・tomcat-5.5-logging
・Tomcat SingleSignOn 検証
・Tomcat クライアント IP 接続制御
・Tomcat manager jmxproxy

メモ

tomcat-5.5.23 を RHEL-4 に zip インストールする

■設定内容

・RHEL-4
・zip を展開して tomcat をインストール
・HTTP 用に 38080
・AJP に 38009
・jvmRoute に tomcat5c
・インストール先 /usr/local/tomcat5c
・ログの出力先 /var/log/tomcat5c

■ファイルの取得

・apache-tomcat-5.5.23.zip

■ファイルの展開

```
# cd /usr/local
# tar tzvf apache-tomcat-5.5.23.zip ( 確認 )
# tar xzvf apache-tomcat-5.5.23.zip ( 展開実行 )
```

■ $TOMCAT_HOME/conf/server.xml 編集

```
<Service name="Catalina">
  <Connector port="38080"  useBodyEncodingForURI="true" />
  <Connector port="38009" protocol="AJP/1.3"  useBodyEncodingForURI="true" />
  <Engine name="Catalina" defaultHost="localhost" jvmRoute="tomcat5c">
```

■ログ出力先調整

```
# rmdir /usr/local/tomcat5c/logs
# mkdir /var/log/tomcat5c (tomcat の実行ユーザーが書き込めるように適宜権限を設定 )
# ln -s /var/log/tomcat5c /usr/local/tomcat5c/logs
```

・ファイルシステム保護のため
  ・インストールデータは /usr
  ・ログは /var/log

■ $TOMCAT_HOME/bin/catalina.sh 編集

・chkconfig 設定
・起動パラメータ設定
・ログファイル文字コード変換
・実行権限追加

```
# chmod +x /usr/local/tomcat5c/bin/catalina.sh
```

■起動ファイルの作成

```
# ln -s /usr/local/tomcat5c/bin/catalina.sh /etc/init.d/tomcat5c
# chkconfig --add tomcat5c
# chkconfig --list tomcat5c
tomcat5c        0:off   1:off   2:off   3:on    4:on    5:on    6:off
```

■起動時のエラー setclasspath.sh 実行権限が足りない

service tomcat5c start
The BASEDIR environment variable is not defined correctly
This environment variable is needed to run this program

```
# chmod +x setclasspath.sh
```

```
<Valve className="org.apache.catalina.valves.FastCommonAccessLogValve"
               directory="logs"  prefix="localhost_access_log." suffix=".txt"
               pattern="common" resolveHosts="false"/>
```

RHEL-4 の tomcat5 の別インスタンスを作成する

■設定

・RHEL4-apptomcat5 の

... 作成中

Tomcat での static な URL から動的パラメータ抽出

・tomcat5 で検索サイト適合した (SEO 的な) URL を提供するための設定は？
・mod_rewrite を使わずに。
・http://server_name/tomcat_context/display.jsp?param1=aaa&param2=bbb ページを用意した場合に
・http://server_name/tomcat_context/display/aaa/bbb.html で表示できるように

■ web.xml の設定

```
<servlet>
  <servlet-name>display.jsp</servlet-name>
  <jsp-file>/display.jsp</jsp-file>
</servlet>
<servlet-mapping>
  <servlet-name>display.jsp</servlet-name>
  <url-pattern>/display/*</url-pattern>
</servlet-mapping>
```

・servlet 内で servlet-class だけでなく jsp-file が使える

■ jsp 内でのパラメータ抽出

```
<%@ import="java.util.regex.Pattern" import="java.util.regex.Matcher"%>
```

```
Pattern p = Pattern.compile("/tomcat_context/display/([^/]+)/([^/]+).html");
```

```
Matcher m = p.matcher(request.getRequestURI());
if (m.matches()) {
  p1=m.group(1);
  pw=m.group(2);
} else {
  p1="";
  pw="";
}
```

参考サイト

■ Tomcat-5.5 でのログ出力が変更された

・http://mail-archives.apache.org/mod_mbox/tomcat-users/200508.mbox/%3C5684A7E6FB10504393A2806C1F4C021003E0F0C8@orion.qas.com%3E
・5.5 で FileLogger がなくなった影響により混乱が生じている。
・tomcat-5.5 のドキュメント Logging に説明があるので読め
・http://tomcat.apache.org/tomcat-5.5-doc/logging.html

■電脳チラシの裏 / Java / Tomcat 5.5 で session persistence (using JDBCStore)

・http://d.hatena.ne.jp/stdcall/20070102

"tomcat5.5 において、セッション情報を RDBMS(PostgreSQL) に格納するということがしたくて休みの間ゴソゴソ試してました。負荷分散したいけど stickysession が使えないという状況下、クラスタ機能を使わずとも、セッションの格納場所を DB に一元化することでどこまで対応できるのかなぁというアタリを付けてみたかったからです。とりあえず下記のような設定で DB への格納自体はうまくいったのでメモ代わりに記しておきます。

1. セッション情報格納に使うユーザ情報やテーブルを作る
2.Context 設定を記述する xml(server.xml とか) に PersistentManager と JDBCStore の設定を追加
    ・あと、${tomcat_home}/common/lib に JDBC ドライバの jar 放り込むのを忘れないでください。WEB-INF/lib では駄目です。
    ・要確認
3. 確認
    ・ポート番号だけ変えて tomcat を二つ上げてみる。（8080 番と 18080 番など）
    ・こんな jsp を置く (セッションのデータを操作する / カウントアップ)
    ・8080 番の URL にアクセス
    ・psql コマンドで DB の中を確認
    ・数秒置いて 18080 番の URL にアクセス
    ・数秒置いて また 8080 番の URL にアクセス
・4. ハマッた点、注意点

" あーでもないこーでもないと色々やっていたら、Context タグに backgroundProcessorDelay なる不審な属性を発見。ソースを見ると、確かにこの値が 1 以上だと Thread 起こして Manager インターフェースの差分反映メソッドを周期的に叩くようになってます。

" 果たして、backgroundProcessorDelay="1" としてみたら……HttpSession#setAttribute が呼ばれた数秒後、無事 DB の値が更新されたのでした！イヤッホウ！

■ The Apache Jakarta Tomcat 5.5 Servlet/JSP Container Class Loader HOW-TO (US)

・http://jakarta.apache.org/tomcat/tomcat-5.5-doc/class-loader-howto.html
・Tomcat のクラスローダー・ツリー

■第 2 回 「クラスローダーを理解する - シングルトンがシングルトンでなくなる日」 / クラスローダーと J2EE パッケージング戦略を理解する

・http://www-06.ibm.com/jp/software/websphere/developer/j2ee/strategy/2.html

WebSphere Application Server が内部でどんなライブラリーを使用していようとも、それに影響されることなく自分の好きなライブラリーの最新バージョンを使用したいという欲求は開発者なら誰もが思うところです。「PARENT LAST」にするのは自然なことです。

以下、サーブレット 2.4 Specification SRV.9.7.2 からの引用です。
It is recommended also that the application class loader be implemented so that classes and resources packaged within the WAR are loaded in preference to classes and resources residing in container-wide library JARs.
アプリケーションのクラスをロードするクラスローダーは、アプリケーション・サーバー、Web コンテナ全体で使用するライブラリー内よりも、WAR 内部のクラスやリソースを優先してロードすることが推奨される。

Tomcat では「common」「share」配下よりも各「webapp」内のクラス・リソースが優先されると、ドキュメントには明記されています。Web アプリケーションのクラスローダーに関しては、やはり「PARENT LAST」の設定になっているわけです。

■ Tomcat の 404 エラーページでバージョン情報を消去するには

web.xml でのエラーページのカスタマイズ

■ Tomcat の HTTP Server ヘッダでのバージョン情報の消去

server.xml の connector 要素の server 属性に返したい文字列を指定する。

■ SSL アクセラレータ利用時のリダイレクト

・http://guns.at.webry.info/200804/article_1.html

...SSL アクセラレータを利用した場合は、...
... リダイレクトの絶対 URL が構築された際に「http」として構築されてしまいます。

この問題を回避するには server.xml の connector タグの属性を変更すれば回避できます。
proxyName=" サーバ名 "
proxyPort=" ポート番号 " ※通常 SSL なら 443
scheme=" スキーマ "　※通常 SSL なら https
secure="true" これは設定しなくてもよいけど、request.isSecure() の戻り値が true になる。

■ Tomcat で Cookie を Secure に設定する方法

・http://gmt-24.net/archives/320

Apache や Tomcat で SSL してる場合はよいのですが、SSL アクセラレータやロードバランサ、stunnel などで SSL を解除していると secure と認識されなくなってしまい、Secure 属性が付与されなくなってしまいます。
Tomcat の AJP13 コネクタ定義には secure という属性があり、こちらを true に設定すると response.isSecure() に true を返してくれるようになります。

・server.xml の connector 要素の secure 属性を "true" に設定する

■ URL Mapping for JSP in Tomcat

・http://nscraps.com/JSP/496-url-mapping-jsp-tomcat.htm