

■ はじめに

本文書では、岐阜経済大学井戸ゼミのみなさんのLinux PCで、Javaサーブレット、JSP を使えるようにする方法を説明します。

■ 1. RPMファイルの入手

通常は、次のSunのWebサイトからダウンロードします。次の3つのファイルをダウンロードします。

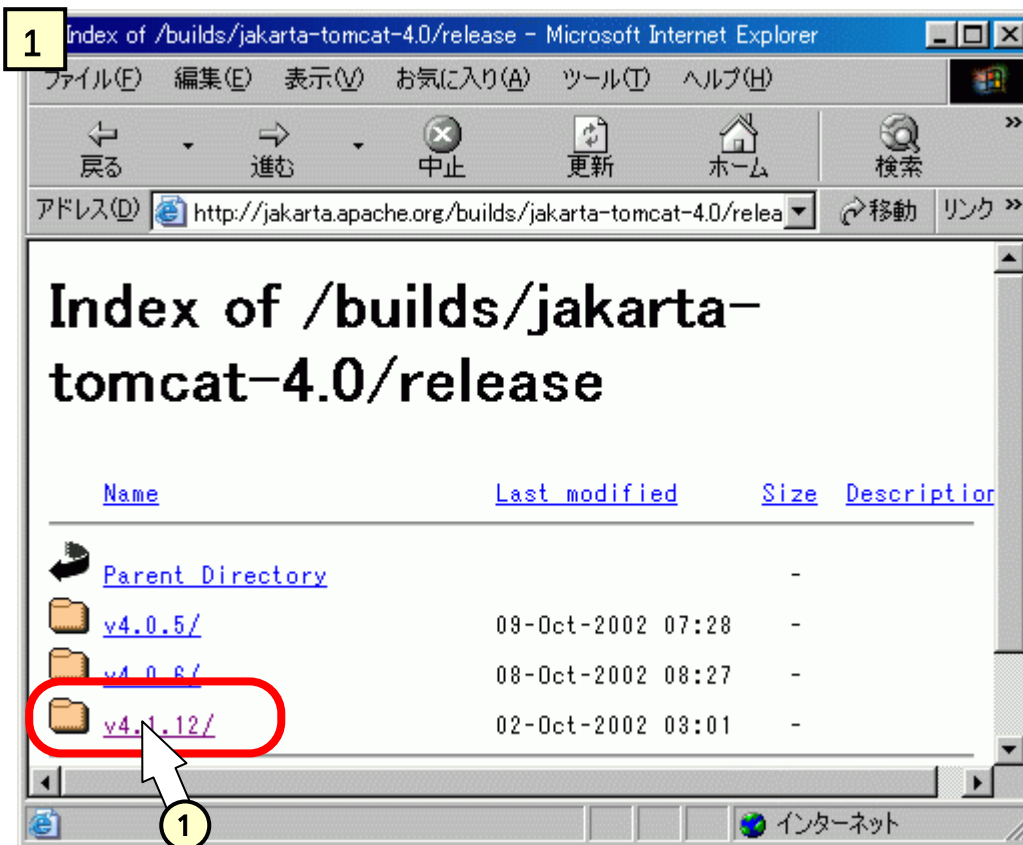
- (a) tomcat4-4.1.18-full.1jpp.nonarch.rpm
- (b) tomcat4-webapps-4.1.18-full.1jpp.nonarch.rpm

授業では、おなじファイルをゼミでのネットワークからftpで転送してくることにします。

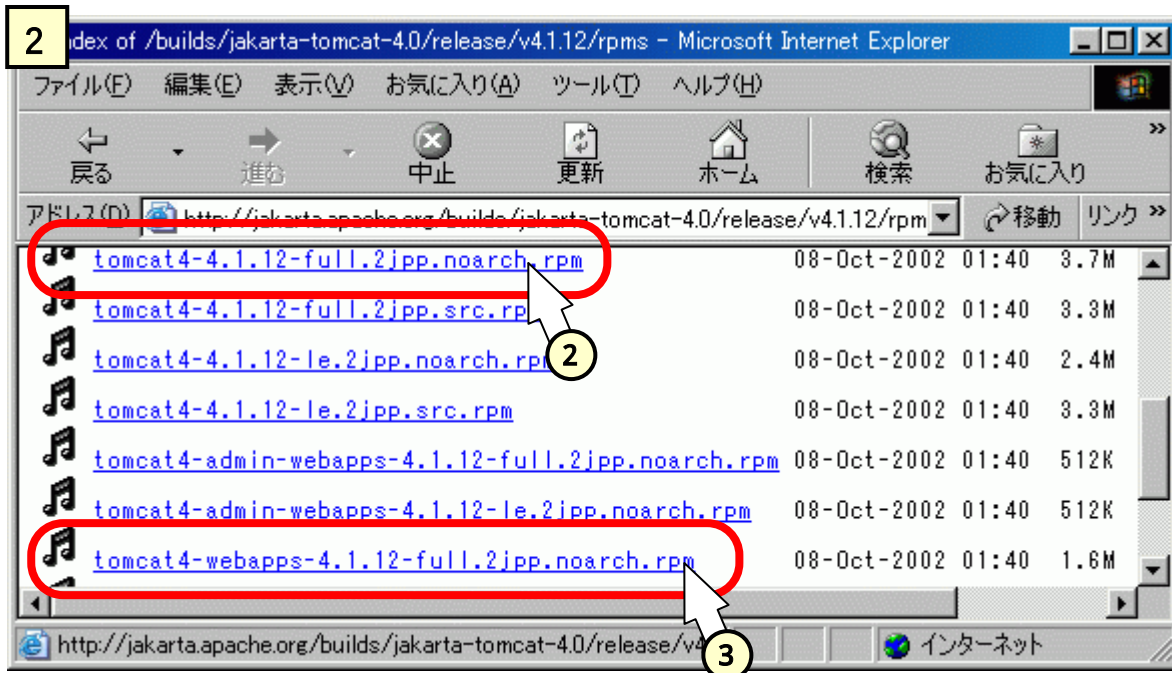
(1) ネットワークからのダウンロード

次のサイトにアクセスします。

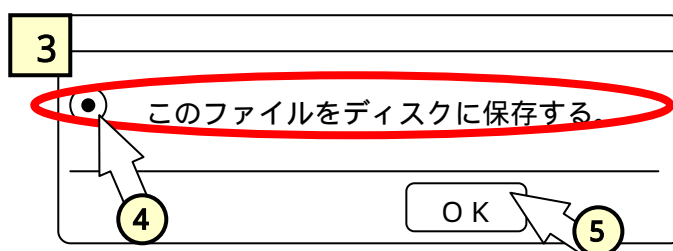
<http://jakarta.apache.org/builds/jakarta-tomcat-4.0/release/>



最新バージョンである“v4.1.12/”をクリック()します(バージョンアップによりさらに新しいバージョンがある場合は、そちらの方をクリックしてください)。似たような画面が出てきますので、今度は、“rpms/”をクリックしてください。すると、[2]の画面が現れます。



それぞれファイル名に相当する行をクリック（ ）してきます。



保存先を問うダイアログが出ますので、適当なディレクトリ（例えば~/prn/RPMS/i386/tomcat4）を作成・選択して、[保存する]をクリックしてください。

（２）井戸ゼミのネットワークからダウンロード

（ア）適当なディレクトリ（例えば~/prn/RPMS/i386/）を作成して、そこへ移動してください。

（イ）井戸のサーバ役PCにftpします。

ネットワーク構築実習時、/etc/hosts のファイルに次の行を追加したと思います。

```
192.168.0.11 left_idonet.net right_idon
```

これが追加されている場合、次のコマンドを投入します。

```
% ftp right_idon
```

追加されていない場合、次のコマンドを投入します。

```
% ftp 192.168.0.11
```

（ウ）Name と Password とを順次入力します。Name は semi と投入してください。Password は口頭で説明します。

（エ）サーバ側でのディレクトリを移動してください。

```
% cd /usr/local/rpms/tomcat
```

（オ）ファイルを get します。

```
% get tomcat4-4.1.18-full.1jpp.nonarch.rpm
```

```
% get tomcat4-webapps-4.1.18-full.1jpp.nonarch.rpm
```

■ 2 . インストール

(ア) まず、スーパーユーザになってください。

```
% su -
```

Password : <パスワードを入力してください>

(イ) ディレクトリを移動します。

```
% cd rpm/RPMS/i386
```

(ウ) ディレクトリを作って、そこへ移動しておきます。

```
% mkdir tomcat4
```

```
% cd tomcat4
```

(ウ) 先ほどダウンロードしたファイル(a)(b)をコピーします。

```
% cp /home/xxxx/rpm/RPMS/i386/tomcat4/
```

```
tomcat4-4.1.12-full.1jpp.nonarch.rpm .
```

```
% cp /home/xxxx/rpm/RPMS/i386/tomcat4/
```

```
tomcat4-webapps-4.1.12-full.1jpp.nonarch.rpm .
```

先ほどのダウンロードで、直接このディレクトリを保存先としても、もちろん結構です。その際には、mozilla および ftp の起動は、スーパーユーザになってから行います。

(エ) rpm コマンドを投入して、インストールを行います。

```
% rpm -ivh tomcat4-4.1.12-full.1jpp.nonarch.rpm
```

```
% rpm -ivh tomcat4-webapps-4.1.12-full.1jpp.nonarch.rpm
```

これで、/var/tomcat4/ というディレクトリにインストールされています。

■ 3 . 環境変数の設定

引き続きスーパーユーザのまま作業を行います。ここでやる作業は、Java へのパスを通すことです。

(ア) “/etc/tomcat4” のディレクトリに移動します。ここで、emacs を用いて、“tomcat.conf” というファイルに、次の行を追加します。

```
% cd /etc/tomcat4
```

```
% emacs tomcat.conf
```

(追加イメージ)

```
# you could also override JAVA_HOME here
```

```
# Where your java installation lives
```

```
# JAVA_HOME="/usr/java/jdk"
```

```
# JAVA_HOME="/opt/IBMJava2-131"
```

```
JAVA_HOME="/usr/java/j2sdk1.4.0_02"
```

追加

上記の “/usr/java/j2sdk1.4.0_02” の部分は、自身の PC で J2sdk をインストールしたディレクトリを入力します。“ls /usr/java” を投入して、確認してください。

(2) サーブレットプログラム、JSP プログラム用のディレクトリの設定

■ 4 . 動作確認

まず、「■ 6 . 停止と起動」に従って、停止・起動を行ってください。

次に、mozilla を立ち上げて、次の URL にアクセスしてみます。

```
http://localhost:8080/
```

次のようなページが見えれば、正常にインストールが完了していることを確認できます。

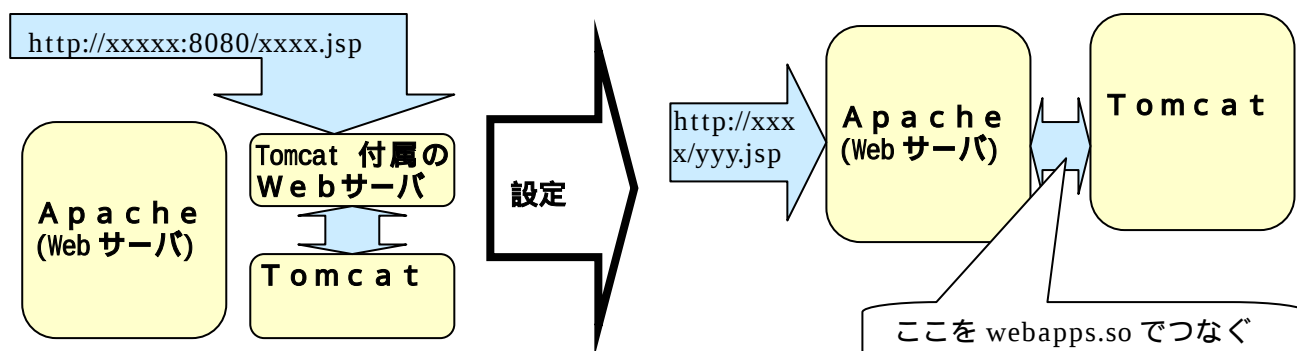


サンプルプログラムがありますので、動かしてみてください。

■ 5 . Tomcat と Apache との連携、作業ディレクトリの設定

この時点で、TOMCAT は正しくインストールされている訳ですが、設定変更を行います。

(目的 1) 上記の 8 0 8 0 ポートからのアクセスでは、Webサーバとして、Apache を使わず、Tomcat に付随したサーバを使っていることになります。この付随したサーバは、機能・能力的に不十分なため、ちゃんと Apache を使うように設定します。



(目的 2) サーブレット・JSP のプログラムは、予め決められたディレクトリでしか動かない状態になっています。各個人のホームディレクトリ配下で作業した方が便利ですので、その設定も行います。次のアカウントのホーム配下に作成することを前提とします。

`/home/ido/xxxxxxx/`

JSP が使えるディレクトリ “xxxxxxx” の名前は、任意です。

サーブレットを作成する際の、“tomcat” 配下のディレクトリ・ファイル構成については、別途説明します。

上記のディレクトリの配下、“/home/ido/xxxxxxx/test.jsp” の JSP ファイルを作ったとすると、次の URL で該 JSP ファイルにアクセスできるようになります。

`http://localhost/yyyyyy/test.jsp`

“yyyyyy” の名前も任意です。“xxxxxxx” と同じでも構いません。“xxxxxxx” は、各ユーザで同じ名前を用いても OK ですが、“yyyyyy” はシステムでユニークである必要があります (URL でユニークにサイトを指定出来なければいけないので、当然ですね)。

□手順

(1) mod_webapps.so のビルド

ここではビルドの方法は省略します。井戸作成の CDR からコピーしてください。

```
# cp /usr/local/rpms/tomcat/mod_webapps.so /etc/httpd/modules
# chmod a+x /etc/httpd/modules/mod_webapp.so
```

(2) apache 側の設定

Apache の設定ファイルである次のファイルを編集します。

“ /etc/httpd/conf/httpd.conf ”

編集前には、バックアップを取っておいてください。

```
# cd /etc/httpd/conf
# cp httpd.conf httpd.conf.original
```

(2 . 1) mod_webapps.so のロード

mod_webapps.so をロードするように、次の行を追加します。追加する場所は、“ LoadModule ” と書かれた一連の行の最後で OK です (“ # ” で始まる行はコメントです)

(編集前)

```
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
```

(編集後)

```
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
# 2003.05.04 ido
LoadModule webapp_module modules/mod_webapp.so          追加
# end of 2003.05.04 ido
```

(2 . 2) コネクタの追加

Web サーバ(Apache)が受け取ったリクエストを、Tomcat へ渡すことを記しておきます。次の行を追加します。追加する場所は、ファイルの末尾で OK です。

(修正後)

```
# 2003.05.04 ido
<IfModule mod_webapp.c>
WebAppConnection warpConn warp      localhost:8008
WebAppDeploy      yyyyyyy warpConn /yyyyyyy/          追加
WebAppInfo                          /webapp-info/
</IfModule>
# end of 2003.05.04 ido
```

点線で囲まれた “ yyyyyyy ” の文字列を含む行が、上記 (目的 2) のために追加した行です。つまり、(目的 2) のようなディレクトリ・URL を他に作りたのであれば、この行を追加すれば OK です。

(2 . 3) ServerName の設定

サーバネームを次のように設定してください。

(編集前)

```
#ServerName new.host.name:80
```

(編集後)

```
#ServerName new.host.name:80
```

```
# 2003.05.04 ido
```

```
ServerName localhost:80
```

```
# end of 2003.05.04 ido
```

追加

(3) Tomcat 側の設定

Apache の設定ファイルである次の 2 つのファイルを編集します。

```
" /etc/tomcat4/server.xml "
```

```
" /etc/tomcat4/web.xml "
```

編集前には、バックアップを取っておいてください。

```
# cd /etc/tomcat4
```

```
# cp server.xml server.xml.original
```

```
# cp web.xml web.xml.original
```

(3 . 1) コネクタの設定 (server.xml)

" server.xml " のファイルの末尾には、Tomcat と Apache とを連動させるための記述がありますが、これはコメントアウト ("`<!--` " と "`-->` " とで囲う) されています。

(編集前)

```
<!--
```

```
<Service name="Tomcat-Apache">
```

```
<Connector className="org.apache.catalina.connector.warp.WarpConnector"
```

```
port="8008" minProcessors="5" maxProcessors="75"
```

```
enableLookups="true" appBase="webapps"
```

```
acceptCount="10" debug="0"/>
```

```
<Engine className="org.apache.catalina.connector.warp.WarpEngine"
```

```
name="Apache" debug="0">
```

```
<Logger className="org.apache.catalina.logger.FileLogger"
```

```
prefix="apache_log." suffix=".txt"
```

```
timestamp="true"/>
```

```
<Realm className="org.apache.catalina.realm.MemoryRealm" />
```

```
</Engine>
```

```
</Service>
```

```
-->
```

コメントをはずして、上記の部分を使えるようにし、かつ、コネクタに関する記述を追加します。

次ページの修正後のイメージ中、点線で囲まれた " yyyyyyy " と " xxxxxxxx " の文字列を含む行が、上

記（目的２）のために追加した行です。つまり、（目的２）のようなディレクトリ・URL を他に作りた
いのであれば、この行を追加すれば OK です。

（修正後）

```
<!-- 2003.05.04 ido : uncomment -->                                コメントはずし
<Service name="Tomcat-Apache">
  <Connector className="org.apache.catalina.connector.warp.WarpConnector"
    port="8008" minProcessors="5" maxProcessors="75"
    enableLookups="true" appBase="webapps"
    acceptCount="10" debug="0"/>
  <Realm className="org.apache.catalina.realm.MemoryRealm" />
  <!-- 2003.05.04 ido -->
  <Host name="localhost" debug="0" appBase="webapps"                追加
    unpackWARs="true" autoDeploy="true">
    <Context path="/yyyyyyy" docBase="/home/ido/xxxxxxxxt" debug="0"
      reloadable="true" crossContext="true">
    </Context>
  </Host>
<!-- end of 2003.05.04 ido -->
</Engine>
</Service>                                コメントはずし
```

（３．２）URL とサーブレットとの関連付け（web.xml）

サーブレットを起動するときに、“http://localhost/servlet/...” の形で起動できるように設
定します（JSP を動かす範囲では関係ありません）。

次のように、コメントはずしを行います。

（修正前）

```
<!--    <servlet-mapping>
        <servlet-name>invoker</servlet-name>
        <url-pattern>/servlet/*</url-pattern>
    </servlet-mapping>
-->
```

（修正後）

```
<!-- 2003.05.05 ido : uncomment -->                                コメントはずし
<servlet-mapping>
  <servlet-name>invoker</servlet-name>
  <url-pattern>/servlet/*</url-pattern>
</servlet-mapping>
<!-- 2003.05.05 ido : uncomment -->                                コメントはずし
```

（４）自身のディレクトリのパーミッションの変更

やろうとしていたことは、Apache+Tomcat があなたのアカウントのディレクトリを見にきて、JSP ファイルやサーブレットを読み出し、起動することが出来るようにすることでした。つまり、あなた以外のユーザがあなたのディレクトリとその配下のファイルを読み出す訳です。あなたのホームディレクトリのパーミッションが、別のユーザからのアクセスを拒否する設定になっていると、このことが出来なくなります。Redhat では、ホームディレクトリのパーミッションのデフォルト値が“ 700 ”(すなわち、自分以外はディレクトリに入ることも出来ない)になっています。これを変更しておきます。

```
# cd
# pwd
/home/ido
# chmod a+xr .
```

■ 6 . 停止と起動

(1) 起動・再起動

Tomcat の停止と起動とは、Apache の停止と起動とを伴って行います。すなわち、最初に起動する際は、次の順で行います。

[Tomcat の起動] [Apache(httpd)の起動]

すでに Tomcat4・Apache が起動されている状態での再起動は、次の順で行います。

[Apache(httpd)の起動] [Tomcat の再起動] [Apache(httpd)の起動]

上記の再起動をコマンドで行う際は、スーパーユーザにログインした後、次のように行います。

```
# service httpd stop
# service tomcat4 restart
# service httpd start
```

(2) 自動起動

システムが立ち上がる際に、Tomcat を自動起動するようにしておくには、スーパーユーザにログインした後、次のように行います (Apache は立ち上がるようになっているはずです)。

```
# chkconfig --add tomcat4
```

システムが立ち上がる際に Tomcat が立ち上がるか否かは、次のコマンドで調べます。

```
# chkconfig --list tomcat4
tomcat4    0:オフ 1:オフ 2:オフ 3:オン 4:オン 5:オン 6:オフ
```

上記の数字 (0 ~ 6) は、ランレベルといわれるものです。詳しくは Linux の教科書を参照してください。ここでは簡単に、「 3 , 4 , 5 のランレベルがオンであれば、システムが立ち上がる時に Tomcat も立ち上がる」と理解しておいてください。なお、“ service ”、“ chkconfig ”のコマンドは、もちろん Tomcat 以外のサービスについても使うことが出来ます。

(3) 再起動が必要な場合

設定ファイルを変更したり、JSP が動くディレクトリを作成した場合などは、その後に Tomcat・Apache を再起動する必要があります。

■ おわりに

TOMCAT4 のインストールについて説明しました。

—以上—