CodeArts PerfTest

Best Practices

Issue	01
Date	2025-01-07

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Huawei Cloud Computing Technologies Co., Ltd.

Address:	Huawei Cloud Data Center Jiaoxinggong Road Qianzhong Avenue Gui'an New District Gui Zhou 550029 People's Republic of China
Website:	https://www.huaweicloud.com/intl/en-us/

Contents

1 CodeArts PerfTest Best Practices

This document summarizes common CodeArts PerfTest operation practices and provides solutions and operation guide to help you easily use CodeArts PerfTest.

Table 1-1 CodeArts PerfTest best practices

Practice	Description
Performance Tests of All-in-One Systems for Government Services	Systems related to national economy and people's livelihood, such as all-in-one systems, respond slowly due to sudden increase of access traffic. To prevent system breakdown during peak hours, CodeArts PerfTest simulates actual scenarios and quickly constructs pressure models to detect service performance bottlenecks in different pressure models.
Native Performance Pressure Test of JMeter Test Projects	CodeArts PerfTest supports the native JMeter engine. You can import JMeter scripts into JMeter projects in CodeArts PerfTest to quickly initiate high-concurrency performance tests with JMeter, and view complete performance test reports.
Using Global Variables	Global variables are used to build data sets and enrich test data. If a global variable is used in a packet of a request, variable values in the packet will be replaced with specified values during a pressure test.

2 Performance Tests of All-in-One Systems for Government Services

2.1 Overview

Scenarios

Systems related to the national economy and people's livelihood, such as all-in-one systems, have been launched in many cities. However, abrupt increases in traffic volumes may cause slow response and even system breakdown. For example:

- In some cities, vouchers are issued on the hour in an all-in-one app or applet.
- Most citizens declare individual income tax (IIT) on such apps or applets at a certain time (January to March each year).

These situations require high performance of all-in-one systems.

Solution Architecture

To prevent system breakdown in peak hours, CodeArts PerfTest provides simulated scenarios and constructs pressure models to quickly detect service performance bottlenecks.

The following simulation scenarios are provided:

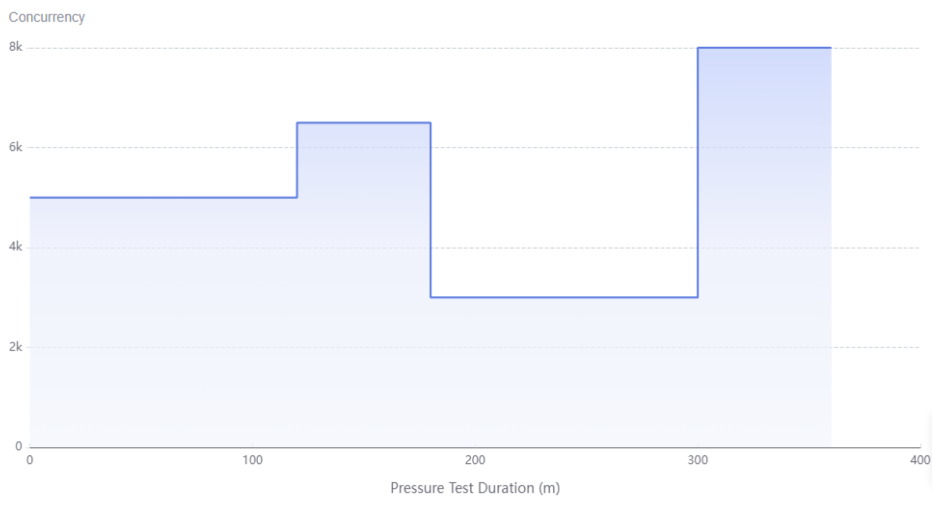
Scenario 1: access in peak hours

Large city (> 10 million people)

- Scenario analysis: The overall traffic gradually increases.
- Reference model and solution: Use the **Concurrency Mode** model to continuously increase the pressure phase by phase based on specifications to check whether the system performance meets the requirements.

  For example, the concurrency is 5000 from 7:00 to 9:00, 6500 from 9:00 to 10:00, 3000 from 10:00 to 12:00, and 8000 at restaurants from 12:00 to 13:00.

Figure 2-1 Model example 1



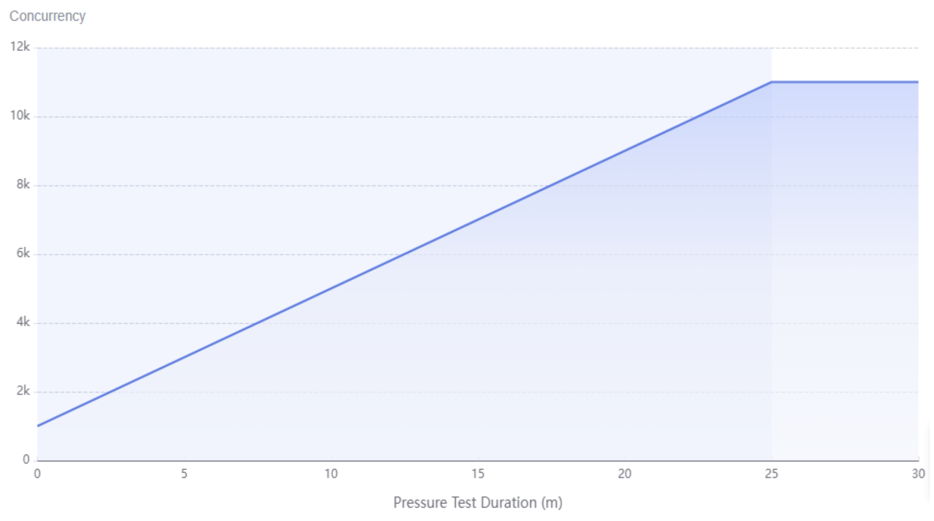
Scenario 2: declaring IIT at the beginning of the year

Large city (> 10 million people): A large number of citizens declare IIT from January to March.

- Scenario analysis: Continuous surge of ultra-large load occurs.
- Reference model and solution: Use the **Concurrency Mode** model.
  a. Apply the start load for a period of time.
  b. Apply a surge of load.
  c. Maintain the surge of load for a long period of time.

  For example, apply a start concurrency of 1000 for 10 minutes, then increase the concurrency to 10,000 (10 times the nominal load) and keep it for 120 minutes.

Figure 2-2 Model example 2



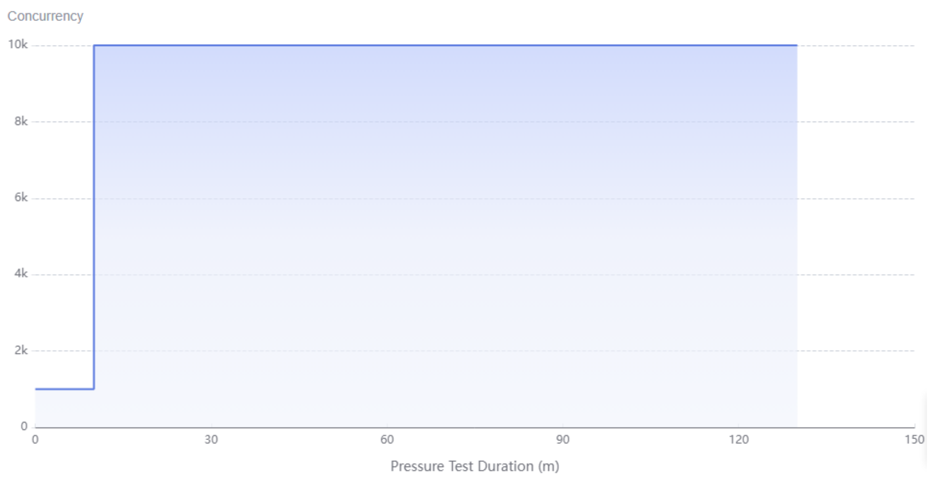
Scenario 3: performance limit investigation

Municipal governments can investigate the performance limits of all-in-one systems.

- Scenario analysis: When the traffic slowly increases to the bottleneck, the task will continue.
- Reference model and solution: Use the **Peakload Mode** model to gradually increase the pressure based on specifications to check whether the system performance meets the requirements.

  For example, set the start concurrency to 1,000, ramp up to 1,500 seconds (25 minutes), and the maximum concurrency to 11,000. The entire process lasts for 30 minutes.

Figure 2-3 Model example 3



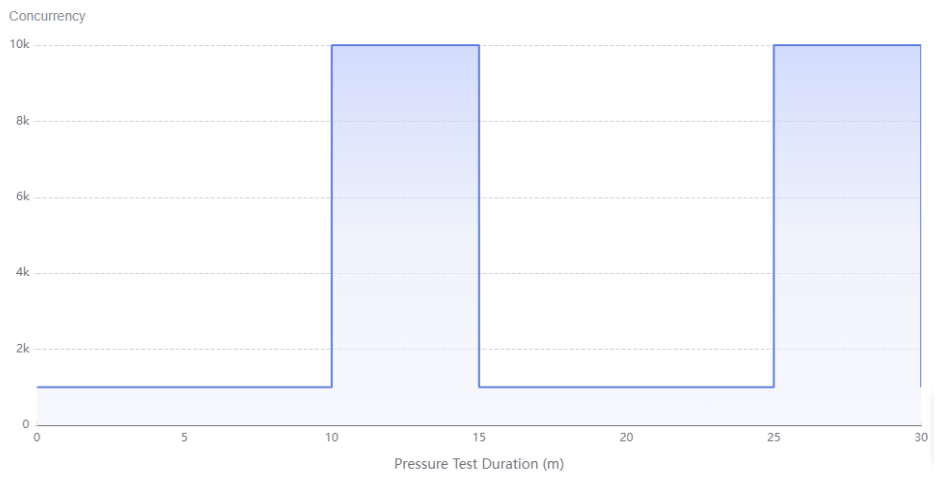
Scenario 4: claiming vouchers on the hour

Medium-sized city (2–10 million people): Claim vouchers at 12:00.

- Scenario analysis: A surge of load suddenly occurs.
- Reference model and solution: Use the **Surge Mode** model.
  a. Apply the start load for a period of time.
  b. Apply a surge of load.
  c. After a period of time, the load quickly reduces to the start concurrency and stay there for a while.

  For example, 10,000 people concurrently claim vouchers twice (for 5 minutes each). In the test, apply a start concurrency of 1000 for 10 minutes, then increase the concurrency to 10,000 and keep it for 5 minutes. Then repeat this process.

Figure 2-4 Model example 4



2.2 Operation Process

Perform the following operations before using CodeArts PerfTest:

- (Optional) **Basic Concepts of CodeArts PerfTest**: Understand some basic concepts.
- **Prerequisites**: Prepare an application to be tested.
- **Preparing Test Resources**: Purchase a proper CodeArts PerfTest package. Create a private resource group. If a shared resource group is used for testing, you do not need to create a resource group.
- **Creating, Debugging, and Starting a Task**: Create a pressure test task based on your service requirements. Multiple tasks can be executed concurrently.
- **Test Report Analysis**: View real-time reports and identify performance bottlenecks of an all-in-one system.

Basic Concepts of CodeArts PerfTest

- **Number of concurrent users**: number of users performing operations on a system at the same time. In CodeArts PerfTest, it is the number of virtual users you define when you configure a test phase.
- **RPS (QPS)**: average number of requests sent per second (RPS = Requests / Taken Time (s)).
- **Response Time**: duration from the time when a client sends a request to the time when the client receives a response from the server.

Prerequisites

Prepare the application to be tested in advance. This example only describes the pressure test method.

2.3 Procedure

Test Resource Groups and Their Constraints

- Test resource groups are classified into shared resource groups and private resource groups. Shared resource groups are provided by the system by default, and private resource groups need to be created.
- Execution nodes of the shared resource group have been bound with an elastic IP address (EIP). When the tested application has network access restrictions, use a private resource group.
- A shared resource group supports a maximum of 1,000 concurrent users and 100 Mbit/s bandwidth. If higher concurrency or bandwidth is required, use a private resource group.
- JMeter test tasks can use only private resource groups.

Preparing Test Resources

Step 1 Log in to the **CodeArts PerfTest console**, choose **Resource Groups** in the navigation pane, and click **Create Private Resource Group**.

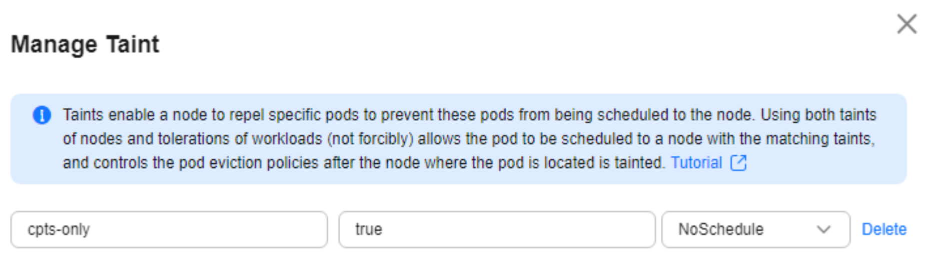
Step 2 (Optional) If this is the first time you create a private resource group, grant CodeArts PerfTest permissions necessary for creating private resource groups.

Step 3 If you do not have a Cloud Container Engine (CCE) cluster, **create a cluster** and then **create a resource group**. If a CCE cluster is available, **create a resource group**.

Step 4 Create a cluster.

Click **Create Cluster**. The page for buying CCE clusters is displayed. For details about how to create a cluster, see **Buying a CCE Cluster**. The cluster parameters are described as follows:

- You are advised to use an independent CCE cluster for pressure tests to avoid configuration conflicts with CCE clusters in test or production environments. When you conduct a pressure test on a CCE cluster in a test or production environment, service loads may be scheduled to nodes that function as executors. To prevent this, go to the cluster's **Nodes** page, click the **Nodes** tab, click **More** > **Manage Taint** in the **Operation** column of the execution node, and set the node as a taint, as shown below.


Manage Taint

Taints enable a node to repel specific pods to prevent these pods from being scheduled to the node. Using both taints of nodes and tolerations of workloads (not forcibly) allows the pod to be scheduled to a node with the matching taints, and controls the pod eviction policies after the node where the pod is located is tainted. Tutorial

cpts-only | true | NoSchedule | Delete

- The cluster management scale is related to the number of execution nodes. Create nodes of the corresponding specifications based on the number of concurrent users for pressure testing. For example, if 20 execution nodes are required, you can set the cluster scale to 50 nodes.

- You are advised to select **Tunnel network** as the network model. **Container CIDR Block** and **Service CIDR Block** must be the same as those of the tested object.
- CentOS is prone to IPVS- or Conntrack-related stability issues on a heavy-load network. Therefore, do not use CentOS as the operating system of cluster nodes when selecting IPVS. When both **IPVS** and **CentOS** are selected, network connection reuse may time out.

Click **Next: Select Add-on**. On the page displayed, select the add-ons to be installed during cluster creation. When selecting add-ons, retain the default settings for the test executor. For example, deselect unnecessary add-ons, such as NodeLocal DNSCache and Cloud Native Cluster Monitoring, to prevent the add-ons from occupying executor resources.

Click **Next: Add-on Configuration**. Retain the default settings.

Click **Next: Confirm configuration**. After confirming that the cluster configuration is correct, select **I have read and understood the preceding instructions** and click **Submit**. It takes about 6 to 10 minutes to create a cluster.

After the cluster is created, return to the cluster management page and click **Create Node**. For details about how to create a node, see **Creating a Node**.

The node parameters are described as follows:

- A node must have at least 4 vCPUs and 8 GB memory.
- Select EulerOS as the operating system.
- At least two nodes (one debugging node and one execution node) are required. The number of nodes depends on the specifications of the pressure test object. For example, 21 execution nodes (one debugging node and 20 execution nodes) are required for 100,000 concurrent users and 4 vCPUs | 8 GB memory.
- If the CCE cluster node and the application to be tested are not in the same VPC network, bind an EIP to the CCE cluster node. You can use existing EIPs. If no EIP is available, click **Auto create** to create one. When EIPs are automatically created, you are advised to pay by traffic and set the bandwidth to a large value to avoid affecting the pressure test. Additionally, an EIP with specified configurations is automatically assigned to each node. If the number of EIPs is less than the number of nodes, the EIPs are randomly bound to the nodes.
- In the **(Optional) Advanced Settings** area, set **Kubernetes Node Name** to **Node private IP**. If you select **Cloud server name**, the node cannot be managed.
- To enhance the security of a CCE node, see **Configuration Suggestions on CCE Node Security**.

Click **Next: Confirm**. After confirming that the node configuration is correct, select **I have read and understood the preceding instructions** and click **Submit**. After the node is created, return to the CodeArts PerfTest console.

Step 5 Create a resource group.

Choose **Resource Groups** in the navigation pane, and click **Create Private Resource Group**.

Set the parameters listed in **Table 2-1**.

Table 2-1 Creating a private resource group

Parameter	Description
Resource Group Name	Name of the private resource group to be created.
Debugging Cluster	Select a CCE cluster from the drop-down list.
Debug Node	Select the management node that performs a pressure test. The debugging node cannot be changed after the resource group is created.
Execution Node	Select a pressure target machine that provides performance data during a pressure test.

Click **Create**.

----End

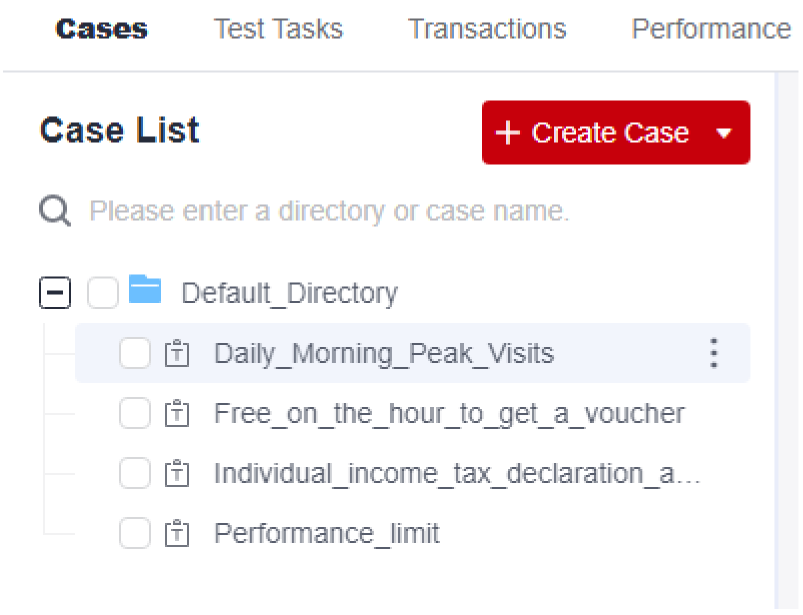
Creating, Debugging, and Starting a Task

Step 1 Log in to the CodeArts PerfTest console and select a region in the upper part of the page.

Step 2 In the navigation pane on the left, choose **Dashboard**. Click **All-in-one Scene** in the **Hot Template** to create a pressure test project.

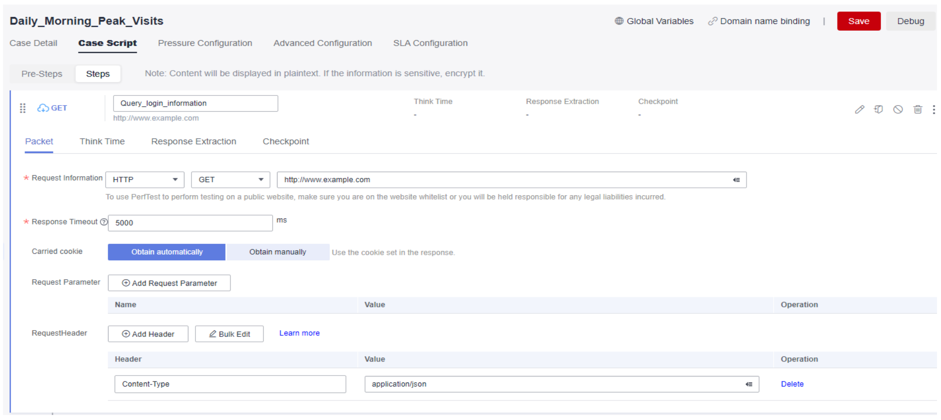
Step 3 After the test project is created, its case details page is displayed.

Figure 2-5 All-in-one system pressure test project


Cases
Test Tasks
Transactions
Performance
Case List
+ Create Case
Please enter a directory or case name.
Default_Directory
Daily_Morning_Peak_Visits
Free_on_the_hour_to_get_a_voucher
Individual_income_tax_declaration_a...
Performance_limit

Step 4 Select a test case and modify the parameters. For example, modify the request information in the **Performance_limit** scenario and click **Save**.

Figure 2-6 Modifying parameters



Note: Content will be displayed in plaintext. If the information is sensitive, encrypt it.

Name	Think Time	Response Extraction	Checkpoint
Query_login_information http://www.example.com	-	-	-

* Request Information: HTTP, GET, http://www.example.com

To use PerfTest to perform testing on a public website, make sure you are on the website whitelist or you will be held responsible for any legal liabilities incurred.

* Response Timeout: 5000 ms

Carried cookie: Obtain automatically | Obtain manually — Use the cookie set in the response.

Request Parameter

Name	Value	Operation

RequestHeader

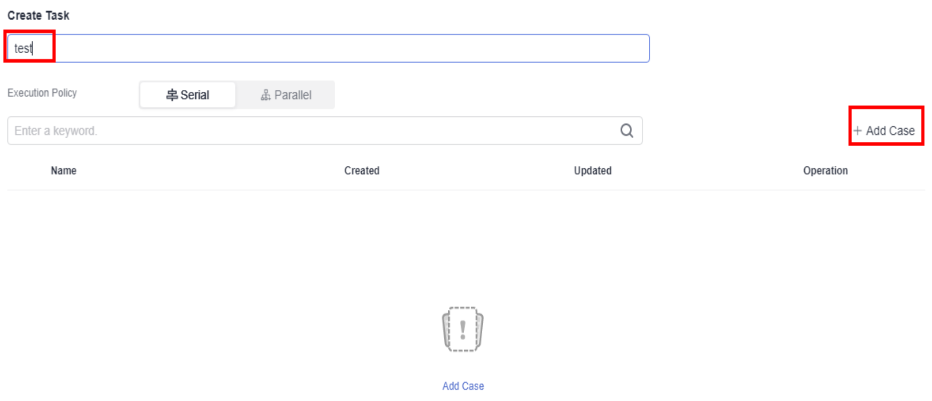
Header	Value	Operation
Content-Type	application/json	Delete

Step 5 Click **Debug**, select the target test resource group as the executor, and click **Start** to start debugging. If an error was reported in the debugging result, edit the case based on the log information and debug it again.

Step 6 On the **Test Tasks** tab, click **Create Task**.

Step 7 Enter a test task name, click **Add Case**. On the page that is displayed, select the target case, and click **OK**. Click **Save**. The test task is created.

Figure 2-7 Adding a case

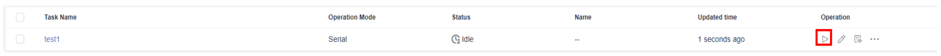


test

Name	Created	Updated	Operation

Step 8 Click the start icon in the **Operation** column of the test task.

Figure 2-8 Starting a task



Task Name	Operation Mode	Status	Name	Updated time	Operation
test1	Serial	Idle	--	1 seconds ago	▷ ✎ ⋯

Step 9 Select a resource group type and click **Start**.

----End

Test Report Analysis

Step 1 After you start a test task, click **View Report** in the displayed dialog box to view the real-time test report.

You can view the real-time monitoring data and chart reports of each metric during the pressure test.

Step 2 After the pressure test is complete, an offline test result report is generated automatically. You can view the report of the completed test task.

Step 3 You can also click **Download report** to download an offline PDF report to a local PC. Identify performance bottlenecks of the all-in-one system based on the report.

----End

3 Native Performance Pressure Test of JMeter Test Projects

3.1 Overview

Scenarios

When you use the local JMeter to perform a pressure test, various plug-ins are required to meet pressure test requirements. To view visual test reports, import JMeter projects to CodeArts PerfTest. CodeArts PerfTest supports the native JMeter engine. You can import JMeter scripts into JMeter projects in CodeArts PerfTest to quickly initiate high-concurrency performance tests with JMeter, and view complete performance test reports.

Notes and Constraints

- If you use CodeArts PerfTest test resources for the first time, authorize CodeArts PerfTest to perform operations on CCE and VPC endpoint (VPCEP).
- The CodeArts PerfTest container integrates the open-source Apache JMeter, CodeArts PerfTest's control code, and some enhanced JMeter capabilities (such as multi-phase pressure configuration and log output).
- By default, CodeArts PerfTest integrates Apache JMeter 5.4. You can upload a custom installation package to the CodeArts PerfTest resources to change the version to Apache JMeter 5.3 or 5.2. The custom installation package is a ZIP package downloaded from the Apache official website.
- If you want to use third-party plug-ins, upload them in the form of third-party JAR packages to the JMeter project in CodeArts PerfTest. This is equivalent to placing JAR packages in the JMeter root directory **\lib\ext**.
- If JMX scripts are deleted on the GUI, CodeArts PerfTest scripts stored in OBS are also deleted.

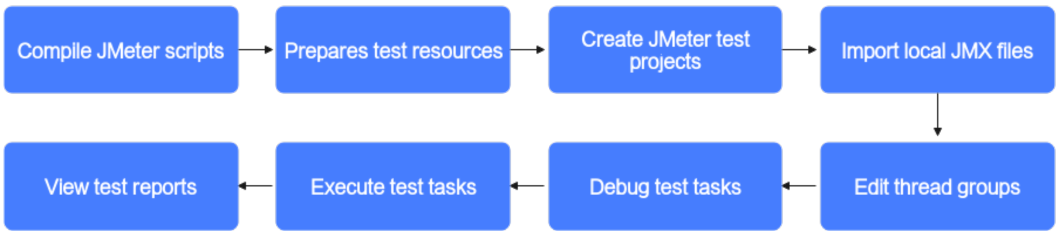
Solution Architecture

CodeArts PerfTest integrates the open-source Apache JMeter to implement pressure tests. The working principles are as follows:

1. You can use CodeArts PerfTest resources to manage CCE nodes (one debugging node and one or more execution nodes) of the tenant. Then, a VPC endpoint is created to upload test data to CodeArts PerfTest. CodeArts PerfTest starts a resident debugging workload through CCE. The workload starts a CodeArts PerfTest container for debugging.
2. Upload JMX scripts to CodeArts PerfTest before you use JMeter projects in CodeArts PerfTest. These scripts are stored in OBS buckets of CodeArts PerfTest. After the CodeArts PerfTest container is started, it downloads your scripts from OBS and executes them.
3. During task execution, a temporary execution load is started through CCE. The load starts one or more CodeArts PerfTest containers based on the task scale for execution. If multiple containers are used for execution, threads in the thread group are equally allocated to each CodeArts PerfTest container.
4. During task execution, the CodeArts PerfTest container uploads the test result to CodeArts PerfTest through VPCEP for the following purposes:
   - Collects performance metrics such as concurrency, latency, RPS, bandwidth, and TP90.
   - Displays request logs and return logs generated during execution.
5. When the task is complete, the CodeArts PerfTest container is destroyed, and only execution records are stored in the executor.

3.2 Operation Process

Figure 3-1 JMeter test project operations



1. Compile a local JMeter script and save it as a JMX file.
2. Prepare test resources and create a private resource group.
3. Create a JMeter test project.
4. Import the local JMX file.
5. Edit the thread group. You can set its parameters based on service requirements.
6. Debug a test task to quickly detect syntax or configuration errors and ensure that the model can be used in the task.
7. Perform the test task and obtain and analyze the performance data of the system running through the test.
8. View the test report. Real-time and offline JMeter test reports allow you to view and analyze test data at any time.

3.3 Procedure

Prerequisites

You have installed the JMeter client.

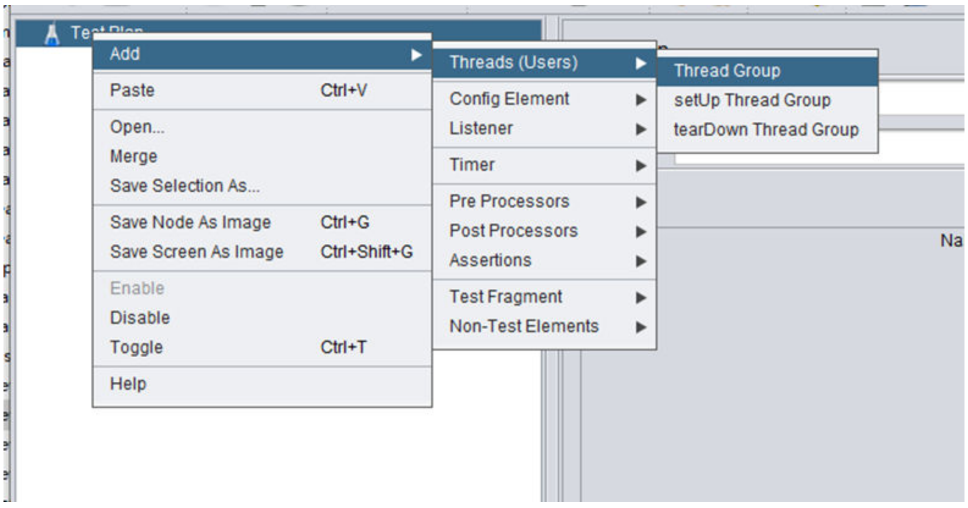
3.3.1 Compiling JMeter Scripts on Your Local PC

Step 1 Start the JMeter client.

Step 2 Create a thread group.

Right-click **Test Plan** in the upper left corner of the JMeter client and choose **Add** > **Threads (Users)** > **Thread Group** from the shortcut menu to create a thread group for the JMeter test plan. A thread group is the basic execution unit of a JMeter project.

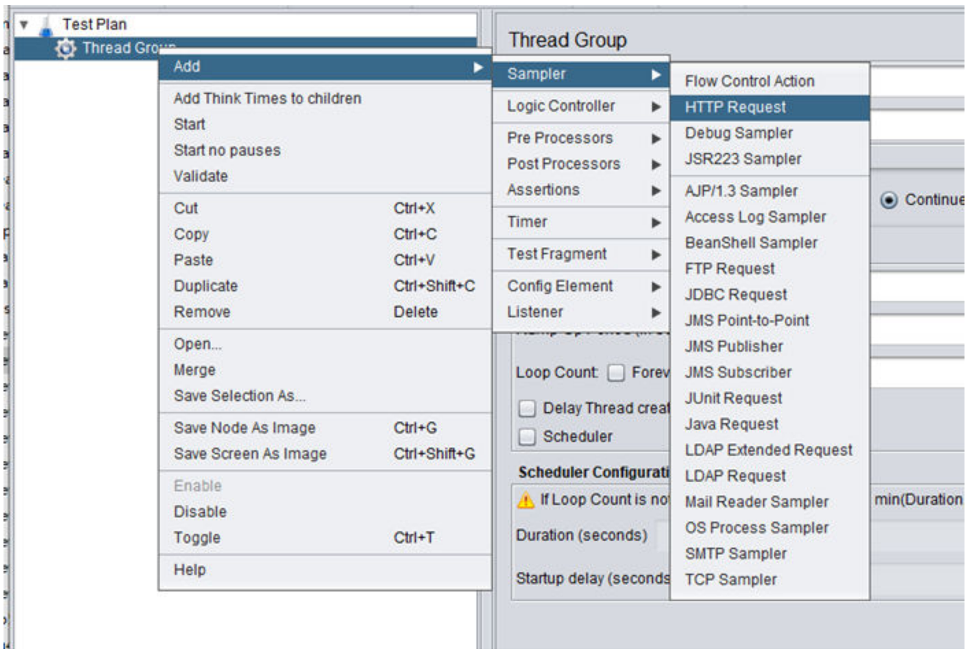
Figure 3-2 Creating a thread group



Step 3 Add a request.

Right-click **Thread Group**, choose **Add** > **Sampler** > **HTTP Request**, and add an HTTP request to the thread group.

Figure 3-3 Adding an HTTP request

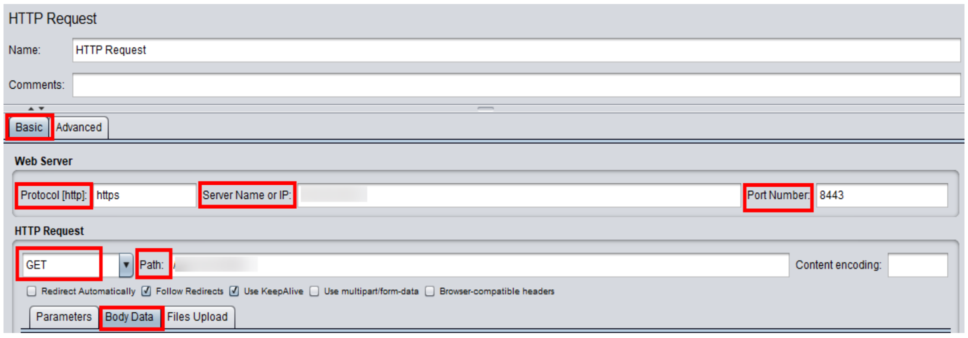


Step 4 Configure the HTTP request.

The configuration on the **Basic** tab page of the HTTP request is as follows:

- **Protocol**: The value can be **HTTP** or **HTTPS**.
- **Server Name or IP**: Enter a domain name or an IP address.
- **Port Number**: Enter the service port number. For HTTP, the default port number is **80**. For HTTPS, the default port number is **443**. You can also manually enter other ports.
- **HTTP Request**: Specify the method, such as GET, POST, PUT, and DELETE. If you select POST, add body parameters.
- **Path**: Enter the service request path.

Figure 3-4 Configurations on the Basic tab


HTTP Request

Name: HTTP Request

Comments:

Basic | Advanced

Web Server

Protocol [http]: https Server Name or IP: [redacted] Port Number: 8443

HTTP Request

GET Path: [redacted] Content encoding:

☐ Redirect Automatically ☑ Follow Redirects ☑ Use KeepAlive ☐ Use multipart/form-data ☐ Browser-compatible headers

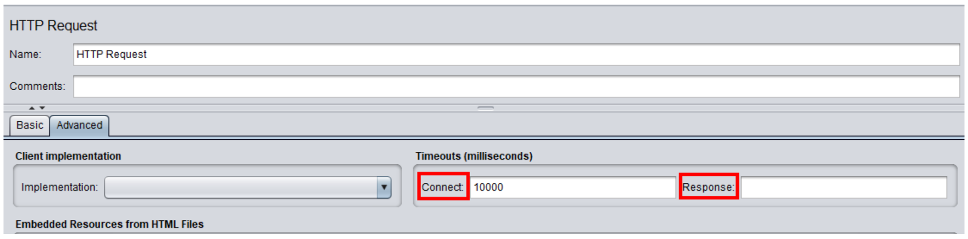
Parameters | Body Data | Files Upload

On the **Advanced** tab page of the HTTP request, set the timeout period in **Timeouts**.

- **Connect**: timeout period for connecting the client to the tested service. The default value is 20 seconds. You can set it to 10 seconds.

- **Response**: response time of the tested service after the connection. By default, there is no limit.

Figure 3-5 Configurations on the Advanced tab



HTTP Request

Name: HTTP Request

Comments:

Implementation:

Timeouts (milliseconds)

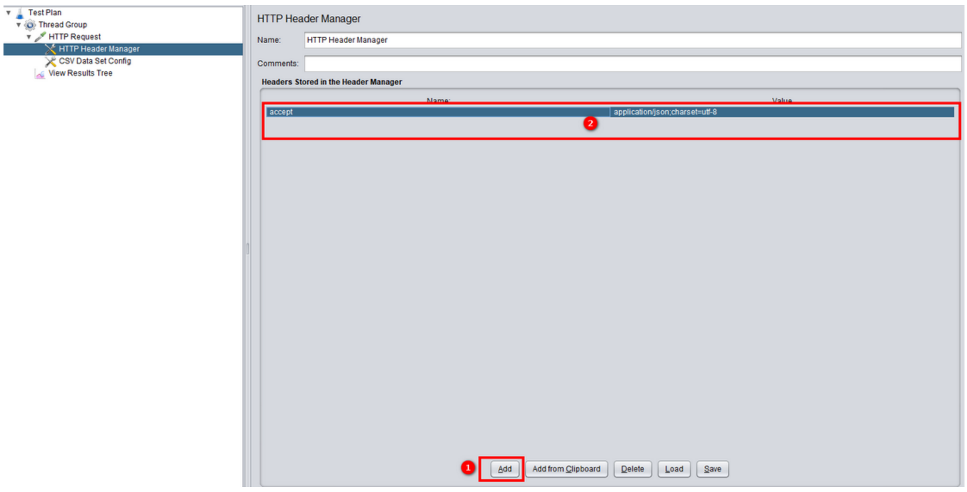
Connect: 10000

Response:

Step 5 Configure HTTP headers.

JMeter manages HTTP headers by thread group. Each thread group can be configured with an HTTP header manager. Right-click **Thread Group** and choose **Add** > **Config Element** > **HTTP Header Manager** to add an HTTP header manager. In the right pane of the **HTTP Header Manager**, click **Add** to add an HTTP Header.

Figure 3-6 Adding an HTTP header



HTTP Header Manager

Name: HTTP Header Manager

Comments:

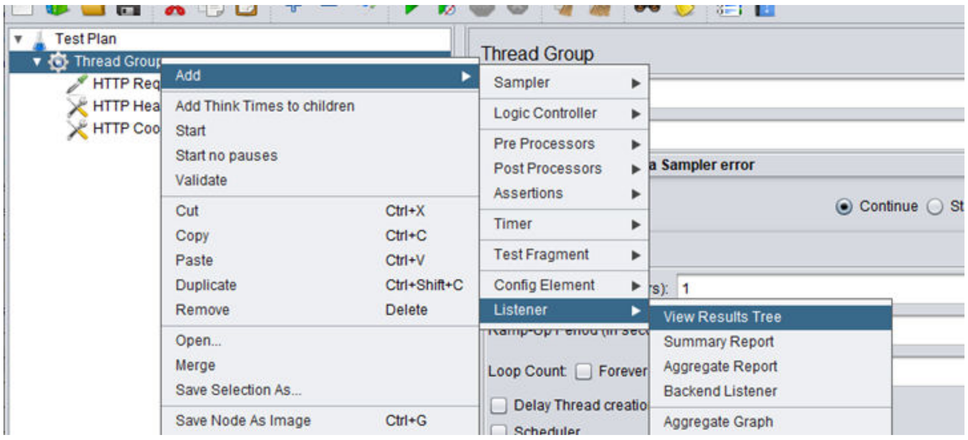
Headers Stored in the Header Manager

Name:	Value
accept	application/json;charset=utf-8

Step 6 Configure **View Results Tree**.

To view the response result, add a listener. Right-click **Thread Group**, choose **Add** > **Listener** > **View Results Tree**, and add the view result tree to the thread group.

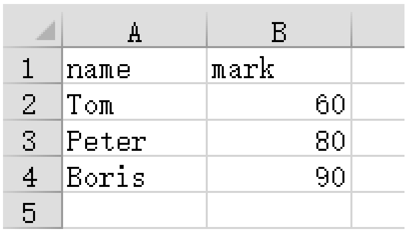
Figure 3-7 Adding a view results tree



Step 7 (Optional) Import file variables, if any.

CSV files are supported. In a CSV file, variable names are in the first row, and data is in other rows.

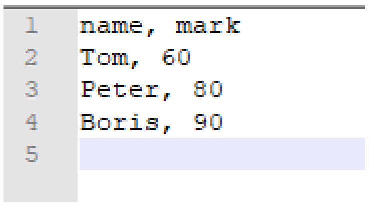
Figure 3-8 A CSV file



	A	B	
1	name	mark	
2	Tom	60	
3	Peter	80	
4	Boris	90	
5			

The following figure shows the variables in text format.

Figure 3-9 Variables in text format

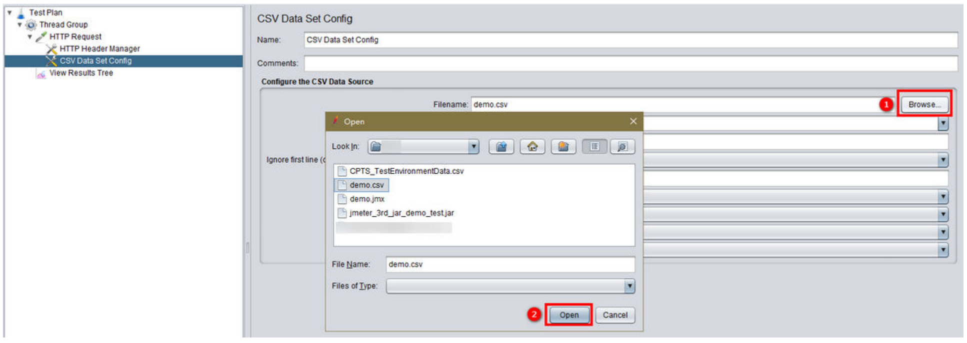


```
name, mark
Tom, 60
Peter, 80
Boris, 90

```

Right-click **Thread Group** and choose **Add** > **Config Element** > **CSV Data Set Config** from the shortcut menu. Click **Browse** to select the target CSV file, and click **Open** to add the file.

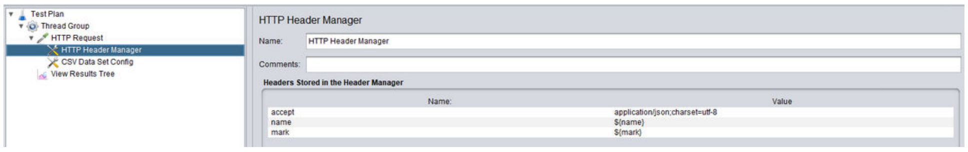
Figure 3-10 Adding a variable file



Name: CSV Data Set Config
Comments:
Filename: demo.csv
Ignore first line (
CPTS_TestEnvironmentData.csv
demo.csv
demo.jmx
jmeter_3rd_jar_demo_test.jar
File Name: demo.csv

To test the effect of referencing variables, you can introduce variables to the HTTP header. A file in ${variable name} format tells JMeter that the variables are read from external systems.

Figure 3-11 Introducing variables to the header



Name: HTTP Header Manager
Comments:

Name:	Value
accept	application/json;charset=utf-8
name	${name}
mark	${mark}

Step 8 Save the test plan file and debug the script.

Click on the top of the page, specify the save path and file name, save the test task as a JMX file, and store the project file (JMX) and variable file (CSV) in the same directory. The name of a JMX file contains only letters, digits, hyphens (-), underscores (_), and periods (.). A JMX file with an invalid name cannot be imported to CodeArts PerfTest.

Click the green triangle button on the top to execute the task. You can view the execution result in **View Results Tree**, including **Request Body**, **Request Header**, **Response Body**, and **Response Header**.

----End

3.3.2 Native JMeter Pressure Test

Step 1 When the JMeter test plan is debugged, save the JMX and CSV files.

Step 2 Prepare test resources and create a private resource group.

1. Log in to the CodeArts PerfTest console, choose **Resource Groups** in the navigation pane, and click **Create Resource Group**.
2. (Optional) If this is the first time you create a private resource group, grant CodeArts PerfTest permissions necessary for creating private resource groups.
3. If you do not have a CCE cluster, perform **Step 2.4** to create one before creating a resource group. If a CCE cluster is available, go to **Step 2.5** to create a resource group.
4. Create a cluster.

In the upper part of the page, click **create one**. The CCE console is displayed. For details about how to create a cluster, see **Buying a CCE Cluster**.

Click **Next: Select Add-on**. On the page displayed, select the add-ons to be installed during cluster creation.

Click **Next: Add-on Configuration**. Retain the default settings.

Click **Next: Confirm configuration**. After confirming that the cluster configuration is correct, select **I have read and understood the preceding instructions** and click **Submit**. It takes about 6 to 10 minutes to create a cluster.

After the cluster is created, return to the cluster management page and click **Create Node**. For details about how to create a node, see **Creating a Node**.

Click **Next: Confirm**. After confirming that the node configuration information is correct, select **I have read and understood the preceding instructions** and click **Submit**.

After the node is created, return to the CodeArts PerfTest console.

5. Create a resource group.

   Choose **Resource Groups** in the navigation pane, and click **Create Private Resource Group**.

   Set the parameters listed in **Table 3-1**.

Table 3-1 Creating a private resource group

Parameter	Description
Resource Group Name	Name of the private resource group to be created.
Debugging Cluster	Select a CCE cluster from the drop-down list.
Debug Node	Select the management node that performs a pressure test. The debugging node cannot be changed after the resource group is created.
Execution Node	Select a pressure target machine that provides performance data during a pressure test.

   Click **Create**.

Step 3 Create a JMeter test project.

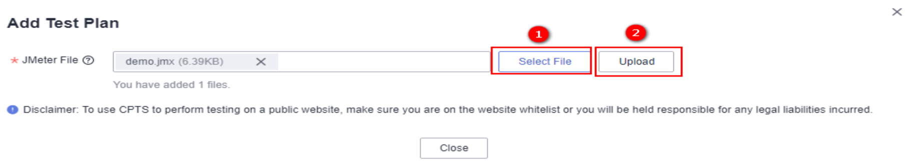
Go back to the CodeArts PerfTest console and choose **JMeter Projects** in the navigation pane. Click **Create JMeter Project**. Enter the project name and click **OK**.

Step 4 Import a JMX file.

1. On the **JMeter Projects** page, click ✎ to edit the test plan.

2. On the **test plan** tab, click **Add Task Plan**.
3. In the dialog box that is displayed, click **Select File**, select a JMX file, and click **Upload**. When the file is imported, the dialog box closes automatically and the **Test Plan List** tab is displayed. You can view the added test plan.

Figure 3-12 Importing a JMX file

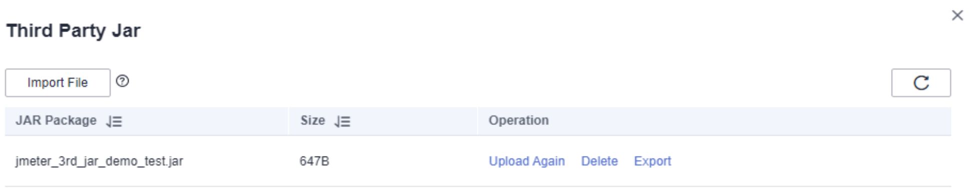


You have added 1 files.

Disclaimer: To use CPTS to perform testing on a public website, make sure you are on the website whitelist or you will be held responsible for any legal liabilities incurred.

4. (Optional) On the **test plan** tab page, click **Third Party Jar**. In the displayed dialog box, click **Import File**, select and import the JAR package on which the test plan depends. After the package is imported, the dialog box closes.

Figure 3-13 Importing a JAR package

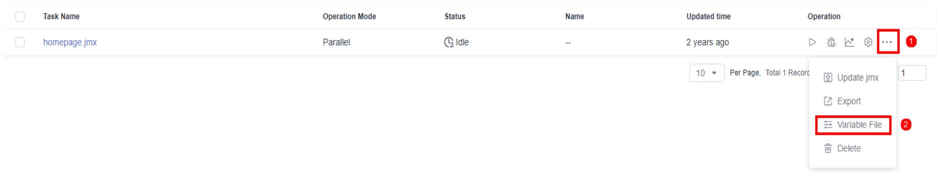


JAR Package	Size	Operation
jmeter_3rd_jar_demo_test.jar	647B	Upload Again Delete Export

Step 5 (Optional) Import the variable file.

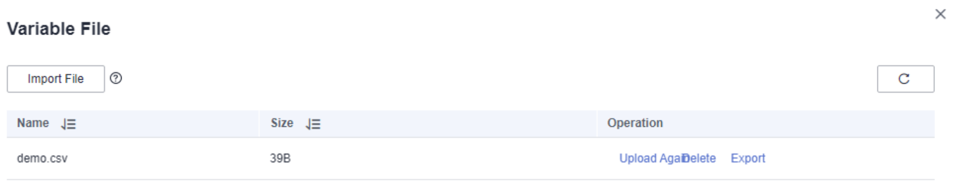
In the test plan list, click ⋯ and choose **Variable File**. In the displayed dialog box, click **Local Import**, select the variable file referenced by the test plan, and import the file. After the file is imported successfully, close the dialog box.

Figure 3-14 Importing variable file 1



	Task Name	Operation Mode	Status	Name	Updated time	Operation
	homepage.jmx	Parallel	Idle	--	2 years ago	

Figure 3-15 Importing variable file 2



Name	Size	Operation
demo.csv	39B	Upload AgaiDelete Export

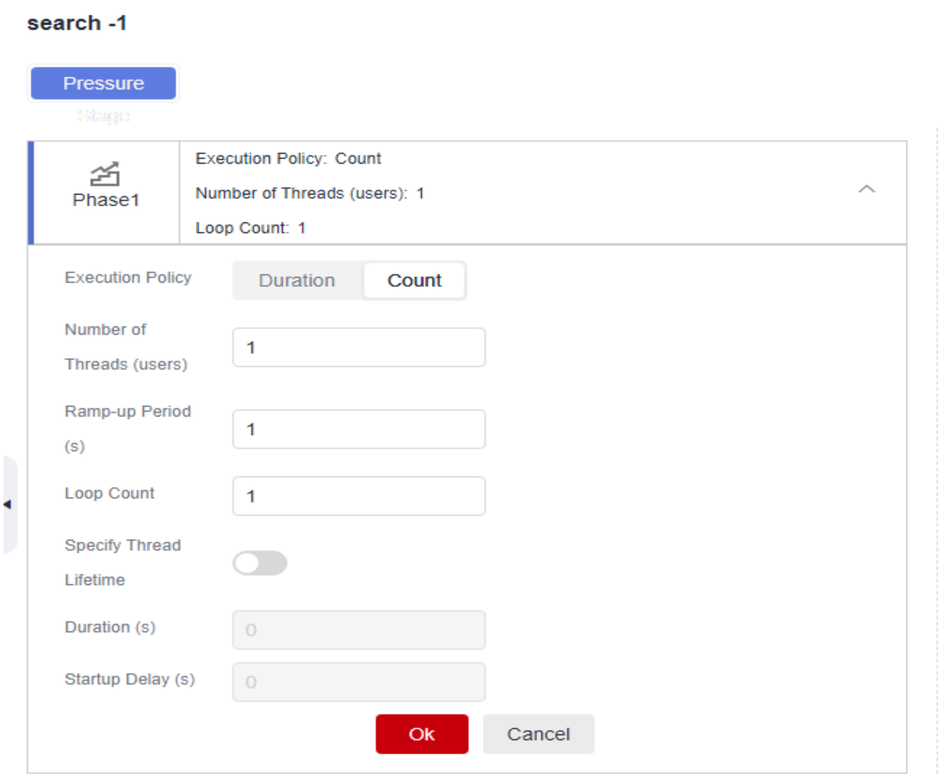
Step 6 Edit a thread group.

Click the task name in the JMeter test plan list. The thread group list is displayed. After the editing is complete, click **OK**.

- **Number of threads** corresponds to **Number of Threads** in the local JMeter program.

- **Warm-up Time** corresponds to **Ramp-up period** in the local JMeter program.
- **Number of Cycles** indicates the number of cycles to be tested.

Figure 3-16 Editing a thread group



search -1

Pressure Stage

Phase1

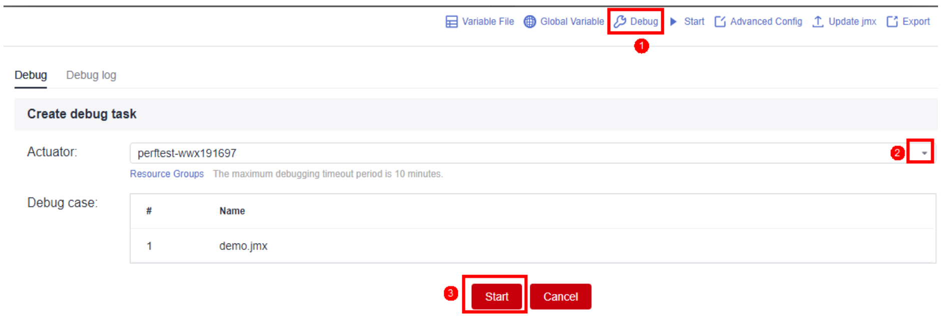
Execution Policy: Count
Number of Threads (users): 1
Loop Count: 1

Execution Policy	Duration [✓ Count]
Number of Threads (users)	1
Ramp-up Period (s)	1
Loop Count	1
Specify Thread Lifetime	
Duration (s)	0
Startup Delay (s)	0

Step 7 Debug the JMeter test task.

Click **Debug**, select the target test resource group as the executor, and click **Start** to start debugging. If an error was reported in the debugging result, edit the case based on the log information and debug it again.

Figure 3-17 Starting debugging



Create debug task

Actuator: perftest-wwx191697

Resource Groups The maximum debugging timeout period is 10 minutes.

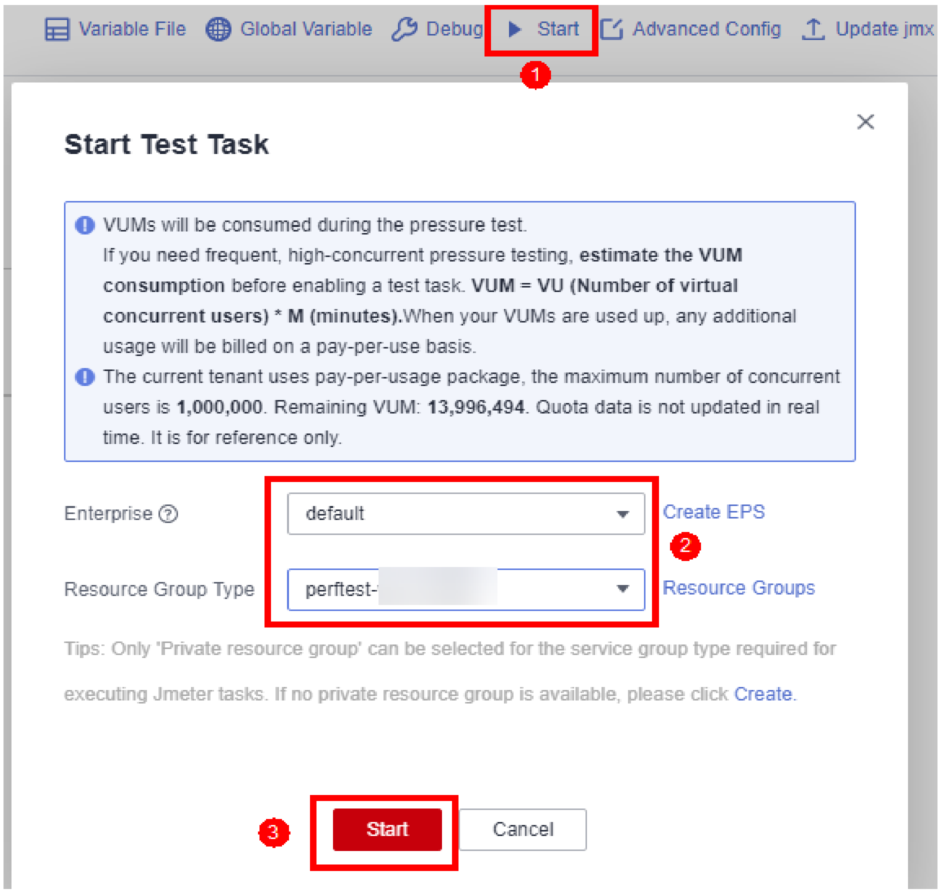
Debug case:

#	Name
1	demo.jmx

Step 8 Execute the JMeter test task.

Click **Start**. In the dialog box that is displayed, specify **Resource Group Type** and **Enterprise**, and click **Start** to start the test task.

Figure 3-18 Starting a test task



Start Test Task

VUMs will be consumed during the pressure test.
If you need frequent, high-concurrent pressure testing, **estimate the VUM consumption** before enabling a test task. **VUM = VU (Number of virtual concurrent users) * M (minutes).**When your VUMs are used up, any additional usage will be billed on a pay-per-use basis.

The current tenant uses pay-per-usage package, the maximum number of concurrent users is **1,000,000**. Remaining VUM: **13,996,494**. Quota data is not updated in real time. It is for reference only.

Enterprise: default

Resource Group Type: perftest-

Tips: Only 'Private resource group' can be selected for the service group type required for executing Jmeter tasks. If no private resource group is available, please click Create.

Step 9 View a test report.

After the test task is started, click **View Report** to view the monitoring data and chart report of each metric generated during the pressure test.

After the pressure test is complete, an offline test result report is generated automatically. You can download and view the report of the completed test task.

----End

4 Using Global Variables

4.1 Overview

Scenarios

Global variables are used to build data sets and enrich test data. If a global variable is used in a packet of a request, variable values in the packet will be replaced with specified values during a pressure test.

Global variables can be used in many scenarios. For example, you can store different usernames and passwords as global variables to simulate a scenario with multiple users.

Solution Architecture

CodeArts PerfTest provides global variables of the integer, enumeration, text, and file types.

4.2 Operation Process

1. Create a test project.
2. Create a test case.
3. Add a global variable.
4. Insert the added global variable into the test case.

4.3 Procedure

Step 1 Log in to the CodeArts PerfTest console, choose **PerfTest Projects** in the navigation pane, and click **Create Test Project**.

Step 2 In the displayed dialog box, enter a test project name (for example, **Web-test**), and click **OK**.

Step 3 Click the name (**Web-test**) of the created test project to access its details page. On the **Cases** tab page, you can view the default directory and sample cases that are automatically generated.

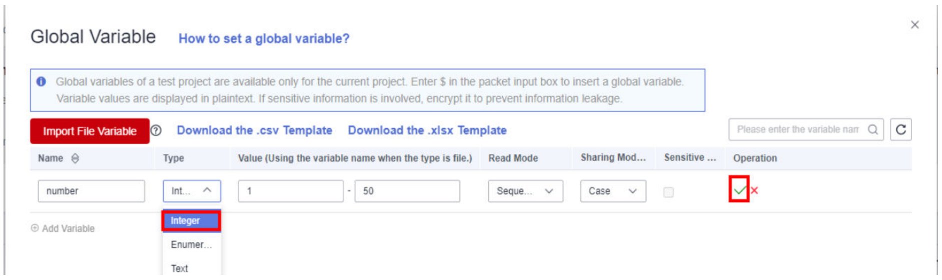
Step 4 On the **Cases** tab, click **Global Variables**.

Step 5 Add a global variable. You can directly add global variables of the integer, enumerated, or text type, or add file global variables with CSV or XLSX files.

- Adding an integer, enumerated, or text global variable

  In the **Global Variable** dialog box, click **Add Variable**, enter a variable name, select a variable type, enter a variable value, select a read mode and sharing mode, and click ✓ in the **Operation** column. In this practice, add an integer global variable named **number**.

  Figure 4-1 Adding a global variable of the integer type


Global Variable
How to set a global variable?
Import File Variable
Download the .csv Template
Download the .xlsx Template
Please enter the variable nam
Add Variable
Integer
Enumer...
Text

Global variables of a test project are available only for the current project. Enter $ in the packet input box to insert a global variable. Variable values are displayed in plaintext. If sensitive information is involved, encrypt it to prevent information leakage.

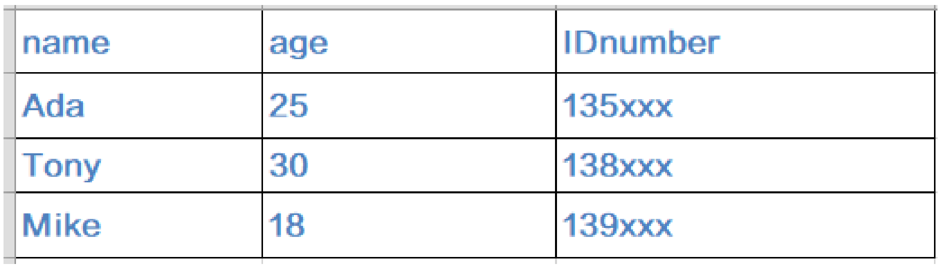
Name	Type	Value (Using the variable name when the type is file.)	Read Mode	Sharing Mod...	Sensitive ...	Operation
number	Int...	1 - 50	Seque...	Case		✓ ×

- Adding a global variable of the file type

  a. In the **Global Variable** dialog box, click **Download the .csv Template** or **Download the .xlsx Template** as required.

  b. Enter the variable names and values by referring to the template. As shown in **Figure 4-2**, enter variable names in the first row and enter variable values from the second row. The rules for filling in the .csv and .xlsx files are the same.

  Figure 4-2 Example of filling file variables



name	age	IDnumber
Ada	25	135xxx
Tony	30	138xxx
Mike	18	139xxx

  The file variables are described as follows:

  i. Use letters, digits, and underscores (_) for variable names. Variable values are not limited.

  ii. Only .csv (UTF-8 without BOM) and .xlsx files can be imported.

  iii. A file name (including the extension) can contain a maximum of 50 bytes. Letters, digits, and underscores (_) are allowed.

  iv. The maximum size of an .xlsx file is 20 MB and that of a .csv file is 80 MB.

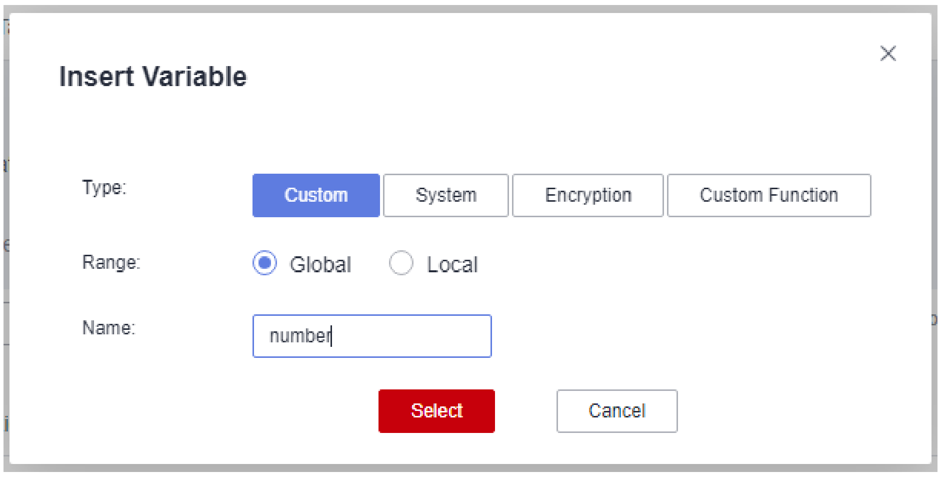
c. Click **Import File Variable** to upload the configured variable file.

Step 6 After global variables are added, close the **Global Variable** dialog box.

Step 7 Insert the added global variable into the test case. For example, insert the created global variable **number** to the sample case request information in **Step 3**.

1. Enter **$** in the request information input box.
2. In the **Insert Variable** dialog box, set **Type** to **Custom** and **Range** to **Global**, and enter **number** as the variable name.

Figure 4-3 Inserting a variable



Insert Variable

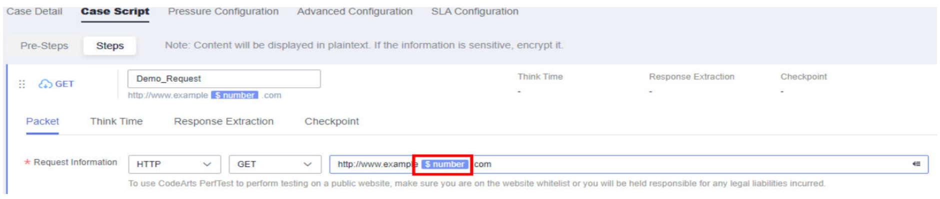
Type: [✓ Custom] System Encryption Custom Function

Range: [✓ Global] Local

Name: number

3. Click **Select** to insert a variable.

Figure 4-4 Insert the variable named number



Note: Content will be displayed in plaintext. If the information is sensitive, encrypt it.

GET Demo_Request
http://www.example $ number .com

Think Time	Response Extraction	Checkpoint
-	-	-

* Request Information: HTTP | GET | http://www.example $ number .com

To use CodeArts PerfTest to perform testing on a public website, make sure you are on the website whitelist or you will be held responsible for any legal liabilities incurred.

----End