ALIBABA CLOUD

阿里云

Serverless 工作流

SDK参考

文档版本：20220610

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
`Courier字体`	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.SDK概述 05
2.Go SDK使用示例 06
3.Java SDK使用示例 09
4.Node.js SDK使用示例 13
5.PHP SDK使用示例 15
6.Python SDK使用示例 17

1.SDK概述

本文介绍Serverless工作流提供的编程语言的SDK。

支持语言列表

- Java
- Python
- PHP
- .NET
- Go
- Node.js

2.Go SDK使用示例

本文介绍使用Go SDK的详细流程，包括环境要求、安装和快速使用三部分。

环境要求

您的系统需要达到环境要求，例如安装了1.10.x或以上版本的Go环境。

安装

使用 `go get` 下载安装Go SDK 。

```
$ go get -u github.com/aliyun/alibaba-cloud-sdk-go/sdk
```

如果您使用了glide管理依赖，您也可以使用glide来安装Alibaba Cloud SDK for Go。

```
$ glide get github.com/aliyun/alibaba-cloud-sdk-go
```

另外，Alibaba Cloud SDK for Go也会发布在 https://develop.aliyun.com/tools/sdk#/go。

快速使用

在您开始之前，您需要注册阿里云账号并获取您的凭证。下文将以创建一个流程，发起一次执行并获取执行详情为例展示如何使用Go SDK调用 Serverless工作流服务。

请求方式

```
package main
import (
    "fmt"
    "time"
    "github.com/aliyun/alibaba-cloud-sdk-go/sdk/requests"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/fnf"
)
var (
    flowDefinitionType = "FDL"
    flowName           = "xxx"
    flowDefinition     = `xxx`
    flowDescription    = "some descriptions"
    roleArn            = "acs:ram::${Your_Account_ID}:${Your_Role}"
    executionName      = "xxx"
)
// CreateFlow ...
func CreateFlow(cli *fnf.Client) (*fnf.CreateFlowResponse, error) {
    request := fnf.CreateCreateFlowRequest()
    request.Name = flowName
    request.Definition = flowDefinition
    request.Description = flowDescription
    request.Type = flowDefinitionType
    request.RoleArn = roleArn
    return cli.CreateFlow(request)
}
// StartExecution ...
func StartExecution(cli *fnf.Client) (*fnf.StartExecutionResponse, error) {
    request := fnf.CreateStartExecutionRequest()
    request.FlowName = flowName
    request.ExecutionName = executionName
    return cli.StartExecution(request)
}
// DescribeExecution ...
func DescribeExecution(cli *fnf.Client) (*fnf.DescribeExecutionResponse, error) {
    request := fnf.CreateDescribeExecutionRequest()
    request.FlowName = flowName
    request.ExecutionName = executionName
    return cli.DescribeExecution(request)
}
// GetExecutionHistory ...
func GetExecutionHistory(cli *fnf.Client) (*fnf.GetExecutionHistoryResponse, error) {
    request := fnf.CreateGetExecutionHistoryRequest()
    // request.Limit and request.NextToken can set here. For easy demo, we passed.
    request.FlowName = flowName
    request.ExecutionName = executionName
    return cli.GetExecutionHistory(request)
}
```

创建客户端并利用上述函数发起一系列调用

> **说明** 如果您需要不加改造进行调试的话，请将下述函数与上述请求方式代码块置于同一个文件中，避免在import时报错。

```
func main() {
    fnfCli, err := fnf.NewClientWithAccessKey("cn-hangzhou", "AccessID", "AccessKey")
    if err != nil {
        panic(err)
    }
    // Create a flow
    _, err = CreateFlow(fnfCli)
    if err != nil {
        panic(err)
    }
    // StartExecution
    _, err = StartExecution(fnfCli)
    if err != nil {
        panic(err)
    }
    time.Sleep(time.Second)
    // DescribeExecution
    desResp, err := DescribeExecution(fnfCli)
    if err != nil {
        panic(err)
    }
    fmt.Println(fmt.Sprintf("%s status: %s", desResp.Name, desResp.Status))
    // GetExecutionHistory
    _, err = GetExecutionHistory(fnfCli)
    if err != nil {
        panic(err)
    }
}
```

3.Java SDK使用示例

本文介绍使用Java SDK的详细流程，包括环境要求、安装依赖和快速使用。

环境要求

- 要使用Alibaba Cloud SDK for Java，您需要一个云账号以及一对 `AccessKey ID` 和 `AccessKey Secret` 。请在阿里云控制台中的AccessKey管理页面上创建和查看您的AccessKey，或者联系您的系统管理员。
- 要使用Alibaba Cloud SDK for Java访问某个产品的API，您需要事先在阿里云控制台中开通这个产品。
- Alibaba Cloud SDK for Java需要1.6以上的JDK。

安装依赖

无论您要使用哪个产品的开发工具包，都必须安装 `aliyun-java-sdk-core` 。例如，对Serverless工作流SDK的调用，您需要安装 `aliyun-java-sdk-core` 和 `aliyun-java-sdk-fnf` 。

下文推荐您通过Maven来管理项目依赖。如果您使用Apache Maven来管理Java项目，只需在项目的 `pom.xml` 文件加入相应的依赖项即可。

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-core</artifactId>
    <version>[4.3.2,5.0.0)</version>
</dependency>
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-fnf</artifactId>
    <version>[1.0.0,5.0.0)</version>
</dependency>
```

如果Maven没有从中央存储库下载JAR包，则需要将此依赖项添加到 `pom.xml` 文件中，否则将报告 `NoClassDefFoundError` 异常。

```
<dependency>
    <groupId>com.google.code.gson</groupId>
    <artifactId>gson</artifactId>
    <version>2.8.5</version>
</dependency>
```

快速使用

下文以创建一个流程，发起一次执行并获取执行详情为例展示如何使用Java SDK调用Serverless工作流服务。

调用Alibaba Cloud SDK for Java的3个主要步骤：

1. 创建Default AcsClient实例并初始化。
2. 创建API请求并设置参数。更多信息，请参见API概览。
3. 发起请求并处理应答或异常。

注意 下文仅提供Serverless工作流产品的使用流程，如果您在使用过程中遇到调试等问题或希望使用高级功能（连接池、HTTPS、代理和日志）等功能，请参见README-CN。

请求方式

```
package com.test;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.fnf.model.v20190315.*;
class FnFOperations {
    static String flowName = "xxx";
    static String execName = "xxx";
    static String flowDesc = "xxx";
    static String flowDef = "xxx";
    static String roleArn = "xxx";
    static String flowType = "xxx";
    //创建一个Serverless工作流流程。
    static CreateFlowResponse createFlow(IAcsClient fnfClient) throws ClientException {
        CreateFlowRequest request = new CreateFlowRequest();
        request.setName(flowName);
        request.setDefinition(flowDef);
        request.setType(flowType);
        request.setDescription(flowDesc);
        request.setRoleArn(roleArn);
        return fnfClient.getAcsResponse(request);
    }
    //发起一次执行。
    static StartExecutionResponse startExecution(IAcsClient fnfClient) throws ClientExcepti
on {
        StartExecutionRequest request = new StartExecutionRequest();
        request.setFlowName(flowName);
        request.setExecutionName(execName);
        return fnfClient.getAcsResponse(request);
    }
    //查询执行结果。
    static DescribeExecutionResponse describeExecution(IAcsClient fnfClient) throws ClientE
xception {
        DescribeExecutionRequest request = new DescribeExecutionRequest();
        request.setFlowName(flowName);
        request.setExecutionName(execName);
        return fnfClient.getAcsResponse(request);
    }
    //查询历史执行记录。
    static GetExecutionHistoryResponse getExecutionHistory(IAcsClient fnfClient) throws Cli
entException {
        GetExecutionHistoryRequest request = new GetExecutionHistoryRequest();
        request.setFlowName(flowName);
        request.setExecutionName(execName);
        return fnfClient.getAcsResponse(request);
    }
}
```

创建客户端并利用上述函数发起一系列调用

说明 如果您需要不加改造进行调试的话，请将下述public类与上述“请求方式”代码块置于同一个文件中，避免在import时报错。

```
public class Main {
    public static void main(String[] args) {
        // Create DefaultAcsClient
        DefaultProfile profile = DefaultProfile.getProfile(
                "<your-region-id>",           // 地域ID。
                "<your-access-key-id>",       // RAM账号的AccessKey ID。
                "<your-access-key-secret>"); // RAM账号的AccessKey Secret。
        IAcsClient client = new DefaultAcsClient(profile);
        try {
            // Create Flow
            CreateFlowResponse createFlowResponse = FnFOperations.createFlow(client);
            System.out.println(creatFlowResponse);
            // Start Execution
            StartExecutionResponse startExeResp = FnFOperations.startExecution(client);
            System.out.println(startExeResp);
            // Describe Execution
            DescribeExecutionResponse descExeResp = FnFOperations.describeExecution(client)
;
            System.out.println(descExeResp);
        } catch (ClientException e) {
            e.printStackTrace();
        }
        try {
            GetExecutionHistoryResponse resp = FnFOperations.getExecutionHistory(client);
            for (GetExecutionHistoryResponse.EventsItem event:resp.getEvents()) {
                System.out.printf("event %s status: %s%n", event.getStepName(), event.getTy
pe());
            }
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

4.Node.js SDK使用示例

本文介绍使用Node.js SDK的详细流程，包括安装和快速使用两部分。

安装

SDK使用npm安装，示例如下。

```
npm install @alicloud/fnf-2019-03-15
```

另外，Alibaba Cloud SDK for Node.js也会发布在GitHub。

快速使用

在您开始之前，您需要注册阿里云账号并获取您的凭证。简单示例如下。

```
const FnFClient = require('@alicloud/fnf-2019-03-15');
async function demo() {
  const client = new FnFClient({
    // 需要手动填写服务地址，请参考API参考-调用方式-接入地址
    endpoint: '{endpoint}',
    accessKeyId: 'xxx',
    accessKeySecret: 'xxx'
  });
  // 创建流程
  const createResp = await client.createFlow ({
    Name : 'test',
    Definition :
`version: v1
type: flow
steps:
  - type: pass
    name: pass1`,
      Description : 'test',
    Type : 'FDL'
  });
  console.log("create: %s", createResp)
  // 开始一次执行
  const startResp = await client.startExecution ({
    FlowName: 'testabc'
  });
  console.log("start: %s", startResp)
  // 查询执行结果
  const getResultResp = await client.getExecutionHistory ({
    FlowName: 'test',
    ExecutionName: 'xxx'
  });
  console.log("start: %s", getResultResp)
  // 更新流程
  const res = await client.updateFlow ({
    Name : 'test',
    Definition :
`version: v1
type: flow
steps:
  - type: pass
    name: pass2`
  });
  console.log("%s", res)
}
demo();
```

5.PHP SDK使用示例

本文介绍使用PHP SDK的详细流程，包括安装和快速使用两部分。

安装

SDK使用composer安装。

```
composer require alibabacloud/fnf
```

如果下载速度过慢，您可以将composer镜像切换为中国内地镜像。

```
composer config repo.packagist composer https://mirrors.aliyun.com/composer/
```

Alibaba Cloud SDK for PHP也会发布在Github。更多信息，请参见README-zh-CN。

快速使用

在您开始之前，您需要注册阿里云账号并获取您的凭证。简单示例如下。

```
<?php
// 如果您在本地使用composer下载依赖并进行调试，请添加下述引用。
require 'vendor/autoload.php';
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Fnf\Fnf;
// 设置全局客户端。
AlibabaCloud::accessKeyClient('foo', 'bar')
            ->regionId('cn-hangzhou')
            ->asDefaultClient();
try {
    // 访问产品APIs
    $request1 = Fnf::v20190315()->createFlow();
    // 设置选项或参数并执行请求。
    // 创建一个Flow。
    $result1 = $request1->withDefinition('xxx') // API的参数。
                                ->withName('xxx')
                                ->withRoleArn('xxx')
                                ->withDescription('xxx')
                                ->withType('FDL')
                                //->scheme("https")
                                //->client('client1') // 指定发送客户端，否则使用全局客户端。
                                ->debug(true) // 开启调试会输出详细信息。
                                ->connectTimeout(10) // 连接超时会抛出异常。
                                ->timeout(10) // 超时会抛出异常。
                                ->request(); // 执行请求。
    // 开始一个执行。
    // 下面示例展示了如何直接使用options传入SDK调用的相关参数。
    // 也可以参照$request1使用withxxx方式调用。
    $options = [
                    'debug'=>true,
                    'connect_timeout'=>10,
```

```
                    'timeout'=>10,
                    'form_params'=>[
                        'FlowName'=>'xxx',
                        'Input'=>'{"execID": "exe1"}',
                    ],
                ];
    $result2 = Fnf::v20190315()
                                    ->startExecution($options)
                                    ->options([
                                        'form_params'=>[
                                            'Input'=>'我会覆盖options中Input的这个参数值，格式需
要为json',
                                        ],
                                    ])
                                    ->debug(false) // 最后调用的会覆盖前者。
                                    ->request();
    // 查询执行结果
    $request3 = Fnf::v20190315()->getExecutionHistory();
    // 设置选项或参数并执行请求
    // 创建一个Flow
    $result3 = $request3->withFlowName('xxx') // API的参数。
                                    ->withExecutionName('xxx')
                                    ->debug(true) // 开启调试会输出详细信息。
                                    ->connectTimeout(10) // 连接超时会抛出异常。
                                    ->timeout(10) // 超时会抛出异常。
                                    ->request(); // 执行请求。
} catch (ClientException $exception) {
    echo $exception->getMessage().PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage().PHP_EOL;
    echo $exception->getErrorCode().PHP_EOL;
    echo $exception->getRequestId().PHP_EOL;
    echo $exception->getErrorMessage().PHP_EOL;
}
```

6.Python SDK使用示例

本文介绍使用Python SDK的详细流程，包括环境准备、SDK安装和快速使用三部分。

环境准备

1. 要使用阿里云Python SDK，您需要一个云账号以及一对 `AccessKey ID` 和 `AccessKey Secret` 。请在阿里云控制台中的AccessKey管理页面上创建和查看您的AccessKey，或者联系您的系统管理员。
2. 要使用阿里云SDK访问某个产品的API，您需要事先在阿里云控制台 中开通这个产品。

SDK获取和安装

建议您使用PIP安装SDK，具体操作如下所示：

1. 执行以下命令，安装阿里云SDK核心库：

```
pip install aliyun-python-sdk-core
```

2. 执行以下命令，安装Serverless工作流SDK：

```
pip install aliyun-python-sdk-fnf
```

快速使用

下文将以创建一个流程，发起一次执行并获取执行详情为例展示如何使用Python SDK调用Serverless工作流服务。

请求方式

```
# encoding: utf-8
import time
from aliyunsdkcore.client import AcsClient
from aliyunsdkcore.acs_exception.exceptions import ServerException
from aliyunsdkfnf.request.v20190315 import CreateFlowRequest
from aliyunsdkfnf.request.v20190315 import StartExecutionRequest
from aliyunsdkfnf.request.v20190315 import DescribeExecutionRequest
from aliyunsdkfnf.request.v20190315 import GetExecutionHistoryRequest
flow_definition_type = "FDL"
flow_name = "xxx"
flow_definition = "xxx"
flow_description = "some descriptions"
role_arn = "acs:ram::${Your_Account_ID}:${Your_Role}"
execution_name = "xxx"
def create_flow(fnf_cli):
    request = CreateFlowRequest.CreateFlowRequest()
    request.set_Type(flow_definition_type)
    request.set_Description(flow_description)
    request.set_Definition(flow_definition)
    request.set_RoleArn(role_arn)
    request.set_Name(flow_name)
    return fnf_cli.do_action_with_exception(request)
def start_execution(fnf_cli):
    request = StartExecutionRequest.StartExecutionRequest()
    request.set_FlowName(flow_name)
    request.set_ExecutionName(execution_name)
    return fnf_cli.do_action_with_exception(request)
def describe_execution(fnf_cli):
    request = DescribeExecutionRequest.DescribeExecutionRequest()
    request.set_FlowName(flow_name)
    request.set_ExecutionName(execution_name)
    return fnf_cli.do_action_with_exception(request)
def get_execution_history(fnf_cli):
    request = GetExecutionHistoryRequest.GetExecutionHistoryRequest()
    request.set_FlowName(flow_name)
    request.set_ExecutionName(execution_name)
    return fnf_cli.do_action_with_exception(request)
```

创建客户端并利用上述函数发起一系列调用。

> **说明** 如果您需要不加改造进行调试的话，请将下述函数与上述请求方式代码块置于同一个文件中，避免在import时报错。

```
def main():
    # 创建AcsClient实例
    client = AcsClient(
        "<your-access-key-id>",
        "<your-access-key-secret>",
        "<your-region-id>"
    )
    try:
        create_resp = create_flow(client)
        print(create_resp)
        start_resp = start_execution(client)
        print(start_resp)
        time.sleep(1)
        desc_resp = describe_execution(client)
        print(desc_resp)
        get_resp = get_execution_history(client)
        print(get_resp)
    except ServerException as e:
        print(e.get_request_id())
if __name__ == '__main__':
    main()
```