INTEGRATING WITH AZURE DEVOPS

Abstract

You can integrate IBM Rational Test Workbench with Azure DevOps so that you can create tests for your application in Rational Test Workbench desktop clients and run those tests in Azure DevOps pipelines.

IBM

Contents

Running IBM Rational Test Workbench Tests in Azure DevOps Pipelines

When you use Azure DevOps for continuous integration and continuous deployment of your application, you can create tests for your application in Rational Test Workbench and run those tests in Azure DevOps pipelines.

Before you begin

- You must have installed any or all of the desktop clients of Rational Test Workbench.
- You must have set up your services for using Azure DevOps and set up your organization and project for running jobs in Azure DevOps Pipelines.
- Depending on the type of tests you want to perform on your application, you must have created any or all of the following types of tests in the desktop clients for the application you are testing:

Type of test	Desktop client
• Accelerated Functional Testing (AFT) suites • Web UI tests • Compound tests • Traditional Functional tests	IBM Rational Functional Tester
• API tests • API Test Suites	IBM Rational Integration Tester
• Compound tests • Performance tests • Schedules	IBM Rational Performance Tester

About this task

You can use the IBM Rational Azure DevOps Extension that enables you to select any type of test created in Rational Test Workbench desktop clients that you can add to your testing job in the Azure DevOps Pipelines.

Task flow

Follow the tasks that you must perform in sequence as listed in the following table. The table also provides you the links to the information about the tasks.

	Task	**More information**
1	Download the IBM Rational Azure DevOps Extension from the Test Workbench Resource Community website.	Test Workbench Resource Community (http://www.testworkbench-community.com/resources.html)
2	Install the IBM Rational Azure DevOps Extension as an **Extension** in your **Organization** in Azure DevOps.	Installing the IBM Rational Azure DevOps Extension as an Azure DevOps Extension
3	Create tests in any or all of the IBM Rational Test Workbench desktop clients to test your application.	• Rational Functional Tester V10.0.2 documentation • Rational Integration Tester V10.0.2 documentation • Rational Performance Tester V10.0.2 documentation
4	Run tests in an Azure DevOps Pipeline	Running tests in an Azure DevOps Pipeline

Disclaimer

The steps for many tasks that are to be performed on third-party applications are only indicative. There might be multiple methods in which you can accomplish the tasks indicated in this document. You might be already aware of such methods or you can perform them by using the on-screen instructions or documentation provided by third-party applications.

Installing the IBM Rational Azure DevOps Extension as an Azure DevOps Extension

You must install the IBM Rational Azure DevOps Extension in your Azure DevOps organization before you can use the extension for running your application tests in an Azure DevOps pipeline. The IBM Rational Azure DevOps Extension supports the running of the tests created in IBM Rational Test Workbench desktop clients.

Before you begin

Important:

The third-party applications that are used in this procedure are compatible with MS Windows OS. You can use alternative applications that are compatible with the OS you are running on your computer.

- You must have downloaded, and extracted or installed the following components on your computer:
    - The IBM Rational Azure DevOps Extension
    - Node.js
    - tfx-cli
- You must have access to the following applications:
    - The Visual Studio Marketplace.
    - Any Git repository to which you can upload the compiled IBM Rational Azure DevOps Extension.

About the task

After you download and extract the IBM Rational Azure DevOps Extension to your local repository, you can create a publisher in **Visual Studio Marketplace**. You must change the `vss-extension.json` file in the Extension package for the `publisher` attribute. Then compile extension package and upload the compiled package to the publisher in **Visual Studio Marketplace**.

You can commit the extension package to the remote repository that you use and specify the remote repository as the source code location in the publisher. You must share the extension in **Visual Studio Marketplace** so that the extension can be added as an extension in your organization in Azure DevOps.

Procedure

1. Log in to **Visual Studio Marketplace** and complete the following steps to create a publisher:

   Note:

   The steps to create a publisher are provided to guide you but you can also create a publisher by following the instructions on the **Visual Studio Marketplace** pages.

   a. Click **Publish extensions**.

      The **Create Publisher** window is displayed.

   b. Specify a name for the publisher in the **Name** field.

   c. Specify an identifier for your publisher.

      Important:

      Note down the identifier that you specify because you must use this value for the `publisher` attribute in the manifest file (`vss-extension.json`) of the IBM Rational Azure DevOps Extension.

   d. Review the Marketplace Publisher Agreement, and then click **Create**.

      You have created a Publisher in the **Visual Studio Marketplace**.

   Note:

> You might be notified of the actions that you perform in your Azure DevOps organization by email on the email address that you registered in Azure DevOps.

2. Open the `vss-extension.json` file from the folder where you extracted the IBM Rational Azure DevOps Extension package. Complete the following steps:

   a. Edit the file to change the value of the `publisher` attribute to the identifier you specified for the publisher in **Visual Studio Marketplace**.

   b. Save the `vss-extension.json` file.

   c. Install or update TFS Cross Platform Command Line Interface (tfx-cli) by using the following command from the command line:

   ```
   npm i -g tfx-cli
   ```

   d. Compile or package the modified IBM Rational Azure DevOps Extension as a `.vsix` file by running the following command from the location that contains the extracted IBM Rational Azure DevOps Extension package:

   ```
   tfx extension create -–manifest-globs vss-extension.json
   ```

   You have created the modified extension package for uploading to the Visual Studio Marketplace.

3. Open the **Visual Studio Marketplace** portal in a browser and complete the following steps to upload the extension package to the publisher you created:

   a. Click **New Extension** and select **Azure DevOps**.

   b. Browse to locate and **Open** the packaged extension that has the `.vsix` file extension, and then click **Upload**.

   The IBM Rational Azure DevOps Extension is published to the **Virtual Studio Marketplace**.

4. Commit the IBM Rational Azure DevOps Extension files from your local repository to the remote repository.

5. Return to the **Visual Studio Marketplace** portal and complete the following steps:

   a. Click the **More actions** icon in the IBM Rational Azure DevOps Extension row and click **Share/Unshare**.

   b. Click **Organization** and enter the name of the organization in Azure DevOps to which you want to share the extension, and then press **Enter**.

      The organization you shared the extension with is displayed.

   c. Click the **Details** tab and specify the path of the remote repository where you have committed the Extension in the **Source code repository** field.

   d. Click **Save** to save the changes you made.

6. Open your **Organization** page in Azure DevOps and complete the following steps:

   a. Click **Organization settings**.

   b. Click **Extensions**.

   c. Click **Shared**.

      The IBM Rational Azure DevOps Extension that you shared with the organization in the **Visual Studio Marketplace** is displayed as a shared Extension.

   d. Click the **IBM Rational Azure DevOps Extension** that is displayed.

   e. Click **Install**.

      The **Visual Studio Marketplace** portal is displayed.

   f. Select the organization in Azure DevOps from the list in which you want to install the extension, and then click **Install**.

      > **Note:**
      >
      > The organizations in Azure DevOps with which you shared the extension are displayed for selection in the list.

The IBM Rational Azure DevOps Extension is installed as an Azure DevOps extension in Azure DevOps organization.

g. Return to the **Organization** page in Azure DevOps, click **Organization settings** > **Extensions**.

   The IBM Rational Azure DevOps Extension is displayed as an installed extension.

Results

You have successfully installed the IBM Rational Azure DevOps Extension as an extension in your Azure DevOps Organization.

What to do next

You can run the tests for the application you are testing that you created in the Rational Test Workbench desktop clients as a job in Azure DevOps pipelines. See Running tests in an Azure DevOps Pipeline.

Running tests in an Azure DevOps Pipeline

After you create the tests in IBM Rational Workbench desktop clients for the application that you are testing and install the IBM Rational Azure DevOps Extension in your organization, you can run the tests as a job in Azure DevOps pipelines.

Before you begin

You must have completed the following tasks:

- Created an organization and a project in Azure DevOps. For more information, refer to Create an organization.
- Committed the modified IBM Rational Azure DevOps Extension extracted files to a remote repository.
- Installed the IBM Rational Azure DevOps Extension in your organization. For more information, see Installing the IBM Rational Azure DevOps Extension as an Azure DevOps Extension.
- Installed an agent in your pipeline. For more information, refer to Azure Pipelines agents.

About the task

After you add the IBM Rational Azure DevOps Extension as an extension in your Azure DevOps organization, you can use an existing pipeline or create a new one to add the IBM Rational Workbench desktop client test tasks. You can install an agent or the one that you have installed in your default agent pool. You can add the IBM Rational Workbench desktop client tests as tasks to your agent job, configure the task, and then run the task in the Azure DevOps pipeline.

Procedure

1. Open your **Organization** page in Azure DevOps and complete the following steps:
   a. Click the project you want to use.
   b. Click **Pipelines**.
   c. Click **Use the classic editor** to create a pipeline without YAML.
   d. Select the repository where you have uploaded the IBM Rational Azure DevOps Extension and complete the addition of the repository.
   e. Search for the **Node.js with gulp** template, select it, and click **Apply**.

      The **Node.js with gulp** template page is displayed.
2. Select the pipeline and complete the following steps:
   a. Change the name for the build pipeline, if required.
   b. Select the **Agent pool** for your build pipeline. You can use the agent from the default agent pool or use the one you have installed.
   c. Select the **Agent specification** for the agent.
   d. Optionally, select the tasks from the list that are not required in your job and remove them.
   e. Optionally, change the name of the agent job.
3. Add a task to the agent job by completing the following steps:
   a. Click the **Add task** + icon for the agent job.

      The Add tasks pane is displayed.
   b. Enter `ibm` in the **Search** box to search for the tasks defined in the IBM Rational Azure DevOps Extension.

      The tasks that you can select are displayed.

      Depending on the type of test that you have created in the desktop client you can select the type of task. Use the following table to identify the task you must select:

Type of test	Task to select
• AFT suites • Web UI tests • Compound tests • Traditional Functional tests	IBM Rational UI Azure Task
• API tests • API Test Suites	IBM Rational API Azure Task
• Compound tests • Performance tests • Schedules	IBM Rational Performance Azure Task

c. Select the identified task and then click **Add** to add the task to the agent job.

The selected task is added to the agent job and displayed with a warning that some settings require attention.

4. Complete the settings for the selected task:

a. Select the task version from the list.

b. Follow the action that is applicable for the task you selected, marked with a checkmark ✓ in the following table:

Note:

All mandatory fields are marked with an asterisk (*) in the UI.

Field	**Description**	**Action**	**Azure Task**			
			API	**Performance**	**UI**	
					Traditional	**Web UI**
Display name	Displays the name of the selected task. For example, IBM Rational Performance Azure Task	Optionally, change the name.	✓	✓	✓	
Configuration Type	The type of test to run.	Select from the list.	✓			
Testcase Type	The type of test to execute.	Select from the list: • Select **Traditional (Java/VB)** to run the functional test. • Select **WebUI** to run the WebUI test.				
Product Path	The path where the desktop client is installed on your computer.	Enter the complete path to the location of the desktop client. For example, if you are testing an API test or suite created in Rational Integration Tester, the path can be: `C:\Program Files\IBM\RationalIntegrationTester`	✓	✓	✓	✓

Field	**Description**	**Action**	**Azure Task**			
			API	**Performance**	**UI**	
					Traditional	**Web UI**
Project Directory	The path where the project containing the test is located on your computer.	Enter the complete path to the location of the project containing the test. For example, `D:\Projects\`	✓		✓	
IMShared Path	The path to the IM Shared folder on your local computer.	Enter the complete path to the location of the IBM IM Shared folder. For example, `C:\Program Files\IBM\IBMIMShared\`		✓		✓
Workspace Location	The path to the project workspace on the computer where the performance test agent is installed.	Enter the complete path to the location where the performance test agent is installed.		✓		✓
Project Name	The name of the project containing the test.	Enter the name of the project containing the test.	✓	✓		✓
Environment	The API test environment to use in the test run.	Enter the name of the environment defined in the test asset.	✓			
Test to run	The name of the test or test suite to run.	Enter the name of the test or test suite that you want to run.	✓			
Test Suite Name	The name of a test within the project to use. A test can be WebUI test, Compound test, Performance schedule or AFT suite.	Enter the name of the test that you want to run.		✓	✓	✓
Log Format	The format of the script logs.	Select the format of the script log from the list.			✓	
Iteration Count	The number of datasets iterations to be run.	Enter the number that you want for the dataset iterations.			✓	
User Arguments	You can specify any arguments that you want to run in the test.	Specify any arguments for the test run. For example, you can specify the playback arguments, if applicable in your test.			✓	

c. Optionally, expand the **Control Options** and configure the settings for your test task.

d. Optionally, expand the **Output Variables** and configure the settings for your test task.

5. Select from the following options:
   - Click **Save** to save the configured settings for the task. The task is not queued for a run.

     You can save the task to a build pipeline and opt to run the build at a later time.
   - Click **Save & queue** to save the configurations and queue the run in the pipeline.

     The **Run pipeline** dialog box is displayed.
6. Complete the following steps:

   a. Enter a comment for the test in the **Save comment** field.

   b. Select the agent that you configured for the test from the **Agent pool** list.

c. Select the branch from the **Branch/tag** list.
d. Optionally, add the variables and demands for the task run from the **Advanced Options** pane.
e. Select the **Enable system diagnostics** check box.
f. Click **Save and run**.

Results

You have run the tests for the application you are testing as a job in the Azure DevOps pipeline.

What to do next

You can view the progress of the job run on the Azure DevOps dashboard.

After the job completes, you can open the job task log to view the details. For IBM Rational Azure Performance tasks, the log file also displays the command line commands that were run on the performance agent.