CrossPlatformNutchScripts

The shell scripts included with Nutch are great, but require a shell. Some IT departments may not allow cygwin to be installed on their windows clients. Alternatives (well, at least one) are listed below.

A quick python port of the 'nutch' shell script (as of 9-11-06 it has only been tested in windows):

```
import os, sys, glob

# The Nutch command script
#
# Environment Variables
#
#   NUTCH_JAVA_HOME The java implementation to use.  Overrides JAVA_HOME.
#
#   NUTCH_HEAPSIZE  The maximum amount of heap to use, in MB.
#                   Default is 1000.
#
#   NUTCH_OPTS      Extra Java runtime options.
#
# ported to python by Ben Ogle (ogle dot ben [at] gmail)

#does not handle links.
thisdir = os.getcwd()
cpsep = ":"
if( os.name == "nt" ):
    cpsep = ";"

if( len(sys.argv) == 1 ):
    print "Usage: python nutch.py COMMAND"
    print "where COMMAND is one of:"
    print "  crawl             one-step crawler for intranets"
    print "  readdb            read / dump crawl db"
    print "  mergedb           merge crawldb-s, with optional filtering"
    print "  readlinkdb        read / dump link db"
    print "  inject            inject new urls into the database"
    print "  generate          generate new segments to fetch"
    print "  fetch             fetch a segment's pages"
    print "  parse             parse a segment's pages"
    print "  segread           read / dump segment data"
    print "  mergesegs         merge several segments, with optional filtering and slicing"
    print "  updatedb          update crawl db from segments after fetching"
    print "  invertlinks       create a linkdb from parsed segments"
    print "  mergelinkdb       merge linkdb-s, with optional filtering"
    print "  index             run the indexer on parsed segments and linkdb"
    print "  merge             merge several segment indexes"
    print "  dedup             remove duplicates from a set of segment indexes"
    print "  plugin            load a plugin and run one of its classes main()"
    print "  server            run a search server"
    print " or"
    print "  CLASSNAME         run the class named CLASSNAME"
    print "Most commands print help when invoked w/o parameters."
    sys.exit(1)

command = sys.argv[1]
#print "COMMAND: " + command

nutch_home = thisdir + "/.."

java_home = os.getenv("NUTCH_JAVA_HOME")
if(java_home != None):
    os.setenv("JAVA_HOME", java_home)
    print java_home

java_home = os.getenv("JAVA_HOME")
if(java_home == None):
    print "Error: JAVA_HOME is not set."
    exit(1)
```

```
java = java_home + "/bin/java.exe"

java_heap_max = "-Xmx1000m"
nutch_heap_sz = os.getenv("NUTCH_HEAPSIZE")
if(nutch_heap_sz != None):
    java_heap_max = "-Xmx"+ nutch_heap_sz +"m"
#print java_heap_max

classpath = nutch_home + "/conf"
classpath = classpath + cpsep + nutch_home + "/lib/tools.jar"

# for developers, add plugins, job & test code to CLASSPATH
if( os.path.exists( nutch_home + "/build/plugins" ) ):
    classpath = classpath + cpsep + nutch_home + "/build/plugins"

flist = glob.glob(nutch_home + "/build/nutch-*.job")
for l in flist:
    classpath = classpath + cpsep + l

if( os.path.exists( nutch_home + "/build/test/classes" ) ):
    classpath = classpath + cpsep + nutch_home + "/build/test/classes"

flist = glob.glob(nutch_home + "/nutch-*.job")
for l in flist:
    classpath = classpath + cpsep + l

if( os.path.exists( nutch_home + "/plugins" ) ):
    classpath = classpath + cpsep + nutch_home + "/plugins"

flist = glob.glob(nutch_home + "/lib/*.jar")
for l in flist:
    classpath = classpath + cpsep + l

flist = glob.glob(nutch_home + "/lib/jetty-ext/*.jar")
for l in flist:
    classpath = classpath + cpsep + l

#print classpath

nutch_log_dir = os.getenv("NUTCH_LOG_DIR")
if(nutch_log_dir == None):
    nutch_log_dir = nutch_home + "/logs"

nutch_log_file = os.getenv("NUTCH_LOGFILE")
if(nutch_log_file == None):
    nutch_log_file = "hadoop.log"

nutch_opts = os.getenv("NUTCH_OPTS")
if( nutch_opts == None ):
    nutch_opts = ""
nutch_opts = nutch_opts + " -Dhadoop.log.dir=" + nutch_log_dir
nutch_opts = nutch_opts + " -Dhadoop.log.file=" + nutch_log_file

# figure out which class to run
theclass = command
if ( command == "crawl" ):
    theclass="org.apache.nutch.crawl.Crawl"
elif ( command == "inject" ):
    theclass="org.apache.nutch.crawl.Injector"
elif ( command == "generate" ):
    theclass="org.apache.nutch.crawl.Generator"
elif ( command == "fetch" ):
    theclass="org.apache.nutch.fetcher.Fetcher"
elif ( command == "parse" ):
    theclass="org.apache.nutch.parse.ParseSegment"
elif ( command == "readdb" ):
    theclass="org.apache.nutch.crawl.CrawlDbReader"
elif ( command == "mergedb" ):
    theclass="org.apache.nutch.crawl.CrawlDbMerger"
elif ( command == "readlinkdb" ):
```

```
    theclass="org.apache.nutch.crawl.LinkDbReader"
elif ( command == "segread" ):
    theclass="org.apache.nutch.segment.SegmentReader"
elif ( command == "mergesegs" ):
    theclass="org.apache.nutch.segment.SegmentMerger"
elif ( command == "updatedb" ):
    theclass="org.apache.nutch.crawl.CrawlDb"
elif ( command == "invertlinks" ):
    theclass="org.apache.nutch.crawl.LinkDb"
elif ( command == "mergelinkdb" ):
    theclass="org.apache.nutch.crawl.LinkDbMerger"
elif ( command == "index" ):
    theclass="org.apache.nutch.indexer.Indexer"
elif ( command == "dedup" ):
    theclass="org.apache.nutch.indexer.DeleteDuplicates"
elif ( command == "merge" ):
    theclass="org.apache.nutch.indexer.IndexMerger"
elif ( command == "plugin" ):
    theclass="org.apache.nutch.plugin.PluginRepository"
elif ( command == "server" ):
    #what goes in place of the $Server?
    theclass="org.apache.nutch.searcher.DistributedSearch$Server"

args = ""
for i in range(2, len(sys.argv)):
    args = args + " " + sys.argv[i]

#windows doesnt like this even though there are quotes around it...
#"\"" + java +"\" "
cmdtorun = "java " + java_heap_max + " " + nutch_opts + " -classpath \"" + classpath + "\" " + theclass + args

#print cmdtorun
os.system(cmdtorun)
```