**工作笔记**

**2013**

# 五月

## 5-18

### 1.git生成ssh key

a)设置git的user name和email：

$ git config --global user.name "xuhaiyan"

$ git config --global user.email "haiyan.xu.vip@gmail.com"

b)生成SSH密钥过程：  
查看是否已经有了ssh密钥：cd ~/.ssh 如果没有密钥则不会有此文件夹.

c)生成密钥：

$ ssh-keygen -t rsa -C “haiyan.xu.vip@gmail.com”  
按3个回车，密码为空。

Your identification has been saved in /home/tekkub/.ssh/id\_rsa.  
Your public key has been saved in /home/tekkub/.ssh/id\_rsa.pub.  
The key fingerprint is:  
………………

最后得到了两个文件：id\_rsa和id\_rsa.pub

d)在github上添加ssh密钥，这要添加的是“id\_rsa.pub”里面的公钥。

### 2. Chrome 恢复关闭网页的快捷键：ctrl+shift+T

### 3. git 忽略文件

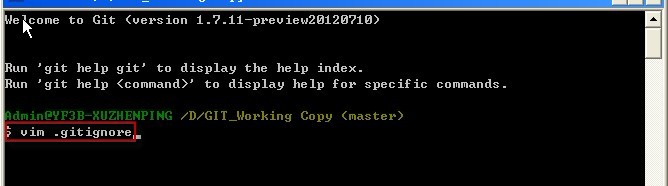
以下涉及的ignore文件均为如下格式：

|  |
| --- |
| # 以'#'开始的行，被视为注释.  # 忽略掉所有文件名是 foo.txt的文件.  foo.txt  # 忽略所有生成的 html文件,  \*.html  # foo.html是手工维护的，所以例外.  !foo.html  # 忽略所有.o和 .a文件.  \*.[oa] |

在仓库目录下新建一个名为.gitignore的文件（因为是点开头，没有文件名，没办法直接在windows目录下直接创建，必须通过右键Git Bash，按照linux的方式来新建.gitignore文件）。如下图所示。

.gitignore文件对其所在的目录及所在目录的全部子目录均有效。通过将.gitignore文件添加到仓库，其他开发者更新该文件到本地仓库，以共享同一套忽略规则。





例如，你要取消同步bin文件，只需在.gitignore中写入bin即可。

### 4. [git使用时遭遇the authenticity of host can't be established](http://blog.csdn.net/youxidonxx/article/details/7961597),一般要重新生成SSH.

### 5. git如何恢复本地删除的文件

直接从本地把文件checkout出来就可以了，用不着从远程服务器上pull下来，因为，所有的历史版本你的本地都有的。

具体做法 git checkout file

### 6. create a new repository:

git init

git add README.md

git commit -m "first commit"

git remote add origin git@github.com:xingyun890/worknote.git

git push -u origin master

## 5-29

### 1.SoapUI可以加载外部的类包(jar)，只需把jar放置到soapUI安装目录的bin/ext下，之后在需要用到的脚本处import即可。注意要重启SoapUI。

### 2.SoapUI脚本

* 使用XMLHolder

import com.eviware.soapui.support.XmlHolder

def xml = new XmlHolder(context.expand('${Pre Get Rack Collection#ResponseAsXml}'))

def node\_server\_namearray = xml.getNodeValues("//object[@class='ODB::Asset::VM']/@name") (或者getNodeValue)

* 统计个数

def node\_server\_count = context.expand( '${Pre Get Rack Collection#ResponseAsXml#count(//object[@class="ODB::Rack"]//object[@class="ODB::Node::Server" and type="esxserver"])}' )

* 截取关键字

Java语法：XXX.startsWith(“XXX”) XXX.endsWith(“XXX”)

Xpath语法：object[@class="ODB::ServiceInstance" and contains(fqdn, "phx")])

object[@class='ODB::ServiceInstance' and ends-with(@name,'-00001')]

object[@class='ODB::ServiceInstance' and starts-with(@name,'-00001')]

* 设置Property

testRunner.testCase.setPropertyValue("delete\_success", "false")

testRunner.testCase.getPropertyValue("delete\_success")

context.put("error\_node\_server\_list", error\_node\_server\_list)

context.get("error\_node\_server\_list")

使用context设值取值可以保存值的具体类型，如list。testRunner.testCase.setPropertyValue只能设置String类型。而且context设的值只在程序运行时有效，程序结束后就失效。

* 跳转

context.testRunner.gotoStepByName("End") 或

testRunner.gotoStepByName("End");

* Xpath中使用json，必须使用单引号括起来，不能使用双引号

def vpool = xml.getNodeValue("//object[@class='ODB::Node::Server' and @name='${node\_server\_name}']//object[@class='ODB::Vpool']/@name")

* 将String转化成JSON格式

import groovy.json.\*

node\_json = JsonOutput.toJson(node)

* Property Transfer这步可以将TestCase里的属性传递到request的body里。还可以一次传递多个参数到不同步骤里。
* 将Response转化成Json，并读取相应属性值

import groovy.json.\*

def appsvc\_cms = context.expand( '${Post Get CMS Appsvc#Response}' )

def appsvc\_json = new JsonSlurper().parseText(appsvc\_cms)

def env\_cms = appsvc\_json.result.get(0).environment.\_oid. get(0) 可能有多个值

### 3.SoapUI脚本完整实例

import com.eviware.soapui.support.XmlHolder

def xml = new XmlHolder(context.expand('${Pre Get Rack Collection#ResponseAsXml}'))

def node\_server\_namearray = xml.getNodeValues("//object[@class='ODB::Asset::VM']//object[@class='ODB::Node::Server']/@name")

def node\_server\_count = context.expand( '${Pre Get Rack Collection#ResponseAsXml#count(//object[@class="ODB::Rack"]//object[@class="ODB::Node::Server" and type="esxserver"]//object[@class="ODB::Asset::VM"]//object[@class="ODB::Node::Server"])}' )

def error\_node\_server\_list = new ArrayList()

for(i=0;i<Integer.parseInt(node\_server\_count);i++)

{

node\_server\_name = node\_server\_namearray.getAt(i)

def vpool = xml.getNodeValue("//object[@class='ODB::Node::Server' and @name='${node\_server\_name}']//object[@class='ODB::Vpool']/@name")

def application\_nugget = xml.getNodeValue("//object[@class='ODB::Node::Server' and @name='${node\_server\_name}']//object[@class='ODB::ApplicationService']/nugget")

def environment\_label = xml.getNodeValue("//object[@class='ODB::Node::Server' and @name='${node\_server\_name}']//object[@class='ODB::Environment']/label")

if(!(((application\_nugget!=null)&&(application\_nugget.startsWith("S1")||application\_nugget.startsWith("S2")||application\_nugget.startsWith("Q1")||application\_nugget.startsWith("Q2"))&&(vpool!=null))||(environment\_label!=null&&environment\_label.equals("IaaS\_Cache\_1"))))

{

error\_node\_server\_list.add( node\_server\_name)

}

}

context.put("error\_node\_server\_list", error\_node\_server\_list)

context.put("count", 0)

context.put("success\_count",0)

testRunner.testCase.setPropertyValue("delete\_success", "false")

log.info("error\_node\_server\_list:"+error\_node\_server\_list)

log.info("error\_node\_server\_list\_length:"+error\_node\_server\_list.size())

## 5-30

### 1.maven2发布自定义jar

如果repository里面没有需要的jar，或者是自定义的jar，需要发布到repository，则可以使用下面的命令：

**Java代码  [收藏代码](javascript:void())**

1. mvn install:install-file -Dfile=jsonplugin-0.32.jar -DgroupId=googlecode -DartifactId=googlecode-jsonplugin -Dversion=0.32 -Dpackaging=jar

这里假设需要发布的jar是当前目录下的jsonplugin-0.32.jar，目的repository(在环境变量中设置M2\_HOME环境变量)是D:/system/maven\_repo（如果要发布到远程服务器，则增加参数如“-DrepositoryId=local -Durl=D:/system/maven\_repo ”使用url，如http://192.168.1.111/repo，但是要把前面的-DrepositoryId=local修改为-DrepositoryId=remote，这里未验证）。

最后maven会在D:/system/maven\_repo里面生成文件：googlecode\googlecode-jsonplugin\0.32\googlecode-jsonplugin-0.32.jar。

# 六月

## 6-3

### 1.sublime 快捷键（很多都与eclipse相通）

* Shift + Ctrl + P 调出命令面板，可以实时搜索到相应的命令、选项、snippet 和 syntex
* Ctrl + P可以快速跳转到当前项目中的任意文件，可进行关键词匹配。
* Ctrl + P + # 可以在当前文件中进行搜索。
* Ctrl + P + @ 可以快速列出/跳转到某个函数。
* Ctrl + P + : + 数字 可以跳转到相应的行。
* 多重选择：
* Ctrl或 Alt，然后在页面中希望中现光标的位置点击。
* 选择数行文本，然后按下 Shift + Command + L。
* 通过反复按下 Ctrl/ Alt + D 即可将全文中与光标当前所在位置的词相同的词逐一加入选择或去除，而直接按下 Alt + F3即可一次性选择所有相同的词。
* 按下**鼠标中键**来进行垂直方向的纵列选择，也可以进入多重编辑状态。
* CTRL + SHIFT + f在指定目录下所有文件中搜索内容。

### 2.Sublime常用插件安装

* 安装Package Control：
* 按Ctrl+`调出console
* 粘贴以下代码到底部命令行并回车：  
  import urllib2,os;pf='Package Control.sublime-package';ipp=sublime.installed\_packages\_path();os.makedirs(ipp) if not os.path.exists(ipp) else None;open(os.path.join(ipp,pf),'wb').write(urllib2.urlopen('http://sublime.wbond.net/'+pf.replace(' ','%20')).read())
* 重启Sublime Text 2。
* 如果在Perferences->package settings中看到package control这一项，则安装成功。
* 用Package Control安装删除插件的方法：
* 按下Ctrl+Shift+P调出命令面板
* 输入install/remove 调出 Install/Remove Package 选项并回车，然后在列表中选中要安装或删除的插件。
* 语法提示插件

我们需要在写代码的时候如果有语法错误，能立即提示我们， 可以安装这两个插件：sublimelint 和Phpcs ， sublimeint 需要系统有php命令。 所以需要设置好php的环境变量。 sublimelint的语法错误提示是显示在状态栏上面的，所以在编写程序的时候注意时常看看状态栏。 而Phpcs的语法错误提示是在我们保存文件时弹出万能面板显示错误，sublimelint的错误提示实时但不明显。 Phpcs的错误提示不是实时的，但很明显。 因此我们一般这两个插件都要安装。

* Ctags插件

这个插件能跨文件跳转，跳转到指定函数声明的地方。 使用package control 搜索ctags 进行安装（安装ctags插件就可以了， 还有一个 CTags for PHP 插件没什么用）。注意安装好插件后要需要安装ctags命令。window 下载 ctags.exe  <http://vdisk.weibo.com/s/7QZd7> 。 将ctags.exe文件放在一个环境变量能访问到的地方。打开cmd， 输入ctags，如果有这个命令，证明成功了。 ubuntu下安装运行命令：sudo apt-get install exuberant-ctags 。 然后在sublime项目文件夹右键， 会出现Ctag:Rebuild Tags 的菜单。点击它，然后会生成.tags的文件。然后在你代码中， 光标放在某个函数上， 点击ctrl+shift+鼠标左键 就可以跳转到函数声明的地方。

* sublimecodeintel 代码提示

sublime默认的代码提示只能提示系统函数，用户自己创建的函数、类不能提示。 如果想要提示自己建立的函数。 可以安装sublimecodeintel插件。

sublimecodeintel 安装后需要配置，文件：插件目录/.codeintel/config 中 增加

1. "PHP": {
2. "php": 'D:\SaeServer\php\php.exe',
3. "phpExtraPaths": ['D:\SaeServer\php\stdlib'],
4. "phpConfigFile": 'D:\SaeServer\apache\php.ini'
5. },

配置了php执行文件的地址， php的配置文件地址， phpExtraPaths 是额外需要代码提示的类库，除了当前项目下的PHP代码可以提示外 phpExtraPaths中定义的目录下的PHP代码也能提示。  D:\SaeServer\php\stdlib 是SaeServer中 SAE本地模拟文件的目录， 所以配置后不管在哪儿 都能有SAE代码的提示。

安装sublimecodeintel后， 按alt+鼠标左键也能和ctags一样跳转到函数声明的地方。 但是如果有两个文件声明了同样名称的函数， sublimecodeintel只会跳转到第一个找到的函数， 而ctags会让你选择要跳转到哪个文件。所以我们一般还是用ctags的跳转功能。

### 3.[eclipse 多项目依赖](http://xuliangyong.iteye.com/blog/551303)可以使用link source

项目越来越大会分割成多个子项目 。主项目依赖多个子项目 。eclipse如何实现呢？

有两个项目 master 和 sub ，要在master中引用sub，通常会打一个 sub.jar，然后再master中引用 。如果sub项目代码经常改动打sub.jar就成了负担 。eclipse中可以这样实现 。

右键master项目--> build path-->source-->link source... 在弹出的对话框中选择sub项目的源文件目录即可 。

具体步骤：properties -->java build path --> Source 里面有个Link Source选项，就是添加项目需要的附加关联代码，进去后，可以直接浏览文件夹，也可以设置Source路径变量。然后把你的共享项目的Source路径加进去，确定就ok了。这样系统会把这些代码的.class发布到当前工程的发布目录下。最后不要忘了**build path –>use as source folder**转换成包的形式。

## 6-6

### 1.Groovy正则表达式

Groovy中，您可用斜线语法“/../”定义字符串。这样以来便可避免在Java正则表达式中使用过多的反斜线。

.代表匹配任意一个字符

\d任意一个数字，0-9 中的任意一个

* 使用模式操作符~来创建Pattern对象，它使得字符串被 编译成一个Pattern实例。

def reg = ~'abc\\d' 或def reg = ~/abc\d/ (注意 = 与 ~ 之间有空格)

* 使用查找操作符=~来获得Matcher对象，它负责从左边的字符串和右边的模式来创 建一个Matcher对象。 (注意 = 与 ~ 之间不能有空格)

import java.util.regex.Matcher

def m='abc1'=~/abc\d/

assert m.matches()==true

其Java实如下：

import　java.util.regex.\*;

Pattern　pattern　=　Pattern.compile("abc\d ");

Matcher　matcher　=　pattern.matcher("abc1");

matcher.matches();

* 使用==~来判断是否匹配成功。该操作符返回一个Boolean值，说明操作符左侧的整个字符串是否右侧的模式相匹配。 (注意 == 与 ~ 之间不能有空格)

assert 123 ==~/\d+/

assert "Groovy" ==~ /G.\*/

assert !123 ==~ /\D+/