HDFS每日一练1 API读取文件

题目

在右侧代码编辑区中编写代码实现如下功能：

- 使用 `FSDataInputStream` 获取 `HDFS` 的 `/user/hadoop/` 目录下的 `task.txt` 的文件内容，并输出，其中 `uri` 为 `hdfs://localhost:9000/user/hadoop/task.txt`。

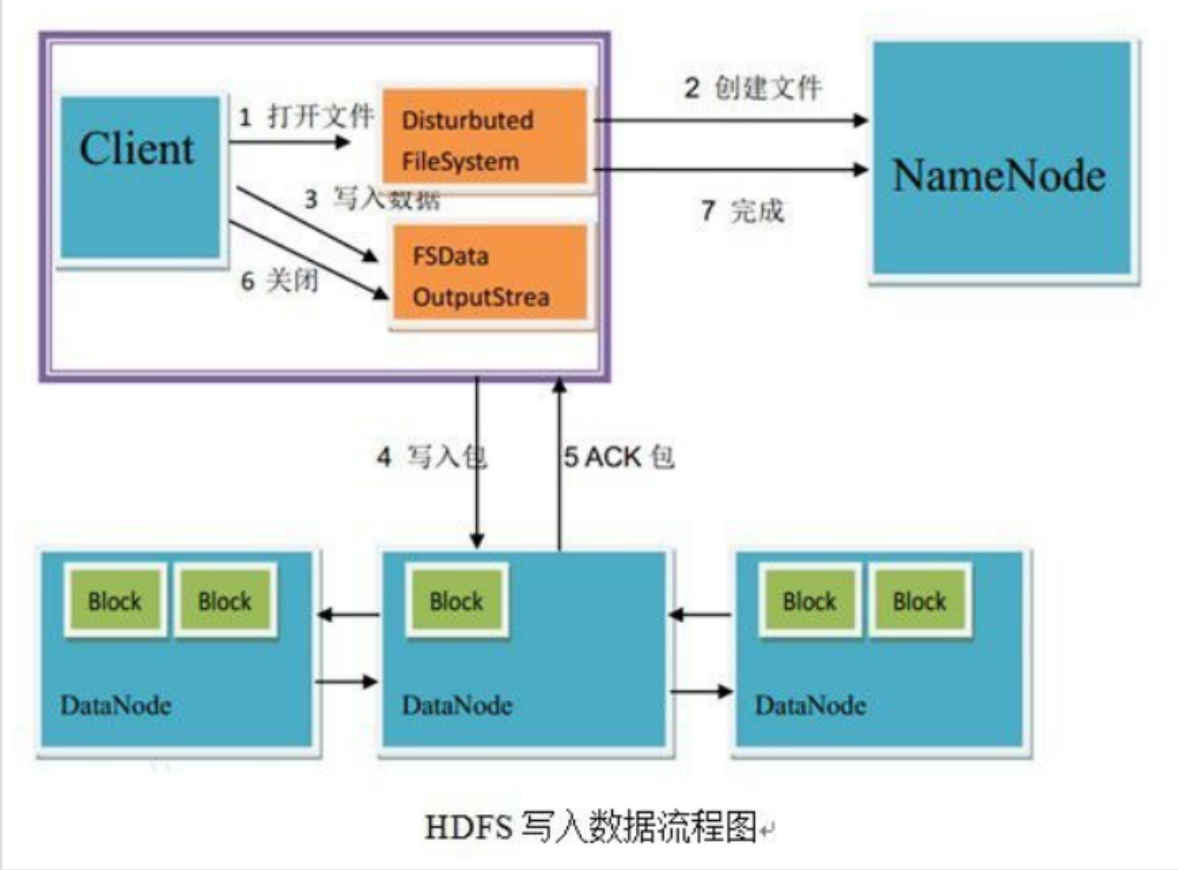
测试说明：

点击评测，平台会通过脚本创建 `/user/hadoop/task.txt` 文件并添加相应内容，无需你自己创建，开启 `hadoop`，编写代码点击评测即可。因为 `Hadoop` 环境非常消耗资源，所以你如果一段时间不在线，后台会销毁你的镜像，之前的数据会丢失（你的代码不会丢失），这个时候需要你重新启动 `Hadoop`。

预期输出：`WARN [main] - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable` 怕什么真理无穷，进一寸有一寸的欢喜。

教程

在本关和之后的关卡中，我们要深入探索 `Hadoop` 的 `FileSystem` 类，它是与 `Hadoop` 的某一文件系统进行交互的 `API`。



HDFS 写入数据流程图

为了完成本关任务，你需要学习并掌握：1. `FileSystem` 对象的使用，2. `FSDataInputSteam` 对象的使用。

FileSyetem对象

> 背景：要从Hadoop文件系统中读取文件，最简单的办法是使用java.net.URL对象打开数据流，从中获取数据。不过这种方法一般要使用FsUrlStreamHandlerFactory实例调用setURLStreamHandlerFactory()方法。不过每个Java虚拟机只能调用一次这个方法，所以如果其他第三方程序声明了这个对象，那我们将无法使用了。 因为有时候我们不能在程序中设置URLStreamHandlerFactory实例，这个时候咱们就可以使用FileSystem API来打开一个输入流，进而对HDFS进行操作。

FileSystem是一个通用的文件系统API， FileSystem实例有下列几个静态工厂方法用来构造对象。

```
public static FileSystem get(Configuration conf)throws IOException
public static FileSystem get(URI uri,Configuration conf)throws IOException
public static FileSystem get(URI uri,Configuration conf,String user)throws
IOException
```

Configuration对象封装了客户端或服务器的配置，通过设置配置文件读取类路径来实现（如：/etc/hadoop/core-site.xml）。

- 第一个方法返回的默认文件系统是在core-site.xml中指定的，如果没有指定，就使用默认的文件系统。
- 第二个方法使用给定的URI方案和权限来确定要使用的文件系统，如果给定URI中没有指定方案，则返回默认文件系统，
- 第三个方法作为给定用户来返回文件系统，这个在安全方面来说非常重要

FSDataInputStream对象

实际上，FileSystem对象中的open()方法返回的就是FSDataInputStream对象，而不是标准的java.io类对象。这个类是继承了java.io.DataInputStream的一个特殊类，并支持随机访问，由此可以从流的任意位置读取数据。

在有了FileSystem实例之后，我们调用open()函数来获取文件的输入流。

```
public FSDataInputStream open(Path p)throws IOException
public abst\fract FSDataInputStream open(Path f,int bufferSize)throws
IOException
```

第一个方法使用默认的缓冲区大小为4KB。

了解了这些，就可以看看实例。

实例

接下来我们通过一个实例来学习它的用法。

首先我们在本地创建一个文件，然后上传到HDFS以供测试。

```
# touch test.txt
# vim test.txt
读作池塘如虎踞，绿荫属下养精神
春来我不先开口，哪个虫儿敢作声

# hadoop fs -put test.txt /user/tmp
```

接下来，我们使用FileSystem，查看咱们刚刚上传的文件。 代码如下：

```
public sattic void main(String[] args){
    URI uri = URI.create("hdfs://localhost:9000/user/tmp/test.txt");
    Configuration config = new Configuration();
    FileSystem fs = FileSystem.get(uri, config);
    InputStream in = null;
    try {
        in = fs.open(new Path(uri));
        IOUtils.copyBytes(in, System.out, 2048, false);
    } catch (Exception e) {
        IOUtils.closeStream(in);
    }
}
```

答案

```
package step2;

import java.io.IOException;
import java.io.InputStream;
import java.net.URI;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IOUtils;


public class FileSystemCat {

    public static void main(String[] args) throws IOException {
        //请在Begin-End之间添加你的代码，完成任务要求。
        /********** Begin **********/
        URI uri = URI.create("hdfs://localhost:9000/user/hadoop/task.txt");
        Configuration conf = new Configuration();
        FileSystem fs = FileSystem.get(uri, conf);
        InputStream in = null;
        try{
            in = fs.open(new Path(uri));
            IOUtils.copyBytes(in, System.out, 2048, false);
        }catch(Exception e){
            IOUtils.closeStream(in);
        }



        /********** End **********/
    }
}
```