

基于Pytest与Postman的数据自动采集接口自动化测试实践	01
Jenkins系列之自动化部署(1)	13
Jenkins系列之Selenium-UI自动化测试(2)	32
Jenkins系列之Jmeter-API自动化测试(3)	56
机器学习之KNN算法,好算法有迹可循!	76
全流程自动化测试·业务规则标准化及资产库建设探索	81
JMeter循环读取CSV文件实现接口批量测试	84
新人应该如何进入测试领域?	90



◆ 作者: kayie

基于 Pytest 与 Postman 的数据自 动采集接口自动化测试实践

一、前言

本文跟大家介绍的是基于 Python 测试框架 Pytest 与 Postman 数据自动采集的接口 自动化测试实践方案,首先了解下为什么是基于 Pytest 框架而不是使用 Unitest? 网上的 对比资料很多,概括起来就是 Pytest 相较于 unittest 代码更加的简洁和灵活,最为跳跃 的一点就是 fixture 机制,并且 Pytest 有很多的第三方插件可以扩展和继承,大家可以 深入去查一查。

第二个了解的是为什么要做数据自动采集,做数据驱动很多教程推荐的是将数据写在 excel 中,然后通过程序去读取。但我更推荐的是通过软件自动采集所需数据,这样可以大大节省手动在 excel 录入数据的时间。推荐使用 Postman 采集数据,它是一款做开发人手必备的接口调试工具,几乎能发送所有类型的 HTPP 请求,在打开代理模式以后能够自动抓取 pc 端浏览器或者 APP 端请求的接口数据,存储并导出后为 json 格式的数据源,简单方便。

二、自动化框架流程

整个代码框架除了数据来源从 Postman 采集,其他所有封装的公共库以及具体的测 试用例均使用代码编辑器如 pycharm 编写 python 代码,其中需要用到的 pytest、 requests、allure 等库可以直接使用 pip 命令安装。从数据采集到完成自动化测试并输出 测试报告的流程如下图:





《51 测试天地》五十六(上)

www.51testing.com



1) 编写公共库:如果某个函数需要被多次引用,那么就可以封装成公共函数,这些封装的函数主要有:断言、全局常量、log日志、发送请求、用户 session、加密等。 跟业务无关,可以先封装好。



2)编写配置文件: 配置文件内容如环境参数、文件存储路径、版本号,以及配置 文件的读写封装等,也是编写测试用例之前必须封装好的内容。







3)数据自动采集:如下图所示,打开代理模式,设置代理端口如:5555,设置抓 包的存储路径,比如一个功能流程就可以单独存一个集合,最后设置过滤地址为本系 统,以免抓到与系统无关的接口。

0.0			Postman				
🕂 New 🔽 Import Runner			🖬 cms 🔻 🚑 Invite	1	\$ 1	•	٢
Q Filter	\bigcirc	POST R	Capture requests and cookies		est		
History Collections	APIs BETA		Requests Cookies BETA	owser with Postman's			
+ New Collection	Trash	POST	built-in proxy. Learn more about capturi	ng requests.		Send	
cms_init ★ 38 requests		147 148 149 150	Source Proxy Interceptor	打开代理模式			
Cms_base 315 requests		151 - 152 153 154	Capture Requests				
Cms_special 16 requests		155 156 157 158	Port	设置代理窗口			
		Body (5555		ms Size	: 2.59 KB	Sav
		Pretty	Save Requests to	圳句方梯败汉			
		1	History	加巴汗順西江			
		2	Hide additional filters	-			
		4	URL Contains	过滤系统地址	/v1/Con	tract.p	hp(343
				Elto	troller	81\\x94 /v1/Con	tract.
			Regular expressions are supported	JULE	ray, Ar reate()	ray)\n# \n#3 <u>/v</u>	2 [int ar/www
0 9 2			URL Does not contain	软件	uild	Brow	/se

4)数据导出 json 文件存储:选中某个集合后,右键选择 export 导出,选择第一个 json 数据格式。导出的数据源放入项目对应的目录 Params->json 下









5) 编写数据读取函数:编写 jsonparam.py 函数,解析 Postman 格式的 json 数据。因为 Postman 有些数据不是我们需要的 有些需要拼接,所以需要单独封装一个转换格式的函数,方便测试用例数据读取使用。



6) 编写测试用例:在 Testcase 下编写测试用例的前置和后置参数文件 conftest.py 文件(此文件名为是固定的,不能写别的)。





• • • • • • • • • • • • • • • • • • • •		pyautoTest [~/PycharmProjects/pyautoTest]/TestCase/conftest.py	
/autoTest 👌 🛅 TestCase 👌 🐞 confte	st.py	🥐 run	🔻 🕨 🎽 🔳 🛛 Git: 🖌 -
Project 👻	+	- 🚜 Assert.py × 🐞 Session.py × 🐉 test_init.py × 🐞 demo.py × 🙀 config.in	ni 🛛 🕌 jsonparams.py 🗡 🍵
 ♣ Log.py ♣ Log.py ♣ Session.py ♣ Session.py ♣ Shell.py ♥ Config.py ♥ Log ֎ config.py ♥ Log ֎ rr.log ֎ log.log ♥ Params ♥ Param ♥ Param ♥ B Param ♥ D Son ▶ Manual Yaml 	U	<pre>13 # @pytest.allure.step # 用于将一些通用的函数作为测试步骤输出到报告, 调 4 # allure.environment(environment=env) #用于定义environment 16 UPP 17 18 O'import allure 19 import pytest 20 from Conf.Config import Config 21 from Conf.Config import Config 22 from Common import Consts 23 24 25 @pytest.fixture() 26 Odef action():</pre>	月此函数的地方会向报告中 单
 jsonparams.py jsontools.py params.py tools.py tools.py Report Report In html xml TestCase Pytest_cache 		<pre>27 # 定义环境 28 env = Consts.API_ENVIRONMENT_DEBUG 29 # 定义报告中environment 30 conf = Config() 31 \$ host = conf.host_debug 32 tester = conf.tester_debug 33 allure.environment(environment=env) 34 allure.environment(hostname=host)</pre>	
<pre></pre>		35 allure.environment(tester=tester) 36 return env ① IDE and Plu	ugin Updates

编写具体的测试用例,测试用例均要使用 test 开头或结尾(否则框架无法识别)



7)运行测试用例:如果需要运行 Testcase 下所有测试用例,可以在根目录建 run.py 文件,若只需要单独运行某个功能的测试用例,也可以在具体的测试文件的 main 函数 里面使用 Pytest.main("test_xxx.py")命令运行。







Project 👻	⊕ ÷ ♥ −
🐞 Request.py	
💑 Session.py	16 Trom Common Import Log
💑 Shell.py	17 Trom Common Import Shell
🔻 🛅 Conf	18 from Conf import Config
i config.ini	19 pimport warnings
💑 Config.py	<pre>20 warnings.filterwarnings("ignore")</pre>
🔻 🛅 Log	21
err.log	22 ▶ 🖓 ifname == 'main':
ie log.log	23 # 初始化
🔻 🛅 Params	24 conf = Config()
🔻 🛅 Param	25 log = log ($\log f$)
🔻 🛅 Json	26 Log info()加強化型電文件 noth-1 t conf conf path
temp.json	20 = tog. mot warball shill ()
Yaml	2/ snett = snett.snett()
🐞 jsonparams.py	28. # 获取按告输出位置
isontools.py	<pre>29 xml_report_path = conf.xml_report_path</pre>
🐌 params.py	<pre>30 html_report_path = conf.html_report_path</pre>
💑 tools.py	31 # 定义测试集
Report	<pre>32 dir = os.path.split(os.path.abspath(file))[0]</pre>
Image: http://www.image.com/ http://www.i	<pre>33 test_case_path = dir + '/TestCase/test_init.py'</pre>
▶ 🖿 xml	34 args = ['-s', '-q', 'alluredir', xml_report_path]
TestCase	35 # args.append(test case path)
.pytest_cache	36 月 # 运行命令
conftest.py	37 putset main(args)
demo.py	an pycectime intergorante % a se % a % (well report path, html report path)
test_init.py	and a state generate was to was a (xint_report_path, thint_report_path)
venv	59 博为峰旗下
README.md	try: 51 rest Tro
in run.py	41 shell.invoke(cmd)
	47 向 except Exception: 软件测试网

8) 输出日志文件: 过程中使用 self.log.info('请求参数: %s' % data)打印日志信息,

作为记录和调试使用。

yautoTest $ angle$ 🖿 Log $ angle$ 🚆 err.	log								2	run 🔻	► ă		Git: 🖌
Project 👻	+ ÷ + -	· 👘 test_init.py	× 🛛 👸 Log.py 🗵	i err.log ×	🗑 log.log	× 🐞 run	.py $ imes$	👼 demo.py	×	conftest.p	y ×	🗐 confi	g.ini $ imes$
pyautoTest ~/Pycharn	mProjects/pyautoTest	Plugins supp	orting *.log files	found.							Inst	all plug	jins Ig
.pytest_cache		1 [6			Response	body m	sor !=	expected	msor	expecte	nd msa	is 1	1 mso
Common					Posponso	body m	 -	oxpocted	, 	ovpocto	d mea		2 mcg
💑 Assert.py					Incorporate	tid mark	sg :-	expected	,	expecte	u_msg	500	∠, ⊪sg
💑 Consts.py		3 [E			Jkesponse	time >	expe	cted_time	, exp	ected_ti	me is	. 500,	cime
💑 Log.py		4 [EI		11 14:23:37]Response -	body m	sg !=	expected	_msg,	expecte	d_msg	; 15 1	, msg 1
提 Request.py		5 [EI		11 14:30:46]Response	body m	sg !=	expected	_msg,	expecte	d_msg	; is 1	, msg i
💑 Session.py		6 [EF]Response	body m	sg !=	expected_	_msg,	expecte	d_msg	; is 1	, msg †
📥 Shell.py		7 [EI]Response	body m	sg !=	expected	_msg,	expecte	d_msg		, msg i
🔻 🛅 Conf		8 [EF]Response	body m	sg !=	expected	_msg,	expecte	d_msg		, msg i
🖆 config.ini		9 [EF]Response	body m	sg !=	expected	_msg,	expecte	d_msg		, msg t
Config.py		10 F ER	ROR 2019-11-	11 15:05:49 [°]	1Response	bodv m	sg !=	expected	msg.	expecte	d msg	is 1	. msg -
🗡 🖿 Log		11 [F		11 15:08:51	 lResponse	body m	sø !=	expected	msg.	expecte	d msg	is 1	. msø i
🗐 err.log)	12		11 15.52.56	Response	body m	-6 · eg l=	expected	, ,	evpecte	d mea		,
🗑 log.log						body m		expected		expecte			,
Params					JResponse	body m	sg :-	expected	_msg,	expecte	:a_msg		, msg i
🔻 🛅 Param		14 [EF	ROR 2019-11-	13 19:52:48	JResponse	body m	sg !=	expected_	_msg,	ехресте	∶a_msg	; 1S 1	, msg -
🔻 🖿 Json		15 [EF		13 19:52:54]Response	body m	sg !=	expected_	_msg,	expecte	ed_msg	; is 1	, msg -
🚯 temp.json		16 [EF		13 19:55:24]Response	body m	sg !=	expected	_msg,	expecte	d_msg	; is 1	, msg i
🕨 🖿 Yaml		17 [EF]Response	body m	sg !=	expected	_msg,	expecte	d_msg		, msg i
腾 jsonparams.py		18 [EF]Response	body m	sg !=	expected	_msg,	expecte	d_msg		, msg i
🐌 jsontools.py		19 []]Response	body m	sg !=	expected	_msg,	expecte	d_msg		, msg t
🐌 params.py		20											
📥 tools.py													
🔻 🛅 Report													
🕨 🖿 html													
🕨 🖿 xml													
🔻 🛅 TestCase													
.pytest_cache													
🐌 conftest.py													
💑 demo.py													

9)输出测试报告:最后可以集成 allure 插件,输出更加直观漂亮的测试报告。集成 过程网上也有很多教程,但是坑非常多,大家需要注意以下内容:

① 建议使用 Pytest 3.8.0 版本

命令: pip install Pytest

注意:勿使用 Pytest 过高版本,且勿使用 allure-Pytest 插件,会一直报错 Pytest 找不到 allure 错



误,网上的办法也解决不了,亲测使用 Pytest 3.8.0 搭配 pytest-allure-adaptor 1.7.10 可以解决

- ② 建议使用 Pytest-allure-adaptor 1.7.10 版本
- 命令: pip install pytest-allure-adaptor
- ③ 需要安装 jdk 1.8 以上版本
- ④ 需要安装 allure-commandline (先安装 npm 包)

命令: npm install -g allure-commandline --save-dev

接下来按照在 run.py 文件中写的代码

⑤ Pytest 命令基础上加--alluredir, 生成 xml 报告。

pytest -s -q --alluredir [xml_report_path]

⑥ 使用 Command Tool 来生成我们需要的美观报告。

allure generate [xml_report_path] -o [html_report_path]

⑦ 直接用 chrome 浏览器打开报告,报告会是空白页面

解决办法: 在 pycharm 中右击 index.html 选择打开方式 Open in Browser 即可

🔵 Allure	ALLURE REPORT 10/30/2019		TREND
Overview Categories	17:33:47 - 17:33:47 (260ms)	100%	
Suites	test cases		
III Graphs D Timeline	SUITES 2 items total		
Behaviors	test_basic		There is nothing to show
Packages		Show all	CATEGORIES 0 items total
	ENVIRONMENT		Show all
	hostname	http://10.34.4.113:8089	EXECUTORS
	environment	debug	There is no information about tests e
En	tester	吴嘉怡 - 测试环境	
< Collapse			









框架代码分为如下几个模块: Common 公共库、Conf 配置库、Log 封装日志、 Params 数据源和读取数据、Report 测试报告、TestCase 测试用例、run.py 执行文件,几 个重要的函数如下,其他代码已经放到开源 github 上,大家自行下载。

源码地址: https://github.com/kayie77/PyTestApiAuto

1) Common->Assert.py 封装断言

直接使用 python 的 assert 断言函数,用于判断一个表达式,使用方式:

assert test.assert_code(response['code'], 200)





YCLA	SS AS	sertions:
₽		init(self):
φ.		self.log = Log.MyLog()
φ.		<pre>assert_code(self, code, expected_code):</pre>
ė –		nnn
		验证response状态码
		<u>:param</u> code:
		<u>:param</u> expected_code:
		<u>:return</u> :
φ.		
φ.		try:
		<pre>assert code == expected_code</pre>
φ.		return True
φ.		except:
		self.log.error("statusCode error, expected_code is %s, statusCode is %s " % (expected_code, code))
		Consts.RESULT_LIST.append('fail')
¢.		raise

2) Common->Request.py 封装请求

目的是封装公共请求参数和封装返回内容

 def post_request(self, url, data, header, user_type):
requests_msg = Session.get_requests_msg(url, user_type)
<pre>header["debug"] = requests_msg.get("debug")</pre>
header[" <u>userid</u> "] = requests_msg.get(" <u>userid</u> ")
<pre>url = requests_msg.get("url")</pre>
self.log.info('测试地址: %s' % url)
self.log.info('请求参数: %s' % data)
if data is None:
<pre>response = requests.post(url=url, headers=header)</pre>
<pre>response = requests.post(url=url, params=data, headers=header)</pre>
except requests.RequestException as e:
<pre>print('%s%s' % ('RequestException url: ', url))</pre>
print(e)
return ()
except Exception as e:
<pre>print('%s%s' % ('Exception url: ', url))</pre>
print(e)
return ()
time_consuming = response.elapsed.microseconds/1000 # time_consuming为响应时间, 单位为毫秒
time_total = response.elapsed.total_seconds()
Common.Consts.STRESS_LIST.append(time_consuming)
response_dicts = dict()
response_dicts['code'] = response.status_code
try:

3) Conf->Config.py 读取配置文件

在 config.ini 定义好文件内容,需要再写一个读写配置文件的函数



《51 测试天地》五十六(上)

www.51testing.com





4) Params->jsonparams.py 读取数据源

Jsonparams 的作用是从 Postman 导出的 json 数据源中截取所需要的数据,如 header、url、body 等参数,拼接成 requests 请求需要的格式返回。





《51 测试天地》五十六(上)

www.51testing.com





5) TestCase->test_xxx.py 测试用例

@allure.story 用于定义被测功能的用户场景,即子功能点

@pytest.mark.parametrize 传多个参数,实现执行不同数据

从数据源获取数据后发起请求,断言结果,设置下一接口所需的变量参数









6) run.py 执行文件

获取报告输出位置;定义好需要运行哪些测试集,如果需要全部运行,则不需要定义,会自动找到 Test 开头(结尾)的包下面 test 开头(结尾)的 py 文件;使用 pytest.main(args)执行测试用例,定义好 allure 报告所需参数,使用 shell 命令生成 allure 报告







Jenkins 系列之自动化部署(1)

作者:合肥人真帅

在上一篇<u>《如何检验自身的技术水平?参加一场比赛就知道了》</u>文章中,我谈了关于 Selenium 和 Jmeter 的一些总结。这都最近比较主流的框架和工具,一个可以实现 UI 自动化测试,一个可以实现 API 和性能测试,其实还有一种工具可以更大限度的让 Selenium 和 Jmeter 在工作生产中发挥自身价值,那就是持续集成。

一、自动化部署的优点

持续集成是一种软件开发实践,即团队开发成员经常集成他们的工作,通常每个成 员每天至少集成一次,也就意味着每天可能会发生多次集成。每次集成都通过自动化的 构建(包括编译,发布,自动化测试)来验证,从而尽早地发现集成错误。这是百度百 科给出的名词解释。如果大家不太理解,我可以给大家举个例子。

比如说,一个产品,定义了很多条需求。如果等到所有需求开发完成,再进行测试,需要耗费大量的时间。如果测试与开发可以并行,就会缩短项目周期。思想如下:



一条业务线运行,先开发,后测试,假设需要10天;那么测试开发并行可能7、8 天就能结束工作;但是我们发现每一次测试前,都需要部署。重复的部署工作,还是耗费了一些时间的,如果可以让电脑自动完成部署工作,是不是6、7天就能完成工作?





这个就要具体去分析部署的工作流程了。



这是一个比较简单的部署流程,除了源码下载,其余各阶段,自动化执行都比手工 敲打命令速度要快。所以部署的节省时间可以用下面这个公式去计算

节省时间=(单次手工部署耗时*部署次数)-(手工编写脚本耗时+单次自动部署耗时* 部署次数)

假设每次手工部署需要 20 分钟,自动化部署需要 8 分钟,部署 10 次,编写脚本 20 分钟

那么节省时间=(20*10)-(20+8*10)=100分钟

若系统需要分布式部署,节省时间就是成倍增长的。只要节省时间较大,自动化部 署方案就可以执行。**这便是自动化部署的第一个有点,节省时间。**

自动化部署的第二个优点是部署流程严格按照规范执行。其实很多公司在设计、开发、测试、运维等工作领域都制定了一套适合自己公司的流程规范以及制度。但实际上,这些流程和规范随着时间的推移,逐渐被淡化,甚至有些公司在立项时,整个项目就已陷入混乱而不自知。但是机器则不同,每次都会按照你所制定的方案严格执行。

举个例子,上述部署流程的第二步是备份老系统。很多公司没有专业的运维人员, 都是开发或测试兼职,在部署上,安全意识淡薄,可能前几次部署,还会去备份,然后 就会觉得麻烦,渐渐抛弃这个步骤,等到部署出现问题,需要回滚时,才发现没有备 份。所以最近几年,不乏有一些大厂出现删库跑路的新闻,若真的备份了,即便删了 库,还原就行了,又何必跑路呢?但是自动化部署就不同,只要你在里面编写备份的脚





本,部署成功后,就一定可以找到备份文件。

自动化部署的第三个优点是输入命令的准确率 100%。这个也很好理解,人总有疲 惫、心烦等各种情绪和状态,即便是满血状态,也可能会在输入命令的时候,打错个别 单词,若是在上述部署流程中的"修改配置文件"环节打错字,系统不会报错,待服务 启动后,发现各种问题,才回去检查之前的步骤,此时已经耽误相当多的时间了。对自 动化部署而言,这条优点就完美体现出来了。

当然优点不止这3条,缺点也是有的,我就不一一列举了。持续集成的工具有很多,如TeamCity、Jenkins、Buildbot等。目前Jenkins的使用人数占比相对较高,下面我就从安装到工程的构建介绍一下Jenkins的使用。

二、Jenkins 安装

Jenkins 的官网可以下载到 jenkins 的安装包,不同的平台,选择不同的入口

3ai ³ 百度	jenkins	;								Ó	百度-	-下
	网页	资讯	视频	图片	知道	文库	贴吧	采购	地图	更多»		
	百度为您找	©到相关结	果约27,80	10,000个					7	7 搜索工具		
这是jenkins的	Jenkins Jenkins 支持构建 https://jen Jenkins是 fr,旨在 功能 启室 https://bai	← 个开源 部署 一百度百 一个开源 提供一个分 ike.baidu. 入门(一) 325日 - je	自动化服 → - 百度 · 本 · 本 · 本 · 本 · 本 · 本 · 本 · 本		enkins是 目的需要。 Java开发的 会,使软件 CSDN博 是一个开始	开源CI&CC 文档下 的一种持续 能的持续集 <u>客</u> 愿的、可指			<u>留过</u> 1000 (() ())) ()))))))))))))	个插件来 复的工		
	码的编译、打包、部署)的基于web界面的平台。允许持续集成和持续交付项目,无 ⓒ CSDN技术社区 → - 百度快照 自动化部署之jenkins及简介 - jimmy_xuli - 博客园 2018年5月10日 - 这就是jenkins流行的原因,因为jenkins什么插件都有 Hudson是Jenkins的前身,是基于Java开发的一种持续集成工具,用于监 控程序重复的工作,Hudson后来被收购,成为商业版。 https://www.cnblogs.com/jimmy → - 百度快照											
	<u>安装Jen</u> Jenkins 是 镜像)下载 https:// je r	https://www.cnblogs.com/jimmy ▼ - 百度快照 安装Jenkins Jenkins 是一个开源自动化服务器 像Jenkins这样的应用程序可以作为只读的"镜像"(或Docker 镜像)下载,每个镜像都作为容器在Docker中运行。Docker容器实际上是Docker镜 https://jenkins.io/zh/doc/book ▼ - 百度快照										
	CitHub	ionking	eci/ionk	ine: Ion	kine auf	omation	eonior					







Jenkins	播音 文档 • 插件 社区 • 子项目 • 关于 • 中文 • 下载
	よの ひを
CD.FOUNDATION	了解持续交付基金会 CDF 充当 Jenkins, Jenkins X, Spinnaker 和 Tekton 等許多波展最快的持续交付項目的供应 海中立平台。 学习认识 CDF
开始使用 Jenkins Jenkins 项目产生两个发行线、长期支持版本 (LTS) 和每周更新版本。相握作 两个版本都以 .war 文件,原生包,安装程序,和 Docker 容器的形式分发。 长期支持版本 (LTS)	博客 文档 ◆ 插件 社区 ◆ 子项目 ◆ 关于 ◆ 中 尔的组织需求,一个可能比另一个更受欢迎。 带有 嗡 齿轮的包由第三方维护。
开始使用 Jenkins Jenkins 项目产生两个发行线,长期支持版本 (LTS) 和每周更新版本,根据代 两个版本都以,war 文件,原生包,安装程序,和 Docker 容器的形式分发。 长期支持版本 (LTS) LTS (长期支持)版本每12周从常规版本流中选择,作为该时间段的稳 定版本,了解更多	「「「「「「「「「」」」」」」」」 「「「」」」」 「「」」」」 「「」」」」 「「」」」」 「「」」」」 「「」」」」 「「」」」」 「「」」」」 「」」 「」」」 「」」」 「」」」 「」」」 「」」 「」」 「」」」 「」」」」 「」」」 「」」 「」」 「」」」 「」」 「」」」 「」」 「」」 「」」」 「」」 「」」 「」」」 「」」 「」」 「」」 「」」 「」」」 「」」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」」 「」」 「」」 「」」 「」」 「」」」 「」」」 「」 」 「」 」 「」 」 「」」 「」」 「」」 「」」 「」」 「」 」 「」 」 「」 」 「」 」 「」 」 「」 」 」 「」 」 「」 」 「」」 「」 」 」 「」 」 「」」 「」 」 」 」 」 」 」 「」 」 「」 」 」 」 」 「」」 「」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」
开始使用 Jenkins Jenkins 项目产生两个发行线,长期支持版本 (LTS) 和每周更新版本,根据代 两个版本都以,war 文件,原生包,安装程序,和 Docker 容器的形式分发。 长期支持版本 (LTS) LTS (长期支持版本 (LTS) LTS (长期支持版本 (LTS) 实更目志 升级指南 以前的发行版	 「「「「「「「」」」」」 「「「」」」」 「「」」」 「「」」」 「「」」」 「「」」」 「」」」 「」」 「」 「」」 「」 <li< td=""></li<>
开始使用 Jenkins Jenkins 项目产生两个发行线、长期支持版本 (LTS) 和每周更新版本。相場代 两个版本都以 war 文件, 原生龟, 安装程序, 和 Docker 容器的形式分发。 长期支持版本 (LTS) LTS (长期支持)版本每12周人常规版本流中选择, 作为该时间段的稳 定版本, 了解更多 変更日志 [升级指南] 以前的发行版 口 部置 Jenkins 2.190.2	 「「「「「「「」」」」」 「「「」」」」 「「」」」 「「」」」 「「」」」 「「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」」」 「」」」 「」」 「」 「」
田始使用 Jenkins Jenkins 項目产生两个发行线, 长期支持版本 (LTS) 和每周更新版本, 根据任 两个版本都以, war 文件, 原生包, 安装程序, 和 Docker 容器的形式分发。 长期支持版本 (LTS) LTS (长期支持) 版本每12周人常规版本流中选择, 作为该时间段的稳定版本, 了解更多 愛更日志 升级指悔 以前的发行版 C 部實 Jenkins 2.190.2 C Deploy to Azure	 「「「「「「「」」」」」」 「「「」」」」 「「」」」」 「「」」」」 「「」」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」「」」」 「」」」 「」」 「」 「」
开始使用 Jenkins Jenkins 项目产生两个发行线、长期支持版本 (LTS) 和毎周更新版本、根漏体 两个版本都以 war 文件, 原生龟, 安装程序, 和 Docker 容器的形式分发。 长期支持版本 (LTS) 氏期支持版本 (LTS) LFS (长期支持版本 (LTS) LFS (长期支持版本 (LTS) 正版本、了解更多 愛更日志 升级指南 以前的发行版 C 部署 Jenkins 2.190.2 C T 電 Jenkins 2.190.2 for: 这边是稳定版	 博客 文档 ・ 插件 社区 ・ 子项目 ・ 关于 ・ 中 外的组织需求,一个可能比另一个更受欢迎。 市有 噏 齿轮的包由第三方维护。 毎月周更新版 毎周更新版 毎周更新版 毎周更新版 ● 百個人の一般のでは、 ● 百個人の一般のでは、 ● 百個人の一般のでは、 ● 百個人の一般のでは、 ● 百個人の一般のでは、 ● 古場 Jenkins 2.204 for: ■ Arch Linux 噏 ■ Docker <u>这边是最新版</u>
	 藤舎 文档 ◆ 插件 社区 ◆ 子项目 ◆ 关于 ◆ 年 第約/组织需求,一个可能比另一个更受欢迎。 蒂有 확 齿轮的包由第三方维护。 每周更全新版 每周更全发布一个新版本,为用户和插件开发人员提供错误修复和功能。 麦更日志 以前的发行版 百 転 Jenkins 2.204 for: Arch Linux 象 Docker 这边是最新版 FreeBSD 象
	 「「「「「」」」」」 「「「」」」」 「「」」」」 「「」」」」 「「」」」」 「」」」 「」」 「」 「」」 「」 「」 「」」
田内は使用 Jenkins Jenkins 项目产生两个发行线、长期支持版本 (LTS) 和毎周更新版本、根漏ゲ 岡ケ版本都以 war 文件, 原生包、安装程序, 和 Docker 容器的形式分发。 长期支持版本 (LTS) ITS (长期支持版本 (LTS) ITS (长期支持版本 (LTS) ITS (长期支持版本 (LTS) ITS (大期支持版本 (LTS) ItS (LTS)	 ・ 描件 社区 、 子项目 、 关于 、 年 ・ ・ ・
田林田市 項目产生两个发行线、长期支持版本(LTS)和海周更新版本、根据作两个版本都以、wer 文件, 原生包。安装程序、和 Docker 容器的形式分发。 大期支持版本(LTS) LTS(长期支持版本(LTS)) DTS(長期支持版本(LTS)) LTS(长期支持版本(LTS)) LTS(长期支持版本(LTS)) LTS(长期支持版本(LTS)) LTS(长期支持版本(LTS)) Docker FreeBSD Gentoo 電 Mar OS X	 「「「「「」」」」」 「「「」」」」 「「」」」」 「「」」」」 「」」」 「」」 「」」 「」」」 「」」」
田内がしていた。 田内がしていた。 サインジングングングングングングングングングングングングングングングングングングン	 糖溶 文档 ・ 插件 社区 ・ 子项目 ・ 关于 ・ 年 外的组织需求, 一个可能比另一个更受欢迎。 市有 噏 齿轮的包由第三方维护。 毎月更新版 毎月更新版 毎月更新版 毎月更新版 毎月更新版 ● 万種 Jenkins 2.204 for: ▲ Tch Linux 噏 □ Docker 这边是最新版 FreeBSD 噏 Gentoo 噏 Mac OS X OpenBSD 噏 统, 选择对成的安装包
田林協協 第四日 今生两个没行线, 长期支持版本 (LTS) 和晦阑更新版本, 根据保 两个版本都以, war 文件, 原生龟, 安装程序, 和 Docker 脊髓的形式分发。 田方枝方板 (LTS) TS (长期支持版本 (LTS) TS (大期支持版本 (LTS) S更目志 升級指南 以前的发行版 S更目志 升級指南 以前的发行版 S里目志 升級指南 以前的发行版 Cocker Decker FreeBSD Gentoo 喩 Mac OS X QenBSD 嗓	 「「「「」」」」」 「「」」」」 「「」」」」 「」」」」 「」」」」 「」」」 「」」 「」」
エーサムは使用 Jenkins Jenkins 项目产生两个没行线、长期支持版本 (LTS) 和每周更新版本、根据体 两个版本都以、war 文件, 原生龟、安装程序、和 Docker 客器的形式分发。 大期支持版本 (LTS) LTS (长期支持版本 (LTS) LTS (长期支持版本 (LTS) 正ち (大期支持版本 (LTS) 正ち (大期支持版本 (LTS) 正ち (大期支持版本 (LTS) 正ち (大期支持版本 (LTS) 正本 (1983) 正ち (大期支持版本 (LTS) 正本 (大期支持版本 (LTS)) 正本 (LTS)	 博客 文档 - 插件 社区 - 子项目 - 关于 - 中 中 中 中 中 中 中 中 市 転 中 市 市 転 支 力 中 支
Hordeben Jenkins Jenkins 项目产生两个发行线、长期支持版本(LTS)和每周更新版本、根据/ Pof 版本都以 war 文件, 原生包、安装程序, 和 Docker 容顯的形式分支。 长期支持版本 (LTS) LS (长期支持版本 (LTS) LS (长期支持版本 (LTS) LS (长期支持版本 (LTS) LS (长期支持版本 (LTS) LS (大期支持版本 (LTS) LS (L	 糖密 文档 - 插件 社区 - 子项目 - 关于 - 年 中部组织需求, 一个可能比另一个更受欢迎。 市有 傘 齿轮的包由第三方维护。 毎月御新版 毎周都会发布一个新版本, 为用户和插件开发人员提供错误修复和功能。 変更日志 以前的发行版 ● T毛 Jenkins 2.204 for: Arch Linux 傘 Docker <u>这边是最新版</u> FreeBSD 傘 Gentoo 傘 Mac OS X OpenBSD 傘 统, 选择对应的安装包 openSUSE Red Hat/Fedora/CentOS Ubuntu/Debian
Hyber B Jenkins Jenkins 项目产生两个发行线, 长期支持版本 (LTS) 和每周更新版本, 根据 两个版本都以 war 文件, 原生龟, 安装程序, 和 Docker 容器的形式分发。 大期支持版本 (LTS) LTS (长期支持版本 (LTS) LTS (长期支持版本 (LTS) LTS (长期支持版本 (LTS)) LTS (长期支持版本 (LTS) LTS (大期支持版本 (LTS) LTS (大期支持版本 (LTS)) LTS (大期支持版本 (LTS) LTS (大期支持版本 (LTS)) LTS (LTS (LTS (LTS))) LTS (LTS (LTS (LTS (LTS))) LTS (LTS (LTS (LTS (LTS (LTS)))) LTS (LTS (LTS (LTS (LTS (LTS (LTS (L	### 文档 + 活件 社区 - 子项目 + 关于 + 年 ### **********************************
Hordeben Jenkins Jenkins 项目产生两个发行线、长期支持版本(LTS)和每周更新版本、根据 Pof 版本都以 war 文件, 原生龟、安装程序, 和 Docker 容器的形式分支。 K期支持版本 (LTS) Its (长期支持版本 (LTS) Its (长期支持版本 (LTS) Its (大期支持版本 (LTS) Its (博客 文档 ・ 插件 社区 、 子项目 、 关于 、 体 中端组织需求, 一个可能比另一个更受欢迎。 市事 噏 齿轮的包由第三方维护。 毎月恵子新版 毎周都会发布一个新版本, 为用户和插件开发人员提供错误修复和功能。 変更日志 以前的发行版 ● Tを出 Jenkins 2.204 for: Arch Linux 噏 Docker 这边是最新版 FreeBSD 噏 Gentoo 噏 Mac OS X OpenBSD 噏 統, 选择对应的安装包 openSUSE Red Hat/Fedora/CentOS Ubuntu/Debian OpenIndiana Hipster 噏 「博为峰旗下」 近日でしていていの。

例如 Windows, Linux 的各个版本等。Windows 直接下载压缩包, 解压后运行 msi, 一直点击下一步,即可完成安装。





💼 jenkins-2.156.zip 名称 × ..(上层目录) 🛃 jenkins.msi Linux 的安装,可以下载到本地安装,也可以在线安装,不同的 Linux 安装命令也 不同,例如 Ubuntu 的命令是 apt, CentOS 的命令是 yum。 **RedHat Linux RPM packages for Jenkins** Jenkins Debian packages To use this repository, run the following command: This is the Debian package repository of Jenkins to automate installation and upgrad o.key | sudo apt-key add sudo wget -0 /etc/yum.repos.d/ sudo rnm --import https://nkg. ins.repo https://pkg.jenk ins.io/redhat-stahle/jenk ns.io/red ns.io.key Then add the following entry in your /etc/apt/sources.list: If you've previously imported the key from Jenkins, the "rpm --import" will fail because you deb https://pkg.jenkins.io/debian-stable binary You will need to explicitly install a Java runtime environment, because Oracle's Java RPMs ar OpenJDK JVM You will need to explicitly install Java, **because Jenkins does not support all Java v** undesired versions of the JVM. Check Java requirements in Jenkins for more details. • 2.164 (2019-02) and newer: Java 8 or Java 11 2.54 (2017-04) and newer: Java 8
1.612 (2015-05) and newer: Java 7 2 164 (2019-02) and newer: Java 8 or Java 11 2.104 (2017-02) and newer: Java 8
 2.54 (2017-04) and newer: Java 8
 1.612 (2015-05) and newer: Java 7 With that set up, the Jenkins package can be installed with Update your local package index, then finally install Jenkins: apt-get update ant-get install jenkin See Wiki for more information, including how Jenkins is run and where the configuration is s See Wiki for more information, including notes regarding upgrade from Hudson. Individual Package Downloads **Individual Package Downloads** If you need *.rpm for a specific version, use these If you need *.deb for a specific version, use these Name Last modified Size CentOS Name jenkins-2.190.2-1.1.noarch.rpm jenkins-2.190.1-1.1.noarch.rpm Ubuntu 2019/10/28 2019/09/25 74.4M 74.4M Last modified Name Size jenkins-2.176.4-1.1.noarch.rpm jenkins-2.176.3-1.1.noarch.rpm ienkins 2,190,2 all deb 2019/10/28 73.9M 2019/09/25 73.6M jenkins_2.190.1_all.deb 2019/09/25 74.2M 2019/08/28 73.6M jenkins 2.176.4 all.deb 2019/09/25 73.2M ienkins-2.176.2-1.1.noarch.rpm 2019/07/17 73.6M 72.2M 73.5M jenkins_2.176.3_all.deb 2019/08/28 jenkins-2.176.1-1.1.noarch.rpm 2019/06/10 73.5M jenkins 2.176.2 all.deb 2019/07/17 jenkins-2.164.3-1.1.noarch.rpm 2019/05/10 73.5M jenkins-2.164.2-1.1.noarch.rpm 2019/04/10 73.5M

安装完成后,打开浏览器,输入 localhost:8080

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

C:\Users\Administrator\.jenkins\secrets\initialAdminPassword

Please copy the password from either location and paste it below.

Administrator password

第一次安装,需要解锁,解锁密码就在红色路径的文件里。解锁后,根据提示安装插件,创建管理员账号,安装就算完成了。如果在安装后发现8080端口被占用,导致jenkins无法运行,应该先停止jenkins服务,然后找到配置文件修改端口号。Windows的配置文件,就在jenkins安装路径jenkins.xml文件中修改。而Linux则是在





/etc/sysconfig/jenkins 文件中修改,不同版本的 linux 或许配置文件位置不同,但都在/etc 下,仔细找找是可以找到的,也可以用 whereis 来查找。

jankin	ns. and	
	AU	THORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	LI	LABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
	00	JT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
	TH	HE SOFTWARE.
3	L	-7
24		
25 8		
26	T	Windows service definition for Jenkins.
8		To uninstall, run "jenkins.exe stop" to stop the service, then "jenkins.exe uninstall" to uninstall the service.
29		Both commands don't produce any output if the execution is successful.
	L	->
	- <s< td=""><td>service></td></s<>	service>
	T	<id>Jenkins</id>
		<name>Jenking</name>
14		<pre><description>This service runs Jenkins automation server.</description></pre> /description>
5		<pre><env name="JENKINS HOME" value="%BASE%"></env></pre>
6 1	Ь.	</td
	T	if you'd like to run Jenkins with a specific version of Java, specify a full path to java.exe.
18		The following value assumes that you have java in your PATH.
9	Ŀ.	
		<pre>securable>%BASE%\ire\bin\iava</pre> define
		<arguments>-Xrs -Xnx256m -Dhudson,lifecycle=hudson,lifecycle.WindowsServiceLifecycle -jar "%BASE%\jenkins.war" -Difile.encoding=utf-8httpPort=8080webroot="%BASE%\war"</arguments>
2 1	ė –	</td
3	Т	interactive flag causes the empty black Java window to be displayed.
4		I'm still debugging this.
5		<interactive></interactive>
6	L .	>
		logmode>rotate
8		
9		<pre><onfailure action="restart"></onfailure></pre>
	ė.	</td
	T	In the case Win3W gets terminated and leaks the process, we want to abort
3		these runaway JAR processes on startup to prevent corruption of JENKINS HOME.
4		So this extension is enabled by default.
S		>
6 8	ė.	<extensions></extensions>
		This is a sample configuration for the RunawayProcessKiller extension
8		<extension <="" enabled="true" td=""></extension>
9		<pre>className="winsw.Plugins.RunawayProcessKiller.RunawayProcessKillerExtension"</pre>
	ė.	id="killOnStartup">
		<pre><pidfile>%BASE%\jenkins.pid</pidfile></pre>
		<stoptimeout>10000</stoptimeout>
3		<stopparentfirst>false</stopparentfirst>
4	-	
65	L.	

三、添加插件、进行各项配置

Jenkins 插件有很多,由于各个公司对源码管理,项目管理等使用的工具都各不相同,这里就不一一介绍了。这里介绍1个大家都用的上的插件,Email Extension Plugin,这个插件可以将每一次的构建结果以邮件的方式发送到指定邮箱。(还有1个 HTML Publisher 插件,我会在下一篇给大家讲解,如果感兴趣,可以自己摸索一下)

Jenkins		系统设置 全局设置和路径
쯜 新建任务 鵗 用户列表		全局安全配置 Jenkins安全,定义谁可以访问或使用系统。
📄 构建历史 🔍 项目关系		凭据配置 配置凭据的提供者和类型
杜童文件指纹 茶統管理	X	全局工具配置 工具配置,包括它们的位置和自动 安装器
43 我的视图 Lockable Resources		读取设置 放弃当前内存中所有的设置信息并从配置文件中重新读取 (
📯 凭据 📄 新建视图	•	插件管理 添加、删除、禁用或启用Jenkins功能扩展插件。 ▲ 可用更新

点击系统管理→插件管理→可选插件,进行搜索,选中后点击安装。安装成功后, 刷新页面,就可以在已安装页面中找到。





Senkins) 插件管理			4 风直线	④ 杨吉波 注销
★ 返回工作会 参 系統管理	可要數可透描件已会被 高级		1338 <u>9</u> ,	
	肩用	名称 ↓	版本	上一个安装的版本 卸载
	Ant Adds Apache Ant support to Jenkins	1	1.9	
	Apache HttpComponents Client 4.x API Bundles Apache HttpComponents Cli	tient 4.x and allows it to be used by Jenkins plugins.	<u>4.5.5-</u> <u>3.0</u>	
	Authentication Tokens API Plugin This plugin provides an API for conve	erting credentials into authentication tokens in Jenkins.	13	2142
	Add Badges for a build and extend th	he build summary	18	
	bouncycastle API Plugin This plugin provides an stable API to	p Bouncy Castle related tasks.	2.17	2141
	Branch API This plugin provides an API for multip	ple branch based projects.	212	
	This plurie allows builds to be sufarm	nstissible tarministiart star i tha enactifiant smount of tima has alsocart	1.19	2945

我这里用的是 git 管理源码,如果是 svn,操作方法是相同的。安装 git 插件,搜索 git,如下图

		1 7	
	Git		
1	011		
		This plugin integrates Git with Jenkins.	

安装后,在系统管理→全局工具配置页面下,配置本机 git 环境

laven 配置			
默认 settings 提供	使用默认 Maven 设置		
默认全局 settings 提供	使用默认 Maven 全局设置	1	
DK			
JDK 安装			
lit			
Git installations	Git		
	Name		
	Path to Git executable	D.\Giltcmdigit.exe	
	□ 自动安装		
			Delete Git
	Add Git 👻		
Fradle			
Gradle 安装	新增 Gradle		
	系统下Gradle 安装列表		
int			
Ant 安装	新增 Ant		
	系统下Ant 安装到表		

如果 jenkins 服务器和部署的目标服务器不是同一台机器,那么就需要配置一个远程 节点。类似于 Jmeter 的分布式压测。

点击系统管理→节点管理→新建节点,如下图





Jenkins ▼ → 节点 ◆ 返回工作台			
🔺 返回工作台	列表 →		
		节点名称	
		◎ 固定节点	
。新建节点 1. 新建节点		添加一个晋通、固定的节点到Jenkins。之所以 ◎ 有則现有节点	以叫做"固定",是因为Jenkins没给这种节点提供更高级的集成
(F HUEL		要复制的任务名称	
构建队列	-		
人列中没有构建任务	20	确定	
构建执行状态	_		
🖷 master			
 1 空闲			
2 空闲			
💻 234			
名字	Linux36		
描述			
并发构建数	1		
远程工作目录			
	▲ 法程日录单心情场		
标签	₩ 延性日來走必項坝		
e a faile.			
用法	尽可能的使用这个节点		
启动方式	Launch agent agents via Of		
	Launch agent agents via 53		
	主机		
	Credentials		云加 👻
		-无-	arthi
		The selected credentials cannot be for	bund
	Host Key Verification Strate	Known hosts file Verification Strategy	

到这个页面后, linux 和 windows 的配置方法是不一样的, 我先介绍一下 linux 的配置方法。描述字段为了后期方便管理, 最好写上和服务器相关的信息, 例如 IP, 项目名称等。远程工作目录是 jenkins 调用该节点时, 工作文件的保存路径。





谷称 Inux36 描述 192.168.1.36 井炊构建数 1 远程工作目录 home/jenkins 标益		
描述 192.168.1.36 井发柏建士 1 远程工作目录 /home/jenkins 标签	名称	linux36
井渋柏建数 1 近曜工作目表 /nome/jenkins 标签 用法 尽可能的使用这个节点 启动方式 Launch agents via SSH 主机 192 168.1.36 Credentials ● ● 添加 ● Host Key Verification Strategy ● 待部署的服务器 受尿服务 ● ● 小山 ● 可用性 尿量保持代理在线 可用性 尿量保持代理在线 「Disable deferred wipeout on this note ● 「其位置列系 「な称 (Git) ghs 工具名称 「現位置列系 「な称 (Git) ghs 工具名称 「現会業路径 (待部署的目标服务器上安装的git目录) 「维力修復工	描述	192.168.1.36
近程工作目录 home/jenkins 标签 用法 尽可能的使用这个节点 启动方式 Launch agent agents via SSH 主机 「192.168.1.36 「cod/****** 「 「192.168.1.36 「cod/****** 「 「 「 「 「 「 「 「 「 「 「 「 「 「 「 「	并发构建数	1
标签 用法 尽可能的使用这个节点 启动方式 Launch agent agents via SSH 主机 192.168.1.36 Credentials 「oot/****** Host Key Verification Strategy 侍部署的服务器 登录服务器的帐号和密码 注接方式	远程工作目录	/home/jenkins
朋法 尿可能的使用这个节点 启动方式 Launch agent agents via SSH 主机 192.168.1.36 Credentials oot/****** Host Key Verification Strategy 何部署的服务器 受录服务器的帐号和密码 连接方式 可用性 尽量保持代理在线 Disable deferred wipeout on this node Image: Tap位置 「Git) ghs 「具位置列環 谷称 「Git) ghs 「具名称 「夏愛」/usr/bin/git 「具安装路径(待部署的目标服务器上安装的git目录)	标签	
启动方式 Launch agent agents via SSH 主机 「2.168.1.36 Credentials Host Key Verification Strategy Non verifying Verification Strategy Non verifying Verification Strategy 学录服务器的帐号和密码 连接方式	用法	尽可能的使用这个节点
主机 192.168.1.36 Credentials root/****** Host Key Verification Strategy 待部署的服务器 登录服务器的帐号和密码 连接方式 可用性 尽量保持代理在线 T 用性 尽量保持代理在线 i L具位置 [192.168.1.36] 「供加」 「日本 ····································	启动方式	Launch agents via SSH
Credentials Foot/******* ● 添加 Host Key Verification Strategy 待部署的服务器 登录服务器的帐号和密码 连接方式 可用性 尽量保持代理在线		主机 192.168.1.36
Host Key Verification Strategy Non verifying Verification Strategy 待部署的服务器 登录服务器的帐号和密码 连接方式 可用性 尽量保持代理在线		Credentials
可用性 尽量保持代理在线		Host Key Verification Strategy Non verifying Verification Strategy 待部署的服务器 登录服务器的帐号和密码
节点属性 □ Disable deferred wipeout on this node ☑ 工具位置 工具位置列表 名称 目录 工具名称 目录 工具安装路径(待部署的目标服务器上安装的git目录) 進为修施下	可用性	尽量保持代理在线
Private L ■ Disable deferred wipeout on this node ■ 工具位置 工具位置 工具位置 「 「Git) ghs 目录 /usr/bin/git 工具安装路径(待部署的目标服务器上安装的git目录) 植为修施下	井占居性	
 ☑ 工具位置 工具位置列表 名称 (Git) ghs 目录 /usr/bin/git 工具名称 工具名称 工具安装路径(待部署的目标服务器上安装的git目录) 1 <l< td=""><td>Disable det</td><td>erred wineout on this node</td></l<>	Disable det	erred wineout on this node
目录 /usr/bin/git 工具安装路径(待部署的目标服务器上安装的git目录)	<u> </u> 具位置列表	名称 (Git) ghs 工具名称
		目录 /usr/bin/git 工具安装路径(待部署的目标服务器上安装的git目录)
51test		
^m 软件测	保存	软 件 测 试

创建好以后,返回节点列表,会显示一个红色的叉,代表没有连接成功。稍等1分钟,刷新页面,若红色叉消失,代表已经连接成功。如下图

Jenkins > 节点列表 >				U	
🚖 返回工作台		S	名称 ↓		
🏠 系统管理			234		
■ 新建节点 ▲ 配置			linux36		
HUEL			master		
构建队列	-		获取到	的数据	
队列中没有构建任务					





若始终显示未连接,则需要检查目标服务器的 IP,帐号密码是否正确,端口号是否被防火墙拦截等。

Windows 配置远程节点,如下图

	名称	234				
	描述	192.168.1.234				
	并发构建数	1				
	远程工作目录	D:\ienkins				
	标签					
	用法					
	启动方式					
		通过Java Web启动代生	₽			
		禁用作目录 自定义作目录				
		内部数据目录				
		当工作目录轴牛时牛附	remoting			
	可用性	民母们结伴理去线				
		《重闲司 》[21]				
	节点属性					
	Disable def	ferred wipeout on this no	ode			
	□ ⊥具121重 □ 环境变量					
	保住					
配置完成后,返回	日节点列表					
Jenkins 🔸 节	点列表→					
🔺 返回丁作台				e	2称	
					יעיים ו	1
					<u>234</u>	· 推为给按下 · · · · · · · · · · · · · · · · · · ·
■ 新建口品					linux36	51testing
🚮 配置						软件测试网

目前状态是未连接,他不会像 linux 那样自动连接,需要我们手动建立连接。点击



)[@SL0/19 软件测试网	《51 测试天地》五十六(上) www.51testing.com
名称,跳转到如下页了	
▲ 3000円枚 ④ 005 ● 005 ● 和田が市 ● 和田が足 ■ 日本 和田が上 日本	CHUE 234 (192.168.1.234) EMMERATION TOF ENVIRONMENT/STORT:

这里提供了两种建立连接的方式,第一种比较简单,点击 Launch 按钮,会下载一个 slave-agent.jnlp 文件,将这个文件发送到远程机上,双击运行,如下图



节点的红色叉消失, 连接成功。

博为峰旗下

第二种方法是点击 agent.jar, 下载文件。复制页面的命令

java -jar agent.jar -jnlpUrl http://localhost:8080/computer/234/slave-agent.jnlp -secret 2a406deefbbc9a1ee1c73945ad0993019356ad02e7c09e8237598a660cd70ce8 -workDir "D:\jenkins"

在 agent.jar 前面加上本地路径,例如 D:/download/agent.jar

localhost 要改成 jenkins 服务器的 ip

运行结果如下,出现 Connected 表示连接成功









四、创建任务,构建自动化部署

在 jenkins 首页,点击新建任务。输入任务名称,选择"构建一个自由风格的软件项目",点击确定。





没 Jenkins Jenkins ▶ 所有 ▶	
[输入一个任务名称
	AutoDeployment
	构建一个自由风格的软件项目 这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来
	构建一个maven项目 构建一个maven项目.Jenkins利用你的POM文件,这样可以大大减轻格
	流水线 精心地组织—个可以长期运行在多个节点上的任务。适用于构建流水
	External Job This type of job allows you to record the execution of a process run
	构建一个多配置项目 适用于多配置项目,例如多环境测试,平台指定构建,等等.
	GitHub 组织 扫描一个 GitHub 组织(或者个人账户)的所有仓库来匹配已定义的标
	多分支流水线 根据一个SCM仓库中检测到的分支创建一系列流水线。
	文件夹 创建一个可以嵌套存储的容器。利用它可以进行分组。 视图仅仅是一
	如果你想根据一个已经存在的任务创建,可以使用这个选项
	复制 輸入自动完成 博为峰旗下 511005 511000

进入任务的配置页面,顶部有很多 tab 页,根据自己的实际情况去添加配置即可, 我这里演示一个最简单的配置方法。如下图

General	源码管理	构建触发器	构建环境	构建	构建后操作	根据自己的实际情况去进行配置	
描述							
						_	
		[纯文本] <u>预览</u>					
GitHub	项目						
This bui	ild requires lock	able resources					
Throttle	builds						
□ 丢弃旧	的构建						
□ 参数化材	勾建过程						
□ 关闭构。	主						
 在必要 	的时候并发构建						
 限制项目 	目的运行节点						
标签表试	大式	linux36					
		Label linux26	in considered by 1	1 nodo De	emissions or oth	er restrictions provided by plugies may prevent this jab from suppley on these pades	
		Laber Infux36	is serviced by	I Houe. Pe	simissions of our	er resarctions provided by plugins may prevent ans job nom running on arose nodes.	171
							22





若 jenkins 服务器和部署的目标服务器不是同一台,可以选择部署的目标服务器作为运行节点。例如我们要部署到 linux36 这台机器上,就可以选择我们刚才添加的 linux36 这个节点。

我使用的是 git 管理源码,在 URL 输入 git 的地址, Credentials 输入帐号密码, BreanchSpecifier 输入需要获取的代码的分支。

源码管理			
◎ 无 ● Git			
Repositories	Repository URL http://192.168.1.86/TestSoft/TestSoft_hf.git	0	0
	م Add Repos	硪… itory	
Branches to build	Branch Specifier (blank for 'any) VStable	×	
	Add Br	anch	
源码库浏览器	bweb 博为峰旗下	· `	0
	RL http://192.168.1.86/TestSoft/TestSoft_hf.git	ПП	10
Additional Behaviours			2
Subversion	软件测	试	XX

若是 SVN 管理源码,则选择 SVN,配置方法类似。

当前部署的项目,是独立项目的,可以选择定时构建。语句分为5段,每段用空格 隔开,从左到右分别表示分钟、小时、日、月、星期,图中的语法就是每周1到周6,中午12点整执行一次。

若是有前置条件的项目,可以选择其他触发方式。

构建触发器							
Build whenever a	Build whenever a SNAPSHOT dependency is built						
🔲 触发远程构建 (例	触发近程均键(例如,便用脚本)						
GitHub hook trig	ger for GITScm polling		0				
□ 其他工程构建后期	□ 其他工程构建后触发						
✓ 定时构建			Ø				
日程表	H 12 * * 1-6		0				
■ 轮询 SCM							

接下来是构建具体的执行命令,执行构建的节点服务器是 windows 的可以选择批处 理,是 linux 的可以选择 shell,可以选择框架或工具类的命令例如 maven, ant 或各种脚 本语言 nodejs, invoke Gradle,若图中没有你想要的构建方法,可以到插件管理页面, 下载想要的插件,安装后,就可以在这里看到选项了。





	増加构建步骤 →	
	Execute NodeJS script	
	Invoke Ant	
	Invoke Gradle script	
	Provide Configuration files	
	Run with timeout	
	Send files or execute commands over SSH	
	Set build status to "pending" on GitHub commit	
	执行 Windows 批处理命令	
	执行 shell	
	- 调用顶层 Maven 目标	
构建		
执行 shell		x
部令 h /var/ Ant Oel orgent. cp - r/var/Ant Oel logner. rn - rf /var/Ant Oel logn cp - r/now/jenkingw add -1 //10// c add -1 //10// c //不同语言論写的第次[] h /var/Ant Oel orgent. rn - rf /home/jenking/	(400, h) ment Nichyalo (AutoDeployment/ Source /var/AutoDeloyment pp/)、192,108,1727, /var/AutoDeployment/database.xml (大)、Vadaut, /var/AutoDeployment/database.xml (本社、人)、1924607, /var/AutoDeployment/database.xml (本社、人)、中國日本的語彙報告会 (本社、人)、中國日本語彙報告会	
查看 可用的环境变级列爆		

我这里用的是 shell, 第一行是停止当前服务, 第二行是备份当前服务, 第三行是删 除当前服务(相当于清理了环境), 第四行是将最新的源码文件拷贝到原来的服务文件 夹, 第五至七行是在配置数据库服务文件, 第八行由于每个公司的源码是用不同的语言 编写的, 有 java, C++, C#, python 等等, 他们的编译命令不同, 就不举例了。第九行 是启动服务, 第十行删除下载的源码(清理 workspace), 这样一个简单的部署就完成 了。部署完成后, 可以让 jenkins 发送邮件, 这样即使你不在电脑前, 手机也可以收到邮 件, 来查看最新的部署结果。





Editable Email Notification			•
Disable Extended Email Publisher			•
	Allows the user to disable the publisher, while maintaining the settings		
Project From			
Project Recipient List	SDEFAULT RECIPIENTS		•
		10	
	Comma-separated list of email address that should receive notifications for this project.		
Project Reply-To List	\$DEFAULT_REPLYTO		
	Comme consisted first of small address that should be in the Darks Te bandra for this section.	10	
Quelest Test	Comma-separated list of email address that should be in the Reply-to neader for this project.		0
Content Type	Both HTML and Plain Text	•	
Default Subject	\$DEFAULT_SUBJECT		0
Default Content	<idoctype html=""></idoctype>	*	0
	<html> <head></head></html>		
	<meta charset="utf-8"/>		
	 <body leftmargin="8" marginheight="4" marginwidth="0" offset="0" topmargin="8"></body>		
	<to><td< td=""><td></td><td></td></td<></to>		
	<hr/> size="2" width="100%" align="center" />		
	্থায়> বাহ		
	 ∢d⊳		
	 div style="font-size:18px"> 		
	<⊪构建名称: \${PROJECT_NAME} /	-	
	《i》构建结果: \$BUILD_STATUS /i》 /i》		
Attachments	report.log	M	9
	Can use wildcards like 'module/dist/**/*.zip'. See the @includes of Ant fileset for the exact format. The base directory is the workspace.		
Attach Build Log	Attach Build Log · · · · · · · · · · · · · · · · · · ·	Ť	XX
		10	1.4.9

这个邮件服务,就用到了我们之前在插件环节介绍的那款 Email Extension Plugin。 大部分字段使用默认值即可, Default Content 可以使用系统管理→系统设置→Extended E-mail Notification 模块设置的通用模版,也可以像上图这样给这个任务设置一个独立的 邮件内容。Attachments 可以选择本次部署生成的日志文件,选择 Attach Build Log,在 发送邮件时,讲日志文件以附件方式一起发送。







Triggers	Add Trigger 👻
時 ○日	Aborted Always Before Build Failure - 1st Failure - 2nd Failure - Any Failure - Still Failure - X Failure -> Unstable (Test Failures) Fixed Not Built Script - After Build Script - Before Build Status Changed Success

选择发送方式, Always 表示只要构建结束就发送邮件, Failure 表示构建结果是失败时,才发送邮件, Success 表示构建结果为成功时发送邮件,其余选项顾名思义。

Send To	Developers	× ()
	Recipient List	× 0
	新增 🗸	
	Culprits	高級。
	Developers	ten 2Y
	Recipient List	
ld Trigger 👻	Requestor	
	Suspects Causing Unit Tests to Begin Failing	
	Suspects Causing the Build to Begin Failing	
	Upstream Committers	

SendTo 是发送给哪些人,可以多选。Developers 是开发者, RecipientList 是收件人列表。可以在系统管理→系统设置→Extended E-mail Notification 模块→Default Recipients 字段设置,多个人用英文逗号隔开

SMTP server	smtp.163.com
Default user E-mail suffix	@qq.com
Default Content Type	HTML (text/html)
Use List-ID Email Header	
Add 'Precedence: bulk' Email Header	
Default Recipients	@qq.com, @qq.com





任务的所有配置结束,点击保存。	回到首页		
😥 Jenkins			
Jenkins 🕨			
音 新建任务			
🜲 用户列表	所有 +		
📂 构建历史	s	w	名称 ↓
0、项目关系		*	AutoDeployment
▲三 检查文件指纹		- 🔆 -	
🔅 系统管理		*	
鵗 我的视图	ā		博为峰旗下
🗞 Lockable Resources		*	Stesting
条 凭据	图标: 小中大		软件测试网

点击任务名称,每次构建的,都会在左边生成一个历史构建,并以数字递增,点击 超链接,查看控制台输出

Jenkins	Þ	
 ▲ 返回面板 Q、状态 ◎ 修改记录 		工程 AutoDeployment
 ■ 工作空间 ② 立即构建 ③ 删除工程 ◇ 配置 ○ 重命名 		工作区 量新修改记录
Build History	<u>构建历史</u> ー ×	 相天 (1) 技 ・ 最近—次构建(#37),1天 21 小时之前 ・ 最近稳定构建(#37),1天 21 小时之前 ・ 最近成功的构建(#37),1天 21 小时之前
 #37 2019-11-16 下午4:34 #36 2019-11-15 上午9:37 #35 2019-11-15 上午9:15 #34 2019-11-11 下午5:38 		• 最近完成的构建(#37),1天 21 小时之前





这里的提示,是构建成功的意思。可以手动去验证,本次自动化部署是否正确。若 存在问题,可以查看构建日志,要对自己的配置及各环节的命令进行错误排查。这样一 套相对简单的自动化部署就完成了。解放部署时间,将节省的时间安排其他工作,提升 工作效率,从此时此刻开始。



博为峰旗下 51 Lest

软件测试网

ína



Jenkins 系列之 Selenium-UI 自动化 测试(2)

◆ 作者:合肥人真帅

持续集成的定义中包括自动化编译、部署、测试,上一篇我们讨论了自动化部署的 优点,以及如何实现自动化部署。这一次我们来谈一谈 UI 自动化测试。

关于自动化,前几天我看到一篇文章,文中有句话:"不懂开发的手工测试已经成 为新时代的文盲,被优化的对象"非常辣眼睛。这是一家培训机构,在招生前写的一些 铺垫,目的是为了引发读者的议论,炒作,吸引流量。虽然在言语上有些激进,但也不 是全无道理。自动化是一种趋势,任何时候都要顺势而为,不要逆势而动。雷军曾说过 "站在台风口,猪也能飞起来",这里的猪,我认为指的是实力不强的人,并不是什么 都不懂的人。所以可以翻译成"顺着大趋势走,实力不强的人也能风生水起",但前提 是你要先成为一个实力不强的人,而不是一点实力都没有。

那么说到自动化测试,相信大家都看过自动化测试金字塔。从图形结构上看,自动 化测试在 UI 层面上受益最少,但为什么在公司招聘的岗位中,UI 自动化需求量最大, 接口次之,单元测试几乎没有,和自动化收益成效相比,呈倒三角。











况下指的是人员费用和时间。为了缩短时间,就需要大量的人员,聘用的人员多,就会 导致企业支出的总费用增加。同时需求在不断变化,前端元素改动,或多或少都会对自 动化脚本造成影响,需要不断维护。所以在自动化测试中,UI自动化测试就出现了维护 成本高收益少,而且还需要大量人员。

2、收益只占 10%,为什么还有那么多公司去做 UI 自动化

UI 自动化测试收益只占 10%,为什么还有这么多公司去做 UI 自动化?因为他们是站在用户视角去考虑问题的。无论什么软件,最终都需要用户来使用。用户不会去关心软件内部是如何实现的,他们只关心是否操作简单,且功能可以满足需求。自动化测试中,UI 测试是模拟键盘和鼠标的操作,最接近用户的操作模式。所以测试结果最接近用户的感官。

举个例子,假设数据库存储的数据是正确的,如果不做 UI 测试,只做单元或接口测试,会产生什么问题?下面是接口返回的结果,与数据库进行对比,结果正确。

FILEI	U Hid	de data UKI	S AU XH	IR JS CSS	Img Medi	a Font D	oc WS Ma	nifest Oth	er							
50 ms 100 ms	150 ms	200 ms	250 ms	300 ms	350 ms	400 ms	450 ms	500 ms	550 ms -	600 ms	650 ms	700 ms	750 ms	800 ms	850 ms	900
Vame		× Hea	ders Pre	eview Re	sponse C	ookies 1	Timing									
		<pre>v data coi cuu v dat * 0 * 1 * 2 * 3 * 4 * 5 * 6 * 7 * 8 * 9 paq to errm errn</pre>	<pre>:: {count: unt: 7203 rrentPage ta: [,] 2: {id: " 1: {id: " 2: {id: " 2: {id: " 3: {id: " 5: {id: " 5: {id: " 5: {id: " 3: {id: "</pre>	: 72030, t 0 554802720 554802720 5549053410 5547905402 5547905402 5547905402 5547905402 5547905402 5547905402 554780636 331231", u 3", user_1 0 7203	1a9172c212 1a9172c211 1a91760e21 1a9176612 1a9176f13 1a9	: 7203, p 62652_887 ec86f_963 953e2_953 85d3b_576 45d89_28 0a50c_671 03d20_738 f3bf9_327 "yjl1", , mobile:	ageSize: 1 '_user", u: _user", u: _user", u: _user", u: _user", u: _user", u: _user", u: '_user", u: '_user", u: '_user", u: '_user", u: '_user", u: '_user'', u: ''user'', u: ''user''	18, currer ser_name: ser_name: ser_name: ser_name: ser_name: ser_name: ser_name: ser_name: 33391972 33311", re	"lesliel" "testi", "zyy", mc "yd", mol "gsy", mot "qc1", mc "ac1", mc "xh-1", r 87", real al_name: '	data: [, , mobile: " obile: "13 obile: "13 obile: "16 obile: "159 obile: "16 obile: "159 obile: "159 obile: "16 obile: "159 obile: "16 obile: "159 obile: "159 ob	<pre>_]} "1568912 17700000 801123456 00123456 55123456 55123456 5301123456 3301123456 3301123456 角大王", u user_rol</pre>	3456", rea 31", real_n , real_nar , real_nar ", real_nar ", real_na 6", real_ e: "[al_name: "孙 _name: "孙 ame: "猪儿 me: "沙悟浴 me: "空贝 ame: "宝贝 """""", ",,	"唐僧",} 唐空",} 爭",} 爭",} 劉袈裟",} 級精",> }		

下面是 UI 展示的效果。请注意,上一步我们验证了接口返回的数据是正确的,而 UI 层面,第三列要求显示手机号码,第四列要求显示用户姓名,明显弄反了。这是因为 后端开发人员在编写接口时,功能正确,但是前段开发人员,在拿到接口数据后,没有 把正确的数据,放到正确的位置上,导致了这个 bug 的产生。所以即便 UI 层自动化测 试的收益只有 10%,但是他是站在用户的视角去验证软件功能。所以即便收益只有 10%,但这 10%是无法替代的,就会有公司坚持去做 UI 自动化。





序号	用户名	手机号码	用户姓名
1	leslie1	唐僧	15689123456
2	test1	孙悟空	1770000001
3	zyy	猪八戒	13801123456
4	yd	沙悟净	13000123456
5	gsy	你是猴子请来的救兵吗	19656123456
6	ly	御弟哥哥	15956123456
7	qc1	宝贝袈裟	18900123456
8	xh-1	黑熊精	15301123456
9	yjl1	金角大王	为峰旗下 13339197287
10	hy	^{银角大王}	て件1399193 网

3、从软件复杂性出发

众所周知,软件开发是一件复杂的事情。同一个功能,100个程序员去做,虽然最 终的输出结果相同,但软件内部的实现方式会有100种。每个开发人员都非常熟悉自己 编写的代码,自测效率会更高,所以大部分公司都会让开发人员完成自测,而不是招聘 专业的白盒测试人员。但是考虑到自测的局限性,只有对代码要求非常高的公司,才会 招聘专业的白盒才是人员,去进行单元测试。所以对单元自动化测试人员的需求量,自 然少的多。毕竟看别人的代码是一件很恶心的事情,而且一直看,一直恶心。






二、UI 自动化的优劣势和运用

1、自动化测试的优点:

①版本迭代时,节省大量回归测试时间。例如一个产品已经测试合格,上线。在客户使用的过程中,不断有需求反馈给产品部。要对其进行不断的细节优化和新功能的开发。但这些功能都相对独立,在优化细节过程中也不会影响基础产品的功能。此时对基础功能模块进行自动化测试脚本的开发,以后版本迭代时,只需要测试新模块,老功能交给自动化测试,可以节省大量时间。例如下图,第一张图的功能已经测试合格,第二张图新增了一个手机号查询功能,那么可以手工测试手机号查询功能,其余老功能交给自动化测试

角色:	请选择	> 清縮入用户;	名姓名 Q	当前功能测试已经合格 编写的自动化测试脚本,通过审核							
	序号	用户名	姓名	所属单位	职位	登录次数	创建时间	操作			
			唐僧	测试	测试						
		test102801	孙悟空	测试单位	测试职位						
			猪八戒	测试							
				测试所属单位	测试职位						
				测试所屈单位	测试职位		2019-10-28 15:30:46				
			玉皇大帝	测试所属单位	测试职位						
			太乙真人								
			金吒	测试							
			哪吒	鼓楼							

共72条 < 1 2 3 ···· 8 > 10条页 × 跳至 1 页

角色: 请选择				手工观试新增的功能,其余功能的验证交给自动化测试							
序号	用户名	姓名	手机号	所属单位	职位	登录次数	创建时间	操作			
1		唐僧	15205609541								
2		孙悟空		测试单位	测试职位						
3											
4				测试听属单位	測试职位						
5				测试所属单位	測試职位						
6		玉皇大帝		測试所屬单位	測试职位						
7		太乙真人									
8											
9		哪吒		鼓楼							
10											
			共72条 <	1 2 3 8	> 10 条页 ~	跳 至 1 页					

②更好地利用资源。这个也很好理解,我从两个方面来分析一下。第一,大部分公





司会给每位测试人员配备一台电脑(有些公司会配备2台或3台),那么为了缩短交付 周期,可以在白天的时候进行新功能的手工测试,在下班以后,运行自动化测试,第二 天来查看测试结果即可,并且在自动化测试过程中,如果出现故障导致无法继续执行, 可以让其发送邮件,当测试人员收到邮件后,在家时可以进行远程登录(登录公司的机 器),排查错误原因,让自动化测试继续执行下去;第二,在评估工作计划的时候,或 多或少都会因为各种原因,导致评估不太准确,一般在时间的评估上,误差在±10%以 内就算是精准了。有些公司缺乏专业的项目经理,而是让开发经理兼职这个岗位的工作 任务,导致误差在±30%,也有很多。时间上出现混乱,必然有部分真空期的存在。比 如测试人员原定30天写完测试用例,评审,修改,然后执行。但是由于功能开发还没 完成,需要推迟5天。这5天测试人员可以优化自动化测试脚本,充分利用这5天,不 让时间浪费。若没有自动化测试的公司,可能会安排员工学习新的测试技术,但笔者从 实际工作经验来看,基本上是没有效果的。

③测试具有一致性。例如下图,是一个新增用户的对话框,若测试用例中有一个操 作步骤写成"点击关闭按钮",但是这个页面有2个关闭按钮,执行这个测试用例时, 有些人会点击下面的关闭,有些人会点击上面的关闭,可能会导致相同的测试用例,不 同的测试结果。

用户信息	×
* 姓名:	1
请输入姓名	
* 手机号: 「请输入手机号	
* 用户角色:	都是关闭按钮
请选择用户角色	~
* 职位:	
请选择	~
	博为峰旗下 5760年1月10日 5761 5761 5761 5761 5761 5761 5761 5761





若是自动化测试,在编写脚本的时候,选择了下面的关闭按钮,那么以后每次执行时,都会点击下面的关闭按钮。那么可以保证每次测试步骤都是相同的。至于右上角的 关闭按钮,可以在优化测试用例时,再增加一条用例,去测试这个按钮。

④增加软件测试可信度。例如一个产品有 500 个功能点,且已经测试合格。最新的一个版本,只新增了一个功能点。站在公司的角度去看,老板不会给你太多时间去测试,你的工作重心只能放在新功能上。当测试结束后,"测的怎么样,合格了吗?",这个问题是不是很难回答?因为你只测了新功能,根本没时间测试老功能,或者老功能只是走马观花的过了一遍。你不确定新功能是否对老功能产生了多大影响。你不敢回答这个问题。所以如果老功能有自动化测试过了一遍,达到高百分比覆盖(不会达到 100%的),你一定会很自信的回答这个问题,增加测试结果的可信度。即便有 1,2 个遗漏的,也是情理之中的,因为软件 bug 是不可能被全部找到的,只要控制在可接受范围内就行了。

2、自动化测试的缺点:

①不能取代手工测试。例如界面的美观,友好性就无法进行自动化测试。下图中可以看出,功能并无错误,但是页面中间有一大块空白,布局较丑,这类问题,自动化测试就无法实现。

序号	名称	类型	公交路线	公交时间	操作
1	XX公交公司	快速公交	线路04	2019-12-30 00:00:00开始	编辑 查看 删除
2	XX公交公司	城市公交	线路03	2019-12-30 00:00:00开始	编辑 查看 删除
3	XX公交公司	城市公交	线路01	2020-01-03 00:00:00开始	编辑 查看 删除
4	XX公交公司	区域公交	线路02	2020-01-03 00:00:00开始	编辑 查看 删除

②手工测试比自动测试发现的缺陷更多,自动化测试不容易发现新的 BUG。自动 化测试是测试已经合格的功能,是为了防止新功能对老功能产生影响而未被发现。所以 已经合格的功能即便被新功能影响到了,bug也不会太多,所以自动化测试发现新 bug 会比手工测试要少。自动化测试的主要目的,还是防止遗漏。

③对测试质量的依赖性极大。若老功能遗留的 bug 较多,且漏测较多,测试质量不高,自动化测试是执行不下去的。

④自动化测试无法提高测试的有效性。

3、不适合自动化测试用例的情况:





①定制化项目。所有功能都只为该项目的客户定制化服务,这个项目的功能,不能 在其他项目上复用。例如,你去商场买衣服,你的身高1米8,体重也很匀称,你可以 找到很多大小合适的衣服,相同款型,相同大小的衣服有很多。而姚明,身高2米27, 他在市场就就买不到衣服,必须定制。为他定做的款型,只能卖给他,即便多生产一件 相同的衣服出来,市场上也不会有人买。为他定制的设计稿,也无法复用。这类项目, 就不适合自动化测试。

②项目周期很短的项目。项目周期很短,测试周期更短,就不值得花精力去投资自动化测试,好不容易建立起的测试脚本,不能得到重复的利用是不现实的。

③业务规则复杂的对象。业务复杂,会导致测试用例步骤较多。若写成自动化测试 脚本,实现难度很大,有一种捡了芝麻丢了西瓜的感觉。

④美观、声音、易用性,人的感观方面的测试。例如音乐软件在测试音频播放时, 整首音乐中出现过杂音,或者视频软件在播放视频时出现了不明显的卡顿,画质清晰度等,也无法使用自动化测试。

③测试很少运行。例如一个产品虽然功能测试合格,但未来几年没有迭代的打算。 若对这样的产品进行自动化脚本开发。实际上也是在浪费成本,因为软件更新换代那么 快。几年都不打算迭代的产品,可能有被淘汰的风险。若开发完的自动化测试脚本,没 有长期运行,那将毫无意义。

●软件不稳定。会由于这些不稳定因素导致自动化测试失败。自动化测试的目的就是不断运行开发出来的测试脚本,若始终因为各种因素无法运行,开发自动化测试脚本,所用的时间也就等于是浪费掉了。

⑦涉及物理交互。例如在宾馆、饭店、网吧都很常见的充电宝,后台是靠软件服务 支撑的,而前端是靠硬件里,嵌入式开发的程序来运行的。这样前端操作影响后端数 据,就无法进行自动化测试了。

4、适合自动化测试的情况:

①产品或产品型项目。每次迭代只改进少量的功能,没有改动过的功能需要反复测试。这部分测试完全可以让自动化测试来承担。新增的功能在测试合格后,也可以脚本化,加入自动化测试。

②增量式开发、持续集成项目。例如小米产品的服务功能,一开始手机后台服务程





序,只为手机服务; 电视后台服务的程序只为电视服务; 然后电视后端服务和手机后端 都测试合格,产品有了一定的成熟性。手机和电视有了多屏互动,手机红外遥控等功 能,多个产品交互的功能开发,是否会影响单个产品的稳定功能? 我们可以把电视、电 脑、手表、电话等等单个产品,合在一起看成一个增量式的,持续集成的智能家居项 目,那么单个产品在测试合格后,很适合自动化测试,我们只需要手动测试各个产品之 间的交互。

③能够自动编译、自动发布的系统。例如我下面要讲的 jenkins

④回归测试。回归测试是自动化测试的强项,它能够很好的验证你是否引入了新的 缺陷,老的缺陷是否修改过来了。

⑤需要频繁进行相同的测试步骤。对于人来说,总是执行相同的动作,就会产生厌烦,乏累的感觉。例如,从1写到600,听上去很简单,但是没有几个人可以完全写对,这个骗局,利用的就是人无法在长期进行重复性动作时,始终保持注意力高度集中(百度一下就知道了)。而机器则不会,只要有电,就能一直按照预先设定的步骤运行。我之前就遇到一个bug,就是上面那张公交车的图,在新增线路后,页面不会自动刷新,需要手动刷新才能看到新增的数据,这实际上属于一个优化类的问题。但是我无意间发现,若不进行手动刷新,新增第28条数据的时候,就会报服务器错误。这个bug 修复后,我也针对这个bug 编写了一个脚本,就是新增路线,我设置成重复执行100次。这种问题,回归起来,很费人力,浪费时间还影响心情,交给自动化就好了。

综上所述,自动化完成不了的,手工测试都能弥补,两者有效的结合是测试质量保证的关键。

三、UI自动化核心是元素识别

Selenium 官方网站在使用教程中说到,定位方式有 8 种,分别是 class name、css selector、id、name、link text、partial link text、tag name、xpath







Q, Search X	Element selection	Element selection strategies					
	There are eight unrere	ות בסווריות כופרווסית וסכמונטה אם מרפשים זה ארפטטרואפר.					
Getting started	Locator	Description					
Introduction	class name	Locates elements whose class name contains the search value (compound class names are not permitted)					
Selenium installation	css selector	Locates elements matching a CSS selector					
Getting started with WebDriver	id	Locates elements whose ID attribute matches the search value					
Browsers	name	Locates elements whose NAME attribute matches the search value					
Third party drivers and plugins	link text	Locates anchor elements whose visible text matches the search value					
Performing actions on the AUT*	partial link text	Locates anchor elements whose visible text matches the search value					
WahDrivar 🗸	tag name	Locates elements whose tag name matches the search value	>				
Remote WebDriver	xpath	Locates elements matching an XPath expression					
Guidelines	T						
Worst practices	Tips on using selec	cors					
Grid 🗸	in general, if HTML IDs that comes with comp	are available, unique, and consistently predictable, they are the preferred method for locating an element on a page. They tend to work very quickly, and forego much processing licated DOM traversals.					
Driver idiosyncrasies	If unique IDs are unave	alable, a well-written CSS selector is the preferred method of locating an element. XPath works as well as CSS selectors, but the syntax is complicated and frequently difficult to					
Support packages	debug. Though XPath	selectors are very flexible, they are typically not performance tested by browser vendors and tend to be quite slow.					
Legacy	Selection strategies ba	sed on link text and partial link text have drawbacks in that they only work on link elements. Additionally, they call down to XPath selectors internally in WebDriver.	_				
Front matter	Tag name can be a dar	ngerous way to locate elements. There are frequently multiple elements of the same tag present on the page. This is mostly useful when calling the findElements(B)/method which	nn				

并且告诉我们在选择使用定位方法的时候,建议遵循以下原则:

1.若 id 和 name 在 html 中是唯一的,则优先使用这 2 种。

2.使用 css 或 xpath,他们都很灵活,但语法复杂。Xpath 性能应该是最慢的。

3.link text、partial link text 缺点在于只对连接元素起作用。

4.class name 不支持复合类名的元素。

5.tag name 是危险的方法,因为一个页面上有很多相同标签的元素。

定位举例:



要定位到上图选中的这一行, 8种定位方式的 java 代码如下:

// id 定位, 一般情况下元素 id 是唯一的, 可以使用这种方法 driver.findElement(By.id("topic"));



《51 测试天地》五十六(上)

www.51testing.com



// name 定位, 一般情况下元素 name 是唯一的, 可以使用这种方法 driver.findElement(By.name("topic"));

// cssSelector 定位,可以使用相对路径或绝对路径 driver.findElement(By.cssSelector("body > div.global > div.layout.qq-main.cf > div.col.col-2.fl > div.mod.m-topic > div.hd.cf > h2 > a"));

// Xpath 定位,可以使用相对路径或绝对路径 driver.findElement(By.xpath("/html/body/div[1]/div[5]/div[2]/div[1]/ div[1]/h2/a"));

// className 定位, 一般情况下一个页面会有多个相同 className 的元素 driver.findElement(By.className("side_ad"));

// linkText 定位,只能定位链接元素,文本内容必须完全匹配 driver.findElement(By.linkText("今日话题"));

// partialLinkText 定位,只能定位链接元素,提供部分的文本内容,就可以定位 driver.findElement(By.partialLinkText("话题"));

// tagName 定位,会定位到所有 a 标签的元素 driver.findElement(By.tagName("a"));

若某种方法定位到多个元素,则会返回第一个元素。其中 xpath 和 cssSelector 上述 代码使用的是绝对路径,也可以使用相对路径,以 xpath 为例, css 类似,代码如下:

// xpath 定位,使用标签名和元素属性定位 driver.findElement(By.xpath("//a[@id='topic']"));

// xpath 定位,仅使用元素属性定位 driver.findElement(By.xpath("//*[@id='topic']"));

// xpath 定位,使用多个元素属性定位 driver.findElement(By.xpath("//*[@id='topic' and @bosszone='jrht_logo']"));

// xpath 定位,使用元素文本内容定位 driver.findElement(By.xpath("//*[text()='今日话题']"));

// xpath 定位,使用元素属性定位,支持模糊查找





driver.findElement(By.xpath("//*[contains(@class,'side')]"));

// xpath 定位,定位其他元素,查找相对位置
// 假设上图选中的这一行,没有 id="topic" 这个属性
driver.findElement(By.xpath("//div[@id='news']/div[2]/div[1]/h2/a"));

四、Jenkins 让 Selenium-UI 自动化测试如虎添翼

Selenium 自动化测试,对于脚本的编写,语言只是一个载体,我熟悉 java,所以用 java 编写。Selenium 支持 Ruby, C#, java, python, javaScript 语言。在官网下载对应的 包就行了。

Selenium Client & WebDriver Language Bindings								
In order to create scripts that interact with the Selenium Server (I	In order to create scripts that interact with the Selenium Server (Remote WebDriver) or create local Selenium WebDriver scripts, you need to make use of language-specific client drivers.							
While language bindings for other languages exist, these are th	e core ones that are supported by the main project hos	ted on GitHub.						
LANGUAGE	VERSION	RELEASE DATE	LINKS					
Ruby	3.142.6	October 04, 2019	Download Changelog API Docs					
JavaScript	4.0.0-alpha.5	September 08, 2019	Download Changelog API Docs					
Java	3.141.59	November 14, 2018	Download Changelog API Docs					
Python	3.141.0	November 01, 2018	Download Changelog API Docs					
C#	3.14.0	August 02, 2018	Download Changelog API Docs					

编写完测试脚本,可以将它集成到 jenkins 上,定时运行。





博为峰旗下 TPS

www.51testing.com



我使用的是 maven+testng+svn 进行管理。具体操作如下:

1.首先写好 testng.xml, pom.xml, 作为运行的对象



《51 测试天地》五十六(上)

www.51testing.com



2.在 jenkins 中安装 Maven Intergration plugin 插件



博为峰旗下

51 testing

软件测试网



www.51testing.com

用		版本	上一个安装的版本	卸载
ø	Apache Http:Components Client 4 x API Bundles Apache Http:Components Client 4 x and allows it to be used by Jenkins plugins.	<u>4.5.5-</u> <u>3.0</u>		
¢.	Command Agent Launcher Plugin Allows agents to be launched using a specified command.	<u>1.3</u>		
¢.	Config File Provider Plugin Ability to provide configuration files (e.g. settings xml for maven, XML, groovy, custom files,) loaded through the UI which will be copied to the job workspace.	<u>3.6.2</u>		
1	Javadoc Plugin	<u>1.5</u>		
1	Usch dependency Jenkins plugin that brings the JSch library as a plugin dependency, and provides an SSHAuthenticatorFactory for using JSch with the ssh-credentials plugin.	<u>0.1.54.</u> ;	2	
¥.	Juni Allows JUnit-format test results to be published.	<u>1.26.1</u>		
2	Maller This plugin allows you to configure email notifications for build results	<u>1.23</u>		
	Maven Integration plugin This plug-in provides, for better and for worse, a deep integration of Jenkins and Maven: Automatic triggers between projects depending on SNAPSHOTs, automated configuration of various Jenkins public terms (Junt,).	<u>3.4</u>		卸载
1	Oracle Java SE Development Kit Installer Allows the Oracle Java SE Development Kit (JDK) to be installed via download from Oracle's website.	<u>1.2</u>		
1	Token Macro	2.5		

3.在 jenkins 中,系统管理→全局工具配置中设置 maven 路径

Jenkins > 全局工具配置	
Gradie	Add Git •
Gradie 安装	新聞 Grade
Ant	2018 Ling Hand X40,214
Ant 安陵	1612 Ant
	悉记载安 InAT 前进
Maven	
Maven 安装	新聞 Mayen
	Maven
	Name mayen3.6
	MAVEN_HOME Diapache-maven-3.6.0
	0 Baga
	Bitt Mayen
	張環 Maven
	系统下Maven安装则表

4.新建任务,选择构建一个 maven 项目







输入一个任务名称
» 论编项
构建一个自由风格的软件项目 这是Jenkins的主要功能 Jenkins将会结合任何SCM和任何构建系统来构建你的项目, 甚至可以构建软件以外的系统.
构建一个maven项目 构建一个maven项目 Jenkins利用你的POM文件,这样可以大大减轻构建配置.
流水线 精心地组织—个可以长期运行在多个节点上的任务。适用于构建流水线(更加正式地应当称为工作流),增加或者组织难以
External Job This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is de
构建一个多配置项目 适用于多配置项目,例如多环境测试,平台指定构建,等等.
GitHub 组织 扫描一个 GitHub 组织(或者个人账户)的所有仓库来匹配已定义的标记。
多分支流水线 根据一个SCM仓库中检测到的分支创建一系列流水线。
文件夹 创建一个可以嵌套存储的容器。利用它可以进行分组。 视图仅仅是一个过滤器,而文件夹则是一个独立的命名空间, 因此
如果你想根据一个已经存在的任务创建,可以使用这个选项
复制 输入自动完成

5.配置 General 内容,在上一篇《Jenkins 系列之自动化部署(1)》中,我们已经详 细介绍过如何配置远程运行节点,这里不再赘述。需要注意的是,UI自动化测试,若在 本机构建可能会出现后台运行的现象,为了测试的有效性,还是建议找一台独立的,环 境干净的机器,让UI测试在前台进行。

	油还								
		[纯文本] 预览							
	GitHub 項目								
	This build requires I	ockable resources							
	Throttle builds								
	□ 去开旧的构建								
	□ 在必要的时候并发格	達							
	☑ 限制项目的运行节点								
	标签表达式	234							
		Label 234 is se	rviced by 1 node. P	ermissions or othe	r restrictions prov	ided by plugins	may prevent this	job from running on those nodes.	
									200 B
6	Ħ								
0.你妈自己	土								





同

《51 测试天地》五十六(上) www.51testing.com

◎ 无			
Git			
Subversion			e
Modules			
	Repository URL	https://192.168.1.77:8666/svn/Auto/ 源码地址	, e
	Credentials	admin/****** 登录账户	
	Local module directory		0
	Repository depth	Infinity •	0
	Ignore externals	ø	0
	Cancel process on externals fail	8	0
	Add module		
Additional Credentials	Add additional credentials		0
Check-out Strategy	Use 'svn update' as much as possibl	e	•
	Use 'svn update' whenever possible	, making the build faster. But this causes the artifacts from the previous build to remain when a new build starts. (博为峰旗下)	
Quiet check-out		51test	Í Nº

上一篇详细介绍了 git 管理源码的配置,这一篇使用的 svn 管理源码,配置基本相

7.构建触发器,每天执行一次,在10点整运行

构建触发器				
Build whenever a S	SNAPSHOT dependency is built	0		
	Schedule build when some upstream has no successful builds	0		
触觉逻程构建(例如,使用脚本)				
GitHub hook trigge	GiHub hook trigger for GITScm polling			
■ 其他工程均違后触发				
☑ 定时构建		0		
日程表	H 10**1-6			
■ 轮询 SCM		0		

8.测试前,清理工作路径下,上一次测试结果(若需要保留,则可以不清理)

Pre Steps	_
执行 Windows 批处理命令	×
ŵ ☆ cd /d D:\Jenkins\workspace\Auto dol report.tit del report.jpg	
	R
	商级
Add pre-build step -	

第一句是进入工作环境文件夹

后面2句是需要删除的文件。

9.编写构建命令







Root POM	pom.xml	0
Goals and options	clean test	0
		高级
D.测试结束 Post Steps	束后,需要执行的命令可以放在 PostSteps 中执行	
·	Run only if build succeeds Run only if build succeeds or is unstable Run regardless of build result should the post-build steps run only for successful builds, etc.	
执行 Windows 批处题	l@\$	×
mb cd /d d:\jen javac -cp. : 3.141.59.jar 3.141.59.jar java -cp. : java -cp. : 3.141.59.jar java -cp. : 3.141.59.jar java -cp. : 3.141.59.jar	<pre>intsScreen ``\jurPackage\selenium-java-0.141.59\java-0lient-0.1.0.jav.D:\jarPackage\selenium-java-3.141.59\selenium-server-standalone-3.141.59\java-0lient-0.1.0.jav.D:\jarPackage\selenium-java-3.141.59\java-0.141.59\java-</pre>	jar jar
		级
参阅 <u>可用环境</u>	8	

我这里是实现了,将本地测试完成后的测试报告打开,然后截图,转换成 base64 编码保存到本地 txt 文档。(目的是为了发送邮件的时候,把图片帖进去)源码如下

```
public class Photo {
  public static void main(String[] args) {
    WebDriver driver = new ChromeDriver();
    System.setProperty("webdriver.chrome.driver",
         "C:\\Program Files (x86)\\Google\\Chrome\\"
         + "Application\\chromedriver.exe");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.get("D:\\Jenkins\\workspace\\Auto\\report.html");
    driver.manage().window().maximize();
    try {
       File scrFile = ((RemoteWebDriver) driver).getScreenshotAs(
            OutputType.FILE);
       Files.copy(scrFile, new File("D:\\Jenkins\\workspace\\Auto"
            + "\\report.jpg"));
       // 图片转换成 base64
       InputStream in = new FileInputStream("D:\\Jenkins\\workspace"
           + "\\Auto\\report.jpg");
       byte[] data = null;
       data = new byte[in.available()];
       in.read(data);
       in.close();
```





启

General	源码管理	构建触发器	构建环境	Pre Steps	Build	Post Steps	构建设置	构建后操作	
构建后	操作								

先说邮件发送,上一篇邮件如何发送已经说过了,但是这次我们加入了测试报告的 截图



《51 测试天地》五十六(上)

www.51testing.com



Default Content	html
	<html></html>
	<head></head>
	<meta charset="utf-8"/>
	<body leftmargin="8" marginheight="4" marginwidth="0" offset="0" topmargin="8"></body>
	构建信息
	<hr align="center" size="2" width="100%"/>
	
	<div style="font-size:18px"></div>
	
	
	
	
	◇I>测试详情: 刀附件report.html
	

因为我们发送的邮件是 html 格式的,所以除了文字,还可以放入图片。图片放入 html 中有两种方式,第一种是链接,由于我们的图片是保存在本地的,以链接的方式加 入到 html 中,发送邮件后,是无法查看的。只能用第二种方式,将图片转换成 base64 编码,将编码放入 html 中,就不再受到网络的限制了。我的第 10 步操作就是将图片, 转换成 base64 编码。report.txt 打开后如下图

🧊 report.txt - 记事本 🛛 🗕 🚽 \times 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H) _{ ^ IU9gAEAAAbQ6D37hJo1W4v9jQEAAKAbH AAAAAAAICGEAADAAAAAAANIQAGAAAA/ UoaAAAAAAAA1k5aULuSEDgr+C0jWJ6en1 P30HsAAAABQwKt//OWYfPyxWLhwobQ5hi segsZfu6dV8D1mfdfL7UWbg3G/fEwAAV} <

12.接下来让 jenkins 任务,可以直接打开测试报告。我们需要安装一个插件 HTML Publisher。





	Activate Pe	erfPublisher for this	project	
	Aggregate	downstream test re	sults	
	Deploy arti	ifacts to Maven repo	sitory	
	Groovy Po	stbuild		
	Publish HT	ML reports		
	Publish Te	stNG Results		
	归档成品			
	构建其他工	[程		
	记录文件的	的指纹用于追踪		
	Git Publish	1er		
	Editable E	mail Notification		
	Editable Er	mail Notification Ten	nplates	
	Send build	l artifacts over SSH		
	Set GitHub	o commit status (uni	versal)	
	Set build s	tatus on GitHub con	nmit [deprecated]	
	Delete wor	rkspace when build i	is done	
	增加构建后操	作步骤 ▼		
Publish H	TML reports			
Reports				
	HTML directory to	o archive D:\Jenki	ns\workspace\Auto	
	Index page[s]	report.ht	ml	
	Index page title[s	i] (Optional)		
	Report title	HTML R	eport	
			博为峰旗下)
	新增		- D IESE	Пg
			软件测试	式 网

安装完插件,就可以在构建后的步骤中,找到 Publish HTML reports 选项了。然后添加配置,第一行代表本地测试报告的路径,第二行是测试报告索引文件。添加了这个功能,在项目页面就会有一个按钮,允许我们直接查看测试报告







┢ 返回面板		Mayon project
2、状态		waven project
🥜 修改记录		2019-10-10创建项目,集成登录功能测试 2019-10-11集成首页数据和超链接验证
11年空间		
> 立即构建		
🔇 删除 Maven project		
🗱 配置		HTML Report
模块		
Email Template Testing		
HTML Report		量新修改
▶ 重命名		量新测试结果 (没有失败)
Build History	构建历史 📼	
find	х	最新测试结果 (没有失败)
● <u>#52</u> 2019-11-22 上午11:45		相关链接
🥘 <u>#51</u> 2019-11-21 上午11:45		- 吴近—·次构理(#50) 2 小时 1/ 公文前

13.由于 jenkins 禁用了第三方页面的样式,所以我们每次构建完成以后,最好都执行一次放开权限的命令。添加 Groovy Script。内容如下

System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")

Groovy Postbuild		×
Groovy Script	System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")	
Additional classpath	Use Groovy Sandbox Add entry	Ø
If the script fails:	Do nothing	

所有配置都完成了。保存,返回项目页面,构建一次。确保配置正确,可以得到预 期的结果。否则根据控制台输出的日志,进行错误排查。

🔮 Jenkins		4 风前派 ● 1注版
Jenkins → webUI →		<u> た 注目 時 新 </u>
★ 返回面板	Maven projectwebUI	
● 体改记录	2019-09-02台議道西日, 每周1-6中午12/fi01执行, 集成證章功能則成 2019-09-06集成首页功能到成 2019-09-06集成首页方面影响成	
 ■ 工作空间 	2019-09-2016年7月19日 2019-09-2016年1日 2019-10-09国政 音响功能形式 2019-10-09国政 音响功能形式	
S 删除 Maven project 参 配置	2019-10-2018-01-01-01-01-01-01-01-01-01-01-01-01-01-	每次集成的新功能,可以在这里添加描述,作为日本
🧱 欄块		entroise Singledering
IS HTML Report ≥重命名	Lat这里查看测试报告	测试结果趋势
Build History 約建历史 =	Turk	80 contra 100 contra
□ 1100 点击这里,然后点击查看控制台, 可以查看测试过程中的日志 2019-11-22 下年12.01	量 最新建立	3
● <u>#169</u> 2019-11-21下午12:01	最新期试验器(没有失敗)	
● 1100 2019-11-20 P+912-01 ● 1167 2019-11-19 下午12:01	最新到试结果(没有失败)	Sector and the sector of the s

这是测试报告的页面。由于 testng 提供的测试报告样式太丑, 所以我使用了 ztest 进





行美化, github 源码地址是 https://github.com/zhangfei19841004/ztest

这套模版最后一次更新是在 2019 年 7 月 31 日,目前还有些简单的 bug,大家可以 自行修复。也可以根据自己项目的需要,在测试报告上添加信息,做二次开发。

Back to Ga	nsuGrid web	JI report					Zip
			测试报告				
服伤汇总							
INTIAL-ID			·				
	用制名称:	WebUI自动化期试	失敗	测试用例运行结果			
	用例总数:	23					
	用何间通过:	23		失败			
	用例失数:	0					
	FEIMINUSE:	0					
	开始时间:	2019-11-22 12:02:00.469					
	运行时间:	1213818ms		127)			
详细数据							
-							_
36.03	ŧ:	· ···································		109	服: 23 成功: 23	失敗:0	Mit: 0
编号	测试类		测试方法	用例描述	耗时	结果	操作
1		.Login_TestCase	errorPassWord	用户名存在, 密码错误登录	7850ms	成功	展开
2		.Login_TestCase	errorUserName	用户名不存在, 登录	6384ms	成功	展开
3		.Login_TestCase	errorVerCode	输入错误验证码登录	6235ms	成功	展开
4		.Login_TestCase	noPassWord	不输入密码登录	6239ms	成功	展开
5		.Login_TestCase	noPowerLogin	无权限登录	6435ms	成功	展开
6		Login_TestCase	noUserName	不输入用户名登录	6260ms	成功	最开
7		Login_TestCase	noVerCode	不输入验证码登录	6675ms	成功	展开
8		Login_TestCase	normalLogin	正常登录	7355ms	成功	展开
9		.LogOut_TestCase	logOut	退出测试	7999ms	成功	嚴开
10		.HomePage_TestCase	homePageTest	首页校验	63339ms	成功	展开
11		:_TestCase	st	查看」	47662ms	成功	展开
12		_TestCase	LookTest	查看.	41842ms	成功	展开
13		B_TestCase	Test	新增失败	17524ms	成功	扁开
14		TestCase	Jest		234364ms	成功	扁开
15		TestCase	earchTest	查询测试	96980ms	成功	展开
16		_TestCase	JSearchTest	查询测试	51589ms	成功	展开
17		.Search_TestCase	SearchTest	埔 为修施	45441ms	196070	#07
18		_TestCase	Test	#AU 将 7 叫手 供	245049ms	10500	###
19		_TestCase	Test		66419ms	Dk.VJ	
20		restLase	AddTast		68277mc	ist th	
21		TestCase	Addlest		75301mc	NDTh -	
22			DeleteTest	新小小	37773#5	57171	
2.5		_iescuse	Deteceresc		5777385	MON J	

图中运行这一列如果是失败,点击展开按钮可以查看到报错信息。

下面是控制台输入的日志





《51 测试天地》五十六(上)

www.51testing.com

Allight Le 状态集 変更记录 控制台输出 型 文本方式重看 编辑编译信息 删除本次生成 Tag this build Test Result	空記 注記 上市理認品が 在远程节点 234 上构建 (yj12) 在工作空间 D:\jenkins\workspace webUI 中 Updating https://1 0/syn/Auto/ at revision '2019-11-22712:01:52.265 +0800'quiet Using sole credentials .// (43) VisualSVN Server' At revision 133 No changes for https:// No chang
変更記录 控制台輸出 型 文本方式査看 編編编译信息 删除本次生成 Tag this build Test Result	由在时器启动 在远程节点 234 上构建(yj12)在工作空间 D:\jenkins\workspacewebUI 中 Updating https://1 《/syn/Auto/at revision '2019-11-22T12:01:52.265 +0800' —quiet Using sole credentials/
変更记录 控制台输出 型 文本方式重看 编辑编译信息 删除本次生成 Tag this build Test Result	由歪时器启动 在远程节点 <u>234</u> 上构建 (yj12) 在工作空间 D:\jenkins\workspacewebUI 中 Updating <u>https://1</u> 3/ <u>syn/Auto/</u> at revision '2019-11-22T12:01:52.265 +0800' —quiet Using sole credentials
控制台输出 文本方式查看 编辑编译信息 删除本次生成 Tag this build Test Result	mierisonewy 社政程序点 234 上构建 (yj12) 在工作空间 D:\jenkins\workspacewebUI 中 Updating <u>https://1 3/syn/Auto/</u> at revision '2019-11-22T12:01:52.265 +0800' —quiet Using sole credentials 3/****** in reala '(<u>https://i 43</u>) VisualSVN Server' At revision 133 No changes for <u>https://i /syn/Auto/</u> since the previous build No easils were triggered.
文本方式查看 编辑编译信息 删除本次生成 Tag this build Test Result	Updating https://1 3/svn/Auto/ at revision '2019-11-22T12:01:52.265 40800' —quiet Using sole credentials 3/****** in reals '< <u>https://1</u> 43> VisualSVN Server' At revision 133 No changes for https://2 /svn/Auto/ since the previous build No easils were triggered.
编辑编译信息 删除本次生成 Tag this build Test Result	Using sole credentials 3/****** in reals ' <u>https:// 43</u> > VisualSVN Server' At revision 133 No changes for <u>https:// /svn/Auto/</u> since the previous build No emails were triggered.
翻除本次生成 Tag this build Test Result	No changes for https:// /svn/Auto/ since the previous build No easils were triggered.
Tag this build Test Result	No enails were triggered.
Test Result	
Test Result	<pre>[webUI] \$ cmd /c call C:\Users\admin\AppData\Local\Temp\jenkins7393518035261684963.bat</pre>
	D:\jenkins\workspace\CwebUI>cd /d D:\Jenkins\workspace\webUI
Redeploy Artifacts	
查看指纹	D:\jenkins\workspace\ =bUI>del report.txt
前一次构建	D:\jenkins\workspace\ebUI>del report.jpg
	D:\jenkins\workspace\'webUI>exit 0
	Parsing POMs
	Modules changed, recalculating dependency graph Extablished TTP socket on 61864
	maven35-agent, jar already up to date
	maven35-interceptor.jar already up to date
	riorod who lurational arrivation array array
	d:\ienkinsScreen\ .webUI>exit 0
	[htmlpublisher] Archiving HTML reports
	[htmlpublisher] Archiving at PROJECT level D:\Jenkins\workspace\
	Email was triggered for: Always
	Sending email for trigger. Always
	bonding owder for trigger, newajo

一套完整的 Selenium-UI 自动化测试持续集成方案介绍完毕。各位小伙伴是不是有 动手的冲动了?既然已经坚持看完了,那就动手试试看吧。





Jenkins 系列之 Jmeter-API 自动化 测试(3)

◆ 作者: 合肥人真帅

自动化测试技术的火热除了表现在 UI 层,还表现在接口层。能实现接口测试的工具和框架有很多,例如 Jmeter、Postman、eoLinker、Request+Unittest 等等。每一种工具或框架都有优缺点。选择哪一种,需要根据自身和公司的实际情况来决定。

一、选型

eoLinker 纯图形界面、零代码量使用方便,支持在浏览器中安装插件,且提供了测试用例的管理,可以自动化执行所有用例。但网页版所有数据都是保存在服务商的数据 库中,若公司涉及到保密问题,则不可以使用网页版,且纯图形界面不支持上传附件这 类接口的测试。虽然开源版支持本地部署,但是功能非常的少。全功能本地部署,费用 昂贵。

∠ → C (i) enlinker	com/			~ Q :
	icom,		請 工作空间(会紀人賞約 ~	↓ · · · · · · · · · · · · · · · · · · ·
🔓 首页		测试用例		
9 API快速测试		+ 新社 - 泉、人员权限 三、秋量操作		
API 管理与自动化测试	^			
API 研发管理		用钢集名称(根目录)	最后编辑时间	操作
API 自动化测试		test	2019-11-18 15:10:59	新伝施打开 編編 更多〜
数据库研发管理		测试	2018-11-09 17:30:57	新物語打开 編編 更多〜
🔀 测试用例与计划管理	^	共2条记录		
测试用例				捕为修旗下
测试的计划				
A. 人员管理				neseno
♀ 公共资源管理	~			
圖 工作空间管理	~			软件测试网





○ 田田: 2.885 ④ 小田藤田谷秋秋東 ▲ A5: 15 ● 田田: 2.885 ● 田: 2.885	위 Studio API研友管理与自动化测试	Data	base Builder 数据库管理	Ŧ	e	
● 确认订单信 ④ 由达对 ④ 中成丁 与定日 ● 小 ● 方 ● ●	 ● 已开通:免费版 每个功能都单独收费 ▲ 人员: 1/5 ■ 到期时间:永久 梁 续费/升级 Q 人员 ••• 更多 ▼ 		日开通: 免费版 人员: 1/3 到期时间: 永久 9 续费/升级	2 人员	更多 ∨	
by Series By Series Difference Diffe	① 确认订单信息	② 育	前往支付 ③ 完成证	「单		
5 病以支付即默认您已阅读并同意签署《服务协议》 前往支付 购买后, AMS 产品可绑定 5个成员 前往支付 购买时长 按年购买 按月购买 工年 2年 3年 购买之后可以开具发票 按年购买价格: ¥27/人/月	填写购买信息 <u>购买类型</u> 企业版 当前产品信息 产品名称 AMS 免费版 已绑定人数 1	~	 确认订单信息 购买类型 产品版本 最大使用人数 购买时长 单价 折扣 总价 过期时间 	新购 企业版 5人 1年 ¥27/人/月 无 ¥1620.00 2020年11月23	按用户数量收费 1个用户,1年長 7日	^表 2324元
购买后,AMS产品可绑定 5个成员 前往支付 购买时长 购买之后可以开具发票 按年购买 按月购买 3年 按年购买价格: ¥27/人/月 第	产nn使用入数 5	~ ~	确认支付即默认您已间	3读并同意签署 《	服务协议》	
购买时长 购买之后可以开具发票 按年购买 按月购买 2年 1年 2年 按年购买价格: ¥27/人/月	购买后,AMS 产品可绑定 5个成员			前往支付		
	购买时长 按年购买 按月购买 1年 2年 3年		购买之后可以开具发票			
	按年购买价格:¥27/人/月					

Request+Unittest 是测试框架, 纯代码级, python 语言。应对各种需求, 开发起来都 很灵活。但是相对于图形化的工具而已, 对人员水平要求较高, 上手速度明显慢于图形 化工具。若公司中有专门的接口自动化测试开发团队, 这将是一个很好的选择。既然是 语言开发类, 也可以使用 java+testng, 可根据团队的语言能力, 选择开发框架。

Postman 大部分 web 项目的开发人员和测试人员都会选择使用这个工具。图形化的





界面,操作方便,执行结果易阅读,易调试。可以实现自动化批量运行。但免费版仅支持 http 和 https 协议,且无法读写数据库。









inport Runner 14.	🔛 My Workspace 🔻 🕹 Invite	🧿 📽 ঌ 🔺	🖤 🛛 Sign In
Q Filter History Collections	POST Untitled Request	No Environment	• • *
Trash Li	POST	Send 💌	Save 💌
	Params Authorization Headers (1) Body Pre-request Script Tests		Cookies Code
Collections let you group related requests, making them easier to access and run.	G + G + G + G + G + G + G + G + G + G +		
	Response		

Jmeter 支持很多种类的协议,支持从数据库动态取值,支持第三方插件,支持接口 对文件的操作。由于代码是开源的,即便没有支持的协议,也可以进行二次开发来完 成。但测试报告偏向于性能,大部分是性能指标,且断言功能不够强大,虽然提供了 beanshell 断言,但是提高了使用门槛,需要一点的编程能力。

(D:\/meterScript)	x) - Apache JMeter (5.0 r1840935)	-	□ ×
文件 编辑 查找 运行 选项 帮助			
		0 🔬 0	0/0 😯
· A Pinte-Kai · 가장면 · 가장면 · DBC Connection Configuration · · · · · · · · · · · · · · · · · · ·	201式计划 名称: 第65十回 注目:		
	用户定义的变量		
▶ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●			
可以对数据库操作			
	祥朝 落加 从野职报送3加 朝鮮 向上 向下		
	■ 独立运行每个线程组(除脚在一个组运行结束后启动下一个)		
	◎ 主线程结束后运行 tearDown线程组		
	■ 結晶類系成量式 丹生自然業品で再一法求从服务基础場合論集成文件的対象素及最低熱制式量式。 送年2~125所最多時世紀。		
	添加且录或jar包gClassPath 浏览 相條 清除		
	序 Dijaiffadagèmysqi-connedorjava-51.47		





为了节省成本,我选择开源软件。为了降低开发时间,我选择图形化的工具,减少 脚本的编写。我自身具备一定的开发能力,且需要做性能测试,所以我选择 Jmeter。在 开发完性能测试脚本后,只要稍加修改,就可以获得接口自动化测试脚本。大家还是需 要根据自身情况合理选择工具。

选择好工具后,编写测试脚本。我们需要创建线程组、配置元件以及一系列的请求。这些都是入门的内容,百度上都有,这里就不说了。下面给大家分享一下每个项目 都用的着的一些技巧:①如何提取响应结果中的某个参数?②获得参数后,如何使用? ③如何让系统自动判断响应结果是否正确?





二、提取参数

在实际项目中,有很多场景需要使用到响应结果中的参数。例如登录的响应结果中 包含了 token,后续的请求,都要使用这个 token 作为身份认证。例如 QQ 邮箱中修改一 封草稿邮件,在修改之前,我们要获取这封邮件的 sid。例如需要确认某个请求响应时 间是否在 5 秒以内,我们需要获取发送时间和响应时间。

目前很多项目的接口返回参数,都是以 json 格式去发送的。所以我们可以用 json 提取器来获取想要的参数。若不是 json 格式,我们可以使用万能的正则表达式提取器来提取。除了这2种常用的方法外, jmeter 还提供了近 10 种方法。

CSS/JQuery提取器
JSON提取器
正则表达式提取器
边界提取器
JSR223 PostProcessor
Debug PostProcessor
JDBC PostProcessor
XPath2 Extractor
XPath提取器
jp@gc - Inter-Thread Communication PostProcessor
jp@gc - XML Format Post Processor
结果状态处理器 初月上日3上0000
Bean Shell PostProcessor 🙀 件 測 试 🕅

/ 283	р.				
→提	添加		断言	►	
🍺 提	插入上级	►	定时器	►	
🥖 第-	·剪切	Ctrl-X	前置处理器	►	
/前	复制	Ctrl-C	后置处理器	►	CSS/JQuery提取器
// 首『 / こつヨ	粘贴	Ctrl-V	配置元件	►	JSON提取器
/ ਮਰਤ 🥒 ਮੋਟਤੋ	复写	Ctrl+Shift-C	监听器	►	正则表达式提取器
 e n					

JSON提取器		
名称: JSON提取器		
注释:		
Apply to:		
Main sample and sub-samples	Main sample only O Sub-samples only O JMeter Variable Name to use	
Names of created variables:		
JSON Path expressions:		
Match No. (0 for Random):		
Compute concatenation var (suffix _ALL)		
Default Values:		

第一种方法, json 提取。选中登录请求,选择后置处理器→json 提取器。



```
Names of created variables: 变量名
    JSON Path expressions: 需要提取内容的路径
    Match No.(0 for Random): 提取第几个匹配的值(若有 N 个匹配的值, 0 表示随机
取值,-1表示全部取值,1表示取第1个,2表示取第2个。。。N表示取第N个)
    Compute concatenation var (suffix_All): 勾选表示取所有的值,并保存到一个变量
中,命名方式是第一行设置的变量名加上_All
    Default Values: 没有提取到值,将给变量一个默认值
    下面举例说明
     "errno": 0,
     "data": {
       "user_name": "zhangsan",
       "mobile": "1990000001",
       "created_at": "2019-09-06 15:55:21",
       "real_name": "张三",
       "group_info": {
         "user_info": {
           "mobile": "1770000001",
           "created_at": "2019-09-06 15:55:21",
           "token_type": 2,
           "user_role": "普通用户",
           "login_count": 188,
           "password": "123456",
           "updated_at": "2019-09-06 15:55:21",
           "id": "991222",
         },
           "mobile": "1770000002",
           "created_at": "2019-09-21 11:33:16",
           "token_type": 2,
           "user_role": "普通用户",
           "login_count": 157,
           "password": "123456",
           "updated_at": "2019-09-22 09:50:01",
```



"id": "991223",



```
},
},
"token_type": 2,
"access_token": "d791640014b411eabf110bd02de67c73",
"login_count": 871,
"user_id": "331236",
"finger_sign": "147"
},
"errmsg": ""
```

```
这是一个登录后的响应请求,若我们需要提取 access_token,则可以这样设置
```

JSON提取器		
名称: 提取token,后续进行身份认证需要	要使用	
注释:		
Main sample and sub-samples	Main sample only 💿 Sub-samples only	JMeter Variable Name to use
Names of created variables:	token	
JSON Path expressions:	\$.data.access_token	
Match No. (0 for Random):		博为峰旗下
Compute concatenation var (suffix _ALL)		51 testina
Default Values:	nodata	

第一行设定变量名为 token, 第二行根据给定的路径进行查找, 找到后将值赋给 token。若找不到, 将最后一行的"nodata"赋给 token。根据\$.data.access_token, 我们可 以找到第 31 行的 d791640014b411eabf110bd02de67c73 并将他赋给 token, 以后的请求, 我们都可以使用这个变量 token 了。

JSON提取器	
名称: 提取电话号码	
注释:	
Apply to: Main sample and sub-samples	Main sample only 💿 Sub-samples only 💿 JMeter Variable Name to use
Names of created variables:	phone
JSON Path expressions:	\$.data.group_info.user_info.mobile
Match No. (0 for Random):	
Compute concatenation var (suffix _ALL):	
Default Values:	nodata

如果我们要提取电话号码,则\$.data.group_info.user_info.mobile 会匹配到第10行和 第20行,这里的 Match No 我设置的是0,代表随机获取这2个号码中的一个赋给



phone。并将两行的内容保存到变量 phone_ALL 里。

Json 提取器很简单,能否准确提取到想要的内容,主要就是 JSON Path expressions 设置的正确与否。在第一篇 selenium 自动化测试中,我们说过 Xpath,其实 json path 与 之类似,具体语法对比如下

XPath	JSONPath	描述
/	\$	根节点
	@	现行节点
/	. or []	取子节点
	n/a	取父节点,Jsonpath未支持
//		recursive descent(就是不管位置,选择所有符合条件的条件).JSONPath 从 E4X规范中继 承了这个语法的使用法。
*	*	匹配所有元素节点
@	n/a	根据属性访问,Json不支持,因为Json是个Key-value递归结构,不需要。
[]	[]	迭代器标示(可以在里边做简单的迭代操作,如数组下标,根据内容选值等)
1	[,]	支持迭代器中做多选。
n/a	[start:end:step]	从ES4规范继承的切片方法(非常实用) 博为峰旗下
[]	?()	支持过滤操作. うたのちに回りつ
n/a	()	支持表达式计算
()	n/a	分组,JsonPath不支持 软件测试网

{ "store": {

"book": [

{ "category": "reference",

"author": "Nigel Rees",

"title": "Sayings of the Century",

"price": 8.95

```
},
```

{ "category": "fiction",

"author": "Evelyn Waugh",

"title": "Sword of Honour",

"price": 12.99

```
},
```

{ "category": "fiction",

"author": "Herman Melville",

"title": "Moby Dick",

"isbn": "0-553-21311-3",

"price": 8.99

},

{ "category": "fiction",

"author": "J. R. R. Tolkien",

"title": "The Lord of the Rings",



《51 测试天地》五十六(上) www.51testing.com



```
"isbn": "0-395-19395-8",
  "price": 22.99
  }
 ],
 "bicycle": {
 "color": "red",
  "price": 19.95
}
示例表达式:
$.store.book[*].author: 商店所有书籍的作者(四个作者)
$...author: 所有作者
$.store.*: 商店所有的东西,包括 book 和 bicycle
$.store..price: 所有东西的价格
$..book[2]: 第三本书
$..book[0,1] /$..book[:2]: 前两本书
$..book[?(@.isbn)]: 用 isbn 编号过滤所有书籍
$..book[?(@.price<10)]: 过滤所有比 10 更便宜的书
$..*: XML 文档中的所有元素
第二种方法,正则表达式提取。
```





正则表达式提取器	
名称: 提取token	
注释:	
Apply to:	
Main sample and sul	b-samples 💿 Main sample only 🔵 Sub-s
321位旦口2时9月2子42 	
	● 主体 ● Body (unescaped) ●
	● 主体 Body (unescaped) token "access_token":"(.*?)"
	 主体 Body (unescaped) token "access_token":"(.*?)" \$1\$
2112日19月12子校 引用名称: 正则表达式: 模板: 匹配数字(0代表随机):	 主体 Body (unescaped) token "access_token":"(.*?)" \$1\$ 0
▲ 1 単 日 可 1 0 0 2 子 校 引用名称: 正则表达式: 模板: 匹配数字(0代表随机): 缺省值:	 主体 Body (unescaped) token "access_token":"(.*?)" \$1\$ 0 0

和 json 类似,关键就在表达式"access_token":"(.*?)"

在响应的数据中,其中有一行是这样的:

"access_token": "d791640014b411eabf110bd02de67c73",

那么我们的表达式,括号中的内容,就代表要提取的内容;

.号代表匹配任意字符串;

+号代表匹配一次或多次;

*号代表匹配任意次(包括0次);

?号代表不要贪婪,找到第一个匹配项,就停止匹配;

正则表达式的语法比较复杂,这里就不过多的介绍了,有兴趣可以自己搜集资料, 去练习。

三、参数传递

参数传递分为两类,线程内传递、线程间传递。以上面提取到的 token 为例,若想使用这个参数,在线程内是这样写的: \${token}







HTTP请求				
名称: 首页				
基本高级				
Web服务器				
协议:	服务器名和	你或IP:		
HTTP请求				
方法: POST	▼ 路径:			
自动重定向	✔ 銀糖重定向 🗹 使用 KeepAlive	对POST使用multipart / form-data	与浏览器兼容的头	
参数 消息体数	据 文件上传			
			同请求一起发访	送参数 :
			值	为峰旗下 编码?
token		\${token}	5	1testing

跨线程传递,需要借助 beanshell 取样器,先将变量通过 setProperty 函数设置为全局 变量,然后其他线程再通过 P 函数去引用,具体做法如下:

 ▼	
▶ 2 设置全局变量	
- ▼ 😳 线程组	
▶ 💉 第一次登录需要设置手势密码	

Beans	Shell 取样器
名称:	设置全局变量
注释:	
Res	et bsh.interpreter before each call
参数(→	• String Parameters 和 String []bsh.args)
脚本文件	ŧ
Script (s	ee below for variables that are defined
1 \${	setProperty(allThreadUseToken_\${token}_)};

登录和设置全局变量 这2个请求在同一个线程组内,所以登录请求获取到的 token,在设置全局变量的请求中,可以直接使用\${token},前面的

"allThreadUseToken"是全局变量名,可以随便起,自己记得住就行,最好做到见名知意。

"第一次登录需要设置手势密码"这个请求和登录请求在不同的线程组,所以无法 直接使用\${token},必须使用 allThreadUseToken,格式为\${__P(allThreadUseToken)},如





《51 测试天地》五十六(上)

www.51testing.com

下图:

HTTPì	清 求				
名称:	第一次登录需要设置	手势密码			
注释:					
基本	高级				
WebAR	务器				
协议:			服务器名称或IP:		
HTTPi	「求				
方法:	POST	▼ 路径:			
	动重定向 🗹 跟随重定	i向 🗹 使用	KeepAlive 📄 对POS	「使用multipart / form-data 📃 与淡	儿凝荣容的头
参数	消息体数据 文件_	上传			
					同请求一起发送参数:
token finger	sian	名称:		\${P(allThreadUseToken)} 147	<u>ii</u>
					^{博为峰旗下} 51上est ing 软件测试网

四、断言

所有请求都可以运行起来,我们还需要检查他的响应结果是否正确。Jmeter 提供了 很多断言方式,和上述的 json 提取器的样式类似,设置都很简单。下面对 beanshell 断 言进行一个介绍,因为其比较灵活,可以自己写断言内容。

	2 + - 🤣 > > 0 0 💐 👹 🚓 ≽ 🚍 👔						
▼ 👗 测试计划 🗙 测试环境	BeanShell断言						
▲ 察看结果树	名称:						
▲ 聚合报告							
▶ 🏦	Reset bsh.interpreter before each call						
	参数(-> String Parameters 和 String []bsh.args)						
× ¢	脚本文件						
	Script (see below for variables that are defined)						
	<pre>if(</pre>						
	<pre>14 FallureNessage = https://www.scipul.collet.</pre>						
	16 }else()						
	1) Failureiesage = 「有時份」(目前時間点);						
	19 🚺						
ድረሀው (17 :) !! 1- (!!							

```
if("${realTime}".equals("0")){
Failure = true;
FailureMessage = "没有获取到时间信息";
}else if("${realCode}".equals("{__P(sendCode)}")){
long t = Long.valueOf("${realTime}")-Long.valueOf("${__P(sendTime)}");
```





```
if(t < 0){
  t = 0 - t;
}
if(t \le 5)
  Failure = false;
  FailureMessage = "服务器的响应时间为" + t + "S" +
    "----响应时间戳: ${realTime}; 发送时间戳: ${__P(sendTime)}";
  }else{
    Failure = true:
    FailureMessage = "服务器的响应时间为" + t + "S" +
    "----响应时间戳: ${realTime}; 发送时间戳: ${ P(sendTime)}";
    }
}else{
  Failure = true;
  FailureMessage = "有响应, 但编码错乱";
  3
```

第一行的\${realTime}是响应结果中获取到的实际时间,第四行的\${realCode}是获取 到的时间格式,第四行的{__P(sendCode)}是另外一个线程内自定义的时间格式,第五行 的\${__P(sendTime)}是另外一个线程内自定义的发送时间。

Beanshell 断言中 Failure 取 true,代表发现错误,预期结果与实际结果不符;Failure 取 false,代表预期结果和实际结果相同;FailureMessage 是输出的错误信息。

上述代码的意思为:先判断是否提取到了真实响应时间,若没提取到,代表请求失败了。若提取到,则判断时间格式是否正确。若格式正确则比较发送时间和响应时间是 否大于5秒。

还是以登录为例,若数据库中正确的帐号密码是"zhangsan","123456"。发送登录 请求,输入正确的帐号密码,请求成功,响应结果如下:



此时我们可以提取 errno 为 0 进行 beanshell 断言。





若发送请求时,给出了错误的帐号密码,响应结果如下

JSON	▼ ▼ 取样器结果 请求 响应数据
_	Response Body Response headers
	Ę
	``"errno": 2000, "errmsg": "当前用户没有权限或用户名V密码错误"
	}

此时我们可以根据 errno 为 2000 进行 beanshell 断言,并同时判断提示信息 errmsg 内容是否正确。断言方式需要灵活选择,尽量直接使用 jmeter 提供的断言方法,减少开 发成本,最后选择 beanshell 断言。接口测试最重要的部分就是断言,一个没有断言的自 动化接口测试脚本,其测试结果毫无意义。

五、集成到 jenkins

Jmeter 图形化页面进行脚本编写,调试,相当方便。为了提高运行速度,建议使用 命令行模式运行脚本。

Jenkins 中, 创建一个自由风格的软件项目。

🚱 Jenkins	
Jenkins → 所有 →	
	输入一个任务名称
	»该字段不能为空,请输入一个合法的名称
	检建一个自由风格的软件项目 这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目,甚至可以构建软件以外的系统。
	构建一个maven项目 构建一个maven项目.Jenkins利用你的POM文件,这样可以大大减轻构建配置.
	流水线 精心地组织一个可以长期运行在多个节点上的任务。适用于构建流水线(更加正式地应当称为工作流),增加或者组织
	External Job This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This

若是需要远程节点运行,在 general 中勾选"限制项目的运行节点",如何添加远程 节点,已经在本系列的第一篇中详细介绍过了。我这里的 jmeter 的脚本,都是基于图形 化界面开发的,所以不需要进行源码管理。定时每周 1 到周 6,中午 12 点 30 分运行一 次。



《51 测试天地》五十六(上)

www.51testing.com



		General	源码管理	构建触发器	构建环境	构建	构建后操作	
			pullus					
		□ 丢弃旧的	的构建					
		□ 参数化构	建过程					
		□ 关闭构建	È					
		□ 在必要的	的时候并发构建	1				
		🗌 限制项目	的运行节点					
				1				
		源码管理	Ŧ					
		◎ 无						
		Git						
		Subvers	ion					
		构建触发	 					
		 	科建 (例如.使	「用脚本)				
		GitHub h	nook triaaer fo	r GITScm polling	3		博为峰旗下	
		□其他工程	記念を		9		51to	ching
				-			JILC	SLUIIL
			E.	_			<u>th</u>	रागा रेने छर
		日桂表	30 12 * * 1-6	5			书 计	测风风
勾建								
执行 V	Vindows 批处理命令							×
命令	del D:\Jenkins\workspace\	_API*.jtl						
	rd /s /q D:\Jenkins\workspace md D:\Jenkins\workspace\	e\ _API\web _API\web						
	jmeter -n -t D:\JmeterScript	\API_AutoTest\(.jnx −l D:\Jenk	rins\workspace\	_API\ '.	jtl —e —o D:'	\Jenkins\workspace\(
_								
	参阅可用环境变量列表							

当前运行环境是 windows 环境,所以用的是批处理命令。若是 linux 就使用 shell 命令。Jmeter 的命令行模式运行的一些命令,可以在官方网站上查看介绍,具体网站是: https://jmeter.apache.org 英语不好的,可以使用 google 浏览器,直接可以将英文网站翻译成中文网站




www.51testing.com

1.4 Running JMeter

To run JMeter, run the jmeter.bat (for Windows) or jmeter (for Unix) file. These files are found in the bin directory. After a short time, the JMeter GUI should appear. GUI mode should only be used for creating the test script, CLI mode (NON GUI) must be used for load testing There are some additional scripts in the bin directory that you may find useful. Windows script files (the .CMD files require Win2K or later): jmeter.bat run JMeter (in GUI mode by default) jmeterw.cmd run JMeter without the windows shell console (in GUI mode by default) jmeter-n.cmd drop a JMX file on this to run a CLI mode test jmeter-t.cmd drop a JMX file on this to run a CLI mode test remotely jmeter-t.cmd drop a JMX file on this to load it in GUI mode jmeter-t.cmd start JMeter in server mode

mirror-server.cmd runs the JMeter Mirror Server in CLI mode shutdown.cmd

Run the Shutdown client to stop a CLI mode instance gracefully

stoptest.cmd Run the Shutdown client to stop a CLI mode instance abruptly

The special name LAST can be used with jmeter-n.cmd, jmeter-t.cmd and jmeter-n-r.cmd and means the last test plan that was run interactively.

There are a few environment variables, that can be used to customize the JVM settings for JMeter. An easy way to set those is by creating a file named **setenv.bat** in the **bin** directory. Such a file could look like:

=value

rem This is the content of bin\setenv.bat, rem it will be called by bin\jmeter.bat set JVM_ARGS="-Xms1024m -Xmx1024m -Dpropname:

The JVM_ARGS can be used to override JVM settings in the jmeter.bat script and will get set when starting JMeter, e.g.

jmeter -t test.jmx …

放开样式权限(系列二中已经介绍过)

Groovy Script: System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")

设置在线查阅测试报告(系列二中已经介绍过)

Publish HTML reports

将测试结果发送到指定邮箱(系列二中已经介绍过)

Editable Email Notification





《51 测试天地》五十六(上)

www.51testing.com

Groovy Script System.s		Property("h	udson.model.DirectoryBrowserSupport.CSP", "")
		ovy Sandbo	х
If the script fails: Do nothing			•
Publish HT	ML reports		
Reports	HTML directory to	archive	Web
	Index page[s]		index.html
	Index page title[s]	(Optional)	
	Report title		HTML Report
	新增		
Editable Er	nail Notification		
	ended Email Publisher		

创建完成后,保存。构建失败可以在控制台中查看失败原因,重新调试程序。



《51 测试天地》五十六(上)

www.51testing.com



Jenkins Jenkins <tr< th=""><th>Vertice API Vertice API Vertice API Vertice Vertice Vertice<th></th><th></th><th></th></th></tr<>	Vertice API Vertice API Vertice API Vertice Vertice Vertice <th></th> <th></th> <th></th>			
Jenkins	Jerkins		🥹 Jenkins	
 ▲ 返回面板 ● 北志 ● 修改记录 ● 工作空间 ● 立即构建 ● ① 前除 工程 ● 節席 ● Email Template Testing ● Email Template Testing ● HTML Report ● ● 和名 	 ▲ 送回面板 ● 休花記 ● 作空间 ● ご印物建 ● 一一作空间 ● ① 印物建 ● 一一節 ● 一○ ● 一○ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●		Jenkins J_A	PI ▶
 ◇ 状态 ◇ 修改记录 ● 工作空间 ◇ 立即构建 ◇ 前除工程 ◇ 前除工程 ◇ 配置 ● Email Template Testing ● TTML Report ● 重命名 	 ・ 休衣 ・ 修衣记录 ・ 一 作空间 ・ ① 一 印作空间 ・ ② 一 助作 九田 ・ ○ 町町 九田 ● 町田I Template Testing ● 田THL Report ● 100 ● 100		🛧 返回面板	
 ○ 修改记录 ○ 工作空间 ○ 立即构建 ○ 前除 工程 ○ 前除 工程 ○ 前席 ○ Email Template Testing ○ HTML Report ○ 重命名 	● 飲む ● 工作空间 ● 立即构建 ● 動除 工程 ● 動除 工程 ● 配調 Template Testing ● HTML Report ● TTT 2019-12-2 下午12:30 ● 町 ● 「find ● 変更記录 ● 「近日 ● 「近日 ● 「「前兄」 ● 「「12-2 下午12:30 ● 愛」 空制台編出 ● 2:30 ● 「」 空利台編出 ● 「」 空利台編出 ● 「」 空利台編出		🔍 状态	
 ■ 工作空间 ② 立即构建 ③ 删除工程 ③ 耐除工程 ④ 配調i Template Testing ● HTML Report ● 重命名 Build History 构建历5 find find 230 ② 短明音編出 230 ② 230 ③ 短 短期音編出 230 ② 230 ③ 短 短期音編出 230 ② 230 ② 100 ② 230 ③ 230 ③ 100 ③ 100 ③ 100 ○ 100 <td> ▶ 二作空间 ※ 立即构建 ※ 耐除 工程 ※ 配置 ※ TMI Template Testing ※ TMIL Report ※ 重命名 </td> <td></td> <td>🤛 修改记录</td> <td></td>	 ▶ 二作空间 ※ 立即构建 ※ 耐除 工程 ※ 配置 ※ TMI Template Testing ※ TMIL Report ※ 重命名 		🤛 修改记录	
 ※ 立即构建 ※ 酬除 工程 ※ 配置 Email Template Testing ※ HTML Report ※ 重命名 Ø 型印名 Ø 型 型 型 型 型 型 型 型 型 型 型 型 型 型 型 型 型 型 型	 ○ 立即构建 ○ 御除 工程 ○ 配相 Template Testing ○ HTML Report ○ Taba ⑦ 重命名 1111111111111111111111111111111111		┣ 工作空间	
 ○ 删除工程 ● 配置 ● Email Template Testing ● HTML Report ● 重命名 Suild History 均理历5 find find ● 第日 2019-12-2 下午12:30 ● 第日 ② 空東记录 2:30 ● ② 控制台輸出 ● 2:30 ● ② 预除本次生成 ● 2:30	 ○ 融除工程 ○ 正都 Template Testing ○ HTML Report ○ 重命名 Suild History 的理历 find find ① 1019-12-2 下午12:30 ○ 前保護 ○ 短期 2:30 ○ 2:30 ○ 回 控制 2:30 ○ 2:30 ○ 回 按制 ○ 副除本次生成 2:30 ○ 2:30 ○ 2:30 ○ 一個 中午小生 ○ 2:30 ○ 一〇 ○ 新除本次生 ○ 2:30			
 ◇ 配置 ◇ Email Template Testing ◇ HTML Report ◇ 重命名 ◇ Build History 均建历9 「find ◇ #TT 2019-12-2下午12:30 ◇ 並利台輸出 2:30 ◇ ご 控制台輸出 2:30 ◇ ご 控制台輸出 2:30 ◇ 資 編編編译信息 2:30 ◇ 動除本次生成 2:30 	 ○ 正式 ○ ○ ○ 正式 ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○		◎ 删除工程	
 Will Email Template Testing ● HTML Report ● 童命名 ● Build History 均建历灯 find ● #77 2019-12-2下午12:30 ● #77 2019-12-2下午12:30 ● ● 並制台輸出 2:30 ● ● 读 接制台輸出 2:30 ● ● 読 接場译信息 2:30 ● ● 動除本次生成 2:30 	 ○ Email Template Testing ○ HTML Report ○ 重命名 111 2019-12-2 下午12:30 ● 111 2019-12-2 下午12:30 ● 111 2019-12-2 □ 12:30 ● 111 20		4 配置	
 ▶ HTML Report ▶ 目前名 ▶ Build History 均建历史 find ₱ #77 2019-12-2下午12:30 ● #77 2019-12-2下午12:30 ● 第 空更记录 2:30 ● 控制台輸出 2:30 ● 控制台輸出 2:30 ● 資 強制台輸出 2:30 ● 資 納編編译信息 2:30 ● 勤除本次生成 2:30 	▶ HTML Report ● ■命名 ● ● ■の名 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●		Email Template Testing	
■ mine report 重命名 Build History 均建历史 find #77 2019-12-2下午12:30 ● #77 2019-12-2下午12:30 ● 愛更记录 2:30 ● 愛更记录 2:30 ● 愛 強制台輸出 2:30 ● 愛 編辑编译信息 2:30 ● 認 始附本次生成 2:30	Image: Notion Construction Image: Definition Image: Definition <			
▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ ▶ >	Puild History 均建历史 find #17 2019-12-2 下午12:30 デア 2019-12-2 下午12:30 2:30 デ要更记录 2:30 2:30 デ約 振編編译信息 2:30 デ動台輸出 2:30 2:30 デ動台輸出 2:30 2:30 デ動除本次生成 2:30 2:30			
Build History 的建历史 find 	Puild History 内建历史 find #17 2019-12-2 下午12:30 デア 2019-12-2 下午12:30 2:30 デア 空更记录 2:30 デ 控制台輸出 2:30 デ 編編編译信息 2:30 デ 御殿本次生成 2:30		🧖 포마디	
find ● #77 2019-12-2 下午12:30 ● 愛更记录 2:30 ● 控制台輸出 2:30 ● 塗 編編編译信息 2:30 ● 塗 編編編译信息 2:30 ● 診 細除本次生成 2:30	find #17 2019-12-2 下午12:30 愛 変更记录 2:30 愛 控制台輸出 2:30 愛 加除本次生成 2:30 2:30 2:30		🔅 Build History	构建历史
#77 2019-12-2 下午12:30 ● 一 空更记录 2:30 ● 空則台輸出 2:30 ● 空制台輸出 2:30 ● 空制台輸出 2:30 ● 空制台輸出 2:30 ● 受 卵除本次生成 2:30	ma #77 2019-12-2下午12:30 ② 変更记录 2:30 ○ 注射台輸出 2:30 ○ 編編编译信息 2:30 2:30 2:30 2:30 2:30 2:30 2:30 2:30 2:30 2:30		find.	
 #// 2019-12-2 下午12:30 ● 变更记录 2:30 ● 控制台输出 12:30 ● 涂辑编译信息 2:30 ● 微 删除本次生成 2:30 	 ● #// 2019-12-2 下年12:30 ● 変更记录 ● 空更记录 ● 控制台输出 ● 空見注意 ● 2:30 			
 ● 受更近策 ● 控制台輸出 ● 違制台輸出 ● 違規编译信息 ● 御除本次生成 2:30 2:30 2:30 	 ○ 受更比束 ○ 受更比束 ○ 2:30 ○ 控制台输出 ○ 深編編编译信息 ○ 別除本次生成 2:30 2:30 2:30 2:30 2:30 		● <u>#//</u> 2019-12-2 ト午12 ● 二二 本面に見る	2:30
 ● 控制台輸出 ● 编辑编译信息 ● 例除本次生成 2:30 12:30 	 ○ 控制台輸出 ○ 控制台輸出 ○ 控制台輸出 ○ 控制台輸出 ○ 控制台輸出 ○ 控制台輸出 			12:30
 ● 編辑编译信息 ● ② 删除本次生成 12:30 	 ○ 編編编译信息 ○ 删除本次生成 2:30 2:30 			12:30
● ● 删除本次生成 12:30	 ◎ ◎ [●] ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●		♀ 編辑编译信息	12:30
	▲ 控制		● 🚫 删除本次生成	12:30
	▲ 控制会输出	<u> </u>	-	
		由定时器启动 在 master 上构建 在工作'	空间 D:\Jenkins\workspace\	d API 中
由定时器启动 在 master 上构建 在丁作空间 D:\Jenkins\workspace\/d API 中	由定时器启动 在 master 上构建 在工作空间 D:\Tenkins\workspace\/ d API 中	No emails were triggered	⊥, j = t (; there (starting for (d. (a. sell β,))#INDOWS) TEND) i selsin = 45°	20140050024764557 b.+
由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(d_API 中 No emails were triggered. [Currended ADI] \$ and (a call CalWINDOWS)TIND\imbine4628142950224764557 hat	[GansuGrid_AFI] & Cmd /(c call C:(WIMDOWS(IEmP(Jenkins40.	30143333324104331. Bat
由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(d_API 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat	由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\	D:\Jenkins\workspace\	API>del D:\Jenkins\works	pace\(
由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(d_API 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat D:\Jenkins\workspace\API>del D:\Jenkins\workspace\(API*.jtl	由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(D:\Jenkins\workspace\	_API>rd /s /q D:\Jenkins\	workspace\API\web
由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\ _d_API 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat D:\Jenkins\workspace\ API>del D:\Jenkins\workspace\^ API*.jtl):\Jenkins\workspace\API>rd /s /q D:\Jenkins\workspace\API\web	由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(d_API 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat D:\Jenkins\workspace\API>del D:\Jenkins\workspace\(API*.jtl):\Jenkins\workspace\API>rd /s /q D:\Jenkins\workspace\API\web			ace\ API\web
由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(d_API 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat D:\Jenkins\workspace\API>del D:\Jenkins\workspace\(API*.jtl D:\Jenkins\workspace\API>rd /s /q D:\Jenkins\workspace\API\web D:\Jenkins\workspace\API>md D:\Jenkins\workspace\API\web	由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(d_API 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat D:\Jenkins\workspace\API>del D:\Jenkins\workspace\(API*.jtl D:\Jenkins\workspace\API>rd /s /q D:\Jenkins\workspace\API\web D:\Jenkins\workspace\API>md D:\Jenkins\workspace\API\web	D:\Jenkins\workspace\	LAPI2md D:\Jenkins\worksp:	
由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(dAPI 中 No emails were triggered. [GansuGrid_API] \$ cmd /c call C:\WINDOWS\TEMP\jenkins4538143959324764557.bat D:\Jenkins\workspace\aPI>del D:\Jenkins\workspace\(API*.jtl D:\Jenkins\workspace\API>rd /s /q D:\Jenkins\workspace\API\web D:\Jenkins\workspace\API>md D:\Jenkins\workspace\API\web D:\Jenkins\workspace\API>jmeter -n -t D:\JmeterScript\API_AutoTest\jmx -l 1 Creating summariser <summary></summary>	由定时器启动 在 master 上构建 在工作空间 D:\Jenkins\workspace\(D:\Jenkins\workspace\ D:\Jenkins\workspace\ Creating summariser <suu< td=""><td>_API>md D:\Jenkins\worksp _API>jmeter -n -t D:\Jmete</td><td>erScript\API_AutoTest\jmx -l I</td></suu<>	_API>md D:\Jenkins\worksp _API>jmeter -n -t D:\Jmete	erScript\API_AutoTest\jmx -l I

构建成功可以在点击 html report 在线查看运行结果:



www.51testing.com

🚹 返回面板				⊤₽¤ ∕										
🔍 状态				上在 API										
🔁 修改记录			-	2019-08-30	创建项目, ^{集成:}	每周1-6中午	F12点30拔 亲[い行, 集成。 1的自动化源	Ref.	· · · · ·			,自动化测试	
💼 工作空间				2010 00 00	HE-402	_								
🕥 立即构建						-								
◎ 删除 工程														
🐡 配置				HTML Report										
Email Template Te	sting			-										
HTML Report				■ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●										
🦻 重命名														
Build History	ł	构建历史 🗝	;	相关链接	ž									
find			x	• <u>最近</u> -	-次构建(#7	<u>77),3 小时 2</u>	8 分之前							
#77 2019-12-2	下午12:30			• <u>最近</u> 程 • 最近6	<u>急定构建(#7</u> 成功的构建(<u>77),3 小时 2</u> (#77),3 小时	<u>8 分之前</u> † 28 分之前	ţ						
#76 2019-11-30	下午12:30			 最近; 	記成的构建(#77),3 小时	128 分之前	<u>ti</u>						
2019-11-29	下午12:30													
ck to API index														Zip
Dashboard														
L Charts C						Tes	st and Rep	ort informatio	ons					
Customs Graphs	Source file Start Time						*12/2/19	12:30 PM"						
	End Time						*12/2/19	12:30 PM"						
	riter for display						-							
		APDE	X (Applica	ation Perform	nance Index)				Re	quests Summar	У		
	Apdex	 T (Tole 500 ms 	ration threshold	 F (Frustra 1 sec 500 r 	tion threshold) •	Label	٠							KO COK
	1.000	500 ms		1 sec 500 m	15									
	1.000	500 ms		1 sec 500 n 1 sec 500 n	15						and the st	1 IA MET		
	1.000	500 ms		1 sec 500 n	15						100%	暉棋		
	<u> </u>											SEU	ng	
				Statistics								测试		
	Remeste		Executions					Response	imes (ms)		-77		Network (KB/sec)	
	Label -	#Samples	 КО 0 	 Error % 0.00% 	Average 49.08	• Min 2	 Max 242 	 90th pct 112.50 	 95th pct 211.50 	 99th pct 242.00 	Throughput 17.03	 Receit 22.22 	ved • Sent 12.14	•
		1	0	0.00%	2.00	2	2	2.00	2.00	2.00	500.00	133.79	133.79	
		1	0	0.00%	72.00	72	72	72.00	72.00	72.00	13.89	21.15	22.54	
	7			0.00%	120.00	120	120	120.00	120.00	120.00	8.33	12.68	13.52	
		1	0				4	4.00	4.00	4.00	250.00	50.29	407.47	
	14.66	1	0	0.00%	4.00	4					4.13	3.25	2.58	
	1444	1	0	0.00%	4.00	242	242	242.00	242.00	242.00			12.44	
		1	0 0 0	0.00%	4.00 242.00 58.00	4 242 58	242	242.00	242.00	242.00	17.24	13.71		
	H. A. H. L. L.	1 1 1 1 1	0 0 0 0 0	0.00%	4.00 242.00 58.00 18.00	4 242 58 18	242 58 18	242.00 58.00 18.00	242.00 58.00 18.00	242.00 58.00 18.00	17.24	13.71 9.71	42.37	
	h a tri a la se	1 1 1 1 1 1	0 0 0 0 0 0 0	0.00% 0.00% 0.00% 0.00%	4.00 242.00 58.00 18.00 61.00	4 242 58 18 61	242 58 18 61	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	17.24 55.56 16.39	9.71 15.96	42.37 13.90	
		1 1 1 1 1 1 1	0 0 0 0 0	0.00% 0.00% 0.00% 0.00% 0.00%	4.00 242.00 58.00 18.00 61.00	4 242 58 18 61	242 58 18 61	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	17.24 55.56 16.39	13.71 9.71 15.96	42.37 13.90	
		1 1 1 1 1 1		0.00%	4.00 242.00 58.00 18.00 61.00	4 242 58 18 61	242 58 18 61	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	17.24 55.56 16.39	13.71 9.71 15.96	42.37 13.90	
		1 1 1 1 1 1		0.00% 0.00% 0.00% 0.00%	4.00 242.00 56.00 61.00 Num	4 242 56 18 61	242 58 18 61 En	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00	17.24 55.56 16.39	13.71 9.71 15.96	42.37 13.90	
		1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		0.00% 0.00% 0.00% 0.00%	4.00 242.00 58.00 18.00 61.00 Num	4 242 58 58 61 18 61	242 58 18 61 En	242.00 58.00 18.00 61.00	242.00 58.00 18.00 61.00 % in errors	242.00 56.00 18.00 61.00	17.24 55.56 16.39	13.71 9.71 15.96	42.37 13.90	
		1 1 1 1 1 1 1		0.00%	4.00 242.00 58.00 18.00 61.00 Num	4 242 56 18 61 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	242 58 18 61 Err	242.00 58.00 18.00 61.00 Fors 5 by sampler	242.00 56.00 18.00 61.00 Numerrors	242.00 58.00 18.00 61.00	17.24 05.56 16.39	13.71 9.71 15.96	42.37 13.90	
		Type of error Type of error Stamples	• # #Errc	0.00% 0.00% 0.00% 0.00% 0.00%	4.00 242.00 58.00 18.00 61.00 Num Num	4 242 00 10 61 0 0 0 7 7 20 20 20 20 20 20 20 20 20 20 20 20 20	242 58 18 61 Err 70p 5 Error	242.00 58.00 51.00 Fors S by sampler reres • En	2 42 00 58 00 18 00 61 00 76 in errors	242.00 56.00 18.00 61.00	• • • • • • • •	13.71 9.71 15.96 * in all samp Error	42.37 13.90 tes 9 8 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7	





机器学习之 KNN 算法,好算法有迹可循!

◆ 作者:咖啡猫

题记: 众所周知,机器学习中有很多算法,像前面几期文章中介绍的逻辑回归算法、 PageRank 算法、K-means 聚类算法等等。其实,每一种算法都有自己的优点和不足,都有自己 特定的使用场景。这就要求我们在具体应用时,学会选择和比较,一般来说,我们会比较算法 的时间复杂度、运行效率和正确率等因素,在某一特定的场景里,选择运行效率高且正确率高 的算法。

言归正传,本文咖啡猫将带大家了解 KNN 算法的原理及 Python 的实现过程,希望小伙伴 们能够掌握它。

一、什么是 KNN 算法

KNN 算法的全称是 K 最近邻算法(即: K-NearestNeighbor)。它是机器学习数据挖掘分类技术中最简单的分类算法之一,它的核心思想是:每个样本的类别取决于它最近的 K 个邻居的类别。

如图1所示,要想知道绿色点属于哪一类(红色三角类或蓝色方框类),我们可以 选取离绿色点最近的 K 个点,统计这 K 个点中大部分点属于哪一类,那么这个绿色点 就属于哪一类。如图当 K 取 3 时,绿色点属于红色三角类,当 K 取 5 时,绿色点属于 蓝色方块类。





二、KNN 的算法流程

- 1) 计算测试数据与各个训练数据的距离。
- 2) 对训练数据按距离从小到大排序
- 3) 取排序中前 K 个点
- 4) 统计这 K 个点所在的类别(即统计每一种类别的元素个数)

5) 以元素最多的类别作为测试数据所属类别

关于K值的选取: 由图1中的例子可以知道,K值的选取对分类结果的影响很大: k 过小,则受噪声的影响大; k 过大,则利用较大邻域中的数据进行分类预测,算法时 间复杂度就会上升。在实际应用中,常用的方法就是K从1开始取到N,统计不同K 值下模型的正确率,最后选择正确率最高的K值。另外,K要尽量取奇数,来保证最后 有只有一个类别的元素个数较多。

关于距离的计算公式:常用的距离计算公式有欧几里得距离、余弦值、相关度、曼 哈顿距离等。在本文中我们选择欧几里得距离公式,即:

$$E(x,y) = \sqrt{\sum_{i=0}^{n} (x_i - y_i)^2}$$
 (其中, n为样本点的维数)

三、KNN 的实现方法

#!/usr/bin/env python
-*-coding: utf-8 -*import numpy as np
from math import sqrt
import operator as opt





```
return max index
def normData(dataSet): #特征缩放,可以帮助梯度下降更快收敛
   maxVals = dataSet.max(axis=0)#返回矩阵中每列最大值
   minVals = dataSet.min(axis=0)#返回矩阵中每列最大值
   ranges = maxVals - minVals#得到每列最大值和最小值的差
   retData = (dataSet - minVals) / ranges#归一化
   return retData, ranges, minVals
def kNN(dataSet, labels, testData, k):
   distSquareMat = (dataSet - testData) ** 2 # 计算差值的平方
   distSquareSums = distSquareMat.sum(axis=1) # 求每一行的差值平方和
   distances = distSquareSums ** 0.5 # 开根号,得出每个样本到测试点的距离
   sortedIndices = distances.argsort() # 从小到大排序,得到排序后的索引
   indices = sortedIndices[:k] # 取最小的 k 个
   labelCount = {} # 存储每个 label 的出现次数
   for i in indices:
       label = labels[i]
       labelCount[label] = labelCount.get(label, 0) + 1 # 次数加一
   sortedCount = sorted(labelCount.items(), key=opt.itemgetter(1), reverse=True) # 对 label 出现的
次数从大到小进行排序
   #labelCount.items()将字典以元组形式输出 Key 调用每个元组索引为1的元素(决定排序对
象为次数),
   return sortedCount[0][0] # 返回出现次数最大的 label
if ______ == "_____main____":
   A = np.zeros((900, 3), dtype=float) # 先创建一个 900x3 的全零方阵 A, 并且数据的类型设
置为 float 浮点型
   \mathbf{B} = []
   f = open('G:/testset.txt') # 打开数据文件文件
   # lines = f.readlines() # 把全部数据文件读到一个列表 lines 中
   \mathbf{n} = \mathbf{0}
   while n < 900:
       list1 = f.readline().strip('\n').split(' ')
       A[n:] = list1[0:3] # 把处理后的数据放到方阵 A 中。list[0:3]表示列表的 0,1,2 列数据放
到矩阵 A 中的 row 行
       B[n:] = list1[3:4] # 把列表第3列(标签)放到B矩阵中
       n = n + 1
   C = np.zeros((100, 3), dtype=float) # 先创建一个 100x3 的全零方阵 C, 并且数据的类型设
置为 float 浮点型
```





```
<mark>初進₩下</mark>

51 Lest 009

软件测试网
```

```
D = []
    n = 0
    while n < 100:
        list1 = f.readline().strip('\n').split(' ')
        C[n:] = list1[0:3] # 把处理后的数据放到方阵 A 中。list[0:3]表示列表的 0,1,2 列数据放
到矩阵 C 中的 row 行
        D[n:] = list1[3:4] # 把列表第3列(标签)放到D矩阵中
        \mathbf{n} = \mathbf{n} + \mathbf{1}
    normDataSet, ranges, minVals = normData(A) #对训练集 A 做归一化处理 normData 就是归一
化函数。
    testData = np.array(C)
    normTestData = (testData - minVals) / ranges #
    correct_rate = [] #存储不同 K 情况下的正确率, K=1.....100
    for k in range(1,1001):
      n=0
      count = 0
      while n < 100:
       test=normTestData[n]
       result = kNN(normDataSet, B, test, k)
       if result == D[n]:#比较预测值与实际值,若相同 count 加1,便于计算正确率
           count = count + 1
       n = n + 1
      #print('当 k 为%d 时, 算法正确率为%f'%(k,float(count/100)))
      correct_rate.append(float(count/100))
    a=maxIndex(correct_rate)#求正确率最高的 K
    #print(correct_rate)
print('当 k 为%d 时,得到最大的算法正确率为%f'% (a+1,correct_rate[a]))
代码要点说明:
```

测试数据集是放在 G 盘一个名为 testset 的 txt 文档中的,该文档有 1000 行 4 列(前 三列为样本特征、最后一列为样本分类标签),我将 1-900 行的数据作为训练集,901-1000 行数据做测试集。





在 main 函数中, 首先将训练集的样本特征放矩阵 A 中、训练集的标签放矩阵 B 中, 然后将测试集的样本特征放矩阵 C 中、测试集的标签放矩阵 D 中, 然后对训练集和 测试集进行归一化处理来提高结果的准确度, (因为如果某个特征的值域很大, 那么距 离计算主要取决于该特征, 从而与实际情况不符), 最后对不同的 K 值运用 KNN 函数得 到预测结果并统计正确率, 将正确率最大的 K 值打印出来。

核心函数 KNN 就是计算每一个测试样本与训练样本的距离,然后统计距离最近的 K个训练样本的标签,选择标签中元素最多的那个标签作为测试样本的标签。其中涉及 一些 numpy 数组运算的内容以及常规排序操作,需要大家熟悉。

四、测试结果分析



由运行结果可知: 当 K 取 2 时,算法的正确率高达 95%。此外,你也可以尝试用其他距离公式来构建 KNN,选择正确率最高的距离公式,作为最终的建模参数。

后记:

KNN 分类算法就介绍到这里了,可见弄清原理之后算法也不是那么高不可攀的, 一定不要有畏难情绪哦!希望小伙伴们能够熟知它的原理并且可以用 PYTHON 来实现 它,下次再遇到数据分类任务的时候,直接套用代码稍微修改一下就能解决实际问题 啦!你 GET 到了么?



全流程自动化测试—业务规则标准 化及资产库建设探索 ◆ 作者 : 于虹

从需求分析开始到案例自动化执行的全流程自动化测试一直是 c 测试人员改改不倦 的追求。目前,业界已有自动化测试平台,实现测试案例的自动化执行。然而,作为测 试工作第一步的需求分析,尚未有成熟的研究成果。

全流程自动化测试探索

一般来讲,测试的基本工作流程可分为三个阶段:需求分析、案例设计和案例执行。业务规则标准化研究小组基于案例设计和案例执行阶段已有的研究成果和测试资产,开展了需求分析阶段规范化和标准化的研究,旨在从业务需求层面积累业务规则,并将业务规则对接到高级测试案例的自动生成和执行,实现从需求分析开始,对接案例设计和案例执行的全流程自动化测试,提升整个测试流程的自动化水平,进而提高测试质量和效率,并逐步建设各个业务领域的业务规则资产库。以此来实现包括需求分析、案例设计、案例执行的全流程自动化测试。







初见雏形

研究小组为了标准化、结构化的诠释和存储业务规则,设计了一种六层结构的业务 需求规范化模型,将一项业务规则表达为"系统-模块-流程-功能-要素-规则"的组合。 业务规则标准化是一个自上而下逐层分析的过程,总体思路是以应用系统为单位,对该 系统业务需求中的各项业务规则按照这六层标准结构进行诠释和展现。



在模型的要素层中存储页面级规则,规则层中存储功能级规则和流程级规则,这三 类规则分别对应高级案例中测试类型为"页面"、"功能"、"流程"的案例。研究小组为 这三类规则分别制定了表达规范,以支持从业务规则到高级案例的自动生成和执行。

页面级规则约定了界面级功能的输入集合中各个独立对象的合法性。一般通过分析 对象类型来确定对象合法性的分析角度。目前高级案例模板中定义了14种对象类型, 需要从是否必输、是否有默认值、长度、数据类型等方面分析合法性,研究小组从这四 个角度分别制定了表达规范,后续还可根据实际测试需要进行扩展。

功能级规则是指在输入要素合法的前提下,描述要素的不同取值组合对整体功能的 影响。分析步骤包括:提取业务规则——进行规则描述——识别规则中涉及的要素—— 配置规则表达式——配置正/反向案例检查点。研究小组提出了提取功能级业务规则的一 般方法,进行规则描述的5条原则和3种句法,配置规则表达式和正/反向案例检查点的 表达规范,以及进行功能级规则维护的各项原则。





《51 测试天地》五十六(上)

www.51testing.com



流程是由一系列相互关联或相互作用的功能串联起来实现将输入转化为输出的活动。在功能正确实现的前提下,不同的功能组合导致了流程处理的差异。分析步骤包括:提取业务规则——进行规则描述——识别规则中涉及的要素及取值——配置流程中各功能的取值——配置正/反向案例检查点。研究小组提出了提取流程级业务规则的一般方法,流程级规则的描述规范,配置流程中各功能的取值和正/反向案例检查点的表达规范,以及进行流程级规则维护的各项原则。

通过进行上述多维度业务规则的标准化,不仅可实现从需求分析开始,对接案例设 计和案例执行的全流程自动化测试,还能逐步积累业务规则,开展各领域的业务规则资 产库建设。

未来可期

基于业务规则标准化及资产库建设探索不仅仅单纯的解决目前手工编制测试案例的 问题,提升整个测试流程的自动化水平,同时,实现各领域内业务规则的积累,将测试 人员的经验固化为组织资产,防止因人员流失造成的知识流失。今后,测试工作将只需 进行业务需求分析,案例的设计和执行都将自动化完成,实现一种全新的全流程自动化 测试模式。



◆ 作者:朱思衡



JMeter 循环读取 CSV 文件实现接口批 量测试

首先要理解为什么要进行批量测试,当我们在工作中进行接口测试时,项目的接口 肯定不止一个,而是很多很多,而且每个接口都需要进行正确参数,错误参数,参数为 空,特殊字符等方式来测试接口是否能够正确返回所需的响应值。

在编写脚本之前可以先汉化 JMeter, 找到 JMeter 解压/安装路径下的 JMeter.properties,用编辑器打开(右键用记事本打开也可以): 找到 #language=en , 在下面添加: language=zh_CN 并重启 JMeter

那么我们该如何通过 JMeter 来完成批量测试呢? 我们先建立一个较为简单的方式 进行测试脚本观察一下。

以去获取创建订单时的【省/市/区信息】接口:

api.test.XXXXXX.com/api/XXXX/XXXX/getAddress 请求方法: post 请求参数: type, code 为例

一、添加线程组

测试计划右键选择添加--线程用户--线程组。

Setup 线程组:用于执行预测试操作

tearDown: 用于执行测试后操作

线程组:一般测试接口时添加的线程,可以把它看做一个虚拟的用户组,线程组中的每一个线程都可以理解为一个虚拟用户





www.51testing.com

文件编辑查找	运行选项帮助	+ - 4		 Image: Image: Ima		1	<i>8</i>	₺ ፲
🔔 Witi	添加	Ctrl+V		线程(用户) 配置元件 监听器		setUp线 tearDo 线程组	线程组 wn线程	
	打开 合并 选中部分保存为			定时器	•			
	保存节点为图片 保存屏幕为图片 中田	Ctrl+G Ctrl+Shift+G		后置处理器 断言 测试 比的))			
	后用 禁用 切换	Ctrl+T	_	非测试元件	•	博为峰 511	at esl	: ^C ina
	帮助					软作	牛测	试网

二、添加 HTTP 信息头管理器

线程组右键选择添加——配置元件——HTTP 信息头管理器

如果接口信息头定义了值的话,需要放到信息头管理器里面,例如:用户 token, 公共参数等。

□ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	HTTP信息头管理器 名称: NTTP信息头管理器 注释: 在::::::::::::::::::::::::::::::::::	
	信息大仔陌往信息大官埋益中 名称:	<u>ä</u>
	token	792431b+6ba1473588ab4+25b8547c721575879779600
		参数值

三、添加 HTTP 请求

线程组右键添加——取样器——HTTP 请求

协议: 默认 HTTP 协议,可不填写,若为 HTTPS 协议需要填写 HTTPS。

服务器名称或 IP: 填写接口域名

端口号: 默认 HTTPS 请求的 80, 若为 HTTPS 请求需要填写 443

方法:直接选择接口的请求方法:GET,POST,PUT,DELETE即可。

路径:填写接口的具体路径

参数:点击下方导航栏添加按钮,在已经添加的数据栏填写参数名,参数值。







四、添加察看结果树

线程组右键添加——监听器——察看结果树

左边:展示请求名,绿色请求结果表示请求成功,红色请求结果表示请求失败。 右边:展示请求值,包括取样器结果,请求头,请求体,响应头,响应体。

 → 3015145 → 2015145 → 20151	察看结果树 名称: 察看结果树 注释:	展示请求值 /
	所有該据写入一个文件 文件名 	「「「「」」「「「」」」「「」」」「「」」」「「」」」「「」」」「「」」」
展示请求名	Text RiffEdda Room	Lange for the second of the s

以上步骤能够确保接口正确的进行访问并返回对应的返回值。

完成以上步骤后,下一步就是要把上面步骤的所有的值以参数的形式传递给 JMeter 脚本,使我们能够以参数化的形式对接口进行批量测试。

五、添加循环控制器

线程组右键添加——逻辑控制器——循环控制器,循环控制器添加后需要把 【HTTP 信息头管理器】【HTTP 请求】移到【循环控制器下】

循环次数:运行脚本后循环控制器下的 HTTP 请求将执行的次数。例如:循环控制





www.51testing.com

器的计数为1,则发送HTTP请求1次;计数为5,则发送HTTP请求5次。设置为永远则需要手动停止请求,否则不会停止请求。

 ■	循环控制器 名称: 循环控制器 注释:	비器 研控制器				
∽ ≪ 祭者绐未附	循环次数 二 永远 1					

六、在循环控制器下添加 CSV 配置文件

循环控制器右键添加——配置元件——CSV数据文件设置,CSV文件里面定义接口 所需要动态取值的参数,例如: URL,路径,请求方法,接口参数等。

a)如何创建 CSV 文件? 电脑桌面新建 TXT 文档,更名为:测试参数文档.CSV (TXT 后缀需要变更为 CSV),打开 CSV 文件新增数据。

b) 文件内参数如何书写? 第一行定义接口所需动态参数, 自第二行起填写接口详 细值。

- caseSeq: 定义接口序号
- apiSeq: 每个接口序列
- apiName: 接口名称
- url: 接口域名
- api: 接口地址
- function: 接口请求方法
- purpose: 描述接口验证类型
- parameter: 接口所需参数拼接,以: "code=500000&type=city"的形式,参数间以 "&" 连接

- 4		В	C	P	E	F	c	н	I	I	K	L	r
1	caseSeq	apiSeq	apiNane	url	api	function	purpose	parameter					
2	C1	1.1	获取省/市/区信息	∃api.test.duias	/api/starkid/order/g	POST	参数正确	code=500000&type=city	100			- State	
3	C1	1.2	获取省/市/区信息	api.test.duias	/api/starkid/order/g	POST	参数错误	type=xxx	楼口	听罢7	い念き	S&V	
4	C1	1.3	获取省/市/区信息	∄api.test.duias	api/starkid/order/ge	POST	参数为空	type=					
5	C2	2.1	获取用户信息	api.test.duias	/api/starkid/getUses	rPOST	参数正确	userId=2251					
6	C2	2.2	获取用户信息	api.test.duias	/api/starkid/getUses	rPOST	参数为空	userId=					
7													
8													
9													
10													



c) csv 数据文件如何设置

文件名: 点击浏览后选择本地的 CSV 文件

文件编码: 一般情况下无需设置

变量名称: "caseSeq, apiSeq, apiName, url, api, function, purpose, parameter", 参数之间需要用英文的逗号隔开, 注意参数不要写错。

忽略首行: True 则不执行 CSV 文件的首行, False 则执行 CSV 文件首行。

分隔符: 英文逗号, 无需变更

是否允许带引号:设置 True 和 False 以当前 CSV 文件都能正常执行

遇到文件结束符再次循环: True则在设置的循环次数内一直循环, False则 CSV 文件所有数据循环一次后就停止。

遇到文件结束符停止线程: True则在设置的循环次数内执行到 CSV 文件数据为空时停止发送 HTTP 请求, False 则在设置的循环次数内执行到 CSV 文件数据为空时继续执行。

线程共享模式:选择所有现场即可

综上:本次 CSV 文件设置时:循环次数"永远",忽略首行"True",是否允许带引号"False",遇到文件结束符再次循环"False",遇到文件结束符停止线程"True",线程共享模式"所有现场"

→ 测试计划 ● 续程组 日 循环控制器 KTIP信息头管理器 SV数据文件设置	CSV 数据文件设置 名称: CSV 数据文件设置 注释:
□□	─ 设置 CSV 数据文件
	文件名: C:/Users/Administrator/Desktop/测试参数文档.csv 浏览
	文件编码:
	变重名称(西文逗号间隔): caseSeq.apiSeq.apiName.url.api,function.purpose.parameter
	忽略首行(只在设置了变里名称后才生效): Irue ▼
	分隔符 (用'\t'代替制表符):,
	是否允许带引号?: False ▼
	遇到文件结束符再次循环?:False 博为峰旗下 🗸 🔻
	遇到文件结束符停止线程?: Irue
	线程共享模式: 所有现场
	软件测试网

七、其他配置文件需要修改写入形式

HTTP 信息头管理器: 除必要参数外,添加: "Content-Type=application/x-www-





form-urlencoded",使用 Json 参数需要用到

HTTP请求: 名称设置为: "\${caseSeq}_\${apiSeq}_\${apiName}" 动态取 CSV 文件 里面配置的参数信息; IP: "\${url}"; 方法: "\${function}"; 路径: "\${api}", 消息体数 据: "\${parameter}" 消息体数据和参数只能设置一种,不能两种同时设置。

🔝 🏶 🤐 🔚 🕺 🎼 🗐 🕂 😑	🦻 🕨 🕲 🕲 🖓 👹 📣 🏷 🔚 🔢 🏍	0:00:01 🚹 0 0/1
□	HTTP请求 名称: \${caseSeq}_\${apiSeq}_\${apiHame} 注释:	:
	自動意定向 又 復福 KeepAlive 所POST使用multipart / form-data 与浏览器集套的头 参数 消息体数据 文件上传 1 1 \${parameter} - - - - -	•

八、执行 CSV 文件, 查看结果

可以看到左边能够很清晰的反应接口执行的信息,右边可以很明确的反应接口返回的数据。并且 CSV 文件内可以设置众多接口的不同验证方式,很方便且很适合日常工作使用。希望以上内容能给大家提供帮助,谢谢。

□ _ 测试计划 □ ④ 线程组 □ □ 循环控制器 □ □ 循环控制器 CSV 数据文件设置 □ ↓ \${caseSeq}_\${apiNem	察看结果树 名称: 察看结果树 注释: 所有数据写入一个文件 文件名
■对企业级项目进行接口测试,	到底该做什么>> <u>https://dwz.cn/SoysVOny</u>



♦ 作者:黎晓萌

新人应该如何进入测试领域?

前些天有读者在后台问道,转行人员应该如何进入测试领域?

今天,我就这个问题做一个完整的回答,顺便说一下进入软件测试这个行业后持续 努力的方向,供各位想转行做测试还没转的、已经转行做测试但时间不久的、刚刚进入 测试领域的,这几类的小伙伴都可以参考下。对于其中有疑问的,也欢迎持续探讨。

一、软件测试的门槛

软件测试,作为软件研发流程中的一环,是软件研发质量的重要保证,而随着互联 网产业、软件等需求持续增加,这样的人才需求也持续旺盛,因此,有越来越多的人看 准这个机会想进入这个行业。一方面,这一行作为信息基础行业,有着相比于其他行业 更高的薪酬待遇;另一方面,随着信息技术更多的进入我们的生活,这方面的发展前景 也得到了大家的认可。所以,这一切也都不难理解。

那么,想从事这一行业,都需要做哪些准备呢?在我看来,**首先要了解的就是软件** 测试的基本概念和思想。软件测试在软件研发流程中处于怎样的过程,软件研发的基本 原理,软件测试的基本方法,这些都是需要了解和掌握的。只有清楚了这些,才能很好 的认识到软件测试是怎么一回事,才能够更好的参与到软件测试工作中,进而为软件的 研发质量保证贡献出自己的一份力量。

软件测试的基本概念和思想,在这里我就不展开讲了。我想讲的是,在我多年的测试管理工作中,面试了不少候选人,有很多候选人对软件测试的概念和方法都不清楚, 觉得软件测试就是将需求文档中的内容点点点,确定都实现了就算测试完成,这个回答 是很糟糕的。





作为软件测试行业的从业人员,我们必须清楚,**软件测试是需要用基本的测试方法 来保障软件的质量的。**还有很多面试者,谈起测试的基本方法来是头头是道,但真的给 了一个需求文档让他去设计测试用例时,就完全不管测试方法中交给我们的东西了,而 是天马行空的发散起来。这种理论与实际脱钩的情况,就会让面试官觉得还是没有很好 的掌握测试方法,经过他测试的软件产品的质量是不可控的,因此也会存在很大的质量 风险。

只有将软件测试的基本概念、方法进行很好的应用到日常工作中,只有能将我们在 候选人简历中经常看到的「熟练掌握软件测试方法」转换成真正的掌握,那在我看来才 是迈过了从事软件测试行业最基本的门槛。而要达到这一点,日常工作中积极的思考测 试方法与被测试软件的测试用例的对照关系,在日常使用软件的过程中培养测试用例设 计的意识,都能够很好的提高自己在这方面的能力。

二、测试的基本技能

在迈过软件测试的门槛后,接下来就是要掌握做测试的基本技能了。这个技能,包 括对**被测试软件的进一步了解和掌握,对测试工作中使用的测试工具的熟悉。**

在面试中,我们看到候选人的简历中提到自己测试项目的基本介绍,包括使用的语 言、框架,但真正问起来,对这些都没有基本的了解,那在我们看来就是不合适的。还 有候选人对自己测试的项目所在平台的一些基本常识的缺乏,在面试中也是很致命的。 比如,在面试中我经历过这样的候选人,做的是移动端的 APP 测试,但不清楚 APP 测 试的基本特点,需要关注哪些方面,兼容性测试怎么做等,这都是日常测试中很基础的 东西,如果都不能很好的讲出来,那就会说很糟糕的。

在我们日常的软件测试中,首先要关注的就是被测试软件所在平台的基本特点,是 Web 的还是 APP? Web 类的产品有哪些特点,实现的原理是怎样的,测试过程中需要考 虑哪些东西,承载 Web 类产品的浏览器是怎样的工作原理,做兼容性测试的时候需要关 注哪些方面。APP 类的产品有哪些特点,所在的系统有哪些特点,与终端系统有哪些交 互,需要考虑哪些方面的指标等。再此基础上,再去结合被测试软件的需求和功能特性 去进行测试用例的设计,功能特性有哪些,边界值法怎么用,等价类怎么划分,异常场 景有哪些,等等。

另外,我们在很多候选人的简历中看到「熟练使用 XXXX 测试工具」,但真的问起





来,这个工具的原理是怎么样的,都有哪些基本操作,可以辅助我们做哪些测试等,他 们就不知道了。更有甚者,只是知道点哪个按钮可以产生什么东西,多问一句产生的这 些东西是怎么得到的,都包括哪些信息,就不知道了。这些就比较不应该了。这样的 「熟练」更多的是对最常规、最基本使用的「重复性熟练」,而非真的对这个测试工具 有更深层次的了解和掌握。

在我看来,使用一个测试工具,这个**工具实现的基本原理**是怎样的,这个工具能做 什么,不能做什么,为什么能做这些,使用过程中能给我们提供哪些数据,我们可以用 这些数据做什么,如何辅助我们进行测试,都是我们在拿到一个工具时首先可以去了解 的。只有对一个工具有了这些了解,我们才能在日常的测试工作中更好的、更熟练的使 用,最大化的辅助我们的测试工作。

三、测试的能力提升

当我们对基本技能有了一定的掌握后,我们就需要进一步的提升自己的软件测试能力。

这时候对测试能力的定义和范围就可以进一步的扩大化,当我们测试一个软件时, 这个软件更底层的东西就需要我们去开始接触,使用的语言、框架的特点,软件的技术 架构等。这些对我们更好的了解被测试软件,发现其中存在的问题,都有很大的帮助。

对于较大型的软件,我们还需要了解整个系统的模块划分是怎样的,模块与模块间 的调用关系是怎样的,调用过程中是否困难存在异常,这里的用例该如何设计,对错误 处理的用例该怎么构造,软件发布上线后,对应的数据是否有上报,上报的问题反应了 怎样的问题,等等,都是我们需要去了解的。

在测试工具的使用上,也对我们有更高的要求,比如能实现类似功能的同类工具还 有哪些,为什么我们要用当前工具,这个工具能得到哪些信息,这些数据是怎么得到 的,准确率是多少,误差是否在我们可接受范围内,我们是否需要再此基础上去做优 化,如此等等。

另外,随着我们对软件测试的要求越来越高,测试的粒度越来越细,一些基本的测试、常规的工具不能满足时,我们是否能够开发出一些脚本、工具来满足我们的测试需要,我们是否能够持续用工具来解决测试中的问题,用工具实现测试效率的提升,测试 深度的加强,等等。





四、测试的职业发展

最后,简单谈下软件测试这个行业的职业发展。软件测试作为一个技术岗,不断的 提升自己的技术能力是最核心的发展要素。在这个大前提的指引下,不断的沉淀自己技 术领域的积累,扩充自己在技术解决测试问题上的知识面和能力,是关键。

从这个意义上讲,软件测试工程师的发展会从解决独立需求测试的初级工程师,向 能独立负责较大系统测试的中级工程师,到能完成测试体系的建设的高级工程师一步步 实现前进,在这个过程中,自己的技术能力也从对被测软件的基本实现,到掌握系统架 构,针对系统架构进行完整测试,从用小工具实现测试效率的改进,到用测试平台、测 试体系实现一整套测试架构的实现。

测试管理也是软件测试工程师发展的一个选项,从独立跟进需求测试,到带领小团队共同完成某个项目的测试,进而到带领更大的测试团队完成包括功能测试、工具建设等多领域的提升。

总之,软件测试是一个需要不断理解行业,不断提升自我,不断思考测试对于项目团队意义的工作。只有不断的前进,自己的职业生涯才会有更好的前途和更大的发展。

■新人也能学的测试, Python 购房宝典>><u>https://dwz.cn/VVNCrwox</u>

《51 测试天地》(五十六)下篇 精彩预览

- 使用 Locust 进行 APP 服务端并发测试浏览器网络捕获器在 python 网络编程中的辅助作用

- 测试女巫紧跟时代脉搏之大数据分析系列
- JMeter 扩展开发之 JMeter 核心源码解读
- nGrinder 压测工具使用教程
- Postman + Newman + Jenkins + 钉钉 实现持续集成的接口自动化
- WEB 自动化测试之元素定位自动生成探索实践
- 安全测试,我有话要说
- 自动化测试实现优劣浅谈



