Application Coverage for Standalone Applications

In this section:

Overview

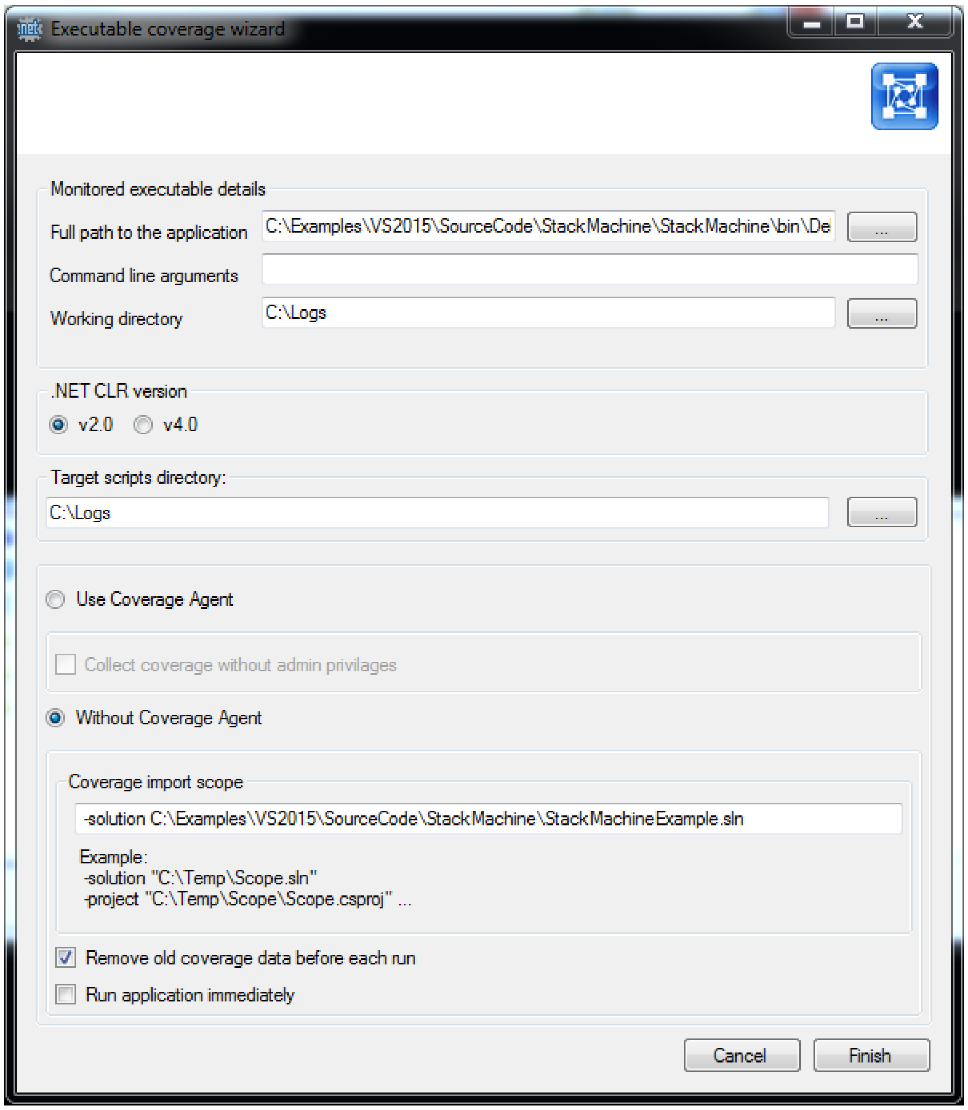
dotTEST DTP Engine ships with the `coverage.exe` tool that facilitates collecting coverage information from .NET managed code during execution of standalone applications. Running `coverage.exe` launches a wizard that allows you to specify the necessary information in the GUI. Alternatively, you can provide the information with the dedicated command line options.

Additionally, dotTEST ships with a component called the coverage agent, which can be enabled or disabled when coverage information is being collected. Enabling the coverage agent allows you to connect with the Coverage Agent Manager (CAM) - a web interface for measuring coverage with a wide range of testing techniques; see the Coverage Agent Manager (CAM) section of the DTP documentation for details.

Both workflows, with and without the coverage agent, enable you to generate a local coverage report and upload the results to DTP.

Configuring the Coverage Wizard

1. Run the `[INSTALL_DIR]\coverage.exe` tool and specify the following information:
   - Full path to the application
   - Command line arguments
   - Working directory


Executable coverage wizard
Monitored executable details
Full path to the application
C:\Examples\VS2015\SourceCode\StackMachine\StackMachine\bin\De
Command line arguments
Working directory
C:\Logs
.NET CLR version
v2.0
v4.0
Target scripts directory:
C:\Logs
Use Coverage Agent
Collect coverage without admin privilages
Without Coverage Agent
Coverage import scope
-solution C:\Examples\VS2015\SourceCode\StackMachine\StackMachineExample.sln
Example:
-solution "C:\Temp\Scope.sln"
-project "C:\Temp\Scope\Scope.csproj" ...
Remove old coverage data before each run
Run application immediately
Cancel
Finish

2. Specify the .NET CLR version used by the application.
3. Specify a directory where the scripts generated by the wizard will be saved.
4. Specify if you want to collect coverage information with or without the Coverage Agent:
   - If you enable the **Use Coverage Agent** option, enable or disable the **Collect coverage without admin privileges** option (see Collecting Coverage without Admin Privileges), then skip steps 5-7 and click **Finish**.
   - If you enable the **Without Coverage Agent** option (default), proceed to steps 5-7.
5. Define the scope of coverage to import by providing `-solution` or `-project` switches that will be passed to the dotTEST executable so that it can locate sources. See Configuring the Test Scope for more information about the switches.
6. Enable or disable the **Remove old coverage data before each run** option. This option is enabled by default.

   By default, dynamic coverage information that was collected during previous executions is deleted. Disabling this option will prevent removing the data from previous runs.

7. Enable or disable the **Run application immediately** option to automatically launch the `monitorCoverage.bat` script (if enabled, the application must be manually launched with the script generated by the wizard in the location specified in step 3).

Using the Command Line Options

You can run `coverage.exe` with the dedicated command line options to specify the information necessary for collecting application coverage:

```
coverage.exe -app [path] -workingDir [path] -appArgs [arguments] -commandsDir [path] -scope [dotTEST scope
switch] -clr [version] -run
```

The following options are available:

Option Name	Value	Description
`-app`	`[path]`	Specifies the full path to the application.
`-workingDir`	`[path]`	Specifies the path to the working directory.
`-appArgs`	`[argument]`	Specifies the application command line arguments.
`-commandsDir`	`[path]`	Specifies the directory for the *.bat scripts generated by the coverage tool.
`-scope`	`-solution [path]` `-project [path]`	Specifies the scope of the coverage to import by providing `-solution` or `-project` switches that will be passed todotTEST executable so that it can locate sources. For example: `-scope "-solution C:\temp\Scope.sln"` `-scope "-project C:\temp\Scope.csproj"` See Configuring the Test Scope for more information about the switches.
`-clr`	`v20` `v40`	Specifies the .NET CLR version used by the application;
`-run`	`none`	Automatically launches the `monitorCoverage.bat` script (optional; you can also open the directory specified with the `-commandDir` option and manually run the script).
`-doNotRemoveOldCoverage`	`none`	Specifies that old coverage data is not deleted before each run. If this option is not specified, coverage data collected during previous runs is deleted by default.
`-withCoverageAgent`	`none`	Specifies that coverage is collected with the Coverage Agent.

Collecting Coverage without Coverage Agent

If you configure the coverage wizard to run without the coverage agent, the tool creates the following scripts in the directory specified as the **Target scripts directory** (see Configuring the Coverage Wizard):

- `monitorCoverage.bat` runs and monitors the application
- `importCoverage.bat` imports and reports coverage data

1. Run the `monitorCoverage.bat` script (skip this step if you launched the coverage wizard with the **Run application immediately** option enabled). This will run the specified application.
2. Interact with the application and perform your tests.
3. Close the application.
4. Run the `importCoverage.bat` script to generate the report in the following location: [Target scripts directory]\reports.
5. View the coverage information.

Collecting Coverage with Coverage Agent

If you configure the coverage wizard to run with the coverage agent, the tool creates the following scripts in the directory specified as the **Target scripts directory** (see Configuring the Coverage Wizard):

- `monitorCoverage.bat` runs and monitors the application
- `runCamAgent.bat` runs the Coverage Agent

1. Run the `runCamAgent.bat` script to launch the Coverage Agent.

   Do not close the Coverage Agent process until you stop working with CAM.

2. Run the `monitorCoverage.bat` script to launch the application.
3. Connect with Coverage Agent Manager and perform testing.
4. Download coverage and test results; see Coverage Agent Manager (CAM) section of the DTP documentation for details.
5. Close the application.
6. Upload the coverage and test results to DTP, see Uploading Test Results and Coverage to DTP.
7. View the coverage information in DTP.

Collecting Coverage without Admin Privileges

By default, dotTEST requires admin privileges to collect coverage with Coverage Agent, but you can configure the tool to collect coverage by users without admin privileges.

1. Enable the **Collect coverage without admin privileges** option (see Configuring the Coverage Wizard) and launch the wizard. As a result, two additional .bat scripts will be generated:
   - runCamAgentInit.bat
   - runCamAgentUninit.bat
2. Run the `runCamAgentInit.bat` script with admin privileges to setup and initialize the Coverage Agent. This will enable the users without admin privileges to perform testing with CAM, as described in Collecting Coverage with Coverage Agent.
3. When testing is completed, run the `runCamAgentUninit.bat` script to stop the process.

Uploading Test Results and Coverage to DTP

To view coverage information collected with CAM, you need to upload the test results and coverage to DTP.

Uploading Test Results

1. Go to **Report Center** in the DTP interface.
2. Click the gear icon and choose **Report Center Settings> Additional Settings> Report Center Administration> Tools> Data Collector Upload Form**.
3. Click **Choose File** and browse for the report.xml file that you downloaded from CAM.
4. Click the **Upload** button to upload the file to DTP.

Uploading Coverage

To upload the coverage to DTP, you need to merge the runtime coverage that you downloaded from CAM with the static coverage generated by the Coverage Agent. To properly merge and upload the coverage data, ensure that the required settings are configured in the .properties file (see Configuration Overview):

- Ensure that dotTEST DTP Engine is properly configured, including DTP, scope and authorship settings. See Connecting to DTP, Sending Results and Publishing Source Code to DTP, Configuration.
- Configure the following settings in the dottestcli.properties file in order to properly merge coverage data:
  - `report.coverage.images` - Specifies a set of tags that are used to create coverage images in DTP. A coverage image is a unique identifier for aggregating coverage data from runs with the same build ID. DTP supports up to three coverage images per report.
  - `session.tag` - Specifies a unique identifier for the test run and is used to distinguish different runs on the same build.
  - `build.id` - Specifies a build identifier used to label results. It may be unique for each build, but it may also label several test sessions executed during a specified build.
  - `report.coverage.limit` (optional) - Specifies the lower coverage threshold. Coverage results lower than this value are highlighted in the report (the default value is `40`).

Generating the Static Coverage File

Generate the static coverage file by running the following test configuration on the solution:

```
dottestcli.exe -config "builtin://Collect Static Coverage" -solution SOLUTION_PATH
```

The dottestcli console output will indicate where the static coverage data is saved:

```
Saving static coverage information into: 'C:\Users\[USER]\Documents\Parasoft\dotTEST\Coverage\Static\[FILE].data
```

Merging the Coverage Data and Uploading to DTP

Copy the runtime coverage and static coverage files to a directory on the same machine and run `dottestcli` using the `-runtimeCoverage` and `-staticCoverage` switches to specify the location of the files:

```
dottestcli.exe -runtimeCoverage [path] -staticCoverage [path] -report [path] -publish -settings [path]
```

The coverage data will be automatically uploaded to DTP.

Reviewing Coverage in DTP

You can use the Coverage Explorer in DTP to review the application coverage achieved during test execution. See the DTP documentation for details on viewing coverage information.