Python的JIRA模块用于和JIRA进行交流，内在的实现方式是通过JIRA的RESTAPI，以及Python的Requests模块进行请求。但是JIRA模块已经进行了封装，我们无需直接去请求JIRA的REST API地址。

1.首先需要建立JIRA实例：

jac = JIRA('https://jira.atlassian.com')

在建立实例的同时进行验证（有OAuth, HTTP BASIC，Cookie Based Authentication等）

authed\_jira = JIRA(basic\_auth=('username', 'password'))

2. 获取一个Issue，获取的Issue将会是一个Issue对象，使用这个对象的方法和域，可以获取Issue的值以及对Issue进行更改

issue = jira.issue("NSPD-190187") // 获取Issue对象

issue.fields.description //获得Issue域中description的值

issue.update(summary='new summary', description='A new summary was added')//更改issue的summary以及description

issue.update(assignee={'name': 'new\_user'}) # reassigning in update requires issue edit permission

3. 新建一个Issue

issue\_dict = {

'project': {'id': 123},

'summary': 'New issue from jira-python',

'description': 'Look into this one',

'issuetype': {'name': 'Bug'},

}

new\_issue = jira.create\_issue(fields=issue\_dict)

4. 查找Issue, 条件中填入JQL查询语言（<https://confluence.atlassian.com/jiracoreserver073/advanced-searching-861257209.html>）， 将返回list of issue objects

Matching\_issues = jira.search\_issues('assignee = currentUser() and due < endOfWeek() order by priority desc', maxResults=5)

5. JIRA模块了，除了可以查找出ISSUE作为obj以外，对Project, Watchers等都是进行查找返回一个object对象。

6.Jira在一个ISSUE中有两种fields，一种是系统自带的例如项目名，summary, 状态等这些叫做build-in fields. 还有一些我们可以根据实际情况自行加入的，叫做custom\_fields.

7. 如果我们在创建JIRA ISSUE时候，不太确定应该给JIRA 传递怎样的字典参数的时候。可以使用JIRA REST API:

[http://xxxx/rest/api/2/issue/createmeta?projectKeys="+project\_name+"&issuetype=PTS&expand=projects.issuetypes.fields](http://xxxx/rest/api/2/issue/createmeta?projectKeys=%22+project_name+%22&issuetype=PTS&expand=projects.issuetypes.fields)"

去查看需要创建的域名中是否包含value,name等。一般分为两种域：

如果是系统自带的域：则使用jira\_issue\_obj.update(fields={"系统自带域名": form.cleaned\_data[域名]})

如果是custom\_field:则使用'customfield\_12128': {'value': form.cleaned\_data['impact']},

8.