链滴

如何发布自己的 jar 包到 maven 中央仓库

作者：xjtushilei
原文链接：https://ld246.com/article/1492069833312
来源网站：链滴

原文来自 xjtushilei

偶然看到了个软件设计大赛，就设计了个通用的爬虫框架，想做到java爬虫会优先考虑我的框架。这任务还需要很长的时间去努力！

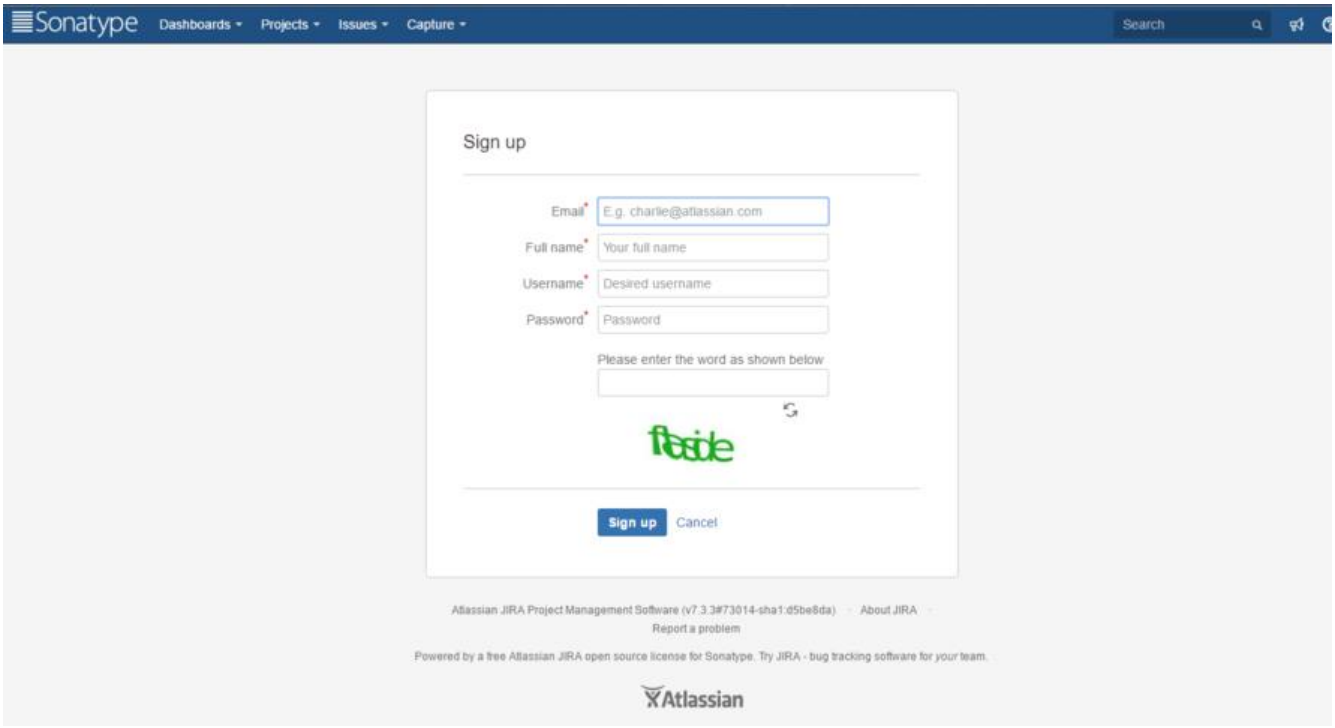
当然了，为了让别人用，肯定要发布到中央仓库的。

准备开车！

注册

首先到sonatype官网进行注册！

找到注册“sign up”，如图，进行注册！

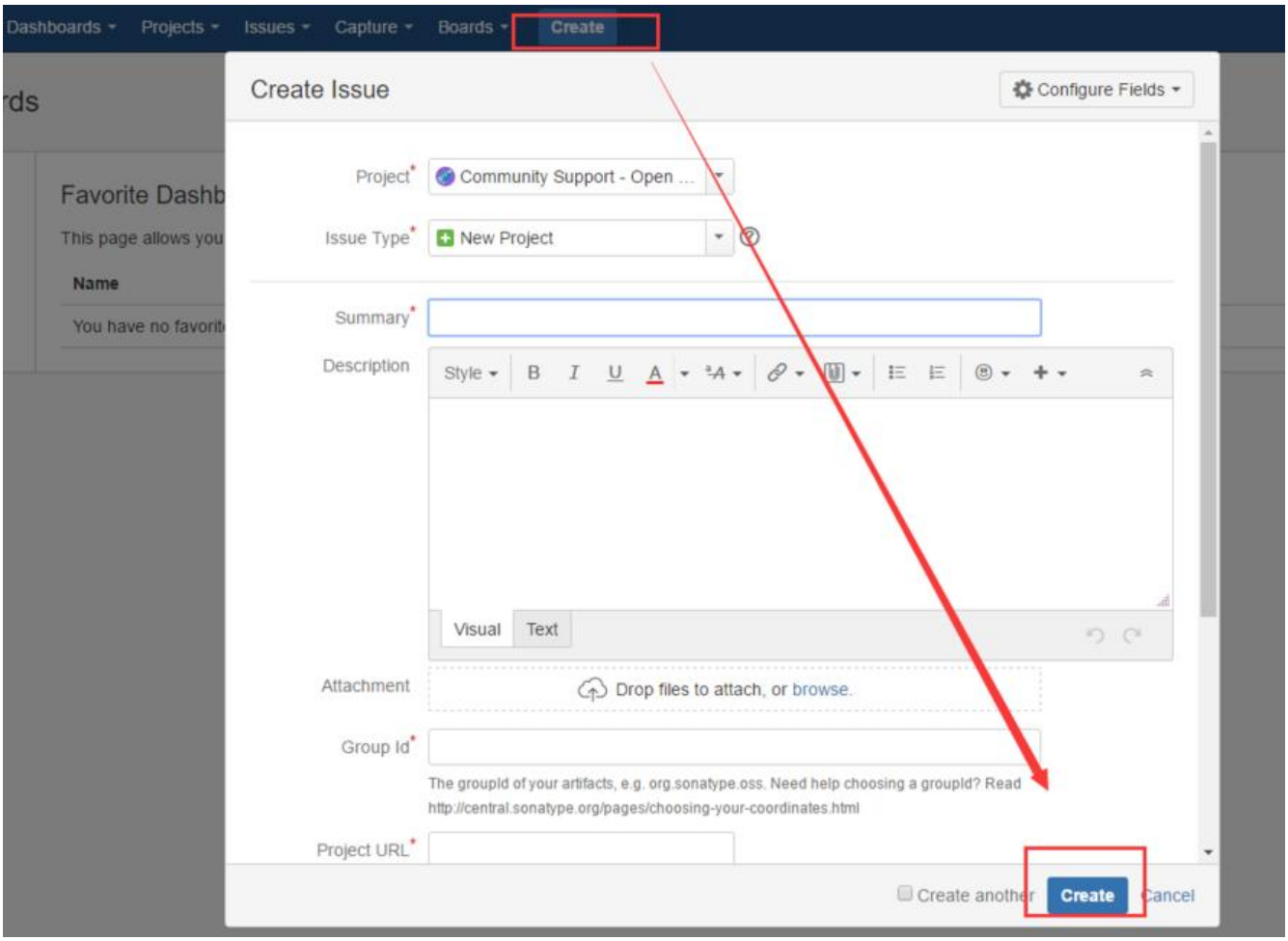


登录

登录后，在页面上方点击“create”进行创建issue。

获得如下图所示。

原文链接：如何发布自己的 jar 包到 maven 中央仓库



- Summary：写你想做什么，你的包的功能。简单概述，要短一点。
- Description：可以直接复制summary，然后加一点描述信息。
- Group Id ： 推荐写github。例如我的是 **com.github.xjtushilei** ，很快能通过，自己域名的我没试过，虽然我有自己的域名。
- Project UR : 你的项目的描述，填写你的项目的github地址就可以了。
- SCM url: 有填写说明的。
- Username(s)：邀请其他人有权限一起完成这个项目。我当时没填。

等待 Issue 审批通过

网上说需要一天到两天。为何我这一步是秒过的。大概20秒过后我就接收到通过邮件了。

或许是我填写的github比较好认真真实性。

配置GPG

如果是 Windows 操作系统，需要下载 Gpg4win 软件来生成密钥对。建议大家下载 Gpg4win-Vanill 版本，因为它仅包括 GnuPG，这个工具才是我们所需要的。

安装 GPG 软件后，打开命令行窗口，依次做以下操作：

1. 查看是否安装成功

```
gpg --version
```

能够显示 GPG 的版本信息，说明安装成功了。

2. 生成密钥对

```
gpg --gen-key
```

此时需要输入姓名、邮箱等字段，其它字段可使用默认值，此外，还需要输入一个 Passphase，相当一个密钥库的密码，一定不要忘了，也不要告诉别人，最好记下来，因为后面会用到。

3. 查看公钥

```
gpg --list-keys
```

如下：

```
D:\IdeaProjects\ScriptSpider>gpg --list-keys
C:/Users/shilei/AppData/Roaming/gnupg/pubring.gpg
------------------------------------------------
pub   2048R/CAD14E5B 2017-04-11
uid       [ultimate] shilei
sub   2048R/91AAE050 2017-04-11
```

我的公钥是：**CAD14E5B**

4. 将公钥发布到 PGP 密钥服务器

```
gpg --keyserver hkp://pool.sks-keyservers.net --send-keys CAD14E5B
```

此后，可使用本地的私钥来对上传构件进行数字签名，而下载该构件的用户可通过上传的公钥来验证名，也就是说，大家可以验证这个构件是否由本人上传的，因为有可能该构件被坏人给篡改了。

5. 查询公钥是否发布成功

```
gpg --keyserver hkp://pool.sks-keyservers.net --recv-keys CAD14E5B
```

实际上就是从 key server 上通过公钥 ID 来接收公钥，此外，也可以到 sks-keyservers.net 上通过公 ID 去查询。

修改 Maven 配置文件

```
...
<servers>
   <server>
   <id>oss</id>  //这里要和后面呼应
   <username>用户名</username>
   <password>密码</password>
   </server>
</servers>
...
```

原文链接：如何发布自己的 jar 包到 maven 中央仓库

修改你工程的pom文件

你可以增加你自己其他的，这个是我当时的pom。

```
    <groupId>com.github.xjtushilei</groupId>
    <artifactId>scriptspider</artifactId>
    <version>0.1.1</version>
    <packaging>jar</packaging>

    <name>scriptspider</name>
    <description>ScriptSpider(SS) is a Java distributed crawler framework that supports hot sw
ppable components.SS是一个java版本的分布式的通用爬虫，可以热插拔各个组件（提供默认的）
自动切换代理，自动结构化数据与存储。使用redis，分布式调度等技术。
    </description>
    <url>https://github.com/xjtushilei/ScriptSpider</url>

    <licenses>
      <license>
        <name>The Apache Software License, Version 2.0</name>
        <url>http://www.apache.org/licenses/LICENSE-2.0.txt</url>
      </license>
    </licenses>

    <developers>
      <developer>
        <name>shilei</name>
        <email>xjtushilei@foxmail.com</email>
      </developer>
    </developers>

    <scm>
      <connection>
        scm:git:https://github.com/xjtushilei/ScriptSpider.git
      </connection>
      <developerConnection>
        scm:git:https://github.com/xjtushilei/ScriptSpider.git
      </developerConnection>
      <url>https://github.com/xjtushilei/ScriptSpider</url>
    </scm>

    <properties>
      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
      <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
      <java.version>1.8</java.version>
    </properties>

    <dependencies>

      。。。


    </dependencies>
```

原文链接：如何发布自己的 jar 包到 maven 中央仓库

```
<build>
  <plugins>
    <plugin>
      <artifactId>maven-compiler-plugin</artifactId>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-javadoc-plugin</artifactId>
      <configuration>
        <aggregate>true</aggregate>
      </configuration>
    </plugin>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-gpg-plugin</artifactId>
    </plugin>
  </plugins>
</build>

<profiles>
  <profile>
    <id>release</id>
    <build>
      <plugins>
        <!-- Source -->
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-source-plugin</artifactId>
          <executions>
            <execution>
              <phase>package</phase>
              <goals>
                <goal>jar-no-fork</goal>
              </goals>
            </execution>
          </executions>
        </plugin>
        <!-- Javadoc -->
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-javadoc-plugin</artifactId>
          <executions>
            <execution>
              <phase>package</phase>
              <goals>
                <goal>jar</goal>
              </goals>
            </execution>
          </executions>
```

```
            </plugin>
            <!-- GPG -->
            <plugin>
              <groupId>org.apache.maven.plugins</groupId>
              <artifactId>maven-gpg-plugin</artifactId>
              <executions>
                <execution>
                  <phase>verify</phase>
                  <goals>
                    <goal>sign</goal>
                  </goals>
                </execution>
              </executions>
            </plugin>
          </plugins>
        </build>
        <distributionManagement>
          <snapshotRepository>
            <id>oss</id>
            <url>https://oss.sonatype.org/content/repositories/snapshots/</url>
          </snapshotRepository>
          <repository>
            <id>oss</id>
            <url>https://oss.sonatype.org/service/local/staging/deploy/maven2/</url>
          </repository>
        </distributionManagement>
      </profile>
    </profiles>
```

上传构件到 OSS 中

mvn clean deploy -P release

或者

mvn clean deploy -P release -Dgpg.passphrase=密码

第一个会弹出来让你输入密码，第二个是告诉它你的密码。Passphase就是GPG 密钥对的密码，只有己才知道。在之前的步骤中已经告诉你要牢记了。

随后会看到大量的 upload 信息，而且速度比较慢，如果出现timeout，需要反复尝试。

在 OSS 中发布构件

在 OSS 中，使用自己的 Sonatype 账号登录后，再左侧的 Staging Repositories 中，在最下方可以到刚才已上传的构件，这些构件目前是放在 Staging 仓库中，可进行模糊查询，快速定位到自己的构。此时，该构件的状态为 Open，需要勾选它，然后点击 Close 按钮。接下来系统会自动验证该构件否满足指定要求，当验证完毕后，状态会变为 Closed，最后，点击 Release 按钮来发布该构件。

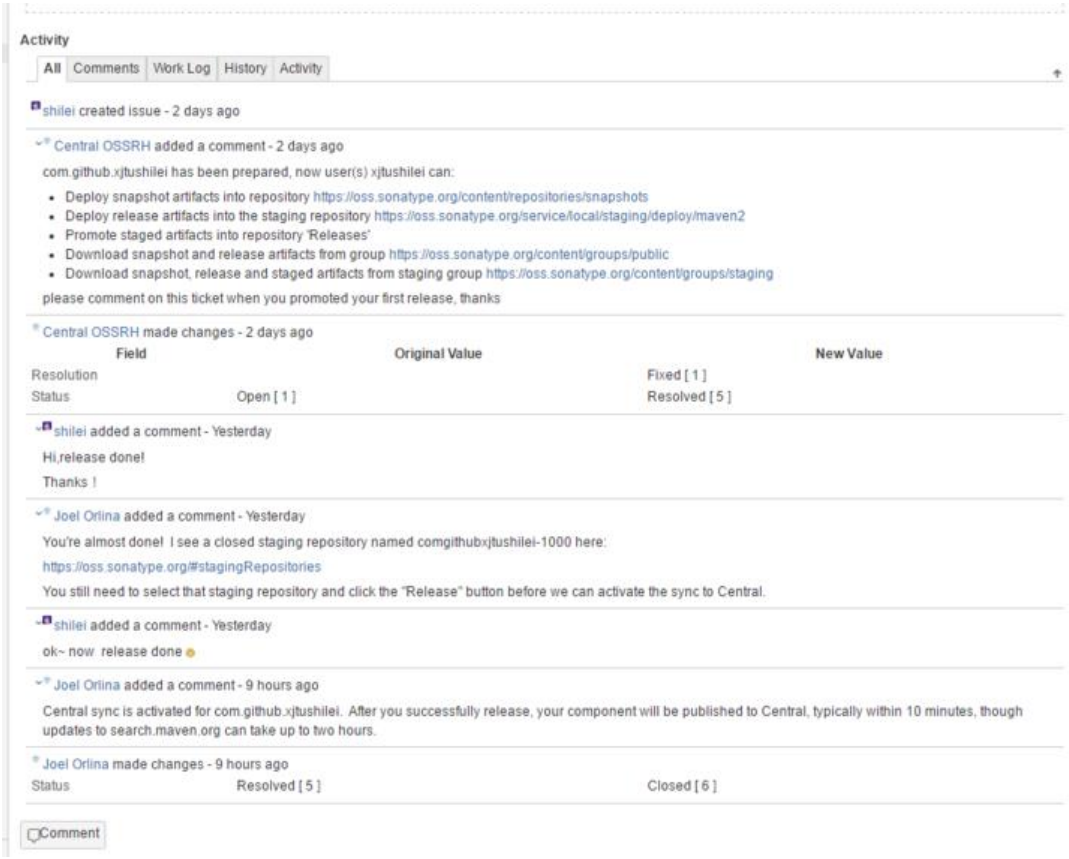
这一步我忘记了点release就进行了下一步，当然，我得到了一个回复。不得不说，他们的回复还是态很好！

通知 Sonatype“构件已成功发布”

需要在曾经创建的 Issue 下面回复一条“构件已成功发布”的评论，这是为了通知 Sonatype 的工作员为需要发布的构件做审批，发布后工作人员会关闭该 Issue。

我的 issue 展示

通过该截图，你应该知道我们都做了什么。



如果是发布的话，请注意artifactId不能是 **-SNAPSHOT**

因为从截图看（经过证明），一旦你的issue第一次被批准，你就可以发布**-SNAPSHOT**了，而且我时尝试了使用我的**-SNAPSHOT**版本的jar，并不需要进行后续操作。

等待构件审批通过

没错，还是要等，我大概等了一个多小时吧。感觉他们的效率还是很高，没有我查到的别人用一到两。

从中央仓库中搜索构件

最后，就可以到中央仓库中搜索到自己发布的构件了！

中央仓库搜索网站：http://search.maven.org/

等多长时间呢？看我的邮件列表的回复。意思大概是10分钟就可以使用了，2小时才能被搜索到。我用镜像的应该更慢，需要同步的。

原文链接：如何发布自己的 jar 包到 maven 中央仓库

After you successfully release, your component will be published to Central, typically within 10 minutes, though updates to search.maven.org can take up to two hours.

第一次之后

你就可以安心的开车了~

以后的发布流程是：

1. 构件准备好之后，在命令行上传构建
2. 在 https://oss.sonatype.org/ “close”并“release”构件
3. 等待同步好（大约2小时多）之后，就可以使用了