

Jenkins Pipeline 实现 http 请求并解析响应

2020-11-23

JENKINS

阅读 3 分钟

阅读量 1994



Jenkins Pipeline 中为了要从某些接口中获得响应，并做解析，需要一系列语法组合。本文做个简单介绍。

发出 HTTP 请求

Jenkins 支持 发送 HTTP 请求。包含两个基本步骤：安装插件和编写pipeline

安装 HTTP Request 插件

如果不安装插件而直接调用 `httpRequest` 命令，可能会得到下面的报错：


```
java.lang.NoSuchMethodError: No such DSL method 'httpRequest' found among steps
```

在Jenkins 后台的插件管理页面，搜索 HTTP Request 关键字，可以搜索到该插件，点击安装，重启 Jenkins以使插件生效。

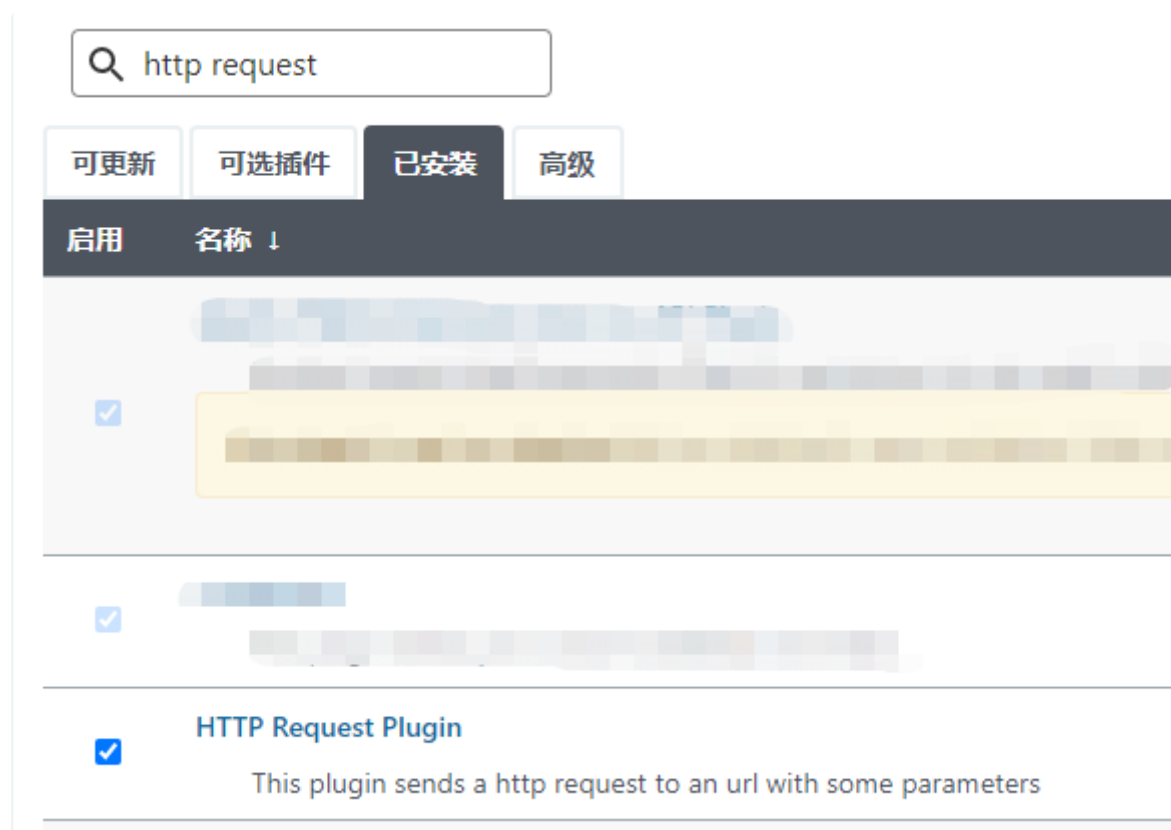


Manage Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

 可用更新

安装后的效果如下：



编写 Pipeline

httpRequest命令的 [官方文档](#)

安装好 HTTP Request 插件 后，我们可以编写 Pipeline ，我们选择使用更为规范的 Declarative Pipeline 编写我们的执行语句。

发送请求的语句使用 httpRequest ，基本的写法是：

```
httpRequest "http://your.site"
```

返回一个 response 对象，包含响应的状态码和主体：

```
1 | def response = httpRequest "http://your.site"
2 | println('Status: '+response.status)
3 | println('Response: '+response.content)
```

默认参数是请求的URL，其他参数加在后方：

```
httpRequest url:"http://your.site", httpMode: "GET", ignoreSslErrors: true
```

再有 Post 的例子（application/x-www-form-urlencoded 类型）：

上面的语句里，参数太多，放在一行里比较难于查看。为了方便查看，把命令改写成多行，如下所示：

```
1 | def response = httpRequest \
2 |     contentType: 'APPLICATION_FORM',
3 |     httpMode: "POST",
4 |     requestBody: "a=1&b=2",
5 |     url: "http://your.site"
```

注意这里 httpRequest 后面用了 \ 用来表示行间连续（continuation）

另一个Post 的例子（application/json）：

```
1 | def requestBody = ["k1":"v1","k2":"v2"]
2 |
3 | def response = httpRequest \
4 |     httpMode: "POST",
5 |     ignoreSslErrors: true,
6 |     contentType: 'APPLICATION_JSON',
7 |     requestBody: groovy.json.JsonOutput.toJson(requestBody),
8 |     url: "http://your.site"
9 |
10 | println response.content
```

解析 HTTP 响应

上一节 中通过 `HttpRequest` 拿到响应后，如果响应是 `Json` 格式的，我们还可以进一步解析 `Json` 响应，并提取其中的部分内容。解析 `Json`使用内置函数 `readJSON`，基本用法为：

```
readJSON text: response.content
```

注意这里和上节的 `HttpRequest` 一样，都需要有命名参数（named argument），即 `text:` 不可省略。

假设我们收到下面这样的响应：

```
1 {  
2   "code": 200,  
3   "message": "OK",  
4   "data": ["apple", "banana", "cherry"]  
5 }
```

就可以用下面的语句解析：

```
1 def props = readJSON text: response.content  
2 def code = props['code']  
3 def data = props['data']  
4 def data0 = props['data'][0]  
5 def data1 = props['data'][1]
```

在 HTTP 响应中搜索

如果响应中有列表，我们希望从列表中找出第一个满足需要的项，除了普通的遍历方法外，还可以用`find`方法，比如要在 上一节 中的响应结果中找到 字母 `b` 开头的项，可以用下面的方法：

```
1 def props = readJSON text: response.content  
2 def data = props['data']  
3  
4 data.find{  
5   if(it.startsWith("b")){  
6     return true  
7   }  
8   return false  
9 }
```

完整的例子

一个完整的 请求httprequest + 解析Json结果的 pipeline 的例子如下:

```
1 pipeline{
2   agent {label 'slave'}
3
4   stages{
5     stage('Example'){
6       steps{
7         script{
8           def body = [k1]("v1","k2":"v2")
9
10          def response = httpRequest \
11              httpMode: 'POST' ,
12              ignoreSslErrors: true,
13              contentType: 'APPLICATION_JSON',
14              requestBody: groovy.json.JsonOutput.toJson(body),
15              url: "http://your.site"
16
17          def props = readJSON text: response.content
18          def data = props['data']
19
20          data.find{
21            if(it.startsWith("b")){
22              return true
23            }
24            return false
25          }
26        }
27      }
28    }
29  }
30 }
```

HTTP Request 插件的其他参数

HTTP Request 插件的 [介绍页面](#) 列有详细的参数列表，下面简单介绍部分参数的用法:

- consoleLogResponseBody , 在 Jenkins 控制台打印出响应的body,取值: true/false, 等效于:

```
println "Response: \n" +response.content
```

- customHeaders , 请求的自定义header , 用 Groovy 的数组表示

```
customHeaders: [ [name: "k1", value: "v1"], [name: "k2", value: "v2"] ],
```

- `outputFile` , 将响应的body写入文件, 后面的参数是文件的路径, 可以是绝对路径或者相对路径。如果写相对路径 (或者单独一个文件名) , 那么起始点是此次构建的workspace目录。

```
outputFile: "output.txt"
```

- `quiet` , 不向控制台打印任何内容。默认情况下, 控制台上会出现以下内容:

```
1 | HttpMethod: POST
2 | URL: https://XXXX
3 | Content-Type: application/json
4 | Sending request to url: https://XXXX
5 | Response Code: HTTP/1.1 302 Found
```

当 `quiet` 设置为 `true` 后, 以上这些内容都不出现在控制台上, 这个选项优先于上面的 `consoleLogResponseBody`

- `validResponseCodes` , 是否检验返回状态码, 如果不在此范围内, 则请求失败。采取类似下面的格式:

```
validResponseCodes: "200:210,300:302,400:403"
```

这里冒号表示范围, 多个范围之间用逗号隔开。

(全文完)

JENKINS