# Cocoon2.1m2SetupLinuxredhat8.0

**Warning**: many pages on this wiki imply that a servlet container must be installed before running Cocoon, this is **wrong** - see NoServletContainerRequired. A basic installation of Cocoon for test or development purposes can be *very simple*.

## Cocoon2.1m2 Setup Linux redhat8.0

### How to setup Cocoon 2.1m2 + Java SDK 1.4.1_02 + Tomcat/4.1.24-LE-jdk14 + on a RedHat8.0 linux box

Note this is basically a copy paste the edit version of Cocoon 2.1DevSetupWin2000 I had some problems installing Cocoon 2.1m2 on my linux server, so I made this how-to for anyone interested.

### Java setup

I'll leave most of this to you because i had probelms with self extracting binareis so can't remeber what i ended up doing. basically i ended up with java at /usr/local/java/j2sdk1.4.1_02/ which i simlinked to /usr/local/java/current

### Tomcat Setup

(1) Download Tomcat 4.1.24-LE-jdk14

(2) install in /usr/local/tomcat/current/ (i install versions in /usr/local/tomcat/<version_number> then symlink /usr/local/tomcat/current to this, it makes upgrading easier)

(3) Run file startup.sh located at /usr/local/tomcat/current/bin

(4) Enter http://localhost:8080 or http://your_server_name:8080 in your browser and check that Tomcat (the servlet engine) is running.

-> Tomcat installed, now you need Cocoon!

---

### Cocoon Setup

(5) Download cocoon source from http://cocoon.apache.org/mirror.cgi/cocoon-2.1m2-src.tar.gz

(6) Extract it to /usr/local/cocoon/cocoon-2.1m2/

(7) Set JAVA_HOME environmant variable to wherever Java SDK 1.4.1_02 is on you system in mycase:
export JAVA_HOME=/usr/local/java/current/

(8) Compile Cocoon using shell interface and type /usr/local/cocoon/cocoon-2.1m2/build.sh webapp

(9) move directory /usr/local/cocoon/cocoon-2.1m2/build/webapp to /usr/local/tomcat/current/webapps/cocoon (copy or move files from /usr/local/cocoon /cocoon-2.1m2/build/webapp-directory to /usr/local/tomcat/current/webapps/cocoon-directory)

(10) You need to copy 3 files: xalan-<version no>.jar, xercesImpl-<version no>.jar and xml-apis.jar from /usr/local/cocoon/cocoon-2.1m2/lib/endorsed into directory /usr/local/tomcat/current/common/endorsed

Note: Before copying the 3 files, the tomcat directory "endorsed" was actually empty.

Run Tomcat again (run file startup.sh)

Direct your browser on http://localhost:8080/cocoon and ... it should be working.

Also check samples at http://localhost:8080/cocoon/samples/

---

Fun starts when you try using a headless configuration setting "java.awt.headless=true" doesn't work complaining that "Exception in thread "main" java.lang.UnsatisfiedLinkError: /usr/lib/j2sdk1.4.1_02/jre/lib/i386/libfontmanager.so: libstdc++-libc6.1-1.so.2: cannot open shared object file: No such file or directory "
This is due to the fact that Redhat 8 uses libc5 and java is looking for libc6....since the difference isn't just a point release (5.1 to 5.32 say)but is a major version out then the solution is probably to use redhat 9 instead or stop the headless stuff and use the pja/eteks libraries.

Upgraded to redhat 9 and jdk 1.4.1_03 and svg -> jpeg conversion works when configured as headless.

---

Join me next week when i will write a thrilling installment entitled "Don't get more than 2 gig of memmory for a x86 (linux, windows) box as the jvm can only use 1960meg" ;-P

Actually you make use of more than 2 gig on a machine, you have to run several instances of tomcat on different ports and load balance across them with ajp or your connector of choice. obviosly each instance is limmited to less than 1960 meg of memmory each.

---

Small addition: With the tomcat4-4.1.24-full.2jpp RPM the above example didn't work until I set in /etc/tomcat4/tomcat4.conf the following variable:
JAVA_ENDORSED_DIRS=$CATALINA_HOME/common/endorsed

Otherwise old libraries from the Java SDK were used.

---

## Running Tomcat as a service on Linux: TomcatServiceOnLinux