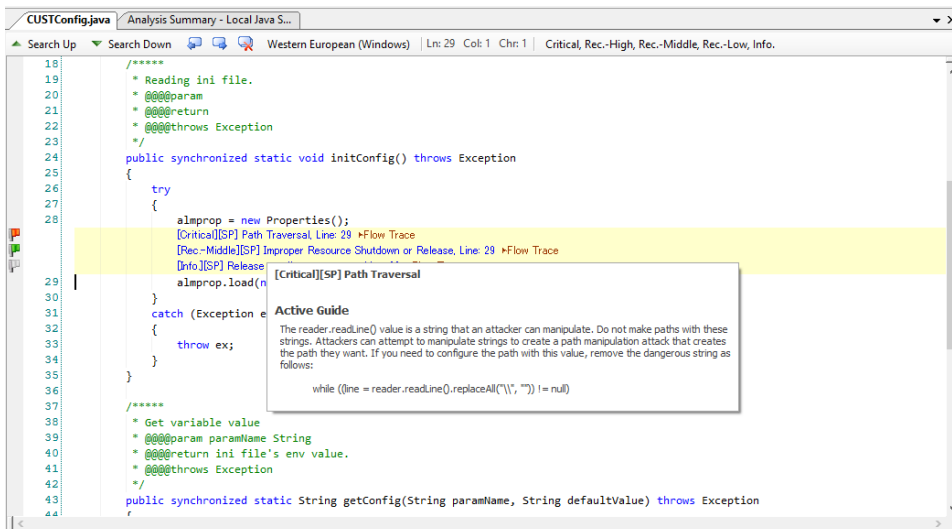
14. Select a **Flow Trace** link in Source Viewer to view the method call chain which provides a trace back to the first method call of the problem.



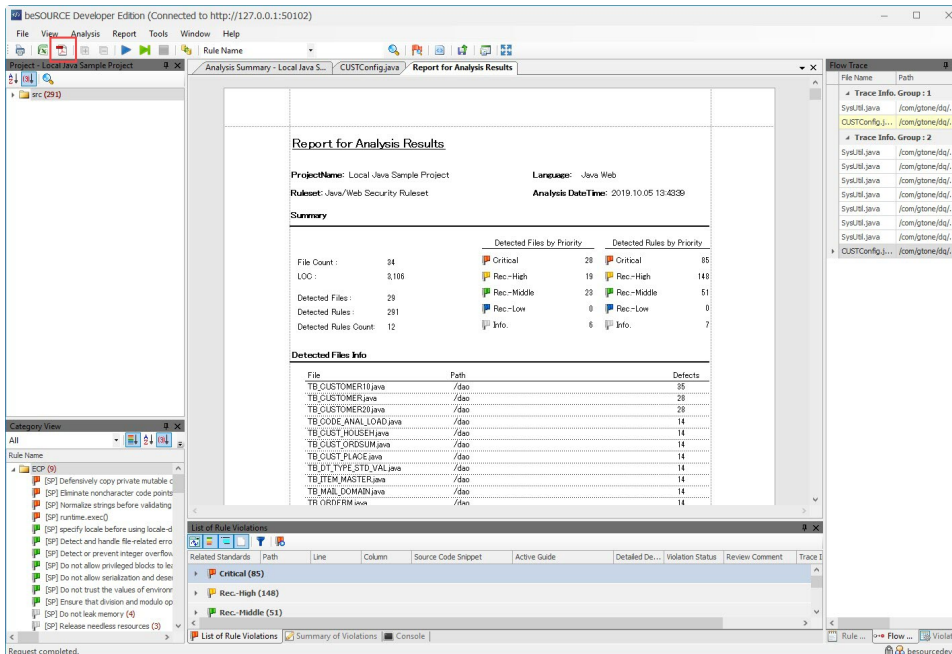
15. Move your pointer over the link of the **Path Traversal** rule violation to display the active guide (a correction guide based on actual source code).

NOTE:

- The detected rule violation is located at the bottom of the active guide window.
- The **Flow Trace** link is not provided for every rule violation.



16. Select **Report > Report for Analysis Results** to generate a report. To export the report to a PDF, select **File > Export > PDF** or select the PDF icon. Select Help to read more about the beSOURCE Developer UI.



Server mode analysis in beSOURCE Developer


You can analyze the same source files on the beSOURCE Server and in beSOURCE Developer.

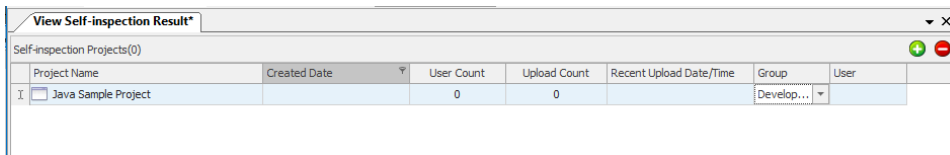
If you use the Server mode analysis in beSOURCE Developer, the beSOURCE Server's analysis setting information (for example, rule sets and libraries) are automatically synchronized with beSOURCE Developer.

NOTE: The Server mode will not synchronize source files. beSOURCE Developer analyzes the local computer's source files with the server-side rule set configuration and libraries required to analyze them.

Server preparation

To use the Server mode analysis, the administrator must do the following:

1. Open and log in to the **Admin Console**.
2. Select **Tools > View Self-inspection Results**.
3. Select the **New Project** () icon.
4. In the **Project Name** cell, enter a name (you can use the same project name currently in Project Explorer).
5. In the **Group** box, assign a group.



Project Name	Created Date	User Count	Upload Count	Recent Upload Date/Time	Group	User
Java Sample Project		0	0		Develop...	

6. Select **File > Save**.

NOTE:

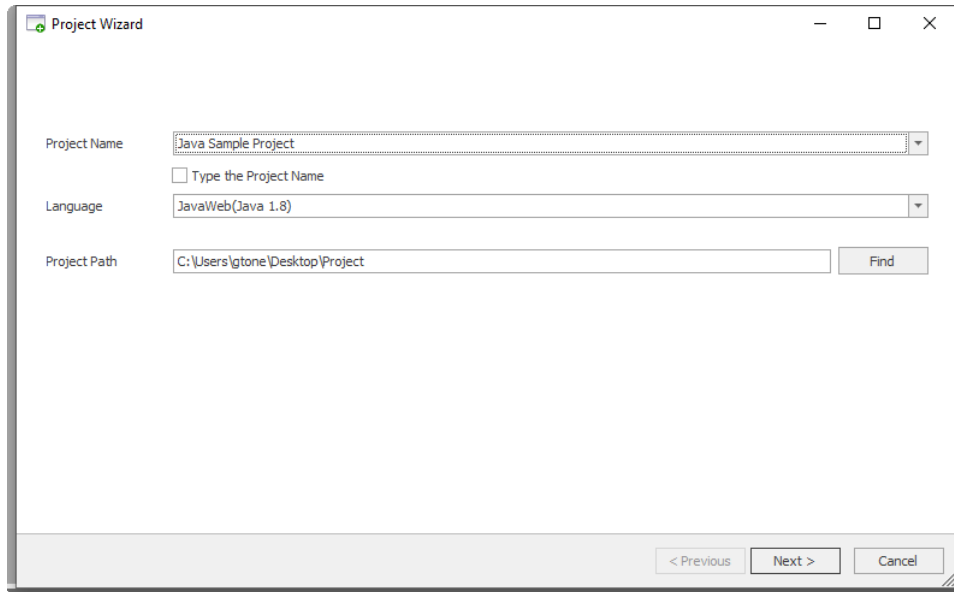
- The administrator must finish setting up the project analysis settings before allowing developers to initiate the server mode analysis.
- These steps assume that you have already copied sample source files from beSOURCE Server to your local PC. The sample file directory on the server is `[beSOURCE_Installation_Directory] \Sample` and this document assumes that you have copied them to the `C:\beSOURCE\Sample` directory.

Server mode analysis

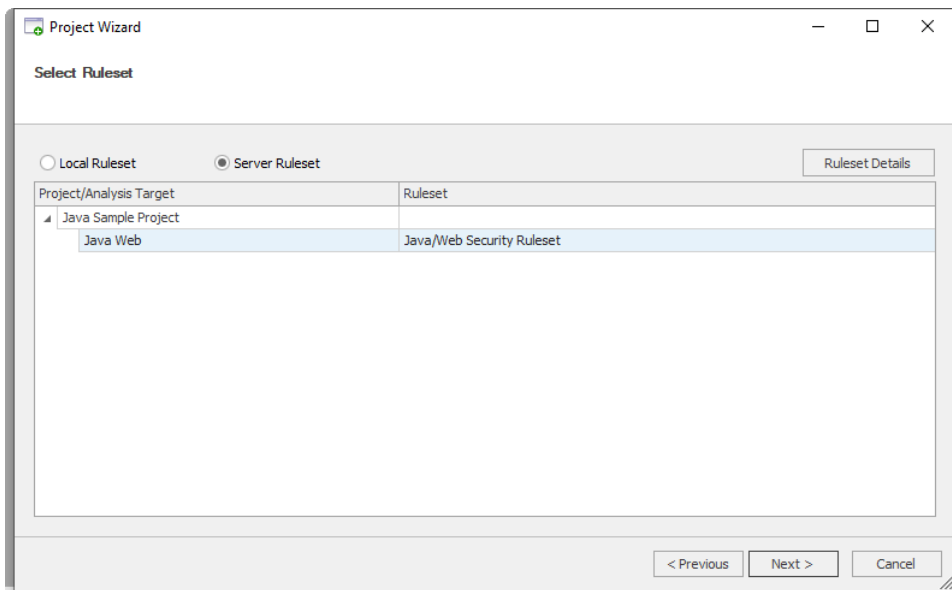
To analyze sample source files with beSOURCE Server synchronization, do the following:

1. Open and log in to **beSOURCE Developer**.
2. **File > New Project**, or select **New Project** on the **Start Page**. The Project Wizard opens.
3. On the first page of the wizard, do the following:

- a. Clear the **Type the Project Name** checkbox.
- b. In the **Project Name** box, select **Java Sample Project** (previously added by the administrator).
- c. In the **Language** box, select **JavaWeb(Java 1.8)**.

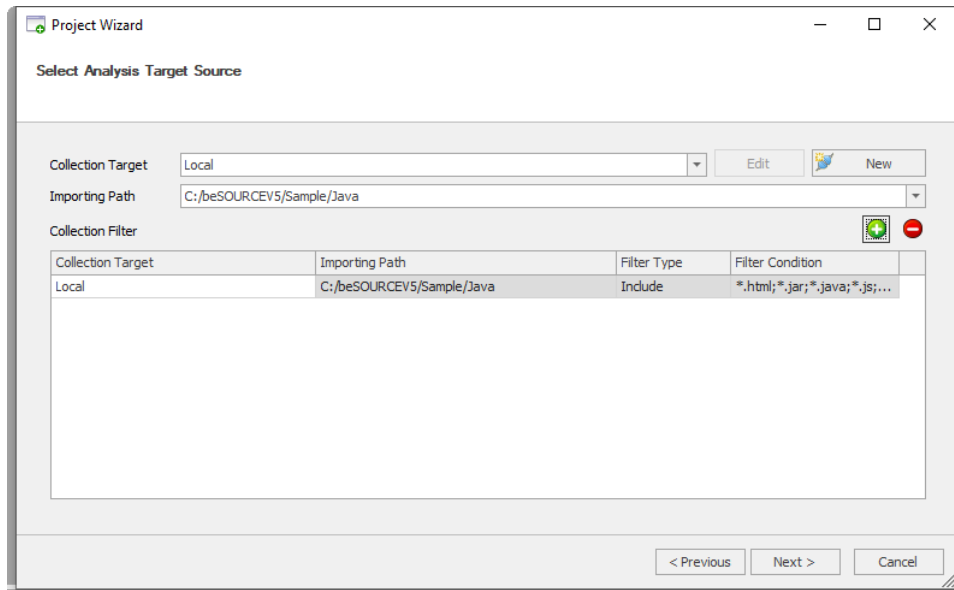


3. Select **Next**.
4. On the **Select Ruleset** page, select **Server Ruleset**.
5. Expand **Java Sample Project**, and then select **Java/Web Security Ruleset**.



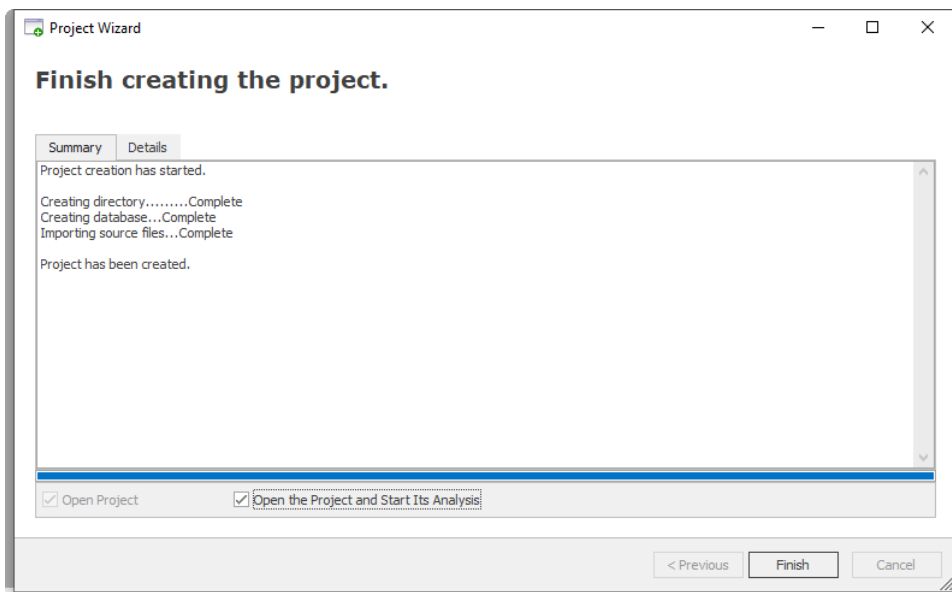
6. Select **Next**.
7. On the **Select Analysis Target Source** page, do the following:

- a. In the **Importing Path** box, enter or select the file path to the sample source files.
- b. Select the **Add (+)** icon.



8. Select **Next**.
9. On the **Summary of Project Configuration** page, review the summary of your project, and then select **Create**.
10. On the **Finish creating the project** page, select **Finish**.


NOTE: If you do not select **Open the Project and Start Its Analysis**, you can select the **Start Analysis** () icon to start analyzing the project.

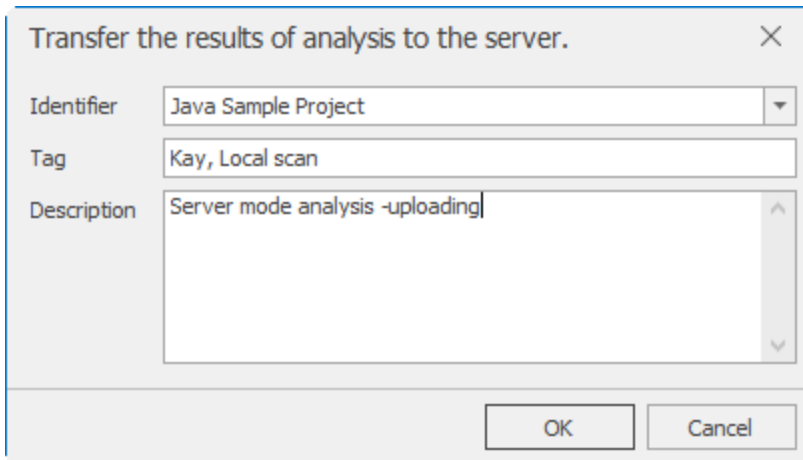


11. The Console window will show the analysis' progress.
12. On the dialog that appears, select **OK** to open the project.
13. The **List of Rule Violations** tab shows rule violations.

Uploading and sharing a developer's analysis results

A developer can upload the analysis results in beSOURCE Developer to beSOURCE Server in the sever mode analysis.

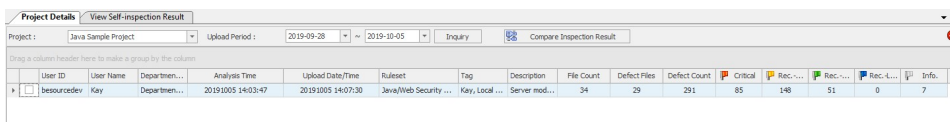
1. Open the project in **beSOURCE Developer**.
2. Select the **Transfer** () icon.
3. In the **Tag** box, enter one or more tags for the transfer.
4. In the **Description** box, enter a description for the transfer.



5. Select **OK**.

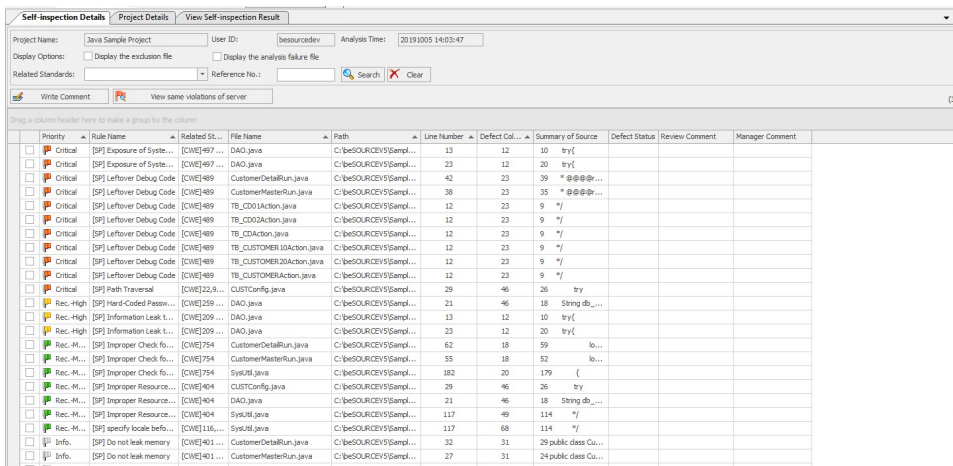
The administrator can then view the uploaded analysis results by doing the following:

1. Open and log in to the **Admin Console**.
2. Select **Tools > View Self-inspection Results**.
3. Double-click **Java Sample Project**. The Project Details opens, displaying a summary of the developer's inspection results.

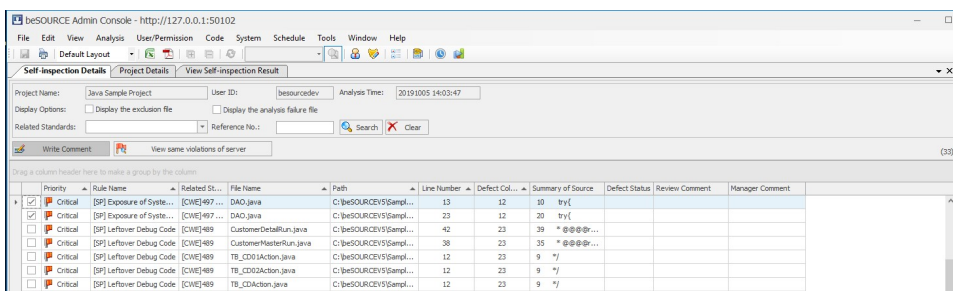


User ID	User Name	Department	Analysis Time	Upload Date/Time	Ruleset	Tag	Description	File Count	Defect Files	Defect Count	Critical	Rec...	Rec...	Rec...	Info
besourcdev	Kay	Department...	20191005 14:03:47	20191005 14:07:30	Java/Web Security ...	Kay, Local ...	Server mod...	34	29	251	65	148	51	0	7

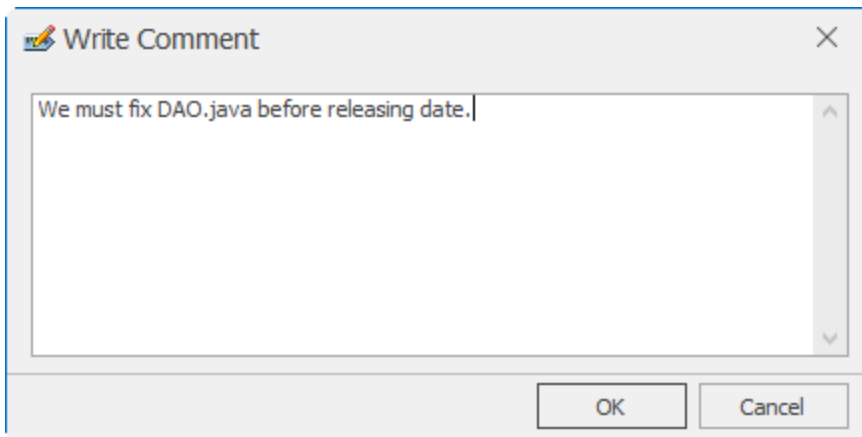
4. Double-click the uploaded item. The Self-inspection Details window appears, displaying details of the analysis' results.




5. Select some violations in the list, and then select **Write Comment**.



6. In the **Write Comment** dialog, enter a comment and then select **OK**.



The developer can then view the comments of administrator by doing the following:

1. Open **beSOURCE Developer**.
2. Select **File > Close Project**.
3. Select the **Start Page** () icon.
4. Select the **Java Sample Project** project to open it again.
5. Select the **List of Violations** tab to view the administrator's comments.

NOTE: The Uploading Analysis Results feature is also available for Local mode analysis.

Related Standards	Path	Line	Column	Source Code Snippet	Active Guide	Detailed De...	Violation Status	Review Comment	Trace Info.	Manager's Comment
Critical (85)										
[SP] Defensively copy private mutable class members before returning their references (1)										
[SP] Exposure of System Data to an Unauthorized Control Sphere (75)										
DAO.java (2)										
[CVE]#9...	/dao	13	12	e.printStackTrace();	Do not print the exception....		Fixed	We must fix DAO.java before releasing date.		
[CVE]#9...	/dao	23	12	e.printStackTrace();	Do not print the exception....		Fixed	We must fix DAO.java before releasing date.		
SQLQuery.java (1)										
TB_CD.java (2)										
TB_CD10.java (2)										
TB_CD20.java (2)										
TB_CODE_ANAL_LOAD.java (4)										
TB_CUST_HOUSEH.java (4)										

Analyzing Source Files in the Eclipse Plugin

This chapter describes how analyze sample source files on the developer computer using the beSOURCE Eclipse plugin.

You can analyze source files using either of the following two modes in the beSOURCE Eclipse plugin:

- Local - Uses a local rule set to scan source files on the local computer.
- Server - Uses server-side setting information such as rule sets and libraries by synchronizing with the beSOURCE Server.

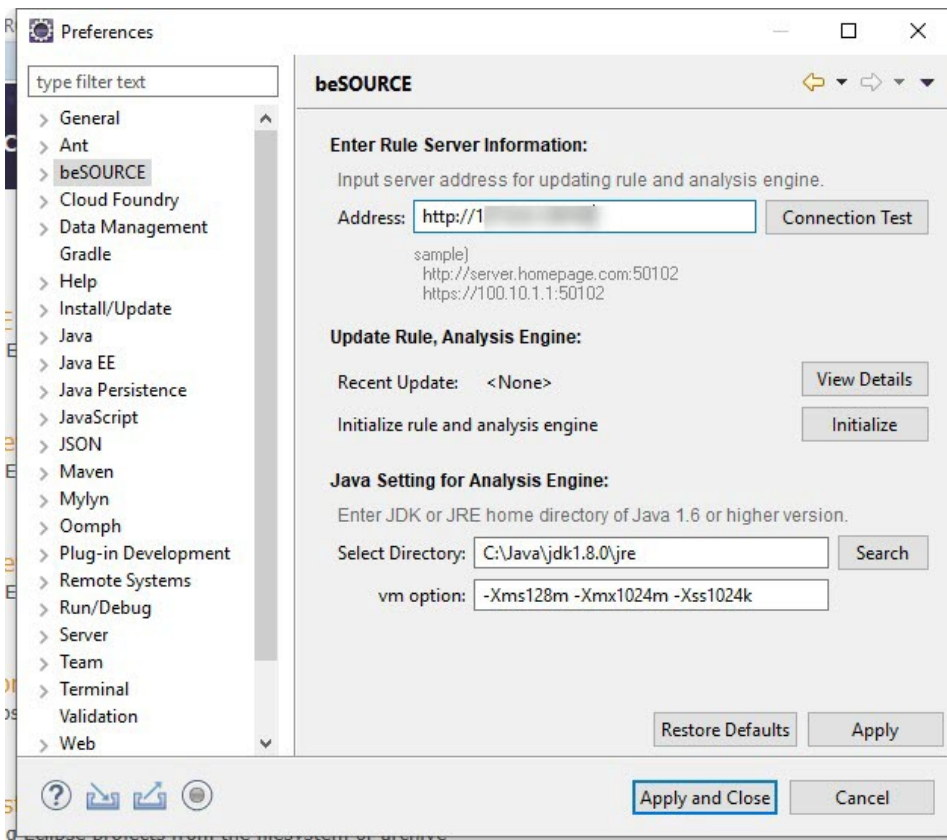
NOTE: It is recommended to copy sample source files from the beSOURCE Server to your local computer. The sample file directory on the server is [beSOURCE_Installation_Directory] \Sample.

Setting up the Eclipse plugin

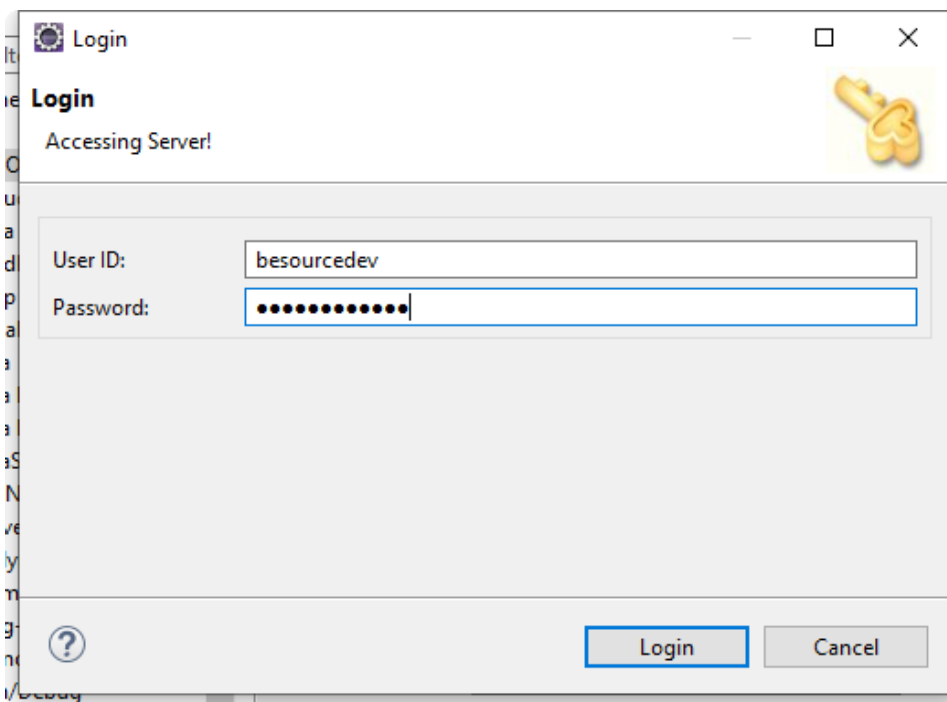
After installing the beSOURCE Eclipse plug, do the following to set up connection information:

NOTE: For more information on installing the beSOURCE Eclipse Plugin, refer to the beSOURCE Eclipse Plugin Installation Guide.

1. Select **Window > Preferences > beSOURCE**.
2. In the **Address** box, enter the beSOURCE Server's address.
3. Select **Connection Test**. The Login dialog opens.



4. In the **User ID** and **Password** boxes, enter **besourcedev**.
5. Select **Login**.



6. Select **OK** to close the connection test.

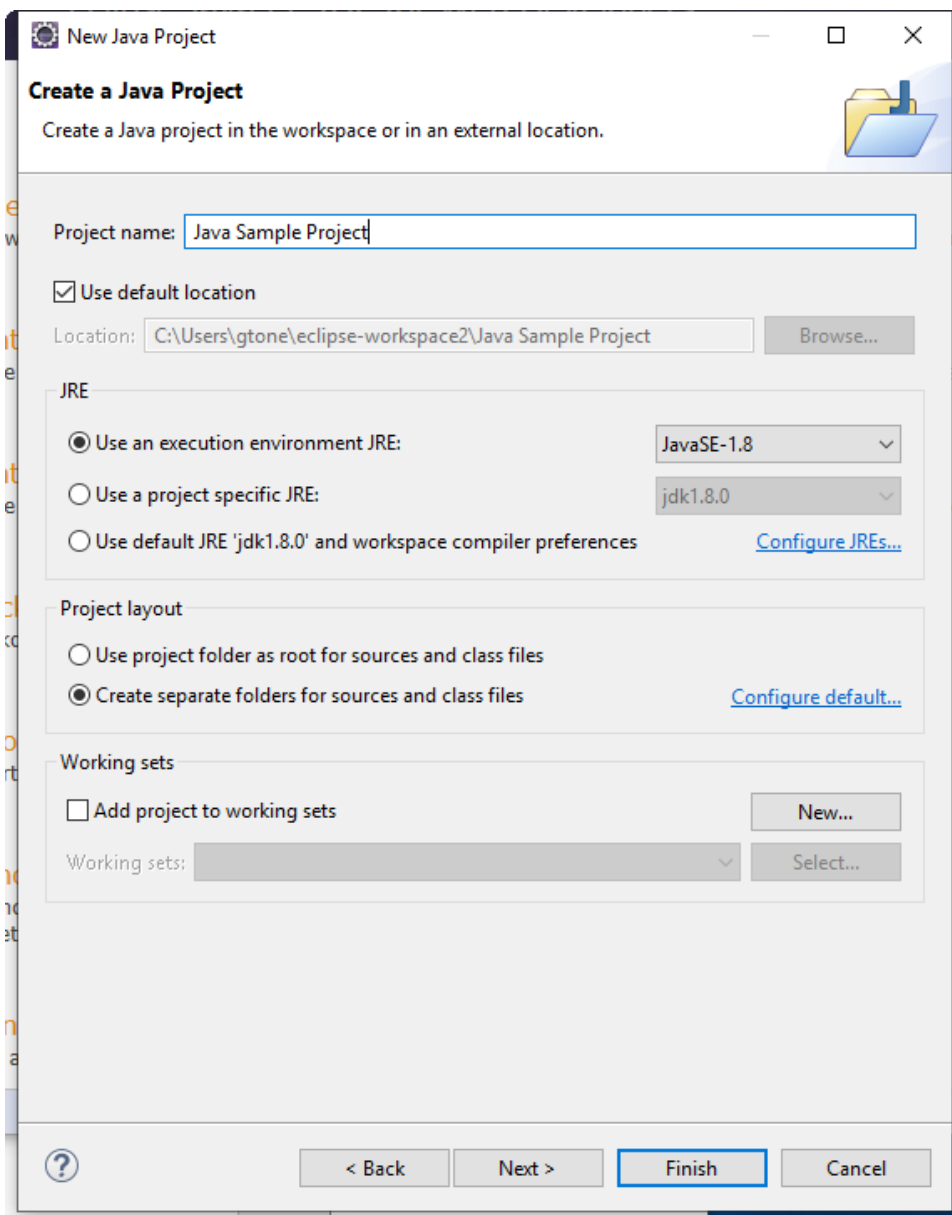
7. Select **Apply and Close** to close the Preferences window.

NOTE: License information is located on the server. When you log in, the license will be checked automatically. You do not need to apply a license for the plugin.

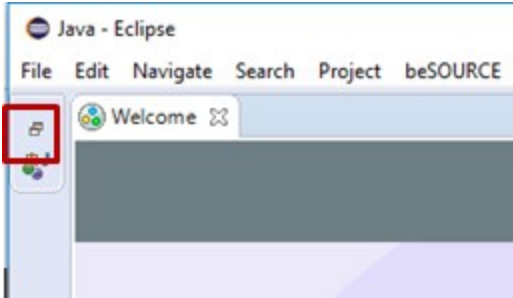
Creating a Java project

Now, you need to create a Java project for this hands-on exercise.

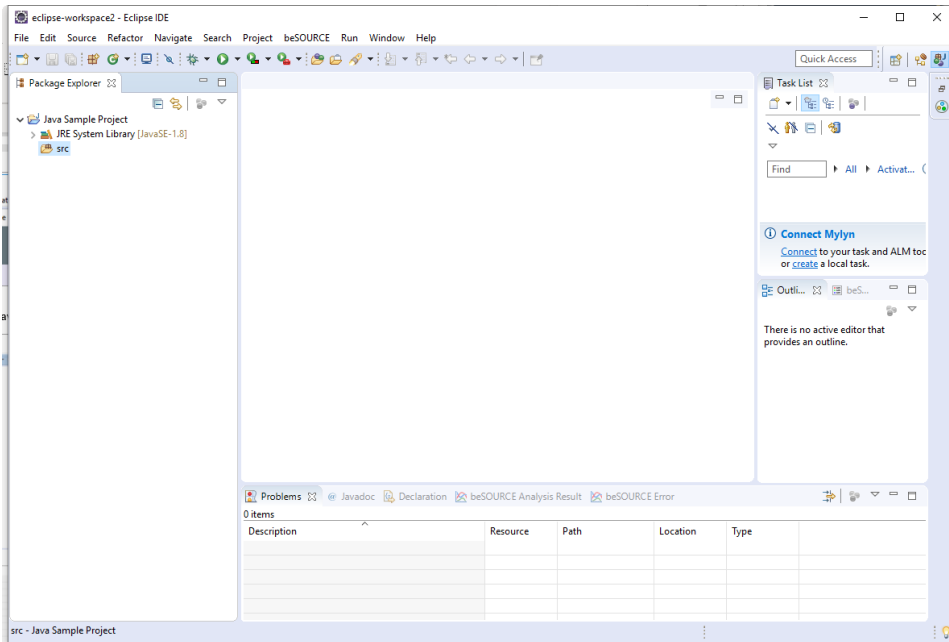
1. Select **File > New > Java Project**.
2. In the **Project name** box, enter a name, and then select **Finish**.



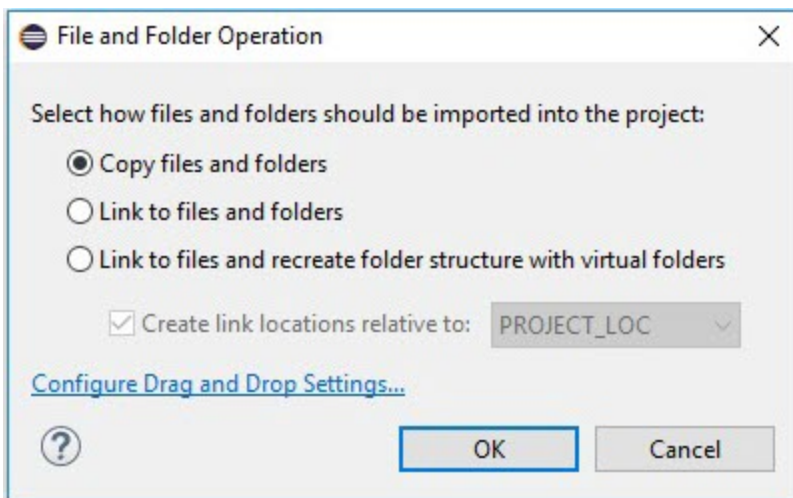
3. Select the **Perspective** icon.



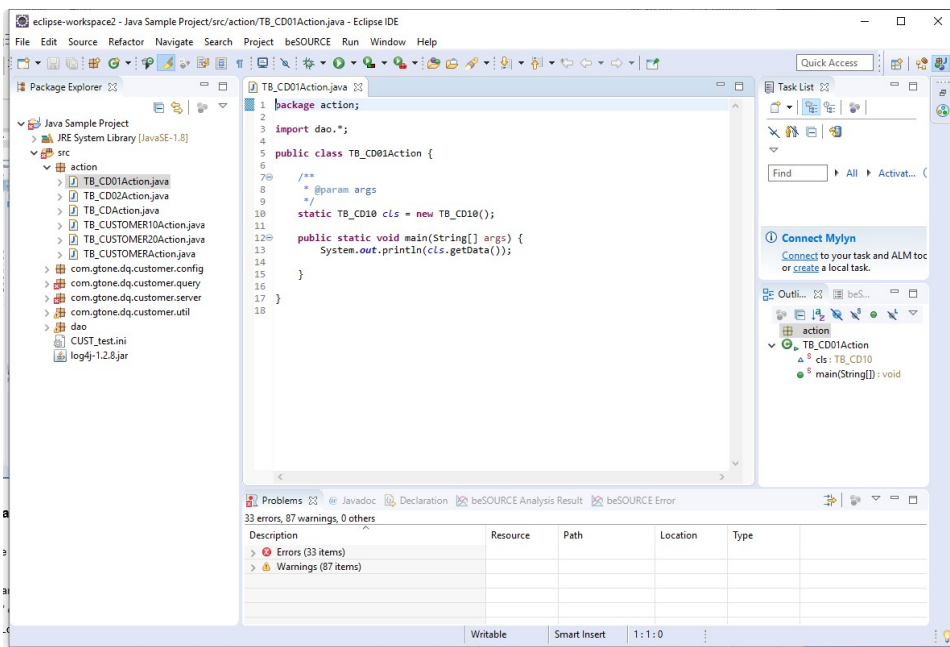
4. You may see Package Explorer with the new Java project.



5. In Windows, open the `c:\beSOURCE\Sample\Java` folder.
6. Select all folders and files within the Java folder, and then drag and drop them into **Java Sample Application > src**.
7. On the **File and Folder Operation** dialog, select **Copy files and folders**, and then select **OK**.



8. The sample Java source files are copied to Eclipse project.



Local mode analysis in the Eclipse plugin

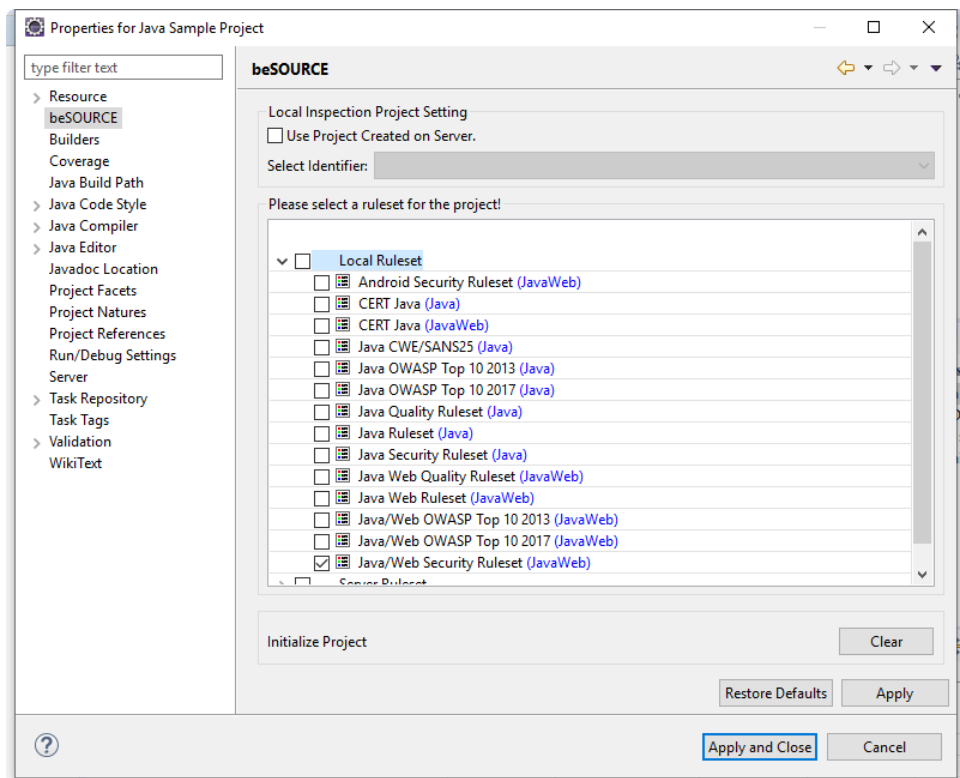
A developer can analyze Java source files with the local rule set inside the plugin by doing the following:

NOTE: When the Eclipse plugin is opened for the first time, it will download some modules, and then require you to restart it.

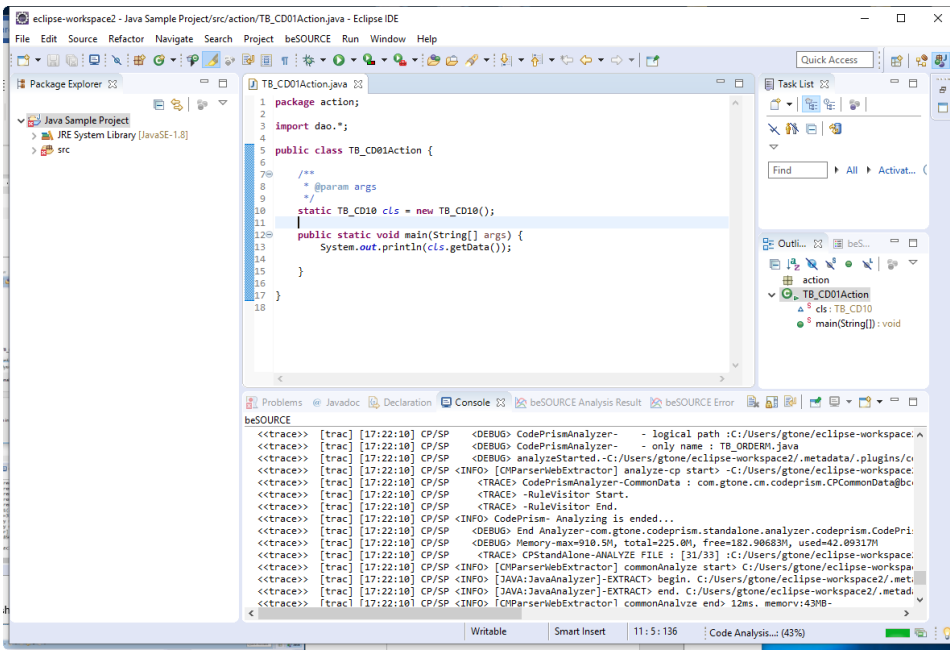
1. Select the project's name > **Properties**.
2. In the left pane, select **beSOURCE**.

- Under the Local Analysis tab, select a rule set (for example, Java/Web Security Ruleset).

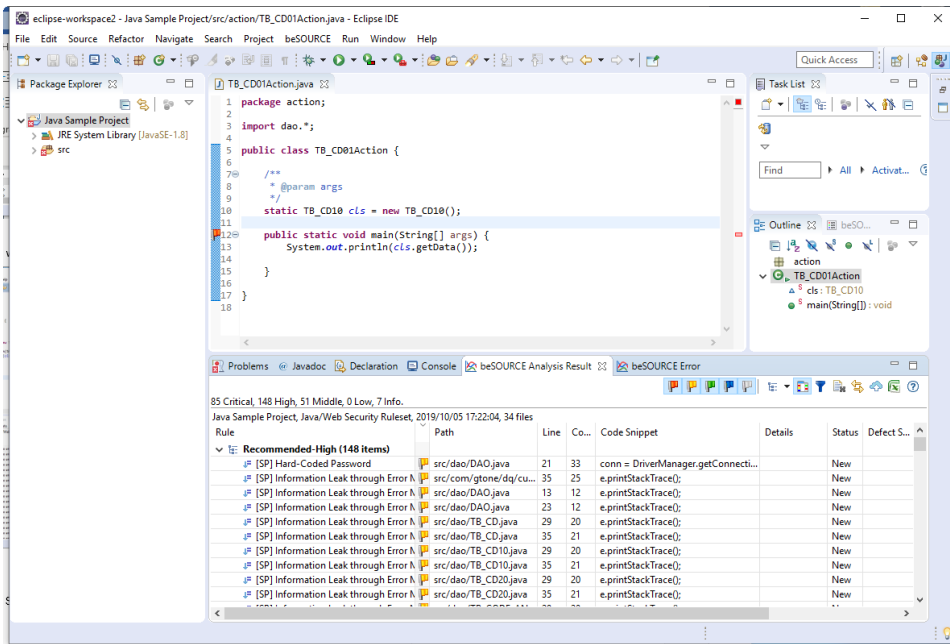
NOTE: The JavaWeb language means it will analyze Java, JSP, XML and Jar files all together.



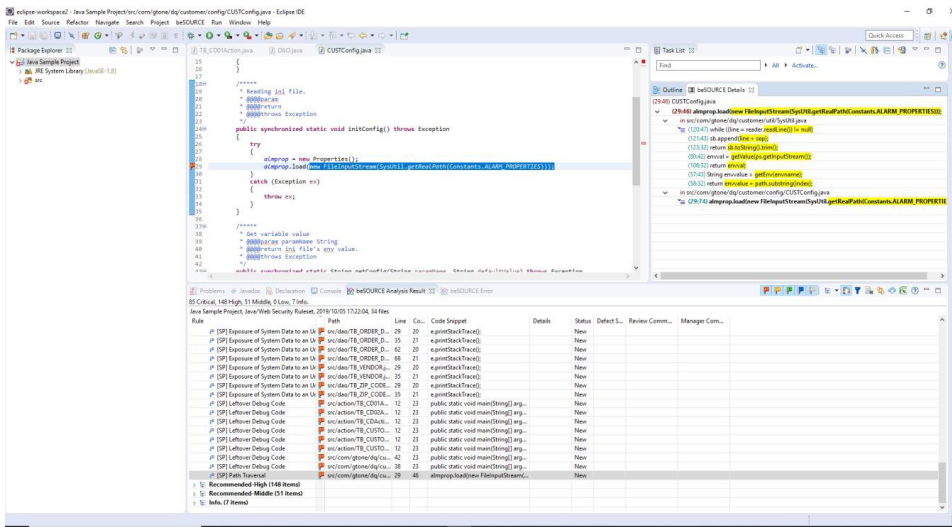
- Select **Apply and Close**.
- Select the project's name > **beSOURCE** > **beSOURCE Analysis**. The analysis on the source files will begin.



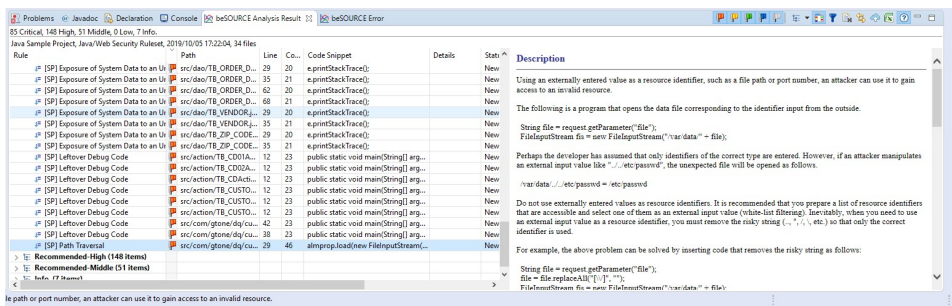
6. The beSOURCE Analysis Results tab will show any violations.



7. Double-click a rule violation to open the Source Editor and Trace Information window.



8. Select the **Detail Information** (🔍) icon to view the rule's description, code examples, and related secure coding standards.

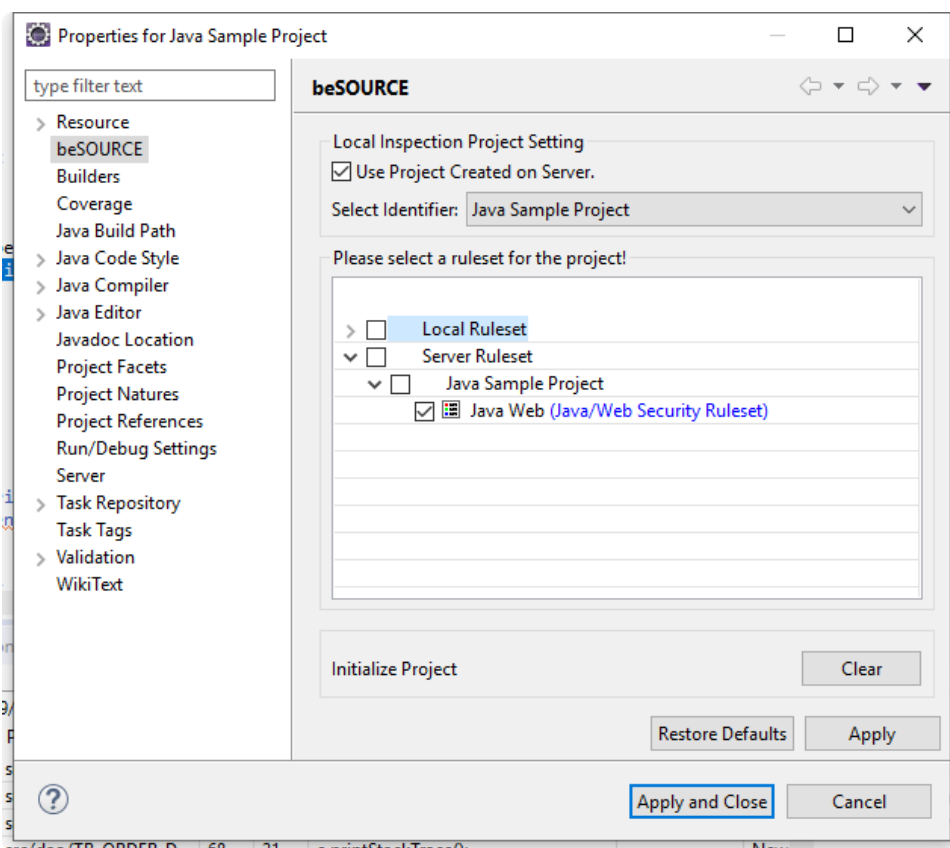



NOTE: For more information about the plugin's functions, select **Help > Help Contents > beSOURCE User Guide**.

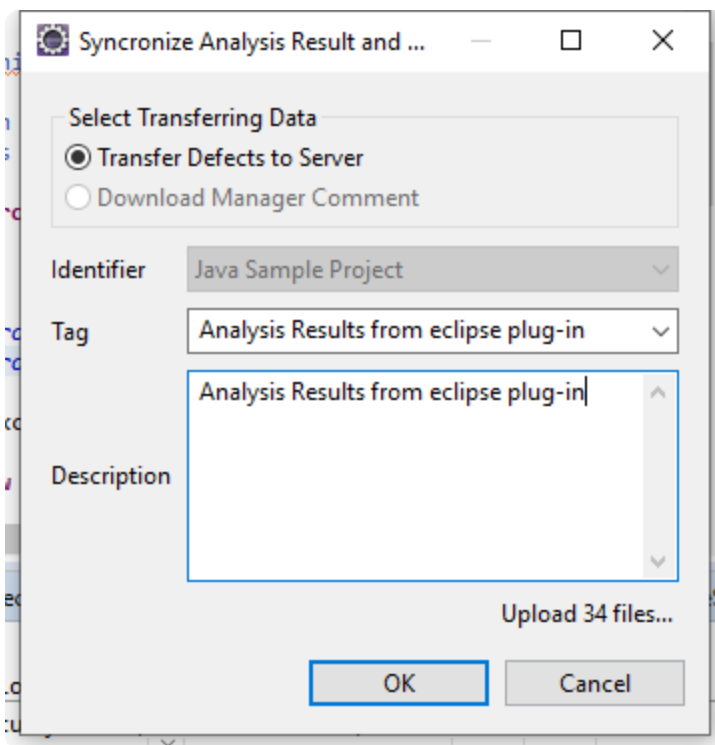
Server mode analysis in the Eclipse plugin


Now, we will change the project property to use the server mode analysis.

1. Select the project's name > **Properties**.
2. Select **Use Project Created on Server**.
3. Select **Java Sample Project** as an identifier (created by administrator in the previous chapter).
4. Select **Server Ruleset > Java Sample Project > Java Web (Java/Web Security Ruleset)**.
5. Select **Apply and Close**.



6. Select the project's name > **beSOURCE** > **beSOURCE Analysis**. The source files will be analyzed after synchronizing the rule set and libraries on the server.
7. Select the **Transfer** () icon. The Synchronize Analysis Result dialog opens.
8. In the **Tag** box, enter one or more tags for the transfer.
9. In the **Description** box, enter a description for the transfer.



10. Select **OK**.
11. On the dialog that appears, select **OK**.
12. Select the **Export** () icon.
13. Specify the file name, and then select **OK**. You can export the analysis results to an Excel file.

User ID	User Name	Department Name	Analysis Time	Upload Date/Time	Ruleset	Tag	Description	File Count	Defect Files	Defect Count	Critical	Rec...	Rec...
besourcedev	Key	Department Category	2019-10-05 17:32:01	2019-10-05 17:34:34	Java/Web Security ...	Analysis Results from eclipse plug-in	Analysis Re...	34	29	291	85	148	51
besourcedev	Key	Department Category	2019-10-05 14:03:47	2019-10-05 14:07:30	Java/Web Security ...	Key, Local scan	Server mod...	34	29	291	85	148	51

NOTE:

- If you run Admin Console and open details of View Self-inspection Result, you can see the uploaded analysis results from Eclipse plugin.
- The uploading analysis results feature is also available for local mode analysis.

Analyzing Source Files in the IntelliJ IDEA Plugin

This chapter describes how to analyze sample source files on the developer computer using the beSOURCE IntelliJ or Android Studio plugin.

You can analyze source files using either of the following two modes in the beSOURCE IntelliJ or Android Studio plugin:

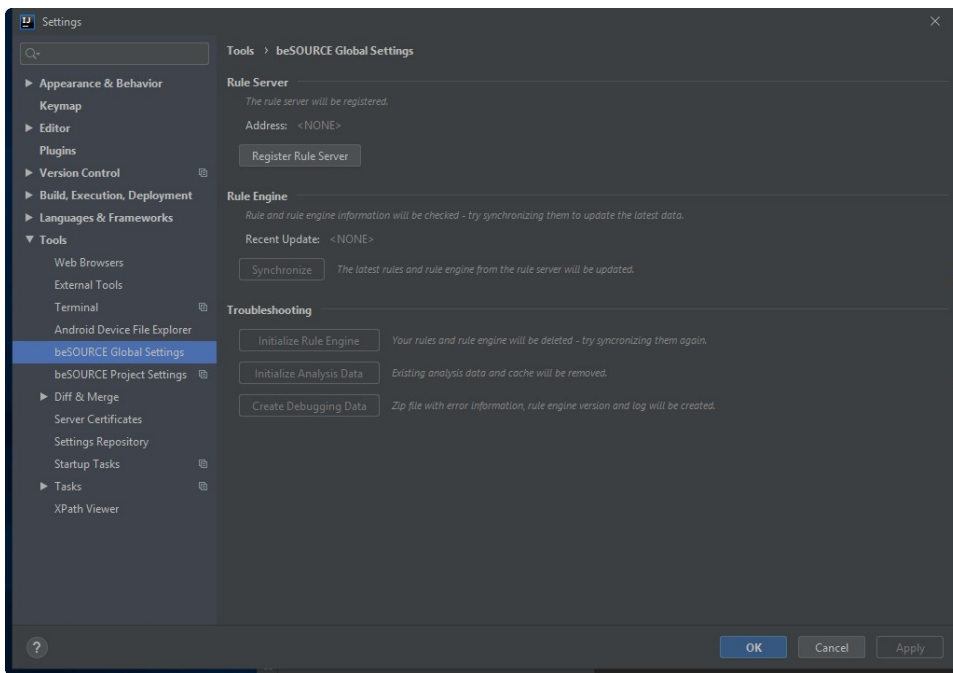
- Local - Uses a local rule set to scan source files on the local computer.
- Server - Uses server-side setting information such as rule sets and libraries by synchronizing with the beSOURCE Server.

NOTE: It is recommended to copy sample source files from the beSOURCE Server to your local computer. The sample file directory on the server is [beSOURCE_Installation_Directory] \Sample.

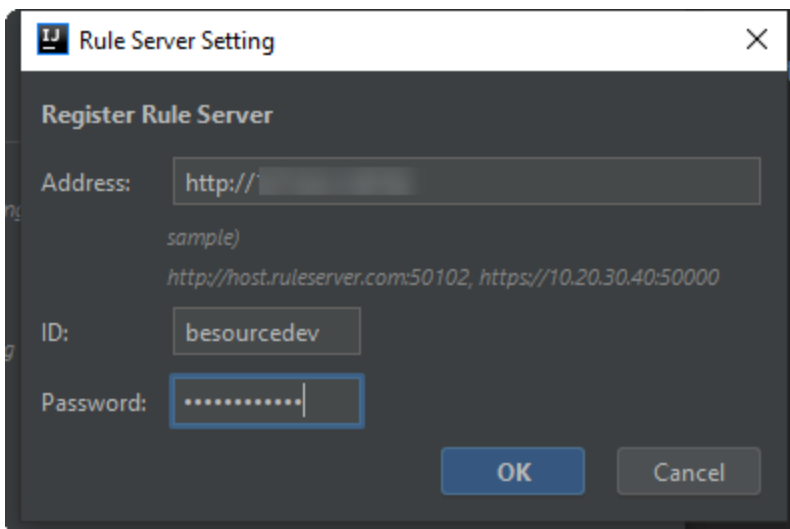
Setting up the IntelliJ or Android Studio plugin

After installing the beSOURCE IntelliJ or Android Studio plug, do the following to set up connection information:

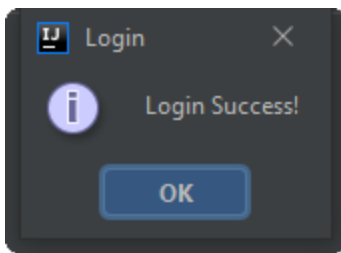
1. Select **File > Settings > beSOURCE Global Settings**, or select **File > Settings > Tools > beSOURCE Global Settings**.
2. Select **Register Rule Server**.



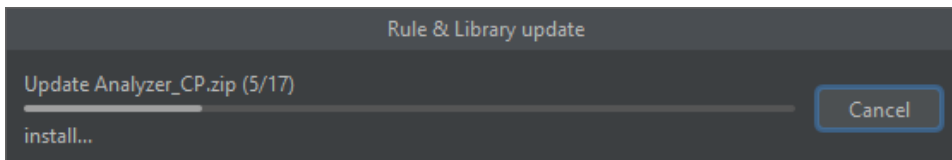
3. On the **Rule Server Setting** dialog, do the following:
 1. In the **Address** box, enter the beSOURCE Server's address.
 2. In the **ID** and **Password** boxes, enter **besourcedev**.
4. Select **OK**.



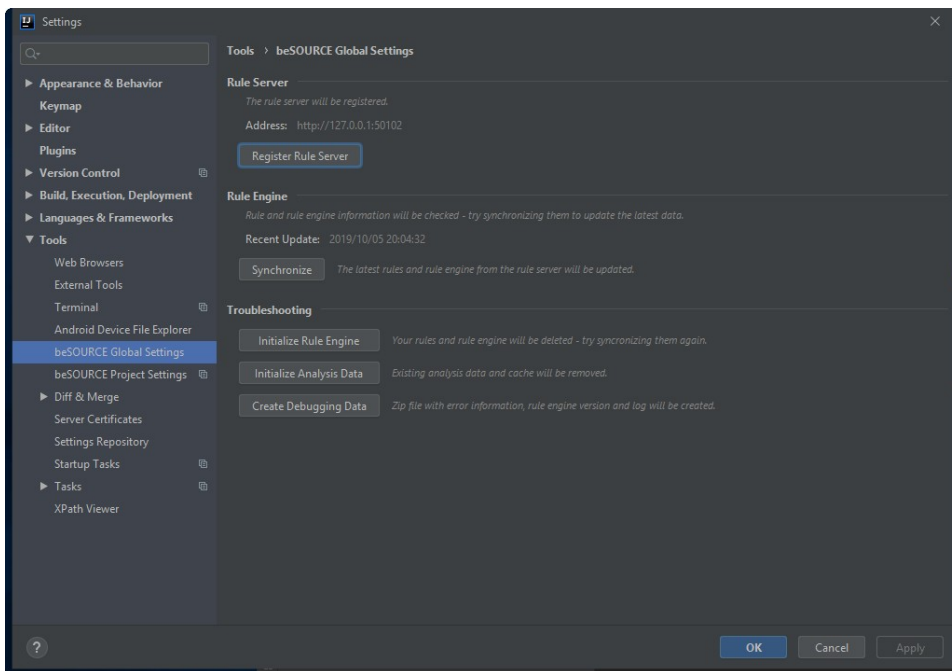
5. On the **Login** dialog, select **OK**.



6. When you log in for the first time, an update will run to download rule sets and the analysis engine.



7. To update the rule set and analysis engine manually, select **Synchronize** on the beSOURCE Global Settings window.

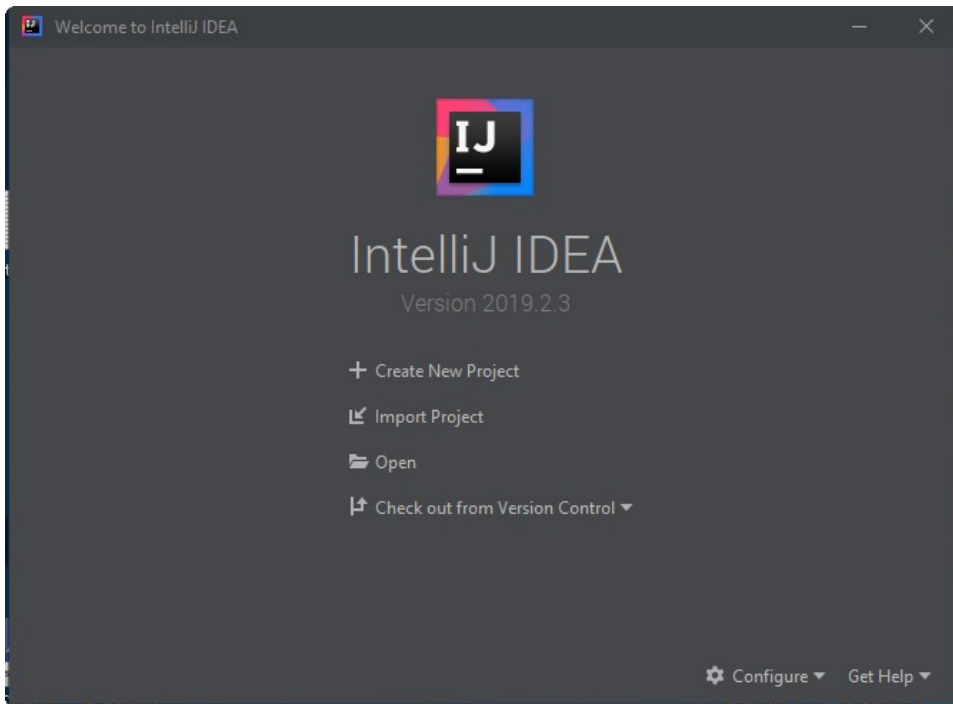


NOTE: If abnormal behavior or an unexpected error occurs in the IntelliJ/Android Studio plugin, select **Initialize Rule Engine** to initialize the rule engine and local rule sets, and then select **Synchronize** to download the rule engine and rule sets again.

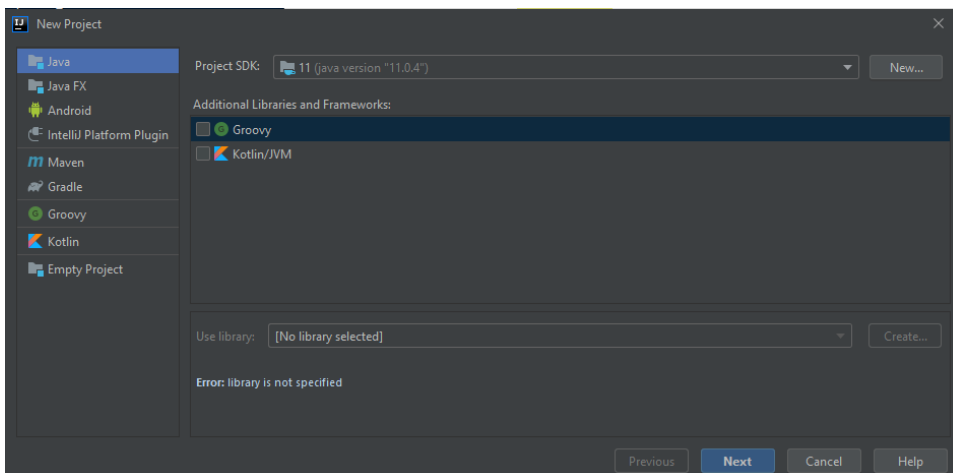
Now, you need to create a Java project for this hands-on exercise.

NOTE: The beSOURCE IntelliJ or Android Studio plugin requires Java 1.8 or later.

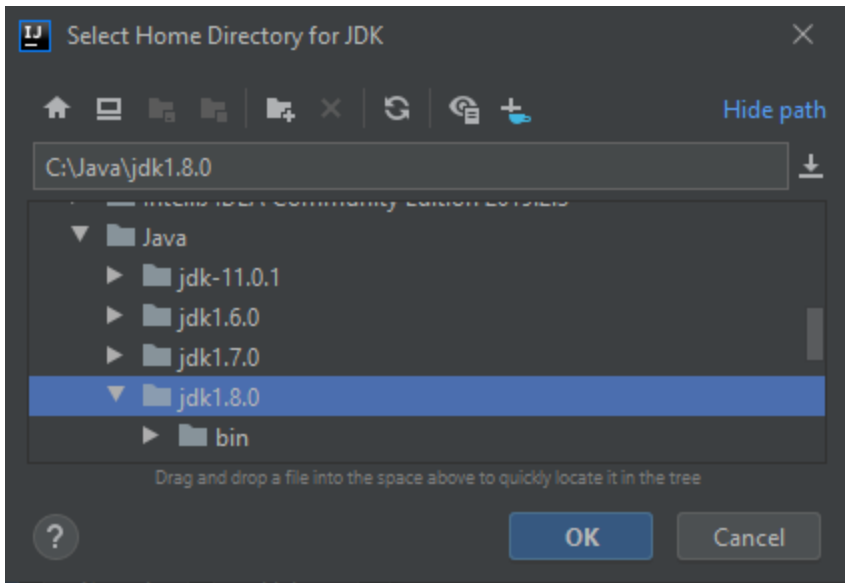
1. Open **IntelliJ IDEA**.
2. Select **Create New Project**.



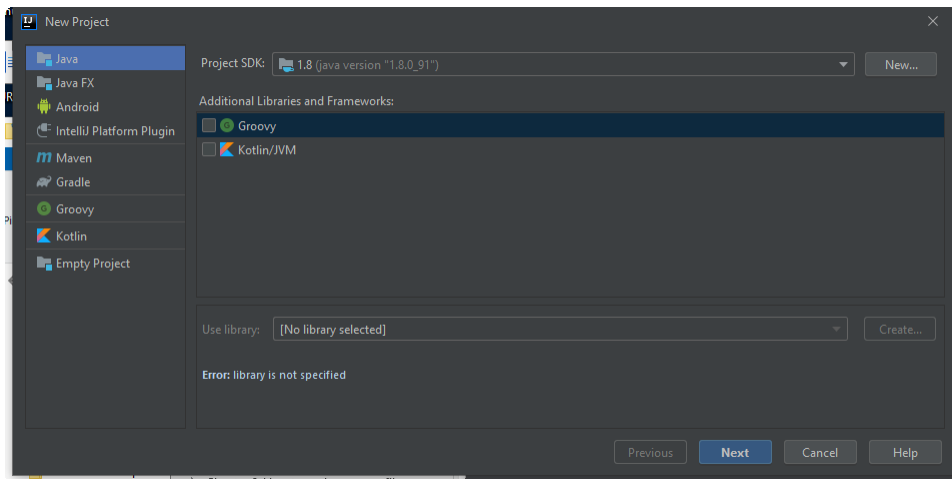
3. Select **Java**.
4. Next to the **Project SDK** box, select **New**.



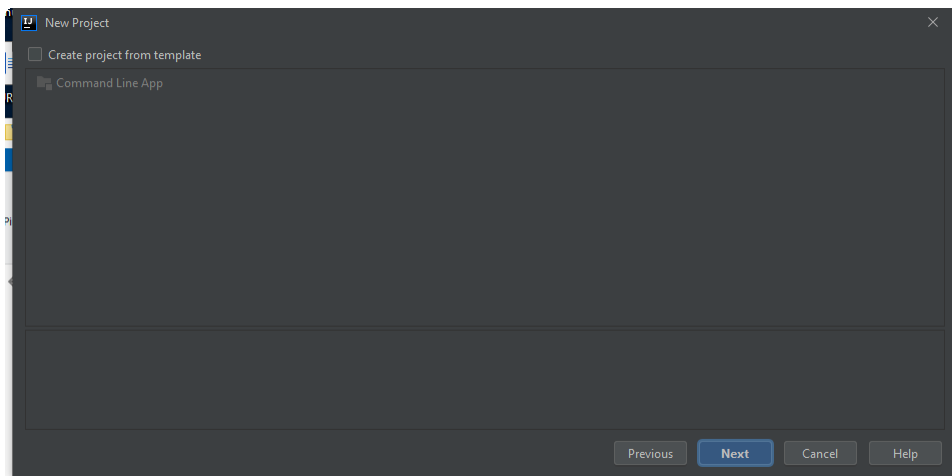
5. Select the location where JDK 1.8 is installed.
6. Select **OK**.



7. Select **Next**.

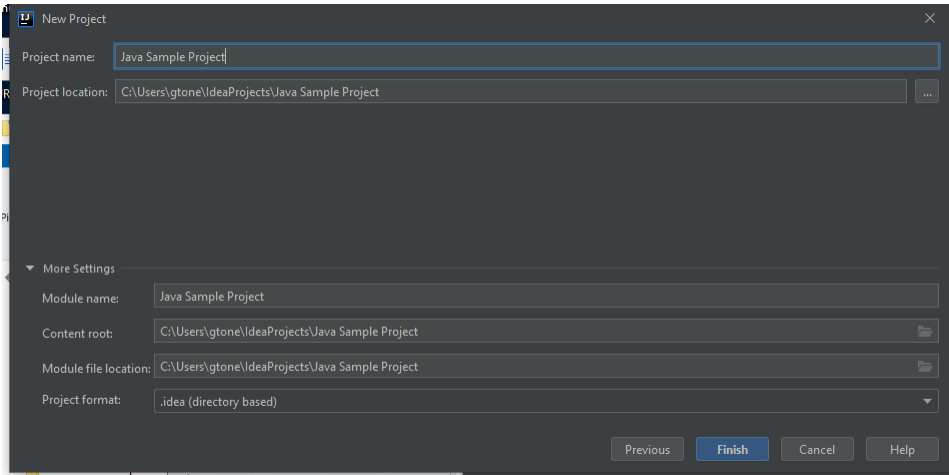


8. Select **Next**.

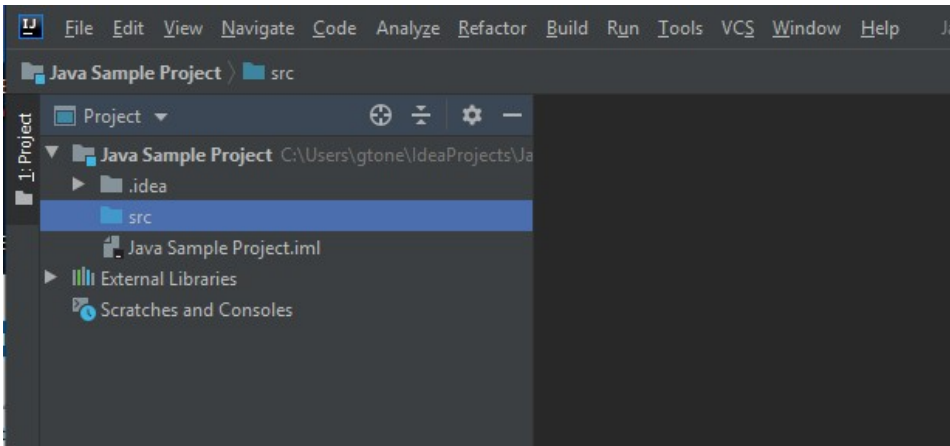


9. In the **Project name** box, enter a name.

10. Select **Finish**.

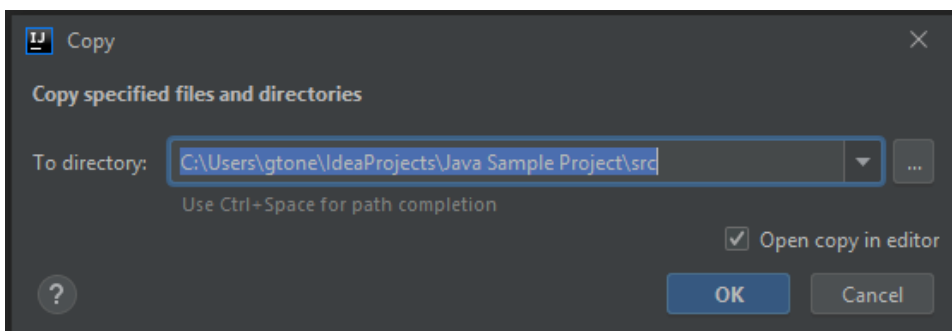


11. Under the new project, select the **src** folder.

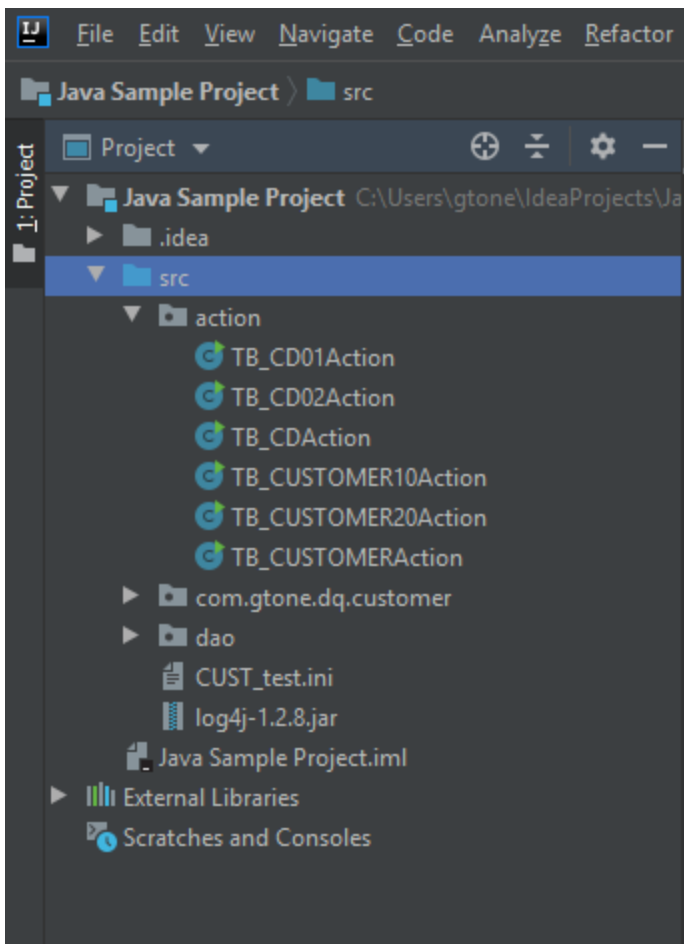


12. Copy all files from the [beSOURCE Server Install directory] \Sample\Java folder, and then paste them into the **src** folder in the project window.

13. Select **OK**.



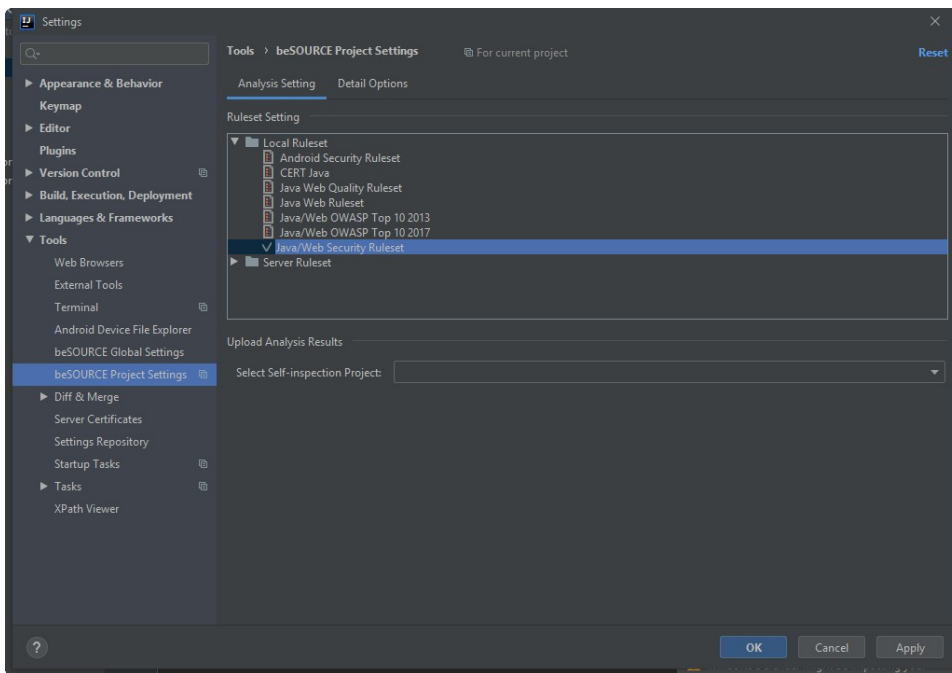
14. The Java sample source files are copied.



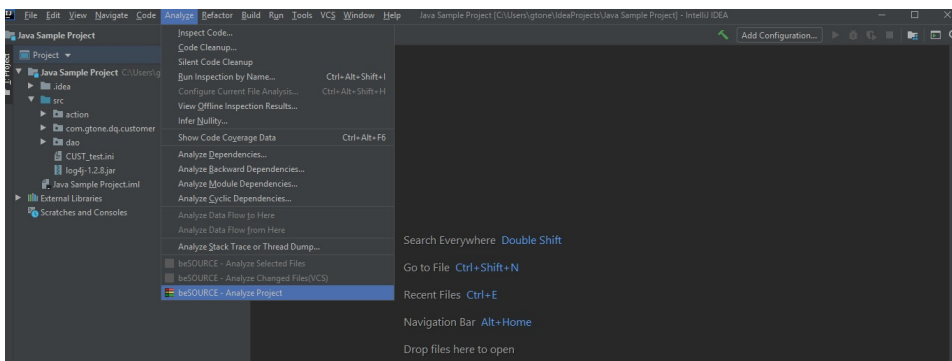
Local mode analysis in the IntelliJ or Android Studio plugin

A developer can analyze Java source files with local rule sets inside the plugin.

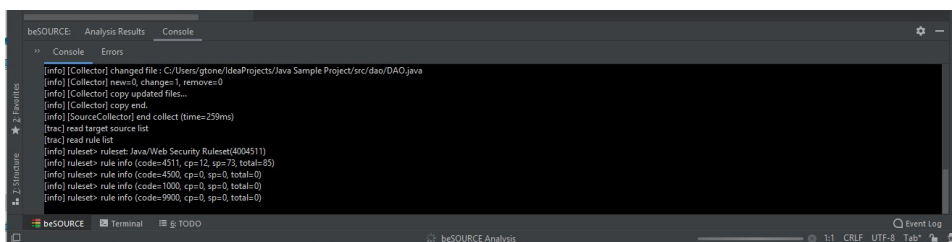
1. Select **File > Settings > beSOURCE Project Settings**, or **File > Settings > Tools > beSOURCE Project Settings**.
2. Select the **Analysis Setting** tab.
3. Select **Local Ruleset > Java/Web Security Ruleset**.
4. Select **OK**.



5. Select **Analyze > beSOURCE – Analyze Project**.

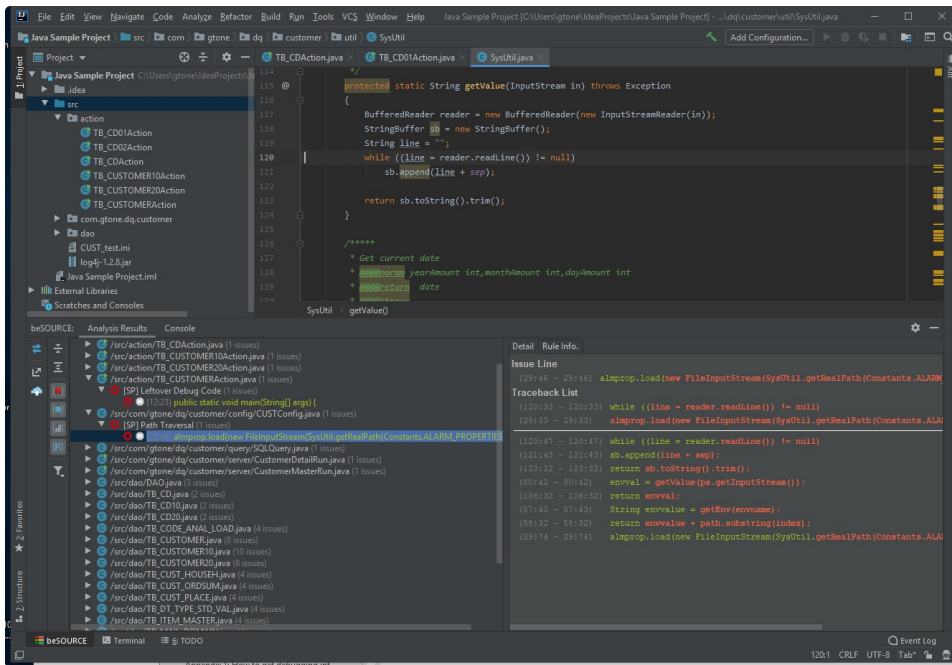


6. The plug-in console opens, and the analysis progress will be shown.

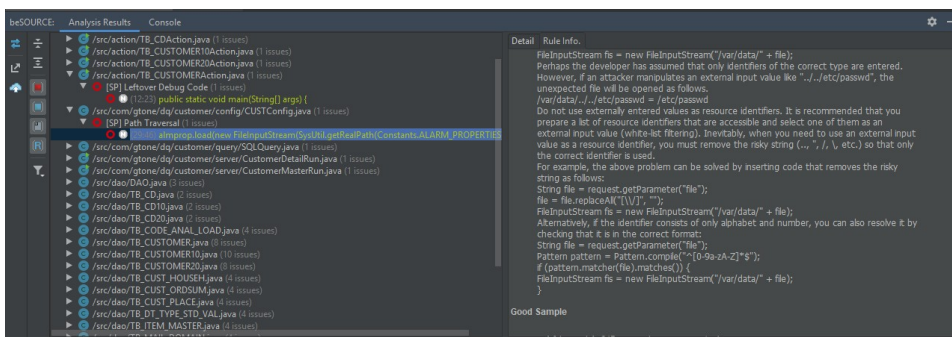


7. The Rule violations list will appear after the analysis job completes.

8. Select a rule violation in the left panel and issue code lines and flow trace information will display in the Detail tab. When you select an issue line link, the Source Code Editor opens.



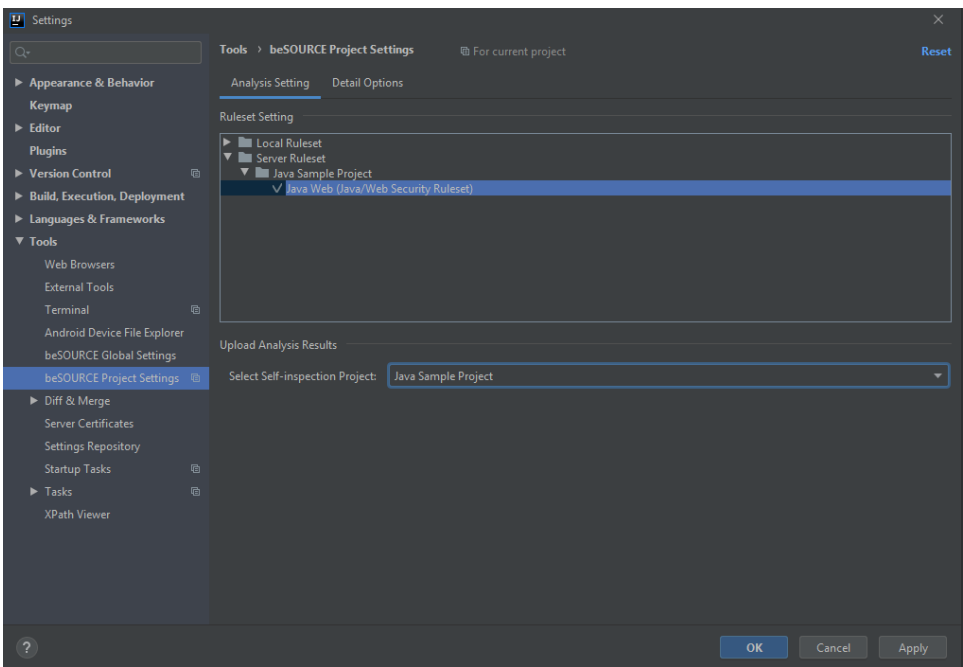
9. Select the **Rule Info** tab to view the rule's description.



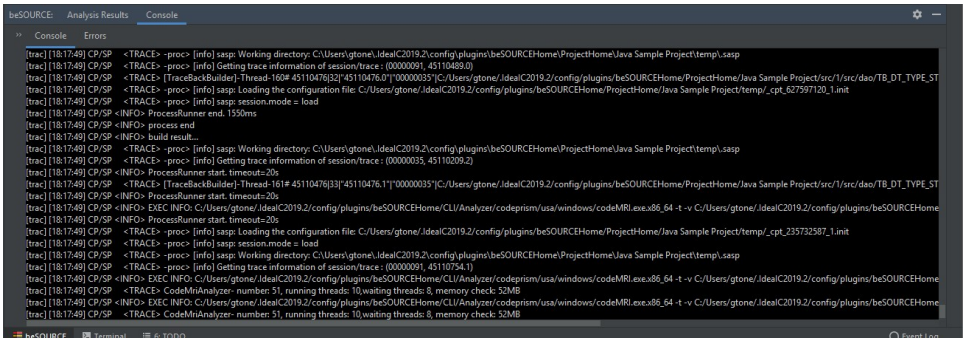
Server mode analysis in IntelliJ or Android Studio plugin

Now, change the project property to use the server mode analysis.

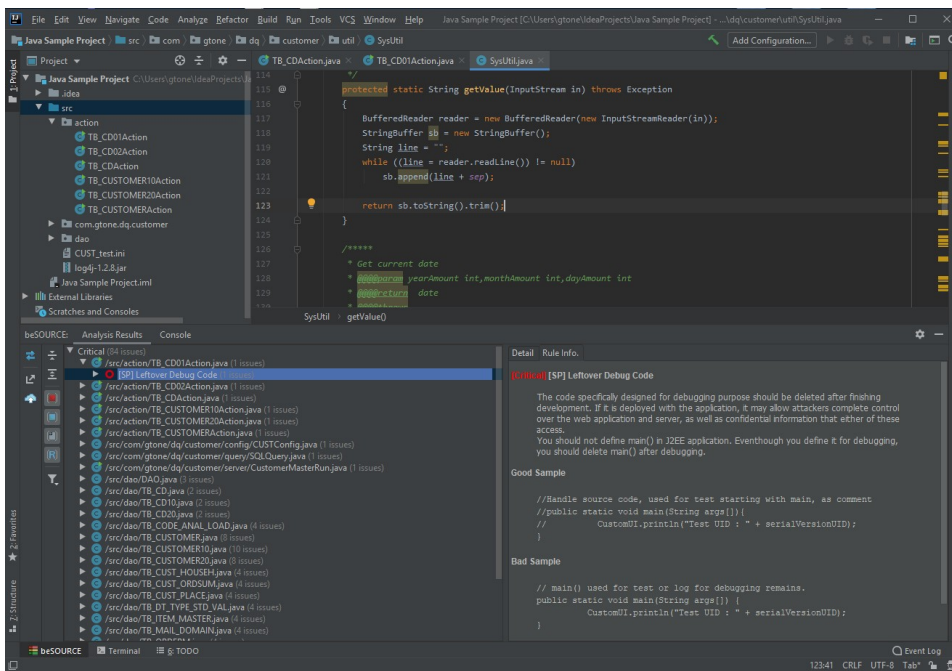
1. Select **File > Settings > beSOURCE Project Settings**, or **File > Settings > Tools > beSOURCE Project Settings**.
2. Select the **Analysis Setting** tab.
3. Select **Server Ruleset > Java/Web Security Ruleset**.
4. Select **Java Sample Project** in the **Select Self-inspection Project** option.
5. Select **OK**.



6. Select **Analyze > beSOURCE – Analyze Project**. It will try synchronization with the server-side configuration and the console panel shows the progress of analysis.



7. The rule violations list will be shown after the analysis job is complete.



NOTE:

- If you run Admin Console and open details of View Self-inspection Result, you can see the uploaded analysis results from Eclipse plugin.
- The uploading analysis results feature is also available for local mode analysis.

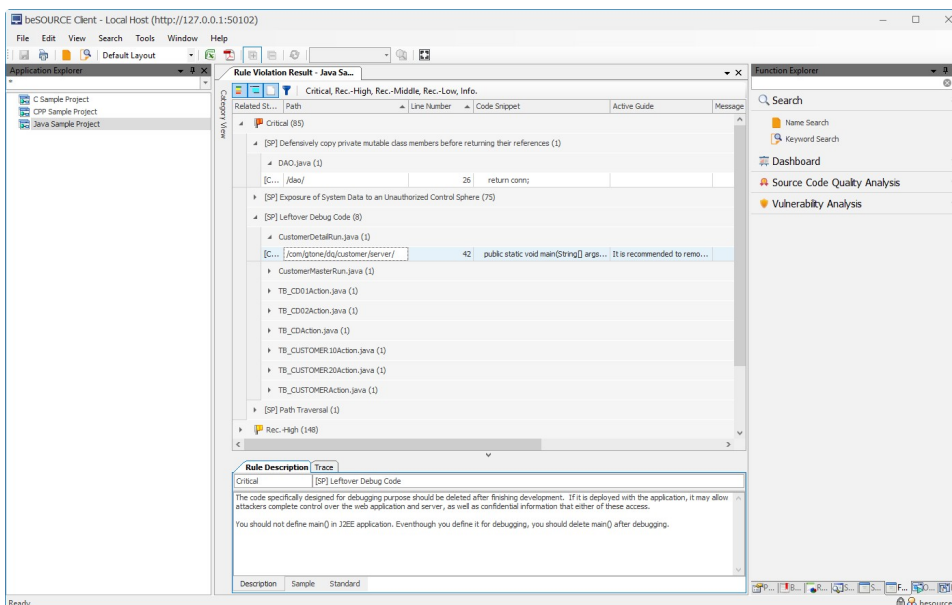
Filtering Out False Alarms

If you find a false alarm in the violations list, you can exclude it by using filtering function. beSOURCE provides status management feature for an individual violation.

Filtering false alarms in beSOURCE Client

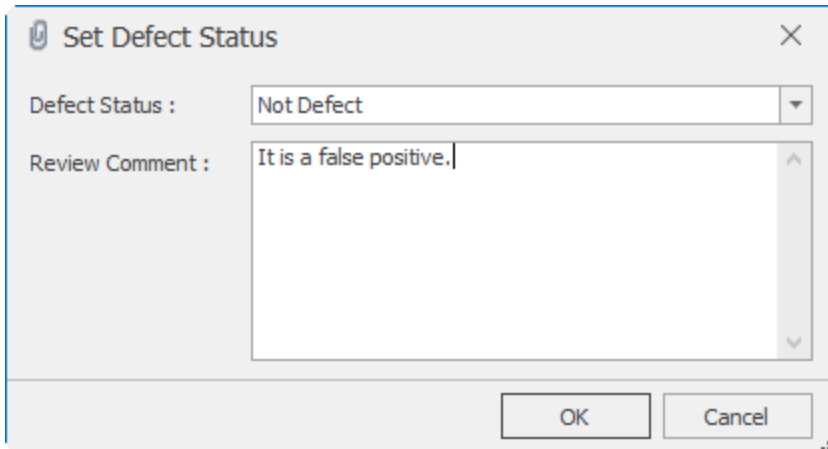
The beSOURCE administrator can view the analysis results on the beSOURCE Server and manage status of each rule violation.

1. Open and log in to beSOURCE_product" style="font-weight: bold;" /> **Client** as an administrator (besource).
2. In the Application Explorer, select **Java Sample Project**.
3. Select **Code Quality/Security Inspection > Show Rule Violations**.
4. Select a rule violation (for example, Left Debug Code – CustomerDetail.java Line no 42).




5. Select **Set Defect Status**.
6. In the dialog that appears, select one value for the rule violation and add a comment.
 - **Cleared** - It is cleared by other actions.
 - **Defect** - It is an explicit rule violation.
 - **Ignorable** - It can be ignored, not critical.

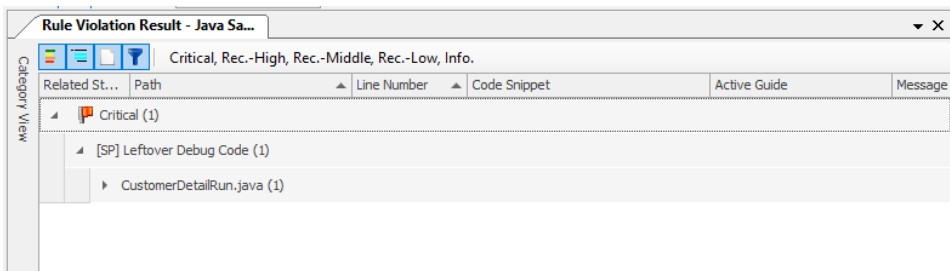
- **Not Defect** - It is not a rule violation and a clear false alarm.
 - **Not Reviewed** - It should be reviewed at a later date.
7. Select **Not Defect** and add your comments.
 8. Select **OK**.



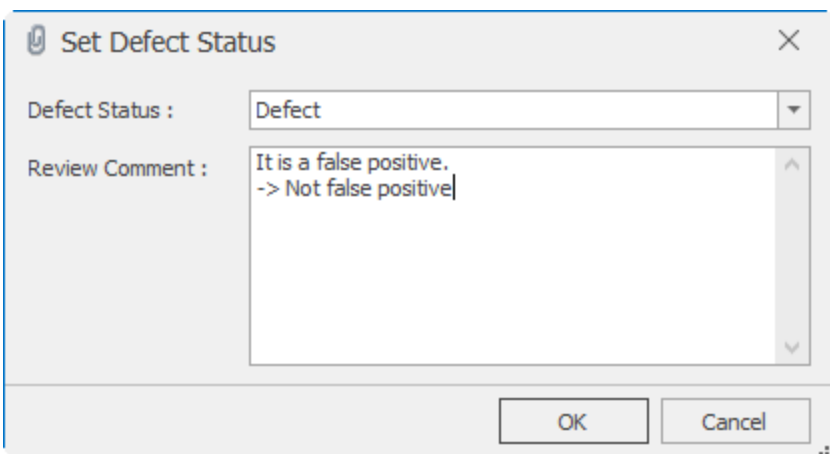
9. The rule violation will disappear in the list and will not be included in any report.


NOTE: The excluded false alarm will be maintained in the next analysis result.

10. Select **View Rule Violations Excluded by Filter** () icon. Only the excluded rule violations appear. Select the icon again to display rule violations.



11. Select **Set Defect Status**.
12. Set the status to **Defect** and add a comment.
13. Select **OK**.

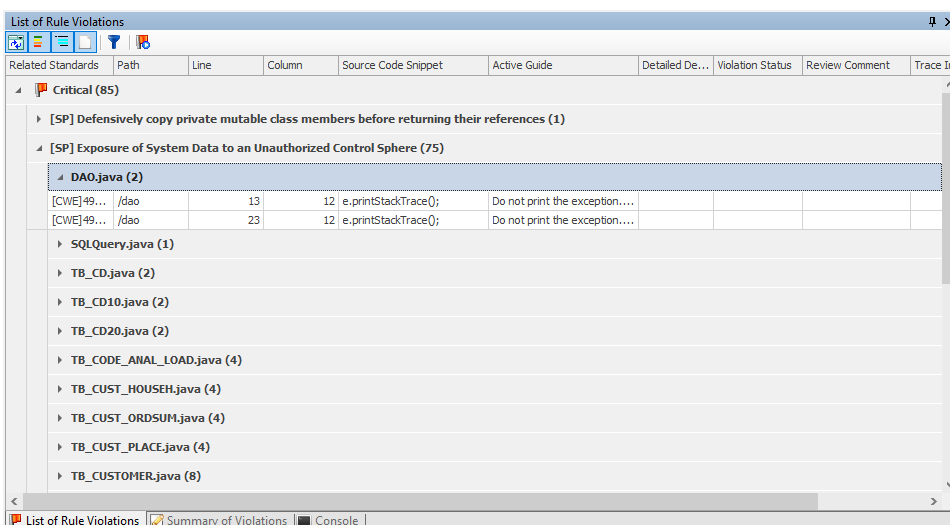


14. The rule violation will disappear from the false alarm list.
15. Select the **View Rule Violations Excluded by Filter** () icon to return to the violations list.

Filtering false alarms in beSOURCE Developer

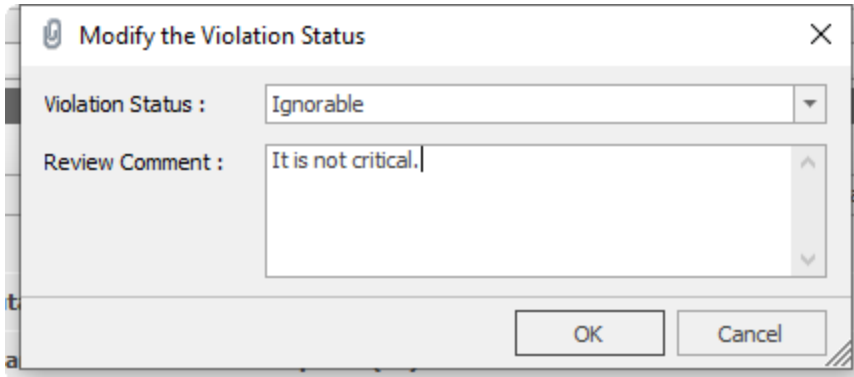
The basic concept is same as beSOURCE Client. Developer can review his own analysis results and set the status of rule violation.

1. Open and log in to beSOURCE_product" style="font-weight: bold;" /> **Developer** as a developer (besourcedev).
2. Open the **Start Page**, and then select **Java Sample Project**.
3. In the **List of Rule Violations** panel, select a rule violation.



4. Select **Modify the Violation Status**.
5. Set to **Ignorable** and add a comment.

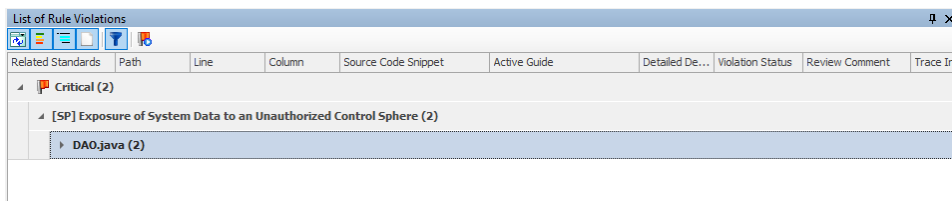
6. Select **OK**.




7. The rule violation will disappear in the list and will not be included in any report.

NOTE: The excluded false alarm will be maintained in the next analysis result.

8. Select **View Excluded Items** () button. Only the excluded rule violation appears.

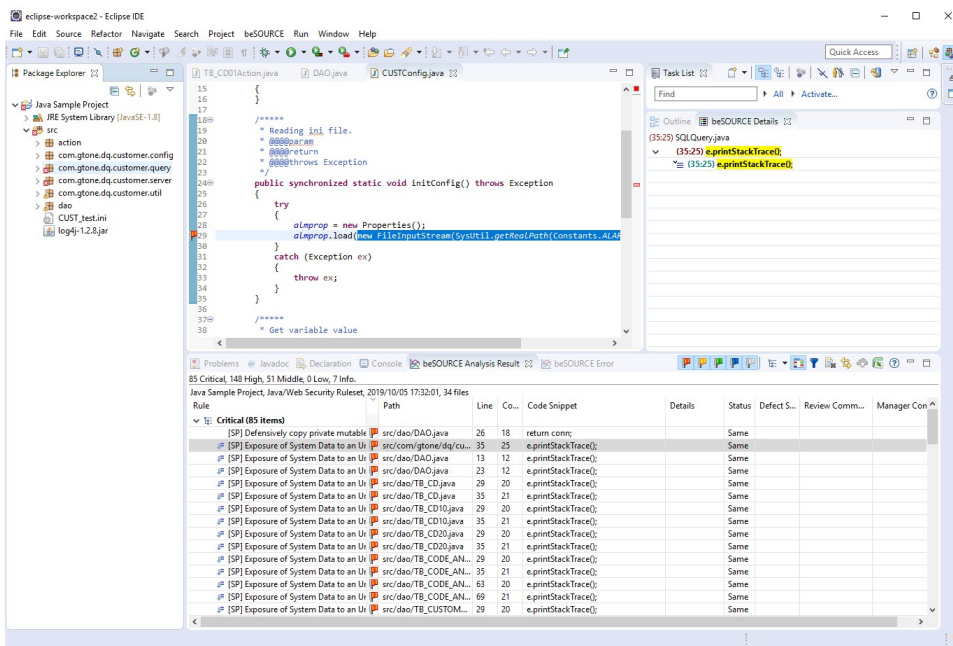


9. Select **Modify the Violation Status**.
10. Set status to **Defect** and add a comment.
11. Select **OK**.
12. Select the **View Excluded Items** () icon to go back to the violations list.

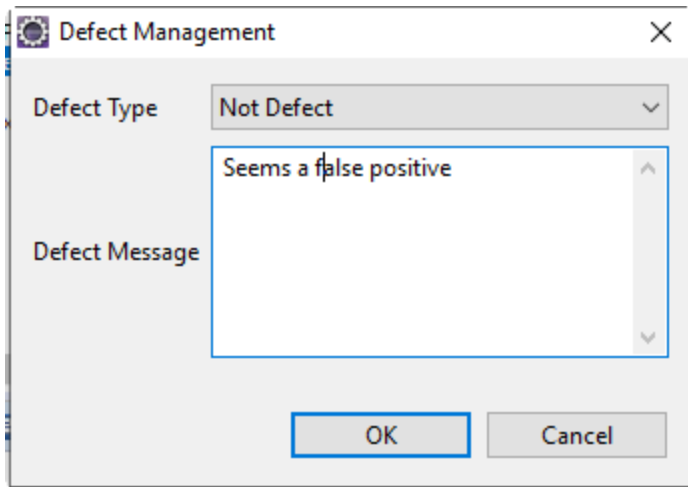
Filtering false alarms in Eclipse plugin

The basic concept is same as beSOURCE Developer. Developer can review his own analysis results and set the status of rule violation.

1. Open and log in to the **Eclipse plugin** as a developer (**beSOURCE > Login**).
2. Open the **Java Sample Project** project. If required, analyze the project again.
3. Select a rule violation in the list.



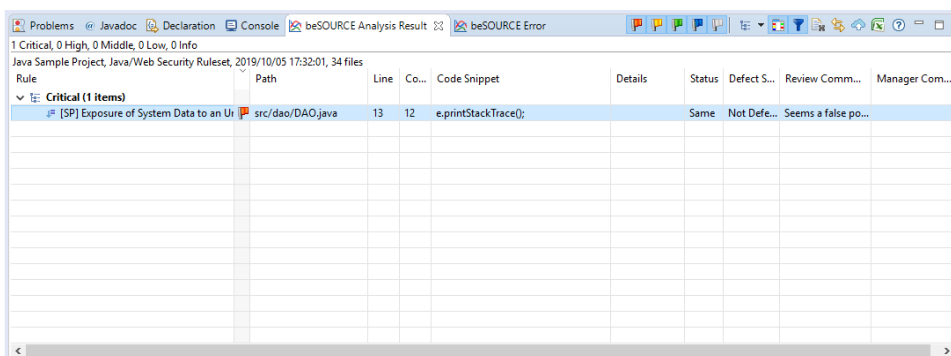
4. Select **Input Defect Message**.
5. Set status to **Not Defect** and add a comment.
6. Select **OK**.




7. The rule violation will disappear in the list.

NOTE: The excluded false alarm will be maintained in the next analysis result.

8. Select the **View Excluded Items** () icon. Only the excluded rule violations appear.




9. Select **Input Defect Message**.
10. Set the status to **Defect** and add a comment.
11. Select **OK**.
12. Click the **View Excluded Items** () icon to go back to the violations list.


NOTE: This feature is same in the IntelliJ/Android Studio plugin

Active Filtering between beSOURCE Server and developer PC

The active filtering means synchronization of rule violation status between server and developer PC. For example, if a project in beSOURCE Developer uses server mode analysis and beSOURCE administrator excludes some rule violations, the false alarm information will be automatically applied to beSOURCE Developer project.

1. Open and log in to the beSOURCE_product" style="font-weight: bold;" /> **Client** as an administrator.
2. In the **Project Explorer**, select **Java Sample Project**.
3. Select **Code Quality/Security Inspection > Show Rule Violations**.
4. Select a rule violation (for example, Left Debug Code – CustomerDetail.java Line no 42).
5. Select **Set Defect Status**.
6. Select the **Not Defect** status and add some comments.
7. Select **OK**.
8. Open and log in to beSOURCE_product" style="font-weight: bold;" /> **Developer** as a developer.
9. Open the **Java Sample Project** project. Its Analysis mode should be set to **SERVER**.
10. Select the **Start Analysis** () icon.
11. Check the 'List of Rule Violations' panel after the analysis is completed.



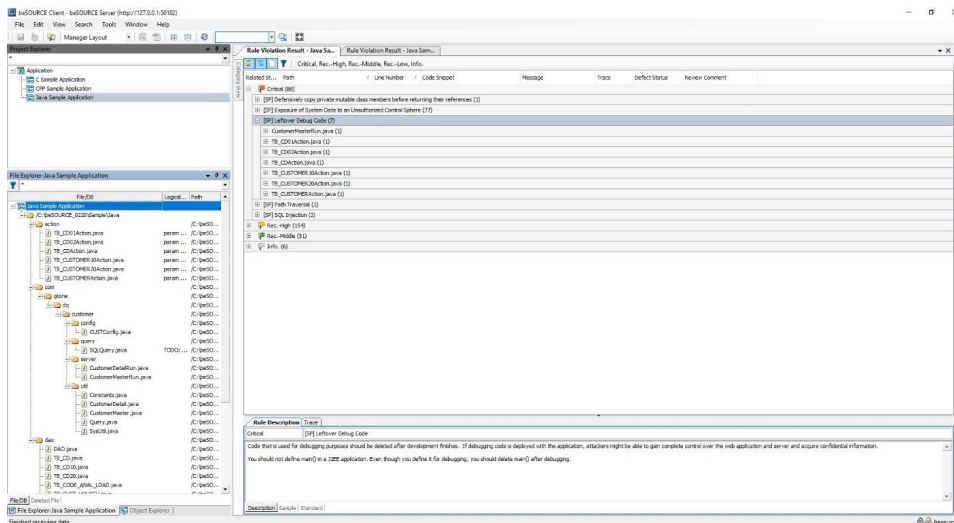
12. Select the **View Excluded Items** () icon. The false alarm set by the administrator is automatically applied to the beSOURCE Developer’s analysis results.

NOTE:

- This feature is same for Eclipse, IntelliJ/Android Studio plugin analysis.
- The active filtering is available for only server mode analysis.

Automatic filtering in analysis engine

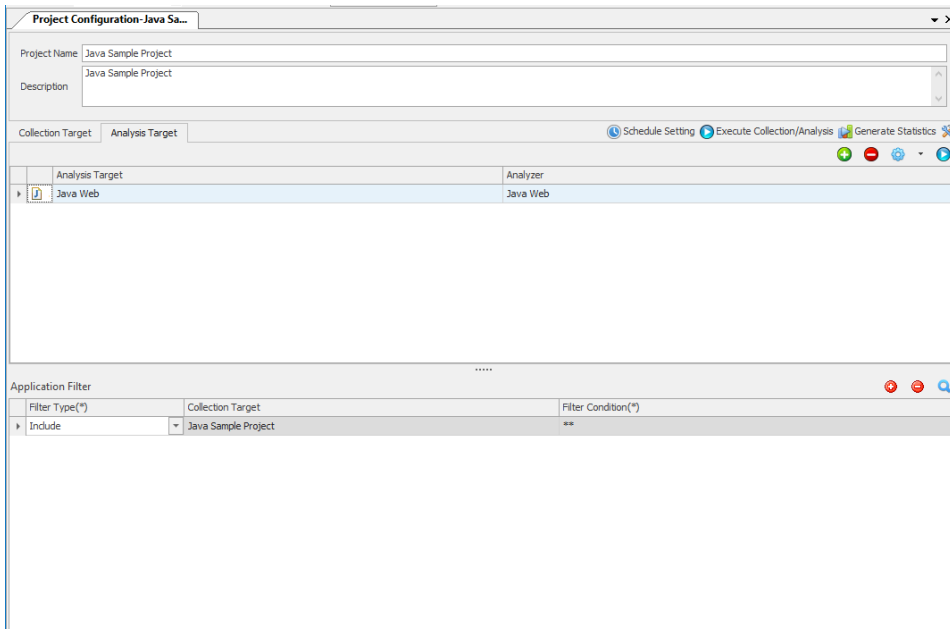
Administrator can set filtering option to exclude some specific rule violations by file or directory.



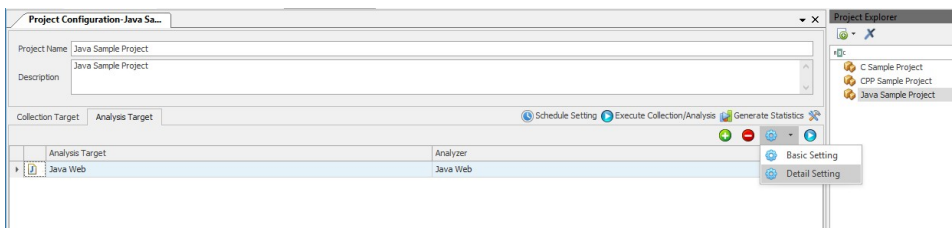
Assume that the administrator wants to remove the Leftover Debug Code rule violations in a series of Action java files from Java Sample Project project.

The project imports source files from C : \beSOURCE\Sample\Java base directory. The base directory has three sub-folders: action, com, and dao. A series of Action java files are located in the **action** folder.

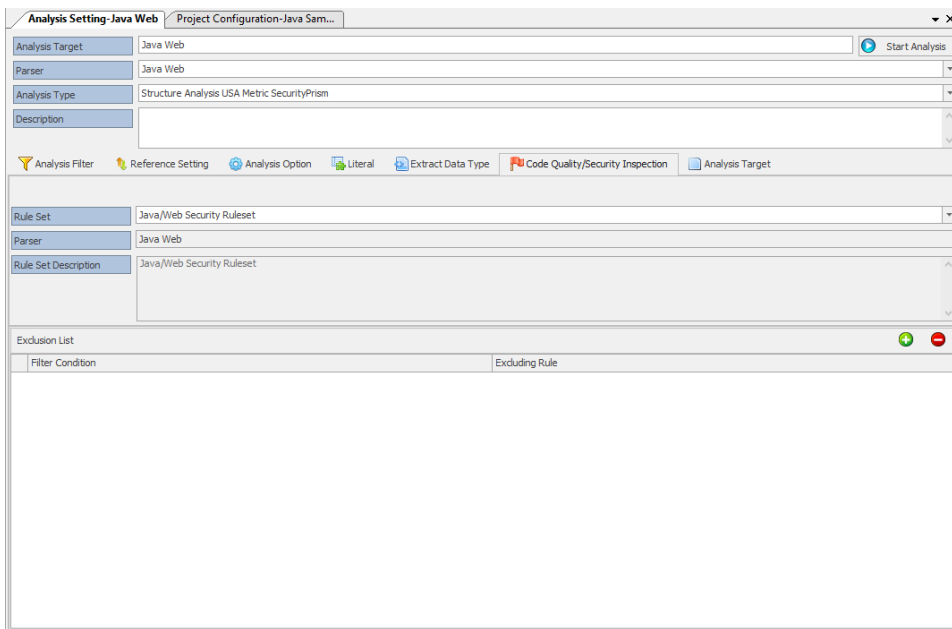
1. Open and log in to the **Admin Console** as an administrator.
2. Select **View > Project Explorer**.
3. Double-click **Java Sample Project**. The Project Configuration window opens.
4. Select the **Analysis Target** tab.



5. Select **Detail Setting**. The Analysis Setting window appears.



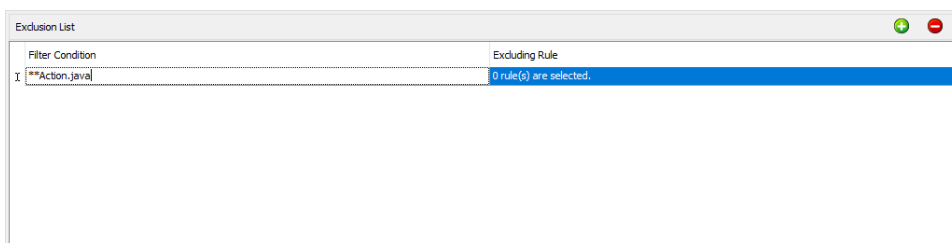
6. Select the **Code Quality/Security Inspection** tab.



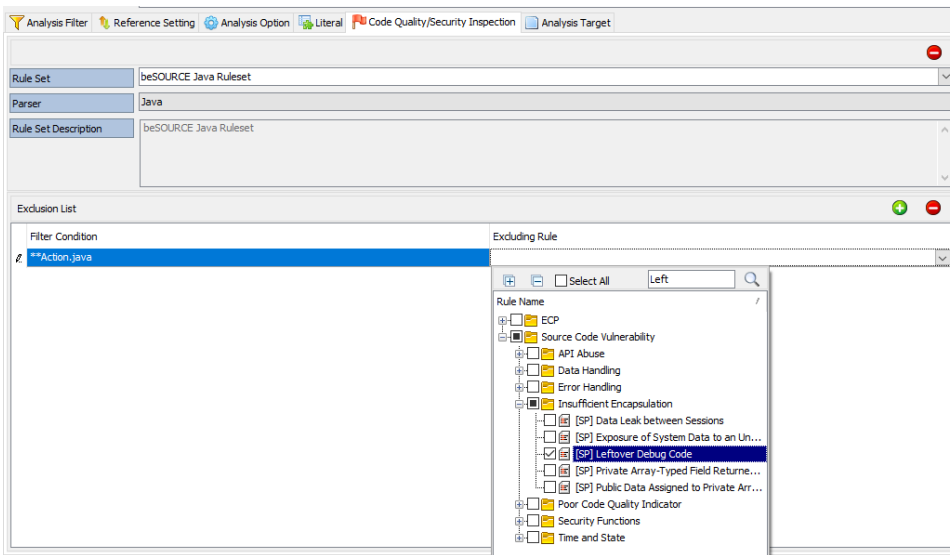
7. Select the **Add (+)** icon.
8. Enter `**Action.java` for the filter condition.

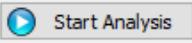

NOTE:

- `**` means everything including sub folders. The `**Action.java` means that all Java files which include Action in the name under all sub folders of the base directory.
- If you want to exclude all files in 'com' directory of the base directory, you can set the filter `**/com/**`.

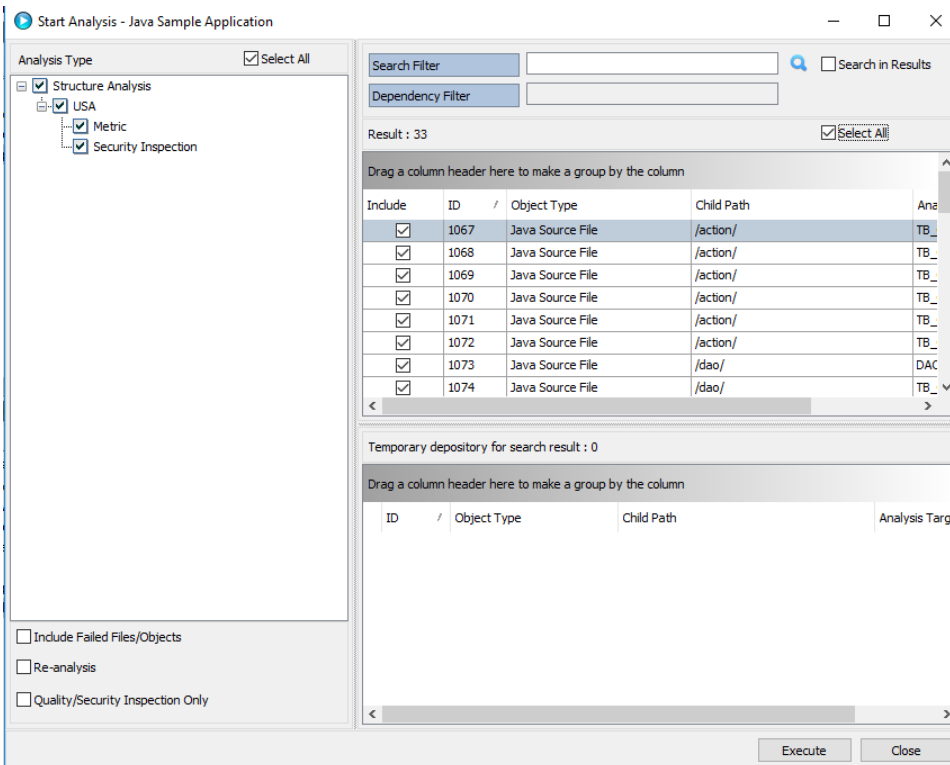


9. In the Excluding Rule box, select the **Leftover Debug Code** rule.

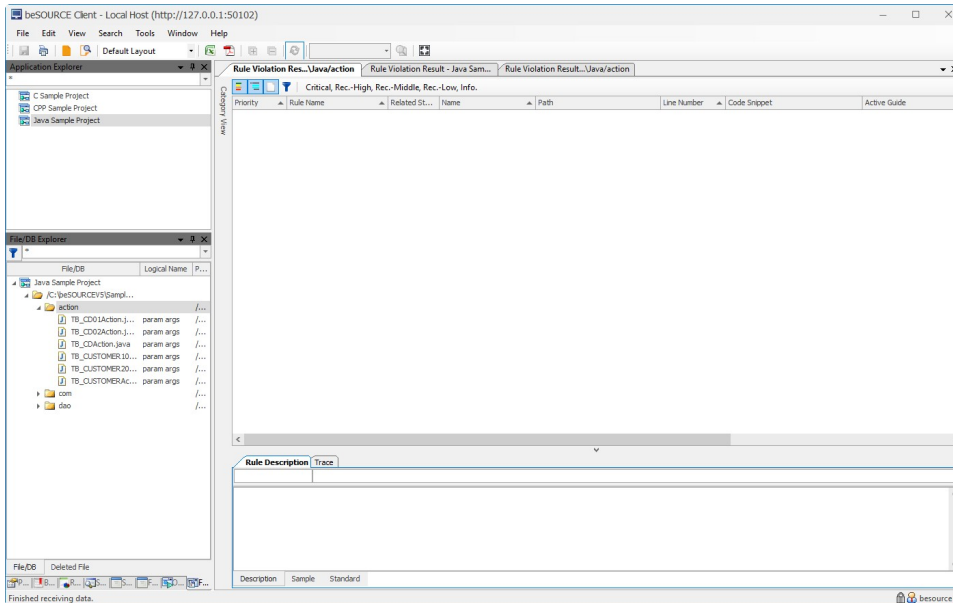


10. Select **File > Save**.
11. Select **Start Analysis** ().
12. Select the **Search** () icon.
13. Select **Select All**.
14. Select **Execute**.

NOTE: These actions are needed to analyze all source files in the project again.



15. Check if the analysis job is completed.
16. Open beSOURCE_product" style="font-weight: bold;" /> **Client**.
17. Double-click the **Java Sample Project** project.
18. Select any Action java file or action folder in the File/DB Explorer.
19. Select **Code Quality/Security Inspection > Show Rule Violations**. You will see there is no rule violation for the Leftover Debug Code.



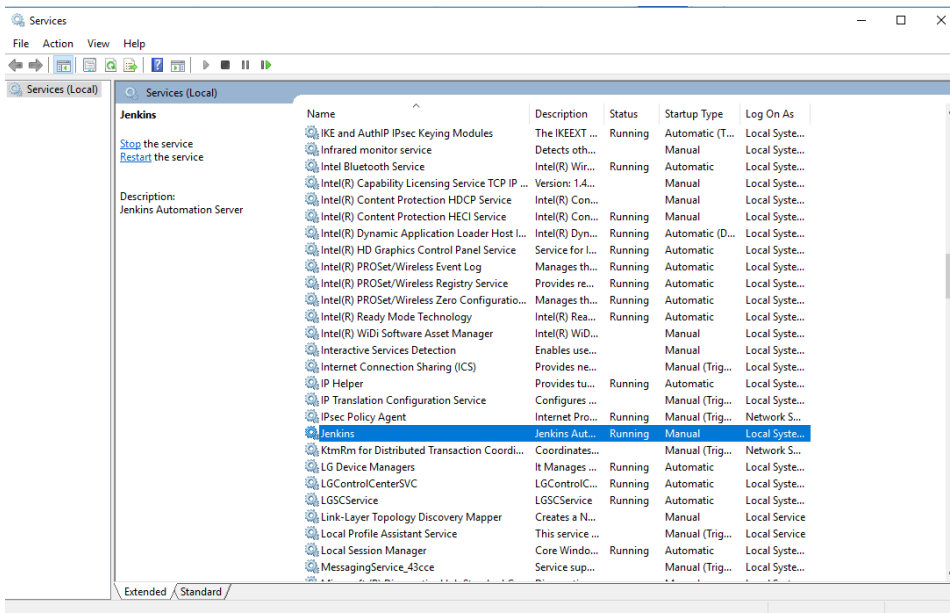
NOTE: If your project in beSOURCE Developer or Eclipse/IntelliJ/Android Studio plugin uses server mode analysis, the above excluding options will be applied automatically to your project in beSOURCE Developer or plugins.

Integrating with Jenkins

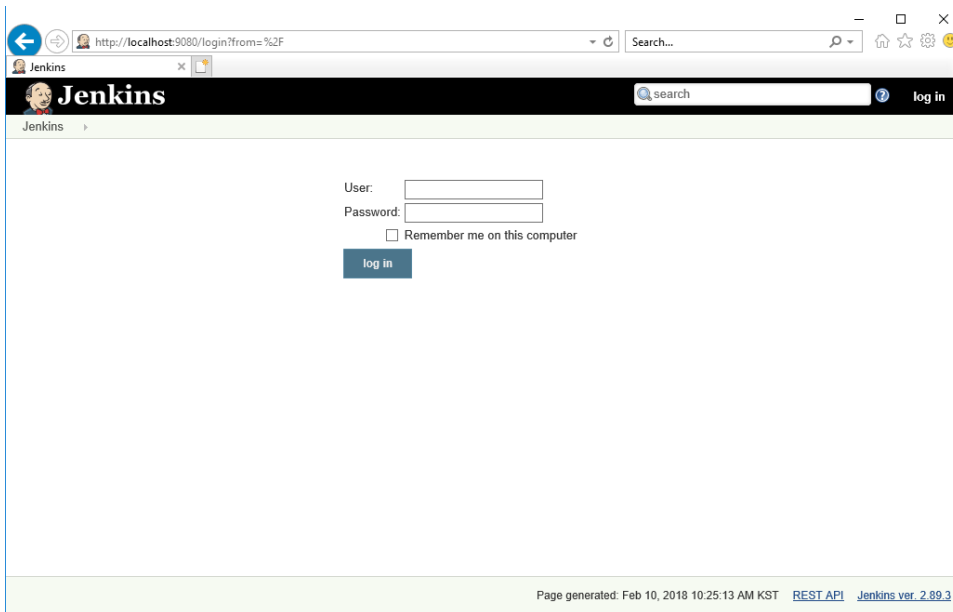
Preparing for Jenkins

If you do not current have Jenkins environment, do the following:

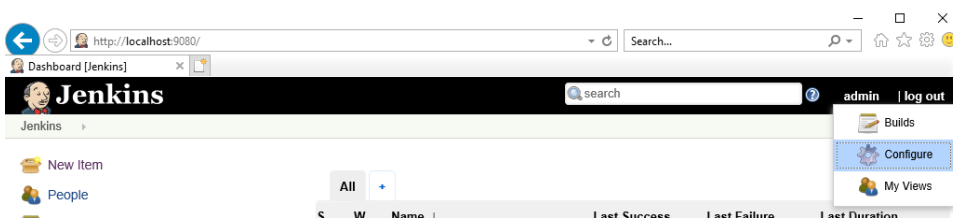
1. Go to <https://jenkins.io/download/>.
2. Download **Jenkins 2.xx.x for: Windows**.
3. Unzip the file.
4. Double-click the **jenkins.msi** file in the unzipped folder and install Jenkins.



5. The program is registered as a Windows service. You can choose **Startup Type** – auto or manual in its property.
6. If you want to change the default port number 8080, open **C:\Program Files (x86)\Jenkins\jenkins.xml**, locate the **--httpPort=8080** argument, and then replace it.
7. Go to **C:\Program Files (x86)\Jenkins\secrets** and locate the **initialAdminPassword** file. Open the file and get the initial password.
8. Open your web browser and go to **http://localhost:8080** (if you changed the port number, update the URL).



9. In the **User** box, enter **admin**.
10. In the **Password** box, enter the initial password from the above file.
11. Select **log in**.
12. Select **admin > Configure**.

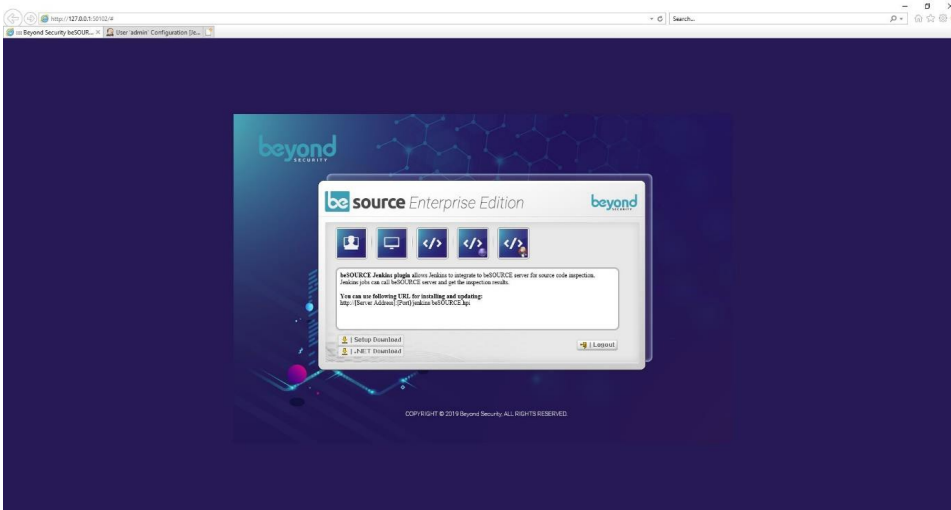


13. Change your password, and then select **Save**.

Installing the Jenkins plugin

You can download the beSOURCE Jenkins plugin from beSOURCE Server web page.

1. Open your web browser.
2. Go to **http://127.0.0.1:50102** to access the beSOURCE Server.
3. Log in as an administrator.
4. Select the Jenkins plugin icon to determine the download URL.



5. Open web browser and go to **<http://127.0.0.1:50102/plugins/jenkins/beSOURCE.hpi>** to download the `beSOURCE.hpi` file.

Installing the Jenkins plugin

To install the Jenkins plugin, do the following:

1. Log in to the Jenkins server.
2. Select **Manage Jenkins**.
3. Select **Manage Plugins** on the management page.
4. Select the **Advanced** tab.
5. Attach the `beSOURCE.hpi` file, and then select **Upload**.
6. On the **Installing Plugins/Upgrades** page, select **Restart Jenkins when installation is complete and no jobs are running**.
7. The **beSOURCE Plugin** appears under the Installed tab of Manage Plugins page once installation is complete

NOTE: For more information, refer to the beSOURCE Enterprise Edition Installation Guide.

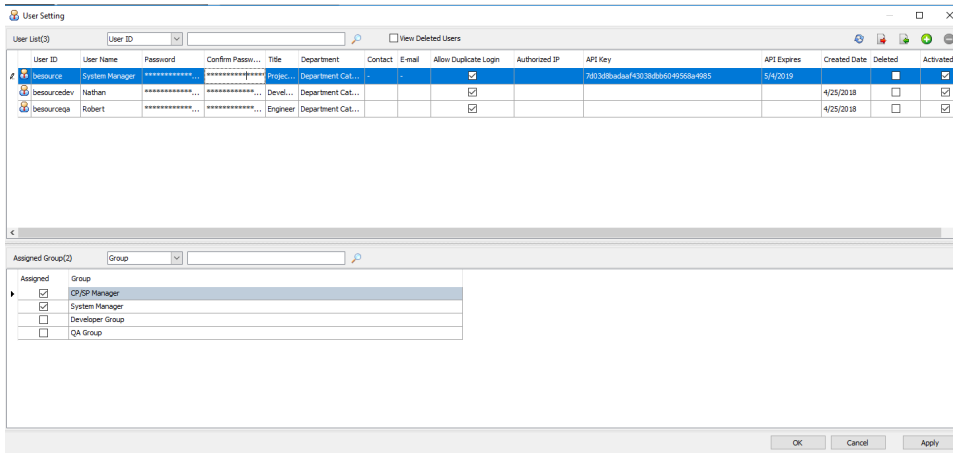
Generating a RESTful API key for the beSOURCE Server

To use the Jenkins plug-in, you need to generate RESTful API key for the beSOURCE server.

To generate the key, do the following:

1. Open and log in to the **Admin Console** as a beSOURCE server administrator.
2. Select **User/Permission > User Setting**.

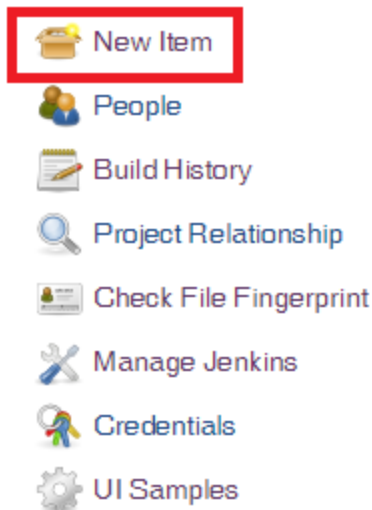
3. Select default system manager (besource) and copy the **API Key** column.
4. Select **Apply**, and then select **OK**.



Connecting Jenkins plugin to beSOURCE Server

To connect Jenkins plug-in to beSOURCE server, do the following:

1. Log in to the Jenkins server.
2. Select **New Item**.



3. Enter the name of item, and then select a project type (Freestyle).
4. Select **OK**.

Enter an item name

beSOURCE Code Inspection Job

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

OK

5. Select **Build** > **Add build step** and add the beSOURCE Plugin.

Build

Add build step ▾

- Execute Windows batch command
- Execute shell
- Invoke top-level Maven targets
- beSOURCE Plugin**

6. Enter values for the following beSOURCE plugin settings:
 - a. **Server URL** - The beSOURCE server URL.
 - b. **API Key** - The RESTful API key generated in the Admin Console.
 - c. **Select Analysis Unit** - The Analysis Unit set in the server.
 - d. **Analysis Type** - **Analyze All** scans all source files, and **Analyze Changes** only scans the changed source files (incremental analysis).
 - e. **Stop the build task when detecting defects** - If some defects with the select priority are found, the build task will stop.
 - f. **Priority** - The beSOURCE inspection rule's priority.
7. Select **Save**.

Build

beSOURCE Plugin

Server URL

API Key

Select Analysis Unit ▾

Analysis Type Analyze All Analyze Changes

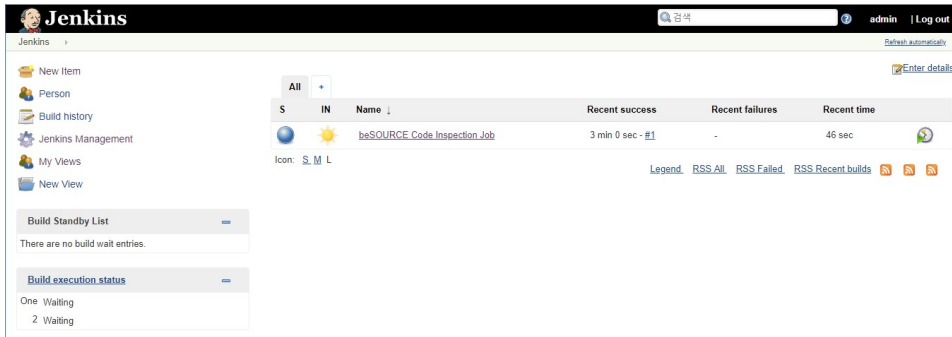
Stop the build task when detecting defects

Add build step ▾

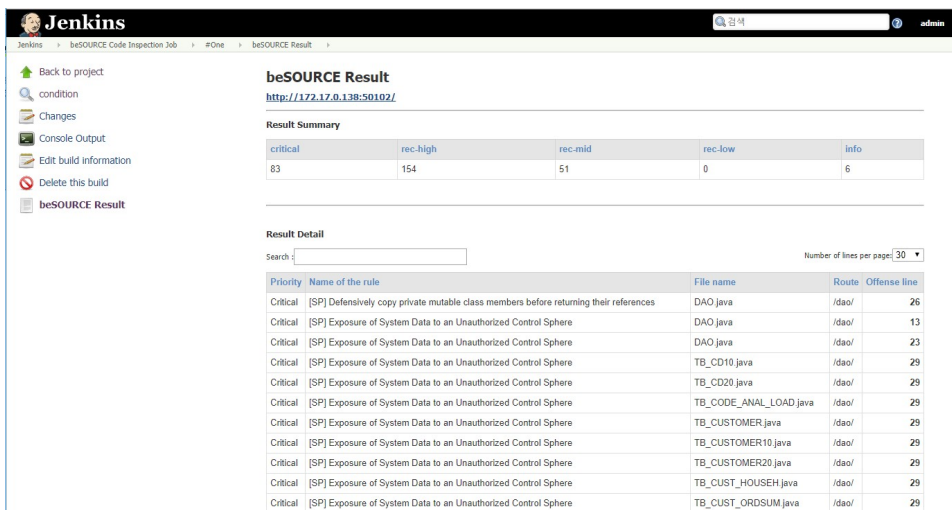
Running beSOURCE Jenkins Plug-in

You can run the plug-in by following the below steps.

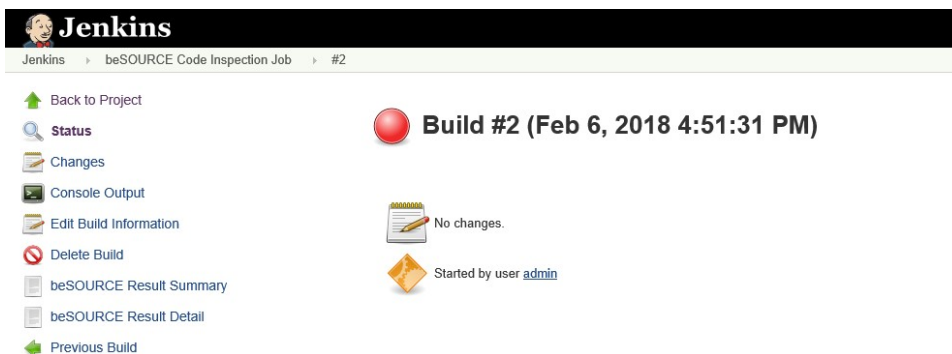
1. Select the Jenkins job to use with beSOURCE plug-in.



2. Select **Build Now**. The build number will appear. Select the number to open the build job.
3. Once the build job is finished, select the build number. The beSOURCE Result menu appears.
4. Select **beSOURCE Result** to see the analysis summary and details.



NOTE: If you select the **Stop the build task when detecting defects** option and there are some defects with a predefined priority, the build job will stop (red bulb). The red bulb does not mean the Jenkins plug-in failed, but that job was stopped due to a rule violation.



Integrating with Git

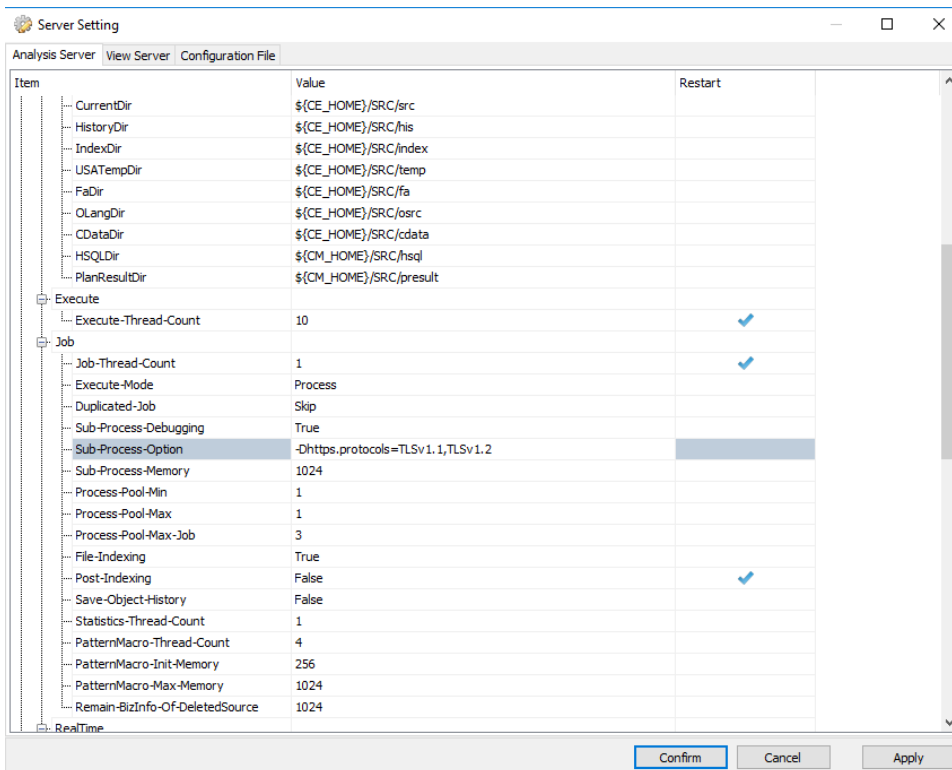
This chapter describes how to import Java sample source files from Git Hub and analyze it.

Setting for Git integration

Java version

If you installed beSOURCE Server with Java 1.7 and run it on the Java version, importing source files from Git or Git Hub would fail due to security policy of Git. A workaround for this issue is setting internal option of the Analysis Server.


1. Open and log in to the **beSOURCE Admin Console**.
2. Select **System > Server Setting**.
3. Select **Analysis Server tab > Job > Sub-Process-Option** and enter `-Dhttps.protocols=TLSv1.1,TLSv1.2`.
4. Select **Confirm**.

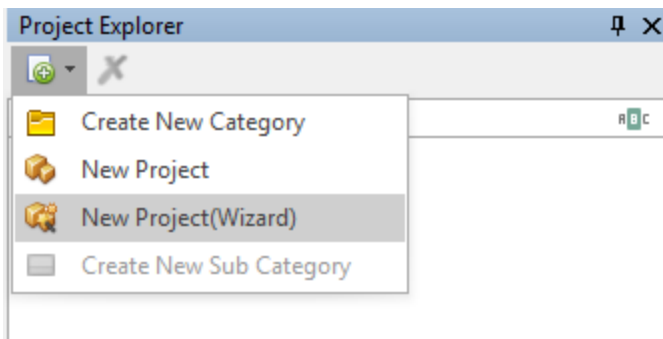


To switch to the Java version, refer to the [Appendix on page 157](#).

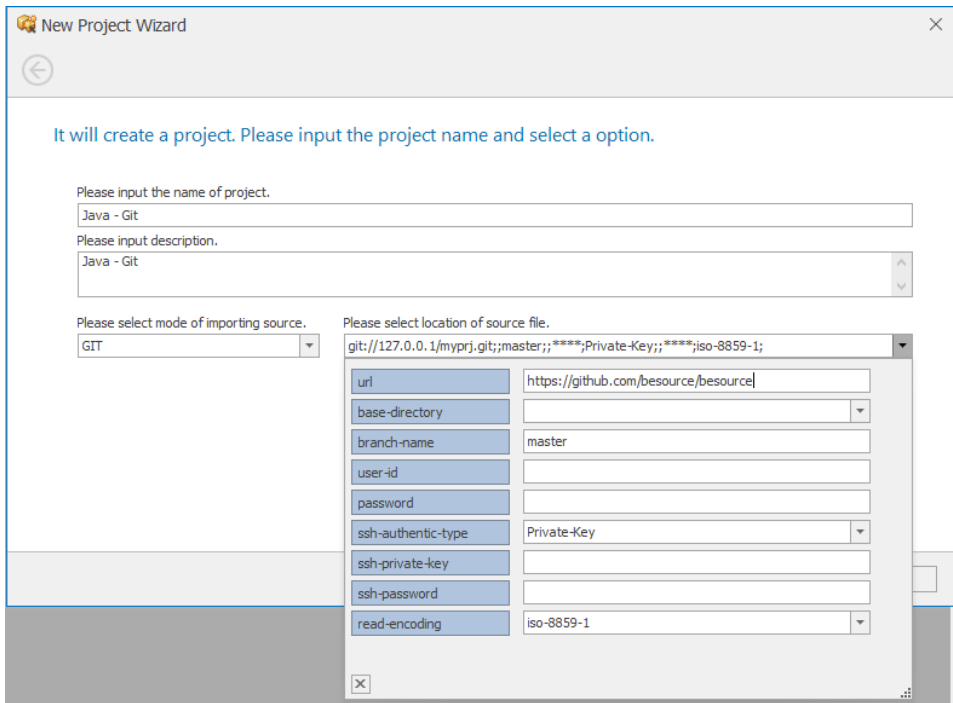
Analyzing Java source files from Git in beSOURCE Server

To analyze Java sample source files from Git Hub, do the following:

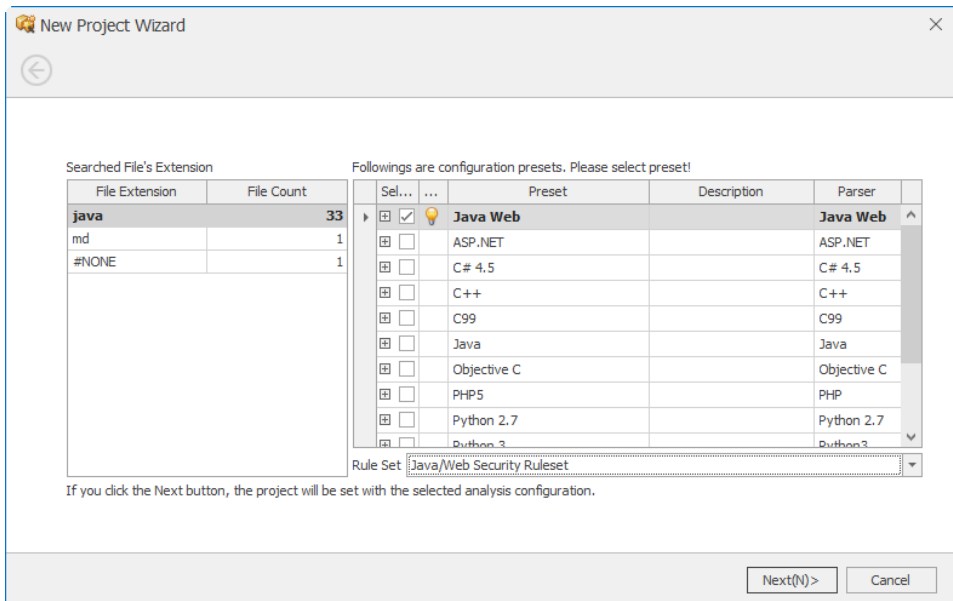
1. Open and log in to the **beSOURCE Admin Console**
2. Select **View > Project Explorer**.
3. In the **Project Explorer**, select the **Add Project Items** () icon (without selecting any project).
4. Select **New Project(Wizard)**. The Project Wizard opens.



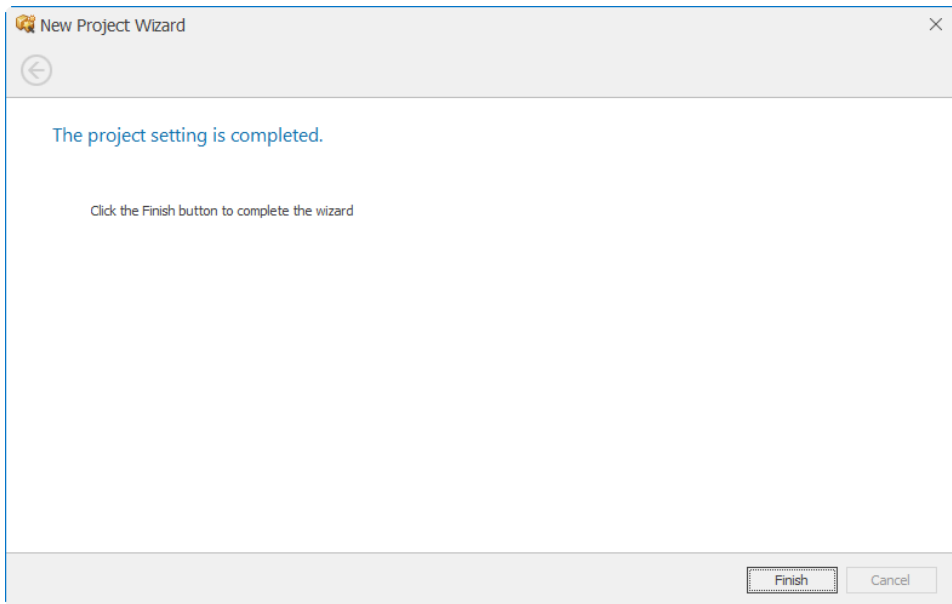
5. On the first page of the wizard, do the following:
 - a. In the **Please input the name of project** box, enter a name.
 - b. In the **Please input description** box, enter a description for the project.
 - c. In the **Please select mode of importing source** box, select **GIT**.
 - d. Select the **Please select location of source file** box, and then enter the URL and account information for your Git Hub repository.
6. Select **Next**.




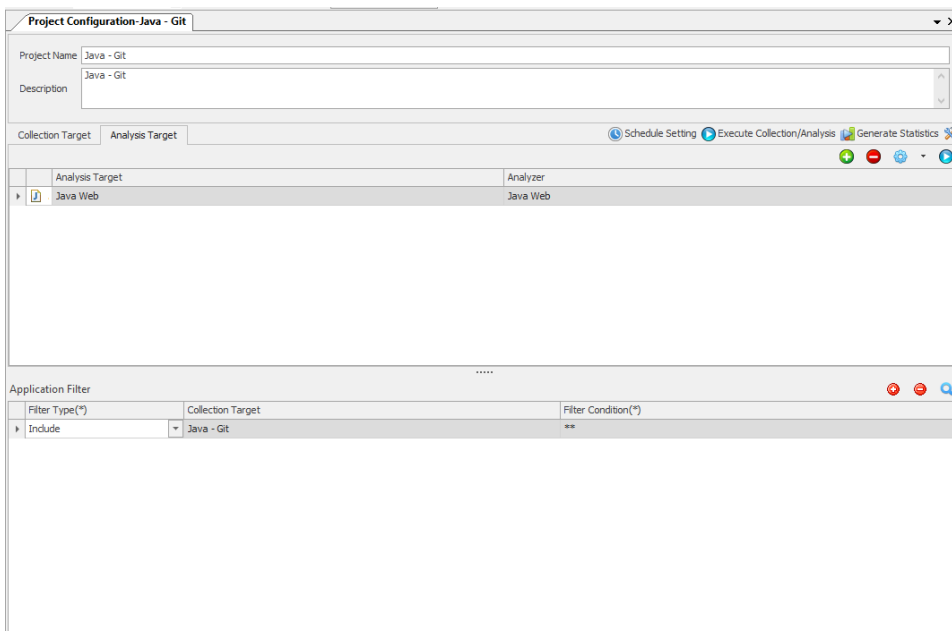
7. In the **Rule Set** box, select a ruleset.
8. Select **Next**.



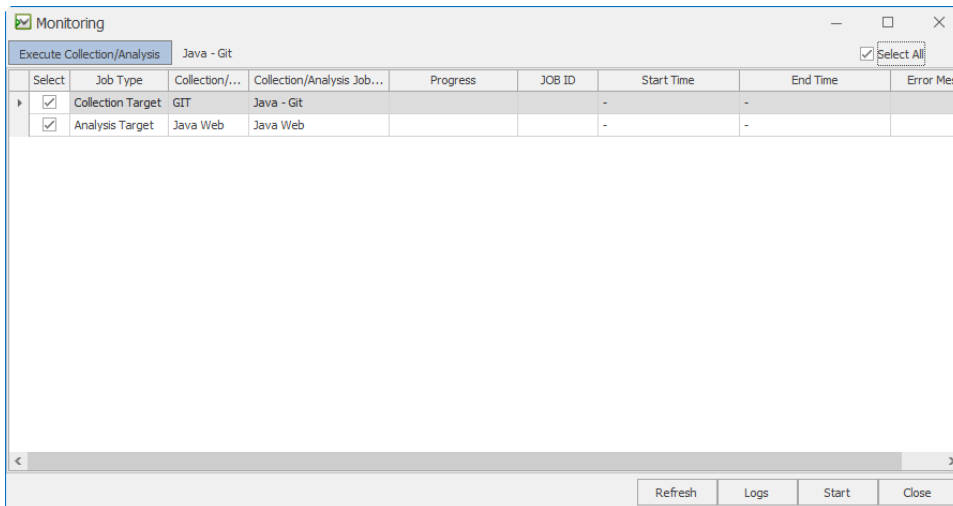
9. Select **Finish**. The project configuration window opens.



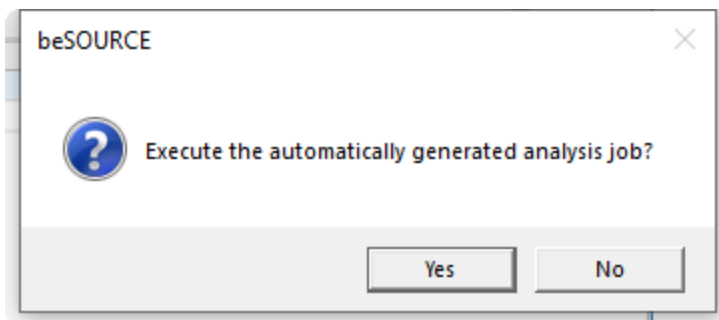
10. Select **Execute Collection/Analysis** ( **Execute Collection/Analysis**). The Monitoring window opens.



11. Select **Start**.



12. On the dialog that appears, select **Yes**.



13. The Monitoring window shows the progress of importing and analyzing source files.

NOTE:

- If you want to see detail job logs, click 'Logs' button. The Monitoring window is closed and the 'View Job Log' window opens. When you click the 'View' button in the right upper corner, the window is refreshed.
- For more information about the View Job Log window, press F1 key in it to open online help.

14. On the **All jobs are completed** dialog, select **OK**.

15. Select **Yes** in the dialog to generate default statistics.

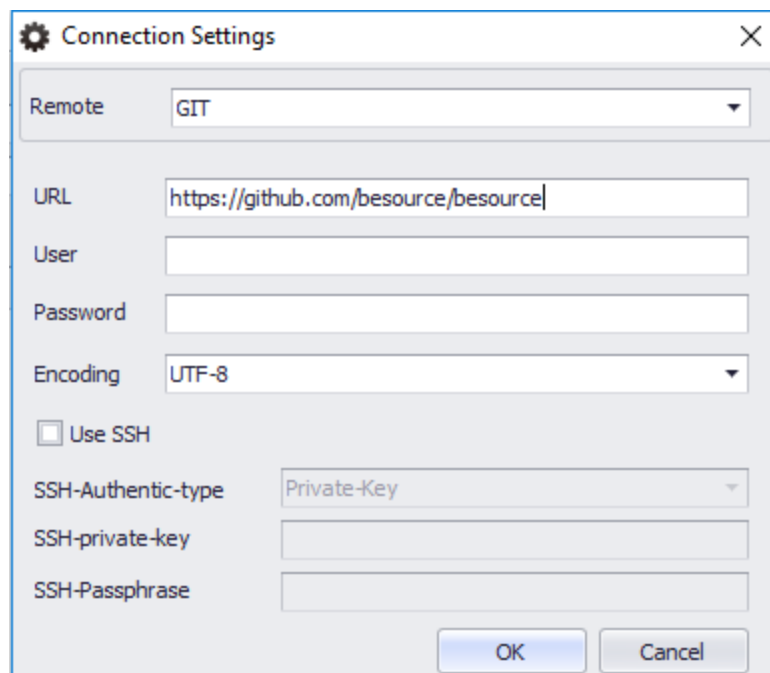
16. Select **OK** on the dialog that appears.

17. Select **Close** to close the Monitoring window.

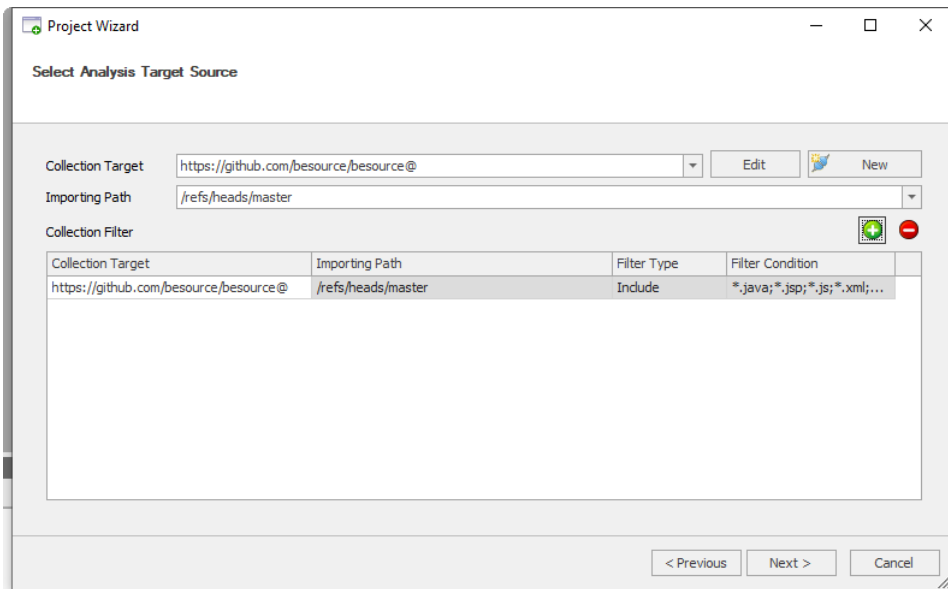
Analyzing Java source files from Git in beSOURCE Developer

To analyze Java sample source files from Git Hub with beSOURCE Developer, do the following:

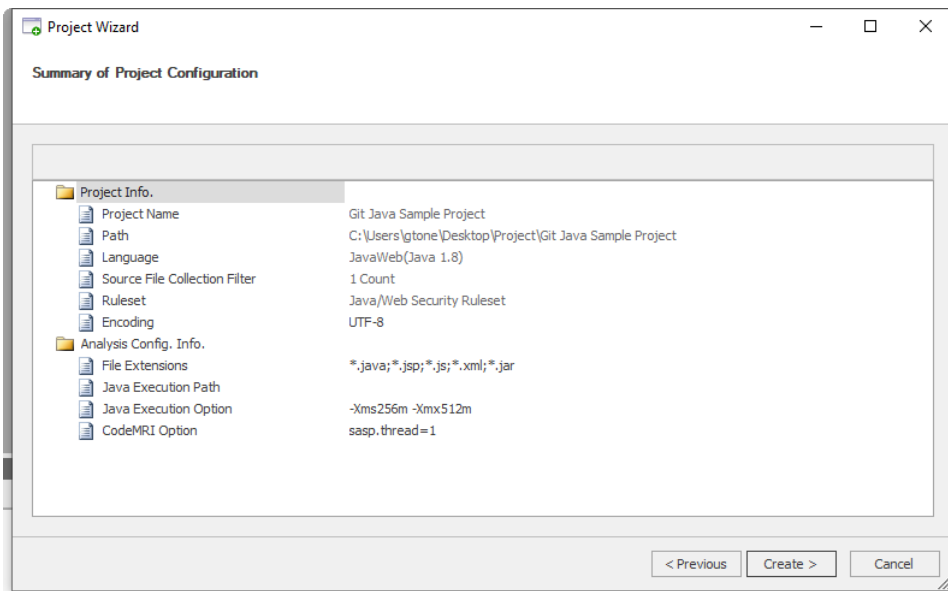
1. Open **beSOURCE Developer**.
2. Select the beSOURCE server name.
3. Enter the **User ID** and **Password** of a developer. For example, besourcedev.
4. Select **Log In**.
5. You will see your connected server information in the title bar of beSOURCE Developer.
6. Select **File > New Project**.
7. Select **Type the Project Name**.
8. Enter a name for your project.
9. In the **Language** box, select **JavaWeb(Java 1.8)**.
10. Select **Next**.
11. Select **Local Ruleset** and a ruleset.
12. Select **Next**.
13. Select **New** ( **New**).
14. Set the following option values:
 - a. **Remote** - GIT
 - b. **URL** - Your Git repository URL
 - c. If available, enter your user ID and password.



15. Select **OK**.
16. In the **Importing Path** box, select your branch.
17. Select the **Add (+)** icon.
18. Select **Next**.



19. Review the summary of your project, and then select **Create**.




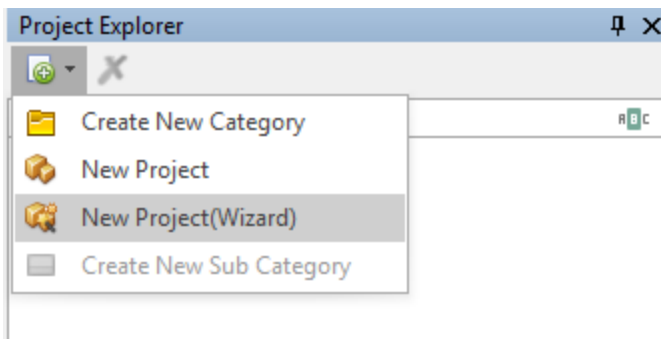
20. Select **Open the Project and Start Its Analysis**, and then select **Finish**.
21. The Console window will show the progress of the analysis.
22. On the dialog, select **OK** to open the project.
23. The **List of Rule Violations** tab shows rule violations.

Integrating with SVN

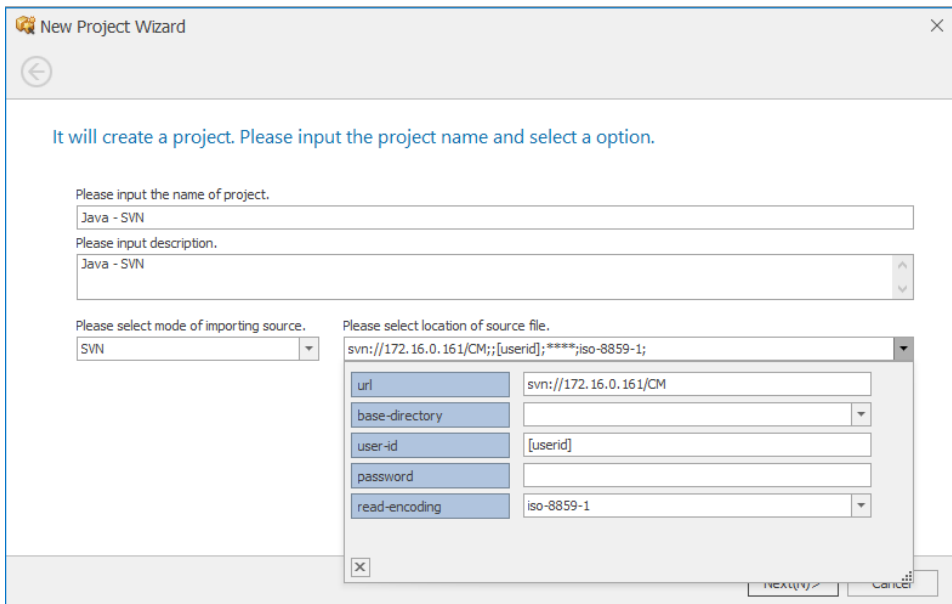
This chapter describes how to import Java sample source files from SVN and analyze it. We assume that you have a proper SVN access permission.

Analyzing source files from SVN in beSOURCE Server

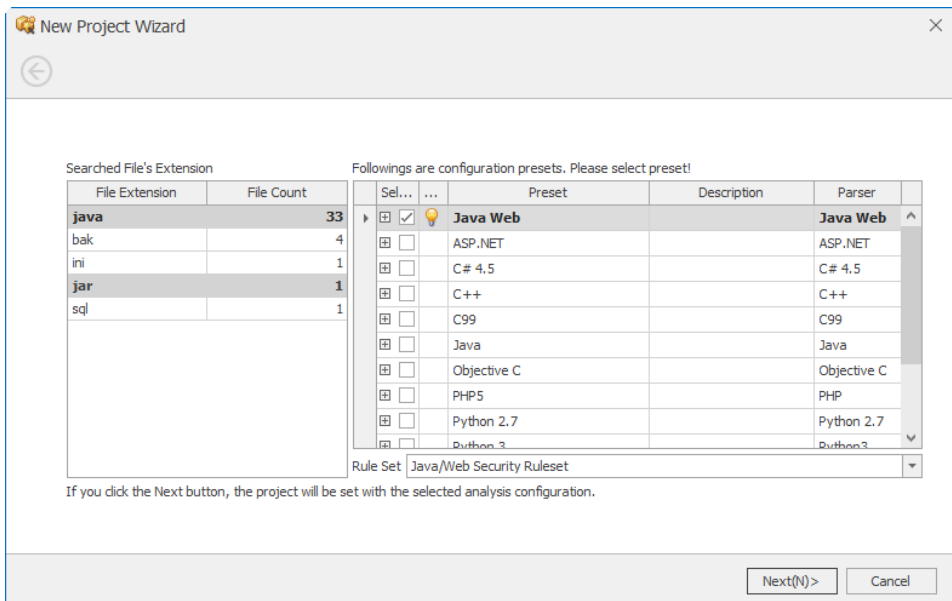
1. Open and log in to the **beSOURCE Admin Console**.
2. Select **View > Project Explorer**.
3. Select the **Add Project Items** () icon in the Project Explorer (without selecting any project).
4. Select **New Project(Wizard)**.



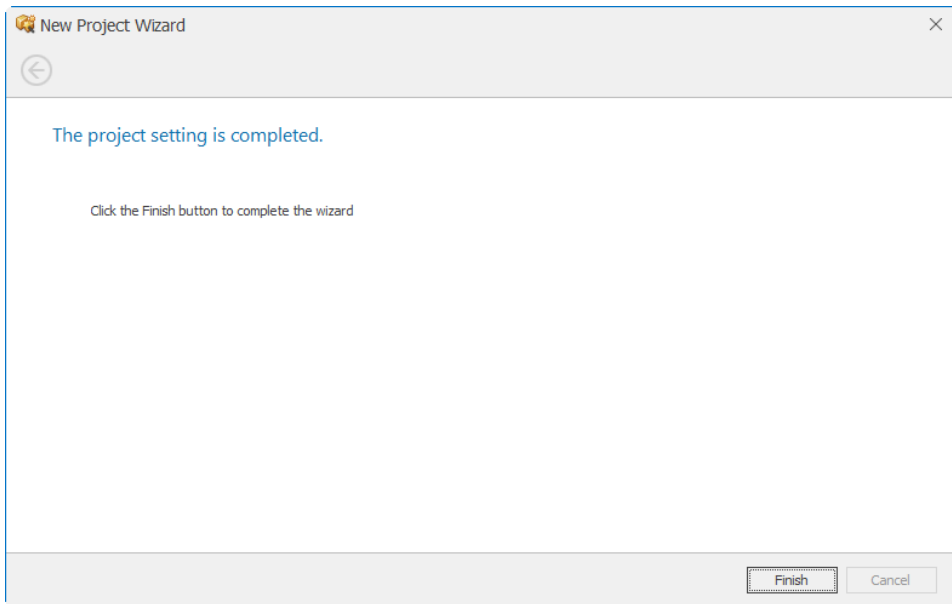
5. On the first page of the wizard, do the following:
 - a. In the **Please input the name of project** box, enter a name.
 - b. In the **Please input description** box, enter a description for the project.
 - c. In the **Please select mode of importing source** box, select **SVN**.
 - d. Select the **Please select location of source file** box, and then enter the URL and account information for your SVN repository.
6. Select **Next**.




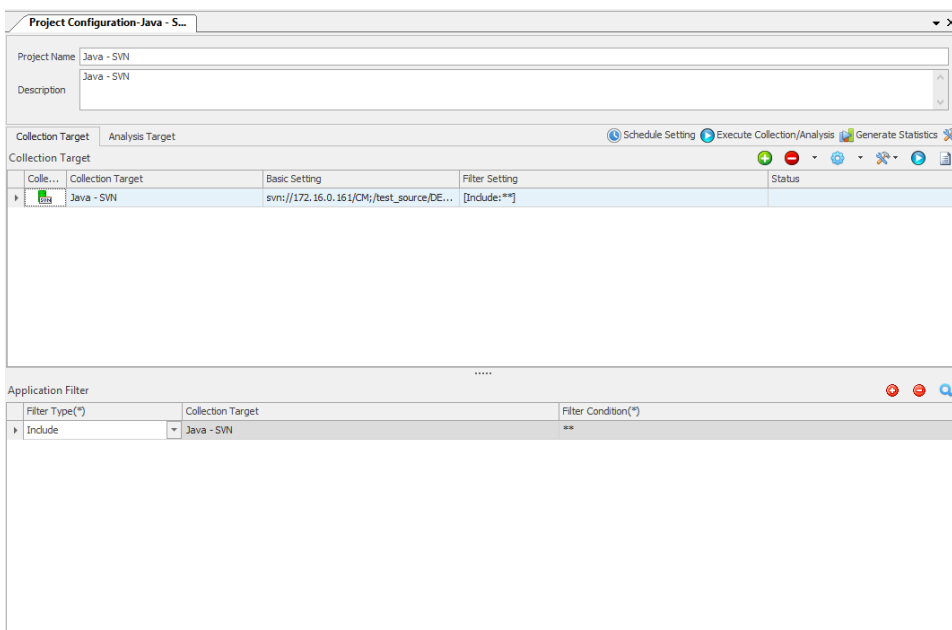
7. In the **Rule Set** box, select a ruleset.
8. Select **Next**.



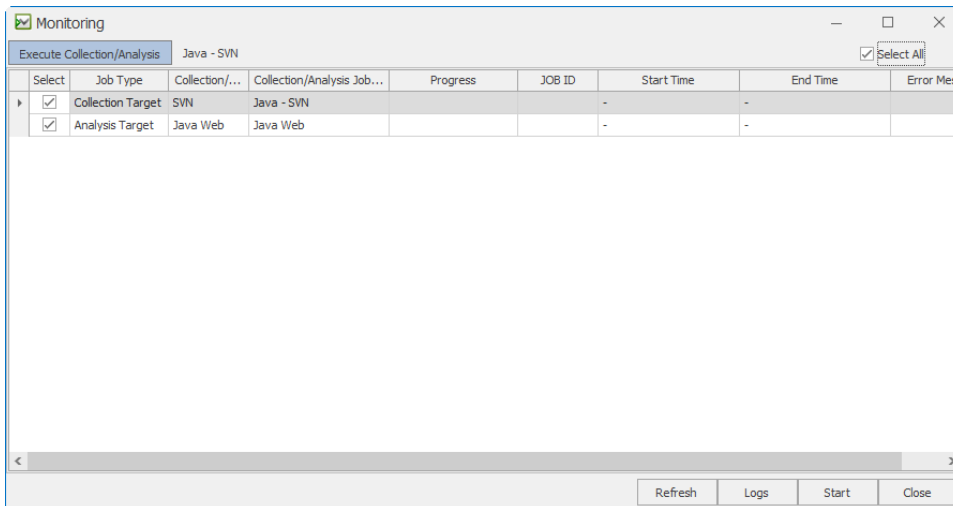
9. Select **Finish**. The project configuration window opens.



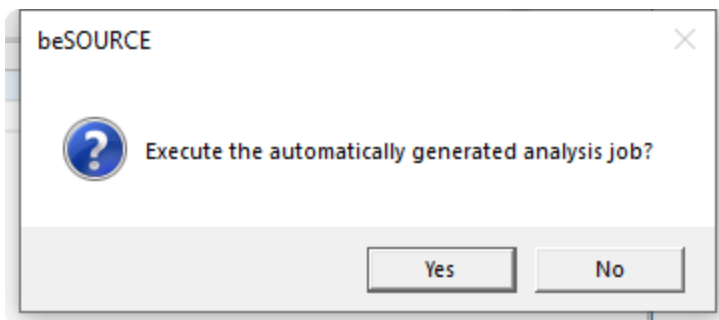
10. Select **Execute Collection/Analysis** ( **Execute Collection/Analysis**). The Monitoring window opens.



11. Select **Start**.



12. On the dialog that appears, select **Yes**.



13. The Monitoring window shows the progress of importing and analyzing source files.

NOTE:

- If you want to see detail job logs, click 'Logs' button. The Monitoring window is closed and the 'View Job Log' window opens. When you click the 'View' button in the right upper corner, the window is refreshed.
- For more information about the View Job Log window, press F1 key in it to open online help.

14. On the **All jobs are completed** dialog, select **OK**.

15. Select **Yes** in the dialog to generate default statistics.


16. Select **OK** on the dialog that appears.

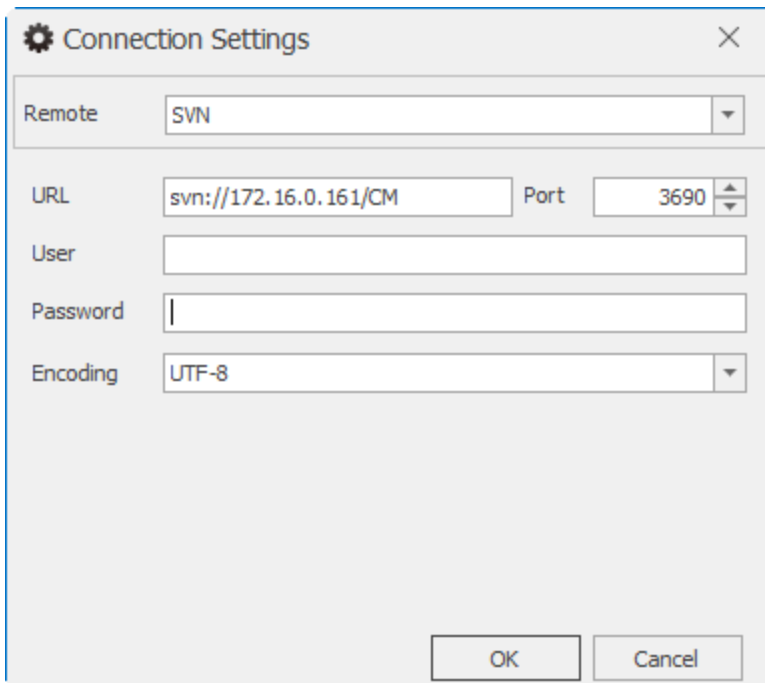
17. Select **Close** to close the Monitoring window.

Analyzing source files from SVN in beSOURCE Developer

To analyze Java sample source files from SVN with beSOURCE Developer, do the following:

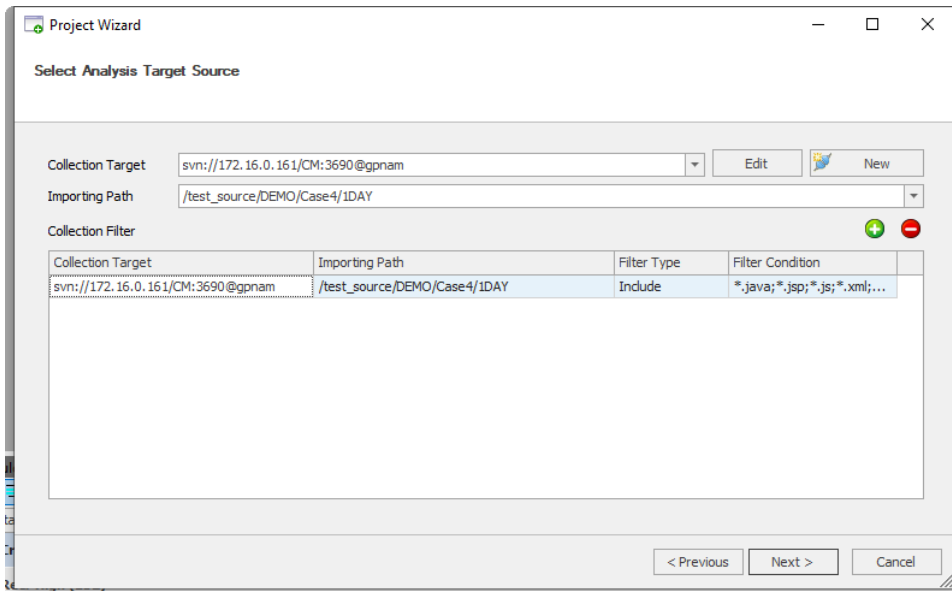
To analyze Java sample source files from Git Hub with beSOURCE Developer,

1. Open **beSOURCE Developer**.
2. Select the beSOURCE server name.
3. Enter the **User ID** and **Password** of a developer. For example, besourcedev.
4. Select **Log In**.
5. You will see your connected server information in the title bar of beSOURCE Developer.
6. Select **File > New Project**.
7. Select **Type the Project Name**.
8. Enter a name for your project.
9. In the **Language** box, select **JavaWeb(Java 1.8)**.
10. Select **Next**.
11. Select **Local Ruleset** and a ruleset.
12. Select **Next**.
13. Select **New** ( **New**).
14. Set the following option values:
 - a. **Remote** - SVN
 - b. **URL** - Your SVN repository URL
 - c. If available, enter your user ID and password.



15. Select **OK**.

16. In the **Importing Path** box, select your branch.
17. Select the **Add (+)** icon.
18. Select **Next**.




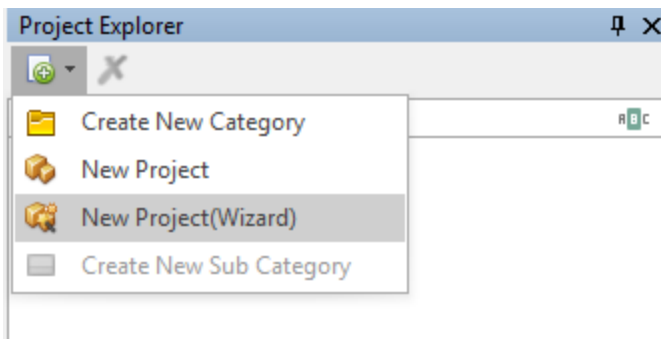
19. Review the summary of your project, and then select **Create**.
20. Select **Open the Project and Start Its Analysis**, and then select **Finish**.
21. The Console window will show the progress of the analysis.
22. On the dialog, select **OK** to open the project.
23. The **List of Rule Violations** tab shows rule violations.

Integrating with TFS

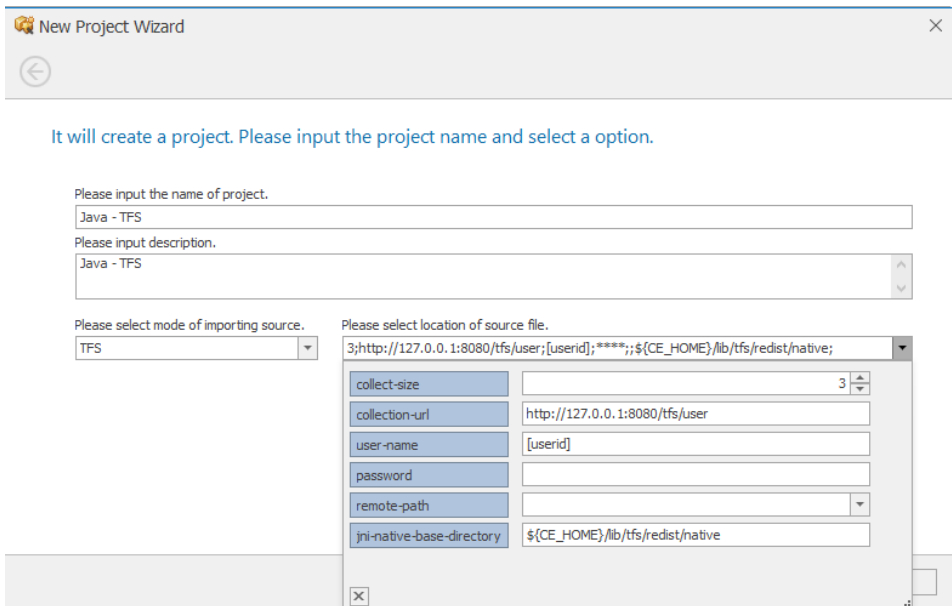
This chapter describes how to import Java sample source files from TFS and analyze it. We assume that you have a proper TFS access permission.


Analyzing source files from TFS in beSOURCE Server

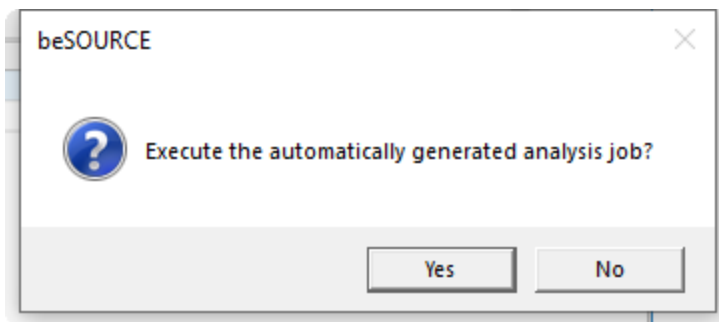
1. Open and log in to the **beSOURCE Admin Console**
2. Select **View > Project Explorer**.
3. In the **Project Explorer**, select the **Add Project Items** () icon (without selecting any project).
4. Select **New Project(Wizard)**. The Project Wizard opens.



5. On the first page of the wizard, do the following:
 - a. In the **Please input the name of project** box, enter a name.
 - b. In the **Please input description** box, enter a description for the project.
 - c. In the **Please select mode of importing source** box, select **TFS**.
 - d. Select the **Please select location of source file** box, and then enter the URL and account information for your TFS repository.
6. Select **Next**.



7. In the **Rule Set** box, select a ruleset.
8. Select **Next**.
9. Select **Finish**. The project configuration window opens.
10. Select **Execute Collection/Analysis** ( **Execute Collection/Analysis**). The Monitoring window opens.
11. Select **Start**.
12. On the dialog that appears, select **Yes**.



13. The Monitoring window shows the progress of importing and analyzing source files.

NOTE:

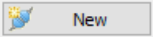
- If you want to see detail job logs, click 'Logs' button. The Monitoring window is closed and the 'View Job Log' window opens. When you click the 'View' button in the right upper corner, the window is refreshed.
- For more information about the View Job Log window, press F1 key in it to open online help.

14. On the **All jobs are completed** dialog, select **OK**.

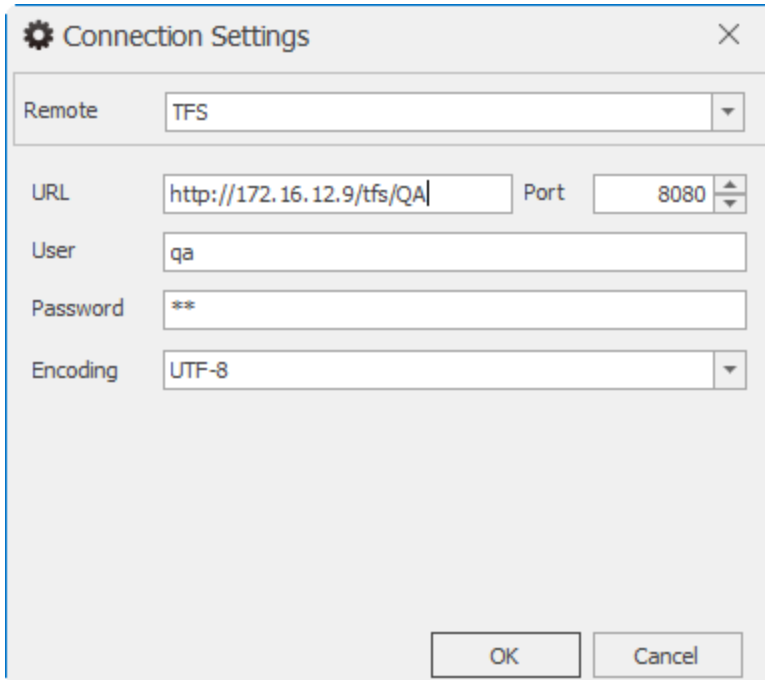
15. Select **Yes** in the dialog to generate default statistics.
16. Select **OK** on the dialog that appears.
17. Select **Close** to close the Monitoring window.

Analyzing source files from TFS in beSOURCE Developer

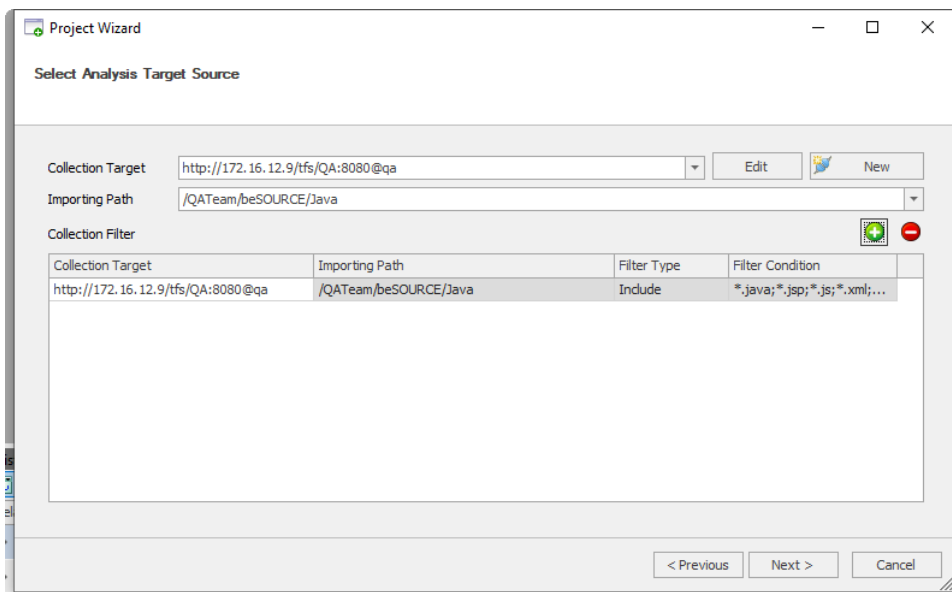
To analyze Java sample source files from TFS with beSOURCE Developer, do the following:

1. Open **beSOURCE Developer**.
2. Select the beSOURCE server name.
3. Enter the **User ID** and **Password** of a developer. For example, besourcedev.
4. Select **Log In**.
5. You will see your connected server information in the title bar of beSOURCE Developer.
6. Select **File > New Project**.
7. Select **Type the Project Name**.
8. Enter a name for your project.
9. In the **Language** box, select **JavaWeb(Java 1.8)**.
10. Select **Next**.
11. Select **Local Ruleset** and a ruleset.
12. Select **Next**.
13. Select **New** ().
14. Set the following option values:
 - a. **Remote** - TFS
 - b. **URL** - Your TFS repository URL

- c. If available, enter your user ID and password.



15. Select **OK**.
16. In the **Importing Path** box, select your branch.
17. Select the **Add (+)** icon.
18. Select **Next**.



19. Review the summary of your project, and then select **Create**.
20. Select **Open the Project and Start Its Analysis**, and then select **Finish**.
21. The Console window will show the progress of the analysis.

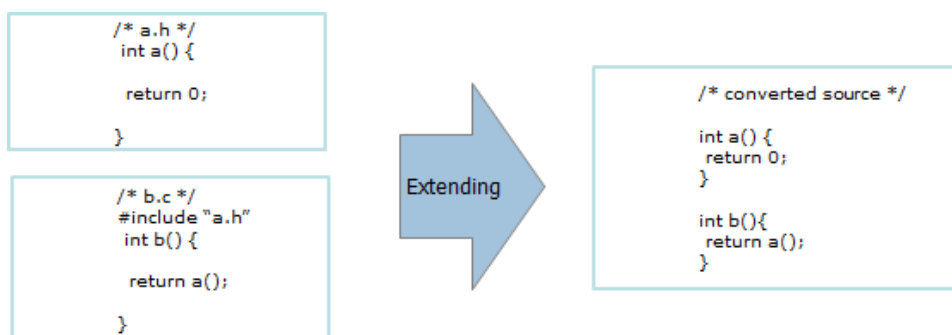
22. On the dialog, select **OK** to open the project.
23. The **List of Rule Violations** tab shows rule violations.

Using a User-defined Header

Overview

One of major characteristics of C family languages is extension of source file. Namely, the languages merge several types of source files such as `.c` and `.h` to one file then compile it to produce executable file. The main pre-processing macro for source extension is `#include`.

Whenever the language parser meets `#include` macro in source files during pre-processing, it merges the original source file and target `include` file. Therefore, if there are some missing header files, parsing errors will occur.

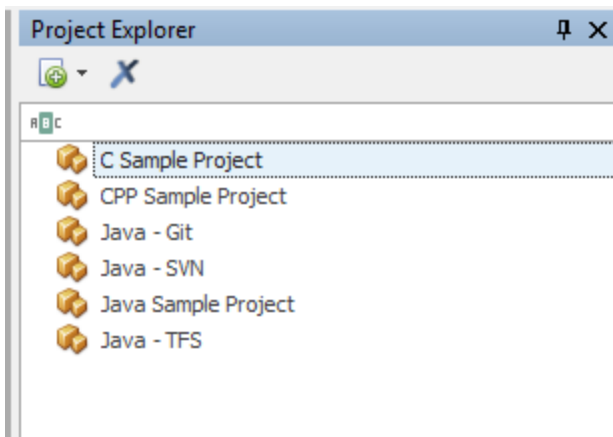


The beSOURCE parsers for C, C++ and Objective C generate `conv2` files after extending source files like the above explanation and `err` files for internal processing errors. The temporary files are located in `Server_Install_Directory\analyzer\SRC\temp\Analysis_Number` directory and the `conv2` file has a naming convention of `Original_Source_Name + postfix + .conv2`.

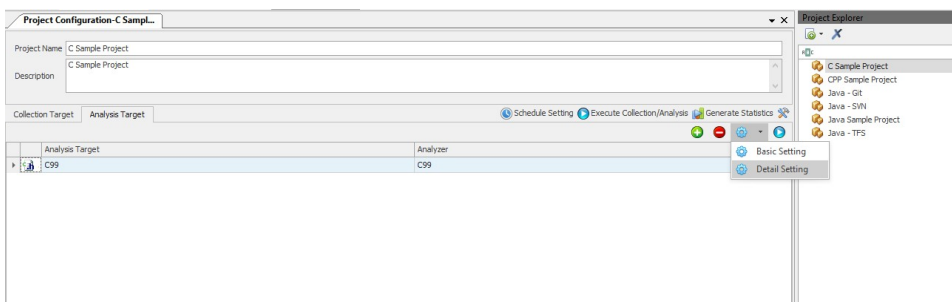
NOTE: Since a `conv2` file is a temporary file, it is automatically deleted after internal use. Optionally, you can keep the file for debugging by setting the **Parser-Options > auto-delete-temp-files** option to **False**.

To keep the temporary file, do the following:

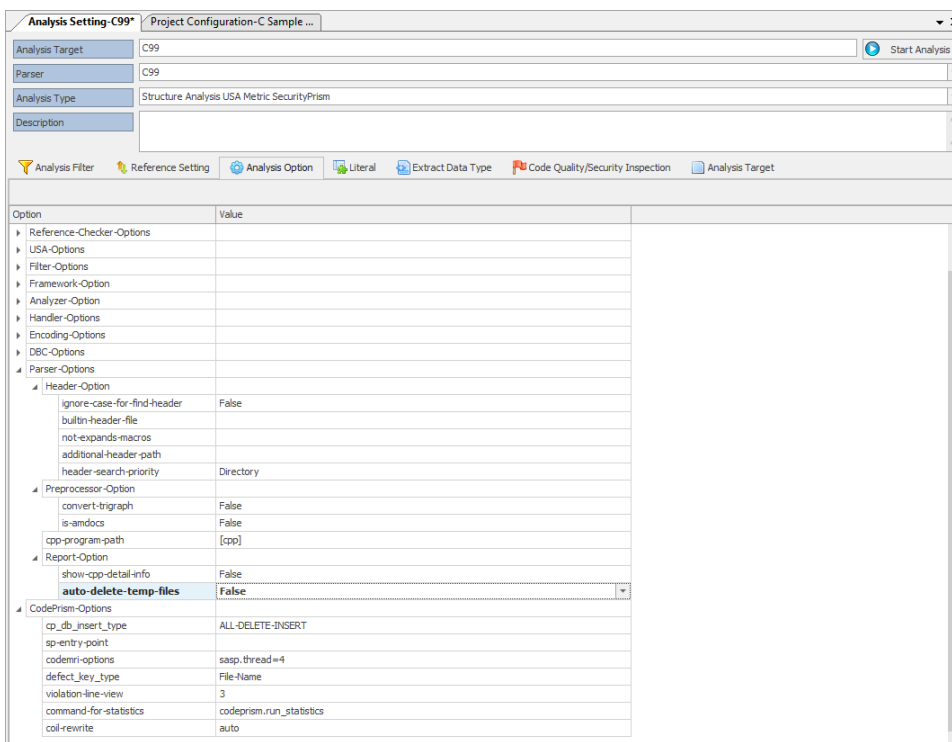
1. Open and log in to the **beSOURCE Admin Console**.
2. Open **Project Explorer**.
3. Double-click **C Sample Project** or **CPP Sample Project**.



4. Open the **Analysis Target** tab.
5. Select the **gear icon**, and then select **Detail Setting**.



6. Open the **Analysis Option** tab.
7. Set **auto-delete-temp-files** to **False**.



8. Select **Save**.

If you happen to have only C or C++ source files without header files or if you have missing header files in your project, you can try analyzing them with the User-defined Header functionality. If the analysis target project has full header information, the above function would not be necessary.

It is useful for relatively clear situations as follows:

- Missing macro declarations
- Missing type definitions

Finding a parsing error in case of missing macro and fixing it

You will try to analyze 'sample1.cpp' file to see how to find a parsing error for missing macro and fix it. For your exercise, copy the below source codes and make 'sample1.cpp' source file (without line numbers). These steps assume that the file is located in the beSOURCE Server Installation\Sample\UDH directory.


sample.cpp

```
#define LPVOID void*
class SAMPLE_Processor {

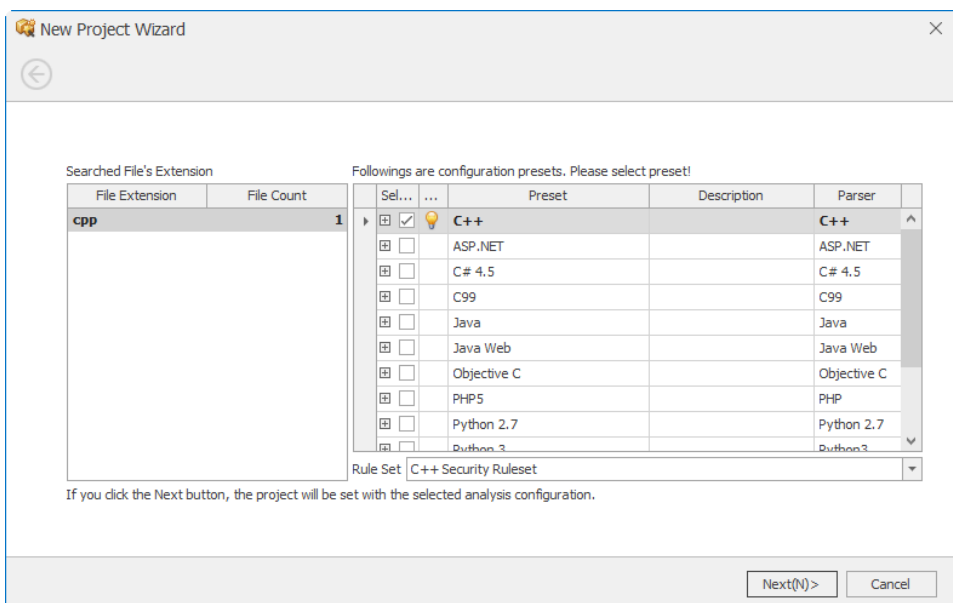
public :
void Run( IN LPVOID parameter );
};
void SAMPLE_Processor::Run( IN LPVOID parameter ) {
//.....
};
```

Analyzing C++ sample source files

To analyze C++ sample source files,

1. Open and log in to the **beSOURCE Admin Console**.
2. Select **View > Project Explorer**.
3. Select the **Add Project Items** () icon in the **Project Explorer** (without selecting any project).
4. Select **New Project(Wizard)**. The Project Wizard opens.

5. On the first page of the wizard, do the following:
 - a. In the **Please input the name of project** box, enter a name.
 - b. In the **Please input description** box, enter a description for the project.
 - c. In the **Please select mode of importing source** box, select **Local Copy**.
 - d. Select the **Please select location of source file** box, select the **base-directory** box, select the beSOURCE Server (Enterprise Edition) installation folder (for example, C:\beSOURCE\), double-click the **sample\UDH** folder, and then select the source files.
6. Select **Next**.
7. The Project Wizard scans the source files and recommends a preset configuration. Select **C++**.
8. Select **Next**.



9. Select **Finish**.
10. The project configuration window opens.
11. Select **Execute Collection/Analysis** ().
12. The Monitoring window opens.
13. Select **Start**.
14. Select **Yes**.
18. The Monitoring window shows the progress of importing and analyzing source files. On the dialog that appears, select **OK**.
19. Select **Yes** in the dialog to generate default statistics.
20. On the dialog that appears, select **OK**.
21. Select **Close** to close the Monitoring window.

Checking the scanning result

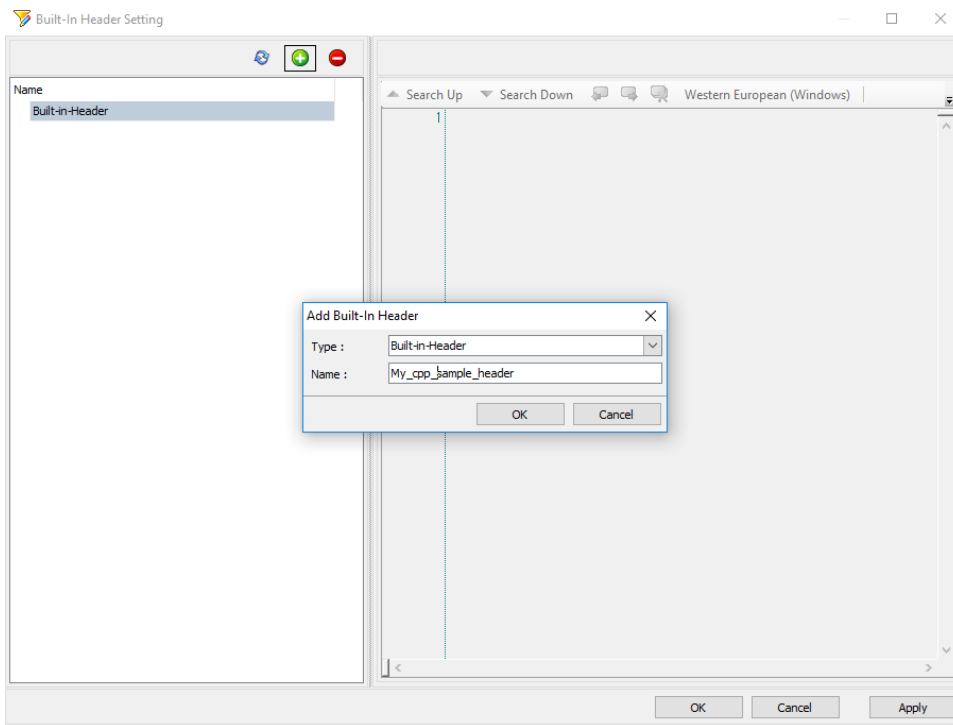
To check the scanning result (parsing error) in the analysis engine and fix it, do the following:

1. Select **System > View Job Log**.
2. Double-click the red column, which shows that there was problem during the analysis.
3. From the **Detail List** tab, select the **Detail Error Messages** tab.
4. Select the `sample1.cpp` file in the left list.

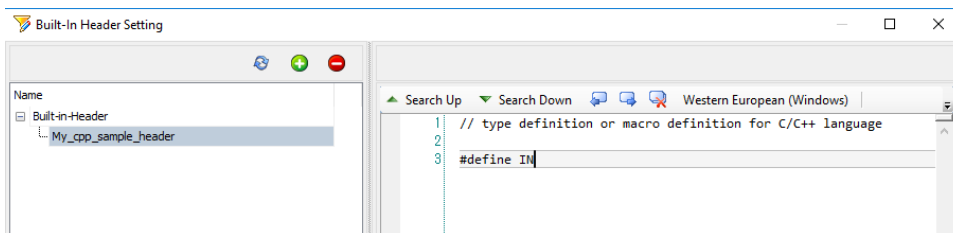
NOTE: The message shows that "void" in the line number 7 has caused the parsing error. You can also find a type qualifier, "IN" within the `Run()` function declaration. It seems not a general C++ expression and you do not know the exact meaning due to missing header information.

The screenshot shows the beSOURCE Admin Console interface. The main window displays a 'View Job Log' for a project named 'My Sample1'. A table shows the status of various jobs, with one job marked as 'Completed' but with a red column indicating a parsing error. The 'Detail List' tab is active, showing the error details for the file 'sample1.cpp'. The error message is: 'com.gtone.cm.parser.cpp.exception.CPPExtractorException: 5123 -2102 : Parsing error: src line :C:/beSOURCEV5/analyser/SRC/temp/20191008000113/sample1.cpp:7 original src : void public : void Run(IN LPVOID parameter); converted src : public : void Run(IN void* parameter); token str : TOKEN MARKED SRC TEXT ::: null'. The error message indicates a parsing error in the original source code, specifically in the function declaration 'void Run(IN LPVOID parameter);', where the type qualifier 'IN' is not recognized.

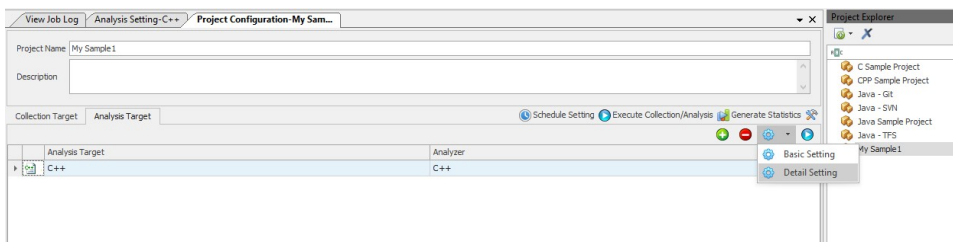
5. Select **Analysis > Built-in Header Setting**.
6. Select **Add (+)**.
7. Enter your built-in header name (for example, `My_cpp_sample_header`).
8. Select **OK**.



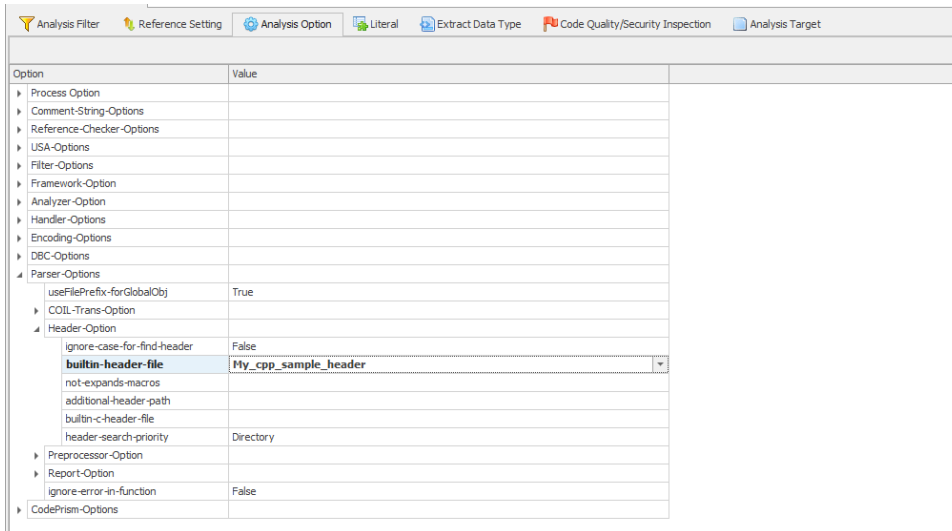
9. Select the new built-in header name in the left list.
10. Enter `#define IN` in the editor. This will prevent the parsing error due to missing header information.
11. Select **OK**.





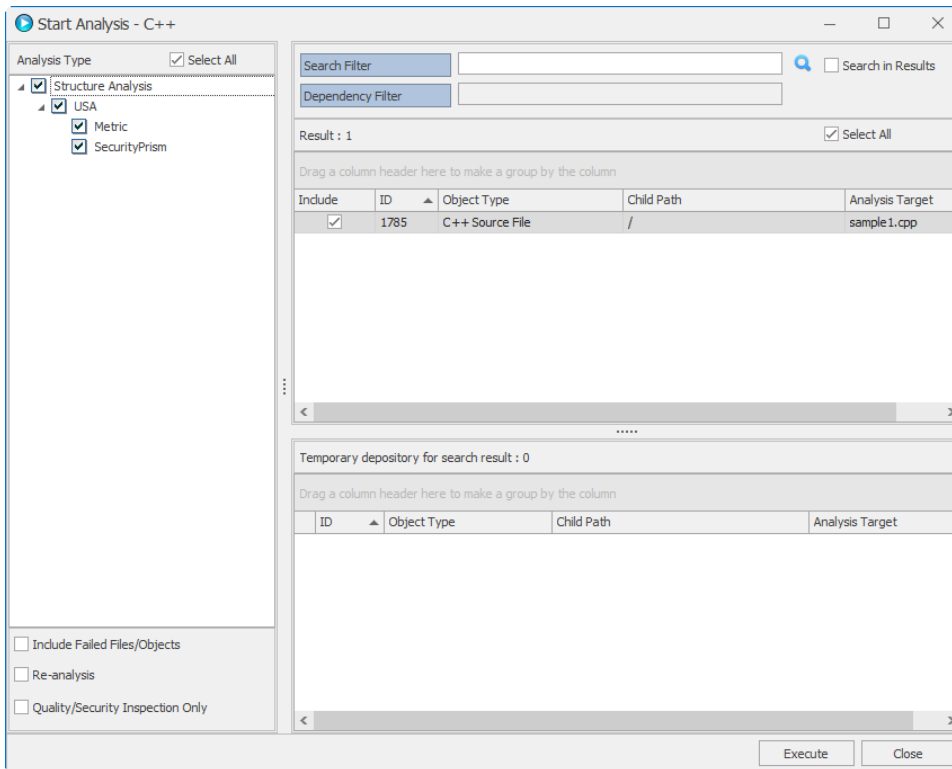
12. On the dialog that appears, select **OK**.
13. Open the **Project Explorer**.
14. Double-click your project name, My sample1.
15. Select the **Analysis Target** tab.
16. Select the gear icon, and then select **Detail Setting**.



- Open the **Analysis Option** tab and assign your built-in header to include a header-file option.



- Select **Save tool**.
- Select **Start Analysis** ( Start Analysis).
- Select **Search** ().
- Select the **Select All** option, or check the `sample1.cpp` file.
- Select **Execute**.
- Select **OK**.



24. Open the **View Job Log** window.
25. Select **View**. You will find that there is no parsing error.

NOTE: The sample source file is just for showing fixing parsing errors. There would be no meaningful security defects in the result.

Collection				Analysis				Statistics/Mart											
Status	Type	Job Target	Message	Structure Analysis			USA			Metric			SecurityPrism						
				All	Success	Fail	All	Success	Fail	All	Success	Fail	All	Success	Fail				
Completed	Manual Job	C++		1	1	0	1	1	0	1	1	0	1	1	0				
Completed	Manual Job	C++		1	0	1	0	0	0	0	0	0	0	0	0				
Completed	Manual Job	Java Web		33	33	0	79	79	0	33	33	0	33	33	0				
Completed	Manual Job	Java Web		282	282	0	358	358	0	282	282	0	282	282	0				

Finding a parsing error in case of missing class declaration and fixing it

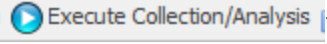
You will try to analyze `sample2.cpp` file to see how to find a parsing error for missing class declaration and fix it. For your exercise, copy the below source codes and make `sample2.cpp` source file (without line numbers). These steps assume that the file is located in `beSOURCE Server Installation\Sample\UDH` directory.

sample2.cpp

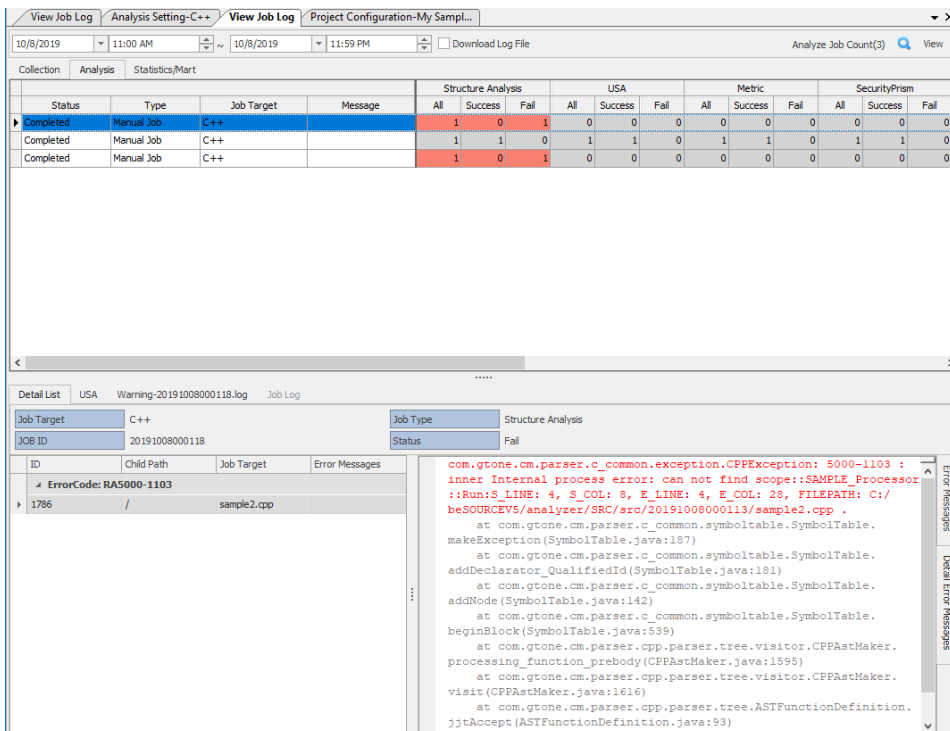
```
#define LPVOID void*
void SAMPLE_Processor::Run( LPVOID parameter )
```



```
{//.....};
```

1. Open **Project Explorer**.
2. Double-click **My Sample1** project
3. Select **Execute Collection/Analysis** (.
4. On the Monitoring window select **Start**.and then **Logs**. beSOURCE Server detects the new added **sample2.cpp** file and automatically imports and analyzes it. The **View Job Log** window opens.
5. Select the **Analysis** tab.
6. Double-click the red column, which shows that there was problem during the analysis.
7. From the **Detail List** tab, select the **Detail Error Messages** tab.
8. Select the **sample2.cpp** file in the left list.

NOTE: The message shows that 'SAMPLE_Processor::Run' in the line number 4 has caused the parsing error because it could not find a proper scope. This kind of scope error is occurred due to missing declaration of the function's parents (namespace or class). In principle, you should find the original header that is missing and analyze it together. However, if you cannot find the missing header, you can try a limited analysis by defining the user-defined header with the Forward Declaration characteristic.

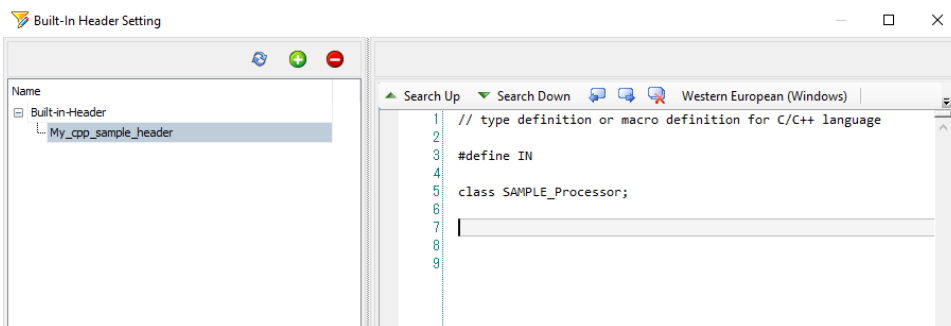



The screenshot displays the beSOURCE IDE's Job Log window. The top section shows a table with columns for Status, Type, Job Target, Message, and various analysis metrics (Structure Analysis, USA, Metric, SecurityPrism). The 'Structure Analysis' column for the 'sample2.cpp' job is highlighted in red, indicating a failure.

Below the table, the 'Detail List' tab is active, showing the 'Warning-20191008000118.log' file. The 'Detail Error Messages' tab is selected, displaying the following error message:

```
com.gtone.cm.parser.c_common.exception.CPPEException: 5000-1103 :
inner Internal process error: can not find scope::SAMPLE_Processor
::Run:S_LINE: 4, S_COL: 8, E_LINE: 4, E_COL: 28, FILEPATH: C:/
beSOURCEV5/analyzer/SRC/src/20191008000113/sample2.cpp .
at com.gtone.cm.parser.c_common.symboltable.SymbolTable.
makeException(SymbolTable.java:187)
at com.gtone.cm.parser.c_common.symboltable.SymbolTable.
addDeclarator_QualifiedId(SymbolTable.java:181)
at com.gtone.cm.parser.c_common.symboltable.SymbolTable.
addNode(SymbolTable.java:142)
at com.gtone.cm.parser.c_common.symboltable.SymbolTable.
beginBlock(SymbolTable.java:539)
at com.gtone.cm.parser.cpp.parser.tree.visitor.CPPAstMaker.
processing_function_prebody(CPPAstMaker.java:1595)
at com.gtone.cm.parser.cpp.parser.tree.visitor.CPPAstMaker.
visit(CPPAstMaker.java:1616)
at com.gtone.cm.parser.cpp.parser.tree.ASTFunctionDefinition.
jttAccept(ASTFunctionDefinition.java:93)
```

9. Select **Analysis > Built-in Header Setting**.
10. Click the new built-in header name in the left list.
11. Enter class **SAMPLE_Processor**; in the editor.
12. Select **OK**.



13. On the dialog that appears, select **OK**.
14. Open the **Project Configuration** window for the **My Sample1** project.
15. Select **Execute Collection/Analysis** ().
16. On the Monitoring window, select **Start**.
17. Wait a few minutes and then select **Logs**.
18. The **View Job Log** window opens. You will find that there is no parsing error.

NOTE: The sample source file is just for demonstrating the correction of parsing errors. There will be no meaningful security defects in the result.


Finding a parsing error in case of missing function declaration and fixing it

You will try to analyze 'sample3.cpp' file to see how to find a parsing error for missing macro and fix it. For your exercise, copy the below source codes and make 'sample3.cpp' source file (without line numbers). This document assumes that the file is located in 'beSOURCE Server Installation\Sample\UDH' directory.

sample3.cpp

```
void Run() {
RawImageManager ImageManager;
ImageManager.AddText (
strVMSID,
pADOQuery->FieldByName( "WIDTH" )->AsInteger, pADOQuery->
FieldByName( "HEIGHT" )->AsInteger, pADOQuery->FieldByName(
"FONTKIND" )->AsInteger, pADOQuery->FieldByName( "FONTCOLOR" )->
AsInteger, pADOQuery->FieldByName( "FONTSIZE" )->AsInteger,
pADOQuery->FieldByName( "FONTALIGN" )->AsInteger, pADOQuery->
```

```
>FieldByName( "DATAS" )->AsString );  
FOR_STL( Generator *, pGenerator, ImageManager.m_  
listImageGenerator ) {  
//..  
}
```

1. Open **Project Explorer**.
2. Double-click **My Sample1** project
3. Select **Execute Collection/Analysis** (.
4. On the Monitoring window select **Start**.
5. Wait a few minutes and then select **Logs**. beSOURCE Server detects the new added **sample3.cpp** file and automatically imports and analyzes it. The **View Job Log** window opens.
6. Select the **Analysis** tab.
7. Double-click the red column, which shows that there was problem during the analysis.
8. From the **Detail List** tab, select the **Detail Error Messages** tab.
9. Select the **sample3.cpp** file in the left list.

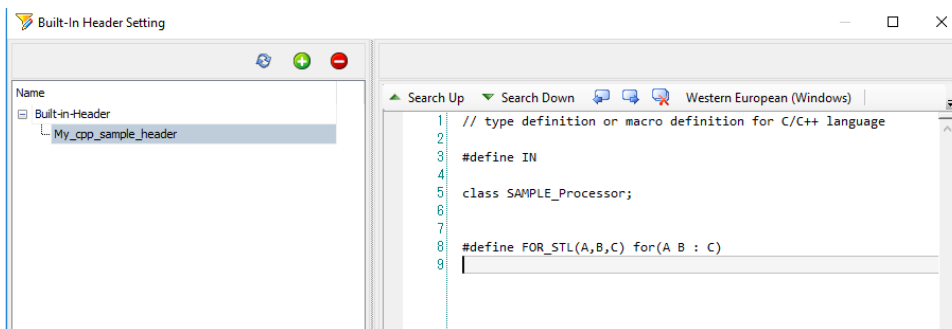
NOTE:


- Developers would write codes to make different statements depending on the C ++ version through macro declarations to avoid problems with C ++ version compatibility. If the macro declaration file is missing in this case, a parsing error will occur.
- The above parsing error occurred when '*' was encountered on line 15. The parser attempted grammar analysis for 'FOR_STR', assuming that it is a function call but the type is unknown. Semantically, the 'FOR_STL' can be inferred as a FOR statement, and the macro declaration can be defined in a custom header as shown below.

Collection				Analysis				Statistics/Mart								
Status	Type	Job Target	Message	Structure Analysis			USA			Metric			SecurityPrism			
				All	Success	Fail	All	Success	Fail	All	Success	Fail	All	Success	Fail	
Completed	Manual Job	C++		1	0	1	0	0	0	0	0	0	0	0	0	0
Completed	Manual Job	C++		1	1	0	1	1	0	1	1	0	1	1	0	
Completed	Manual Job	C++		1	0	1	0	0	0	0	0	0	0	0	0	
Completed	Manual Job	C++		1	1	0	1	1	0	1	1	0	1	1	0	
Completed	Manual Job	C++		1	0	1	0	0	0	0	0	0	0	0	0	
Completed	Manual Job	Java Web		33	33	0	79	79	0	33	33	0	33	33	0	
Completed	Manual Job	Java Web		282	282	0	358	358	0	282	282	0	282	282	0	
Completed	Scheduler Job	Java Web														
Completed	Scheduler Job	C++														
Completed	Scheduler Job	Java Web														
Completed	Scheduler Job	Java Web														
Completed	Scheduler Job	C99														

Detail List				USA		Warning-20191008000127.log		Job Log	
Job Target	C++	Job Type	Structure Analysis	Status	Fail				
Job ID	20191008000127	Status	Fail						
ID	Child Path	Job Target	Error Messages						
ErrorCodes: RAS123-2102									
1787	/	sample3.cpp	<pre> com.gtone.cm.parser.cpp.exception.CPPExtractorException: 5123 -2102 : Parsing error: src line :C:/beSOURCEVS/analyser/SRC/src/20191008000113/sample3. cpp:12 conv line :C:/beSOURCEVS/analyser/SRC/temp/20191008000112/sample3. cpp_4108972246331424438.conv2:23 error:parse statement : SimpleDeclaration , encounter : FOR_STL original src : pADOQuery->FieldByName("FONTALIGN")-> AsInteger, pADOQuery->FieldByName("DATAS")->AsString) ; converted src : token str : TOKEN MARKED SRC TEXT ::: null </pre>						

10. Select **Analysis > Built-in Header Setting**.
11. Click the new built-in header name in the left list.
12. Enter class `#define FOR_STL(A,B,C) for(A B : C)` in the editor.
13. Select **OK**.



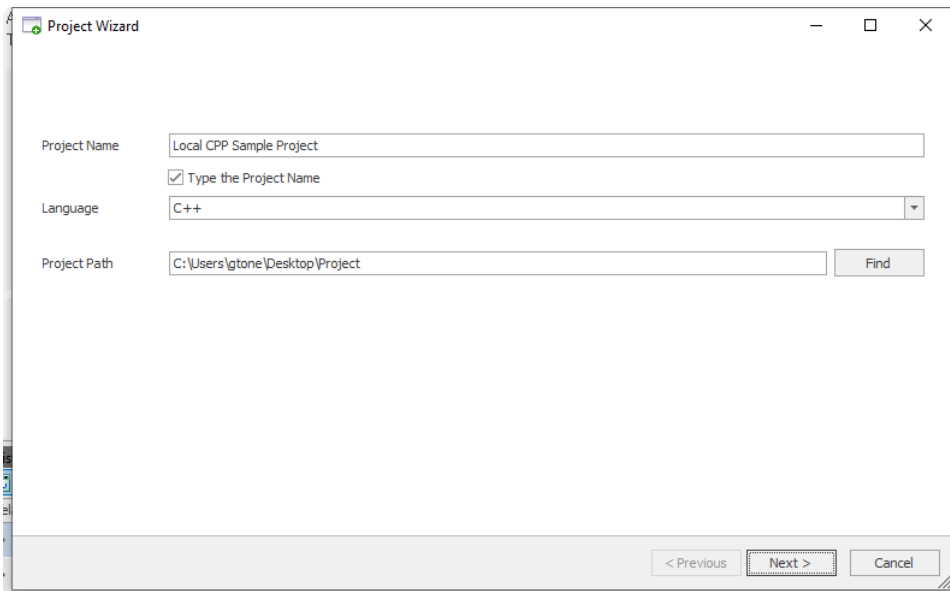
14. On the dialog that appears, select **OK**.
15. Open the **Project Configuration** window for the **My Sample1** project.
16. Select **Execute Collection/Analysis** ().
17. On the Monitoring window, select **Start**.
18. Wait a few minutes and then select **Logs**.
19. The **View Job Log** window opens. You will find that there is no parsing error.

NOTE: The sample source file is just for showing fixing parsing errors. There would be no meaningful security defects in the result.

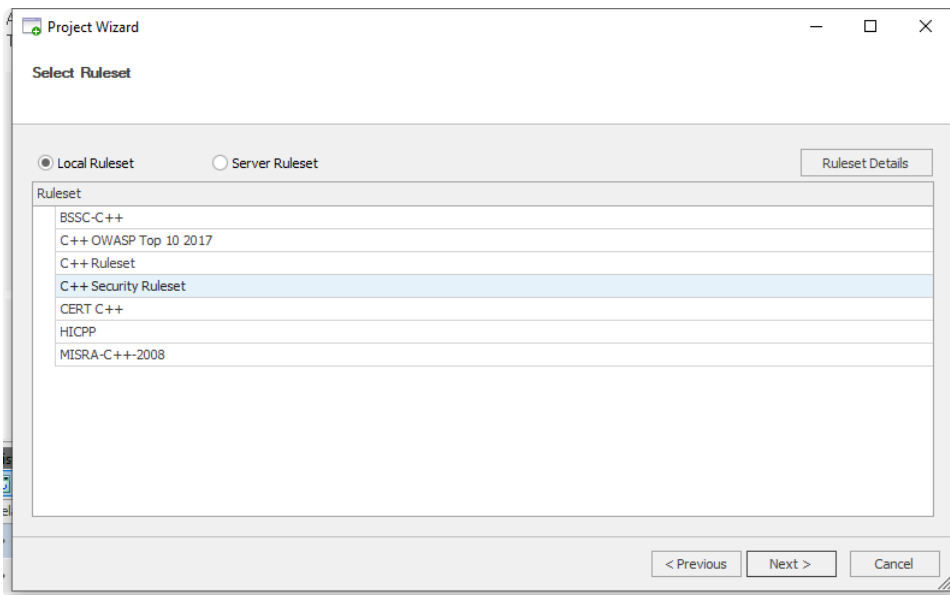
Setting user defined header in beSOURCE Developer

You can also define the custom header in beSOURCE Developer.

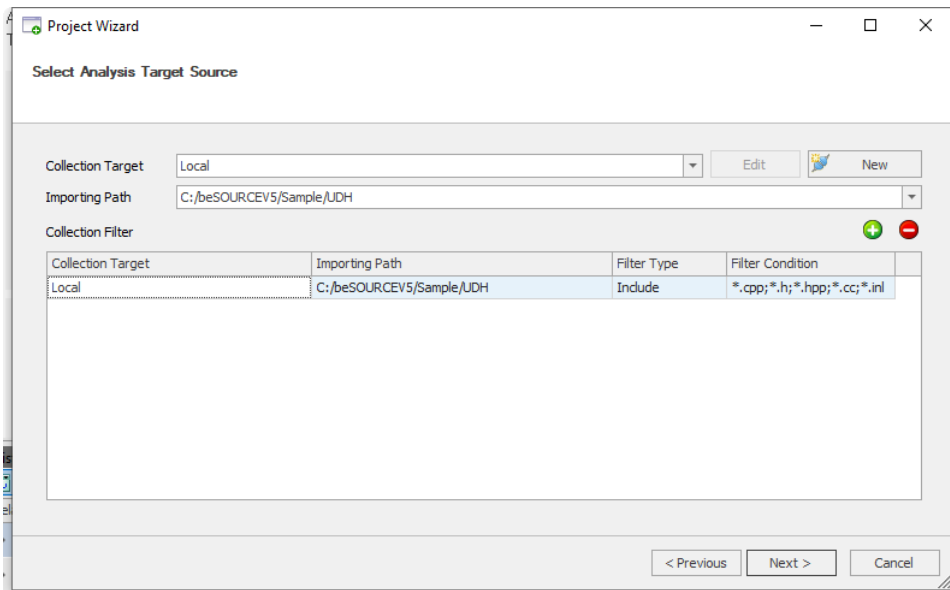
1. Open **beSOURCE Developer**.
2. Select your beSOURCE Server.
3. Enter the **User ID** and **Password** of a developer. For example, besourcedev.
4. Select **Log In**. You will see your connected server information in the title bar of beSOURCE Developer.
5. Select **File > New Project**, or select **New Project** on the Start Page.
6. Select **Type the Project Name**.
7. In the **Project Name** box, enter a name.
8. In the **Language** box, select **C++**.
9. Select **Next**.



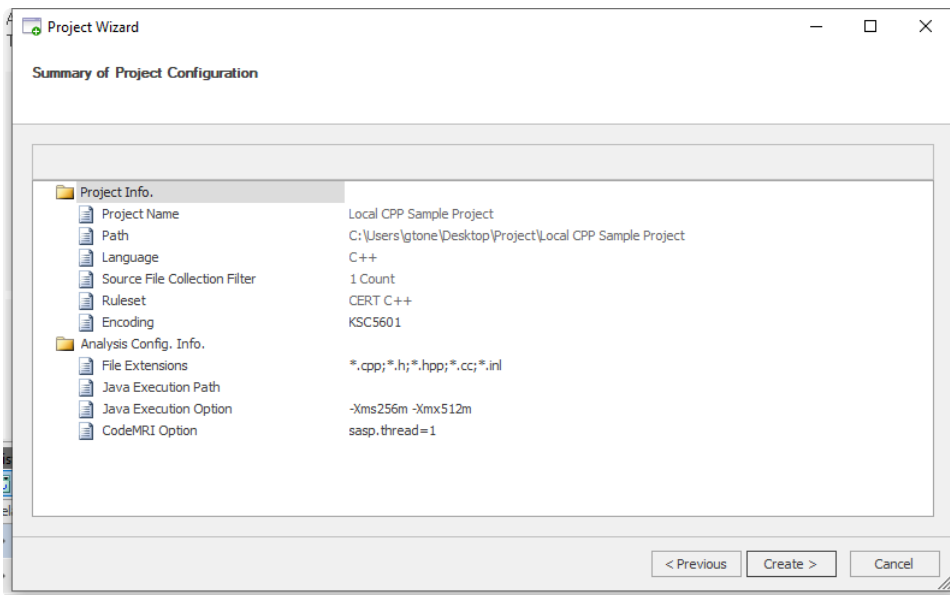
11. On the **Select a Ruleset** page, select **Local Ruleset** option and a ruleset.
12. Select **Next**.



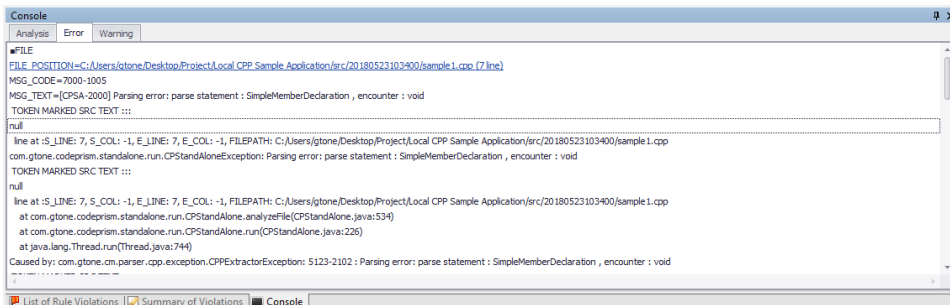
13. In the **Importing Path** box, select the sample source files path.
14. Select **Add (+)**.
15. Select **Next**.



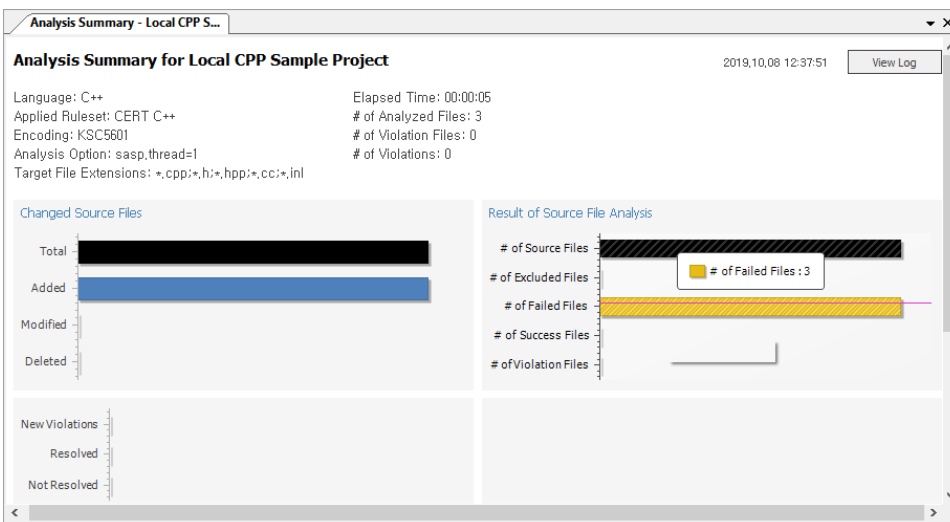
16. Review the summary of your project, and then select **Create**.



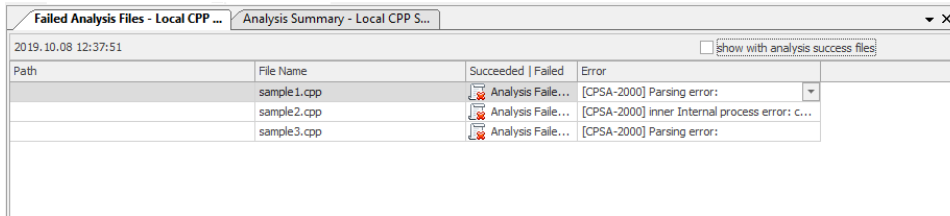
17. Select **Open the Project and Start Its Analysis** option.
18. Select **Finish**.
19. The **Error** tab in the Console window will show the parsing errors.



20. Otherwise, you can select the **# of Failed Files** graph in the Analysis Summary window.



21. You can get the parsing errors like below.

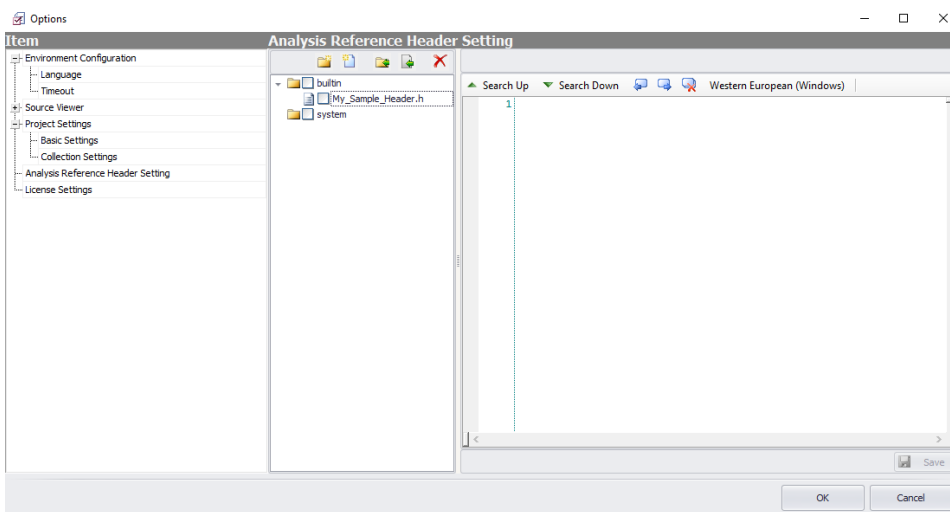


Path	File Name	Succeeded	Failed	Error
	sample1.cpp			[CPXA-2000] Parsing error:
	sample2.cpp			[CPXA-2000] inner Internal process error: c...
	sample3.cpp			[CPXA-2000] Parsing error:

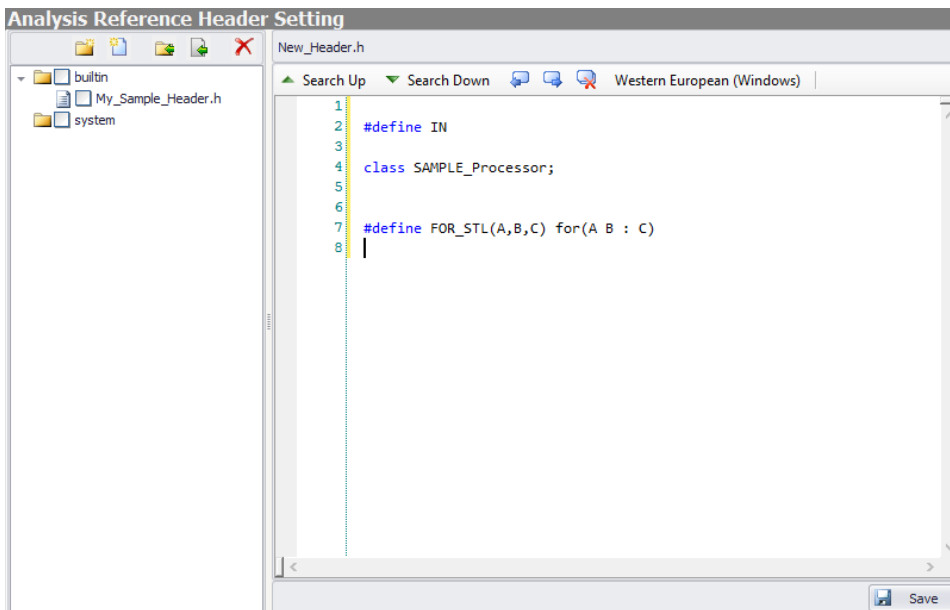
22. Select **Tools > Options > Analysis Reference Header Setting**.

23. Select **builtin**, and then select **Add a Header File**.

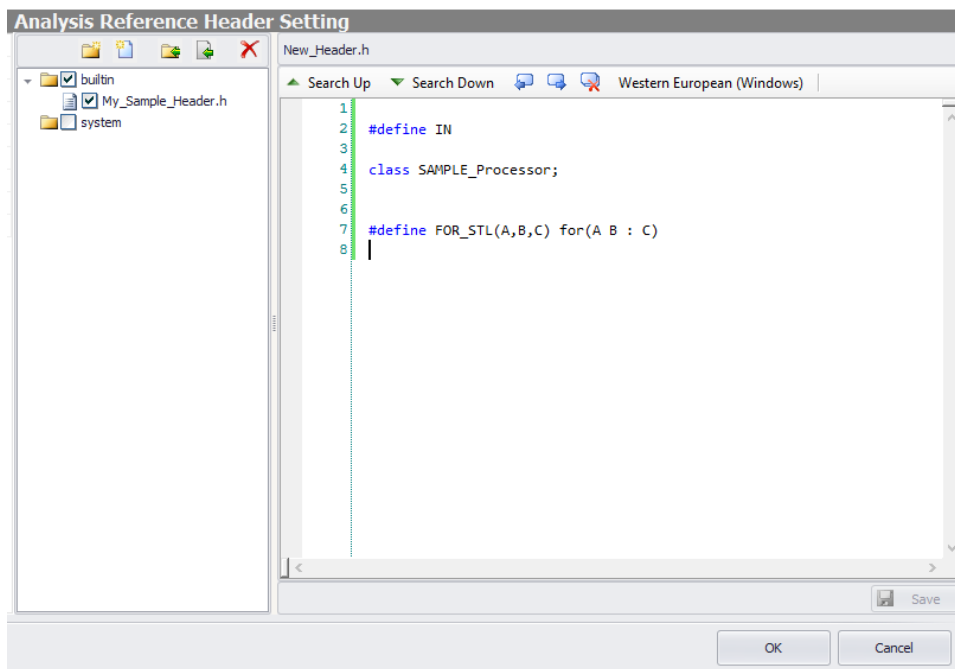
24. Change the new header file name.



25. Double-click the new header file and input definitions like below.



26. Select **Save** and check the new custom header.



27. Select **OK**.

28. Select **Start Analysis**. You will not get parsing errors.

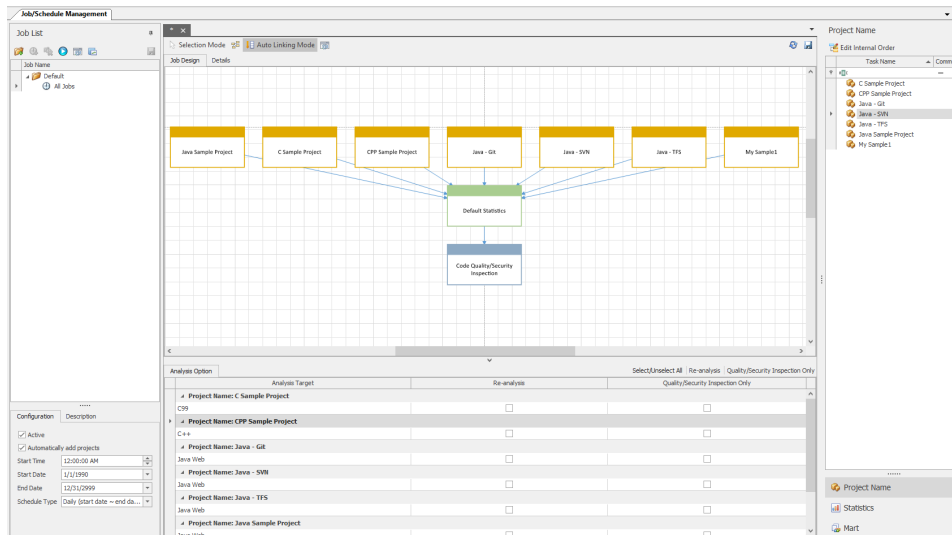
NOTE: The sample source files are just for showing fixing parsing errors. There would be no meaningful security defects in the result.

Automate Importing and Analyzing Source Files

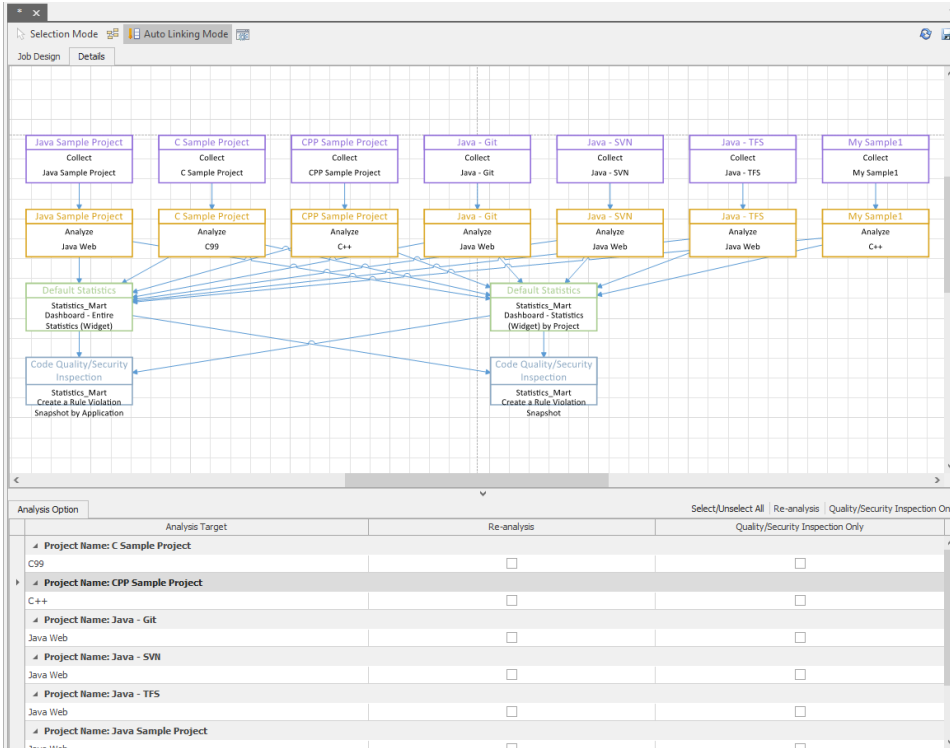
Scheduler of project

beSOURCE Server provides a scheduler, allowing administrator to automate importing and analyzing source files in the existing project. If you create and finish analysis of source files by using the Project Wizard, the scheduler job for the project is automatically added.

1. Open and log in to the **beSOURCE Admin Console**.
2. Select **Schedule > Schedule Setting**. The Job/Schedule Management window opens.



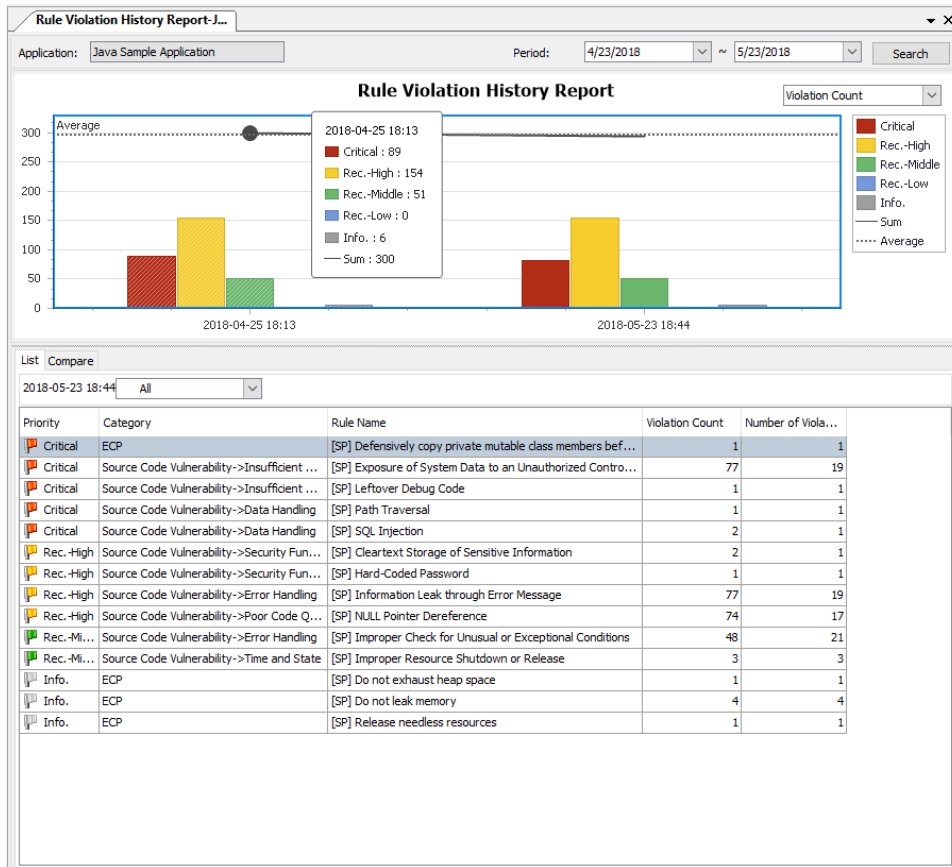
3. Select the **Details** tab. The detail tasks are shown.



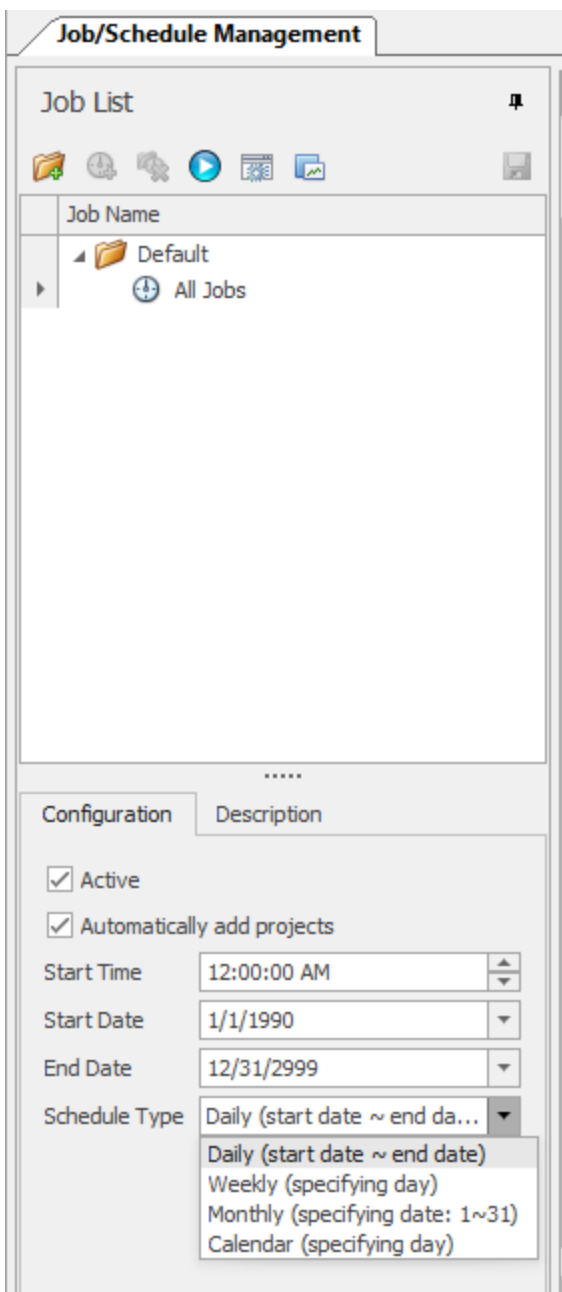
Each task does the following:

- **Collect** – Imports source files and libraries based on project configuration.
- **Analyze** – Scans source files by rule set.
- **Default Statistics**
 - **Statistics for all projects** – Builds internal data mart or calculates statistics for all projects. It is used in dashboard or reports.
 - **Statistics by project** - Builds internal data mart or calculates statistics by project. It is used in dashboard or reports.
- **Code Quality/Security Inspection**
 - **Create snapshot for all projects** – Makes a snapshot of code inspection. It is used for trends data.
 - **Create snapshot for by project** – Makes a snapshot of code inspection by project. It is used for trends data.

NOTE: An example of trends data is Rule Violation History report.




You can adjust some actions of job scheduler with configuration options.

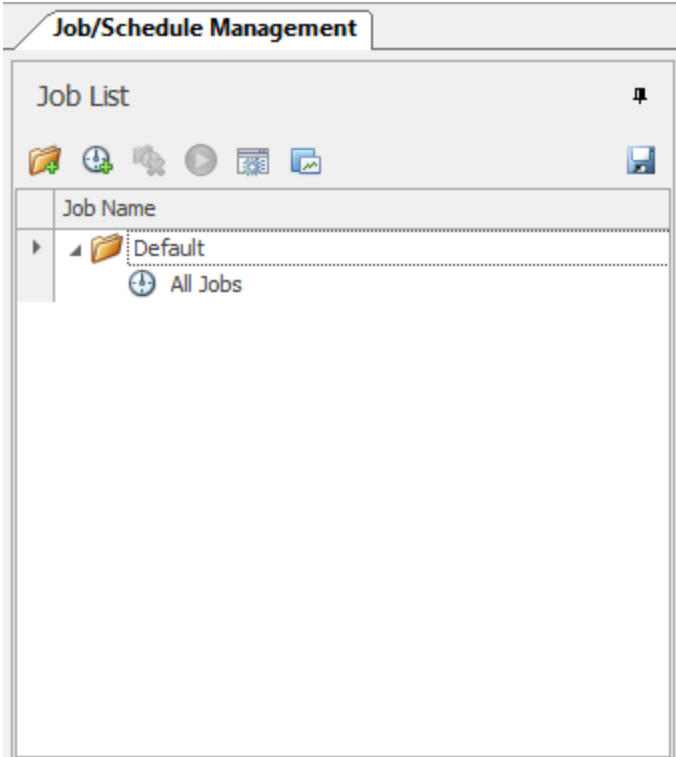



- **Active** - If you clear this checkbox, the scheduler job will not run.
- **Automatically add projects** - If you clear this checkbox, the Project Wizard will not make a scheduler for the new project.
- **Schedule Type** - You can select a scheduler type such as daily, weekly, monthly, or specific date (Calendar). You can set sub -options based on the option value.

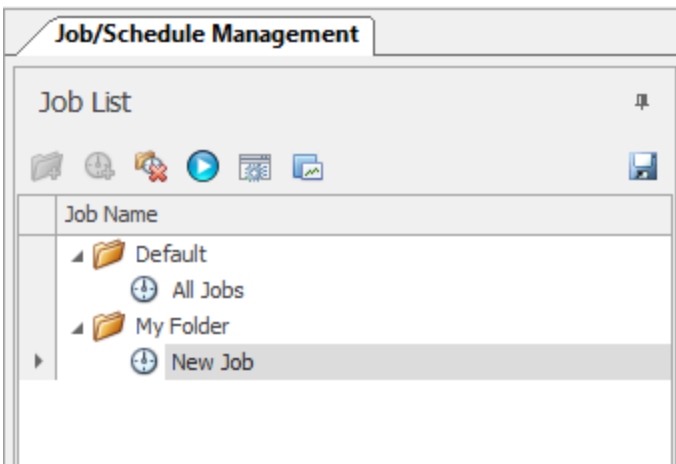
Making your own scheduler job

You can make your own scheduler job by doing the following:

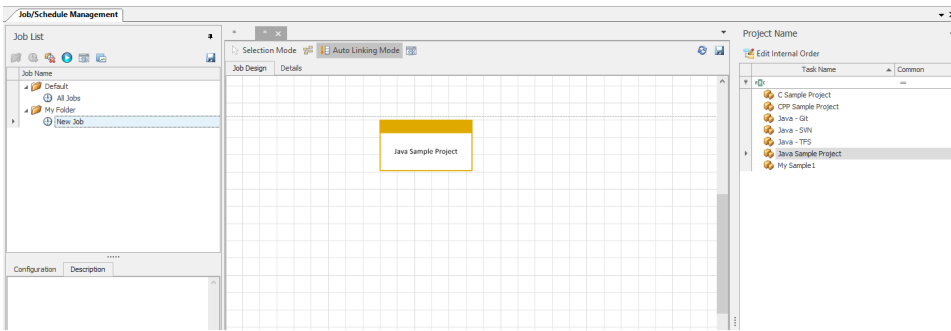
1. Open and log in to the **Admin Console** as an administrator.
2. Select **Schedule > Schedule Setting**.
3. Select the **Add Job Folder** () icon in the Job List window.



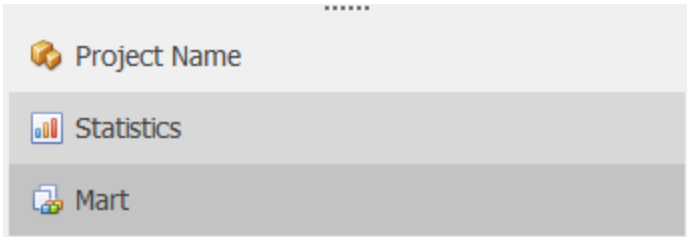
4. Change the folder name, and then select the **Add Job** () icon.



5. Select a project from the Job List.
6. Drag and drop it to the central area (designer).

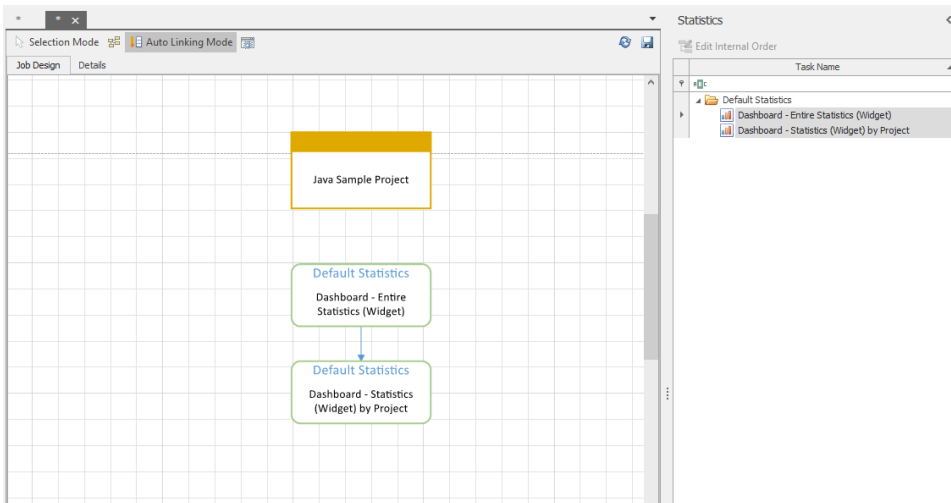


7. Select **Statistics**.

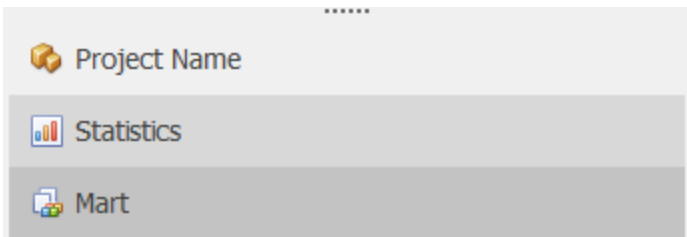


8. Select two statistics tasks in the right-side panel.

9. Drag and drop them to central area (designer).

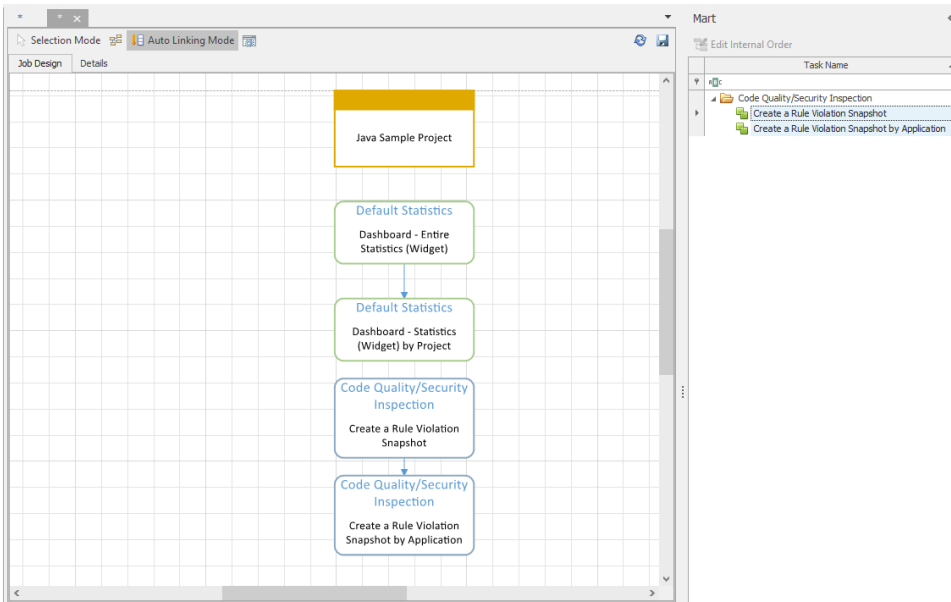


10. Select **Mart**.

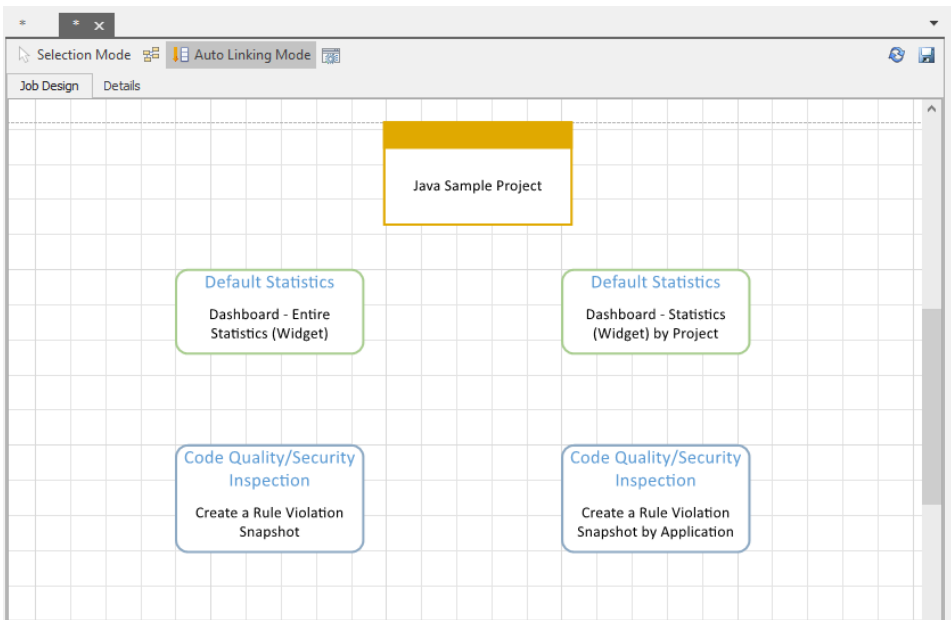


11. Select two statistics tasks in the right-side panel.

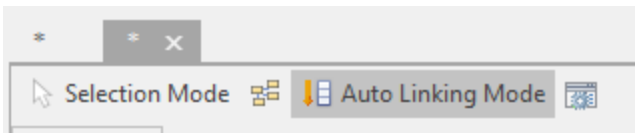
12. Drag and drop them to central area (designer).



13. Adjust each task as shown in this image. You can delete the arrow by pressing DELETE after selecting the arrow.



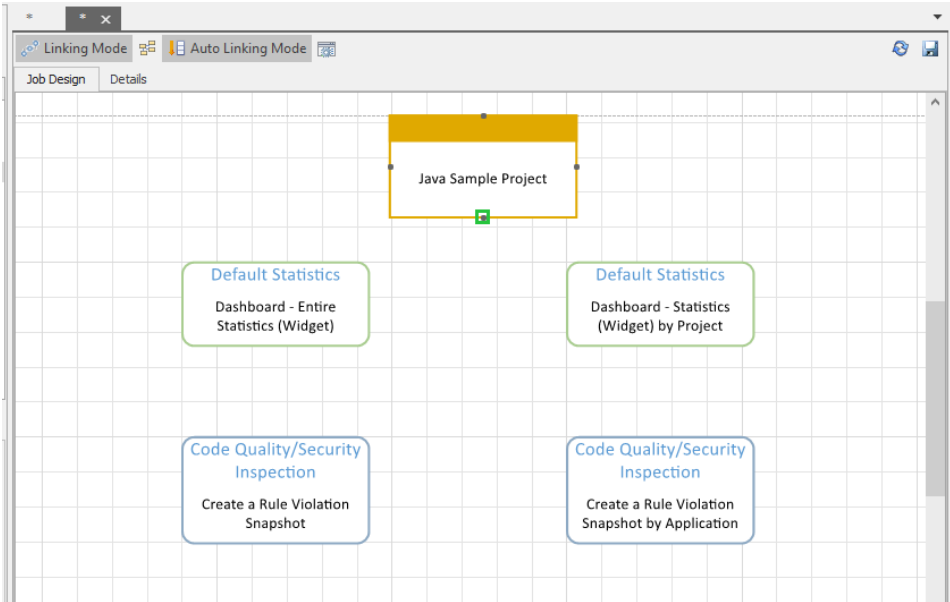
14. Select **Selection Mode**.



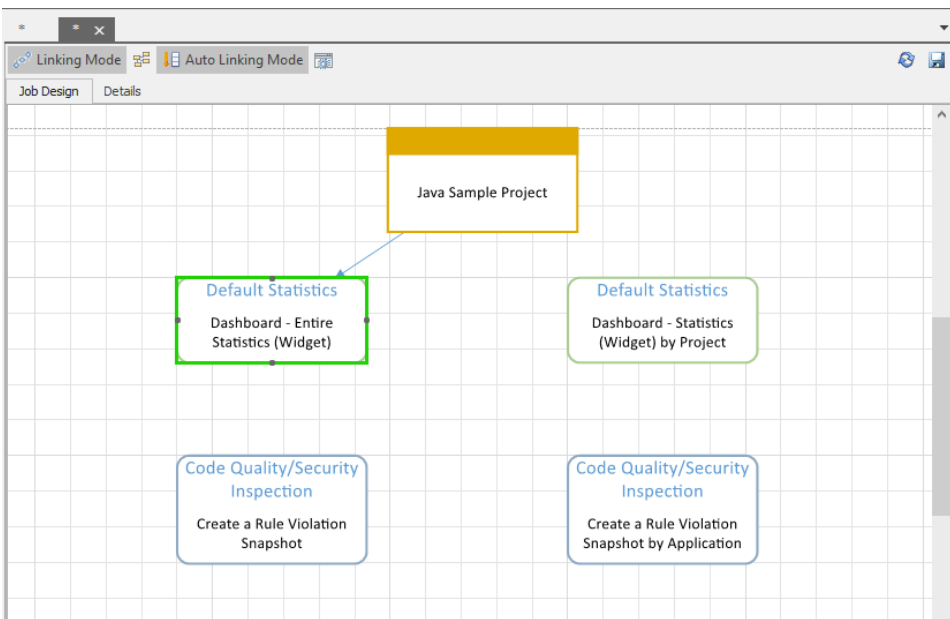
15. It will toggle to **Linking Mode**.



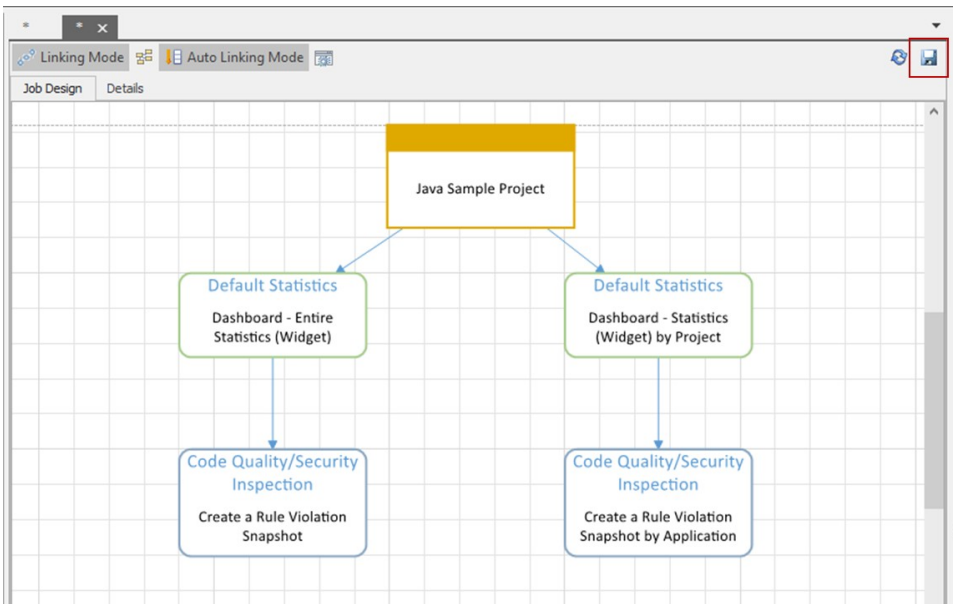
16. Move your pointer over on the bottom of the project box. A linking point appears.



17. Drag your pointer to a Default Statistics box to make an arrow.

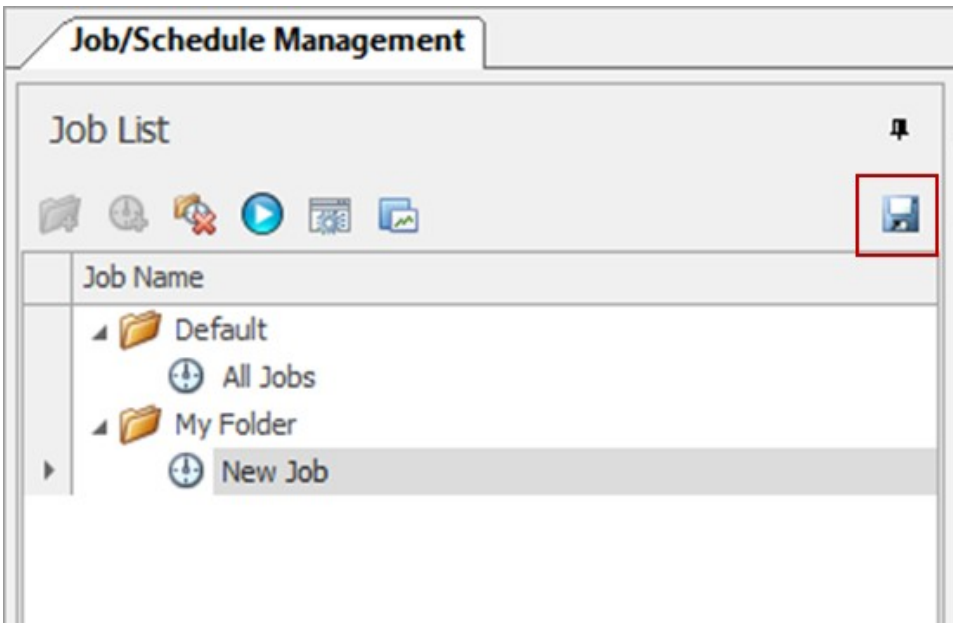


18. Set arrows as shown in this image, and then select **Save**.

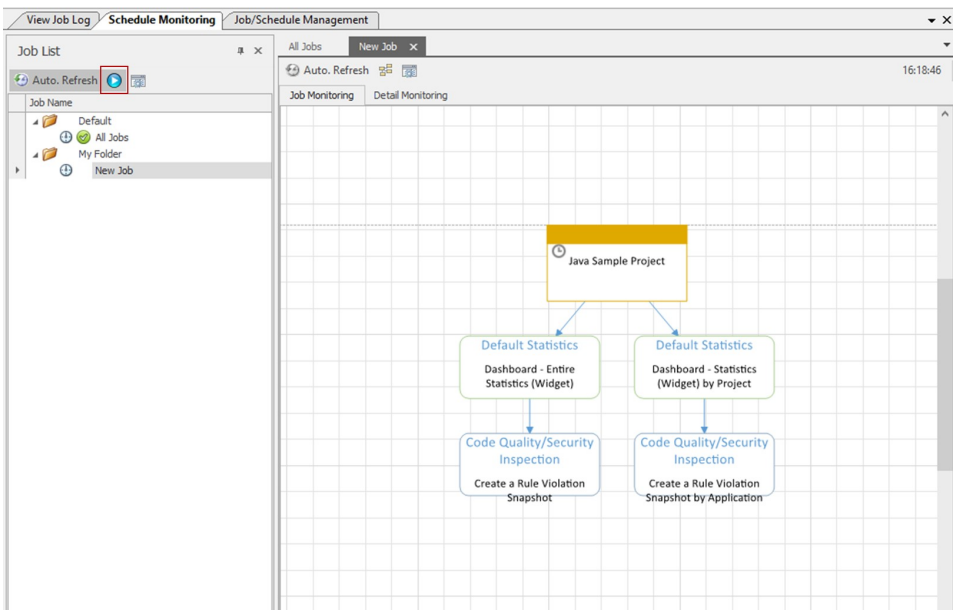


19. On the Configuration tab, set your options as shown in this image:

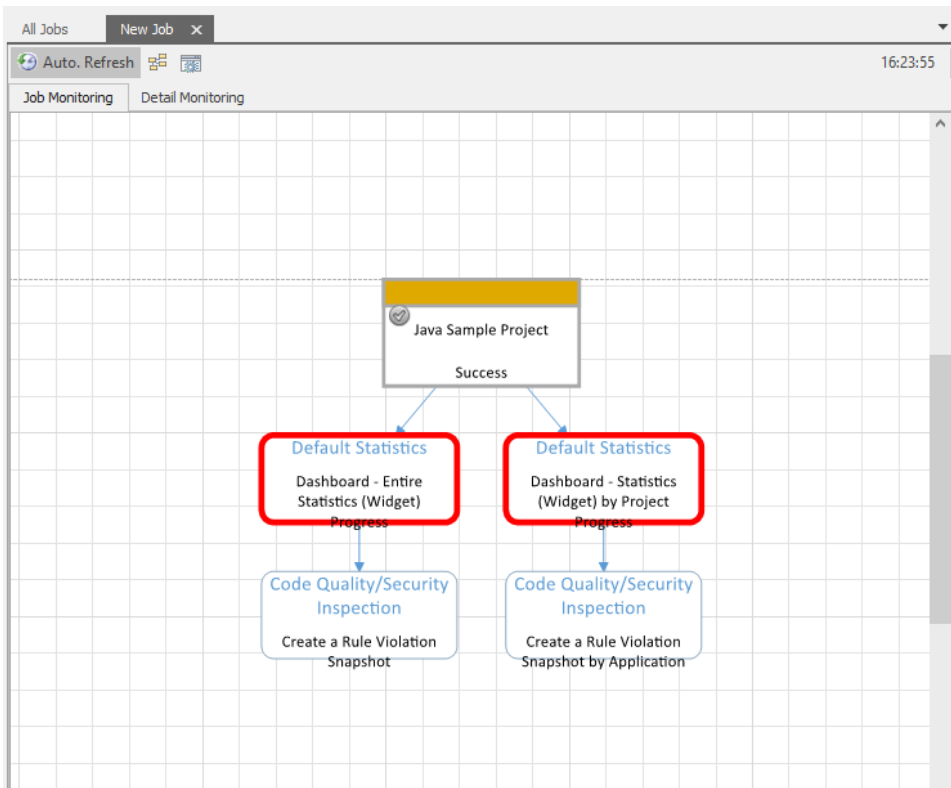
20. Select the **Save** icon on the Job List panel.



21. Select **Schedule > Schedule Monitoring**.
22. Select **New Job**.
23. Select the **Immediate Execute** icon.



24. On the dialog that appears, select **No**.
25. Select **Auto. Refresh**. The progress of the running schedule job tasks is shown.



Using the SSL (HTTPS) protocol

The beSOURCE Viewer Server embeds Tomcat web application server and it supports SSL protocol (TLS1.2) to ensure secure communication between the beSOURCE Server and Client.

To use SSL (HTTPS) protocol, security manager or administrator has to register certification issued from an official certification authority. The security manager or administrator should create Certificate Signing Request (CSR) first.

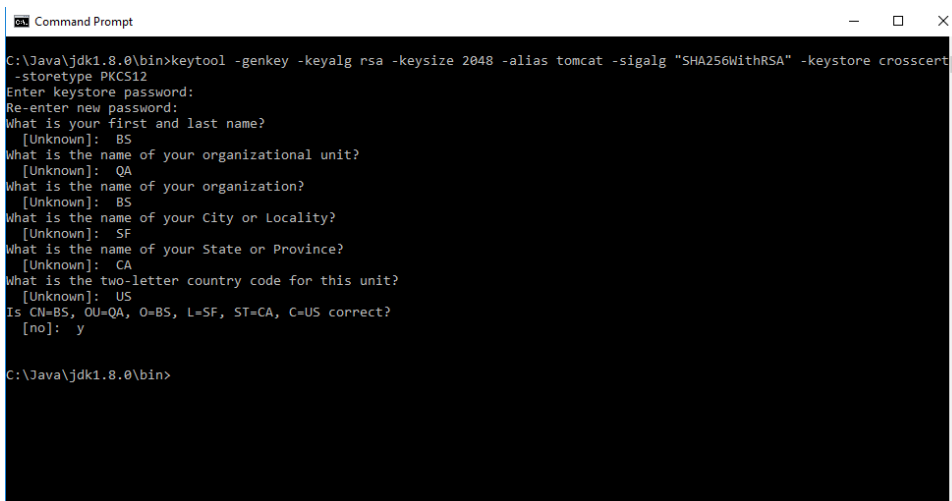
NOTE: Administrator can enable HTTPS protocol without the official certification but user's web browser will show message of invalid certification.

Creating the CSR in Java 1.8 or later

The below steps describe to create the CSR in a Java 1.8 or later environment by using keytool package.

1. In Windows, open **Command Prompt** as an administrator.
2. Move to your Java bin directory (for example C:\>CD Java\jdk1.8.0\bin).
3. Input following command and the required information to generate keystore:

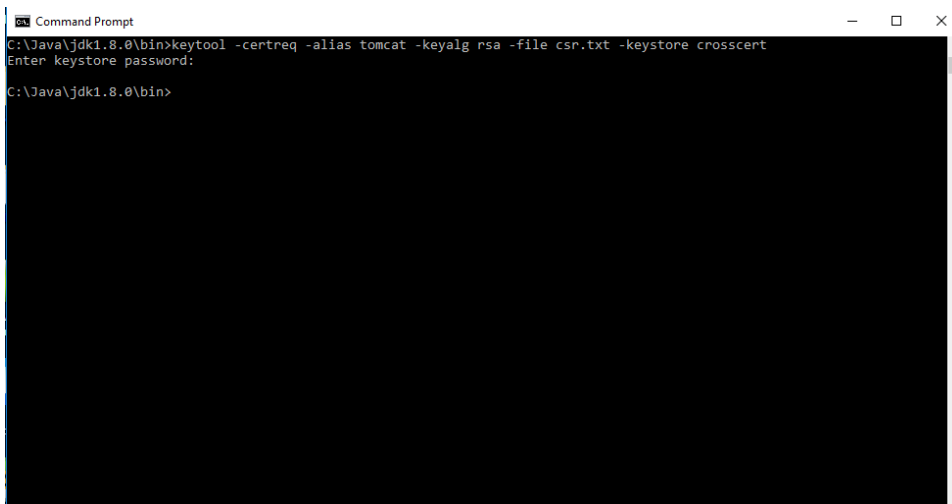
```
keytool -genkey -keyalg rsa -keysize 2048 -alias tomcat -sigalg "SHA256WithRSA" -keystore crosscert -storetype PKCS12
```



```
Command Prompt
C:\Java\jdk1.8.0\bin>keytool -genkey -keyalg rsa -keysize 2048 -alias tomcat -sigalg "SHA256WithRSA" -keystore crosscert
-storetype PKCS12
Enter keystore password:
Re-enter new password:
What is your first and last name?
[Unknown]: BS
What is the name of your organizational unit?
[Unknown]: QA
What is the name of your organization?
[Unknown]: BS
What is the name of your City or Locality?
[Unknown]: SF
What is the name of your State or Province?
[Unknown]: CA
What is the two-letter country code for this unit?
[Unknown]: US
Is CN=BS, OU=QA, O=BS, L=SF, ST=CA, C=US correct?
[no]: y
C:\Java\jdk1.8.0\bin>
```

4. Enter the following command and enter your password to create a CSR file:

```
keytool -certreq -alias tomcat -keyalg rsa -file csr.txt -  
keystore crosscert
```



5. You may be able to find the CSR file in the Java bin directory.
6. You can request official certification for the CSR file with the certification authority. For more information about registering the official certification to your OS, refer to guideline of the certification authority.

NOTE: The keytool package of JDK would behave differently depending on JDK version. For more information of the use of keytool, refer to JDK guideline.

Configuring the beSOURCE View Server for the SSL (HTTPS) protocol

1. Open the `beSOURCE_Install_Directory\WAS\conf\server.xml` file with a text editor.
2. Add following `<Connector>` attribute:

NOTE:

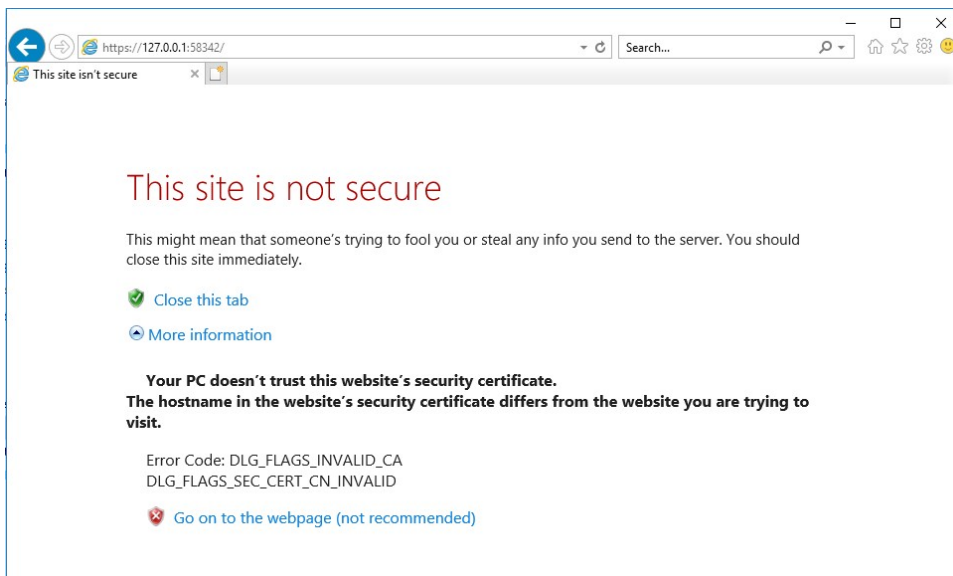
- The port number '58342' is an example. The important property is `SSLEnabled="true"`.
- The `keystoreFile="C:\Java\jdk1.8.0\bin\crosscert"` should be changed to your environment.
- The `keystorePass="besource"` should be changed to your password.

```
<Connector port="50102"
maxHttpHeaderSize="8192"maxThreads="150" minSpareThreads="25"
maxSpareThreads="75" enableLookups="false" redirectPort="9443"
acceptCount="100" URIEncoding="iso-8859-1"
useBodyEncodingForURI="true" />

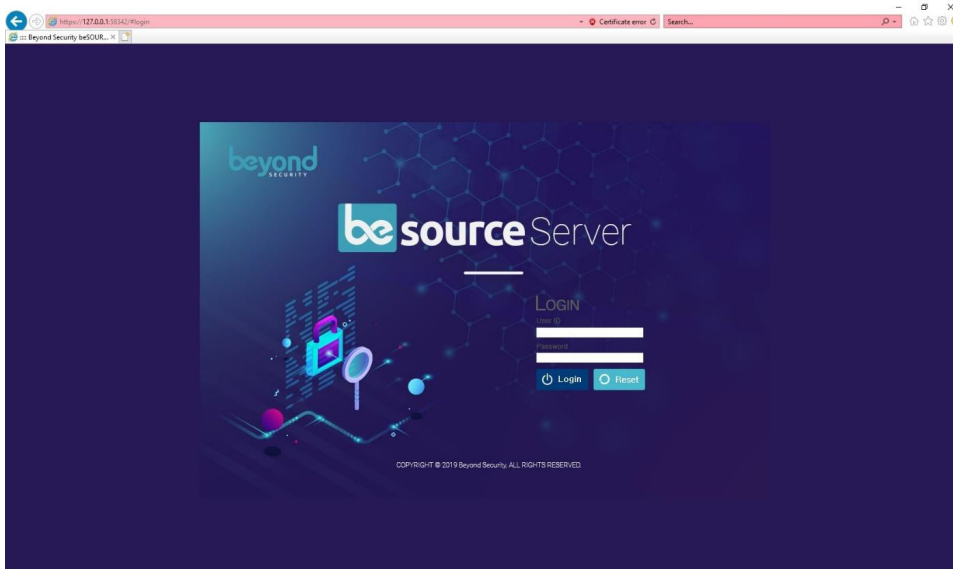
<!-- A "Connector" using the shared thread pool-->

<Connector port="58342" protocol="HTTP/1.1"
connectionTimeout="20000"
maxHttpHeaderSize="8192"maxThreads="150" minSpareThreads="25"
maxSpareThreads="75"maxPostSize="1048576"
maxParameterCount="40" enableLookups="false"
redirectPort="9443" acceptCount="100" URIEncoding="euc-kr"
useBodyEncodingForURI="true" SSLEnabled="true"
clientAuth="false" scheme="https" secure="true"
sslProtocols="TLS"
keystoreFile="C:\Java\jdk1.8.0\bin\crosscert"
keystorePass="besource" keystoreType="PKCS12" />
```

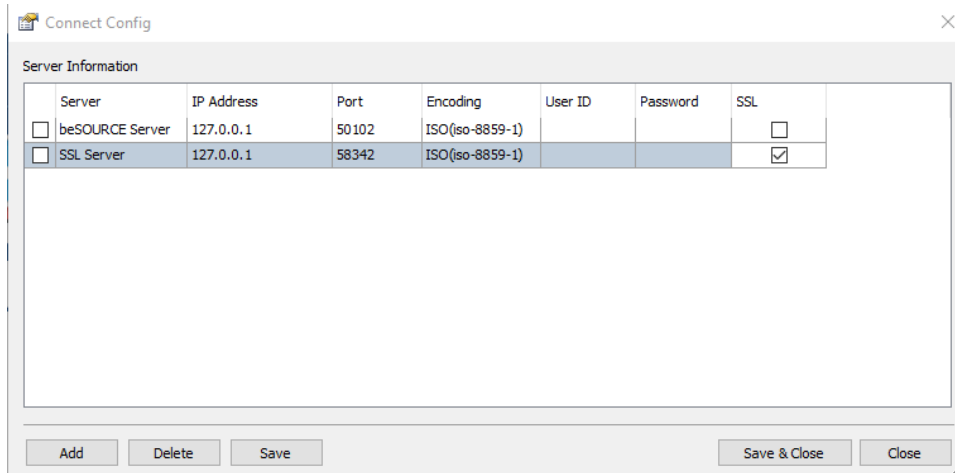
3. Add the certified domain and IP address to the hosts file (Not necessary for localhost testing).
 - Windows: C:/Windows/System32/drivers/etc/hosts
 - Linux: /etc/hosts
4. Restart the beSOURCE View Server.
5. In your web browser, go to **https://127.0.0.1:58342**. The browser will say that this site is not secure if you did not apply official certification.
6. Select **More Information**.
7. Select **Go on to the webpage (not recommended)**.



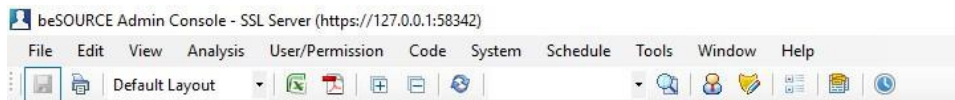
8. You can use the website for testing before applying official certificate.



9. Open the **beSOURCE Admin Console**, **beSOURCE Client**, or **beSOURCE Developer**.
10. Select **Config**.
11. Select **Add**.
12. In the **Connect Config** dialog, enter your beSOURCE Server's information.



13. Select **Save & Close**.
14. Log in to the SSL Server.
15. You can see the HTTPS protocol server URL in the title bar.

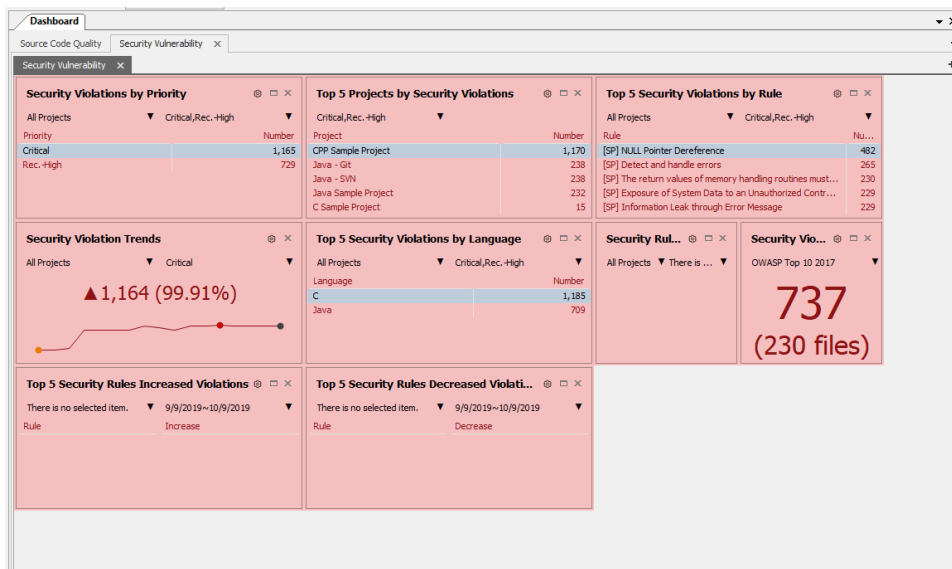



Dashboard

Viewing the dashboard

The beSOURCE Client provides a personalized dashboard view.



1. Open and log in to the beSOURCE **Client** as the administrator (User ID: **besource**, Password: **besource**).
2. Select **View > Function Explorer**.
3. Select **Dashboard > Security Vulnerability** to open the Dashboard.

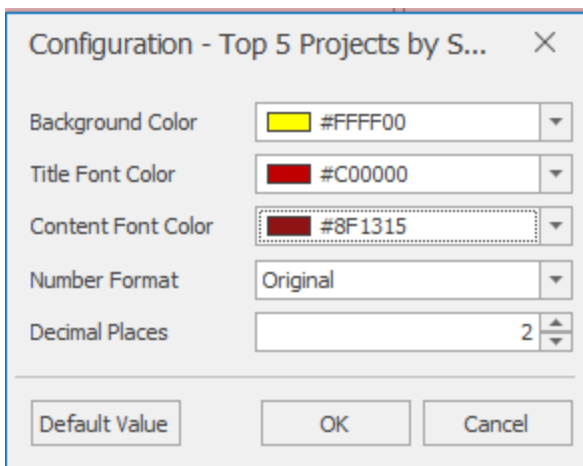


4. Selecting the **View List** icon () for a dashboard data item shows the detail list.

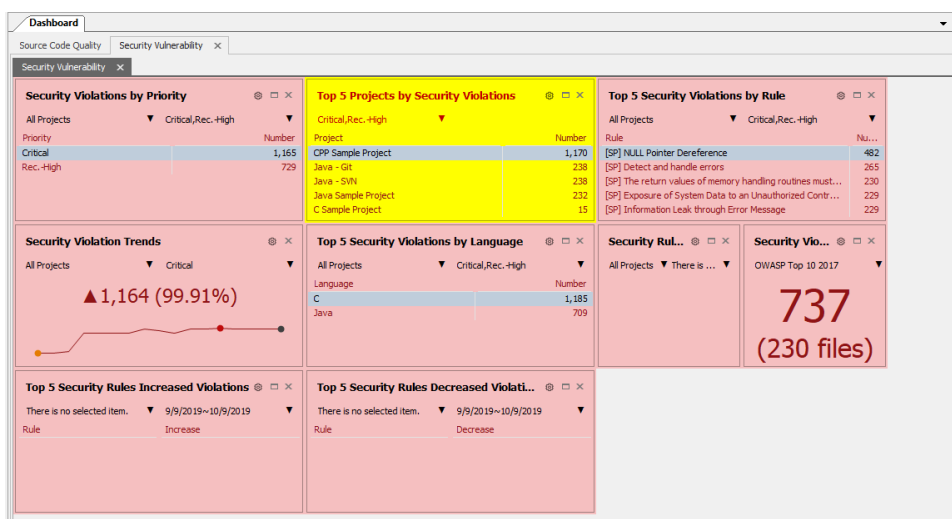
The screenshot shows a table titled "Violations Count by Project" with the following data:

Project	Number
CPP Sample Project	1170
Java - Git	238
Java - SVN	238
Java Sample Project	232
C Sample Project	15
Java - TFS	1

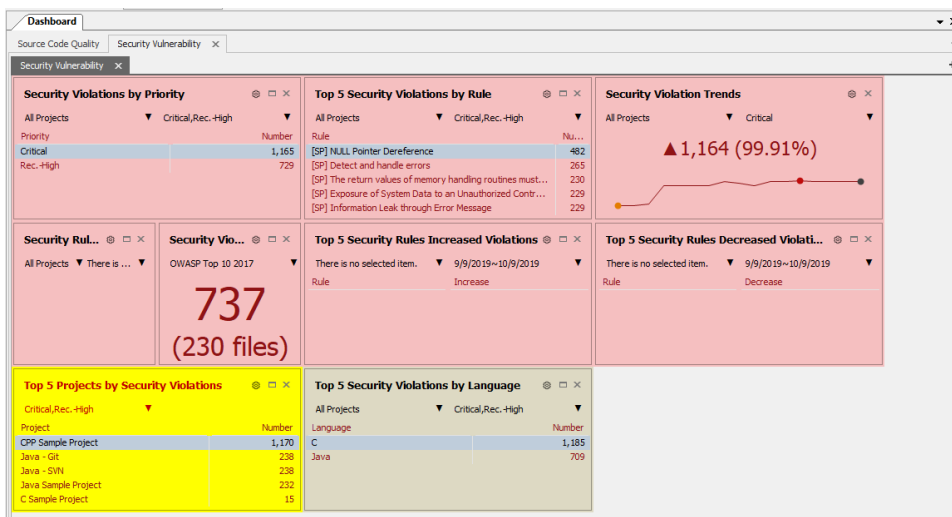
5. To open the dashboard again, select the **Return** () icon.
6. To set options for the selected dashboard item, select the **Configuration** () icon. Change some options, and then select OK.



7. Your changes will appear in the customized dashboard.



8. You can rearrange each dashboard item by dragging and dropping them.

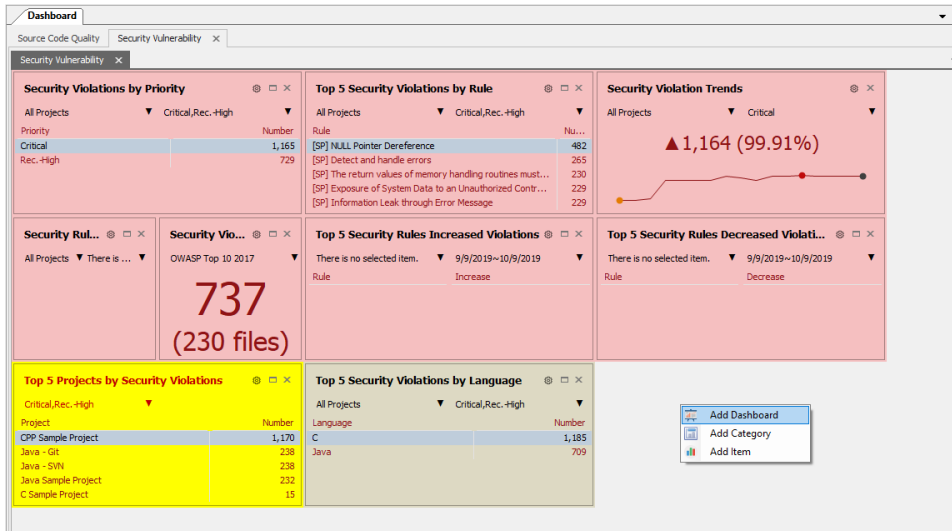


The personalized dashboard configurations are saved on the beSOURCE Server.

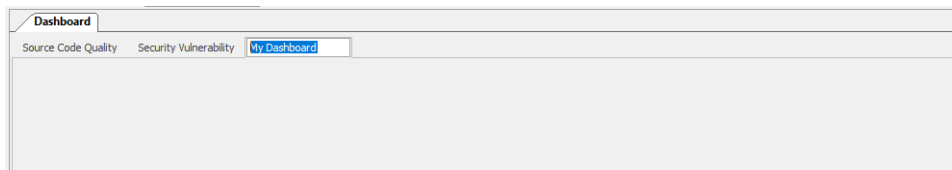
Making a new dashboard

You can make your own dashboard.

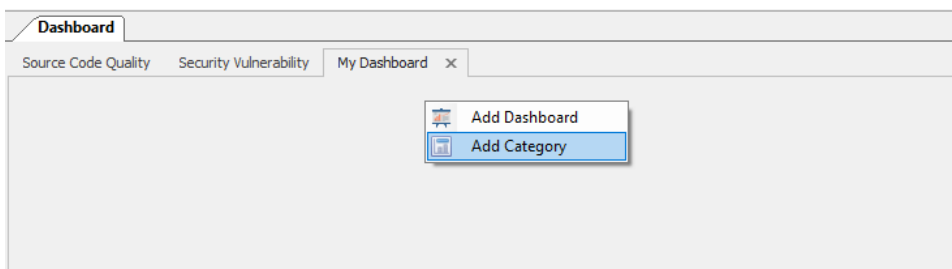
1. Right-click anywhere on the dashboard to open the shortcut menu.
2. Select **Add Dashboard**.



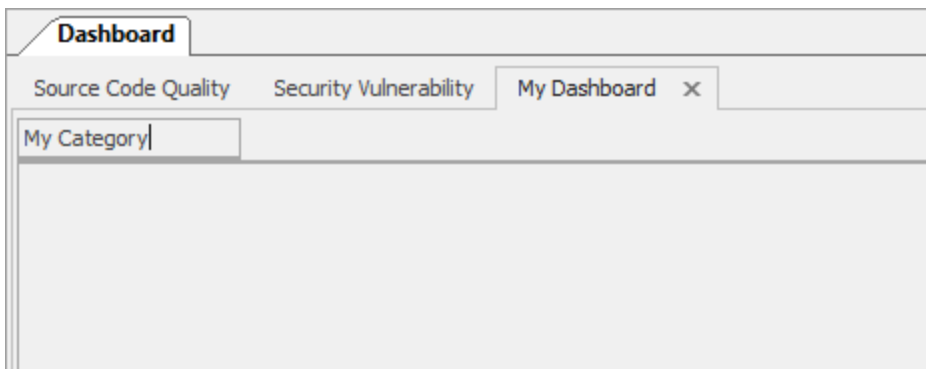
3. Double-click the dashboard name and change it.



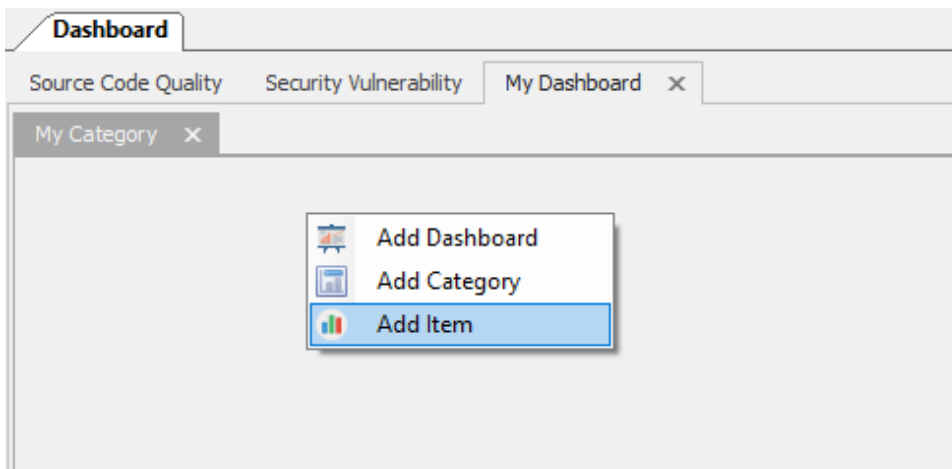
4. Right-click the area below the tab, and then select **Add Category**.



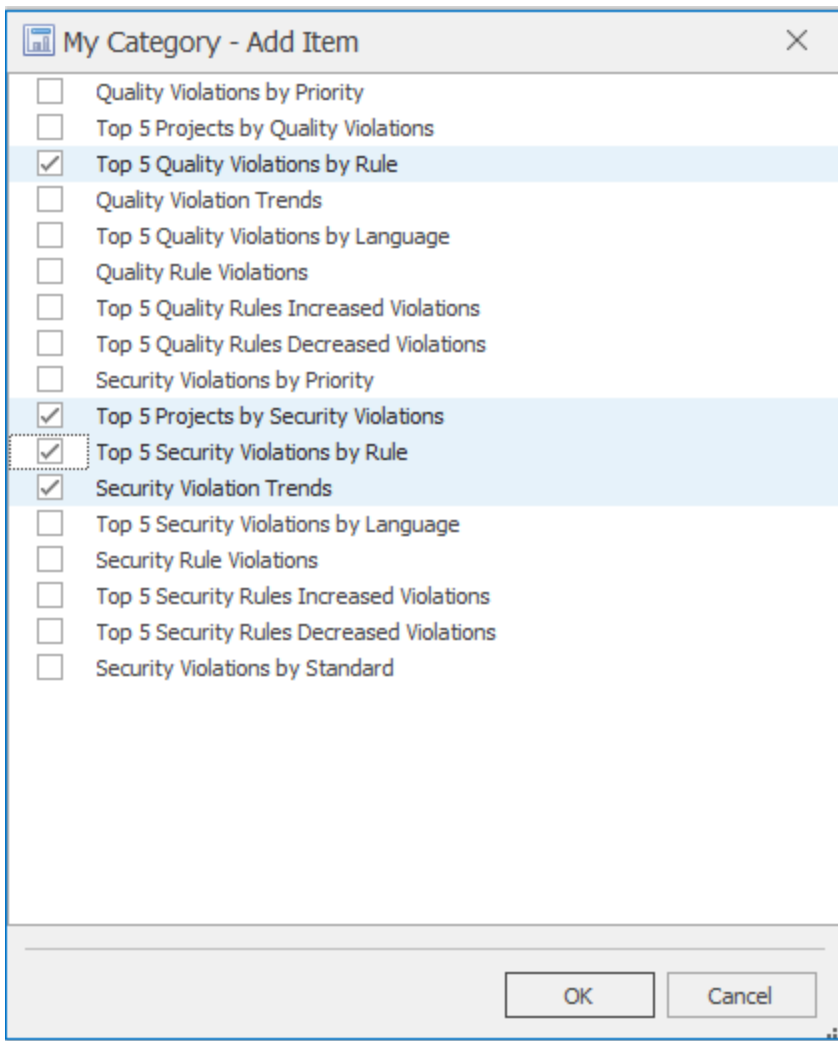
5. Double-click the new category name and change it.



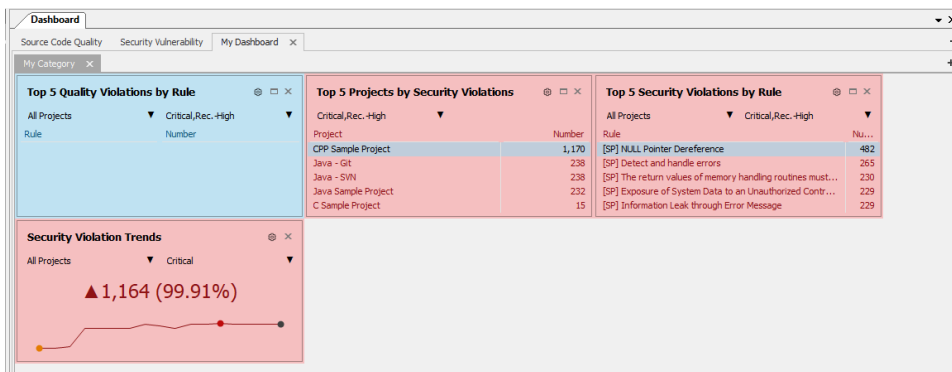
6. Right-click the area below, and then select **Add Item**.



7. Select some dashboard items, and then select **OK**.



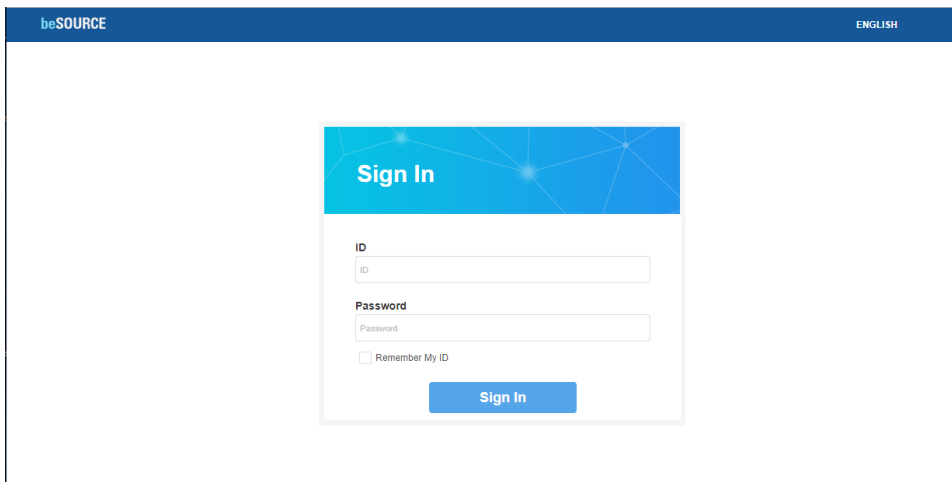
8. Your new dashboard now appears.



Web User Interface

beSOURCE V5 Enterprise Edition supports a web client user interface.

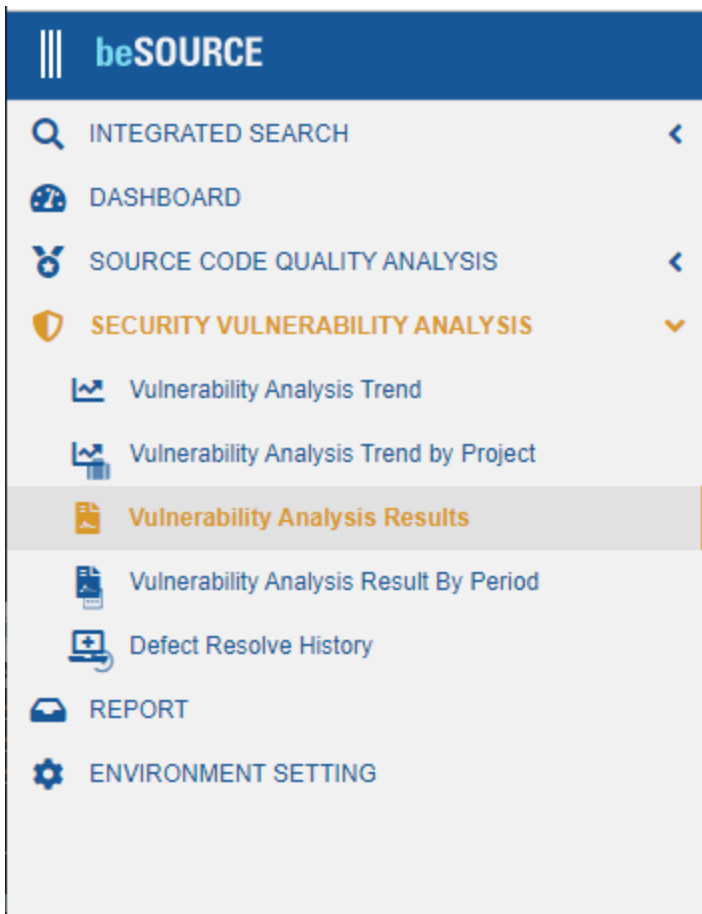
1. Open your web browser.
2. Go to `http://[beSOURCE Server IP]:50104` (for example, `http://127.0.0.1:50104`).
3. On the **Sign In** page, enter your **User ID** and **Password**.



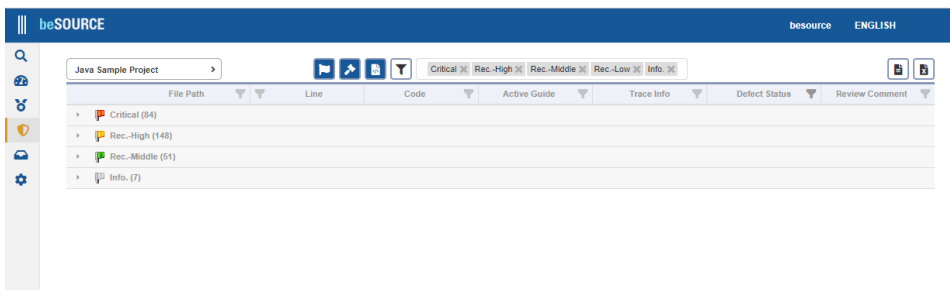
4. After signing in, the beSOURCE client defaults to the My Dashboard page.



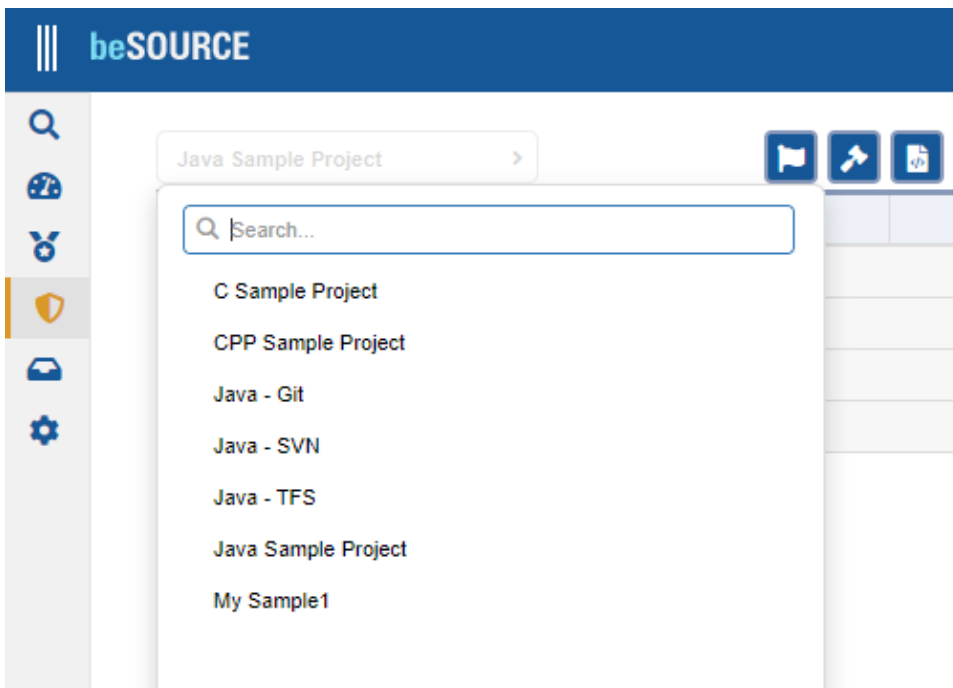
5. From the left menu bar, select **Security Vulnerability Analysis > Vulnerability Analysis Results**.



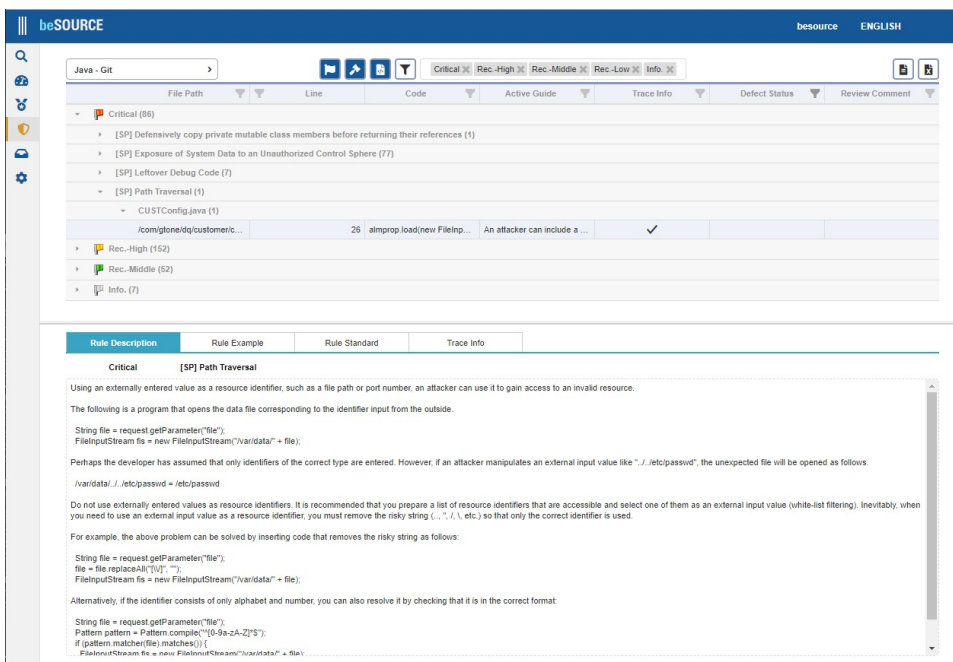
6. The rule violation list appears.



7. You can select another project from the box at the top.

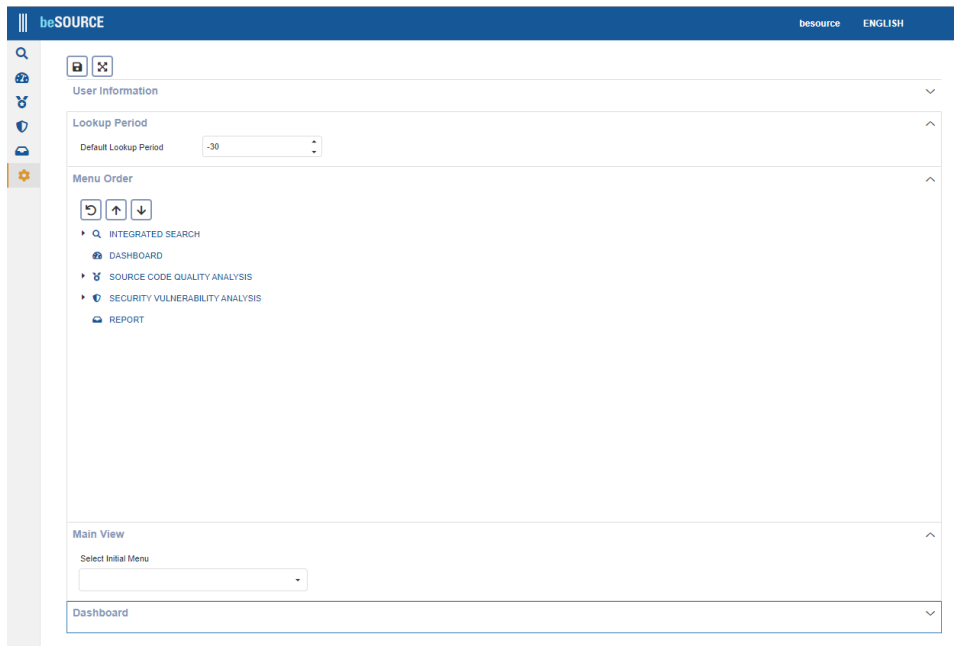


8. When you select a rule violation, the lower area shows information such as rule description, rule example, referencing standards, and flow trace data.



9. Select Environment Setting from the left menu bar.
10. You can adjust several options for web user interface:
- **User account information** Changes your password.
 - **Lookup Period** - The default number of days for viewing periodical data.
 - **Menu Order** - Customizes menu display order.

- **Main View** - Selects the first page.
- **Dashboard** - Displays the Dashboard view.



Appendix

How to get debugging information for technical support

If you suspect an analysis problem with a particular source file, Fortra Technical Support ask you to submit certain internal files for more accurate cause tracking. Typically, there are C/C++ source extension file (conv2 file) for parsing errors, and intermediary file (coil file) for security vulnerability analysis.

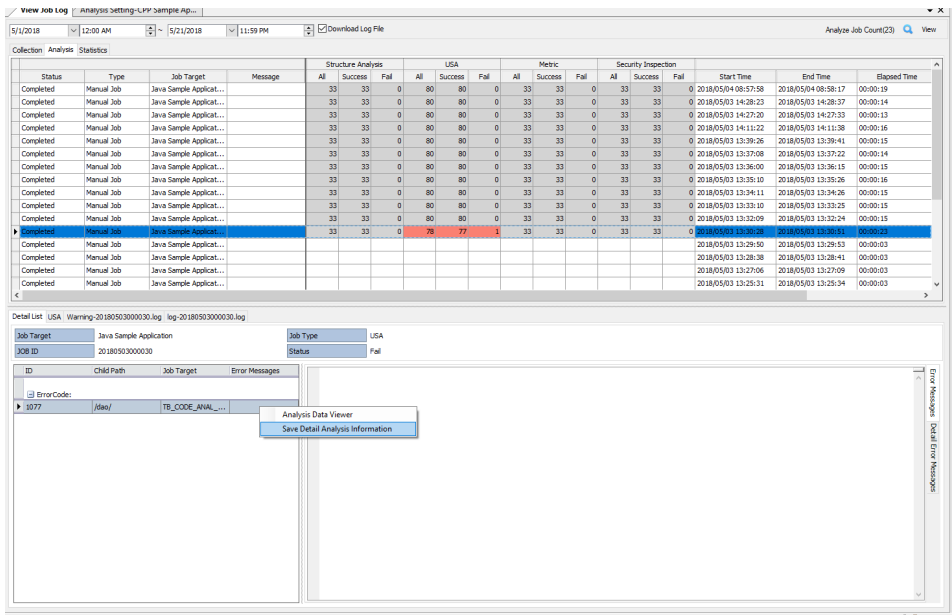
NOTE: The beSOURCE language parser internally generates Common Objective Intermediary files for security vulnerability analysis for most languages. These files have the extension * `.coil` and are binary. The beSOURCE Analysis Engine analyzes the coil files.

Providing debug information

If you are asked to provide more detail debugging information, do the following:

1. Log in to the **Admin Console** as an administrator.
2. Select **System > View Job Log**.
3. Select the **Analysis** tab.
4. Double-click an analysis job in the list.
5. Select one or more error source files in the lower left list.
6. From the shortcut menu, select **Save Detail Analysis Information** to generate zip files. A zip file includes following:
 - Analysis configuration information file
 - The source file

- The coil file



Providing the conv2 file for C/C++ language

If you are asked to provide conv2 file for C/C++ language, do the following:

1. Log in to the **Admin Console** as an administrator.
2. Select **System > View Job Log**.
3. Select the **Analysis** tab.
4. Double-click an analysis job in the list.
6. From the lower panel, select the log file and then press CTRL + F key. The Find window appears.

View Job Log

2018-05-17 오전 12:00 ~ 2018-05-17 오후 11:59 Download Log File

Collection Analysis Statistics Pattern

Status	Type	Job Target	Message	구분별			USA			Metric		
				All	Succ	Fail	All	Succ...	Fail	All	Succ...	Fail
성공	수동작업	C++		13	13	0	13	13	0	13	13	0

Detail List USA Warning-20180517000122.log log-20180517000122.log

```

1 [09:54:25:373] 20180517000122 <INFO> Start ChangeMiner ($Revision: 1.5 $)-ManagedServer [20180517000122]
2 [09:54:25:373] 20180517000122 <INFO> * vm-name -Java HotSpot(TM) 64-Bit Server VM
3 [09:54:25:373] 20180517000122 <INFO> * vm-version-20.45-b01
4 [09:54:25:373] 20180517000122 <INFO> * vm-vendor -Sun Microsystems Inc.
5 [09:54:25:687] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal010.jar : 8.3.2.0 [2017.12.05]
6 [09:54:25:696] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal011.jar : 8.2.0.1 [2017.02.07]
7 [09:54:25:708] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal013.jar : 8.3.2.0 [2017.12.05]
8 [09:54:25:722] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal016.jar : 8.3.2.0 [2017.12.05]
9 [09:54:25:730] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal020.jar : 8.3.2.0 [2017.12.05]
10 [09:54:25:737] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal030.jar : 8.3.2.0 [2017.12.05]
11 [09:54:25:750] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal040.jar : 8.3.2.0 [2017.12.05]
12 [09:54:25:758] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal050.jar : 8.3.2.0 [2017.12.05]
13 [09:54:25:769] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal055.jar : 8.3.2.0 [2017.12.05]
14 [09:54:25:779] 20180517000122 <DEBUG> Addon-cm.itplus.cm.ceal056.jar : 8.3.2.0 [2017.12.05]

```

7. In the Find window, find a problem file name (ex: assert.cpp).

Detail List USA Warning-20180517000122.log log-20180517000122.log

```

370 22 <TRACE : 5000> current built in file :-D:/AppGov/analyzer/server/CE/cpp_buittin.h
371 22 <TRACE : 5000> doLocalAnalyze-/cp/source/DoNotThrowInDestructorVisitor.cpp's data analyze time ::
372 22 <TRACE : 5000> [CoilTransManager] coil write: 20180517000183/cp/source/DoNotThrowInDestructorVisitor.cpp to I:/APPGOV_SRC/SRC/oa
373 22 <TRACE : 5000> doLocalAnalyze-/cp/source/DoNotThrowInDestructorVisitor.cpp's data analyze time ::
374 22 <DEBUG> Common-Parsing-[33104] Find
375 22 <INFO> Common-Parsing-[33104]
376 22 <TRACE : 5000> current built in file cnt is -1
377 22 <TRACE : 5000> current built in file cnt is -1
378 22 <DEBUG> ## initializeForCommonAnalyzeUnit
379 22 <TRACE : 5000> [C Preprocessor]
380 22 <TRACE : 5000> [C Preprocessor]
381 22 <TRACE : 5000> [C Preprocessor]
382 22 <DEBUG> CustomDPD-inserted
383 22 <DEBUG> TEMP_FILE_PREFIX -I:/APPGOV_SRC/SRC/temp/20180517000182/mustDeclConstructorVisitor.cpp_349747696794854549
384 22 <TRACE : 5000> current built in file cnt is -1
385 22 <TRACE : 5000> current built in file :-D:/AppGov/analyzer/server/CE/cpp_buittin.h
386 22 <TRACE : 5000> doLocalAnalyze-/cp/source/MustDeclConstructorVisitor.cpp's data analyze time ::
387 22 <TRACE : 5000> [CoilTransManager] coil write: 20180517000183/cp/source/MustDeclConstructorVisitor.cpp to I:/APPGOV_SRC/SRC/osrc/
388 22 <TRACE : 5000> doLocalAnalyze-/cp/source/MustDeclConstructorVisitor.cpp coil write.
389 22 <DEBUG> Common-Parsing-[33105]/cp/source/MustDeclConstructorVisitor.cpp [25ms Elapsed.]
390 22 <INFO> Common-Parsing-[33106]/CR/9566/assert.cpp
391 22 <TRACE : 5000> current built in file cnt is -1
392 22 <TRACE : 5000> current built in file :-D:/AppGov/analyzer/server/CE/cpp_buittin.h
393 22 <DEBUG> ## initializeForCommonAnalyzeUnit :: runCPP.cppArg.isAutoDeleteTempFile() is called=true
394 22 <DEBUG> CustomDPD-inserted user dpd cc line. (0)
395 22 <DEBUG> Structure-Analysis-[33101]/cp/source/DoNotAssignAddrToParamVisitor.cpp [134ms Elapsed.]

```

8. Determine the conv2 file path of the corresponding file.

Find

Find What: assert.cpp

Find Next

Match Case Search Hidden Text Mark All

Match Whole Word Only Search in Selection Close

Search Up Search Down Use Option: Regular Expression

```

375 22 <INFO> Common-Parsing-[33105]/cp/source/MustDeclConstructorVisitor.cpp
376 22 <TRACE : 5000> current built in file cnt is -1
377 22 <TRACE : 5000> current built in file cnt is -1
378 22 <DEBUG> ## initializeForCommonAnalyzeUnit
379 22 <TRACE : 5000> [C Preprocessor]
380 22 <TRACE : 5000> [C Preprocessor]
381 22 <TRACE : 5000> [C Preprocessor]
382 22 <DEBUG> CustomDPD-inserted
383 22 <DEBUG> TEMP_FILE_PREFIX -I:/APPGOV_SRC/SRC/temp/20180517000182/mustDeclConstructorVisitor.cpp_349747696794854549
384 22 <TRACE : 5000> current built in file cnt is -1
385 22 <TRACE : 5000> current built in file cnt is -1
386 22 <TRACE : 5000> doLocalAnalyze-/cp/source/mustDeclConstructorVisitor.cpp's data analyze time ::
387 22 <TRACE : 5000> [CoilTransManager] coil write: 20180517000183/cp/source/MustDeclConstructorVisitor.cpp to I:/APPGOV_SRC/SRC/osrc/2018051700
388 22 <TRACE : 5000> doLocalAnalyze-/cp/source/MustDeclConstructorVisitor.cpp coil write.
389 22 <DEBUG> Common-Parsing-[33105]/cp/source/MustDeclConstructorVisitor.cpp [25ms Elapsed.]
390 22 <INFO> Common-Parsing-[33106]/CR/9566/assert.cpp
391 22 <TRACE : 5000> current built in file cnt is -1
392 22 <TRACE : 5000> current built in file :-D:/AppGov/analyzer/server/CE/cpp_buittin.h
393 22 <DEBUG> ## initializeForCommonAnalyzeUnit :: runCPP.cppArg.isAutoDeleteTempFile() is called=true
394 22 <DEBUG> CustomDPD-inserted user dpd cc line. (0)
395 22 <DEBUG> Structure-Analysis-[33101]/cp/source/DoNotAssignAddrToParamVisitor.cpp [134ms Elapsed.]
396 22 <TRACE : 5000> [C Preprocessor]-input file - I:/APPGOV_SRC/SRC/temp/20180517000182/assert.cpp_538942250642250891.cpp
397 22 <TRACE : 5000> [C Preprocessor]-preprocessed file - I:/APPGOV_SRC/SRC/temp/20180517000182/assert.cpp_538942250642250891.cpp
398 22 <TRACE : 5000> [C Preprocessor]-error file - I:/APPGOV_SRC/SRC/temp/20180517000182/assert.cpp_538942250642250891.err
399 22 <DEBUG> TEMP_FILE_PREFIX -I:/APPGOV_SRC/SRC/temp/20180517000182/assert.cpp_538942250642250891
400 22 <DEBUG> INCLUDE FILE NOT FOUND -cassert:S_LINE: 1, S_COL: -1, E_LINE: 1, E_COL: -1, FILEPATH: I:/APPGOV_SRC/SRC/src/20180517000181/CR/9566/
401 22 <DEBUG> INCLUDE FILE NOT FOUND -cassert:S_LINE: 2, S_COL: -1, E_LINE: 2, E_COL: -1, FILEPATH: I:/APPGOV_SRC/SRC/src/20180517000181/CR/9566/a
402 22 <INFO> Structure-Analysis-[33101]/cp/source/DoNotReturnArrayAddrVisitor.cpp

```

How to switch JDK

If you have installed beSOURCE Server with Java 1.7, but want to use Java 1.8, do the following:

NOTE: These steps assume you have installed Java 1.7 and Java 1.8 on your computer. For example, 'C:\Java\jdk1.7.0' and 'C:\Java\jdk1.8.0'.

1. Open the **beSOURCEInstallFolder\analyzer\server\CE** folder.
2. Using a text editor (for example, Notepad), open the **startServer.cmd** or **startServer.sh** file.

```

1 echo Starting Analyzer on Stand-alone mode...
2 echo off
3
4 set JAVA_HOME=C:\Java\jdk1.7.0
5 set CM_HOME=C:\beSOURCE
6 set CE_HOME=%CM_HOME%\analyzer
7 set CLASSPATH=%CE_HOME%\lib\internal\com.itplus.cm.startup.jar
8
9 @rem set PATH=
10 set PATH=%CE_HOME%\lib\usa\windows;%PATH%
11 @rem set JPDA_OPTS=-Xdebug -Xrunjdp:transport=dt_socket,address=9999,server=y,suspend=n
12
13 "%JAVA_HOME%\bin/java" -DVER=-DCE_HOME=%CE_HOME% -DCM_HOME=%CM_HOME% -DJAVA_HOME="%JAVA_HOME%" %EXTRACTOR_OPTION% -
14

```

3. Change the **set JAVA_Home=C:\Java\jdk1.7.0** path of the Analysis Server to **set JAVA_Home=C:\Java\jdk1.8.0**, and then save your changes.

```

1 echo Starting Analyzer on Stand-alone mode...
2 echo off
3
4 set JAVA_HOME=C:\Java\jdk1.8.0
5 set CM_HOME=C:\beSOURCE
6 set CE_HOME=%CM_HOME%\analyzer
7 set CLASSPATH=%CE_HOME%\lib\internal\com.itplus.cm.startup.jar
8
9 @rem set PATH=
10 set PATH=%CE_HOME%\lib\usa\windows;%PATH%
11 @rem set JPDA_OPTS=-Xdebug -Xrunjdp:transport=dt_socket,address=9999,server=y,suspend=n
12
13 "%JAVA_HOMES%\bin/java" -DVER=-DCE_HOME=%CE_HOME% -DCM_HOME=%CM_HOME% -DJAVA_HOME="%JAVA_HOME%" %EXTRACTOR_OPTION% -
14

```

4. Open the **beSOURCE Install Folder\WAS\bin** folder.
5. Using a text editor, open the **catalina.bat** or **catalina.sh** file.

```

79 rem           Example (all one line)
80 rem           set TITLE=Tomcat.Cluster#1.Server#1 (%DATE% %TIME%)
81 rem -----
82 set JAVA_HOME=C:\Java\jdk1.7.0
83 set CATALINA_OPTS=-Xms256m -Xmx256m -Djava.net.preferIPv4Stack=true
84
85 rem Guess CATALINA_HOME if not defined
86 set "CURRENT_DIR=%cd%"
87 if not "%CATALINA_HOME%" == "" goto gotHome
88 set "CATALINA_HOME=%CURRENT_DIR%"
89 if exist "%CATALINA_HOME%\bin\catalina.bat" goto okHome
90 cd ..

```

6. Change the **set JAVA_Home=C:\Java\jdk1.7.0** path of the View Server to **set JAVA_Home=C:\Java\jdk1.8.0**, and then save your changes.

```

79 rem           Example (all one line)
80 rem           set TITLE=Tomcat.Cluster#1.Server#1 (%DATE% %TIME%)
81 rem -----
82 set JAVA_HOME=C:\Java\jdk1.8.0
83 set CATALINA_OPTS=-Xms256m -Xmx256m -Djava.net.preferIPv4Stack=true
84
85 rem Guess CATALINA_HOME if not defined
86 set "CURRENT_DIR=%cd%"
87 if not "%CATALINA_HOME%" == "" goto gotHome
88 set "CATALINA_HOME=%CURRENT_DIR%"
89 if exist "%CATALINA_HOME%\bin\catalina.bat" goto okHome
90 cd ..

```

7. Stop the **Analysis Server**, and then restart it.
8. Stop the **View Server**, and then restart it.