做 Nutch 二次开发，开发阶段用什么操作系统都可以，只要有 JDK 和 Eclipse 即可，源代码的管理需要使用一个集中的版本控制系统，可以使用 SVN 或 GIT，建议采用 Bitbucket 免费的私有库托管。如果想阶段性地在 Hadoop 集群上面试运行，需要搭建一个 Hadoop 伪分布式集群或完全分布式集群，开发组可以共用一个集群。

## 1、下载并解压 eclipse（集成开发环境）

使用 Standard 版

下载地址：Eclipse Standard 4.3.2 For Windows 64 Bit
Eclipse Standard 4.3.2 For Windows 32 Bit
其他操作系统版本

## 2、安装 Subclipse 插件（SVN 客户端）

Help > Install new software… > Add… > Name:subclipse > Location: http://subclipse.tigris.org/update_1.8.x > 选中 **Subclipse** and **SVNKit** > Next > Next >选中 **I accept** … > Finish > continue? OK > restart? > Yes

## 3、安装 IvyDE 插件（下载依赖 Jar）

Help > Install new software… > Add… > Name:ivyde > Location: http://www.apache.org/dist/ant/ivyde/updatesite/ >选中 **Apache Ivy Library** and **Apache IvyDE Eclipse plugins** > Next > Next >选中 **I accept** … > Finish > continue? OK > restart? > Yes

## 4、签出代码

File > New > Project > SVN > 从 SVN 检出项目 > Next >选中 创建新的资源库位置 > Next > URL：https://svn.apache.org/repos/asf/nutch/tags/release-1.7/ > Next >选中 URL > Finish > 弹出 New Project 向导，选择 Java Project > Next > 输入 Project name：nutch1.7 > Finish > 确认覆盖？OK

## 5、配置文件编码和环境变量

在左部 Package Explorer 的 nutch1.7 文件夹上单击右键 > Properties > 左边选中 Resource > 右边 Text file encoding > 选中 other > 值为:UTF-8

在左部 Package Explorer 的 nutch1.7 文件夹上单击右键 > Build Path > Configure Build Path... > 选中 Source 选项 > 选择 src > Remove > Add Folder... > 选择 src/java, src/test 和 src/testresources（可选：对于插件，需要选中 src/plugin 目录下的每一个插件目录下的 src/java ， src/test 文件夹） > OK

切换到 Libraries 选项 >

Add Class Folder... > 选中 nutch1.7/conf > OK

Add Library... > IvyDE Managed Dependencies > Next > Main > Ivy File: > Project… > 选中 nutch1.7/ivy/ivy.xml > OK > Finish（可选：对于插件，还需要：Add JARs... > IvyDE Managed Dependencies > Next > Main > Ivy File: > Project… > 选中 nutch1.7/src/plugin/xxx/ivy.xml > OK，把这里的 xxx 替换为相应的插件名称）

切换到 Order and Export 选项>

选中 conf > Top（重要!）

## 6、执行 ANT 下载依赖构建项目

在左部 Package Explorer 的 nutch1.7 文件夹下的 build.xml 文件上单击右键 > Run As > Ant Build > BUILD SUCCESSFUL > 生成 nutch1.7/build/apache-nutch-1.7.job

在 nutch1.7\runtim 目录下生成两个目录 deploy 和 local，他们用于生产环境，跟二次开发没什么关系，deploy 依赖 Hadoop，local 不依赖 Hadoop

在左部 Package Explorer 的 nutch1.7 文件夹上单击右键 > Refresh

在左部 Package Explorer 的 nutch1.7 文件夹上单击右键 > Build Path > Configure Build Path... > 选中 Libraries 选项 > Add Class Folder... > 选中 build > OK

## 7、修改配置文件

如 nutch1.7/conf 下不存在 nutch-site.xml，则将 nutch1.7/conf/nutch-site.xml.template 复制一份改名为 nutch1.7/conf/nutch-site.xml

如 nutch1.7/conf 下不存在 regex-urlfilter.txt，则将 nutch1.7/conf/regex-urlfilter.txt.template 复制一份改名为 nutch1.7/conf/regex-urlfilter.txt

如新增了文件，则在左部 Package Explorer 的 nutch1.7 文件夹上单击右键 > Refresh

配置 nutch-site.xml，新增：

```
<property>
    <name>http.agent.name</name>
    <value>nutch-crawler</value>
</property>
<property>
    <name>http.content.limit</name>
    <value>-1</value>
</property>
<property>
    <name>db.max.outlinks.per.page</name>
    <value>10000</value>
</property>
```

配置 regex-urlfilter.txt，将

```
# accept anything else
+.
```

替换为：

```
+^http://([a-z0-9]*\.)*news.163.com/
-.
```

## 8、开发运行调试

在左部 Package Explorer 的 nutch1.7 文件夹上单击右键 > New > Folder > Folder name：urls

在刚新建的 urls 目录下新建一个文本文件 url，文本内容为：http://news.163.com

打开 src/java 下的 org.apache.nutch.crawl.Crawl.java 类，单击右键 Run As > Java Application > Console 显示：Usage: Crawl <urlDir> -solr <solrURL> [-dir d] [-threads n] [-depth i] [-topN N]

在 Crawl 类上重新单击右键 Run As > Run Configurations… > Arguments > 在 Program arguments 输入框中输入：urls -dir data -depth 3 > Run

在 windows 环境中如果抛出异常 Failed to set permissions of path，则需要下载修改过的 hadoop，替换 nutch 依赖的 hadoop。下载地址：http://pan.baidu.com/s/1o60QtD4 ，因为 nutch1.7 依赖的 hadoop 版本为 1.2.0，提供下载的是 1.2.1，所以将下载的 hadoop 拷贝到 ivy 本地存储库 C:/Users/ysc/.ivy2/cache/org.apache.hadoop/hadoop-core/jars 目录，删除原来的 1.2.0，将 1.2.1 改为 1.2.0。

在需要调试的地方打上断点 Debug As > Java Applicaton

## 9、查看结果

查看 segments 目录：

打开 src/java 下的 org.apache.nutch.segment.SegmentReader.java 类

单击右键 Run As > Java Applicaton，控制台会输出该命令的使用方法

单击右键 Run As > Run Configurations > Arguments > 在 Program arguments 输入框中输入：-dump data/segments/* data/segments/dump

用文本编辑器打开文件 data/segments/dump/dump 查看 segments 中存储的信息

查看 crawldb 目录：

打开 src/java 下的 org.apache.nutch.crawl.CrawlDbReader.java 类

单击右键 Run As > Java Applicaton，控制台会输出该命令的使用方法

单击右键 Run As > Run Configurations > Arguments > 在 Program arguments 输入框中输入：data/crawldb -stats

控制台会输出 crawldb 统计信息

查看 linkdb 目录：

打开 src/java 下的 org.apache.nutch.crawl.LinkDbReader.java 类

单击右键 Run As > Java Applicaton，控制台会输出该命令的使用方法

单击右键 Run As > Run Configurations > Arguments > 在 Program arguments 输入框中输入：data/linkdb -dump data/linkdb_dump

用文本编辑器打开文件 data/linkdb_dump/part-00000 查看 linkdb 中存储的信息

## 10、全网分步骤抓取(可选)

在左部 Package Explorer 的 nutch1.7 文件夹下的 build.xml 文件上单击右键 > Run As > Ant Build

```
cd  /home/ysc/workspace/nutch1.7/runtime/local
#准备 URL 列表
wget http://rdf.dmoz.org/rdf/content.rdf.u8.gz
gunzip content.rdf.u8.gz
mkdir dmoz
bin/nutch org.apache.nutch.tools.DmozParser content.rdf.u8 -subset 5000 > dmoz/url
#注入 URL
bin/nutch inject crawl/crawldb dmoz
#生成抓取列表
bin/nutch generate crawl/crawldb crawl/segments
#第一次抓取,s1 是最新产生的 segment
s1=`ls -d crawl/segments/2* | tail -1`
echo $s1
#抓取网页
bin/nutch fetch $s1
#解析网页
bin/nutch parse $s1
#更新 URL 状态
bin/nutch updatedb crawl/crawldb $s1
```

```
#第二次抓取,生成抓取列表，s2 是最新产生的 segment
bin/nutch generate crawl/crawldb crawl/segments -topN 1000
s2=`ls -d crawl/segments/2* | tail -1`
echo $s2
bin/nutch fetch $s2
bin/nutch parse $s2
bin/nutch updatedb crawl/crawldb $s2
#第三次抓取,生成抓取列表，s3 是最新产生的 segment
bin/nutch generate crawl/crawldb crawl/segments -topN 1000
s3=`ls -d crawl/segments/2* | tail -1`
echo $s3
bin/nutch fetch $s3
bin/nutch parse $s3
bin/nutch updatedb crawl/crawldb $s3
#生成反向链接库
bin/nutch invertlinks crawl/linkdb -dir crawl/segments
#建索引
bin/nutch solrindex http://localhost:8983/solr/collection1 data/crawldb -
linkdb data/linkdb -dir data/segments
```

bin/crawl 提供了更简单的增量抓取脚本

## 11、索引和搜索

下载解压 solr，为了查看索引文件的格式，使用 solr-4.6.1

下载地址： http://pan.baidu.com/s/1hqxEFxq

#配置 solr core

复制 nutch 的 conf 目录中的 schema-solr4.xml 文件到 solr-4.6.1/example/solr/collection1/conf 目录，覆盖名为 schema.xml 的文件

修改 solr-4.6.1/example/solr/collection1/conf/schema.xml，在<fields>下增加：<field name="_version_" type="long" indexed="true" stored="true"/>

#配置中文分词

下载中文分词依赖的 Jar：http://pan.baidu.com/s/1i37gcg1

创建目录 solr-4.6.1/example/solr/lib，并将下载下来的压缩文件中的 3 个 jar 文件提取出来放到该目录

修改文件 solr-4.6.1/example/solr/collection1/conf/schema.xml

将字段类型 text_general 的 analyzer 的 index 和 query 的 tokenizer 分别改为：

<tokenizer class="org.ansj.solr.AnsjTokenizerFactory" conf="ansj.conf"/>

和

<tokenizer class="org.ansj.solr.AnsjTokenizerFactory" analysisType="1"/>

创建文件 solr-4.6.1/example/solr/collection1/conf/ansj.conf，输入：

```
lastupdate=123
files=dic/customDic.txt
```

创建文件 dic/customDic.txt，这就是自定义用户词典

#启动 SOLR 服务器

运行 Jar 文件：solr-4.6.1/example/start.jar

#浏览器管理界面

http://localhost:8983/solr
http://localhost:8983/solr/#/colletion1

## 12、查看索引信息

下载 **Luke**(Lucene Index Toolbox)：http://pan.baidu.com/s/1bn6CuQV

将索引文件 solr-4.6.1/example/solr/collection1/data/index 复制一份到其他目录，删除文件 write.lock

File > Open Lucene Index > 选择复制的索引文件路径

## 13、插件开发

13.1 复制 nutch1.7/src/plugin/parse-html，重命名为 parse-jsoup，去掉对 lib-nekohtml 的依赖，修改 build.xml、ivy.xml 和 plugin.xml，依赖的 jar 包 jsoup 配置到 ivy.xml

13.2 修改 nutch1.7/src/plugin/build.xml，加入新的插件 parse-jsoup 的配置，以便构建的时候能编译新的插件，有 3 个配置项，参考 parse-html

13.3 将新插件加入 Build Path（Source 和 Libraries）

13.4 修改 nutch1.7/conf/parse-plugins.xml，加入 mimeType 和 alias