ALIBABA CLOUD

Alibaba Cloud

DataWorks

APP Studio

Document Version: 20220415

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Overview

App Studio is a tool designed to facilitate your data product development. It comes with a rich set of frontend components that you can drag and drop to easily and quickly build frontend apps.

With App Studio, you do not need to download and install a local integrated development environment (IDE) or configure and maintain environment variables. Instead, you can use a browser to write, run, and debug apps and enjoy the same coding experience as that in a local IDE. App Studio also allows you to publish apps online.

Benefits

App Studio has the following core advantages:

- Development anytime and anywhere

  You do not need to download and install a local IDE or configure and maintain environment variables. Instead, you can use a browser to develop data in your office, at home, or anywhere you can connect to the network.

- Editor with complete features

  App Studio provides a browser-based editor that allows you to easily write, run, and debug projects. When you enter code, App Studio intelligently displays code hints, completes the code, highlights syntax errors, and provides error fix suggestions. You can also search for references and definitions of methods and use the code that is automatically generated.

- Online debugging

  App Studio comes with all breakpoint types and operations of a local IDE. It supports thread switching and filtering, variable viewing and watching, remote debugging, and hot code replacement.

- Multi-feature terminal

  You can directly access the runtime environment, which is currently built based on CentOS as the base image. The multi-feature terminal supports all bash commands, including vim and other interactive commands.

- Collaborative coding

  You and your team members can use App Studio to share the development environment for collaborative coding. Currently, App Studio allows a maximum of eight users to edit the same file of a project online at the same time, improving the work efficiency. In the future, App Studio will support features such as chatting, bullet screen messages, code annotations, and videos to make teamwork efficient and pleasant.

- Plug-in system

  App Studio supports business plug-ins, tool plug-ins, and language plug-ins.

  - App Studio allows you to customize menus and add business entries based on your business requirements.
  - You can customize project management processes, project types, and templates dedicated to your business.
  - You can develop common tools, such as enhanced Git features, code rule scanning, keyboard shortcuts, enhanced editing features, and code snippets, and integrate them into App Studio.

  - You can use language plug-ins to enrich the languages supported by App Studio, enabling App Studio to serve users with more languages while addressing your own business needs.

- Visual building

  App Studio provides rich components and highly integrates with DataService Studio and DataStudio. You can call some DataWorks APIs in App Studio only. You can also drag and drop components and configure them in a visual way to quickly build frontend web apps without the need to write any code.

- Various templates and flexible project management

  App Studio provides various template-based projects, allowing you to develop your project accordingly with less labor and higher efficiency. You can also save your project as a template for future development and use, or share it with other users.

2.Version history

This topic lists the version history of App Studio.

App Studio V1.0

Released on: April 3, 2019

Content: App Studio provides an IDE that is used to publish apps based on Function Studio. It has the following core features:

- Language Server Protocol (LSP) based language service

  App Studio supports features such as syntax highlighting, code hinting, code completion, smart diagnosis, definition search, and reference search, providing the same experience as editing in a local IDE.

- Debugging

  App Studio comes with all breakpoint types and operations of a local IDE. It supports thread switching and filtering, variable viewing and watching, remote debugging, hot code replacement, and multi-feature terminal.

- API-based frontend and backend development

  In App Studio, you can configure backend APIs and associate them with frontend visual components.

- Frontend visual building

  You can drag and drop components to flexibly build frontend apps. This feature is applicable to users who do not have experience in developing frontend apps. App Studio also supports frontend template management and switching between the visual mode and code mode to meet the higher development requirements of developers.

- Code version control
- Online deployment and real-time app preview
- Collaborative coding

  Currently, App Studio allows a maximum of eight users to edit the same file of a project online at the same time.

- Custom project templates and strong project management capabilities
- Plug-in development and integration capabilities

  You can develop plug-ins and customize business-specific IDEs. (This feature will be published in App Studio V1.1 together with Plug-in Market.)

- Support of multiple languages, such as Java, JavaScript, CSS, HTML, and Python
- Automatic generation and running of unit testing (UT) code
- Project sharing through link (This feature will be published in App Studio V1.1 together with Plug-in Market.)
- Online publishing of developed apps (This feature will be published in App Studio V1.2.)

3.Get started

与产品方确认，该应用已经下线，相关文档无需翻译

4.Features

4.1. Navigation page

4.1.1. Projects

On the Projects page, you can create and manage projects.

The Projects page displays all projects you have created. On this page, you can use three methods to create a project. For more information, see Project management.

You can click a project card to go to the project development page. You can also click **Create Template** or **Manage** on a project card to perform corresponding operations.

Create Template

1. Click **Create Template** on a project card.
2. In the **Create Template** dialog box that appears, set parameters.

Parameter	Description
Template Name	The name of the template.
Description	The brief description of the template.
Type	The type of the template.

3. Click **OK**.

Manage

You can publish a project as an app. To facilitate version management, you can publish the project into different versions before publishing the app.

1. Click **Manage** on a project card to go to the project management page.
2. Click **Publish New Version** in the upper-right corner of the page. In the dialog box that appears, select the app for which a new version needs to be published and set parameters.

> **Note** Before publishing a new version, you must associate the project with a Git repo.

3. Click **Update** to publish the new version.

4.1.2. Apps

The Apps page consists of three tabs: Created by Me, Shared by Me, and Shared to Me.

> Note
>
> - The **Shared by Me** tab is available only to users who have purchased the Flagship Edition of DataWorks.
> - The **Shared to Me** tab is available only to users who have purchased the Enterprise Edition or Flagship Edition of DataWorks.

Created by Me

The **Created by Me** tab displays all apps you have created. You can click Publish on an app card to publish the current app. On the app card, you can also click **Manage Deployment** to go to the app **O&M** page.

> Note The **Share** button is available only to users who have purchased the Flagship Edition of DataWorks.

Manage Deployment

You can click **Manage Deployment** on an app card to go to the app O&M page.

The app O&M page displays the O&M status of each app. You can select an app from the drop-down list on the left and view details of the app.

- **Overview**

  The Overview tab displays the following information about an app: **App Info**, **App Status**, **ECS Group Info**, **ECS Instance Info**, **ECS Group List**, and **ECS Instance List**.

- **Monitoring**

  The Monitoring tab displays detailed O&M metrics of an app, including three app metrics, eight system metrics, and seven JVM metrics.

- **Image**

  The Image tab displays the image ID of each ECS group and the build time of each image.

- **Change**

  The Change tab displays change operations you have initiated, such as deploying an app, expanding the capacity for an app, and removing an ECS instance for an app. You can click a **change order ID** to view details of the change.

  If you click the change order ID of an app that is being deployed, you can view detailed deployment information and logs.

- **Resource**

  The Resource tab displays the VPCs you have purchased. After you purchased a VPC, you need to click Add VPC to add it here.

  You can click a VPC ID to go to the VPC details page.

- **Actions**

  The Actions drop-down list provides four operations: **Restart app**, **Restart ECS instance**, **Unpublish ECS instance**, and **Expand capacity**.

- **Restart app**

  To restart an app, select Restart app from the Actions drop-down list. In the **Restart app** dialog box that appears, describe your operation and click **Run**.

- **Restart ECS instance**

  To restart a specified ECS instance in a specified ECS group of the current app, select Restart ECS instance from the Actions drop-down list. In the **Restart ECS instance** dialog box that appears, specify **ECS Group** and **ECS Instance** , describe your operation, and then click **Run**.

- **Unpublish ECS instance**

  To release a specified ECS instance from a specified ECS group of the current app to the resource pool, select Unpublish ECS instance from the Actions drop-down list. In the **Unpublish ECS instance** dialog box that appears, specify **ECS Group** and **ECS Instance**, describe your operation, and then click **Run**.

- **Expand capacity**

  To add an ECS instance from the resource pool to a specified ECS group of the current app, select Expand capacity from the Actions drop-down list. In the **Expand capacity** dialog box that appears, specify **Target ECS group** and **Available ECS instances**, describe your operation, and then click **Run**.

Publish

To publish an app, click **Publish** on the app card. For more information, see App deployment.

Share

To share an app with other users, click **Share** on the app card. Note that only users who have purchased the Enterprise Edition or Flagship Edition of DataWorks can share their apps with other users. After sharing, you can view the shared apps in the **Shared by Me** list. Users with whom you share the apps can view the shared apps in the Shared to Me list.

Shared by Me

To view the apps you have shared, choose **Apps > Shared by Me**.

To push updates of your app code to users with whom you share the app, click **Notification** on the app card. In the App Update Notification dialog box that appears, set parameters and click **Confirm**.

Shared to Me

To view the apps shared to you by other users, choose **Apps > Shared to Me**. On this tab, you can also deploy and publish the apps. For more information, see the description about the Created by Me tab.

4.1.3. Templates

The Templates page displays all templates you have created from projects.

You can click a template card to go to the template details page. Then, you can click **Code Editor** to view the project code of the current template.

You can also click **Create Project** on a template card to go to the page where you can create a project by using the current template.

4.2. Project management

This topic describes how to create and manage projects.

You can create a template-based or code-based project or import a project from Git.

Create a template-based project

1. Log on to App Studio. On the **Projects** page, click **Create Project from Template**.
2. On the **Create Project** page, set **Project Name** and **Project Description**, and select a template.

   > **Note**
   > - You can select a custom template or a template provided by the system.
   > - All projects created by using templates support WYSIWYG development.

3. After the configuration is completed, click **Submit**.

Create a code-based project

You can create a project by running code. App Studio provides code templates for four types of runtime environments. Select a code template as required.

1. Log on to App Studio. On the **Projects** page, click **Create Project from Code**.
2. On the **Create Project** page, set **Project Name** and **Project Description**, and select a template.
3. After the configuration is completed, click **Submit**.

Import a project from Git

If you have Git code, you can import the Git code to create a project. You can only import Git code from code.aliyun.com.

1. Log on to App Studio. On the **Projects** page, click **Import Project from Git**.
2. On the **Create Project** page, set **Git Repo URL**, **Project Name**, and **Project Description**, and select a runtime environment.
3. After the configuration is completed, click **Submit**.

View the list of projects

You can view the created projects on the **Projects** page.

You can click a project name to go to the project editing page. You can also click **Create Template** of a project to create a template based on the project.

App Studio supports managing the deployment versions of projects. You can click **Manage** of a project to go to the deployment version control page.

On the Project Details page, click **Publish New Version** to publish a version. Then, go to the **Apps** page to deploy the corresponding project version.

> **Note** Before publishing a project version, you must associate the project with Git.

4.3. Version control

App Studio integrates general Git services. This topic describes how to use VCS-Git in App Studio.

Create a project and associate it with Git

1. Create a project.
2. Enter basic user information.

   Before associating the project with Git, you must enter basic user information. Click **Settings** in the top navigation bar and select SSH Key. Generate an SSH key and add it to the public key list of the account that owns the Git repo as prompted.

   > **Note** The new project is not associated with Git by default. To use Git, associate the current project with your Git repo.

3. Create a Git repo.
4. Obtain the HTTPS URL of the current repo.
   i. Click **HTTPS**. The HTTPS URL of the current repo appears.
   ii. Click the Copy icon next to the HTTPS URL to copy it to the clipboard.
5. Associate the project with the Git repo.
   i. In the top navigation bar, choose **Version > Connect to Remote Repo**.
   ii. In the **Connect to Remote Repo** dialog box, enter the HTTPS URL of the Git repo and click **Submit**.
   iii. After the association is completed, the version control icon appears in the left sidebar of App Studio.
   iv. In the top navigation bar, choose **Version** > **Push** to push the local code to the remote repo.

Entry to Git-related operations

You can click the version control icon in the left sidebar or click Version in the top navigation bar and select options to perform Git-related operations.

Git control panel

The file editing status is dynamically updated on the Git control panel.

You can perform basic Git-related operations, such as `git add, git rm, git commit, and git revert`, on the Git control panel.

Basic Git operations

Edited files are listed on the Git panel, including the file names and paths. The basic operations that are supported are displayed on the right.

As shown in the preceding figure, the supported operations and file icons are marked by the red boxes.

- **Source Code: Git**

  You can perform the commit, refresh, pull, and push operations.

- Commit: Click and select Commit & Push.
- Refresh: Click to refresh the current control panel. This operation is equivalent to running the **git status** command and refreshing the page.
- Pull and push: Click ... and select **Pull** or **Push** as required.

- **Save Edits**
  - : discards all edits. This operation is equivalent to running the **git reset** command.
  - 2: indicates the number of files.
  - SyncPaiApiClient.java src/main/java/co... M: indicates that the file is edited.
- **Modify**
  - : discards all edits.
  - : adds all files to the cache. This operation is equivalent to running the **git add** command.
  - 2: indicates the number of files.
  - The following operations can be performed for the listed files:
    - : discards all edits.
    - : stages all edits.
    - M: indicates that the file is edited.

> **Note**
> - The logic of the Git client is the same. You must perform the push operation so that the local code is pushed to the remote repo.
> - Similarly, you must perform the pull operation so that the remote code is pulled to the local repo.

Manage branches

Open the branch management window. Click the branch name in the status bar at the bottom of the page. The branch management window appears.

Create a local branch

After a branch is created, the page of the new branch appears.

Create, switch, and merge branches

After a local branch is created, it can be directly pushed to the remote repo. The name of the local branch is the same as that of the remote one.

Show Git history

You can right-click a file and choose **Git > Show History** to view its Git history. You can compare the differences between the specific commit version and the current version.

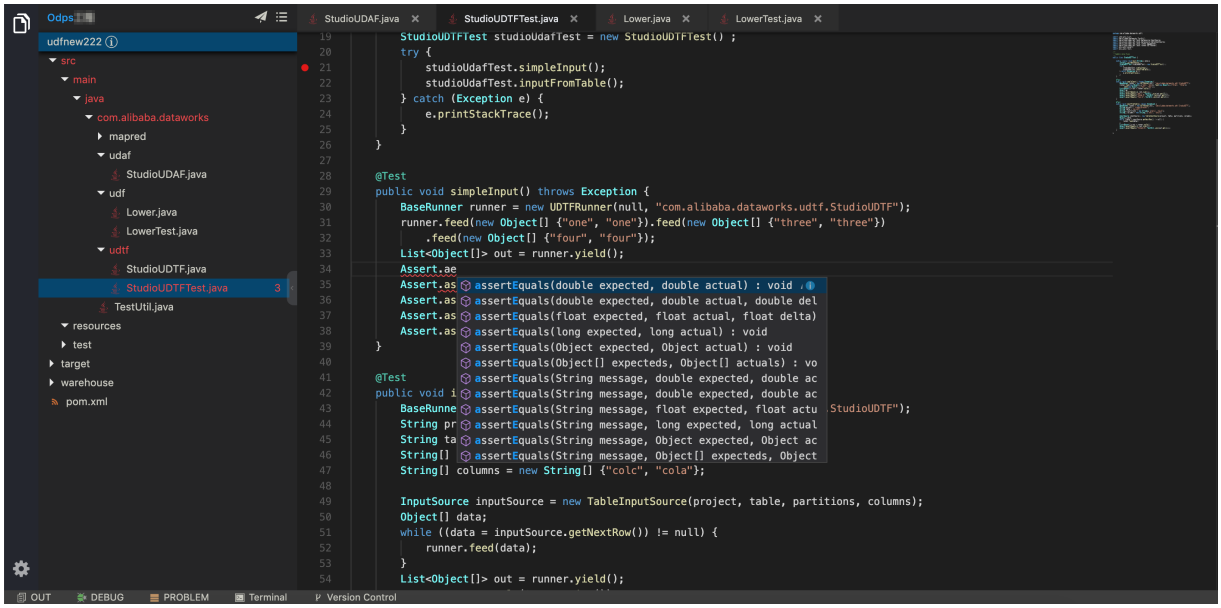
View Git logs

In the top navigation bar, choose **Version > View Log**. On the Log tab, you can view the message, time, and committer of the submitted logs. You can also filter the submitted logs by message, branch, committer, and time.

4.4. Code editing

4.4.1. Overview of code editing

Code editing supports common IDE features, such as automatic completion, code hinting, syntax diagnosis, and global content search.



The following tables list the basic and advanced features that App Studio supports in different languages.

Basic feature	Java	Python	JavaScript and TypeScript
Completion	Supported	Supported	Supported
Hover	Supported	Supported	Supported
Diagnostics	Supported	Supported	Supported
SignatureHelp	Supported	Supported	Supported
Definition	Supported	Supported	Supported
References	Supported	Supported	Supported

Basic feature	Java	Python	JavaScript and TypeScript
Implementation	Supported (coming soon)	Not supported	Not supported
DocumentHighlight	Supported	Supported	Supported
DocumentSymbol	Supported	Supported	Supported
WorkspaceSymbol	Supported	Supported	Supported
CodeAction	Supported (Alibaba Java Guidelines coming soon)	Supported	Supported
CodeLens	References implementation	Not supported	Not supported
Formatting	Supported	Supported	Not supported
RangeFormatting	Supported	Not supported	Not supported
FindInPath	Supported	Supported	Supported

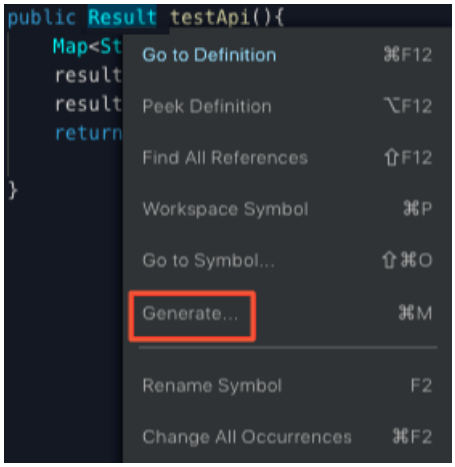
Advanced feature	Java	Python	JavaScript and TypeScript
Rename	Supported	Supported	Supported
WorkspaceEdit	Supported	Not supported	Not supported
UnitTest (quick start)	Supported	Not supported	Not supported
MainClass	Supported	Not supported	Not supported
MainClassQuickStart	Not supported	Not supported	Not supported
ListModules	Supported	Not supported	Not supported
Generate	Constructor Override Getter and Setter Implement	Not supported	Not supported

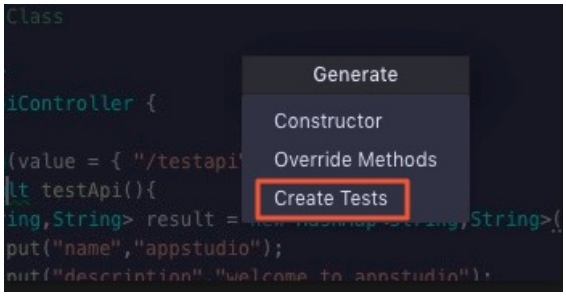
4.4.2. Run UT

App Studio currently supports unit testing (UT), including automatically generating UT code, detecting the entry for UT, running UT code, and displaying the UT result.

Automatically generate UT code

Open the target file, right-click the code editing area, and then choose **Generate > Create Tests**. The UT class file and UT code are automatically generated in the test directory.

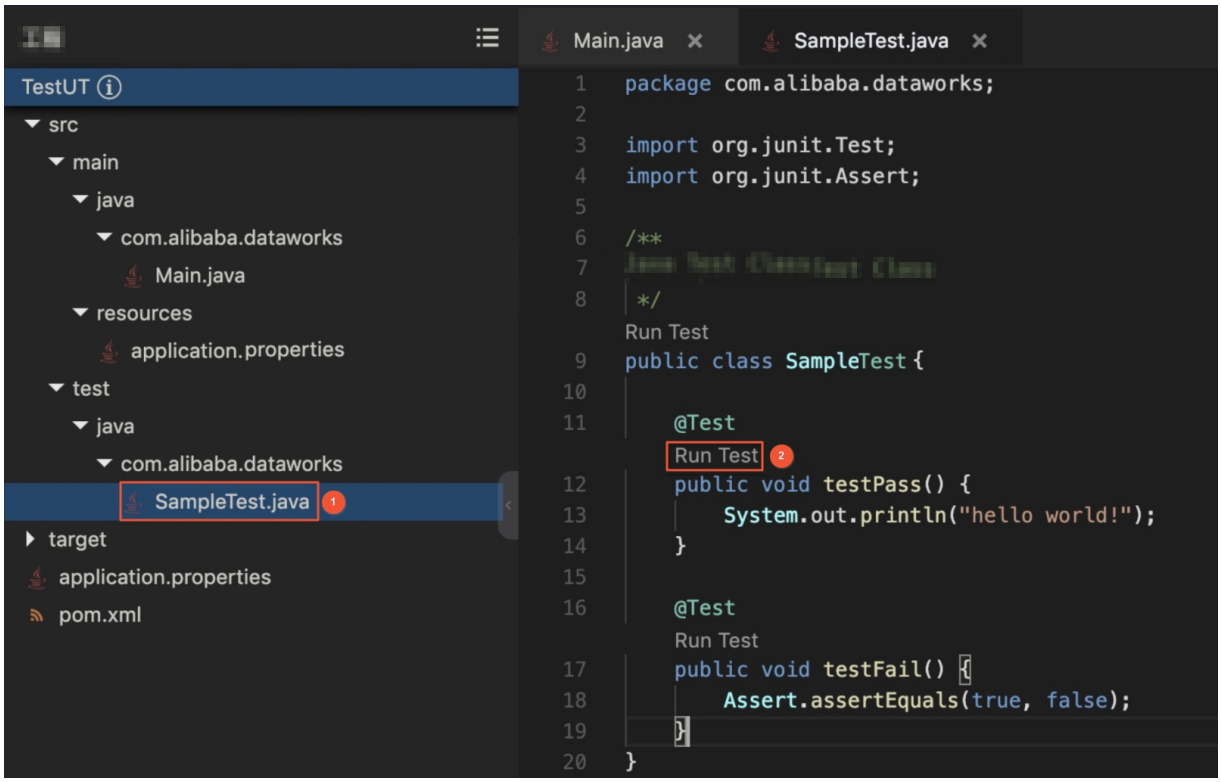




Detect the entry for UT

> **Note**
>
> - UT class files must be stored in the *src/test/java* directory. A Java UT class file that is not stored in this directory cannot be identified as the Java UT class.
> - For a method annotated with @Test annotation, Run Test appears, indicating the entry for UT.

After the Java UT class file is created, add the **@Test** annotation of `org.junit.Test` to the corresponding sample UT method.



Run UT code

Click the Run icon in the upper-right corner. The sample UT starts.

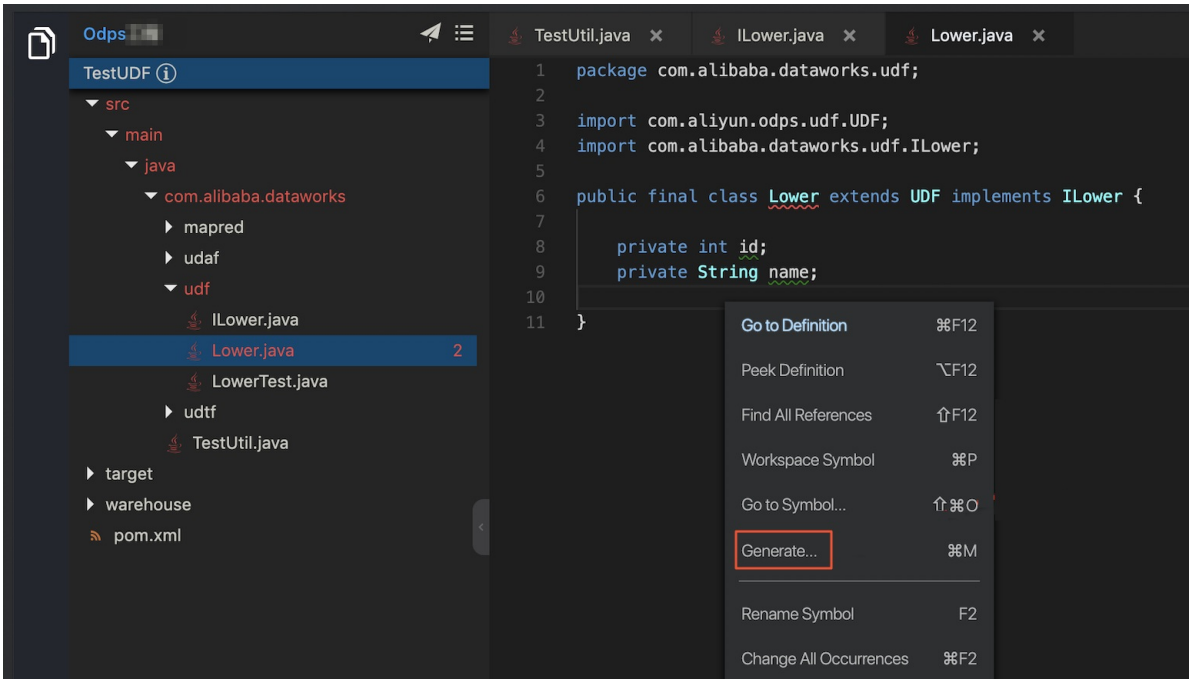
4.4.3. Generate code snippets

Currently, App Studio supports the Java class constructor, getter and setter functions, override methods of the parent class that a child class inherits, and API methods to be implemented.

Procedure

Perform either of the following operations to generate the Java code:

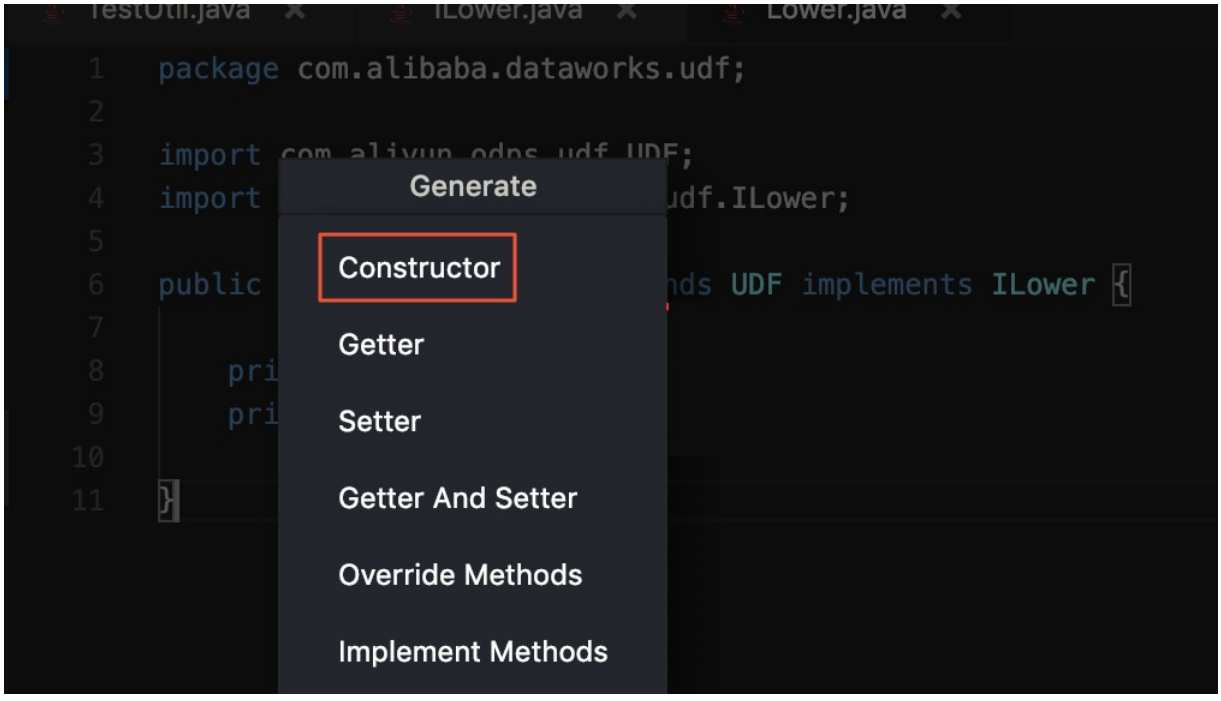
- Right-click the code area and select **Generate**.



- Press Command+M on the keyboard. The Java code is automatically generated.

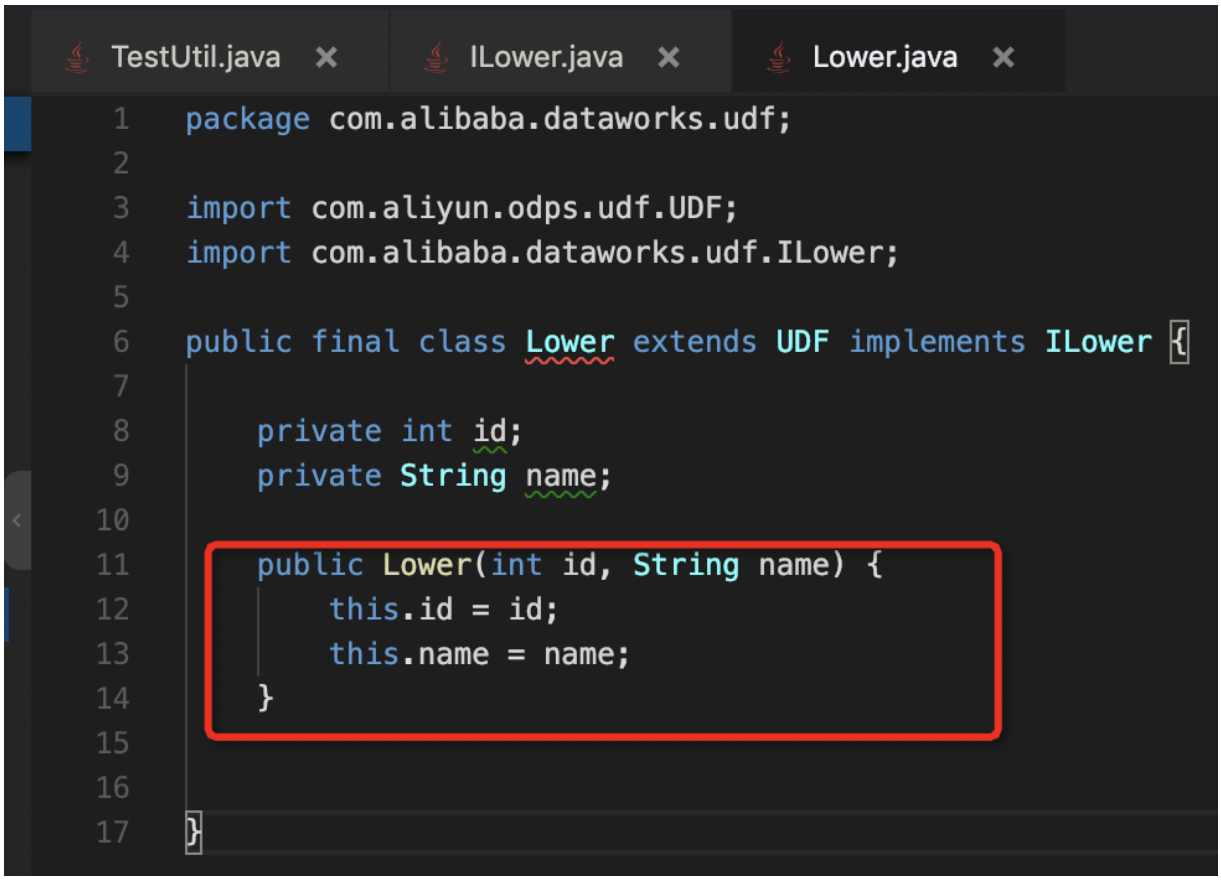
Constructor

On the Generate panel, select **Constructor**.



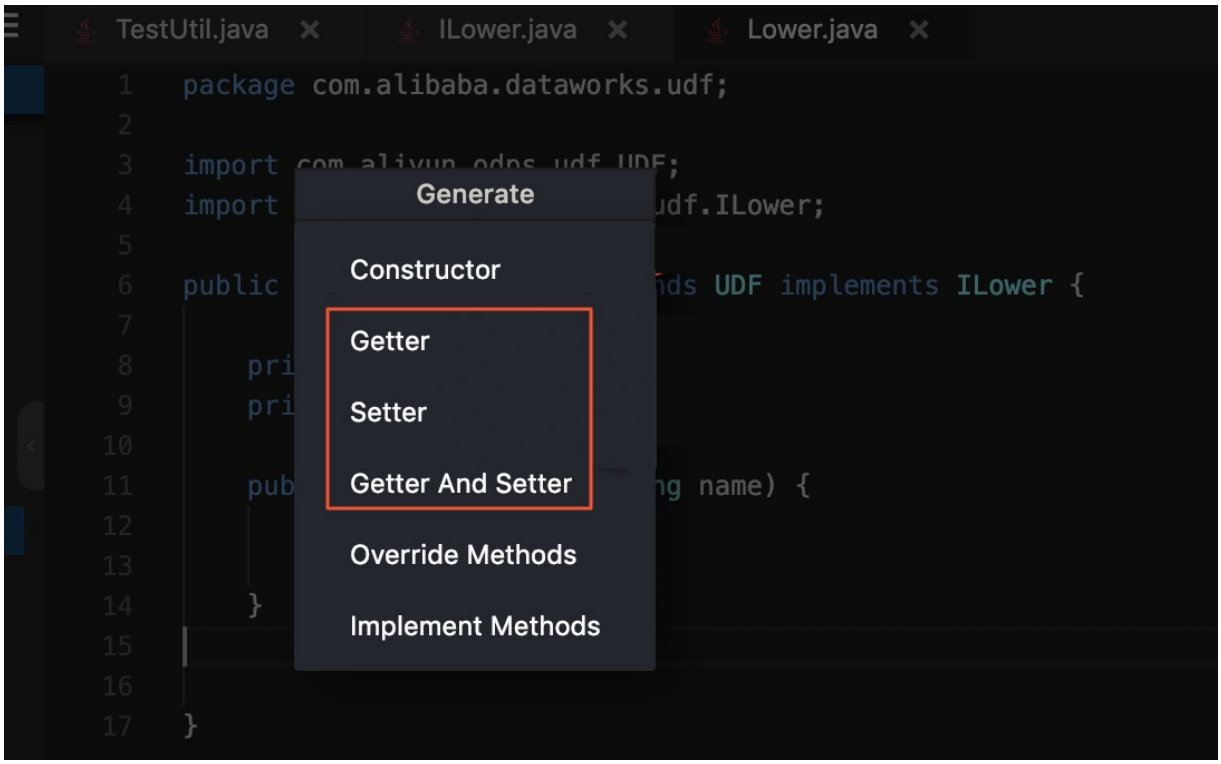
Select the fields to be included in the constructor and click OK.

The constructor that contains the initialization statement of the fields is generated.



Getter and setter functions

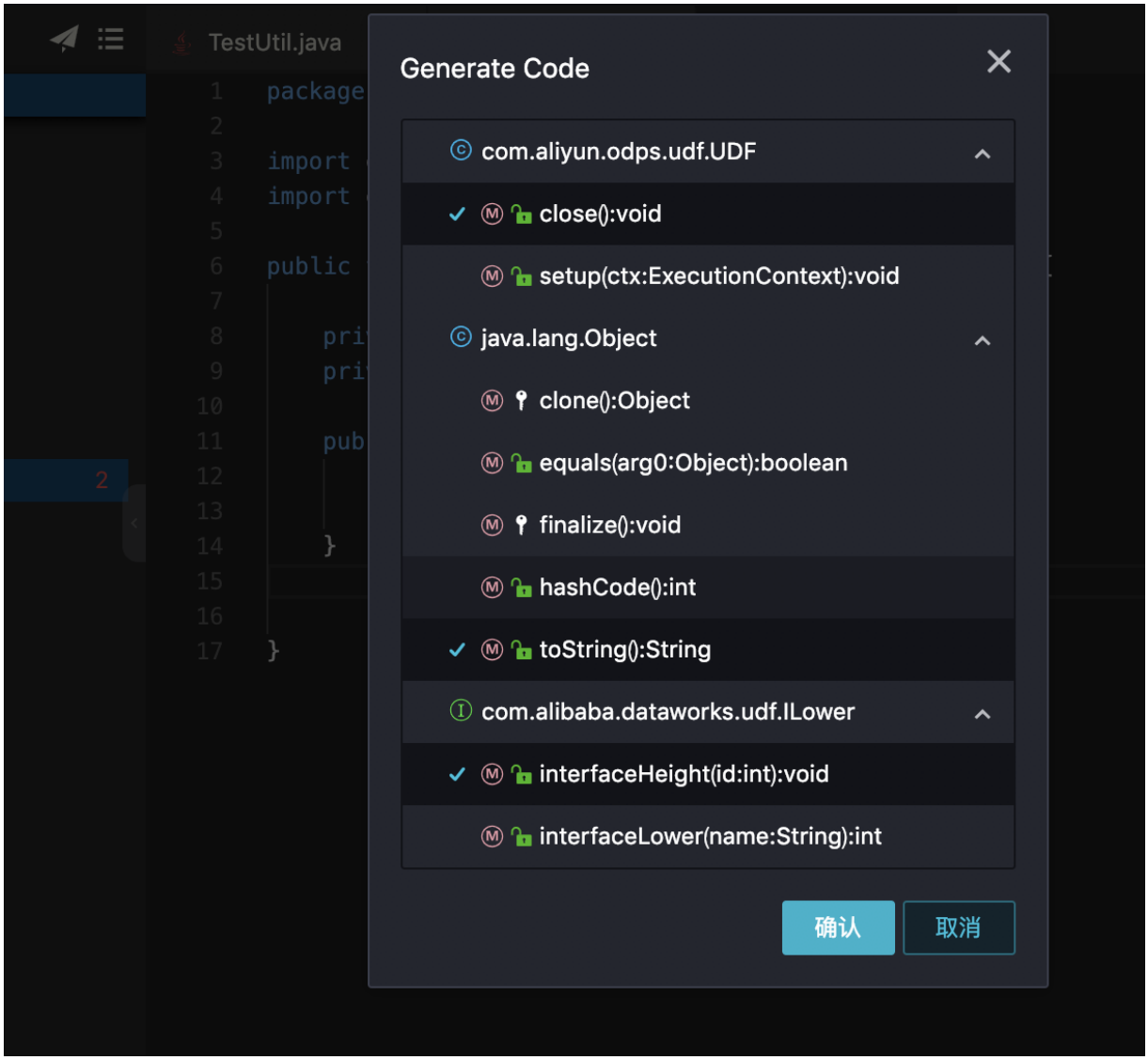
Generate the getter and setter functions in a way similar to the constructor.



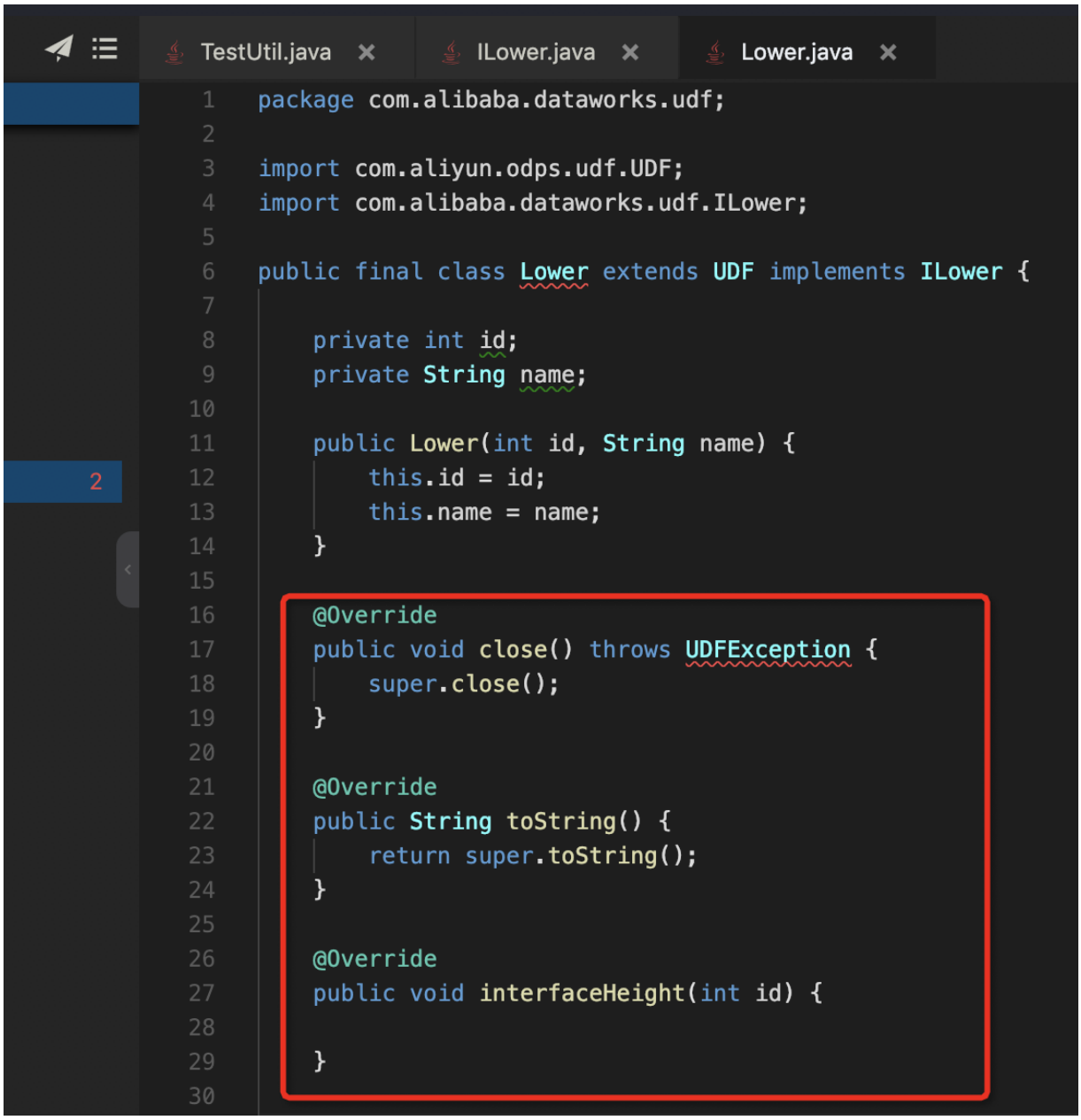
> **Note** If a Java class does not have any field or the Java class is overwritten by the @data annotation of lombok, the getter or setter function is not required for the Java class. In this case, the Getter, Setter, and Getter And Setter options do not appear on the Generate panel.

Override methods

Select Override Methods on the Generate panel. All methods that can be overridden are listed on the Generate Code panel.

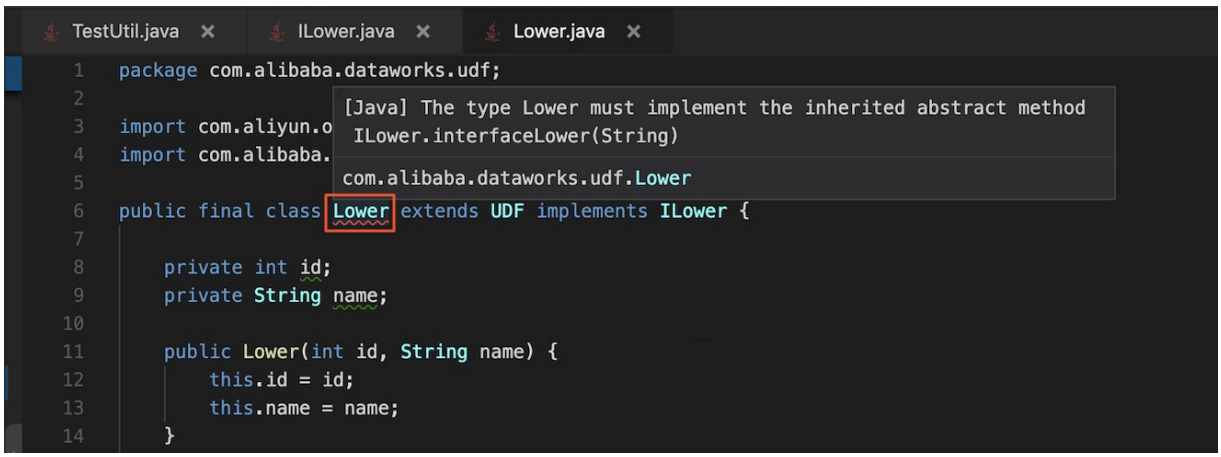


Select a method. The corresponding method is generated.

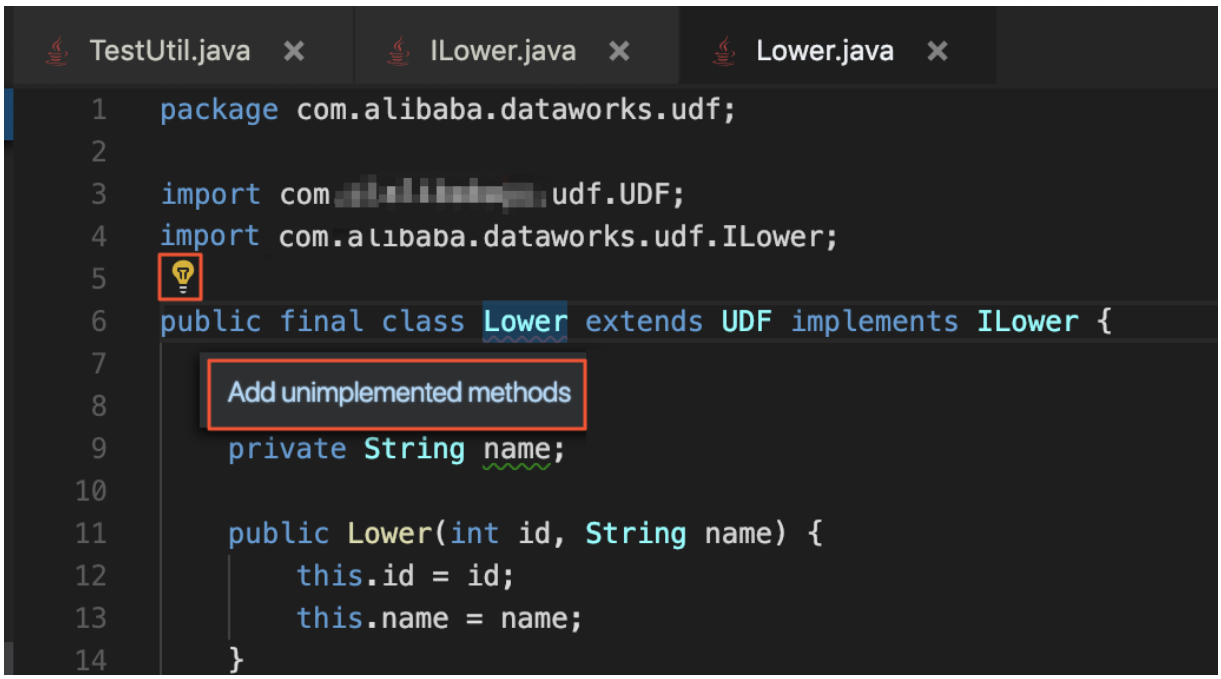


Implement methods

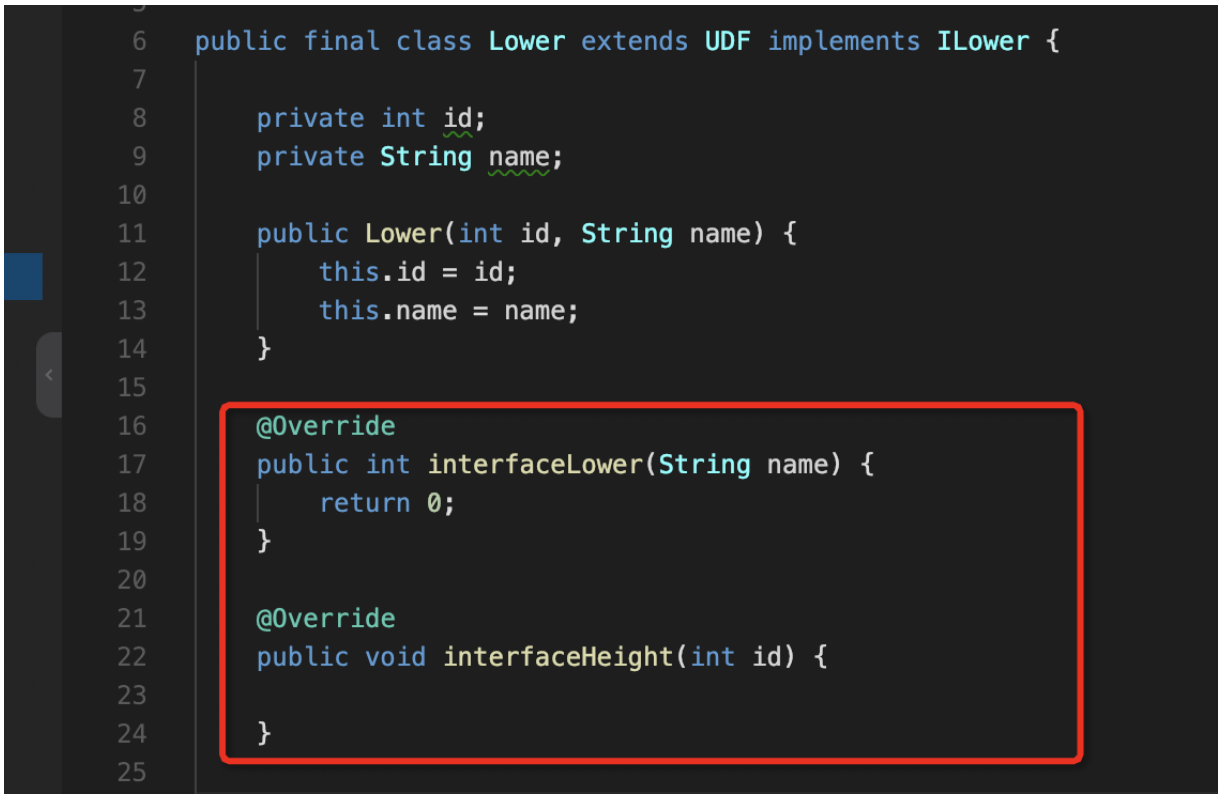
The way of implementing a method is similar to that of overriding a method. In Java, a class that implements an API must define the methods of the API. If a method is not implemented, the class syntax is incorrect, which is underlined with a red wavy line.



In addition to selecting Implement Methods on the Generate panel, you can also use the code hinting feature to implement a method.



The following figure shows the generated code.



```
6   public final class Lower extends UDF implements ILower {
7
8       private int id;
9       private String name;
10
11      public Lower(int id, String name) {
12          this.id = id;
13          this.name = name;
14      }
15
16      @Override
17      public int interfaceLower(String name) {
18          return 0;
19      }
20
21      @Override
22      public void interfaceHeight(int id) {
23
24      }
25
```

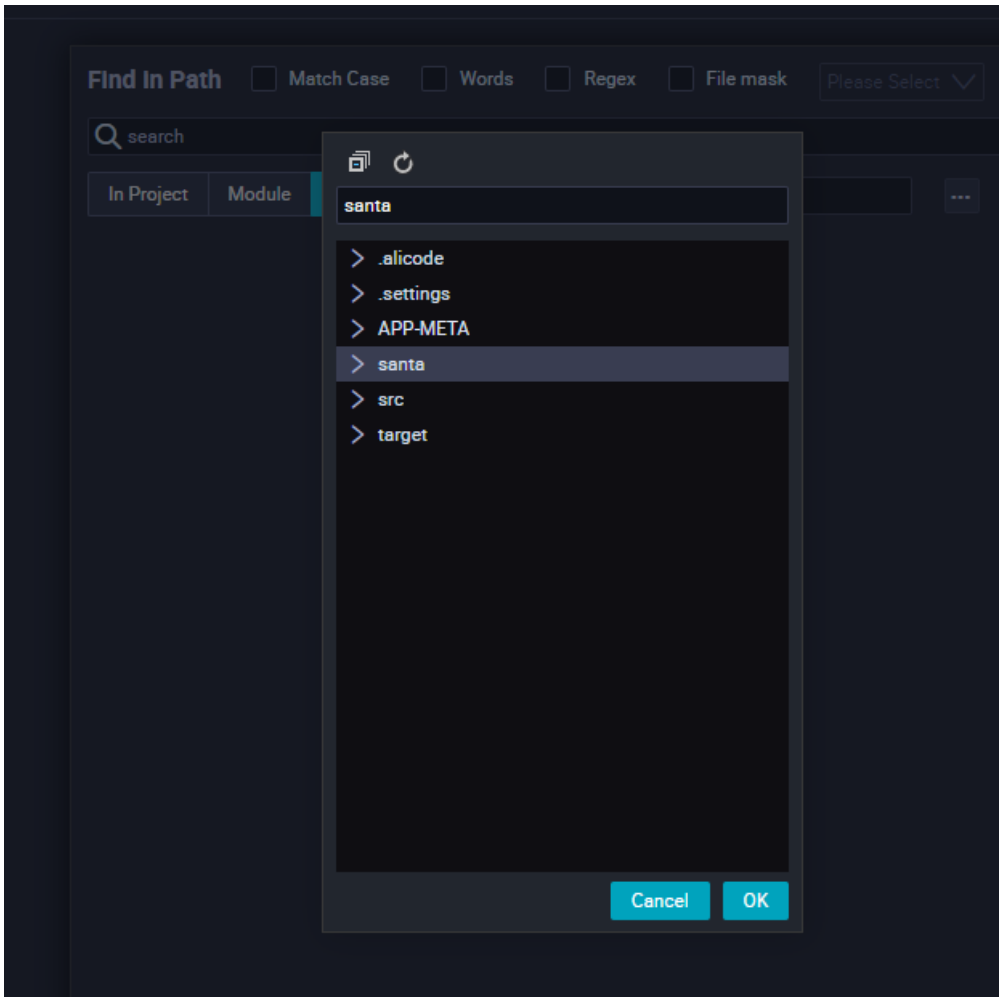
4.4.4. Find in Path

App Studio provides the Find in Path feature to support global content search.

In the top navigation bar, choose **Edit > Find in Path**.

You can select Match Case, Words, Regex, or File Mask to set the filter criteria.

You can also click Module or Directory to search files by module or directory.



After selecting a file, you can locate the searched content in the file and open the file in the editor.

4.5. Debugging

4.5.1. Run/Debug configurations

You can configure the entry function, start debugging, and set breakpoints to debug an app.

Configure the entry function

Parameter	Description
Main class	The class of the main function to be started. Select a value from the drop-down list.
VM options	The parameters for starting a Java Virtual Machine (JVM), for example, -D, -Xms, and -Xmx.
Program arguments	The startup parameter, which is obtained by the args parameter in the main function.

Parameter	Description
Environment Variables	The environment variable parameters.
PORT	The port to be exposed in the app, for example, classic port 7001 or port 8080 for Spring Boot-based projects.
Machine	The type of the ECS instance used for debugging.
HotCode	This configuration takes effect only in Run mode. Alibaba Cloud's HotCode2 plug-in is used by default.

Start debugging

In the top navigation bar, choose **Debug > Start Debugging**.

The initial startup process takes a longer time because App Studio needs to prepare the runtime environment and download Maven dependencies. When you restart debugging, App Studio skips this process and provides user experience similar to that in a local IDE.

4.5.2. Online debugging

App Studio supports online debugging of Java apps and Spring Boot-based web projects.

Before online debugging, you must Configure the entry function and Start debugging.

Exposed service

After your app is started, two basic services are provided. You can click the link next to Backend to debug the backend Java code.

Panels

- Output panel

  The Output panel displays the standard output of all apps (System.in is not supported currently). It supports the ANSI color and ensures the consistent experience as a local terminal.

- Call Stack
- Breakpoint

  The Breakpoint panel displays the breakpoints that are currently set. For more information about the breakpoint types and usage, see Breakpoint types.

- PROBLEM

  The **PROBLEM** panel displays compilation problems of apps. You can click a record to go to the corresponding line in the file.

Breakpoints

App Studio supports normal line breakpoints, function breakpoints, and exception breakpoints. For more information, see Breakpoint types.

Debugging buttons

Button	Description
Continue	Resumes the current breakpoint to continue the current thread.
Step Over	Steps to the next line of code without entering the function.
Step Into	Enters the function.
Force Step Into	Forcibly enters the function if the Step Into button does not work for any reason. Different from Step Into, Force Step Into can lead the program to run from the breakpoint to the class library that comes with Java.
Step Out	Jumps out of the current function.
Restart	Currently, the Restart button is not perfect enough and may not be able to clean up the program. This button is being optimized.
Stop	Stops debugging.

4.5.3. Breakpoint types

App Studio supports normal line breakpoints, function breakpoints, and exception breakpoints.

Normal line breakpoint

You can click the blank area next to a line in the current file to generate a breakpoint for that line. The breakpoint also appears on the Breakpoint panel.

Function breakpoint

Different from a line breakpoint or an exception breakpoint, a function breakpoint triggers two events, namely, entry and exit. You can manually add a function breakpoint, or set a breakpoint at the place where the function is defined.

If the function breakpoint is triggered, the program stops when stepping into or out of the function.

Exception breakpoint

If an exception breakpoint is set, the program stops when encountering the exception.

As shown in the following figure, after index is triggered, the program stops in line 23 because **NullPointerException** appears.

4.5.4. Breakpoint operations

The Breakpoint panel displays the breakpoints that are currently set. This topic describes how to operate breakpoints.

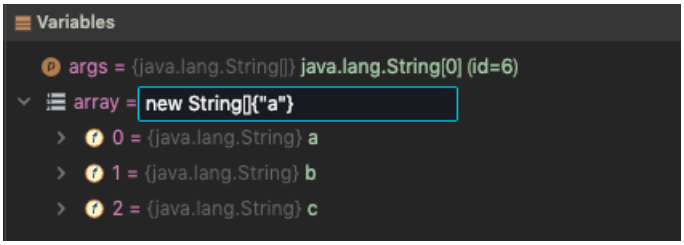
Breakpoints can be classified into normal line breakpoints, function breakpoints, and exception breakpoints. For more information, see Breakpoint types.

Debugging buttons

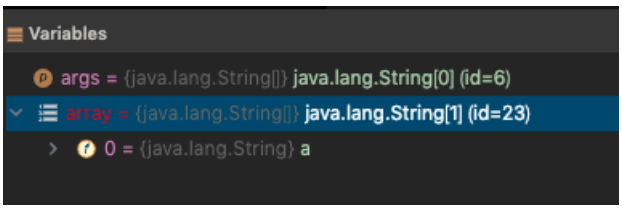
Button	Description
Continue	Resumes the current breakpoint to continue the current thread.
Step Over	Steps to the next line of code without entering the function.
Step Into	Enters the function.
Force Step Into	Forcibly enters the function if the Step Into button does not work for any reason. Different from Step Into, Force Step Into can lead the program to run from the breakpoint to the class library that comes with Java.
Step Out	Jumps out of the current function.
Restart	Currently, the Restart button is not perfect enough and may not be able to clean up the program. This button is being optimized.
Stop	Stops debugging.
Drop to Frame	Deletes the current stack and returns to the previous function.
Run to Cursor	Runs to the current line of code. You can set a temporary breakpoint in a line.
Evaluate Expression	Calculates an expression.

- Assign a value to a variable

  You can assign a value to a variable at a breakpoint.

  Double-click a field, create an expression to assign a value to the current variable, and then press Enter to make the setting effective.

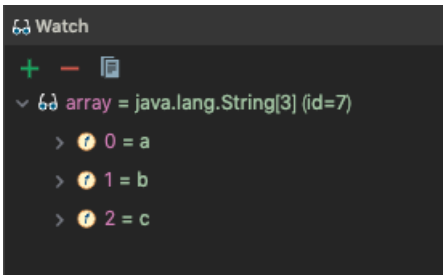
- Calculate an expression

  On the Evaluate Expression panel, enter an executable expression.

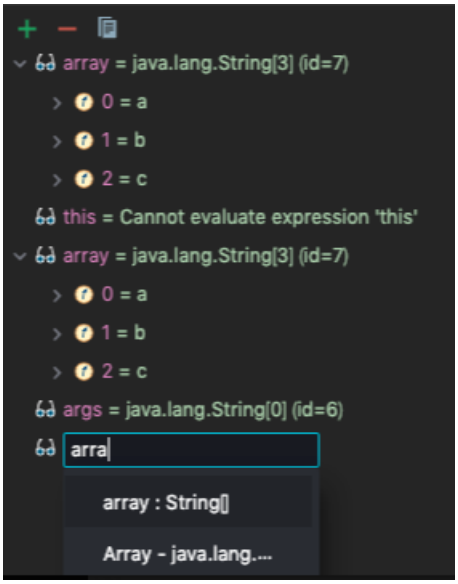
- Watch a variable

  Right-click a variable and select **Add Watch**.

The variable that is being watched appears on the right panel.

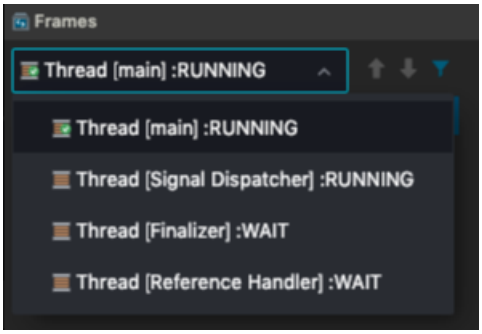


You can also manually add a variable in the Watch area.



- View threads

  You can view threads on the debugging panel.

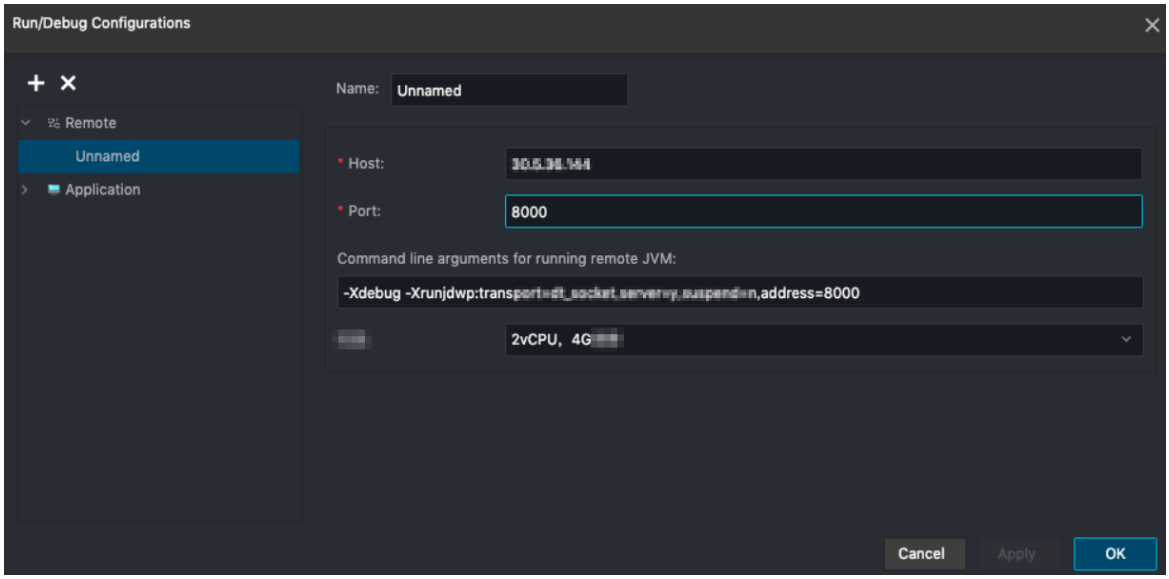


  Based on the running progress of the current thread, different information such as RUNNING or WAIT appears in the drop-down list. If you select another thread, information on the variable panel changes accordingly.

4.5.5. Remote debugging

You can only debug apps deployed in the daily environment where the ECS instance used for debugging runs.

1. Configure debugging information.



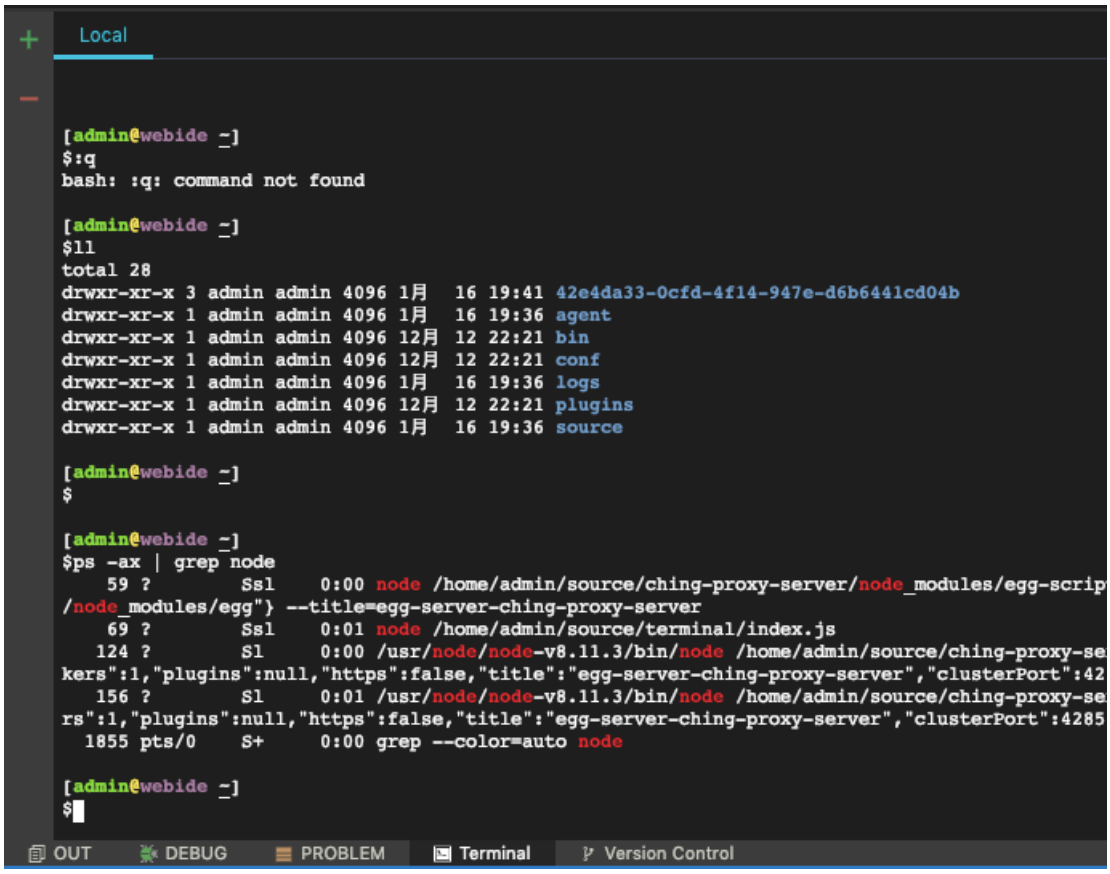
Note Set the Host and Port parameters to specify the remote service that the Java Virtual Machine Tool Interface (JVMTI) needs to connect to.

2. Click **Debug**. If the debugger information appears, the connection is successful. Then you can start debugging.

During remote debugging, the JVMTI is used for socket connection. The debugger and debuggee only transmit JVM running information between each other and do not transmit standard output or error output information.

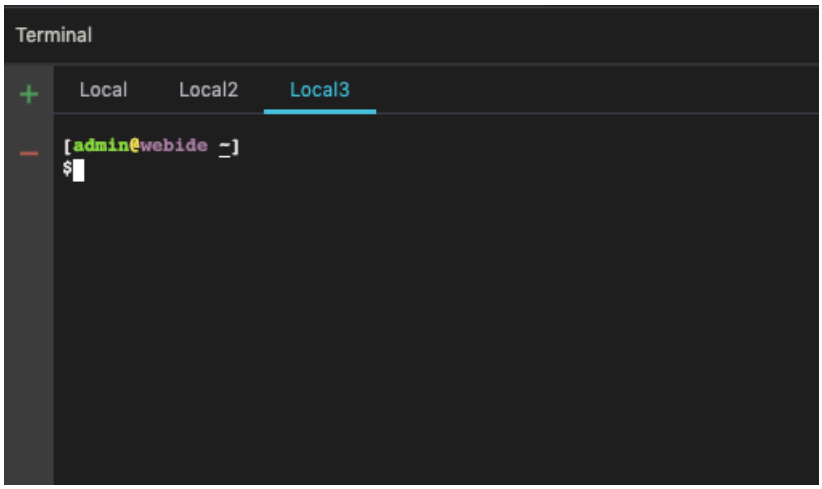
4.5.6. Terminal

The **Terminal** tab appears on the bottom of the panel.



App Studio supports common shell commands such as **ls** and **cat** and interactive commands such as **vi** and **top**.

You can also start multiple terminals.



4.5.7. Hot code replacement

Using the hot code replacement feature, you can edit the running code of an app and make the edits effective without restarting the app.

For example, after you edit the code while debugging a Spring Boot-based app, you do not need to restart the app. The edited code takes effect once it is saved. App Studio supports this feature by default.

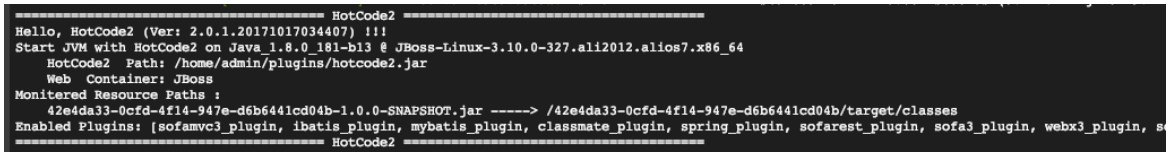
App Studio also supports hot code replacement while an app is running. To trigger hot code replacement, you only need to save the file without installing any plug-in or manually compiling the file.

If you are editing the code in Debug mode, App Studio automatically deletes the current running stack and returns to the function entry.

Configure hot code replacement in Run mode

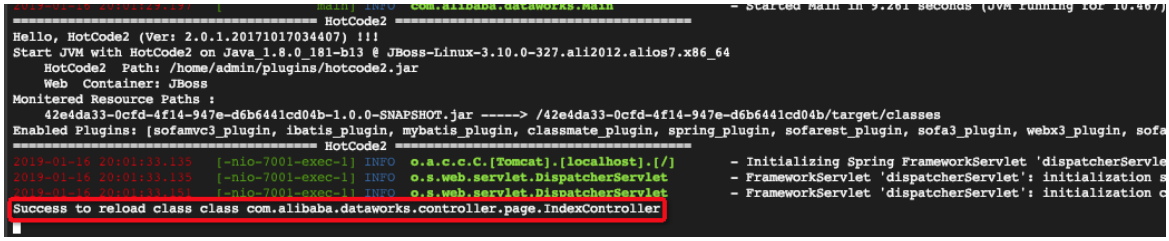
1. Enable hot code replacement on the Run/Debug Configurations page.

   After you click Run or Debug, the output information of the HotCode2 plug-in appears on the OUT tab.



2. Trigger hot code replacement.

   Save the file after editing it.



3. After the incremental code synchronization is completed, if the output of a reload class appears in the console, hot code replacement takes effect. The sample code is as follows:

```
public class IndexController {
    @RequestMapping("/")
    @ResponseBody
    public String index(){
        return "cccc";
    }
}
```

   You can replace the return string with another string to make the edit effective immediately.

Configure hot code replacement in Debug mode

You can use the native Java Debug Interface (JDI) to enable hot code replacement in Debug mode. However, due to Java Virtual Machine (JVM) restrictions, hot code replacement is unavailable when a method is added to or deleted from a class. You can save the file to trigger hot code replacement.

> **Note** The native JVM supports hot code replacement for operations such as adding or deleting a class. However, hot code replacement is unavailable when you change the class structure.

4.6. Collaborative coding

This topic describes how to invite collaborators, join a collaborative project, and view the status of collaborators on the collaborator panel. This topic also introduces permissions of collaborators.

App Studio supports real-time collaborative coding. Multiple collaborators of a team can develop and write code at the same time in the same project, and view changes made by other collaborators in real time. This feature helps avoid the hassle of synchronizing code and merging branches and significantly improve the development efficiency.

Invite a new collaborator

The project owner can invite other developers to join the project for collaborative coding.

1. Open the project that you want to share.
2. Click **Share** on the right to expand the collaborator panel.
3. Click **Invite** in the upper-right corner to enter the invitation process.
4. In the **Invite New Member** dialog box, set parameters.

Parameter	Description
Username	Enter the username of the collaborator to be invited.
Permission	Select **Read-Only** or **R/W** based on your business requirements.

5. Click **OK**.

Join a collaborative project

When you are invited to join another developer's project, you can click **Shared from Others** on the project panel to view the collaborative projects that you have joined. Click a project to join it for real-time collaborative coding.

Collaborator panel

During real-time collaborative coding, collaborators can view the status of each other.

1. Click **Share** on the right of the page. The list of collaborators appears.
2. View the **online status**, **file being edited**, and **permissions** of a collaborator.

> **Note** The project owner can click **Remove** to remove a collaborator.

Permission description

During collaborative coding, a collaborator may have the following permissions:

- Owner: The owner is the creator of the project and cannot be changed. The owner can invite other developers to join the project or remove other collaborators.
- Read/write permissions: Collaborators with the read/write permissions can view all files in the project and edit these files.
- Read-only permission: Collaborators with the read-only permission can only view the files in the project and cannot edit them.

4.7. App deployment

This topic describes how to create an app in App Studio and deploy it in the production environment to make it accessible through the Internet.

Go to App Studio

Create a project

1. Log on to the DataWorks console and click **Go to Data Development** next to a workspace.
2. Click the icon in the upper-left corner and hover the cursor over **All products**. Choose **App Development > App Studio**.
3. On the **App Studio** page, create a template-based or code-based project or import a project from Git.
4. Set project parameters as needed and click **Submit**.

Associate your project with Git

Before publishing an app, you need to initialize Git.

1. Create a repo on code.aliyun.com and take down the SSH URL of the repo.
2. Go to App Studio and click **Version**. Select **Connect to Remote Repo**.
3. In the **Connect to Remote Repo** dialog box that appears, set parameters and click **Submit**.

> **Note** If you have not bound an SSH key or Git email address, follow the instructions on the page.

Publish the app

After associating your project with Git, you can create an app through **Publish New Version**.

1. Return to the **Projects** page and click **Manage** on the project card.
2. On the project management page that appears, click **Publish New Version** in the upper-right corner. In the dialog box that appears, select the app for which a new version needs to be published.
3. Click **Publish**.

Deploy the app

1. After you click **Publish**, a guide page appears, as shown in the following figure.

   Go to the purchase page and purchase an App Studio workspace. Go to App Studio, create an ECS group, and then add the purchased ECS instance to the ECS group.

2. Click **Purchase Link**. On the page that appears, follow the instructions to purchase an App Studio workspace in your specified region.
3. Click **Manage Deployment** on the app card to go to the app O&M page.

   > **Note** You need to unbind the host bound previously.

4. Click **Create ECS Group** in the upper-right corner of the **ECS Group List** area and create an ECS

group.

5. Choose **Actions > Expand capacity**. In the dialog box that appears, add the ECS instance you just purchased to the newly created ECS group.
6. After the Apps page is refreshed, click **Publish** to publish the app to the default ECS group.

   The app has been deployed on your ECS instance and started to provide services.

Perform VPC deployment

VPC deployment refers to adding a VPC to the CIDR block of your purchased ECS instance. You need to perform VPC deployment for each project on Alibaba Cloud and App Studio. However, you need to perform this operation only once for each project. When there is an iterative version, you only need to deploy the app. For more information, see the preceding section.

Authorize VPC access

An ECS instance used for publishing apps in App Studio connects to your VPC through an ENI. You need to grant your service account of App Studio the permission to manage ENIs.

1. Go to the RAM Roles page and click **Create RAM Role**. Select **Alibaba Cloud Account** and click Next. Select **Other Alibaba Cloud Account** and enter 1591568227964362. Enter a custom name in **RAM Role Name** and click **OK**.

   > **Note** You must enter 1591568227964362 in **Other Alibaba Cloud Account**.

2. Click **Add permissions** for the newly created RAM role in the Actions column. Grant the RAM role the AliyunECSNetworkInterfaceManagement permission to manage ENIs. Then, click **OK**.
3. Click the RAM role and view the ARN.

Create a VPC and a vSwitch

You must create a VPC and a vSwitch in the region to which your App Studio workspace belongs. Here, China (Shanghai) is used as example.

Log on to the VPC console and create a VPC. For more information, see Create an IPv4 VPC.

> **Note** The IPv4 CIDR block of the VPC must be different from the CIDR block selected before the app is deployed.

After the vSwitch is created, take down the vSwitch ID on the vSwitch page for later use.

Create a security group

Log on to the ECS console and create a security group. For more information, see Create a security group.

After the security group is created, take down the security group ID for later use.

Add the VPC in App Studio

1. Click **O&M** in the upper-right corner of the **App Studio** page.
2. Choose **Resource > VPC** and click **Add VPC**.
3. In the **Add VPC** dialog box that appears, enter the recorded ARN, security group ID, and vSwitch ID in **Role ID**, **Security Group ID**, and **VSwitch ID**, respectively. Provide a brief description of your operation.

4. Click **Run**.

Create an ENI and associate it with an ECS instance

1. Click the **ID** of your VPC to go to the ENI management page.
2. Click **Add ENI**.
3. After the ENI is added, click **Associate with ECS instance**.
4. In the **Associate with ECS instance** dialog box that appears, set **VpcID**, **EniID**, **ECS Group**, and **ECS Instance**.

After the preceding operations are performed, App Studio creates an ENI for you and associates it with the specified ECS instance.

Enable access from the Internet

Next, you can associate the ENI with an elastic IP address to expose the app to the Internet. You can also enable SLB in this procedure.

The procedure for associating the ENI with an elastic IP address is as follows:

1. Log on to the VPC console and purchase an elastic IP address. For more information, see Apply for an EIP.
2. Associate the ENI with the elastic IP address. For more information, see Associate an EIP with an ECS instance.
3. After the preceding operations are performed, your app can be accessed through the Internet.

4.8. Access third-party services

4.8.1. DataService Studio

This topic describes how to check DataService Studio APIs that you have permissions to call in App Studio. This topic also describes how to generate code snippets to quickly access DataService Studio APIs through App Studio.

For more information about how to apply for and call DataService Studio APIs and SDKs, see DataService Studio.

Prerequisites

Before accessing DataService Studio, make sure that the following conditions are met:

- You have created a workspace in DataService Studio and applied for permissions to call the APIs for the workspace.

  This topic is applicable to DataService Studio APIs that you have permissions to call. Therefore, you must first go to **DataService Studio** and check whether a workspace is available and whether the APIs that you have permissions to call exist in the workspace.

- You have created a Java project in App Studio.

  The following section uses a Spring Boot-based project as an example to describe how to generate code snippets.

  i. Log on to App Studio. On the **Projects** page, click **Create Project from Code**.

  ii. On the **Create Project** page, specify **Project Name** and **Project Description**, and set Runtime

Environment to **springboot sample template**.

iii. After the configuration is completed, click **Submit**.

After the project is created, make sure that the pom.xml file contains the dependency of DataService Studio. For more information about the Maven coordinates, see Nexus Repository Manager.

```
<dependency>
  <groupId>com.aliyun.dataworks</groupId>
  <artifactId>aliyun-dataworks-dataservice-java-sdk</artifactId>
  <version>0.0.1-aliyun</version>
</dependency>
```

Use DataService Studio APIs in App Studio

You can use DataService Studio APIs through code or the WYSIWYG designer.

- Use DataService Studio APIs through code.

  The following section describes how to view available DataService Studio APIs in App Studio by keyword, project, and service group. You can also use the feature of generating code snippets to quickly create the code to call a DataService Studio API.

  i. View the list of DataService Studio APIs.

  Click the Data tab on the right. The list of DataService Studio APIs appears. You can filter the APIs by name, project, or service group.

  ii. Create an API on **DataService Studio**.

  You can click Create API in DataService Studio in the upper-right corner and create an API to call.

  iii. View details about DataService Studio APIs.

  Click **Details** in the Actions column of a DataService Studio API. The DataService Studio page appears, showing the details of the API.

  iv. Quickly generate the access code.

  App Studio allows you to create the access code with one click. It automatically enters the AppKey and AppSecret and generates the sample controller code, facilitating you to directly insert a project.

  Click **Select** in the Actions column of a DataService Studio API. The details page that includes the sample access code appears.

  The following section provides an example of the complete controller code. In the generated InvokeApi2252() method, the path, host, AppKey, and AppSecret required for accessing the DataService Studio API are automatically entered. ApiRequest2252DTO contains all parameters required for accessing the DataService Studio API.

```
package com.alibaba.dataworks.dataservice;
import com.aliyun.dataworks.dataservice.model.api.protocol.ApiProtocol;
import com.aliyun.dataworks.dataservice.sdk.facade.DataApiClient;
import com.aliyun.dataworks.dataservice.sdk.loader.http.Request;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import org.springframework.web.bind.annotation.RestController;
import java.lang.reflect.Field;
import java.util.HashMap;
/**
 * @author ****
 * @date 2019-03-21T17:23:17.040
 * - Before use, make sure that the pom.xml file contains the latest data-service-cli
ent dependency.
 *     <dependency>
 *         <groupId>com.alibaba.dataworks</groupId>
 *         <artifactId>data-service-client</artifactId>
 *         <version>${latest-data-service-version}</version>
 *     </dependency>
 * - Before use, make sure that the spring config class is separately configured and
is not combined with other config classes.
 *     @Configuration
 *     @ComponentScan(basePackageClasses = { DsClientConfig.class })
 *     public class DsClientConfig {
 *         @Bean
 *         public BeanRegistryProcessor beanRegistryProcessor(){
 *             return new BeanRegistryProcessor();
 *         }
 *     }
 */
@RestController
public class Test2252Controller {
    private Logger logger = LoggerFactory.getLogger(Test2252Controller.class);
    @Autowired
    private DataApiClient dataApiClient;
    /**
     * Sample Result:
     * {
     *     "data": {
     *         "totalNum": 1000,
     *         "pageSize": 100,
     *         "rows": [
     *             {
     *                 "pageNum": "...", // The number of the page. This is a default
pagination parameter. The value is an integer.
     *                 "pageSize": "...", // The number of entries on each page. This
is a default pagination parameter. The value is an integer.
     *                 "totalNum": "...", // The total number of pages. This is a def
ault pagination parameter. The value is an integer.
     *                 "id": "...", // Integer.
     *                 "name": "...", // String.
     *                 "sex": "...", // String.
     *                 "age": "...", // Integer.
     *             }
     *             ......
     *         ],
     *         "pageNum": 1
     *     },
     *     "errCode": 0,
     *     "requestId": "478cae2f-0***-42fb-a439-c0***e6f",
```

```
     *     "errMsg": "success"
     * }
     */
    private HashMap InvokeApi2252(ApiRequest2252DTO dto) throws Exception {
        Request request = new Request();
        request.setMethod("GET");
        request.setAppKey("15810204");
        request.setAppSecret("******************************");
        request.setHost("http://0e5e6cd70******5e64****hai.a***pi.com");
        request.setPath("/test");
        for (Field f : dto.getClass().getDeclaredFields()) {
            try{
                if(f.get(dto)!= null) {
                    request.getBodys().put(f.getName(), f.get(dto).toString());
                }
            }catch(Exception e){}
        }
        request.setApiProtocol(ApiProtocol.HTTP);
        return dataApiClient.dataLoad(request);
    }
    /**
     * Response:
     */
    @RequestMapping(value = "/sample/test2252", method = RequestMethod.POST)
    public HashMap testApi(@RequestBody ApiRequest2252DTO dto) throws Exception {
        return InvokeApi2252(dto);
    }
}
/**
 * Request
 */
class ApiRequest2252DTO {
    public Integer pageNum;
    public Integer pageSize;
    public Integer id;
    public String name;
    public String sex;
    public Integer age;
}
```

Note You can refer to the generated sample code in your code development. You can also click **Save** to add the code to the dataservice package in the current code directory.

- Use DataService Studio APIs through the WYSIWYG designer.

  Components of the WYSIWYG designer are highly integrated with DataService Studio APIs and use the default format of data returned by DataService Studio. This means any configuration can take effect immediately. For more information, see WYSIWYG designer.

4.8.2. DataOS API

App Studio provides the DataOS API and DataService Studio API. This topic describes the functions, request parameters, and response parameters of the DataOS API operations, and provides guidance on configuring and using the DataOS API.

CheckMetaTable

- Function: checks whether a table exists.
- Request: The tableGuid parameter is required.
- Syntax: `odps.<project>.<table>`.
- Response: true or false.
- Example:
  - Request: `request.setTableGuid("odps.autotest.daily_test");`
  - Response: `{"requestId":"0b85c9d915548770462378104e","errMsg":"success","errCode":0,"data":true}`

GetMetaDB

- Function: gets the information of a MaxCompute project.
- Request: The project GUID is required.
- Syntax: `odps.<project>`.
- Response: The details of the project are returned, including the parameters listed in the following table.

Parameter	Description
appGuid	The GUID of the project.
project	The name of the project in English.
projectNameCn	The name of the project in Chinese.
comment	The comments on the project.
ownerId	The ID of the project owner.
createTime	The time when the project was created.
modifyTime	The time when the project was modified.

- Example:
  - Request: `request.setDbGuid("odps.autotest");`

- Response:

```
{
    "requestId": "0bfaefec****61500671805e",
    "errMsg": "success",
    "errCode": 0,
    "data": {
        "appGuid": "odps.meta",
        "projectName": "meta",
        "projectNameCn": "ODPS metadata",
        "comment": "",
        "ownerId": "13101879118",
        "createTime": "2014-02-18",
        "modifyTime": "2018-04-16"
  }
}
```

GetMetaTable

- Function: gets the information of a MaxCompute table.
- Request: The tableGuid parameter is required.
- Syntax: `odps.<project>.<table>`.
- Response: The details of the table are returned, including the parameters listed in the following table.

Parameter	Description
appGuid	The GUID of the project.
tableGuid	The GUID of the table.
tableName	The name of the table.
id	The ID of the database.
ownerId	The ID of the project owner.
hasPart	Indicates whether the table is partitioned. The value 1 indicates that the table is partitioned. The value 0 indicates that the table is not partitioned.
dataSize	The size of data in the table.
createTime	The time when the table was created.
lastDdlTime	The last time when a Data Definition Language (DDL) statement was executed for the table.
lastModifyTime	The last time when the table was modified.

- Example:
  - Request: `request.setTableGuid(tableGuid);`

- Response:

```
{
    "requestId": "0b8906da****8175861e",
    "errMsg": "success",
    "errCode": 0,
    "data": {
        "appGuid": "odps.meta",
        "tableGuid": "odps.meta.m_table",
        "tableName": "m_table",
        "id": 64809,
        "OwnerId": "dp-base-odps@example.com",
        "hasPart": 1,
        "dataSize": 49397610904693,
        "createTime": "2014-12-10 21:20:23",
        "lastDdlTime": "2017-04-18 10:10:06",
        "lastModifyTime": "2019-04-09 20:24:08"
  }
}
```

ListMetaTableColumn

- Function: gets the column information of a MaxCompute table.
- Request: The tableGuid parameter is required.
- Syntax: `odps.<project>.<table>`.
- Response: The details of columns in the table are returned, including the parameters listed in the following table.

Parameter	Description
appGuid	The GUID of the project.
tableGuid	The GUID of the table.
tableName	The name of the table.
columnGuid	The GUID of a column. Syntax: `odps.<project>.<table>.<col>`.
columnName	The name of the column.
columnType	The type of the column.
seqNumber	The sequence number of the column, which starts from 1.
isPartitionCol	Indicates whether the column is partitioned. The value 0 indicates that the column is not partitioned. The value 1 indicates that the column is partitioned.
comment	The comments on the project.

Parameter	Description
safeLevel	The safety level of the project.

- Example:
  - Request: `request.setTableGuid(tableGuid);`
  - Response:

```
{
    "requestId": "0b8906d*****9796824e",
    "errCode": 0,
    "errMsg": "success",
    "columnList": [{
        "appGuid": "odps.meta",
        "tableGuid": "odps.meta.m_table",
        "tableName": "m_table",
        "columnGuid": "odps.meta.m_table.project_name",
        "columnName": "project_name",
        "columnType": "string",
        "seqNumber": 1,
        "isPartitionCol": 0,
        "comment": "project name",
        "safeLevel": "C2"
  },
{
    "appGuid": "odps.meta",
    "tableGuid": "odps.meta.m_table",
    "tableName": "m_table",
    "columnGuid": "odps.meta.m_table.name",
    "columnName": "name",
    "columnType": "string",
    "seqNumber": 2,
    "isPartitionCol": 0,
    "isPrimaryKey": 0,
    "isNullable": 0,
    "comment": "table name",
    "safeLevel": "C2"
  } ... ]
}
```

ListMetaTablePartition

- Function: gets the partition information of a MaxCompute table.
- Request:

Parameter	Description
tableGuid	The GUID of the table. Syntax: `odps.<project>.<table>`.
pageNum	The number of the page to return.

Parameter	Description
pageSize	The number of entries to return on each page.

- Response: The partition details of the table are returned, including the parameters listed in the following table.

Parameter	Description
appGuid	The GUID of the project.
tableGuid	The GUID of the table.
tableName	The name of the table.
partitionGuid	The GUID of a partition. Syntax: `odps.<project>.<table>.<partition>`.
partitionName	The name of the partition.
createTime	The time when the partition was created.
modifyTime	The time when the partition was modified.
dataSize	The size of data in the partition.
records	The number of entries in the partition.
pageNum	The number of the page that is returned.
pageSize	The number of entries on the page that is returned.
totalNum	The total number of entries.

- Response example:

```
{
    "requestId": "0baf3e0*****5025570e",
    "errCode": 0,
    "errMsg": "success",
    "pageNum": 1,
    "pageSize": 10,
    "totalNum": 1101,
    "partitionList": [{
        "appGuid": "odps.meta",
        "tableGuid": "odps.meta.m_table",
        "tableName": "m_table",
        "id": 168504514,
        "partitionGuid": "odps.meta.m_table.ds\u003d20190408",
        "partitionName": "ds\u003d20190408",
        "createTime": "2019-04-08 13:59:52",
        "modifyTime": "2019-04-08 19:54:51",
        "dataSize": 273248012568,
        "records": 720503170
  } ... ]
}
```


SearchMetaTables

- Function: performs fuzzy search in a table.
- Request:

Parameter	Description
keyword	The keyword of the table name.
pageNum	The number of the page to return.
pageSize	The number of entries to return on each page.

- Response:

Parameter	Description
appGuid	The GUID of the project.
tableGuid	The GUID of the table.
tableName	The name of the table.
ownerId	The ID of the project owner.
createTime	The time when the table was created.
lastDdlTime	The last time when a DDL statement was executed for the table.
lastModifyTime	The last time when the table was modified.

- Example:
  - Request: `request.setKeyword("test");`
  - Response:


```
{
    "message": null,
    "code": 200,
    "success": true,
    "data": {
        "requestId": "0be41b***22277597924e",
        "errCode": 0,
        "errMsg": "success",
        "pageNum": 1,
        "pageSize": 2,
        "totalNum": 5000,
    "data": [{
        "appGuid": null,
        "tableGuid": "odps.ant_p13n.finance_newsrec_tab_dataset_ds",
        "tableName": "finance_newsrec_tab_dataset_ds",
        "createTime": "2018-07-06 16:24:41",
        "lastModifyTime": "2019-04-26 10:49:23",
        "lastDdlTime": null,
        "lastAccessTime": null,
        "ownerId": "163585"
    },
    {
        "appGuid": null,
        "tableGuid": "odps.tbcdm.dws_tm_itm_cate_food_ftr_test_cm",
        "tableName": "dws_tm_itm_cate_food_ftr_test_cm",
        "createTime": "2017-11-23 17:06:18",
        "lastModifyTime": "2019-04-26 20:34:12",
        "lastDdlTime": null,
        "lastAccessTime": null,
        "ownerId": "108292"
    }]
  },
    "timestamp": 1556452227875,
    "sessionId": null
}
```


Call the DataOS API

Configure the pom file as follows:

```
<! --DataOS Start-->
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-dataworks-enterprise-ultimate</artifactId>
    <version>0.0.3</version>
</dependency>
<! -- JSON 2.8.5 or later -->
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-core</artifactId>
    <version>4.4.0</version>
</dependency>
<! --DataOS End-->
```

Configure the hosts file as follows:

```
# from src/main/resources/application.properties
# dataos api configuration
dataworks.dataos.auth.accessId= <indicate user accessid, refer to aliyun>
dataworks.dataos.auth.accessKey= <indicate user accessid, refer to aliyun>
dataworks.dataos.region=cn-shanghai
dataworks.dataos.endpoint=dataworks-ee-ue-share.cn-shanghai.aliyuncs.com
dataworks.dataos.product=dataworks-enterprise-ultimate
```

The Java code is as follows. When creating IClientProfile, you must specify the AccessKey ID and AccessKey Secret of your Alibaba Cloud account. For more information, see the following FAQ.

```
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.dataworks.model.v20171212.CheckMetaTableRequest;
import com.aliyuncs.dataworks.model.v20171212.CheckMetaTableResponse;
import com.aliyuncs.dataworks.model.v20171212.GetMetaDBRequest;
import com.aliyuncs.dataworks.model.v20171212.GetMetaDBResponse;
import com.aliyuncs.dataworks.model.v20171212.GetMetaTableRequest;
import com.aliyuncs.dataworks.model.v20171212.GetMetaTableResponse;
import com.aliyuncs.dataworks.model.v20171212.ListMetaTableColumnRequest;
import com.aliyuncs.dataworks.model.v20171212.ListMetaTableColumnResponse;
import com.aliyuncs.dataworks.model.v20171212.ListMetaTablePartitionRequest;
import com.aliyuncs.dataworks.model.v20171212.ListMetaTablePartitionResponse;
import com.aliyuncs.dataworks.model.v20171212.SearchMetaTablesRequest;
import com.aliyuncs.dataworks.model.v20171212.SearchMetaTablesResponse;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import com.google.gson.Gson;
public class Simple {
  IAcsClient client = null;
  @org.junit.Test
  public void testCheckMetaTable() throws ServerException, ClientException {
    String tableGuid = "odps.meta.m_table";
    CheckMetaTableRequest request = new CheckMetaTableRequest();
    request.setTableGuid(tableGuid);
```

```
    CheckMetaTableResponse response = client.getAcsResponse(request);
    System.out.println(new Gson().toJson(response));
  }
  @org.junit.Test
  public void testGetProject() throws ServerException, ClientException {
    String appGuid = "odps.meta";
    GetMetaDBRequest request = new GetMetaDBRequest();
    request.setDbGuid(appGuid);
    GetMetaDBResponse getMetaDBResponse = client.getAcsResponse(request);
    System.out.println(new Gson().toJson(getMetaDBResponse));
  }
  @org.junit.Test
  public void testGetPartitions() throws ServerException, ClientException {
    String tableGuid = "odps.meta.m_table";
    ListMetaTablePartitionRequest request = new ListMetaTablePartitionRequest();
    request.setTableGuid(tableGuid);
    request.setPageNum(1);
    request.setPageSize(10);
    ListMetaTablePartitionResponse response = client.getAcsResponse(request);
    System.out.println(new Gson().toJson(response));
  }
  @org.junit.Test
  public void testSearchTables() throws ServerException, ClientException {
    SearchMetaTablesRequest request = new SearchMetaTablesRequest();
    request.setKeyword("test");
    request.setPageNum(1);
    request.setPageSize(10);
    SearchMetaTablesResponse response = client.getAcsResponse(request);
    System.out.println(new Gson().toJson(response));
  }
    @org.junit.Test
    public void testGetColumns() throws ServerException, ClientException {
        String tableGuid = "odps.meta.m_table";
        ListMetaTableColumnRequest request = new ListMetaTableColumnRequest();
        request.setTableGuid(tableGuid);
        ListMetaTableColumnResponse response = client.getAcsResponse(request);
        System.out.println(new Gson().toJson(response));
    }
  @org.junit.Test
  public void testGetTable() throws ServerException, ClientException {
    String tableGuid = "odps.meta.m_table";
    GetMetaTableRequest request = new GetMetaTableRequest();
    request.setTableGuid(tableGuid);
    GetMetaTableResponse response = client.getAcsResponse(request);
    System.out.println(new Gson().toJson(response));
  }
  public Simple() throws ClientException {
    IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou", "<!!!! id>",
        "<!!! key>");
    DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", "dataworks", "dataworks-share.
aliyuncs.com");
    client = new DefaultAcsClient(profile);
  }
}
```

FAQ

- Why does the API operation calling fail, with the following information returned?

  ```
  Exception in thread "main" com.aliyuncs.exceptions.ClientException: InvalidApi.NotFound :
  Specified api is not found, please check your url and method.
  RequestId : B081CCF1-9F19-473E-9B99-68F202E7572B
  ```

  You do not have the permission to call the API operation.

- How do I query the AccessKey ID and AccessKey Secret?

  In the Alibaba Cloud console, click your account in the upper-right corner and select **accesskeys** from the drop-down list. Then, the AccessKey ID and AccessKey Secret appear.

4.9. WYSIWYG designer

4.9.1. Overview of the WYSIWYG designer

The WYSIWYG designer of App Studio is a tool provided to assist in developing frontend pages. It provides common webpage components that allow you to create frontend pages by simple drag-and-drop operations. This topic describes the features of the WYSIWYG designer.

Use a compatible framework

The WYSIWYG designer is compatible with React, Angular, and Vue because it uses a common description language to describe the properties such as structures, appearances, and actions of pages at the underlying layer.

Integrate a simple data processing method to meet complex interaction needs

App Studio integrates a global state management solution to manage page data and associate components.

Provide the code mode to build complex interactive pages

The WYSIWYG designer uses the common structured description language domain-specific language (DSL) at the intermediate layer to switch between the visualized drag-and-drop mode and code mode. You can directly edit the code based on DSL. This feature brings the advanced user experience for high-end developers.

Visually configure component association

The WYSIWYG designer allows you to visually link parameters of components to associate them.

Publish and run pages without building

App Studio compiles the intermediate DSL code online into the code that can be directly run in the browser for page rendering.

Connect to DataService Studio of DataWorks and quickly integrate data APIs

App Studio seamlessly connects to DataService Studio APIs of DataWorks, allowing you to debug the APIs in real time.

Provide abundant components and templates

App Studio provides abundant components. You can also customize components and upload them to the component library.

In addition, App Studio provides abundant templates. You can quickly create a page based on a template. You can also save the created page as a template and publish it to the template market for use by others.

4.9.2. Basic usage

This topic describes basic operations in the WYSIWYG designer, including creating a project and building a visual page.

Create a project

1. Log on to App Studio. On the **Projects** page, click **Create Project from Code**.
2. On the **Create Project** page, specify **Project Name** and **Project Description**, and set **Runtime Environment** to **appstudio sample template**.
3. After the configuration is completed, click **Submit**.
4. Go to the *santa/pages* directory.
5. Click any .santa file to go to the WYSIWYG designer.

   You can also right-click **pages** and choose **Create > Template** to develop the page based on a template.

Build a visual page

The WYSIWYG designer consists of the component menu and operation panel.

- Component menu

  The component menu lists all components that the WYSIWYG designer presets, including layout components, basic components, form components, chart components, and advanced components.

- Operation panel

  You can click the corresponding icon on the operation panel to switch to the code mode, configure the navigation, configure a global data flow, revoke or redo an operation, preview the rendering result, or save edits.

- Visual operation area

  Select a component from the component menu and drag and drop it to the visual operation area.

- Component property configuration panel

  The component property configuration panel consists of the Properties, Style, and Advanced Settings tabs.

Click the Navigation Settings icon in the upper-right corner of the operation panel to go to the page for configuring the navigation of an app.

On the navigation configuration page, you can configure the public header, sidebars, and menu of the app.

The WYSIWYG designer adds the public header and sidebars to an app by default. You can click the Navigation Settings icon to customize the navigation configuration, for example, hiding the sidebars. The system supports the following configuration:

- You can set the following parameters for the header:
  - Logo Image
  - Title
  - Menu Items
  - Enabled
  - Fix to Page Top
  - Theme: Valid values: Dark and Light.
- You can set the following parameters for the sidebars:
  - Menu Items
  - Enabled
  - Enable Folding
  - Theme: Valid values: Dark and Light.

Configure a global data flow

For more information, see Global configuration.

- Configure component properties

  On the Properties tab, you can visually configure component properties.

  Based on the rules for configuring component properties, a visual form is generated on the Properties tab. After you configure component properties in this form, the WYSIWYG designer re-renders the component in the visual operation area based on the new properties. You can view the rendering results of the component with different properties in real time.

- Configure component styles

  On the Style tab, you can configure the styles of a component.

  A visual panel for configuring common styles is provided on the Style tab. On this panel, you can customize the basic styles of a component, including the layout, text, background, border, and effect.

  After you add or modify the component styles on this tab, the WYSIWYG designer collects all the style settings and re-renders the component in the visual operation area based on the new component style. You can view the component configuration effect in real time.

- Configure association between components

  On the Advanced Settings tab, you can configure association between components.

  Select a component in the visual operation area and click the Advanced Settings tab. The properties of the selected component are listed on the left of the tab. Click the icon on the right and select the component to be associated to your selected component.

  The properties of the associated component appear on the right of the tab.

- Select a property, for example, searchParams, in the left property list and connect it to a property, for example, requestParams, in the right property list.

In this way, any change of the searchParams parameter of the left component is transferred to the requestParams parameter of the right component in real time. This achieves property-based association between the two components.

Configure the code mode

By using the code mode, you can implement complex interactions in a more advanced way. For more information, see Code mode.

Save, preview, run, and hot code replacement

For more information, see Save, preview, run, and hot code replacement.

4.9.3. Common components

The WYSIWYG designer comes with more than 80 components to fully meet your needs for building basic pages. This topic describes the default components of the WYSIWYG designer.

Layout components

The layout components include a 24-grid system component.

- Grid Ratio

  By default, the system splits the 24-grid component with a ratio of 12:12. You can also set the grids to other common ratios or customize a ratio. When customizing a ratio, make sure that the sum of grids in the ratio is 24. The system splits the layout based on the grid ratio that you set.

- Horizontal Arrangement

  This parameter specifies the arrangement of grids in the parent node.

- Vertical Arrangement

This parameter specifies the vertical alignment mode of sub-elements.

- Grid Gutter

  Grids are usually separated by gutters. You can set this parameter to specify the width of the gutter.

- Block Container

  A block container component can be used as the parent component of certain components. It is similar to the div container in HTML.

Basic components

All basic components support common property settings related to components.

- Text
  - Text
  - Paragrap
    - **Component Size**

      This parameter specifies the size of text in a paragraph.

    - **Paragraph Display Method**

      A paragraph is used to distinguish short text from long text. The line spacing of short text is smaller (usually fewer than three lines).

- Media
  - Video
    - **Video Url**: specifies the URL of the video to be played.
    - **Thumbnail Url**: specifies the URL of the video thumbnail.
    - **Enable Automatic Playback**: specifies whether to automatically play the video after the component is loaded.

- Image

  Image Url: specifies the URL of the image to be displayed. You can upload an image.

- Icon
  - **Icon Size**

    This parameter specifies the display size of the icon.
  - **Icon Type**

    This parameter specifies the type of the icon.
- Button

  For more information about the button properties, see Button documentation.
- Link
  - **Link Text**: specifies the displayed text of the link.
  - **Link Url**: specifies the URL to be jumped to.
  - **Link Property**: specifies whether to open the link in the current window or in a new window.

Form components

Forms can be classified into in-line forms, horizontal forms, and vertical forms.

For more information about how to upload images and attachments, see Upload.

For more information about how to filter data, see Search.

For more information about the input box, see Input.

Chart components

- Data table

Parameter	Description
Data Source	The API address to which a request is sent.
Request Method	The request method. Valid values: Get, Post, Put, and Delete.
Search Parameters	The requested parameter.
Response Data Processing Function	The function that processes data returned by the API.
Table Columns	The column to be displayed in the data table.
Table Size	The size of the table.
Show Table Border	Specifies whether to display the table borders.
Show Table Header	Specifies whether to display the table header.

  For paged data tables, you can also specify the number of records that are displayed on each page.

- Excel table

Parameter	Description
Data Source	The API address to which a request is sent.
Request Method	The request method. Valid values: Get, Post, Put, and Delete.
Search Parameters	The requested parameter.
Response Data Processing Function	The function that processes data returned by the API.
Data	The data to be displayed in the Excel table.

- Line chart

Parameter	Description
Data Source	The API address to which a request is sent.
Request Method	The request method. Valid values: Get, Post, Put, and Delete.
Search Parameters	The requested parameter.
Response Data Processing Function	The function that processes data returned by the API.
Chart Configuration	The code used to configure the chart.
Show Chart Title	Specifies whether to display the chart title.
Chart Title	The chart title to be displayed.
Chart Data	The data to be displayed on the chart.
X-axis Field	The name of the field to be displayed on the X axis in the returned data.
Y-axis Field	The name of the field to be displayed on the Y axis in the returned data.

> **Note** You can configure components in a column chart, bar chart, area chart, pie chart, map, word cloud, or scatter chart in the same way as configuring a line chart.

Advanced components

All advanced components support common property settings related to components.

- Selection-oriented components include Select, Checkbox, CascadeSelect, Radio, Range, Switch, and Rating.
- **Tab**: This component is used to switch between tasks, views, and modes. It is used for global navigation and allows you to view and switch between global features. For more information, see Tab.
- **Slider**: This component is used to horizontally display various content on the page as slides. For more information, see Slider.

- **Step**: This component is used for display by default. For an upper-layer component, you can modify the value of the current parameter to set the current step number. You can also set the click event on each node to customize a callback. For more information, see Step.
- **Progress**: This component is used to display the current progress of your operation. For more information, see Progress.
- **Menu**: You can select a menu as required. For more information, see Menu.
- **Nav**: This component consists of the top navigation and side navigation. The top navigation provides global categories and features, while the side navigation provides a multi-level structure to display and arrange website architectures. For more information, see Nav.

4.9.4. Code mode

By using the code mode, you can implement complex interactions in a more advanced way.

Click the Code Mode icon in the upper-right corner of the operation panel to enable the code mode.

The code area appears on the right of the page.

The WYSIWYG designer uses DSL at the intermediate layer to switch between the visualization mode and code mode. DSL can be considered as a simplified version of React. The DSL syntax is basically the same as the React syntax.

As shown in the code area in the preceding figure, DSL uses a tag to describe a component. The tag properties are the component properties. The property value can be of a simple data type such as a string or a number. The property value can also be an expression. You can enter `state.xxx` to obtain data from the global data flow.

The code mode has the following features:

- If you drag and drop a component or configure the component properties in the visualization area, the edits are updated in the code in real time.
- If you edit the code in the code area, the edits are updated in the visualization area in real time.
- The drag-and-drop operation and component property configuration in the visualization area and code edits in the code area can be converted between each other.

4.9.5. DSL syntax

DSL is a component-based language developed based on the features of React JSX and Vue templates and is more suitable for UI layout design.

JSX

The DSL syntax is similar to the JSX syntax in the React.render method. The following section provides a brief description of JSX:

- `{}` is used to switch an HTML scope to a JavaScript scope. In a JavaScript scope, you can write any valid JavaScript expression. The return value is displayed on the page. For example, `<div>{'Hello' + ' Relim'}</div>`.

  > **Note** You can write any JavaScript expressions such as computing statements or literals in `{}`.

- An HTML tag is used to switch a JavaScript scope to an HTML scope. For example, `{<div>Hello Reli`

`m</div>}` .

- The HTML scope and JavaScript scope can be nested. For example, `{<div>{'Hello' + ' Relim'}</div>}` .

For more information about JSX syntax, see React JSX.

Valid JavaScript expressions

```
// Computing statements
{aaa} // √ Variable aaa must be defined.
{aaa * 111} // √
{1 == 1 ? 1 : 0} // √
{/^123/.test(aa)} // √
{[1,2,3].join('')} // √
{(()=>{return 1})()} // Self-executing function √
// Literals
{1}
{true}
{[11,22,33]} // √
{{aa:"11",bb:"22"}} // √
{()=>1} // Describe a function, which is valid but meaningless. √
```

Note If certain complex logic must be implemented by multiple computing statements rather than only one statement, you can wrap the logic in a self-executing function, which must be a valid expression. Example:

```
{(function(){
    // Sum the even digits of a number array.
    var input = [1,2,3,4,5,6,7,8,9,10];
    var temp = input.filter(i => i % 2 == 0)
    return temp.reduce((buf, cur) => buf + cur, 0)
})()}
```

Invalid JavaScript expressions

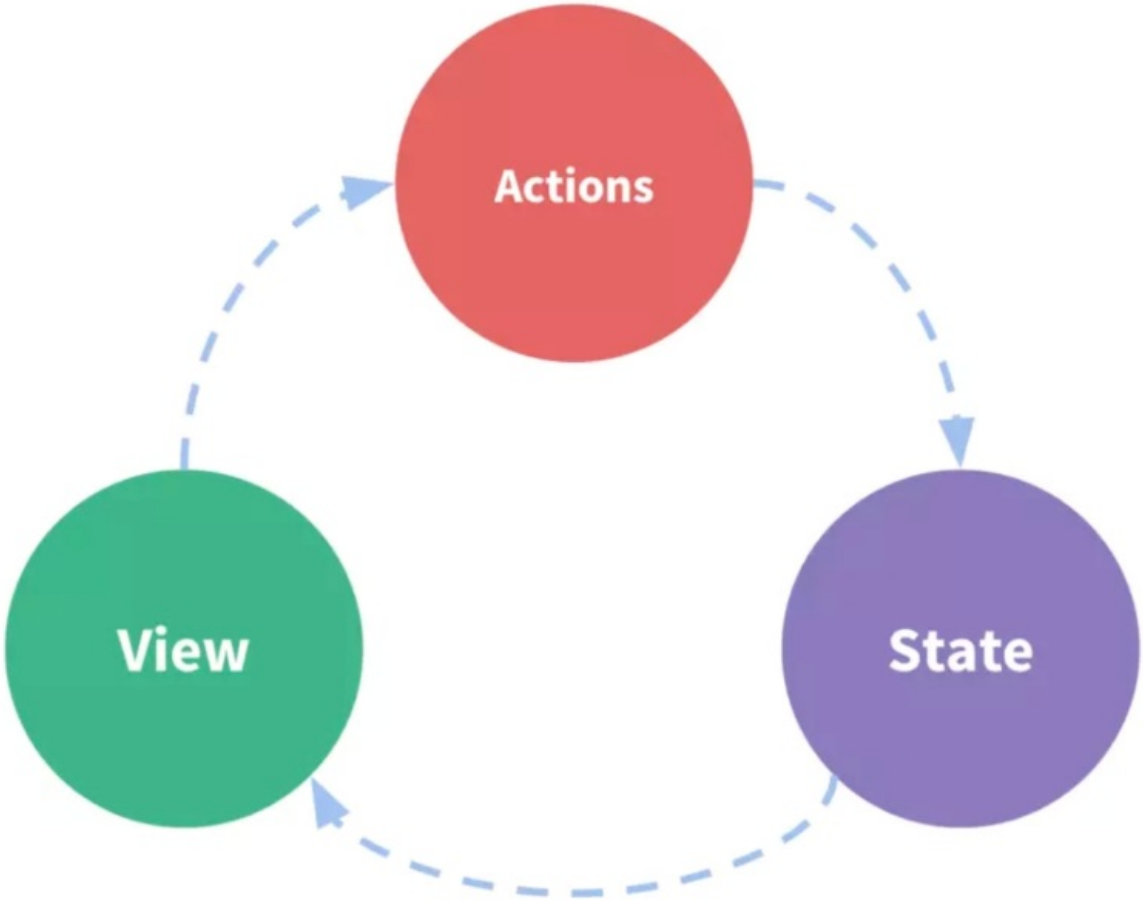
```
{ var a = 1 } // Value assignment statement
{ aaa * 111; 2} // Multiple statements separated with semicolons (;)
```

4.9.6. Global data flow

A global data flow is used for frontend data management. For multiple components that need to share a state, it is difficult to transfer the state among them. To resolve this issue, you can extract the shared state and use a global data flow to transfer it to all related components.

Principles

In a global data flow, global data is transferred in a globally unique way. Once the data declared in global data changes, the data flow shown in the following figure is executed.



1. A component triggers an action when, for example, a user clicks the component.
2. The action triggers global data changes.
3. Upon the global data changes, components that reference the global state are automatically re-rendered.

Scenarios

A global data flow is applicable to the association of two or more components on a page. You can refine public data into global data for unified management, and then use a global data flow to associate two or more components.

Configuration

1. Click the icon for configuring a global data flow in the upper-right corner.
2. In the dialog box that appears, enter the variable name and value.
   - The variable value can be a number, character string, or JSON string.
   - The variable value is declared as an API address. Data obtained from the API is automatically used as the value of the variable name.

Usage

- Obtain global data

Use `state.name` in the component to obtain global data.

```
<Input value={state.name} />
```

- Modify global data

  Use the $setState() method in the component to modify global data.

  ```
  <Input onChange={value => $setState({ name: value })} />
  ```

  > **Note** You must use the $setState() method to modify global data. If you use `state.name = 'new value'`, re-rendering cannot be triggered.

4.9.7. Navigation configuration

This topic describes how to configure the site navigation in the WYSIWYG designer.

The WYSIWYG designer provides each app with a public page header, a public bottom bar, and public sidebars, where you can configure various menus and themes. You can also specify whether to display the public header, bottom bar, and sidebars as required.

Click the **Navigation Settings** icon in the upper-right corner to go to the navigation configuration page.

Configure the public header

You can configure the public header based on your business requirements.

Parameter	Description
Enabled	Specifies whether to display the public header.
Theme	The theme. You can select a dark or light theme.
Logo Image	The site logo image. You can enter an image URL or upload a local image.
Title	The site title.
Fix to Page Top	Specifies whether to fix the public header to the top of the page. If you turn on this switch, the public header stays at the top of the page when the page scrolls.
Menu Items	The menu items such as the link name and link URL that are displayed in the public header.

Configure the sidebars

You can configure the sidebars based on your business requirements.

Parameter	Description
Enabled	Specifies whether to display the sidebars.

Parameter	Description
Theme	The theme. You can select a dark or light theme.
Title	The site title.
Enable Folding	Specifies whether the sidebar menus can be collapsed.
Menu Items	The menu items such as the link name and link URL that are displayed in the sidebars.
Automatically Expand All Menus	Specifies whether all menus (including submenus) can be automatically expanded.

Configure the public bottom bar

You can configure the public bottom bar based on your business requirements.

Parameter	Description
Enabled	Specifies whether to display the public bottom bar.
Content	The text that appears in the public bottom bar.

4.9.8. Save, preview, run, and hot code replacement

In the WYSIWYG designer, you can perform operations such as saving edits, previewing the rendering result, running an app, or making edits in hot code replacement mode.

Save

The WYSIWYG designer periodically saves your edits. You can also click the **Save** icon in the upper-right corner of the operation panel to save edits.

Preview

In the WYSIWYG designer, code in the operation area is in the editable status. However, special processing is added for the editable status of some components. For these components, you can run the rendering logic only when the app is running. To preview the rendering result, click the Preview icon in the upper-right corner of the operation panel.

Run

In the WYSIWYG designer, you can open and edit only one santa file at a time. To view the effect of the entire app,

click the Run Program icon on the Debug panel of App Studio to run the app.

Hot code replacement

If you are not satisfied with any page after running the app, you can edit the code in the WYSIWYG designer and save the edits.

The edited code takes effect on the running page in hot code replacement mode.

4.9.9. Save as template

You can save a created frontend page as a template for later use or share it with other users.

1. Click **Template** in the upper-right corner.
2. Click **Confirm** to save the template.
3. Choose **santa > pages**. Right-click pages and choose **Create > Template**.

The template that you saved appears. You can use it to create a page and develop features.