ReadMe.md

仓库信息

仓库详细信息在 repos-info/{name} 目录下，文件格式为 {name}.json

- 例： 12306详细信息存放在 repos-info/12306/12306.json
- （后文均以12306为例）

仓库一些重要信息存放在 repos-info/{name} 目录下 文件格式为 {name}-part.json

- 例： 12306重要信息存放在 repos-info/12306/12306-part.json

项目重要信息中：

- "html_url" 为项目在github上的主页
- "decription" 为项目描述
- "url" 可获得项目详细信息，即存放在12306.json中的内容
- "languages_url" 可获得项目所使用的语言的信息，结果为：

```
{
  "Python": 190210,
  "Shell": 961,
  "Dockerfile": 778
}
```

- "issues_url" 可获得包含项目issue信息的json格式信息，其中的"html_url"为该issue主页。根据12306-part.json中的issues_url: https://api.github.com/repos/testerSunshine/12306/issues{/number} ,我们访问 https://api.github.com/repos/testerSunshine/12306/issues/1 ， 获得的信息如下(仅列出部分):

```
{
    "url": "https://api.github.com/repos/testerSunshine/12306/issues/1",
    ...
    ...
    "html_url": "https://github.com/testerSunshine/12306/issues/1",
    ...
    ...
    "performed_via_github_app": null
}
```

通过其中的"html_url"可访问该issue主页

- "created_at" 为项目创建日期 结果为"2017-05-17T12:23:40Z"
- "pushed_at" 为最新push的日期 结果为"2020-09-08T03:38:39Z"
- "updated_at" 为项目信息更新日期，结果为`"2020-09-19T11:38:09Z"
- "git_url" 用于克隆仓库(ssh)
- "clone_url"用于克隆仓库(https)
- "ssh_url" 通过SSH提供 Git 仓库的访问权限
- "size" 为项目大小(KB)
- "stargazers_count" 为星数
- "watchers_count" 为watchers数量
- "forks_count" 为复制仓库的数量
- "open_issues_count" 为open issue数量
- "license" 为license信息

代码

- get_repo_info.py 用于按星级降序从github上收集主语言为python的项目，按照每页100个项目，共10页来收集。收集到的项目信息说明见上文。收集到的项目名字及项目个数以json格式存放在repo_name.json中。
- part_repo_info.py 用于收集项目信息中的部分重要项，需要收集的重要项条目存放于part.json中。若要改变需要收集的重要项，请修改part.json文件
- get_repo.py 用于从github上clone仓库。根据存放在repos-info/{name}/{name}.json中的"git_url"来复制仓库。复制路径为 repos/{name} 若要运行此python文件，请先运行get_repo_info.py

get_repo_info.py

```
//get_repo_info.py
import requests
import json
import os

#to update code
```

```
URL = 'https://api.github.com/search/repositories?q=language:python&sort=s
#URL = 'https://api.github.com/search/repositories?l=C%23&q=dotnet&sort=st
isfile = os.path.isfile('repo_name.json')

if not isfile:
    f = open('repo_name.json','w+')
    json.dump({'repo_num':0},f,indent=4)
    f.close()

f = open('repo_name.json','r')
repo_list = json.load(f)
repo_num = repo_list['repo_num']

def mkdir(path):
    folder = os.path.exists(path)
    if not folder:
        os.makedirs(path)
        print ("new folder: %s" % path)

for i in range(1,11):
    try:
        print('now getting page: %s' % str(i))
        r = requests.get(URL+str(i))#to update code
        result = json.loads(r.content)#to update code
       #result = json.loads(r.content.decode('UTF-8'))   in python3.5
        for repo in result['items']:
            try:
                name = repo['name']
                if name not in repo_list.values():
                    repo_num += 1
                    repo_list['repo_'+str(repo_num)]=name
                    mkdir('repos-info/'+name)
                    with open('repos-info/'+name+'/'+name+'.json','w+') as
                        json.dump(repo,f,indent=4)
                        f.close()
            except Exception as e:
                print(e)
    except Exception as e:
        print(e)

repo_list['repo_num'] = repo_num
print('repo_num:'+str(repo_num))
f = open('repo_name.json','w+')
```

```
json.dump(repo_list,f,indent=4)
f.close()
```

part_repo_info

```
//part_repo_info
import json

f = open('part.json','r')
config = json.load(f)
f.close()
items = config['items']
f = open('repo_name.json','r')
all_repo = json.load(f)
num = all_repo['repo_num']
f.close()

for i in range(0,num):
    name = all_repo['repo_'+str(i+1)]
    path = 'repos-info/'+name+'/'
    f = open(path+name+'.json','r')
    repo_info = json.load(f)
    f.close()
    part_info = {}
    for item in items:
        part_info[item] = repo_info[item]
    f = open(path+name+'-part.json','w+')
    json.dump(part_info,f,indent=4)
    f.close()
```

get_repo

```
import json
import os
os.environ["GIT_PYTHON_REFRESH"] = "quiet"
import git
info_path = 'repos-info/'
repo_path = 'repos/'
```

```
f = open('repo_name.json','r')
all_repo = json.load(f)
f.close()
num = all_repo['repo_num']
for i in range(0,num):
    try:
        name = all_repo['repo_'+str(i+1)]
        print('now getting repo:'+name)
        path = info_path+name+'/'
        f = open(path+name+'.json','r')
        repo_info = json.load(f)
        f.close()
        git.Repo.clone_from(url=repo_info['git_url'],to_path=repo_path+nam
    except Exception as e:
        print(e)
```

打造App-GitHub 开放API大总结

[参考资料]：https://juejin.im/post/6844904024307662855

打造App-GitHub 开放API大总结