

目 录

目录	1
利用云函数和飞书开发文件消息推送机器人	2
使用场景	2
操作步骤	2
1. 创建云函数	2
代码示例	2

利用云函数和飞书开发文件消息推送机器人

使用场景

本文介绍如何使用金山云云函数（KCF）实现向对象存储（KS3）上传文件后，自动向飞书Bot推送消息的功能。其中KS3存储用户上传的文件，KCF收到从KS3发送的文件上传事件后，触发执行自动推送文件消息内容到飞书机器人。

使用云函数、对象存储、推送飞书机器人消息的运行原理如下：

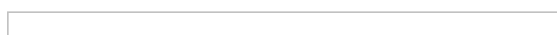


操作步骤

1. 创建云函数

a. 创建函数，通过函数环境配置和网络配置使函数可以访问代理服务器。

- 配置环境配置>环境变量。



Key:WEBHOOK_ID, value: 飞书群消息机器人标识

飞书群消息机器人标识获取方式：飞书客户端点击“+” > “创建群组”，群组中点击“设置” > “群机器人” > “Webhook地址Hook后接ID值”

- 开启网络配置>允许函数访问公网。



b. 创建触发器，选择指定的Bucket，设置触发规则为文件上传



c. 在触发器配置的Bucket中上传一个文件

d. 验证：查看飞书群消息机器人接收到的消息

代码示例

示例代码

```
package com.ksyun;
import io.javalin.Javalin;
import com.fasterxml.jackson.databind.ObjectMapper;

import io.cloudevents.CloudEvent;
import io.cloudevents.core.data.PojoCloudEventData;
import io.cloudevents.http.HttpMessageFactory;
import io.cloudevents.jackson.PojoCloudEventDataMapper;
import static io.cloudevents.core.CloudEventUtils.mapData;
import lombok.extern.slf4j.Slf4j;
import com.ksyun.KS3EventDataModel.Ks3CloudEventData;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.Proxy;

import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;

@Slf4j
public class App {
    public static void main(String[] args) {
        System.out.println("Hello KCF!");
        Javalin app = Javalin.create(
            config -> {
                config.requestLogger((ctx, ms) -> {
                    log.info(ctx.body());
                });
            })
            .start(8080);
    }
}
```

```

app.get("/", ctx -> ctx.result("Hello KingSoft Function"));

// Health check entry.
app.get("/health", ctx -> {
    ctx.result("Up");
});

// KCF function entry.
app.post("/event-invoke", ctx -> {
    log.info("starting process...");
    CloudEvent ce = HttpMessageFactory.createReader(ctx.headerMap(), ctx.body().getBytes()).toEvent();
    ObjectMapper objMapper = new ObjectMapper();
    PojoCloudEventData<Ks3CloudEventData> cloudEventData = mapData(ce,
        PojoCloudEventDataMapper.from(objMapper, Ks3CloudEventData.class));
    Ks3CloudEventData ks3EventData = cloudEventData.getValue();
    doProcess(ks3EventData);
    log.info("done processing...");
    ctx.result("ok");
});

private static void doProcess(Ks3CloudEventData ks3EventData) {
    // Show how to get necessary data from KS3 event data.
    // Add your own process in this function.
    if (ks3EventData != null) {
        log.info("KCF.DEMO: {}", ks3EventData.toString());
        log.info("KCF.DEMO: {}", ks3EventData.getKs3().toString());
        log.info("KCF.DEMO: {}", ks3EventData.getKs3().getObject().toString());
        String msg = "上传文件[%s]到[%s]桶中";
        SendMessageToFeiShu(String.format(msg, ks3EventData.getKs3().getObject().getKey(), ks3EventData.getKs3().getBucket().getKeyName()));
    } else {
        log.error("KCF.DEMO: {}", "Event data disrupted!");
    }
}

public static void SendMessageToFeiShu(String msg) {
    // Get forward proxy IP address and port from ENV.
    final String forwardProxyIp = System.getenv("PROXY_IP");
    // Get url of FeiShu Bot ID from ENV
    final int forwardProxyPort = Integer.parseInt(System.getenv("PROXY_PORT"));
    final String WebhookBotId = System.getenv("WEBHOOK_ID");
    // PLEASE Confirm 'forwardProxyIp' with CORRESPOND IP ADDRESS of PROXY Server which in the same VPN with KCF instance!!!
    final Proxy vpcProxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(forwardProxyIp, forwardProxyPort));
    final MediaType JSON = MediaType.get("application/json; charset=utf-8");
    final OkHttpClient client = new OkHttpClient().newBuilder().build();
    // FeiShu notifyURL
    String notifyURL = String.format("https://open.feishu.cn/open-apis/bot/v2/hook/%s", WebhookBotId);
    String postBody = "{\"msg_type\":\"text\",\"content\":{\"text\":\"%s\"}}";
    RequestBody body = RequestBody.create(String.format(postBody, msg), JSON);
    Request request = new Request.Builder()
        .url(notifyURL)
        .post(body)
        .build();
    try (Response response = client.newCall(request).execute()) {
        log.info("KCF.DEMO: {}", "Sent message to FeiShu success.");
    } catch (IOException ex) {
        log.error("KCF.DEMO: {}", "Error while send message to FeiShu!");
        log.error("KCF.DEMO: {}", ex.toString());
    }
}
}

```

Pom文件

```

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.ksyun</groupId>
    <artifactId>kcf-demo</artifactId>
    <version>1.0-SNAPSHOT</version>

    <name>kcf-demo</name>
    <!-- FIXME change it to the project's website -->
    <url>http://www.example.com</url>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>1.8</maven.compiler.source>
    </properties>

```

```

    <maven.compiler.target>1.8</maven.compiler.target>
</properties>

<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.11</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>io.javalin</groupId>
    <artifactId>javalin</artifactId>
    <version>4.5.0</version>
  </dependency>
  <dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>2.0.0-alpha6</version>
  </dependency>
  <dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-simple</artifactId>
    <version>2.0.0-alpha6</version>
  </dependency>
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.24</version>
  </dependency>
  <dependency>
    <groupId>io.cloudevents</groupId>
    <artifactId>cloudevents-core</artifactId>
    <version>2.3.0</version>
  </dependency>
  <!-- To use the json format and the cloudevent data mapper -->
  <dependency>
    <groupId>io.cloudevents</groupId>
    <artifactId>cloudevents-json-jackson</artifactId>
    <version>2.3.0</version>
  </dependency>
  <dependency>
    <groupId>io.cloudevents</groupId>
    <artifactId>cloudevents-http-basic</artifactId>
    <version>2.3.0</version>
  </dependency>
  <dependency>
    <groupId>com.squareup.okhttp3</groupId>
    <artifactId>okhttp</artifactId>
    <version>4.10.0</version>
  </dependency>
</dependencies>

<build>
  <pluginManagement>
    <!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->
    <plugins>
      <!-- clean lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean_Lifecycle -->
      <plugin>
        <artifactId>maven-clean-plugin</artifactId>
        <version>3.1.0</version>
      </plugin>
      <!-- default lifecycle, jar packaging: see https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin_bindings_for_jar_packaging -->
      <plugin>
        <artifactId>maven-resources-plugin</artifactId>
        <version>3.0.2</version>
      </plugin>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.0</version>
      </plugin>
      <plugin>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>2.22.1</version>
      </plugin>
      <plugin>
        <artifactId>maven-jar-plugin</artifactId>
        <version>3.0.2</version>
      </plugin>
      <plugin>
        <artifactId>maven-install-plugin</artifactId>
        <version>2.5.2</version>
      </plugin>
    </plugins>
  </pluginManagement>
</build>

```

```
<plugin>
  <artifactId>maven-deploy-plugin</artifactId>
  <version>2.8.2</version>
</plugin>
<!-- site lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#site_Lifecycle -->
<plugin>
  <artifactId>maven-site-plugin</artifactId>
  <version>3.7.1</version>
</plugin>
<plugin>
  <artifactId>maven-project-info-reports-plugin</artifactId>
  <version>3.0.0</version>
</plugin>
</plugins>
</pluginManagement>
<plugins>
  <plugin>
    <artifactId>maven-assembly-plugin</artifactId>
    <version>3.3.0</version>
    <configuration>
      <descriptorRefs>
        <descriptorRef>jar-with-dependencies</descriptorRef>
      </descriptorRefs>
      <appendAssemblyId>false</appendAssemblyId>
      <archive>
        <manifest>
          <mainClass>com.ksyun.App</mainClass>
        </manifest>
      </archive>
    </configuration>
    <executions>
      <execution>
        <id>make-assembly</id> <!-- this is used for inheritance merges -->
        <phase>package</phase> <!-- bind to the packaging phase -->
        <goals>
          <goal>single</goal>
        </goals>
      </execution>
    </executions>
  </plugin>
</plugins>
</build>
</project>
```