

Kangaroo-egg webserver

Version:0.2.9/0.2.2 (beta)

User Manual

Shemin Dunne

You must have basis of JAVA and HTML before reading the manual.

Directory

| | | |
|-----------|--------------------------------------|----|
| Chapter 1 | Introduce kangaroo-egg..... | 3 |
| 1.1 | Kangaroo-egg and DQM container | 4 |
| 1.2 | Intsall kangaroo-egg..... | 4 |
| Chapter 2 | Config kangaroo-egg | 6 |
| 2.1 | systemSet element..... | 6 |
| 2.2 | connectors element..... | 9 |
| 2.3 | mainHost element | 12 |
| 2.4 | userApps element | 20 |
| 2.5 | vHost element | 23 |
| 2.6 | webfile folder | 27 |
| Chapter 3 | DQM technology | 27 |
| 3.1 | Introduce DQM | 28 |
| 3.2 | DQM directives..... | 28 |
| 3.3 | Fragments of java program | 30 |
| 3.4 | Dqm file released | 30 |
| Chapter 4 | Built-in object : out..... | 32 |
| 4.1 | Methof of print and println..... | 32 |
| 4.2 | Method of write..... | 34 |
| 4.3 | Method of compress web content | 35 |
| 4.4 | Method of buffer | 37 |
| 4.5 | Directly output data..... | 38 |
| Chapter 5 | Built-in object : resopnse..... | 39 |
| 5.1 | Redirect | 39 |
| 5.2 | Set HTTP head's attribute | 41 |
| 5.3 | Setup ContentType | 42 |
| 5.4 | Method of encodeURL..... | 43 |
| 5.5 | Using cookie | 44 |
| 5.6 | Insert static document | 44 |
| 5.7 | Option output | 45 |
| Chapter 6 | Built-in object : request | 52 |
| 6.1 | Get submit data | 52 |
| 6.2 | Temporary data | 56 |
| 6.3 | Read client information..... | 56 |
| 6.4 | Read server information..... | 58 |
| 6.5 | Read cookie..... | 59 |
| 6.6 | Read dqm file path information | 59 |
| 6.7 | GET multipart data..... | 64 |
| Chapter 7 | Use cookie | 74 |
| 7.1 | DCookie object | 74 |

| | | |
|------------|--|-----|
| 7.2 | Example of use DCookie | 76 |
| Chapter 8 | Built-in object : session | 78 |
| 8.1 | About session object | 78 |
| 8.2 | Write and read data into session..... | 78 |
| 8.3 | Session object's information..... | 80 |
| 8.4 | How to release session | 82 |
| Chapter 9 | Built-in object : application | 83 |
| 9.1 | Write and read data into application..... | 84 |
| 9.2 | How to use application..... | 84 |
| Chapter 10 | Built-in object : command | 88 |
| 10.1 | Introduce command object..... | 88 |
| 10.2 | Use command to process session and application..... | 88 |
| 10.3 | Use command to process Dqm Instance Buffer Pool..... | 90 |
| 10.4 | Use command to process Class Buffer Pool | 94 |
| 10.5 | Soft restart the server | 97 |
| 10.6 | Log operation..... | 97 |
| 10.7 | Password protection | 98 |
| Chapter 11 | Use JavaBean..... | 99 |
| 11.1 | Introduction JavaBean..... | 99 |
| 11.2 | DQM JavaBean tags..... | 100 |
| 11.3 | JavaBean object Scope | 101 |
| Chapter 12 | Create static web page..... | 104 |
| 12.1 | The ID of static web page | 104 |
| 12.2 | Regist static web page..... | 109 |
| 12.3 | The flow of static web page | 116 |
| 12.4 | Matters needing attention..... | 116 |
| 附录 1 | DQM 容器介绍 | 117 |
| F1.1 | 编译动态文件 | 118 |
| F1.2 | DQM 容器流程 | 119 |
| F1.3 | 关闭自动编译时 DQM 容器流程 | 120 |
| 附录 2 | 国际化问题..... | 123 |
| F2.1 | 编译和执行时的字符集 | 123 |
| F2.2 | 读取客户端请求时的字符集 | 124 |
| F2.3 | 读取数据时的字符集 | 128 |
| F2.4 | 设置 http 头时的字符集 | 131 |
| 附录 3 | 重新编译和隐藏源代码..... | 133 |
| F3.1 | 重新编译动态文件 | 133 |
| F3.2 | 隐藏源代码的原理 | 134 |
| F3.3 | 隐藏源代码的工具 | 135 |
| F3.4 | tools.jar 文件 | 135 |
| 附录 4 | 类的载入..... | 137 |
| F4.1 | 公用类的载入 | 137 |
| F4.2 | 服务器类的载入 | 138 |
| 附录 5 | 内部变量..... | 141 |

| | | |
|------|-----------------------------|-----|
| F5.1 | \$ke\$sourceLineNumber..... | 141 |
| 附录 6 | 生成证书..... | 146 |
| F6.1 | 创建证书的 2 种方法 | 146 |
| F6.2 | 使用 JDK 工具创建证书 | 147 |

Chapter 1 Introduce kangaroo-egg

Kangaroo-egg is a new webserver, it use Java language developed. It followed HTTP1.1 protocol and has DQM script language and container (very like Servlet/JSP). So it can use in medium or small web application. We believe it will become a popular webserver soon.

This chapter will introduce kangaroo-egg basic elements and how to install and startup it.

1.1 Kangaroo-egg and DQM container

Kangaroo-egg like JSP server, but it has own container to execute dynamic web page. This container name is DQM and dqm script language. DQM is a groupware must run at Java environment, and this groupware is a part of kangaroo-egg web server. The function of DQM container is execute dynamic web page that coding by dqm script language. If you known asp, jsp and php then dqm like them, and dqm script language is very like jsp script language, if you known jsp use dqm is very easy.

Ok, what special functions we have? First dqm script language is very easy. Secondly DQM container run dynamic web page after compile it. Lastly DQM container support define web page's extended yourself. Table 1-1-1 inform the difference of popular web script language.

| Script language | Container | Dhtml ext name | Execute process |
|-----------------|---------------|---------------------------------|-----------------|
| ASP | Microsoft IIS | .asp | Translate |
| JSP | Tomcat | .jsp | Compile |
| PHP | Apache | .php | Translate |
| DQM | kangaroo-egg | Any extended name that you want | Compile |

Table 1-1-1

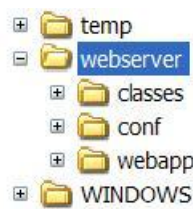
1.2 Intsall kangaroo-egg

Kangaroo-egg web use Java tech, so can deploy at Windows, linux and unix OS, but we only test our server at Windows, redhat linux and Solaris10, so maybe will has some compatible problem at other OS.

You must install JDK1.5 before install kangaroo-egg. Please refer to other correlative document to get how to install JDK1.5.

Notice: Kangaroo-egg must run at JDK1.5 or later.

You can download kangaroo-egg server from www.kangaroo-egg.com, it's a ZIP file, uncompress this file you can see the folders like picture 1-2-1.

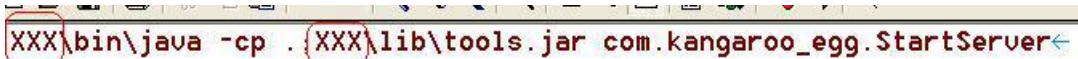


Picture 1-2-1

There are three folders, main program files at the “classes” folder, configuration files at the “conf” folder, and default web application files at the “webapp” folder.

Enter the “classes” folder you can see the two files run.bat and run.sh. The run.bat file is a startup script command under windows OS. The run.sh file is a startup script command under linux or unix OS. But these two files can’t execute now, because need config its first.

1. Open the run.bat, you will see:



XXX\bin\java -cp .\XXX\lib\tools.jar com.kangaroo_egg.StartServer

Picture 1-2-2

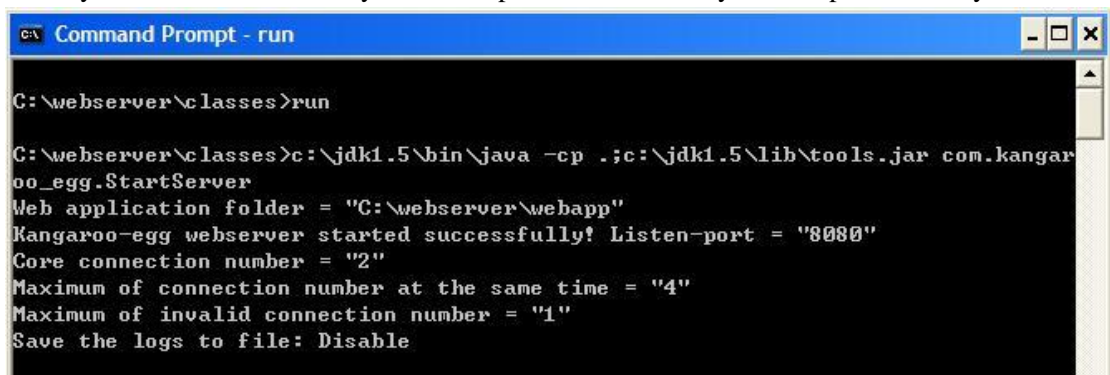
Please notice “XXX” at red mark, change “XXX” to your JDK installed folder. For instance your JDK installed to “c:\jdk1.5” folder then you must change the “XXX” to:



c:\jdk1.5\bin\java -cp .;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.StartServer

Picture 1-2-3

Ok, now you can execute run.bat, you can see picture 1-2-4 when you startup successfully.



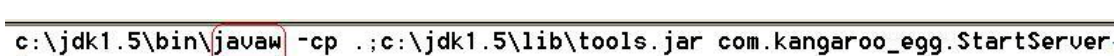
```
C:\> Command Prompt - run

C:\webserver\classes>run

C:\webserver\classes>c:\jdk1.5\bin\java -cp .;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.StartServer
Web application folder = "C:\webserver\webapp"
Kangaroo-egg webserver started successfully! Listen-port = "8080"
Core connection number = "2"
Maximum of connection number at the same time = "4"
Maximum of invalid connection number = "1"
Save the logs to file: Disable
```

Picture 1-2-4

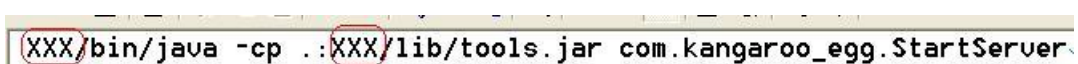
If you want to kangaroo-egg run at background, change “java” to “javaw”, like picture 1-2-5.



c:\jdk1.5\bin\javaw -cp .;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.StartServer

Picture 1-2-5

2. Open the run.sh, you will see:



XXX/bin/java -cp .:XXX/lib/tools.jar com.kangaroo_egg.StartServer

Picture 1-2-6

Please notice “XXX” at red mark too, change “XXX” to your JDK installed folder. For instance your JDK installed to “/jdk1.5” folder then you must change the “XXX” to:



/jdk1.5/bin/java -cp .:/jdk1.5/lib/tools.jar com.kangaroo_egg.StartServer

Picture 1-2-7

Ok, now you can execute run.sh, you can see same picture 1-2-4 when you startup successfully.

Notice: run.sh file's attribute must be execute, if not it can't be execute.

If you want to kangaroo-egg run at background under linux or unix OS, please use this command `./run.sh &`.

Notice: linux or unix OS has `nohup` command, this command can log console's content to nohup.out file.

Command: `nohup ./run.sh &`

Chapter 2 Config kangaroo-egg

Kangaroo-egg server has a `conf` folder under it installed folder.

Under `conf` folder there has a `webconfig.xml` file and a `webfile` folder, first we look `webconfig.xml`, it's a standard XML file, you can set kangaroo-egg use this file, but must reboot kangaroo-egg after change configuration.

`webconfig.xml` top element is `<kangaroo-egg>`, it's include these elements:

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE kangaroo-egg (View Source for full doctype...)>
- <kangaroo-egg>
+ <systemSet>
+ <connectors>
+ <mainHost>
+ <vHost number="1">
+ <vHost number="2">
</kangaroo-egg>
```

Picture 2-0-1

2.1 systemSet element

You can modify `systemSet` element to change kangaroo-egg server setting.

```

<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE kangaroo-egg (View Source for full doctype...)>
- <kangaroo-egg>
  - <systemSet>
    <systemPsw>123456</systemPsw>
    <regionSet>us</regionSet>
    <urlEnc>default</urlEnc>
    <dataEnc>default</dataEnc>
    <bufferSize>16</bufferSize>
    <threadPriority>5</threadPriority>
    <clearTimeoutSession>60</clearTimeoutSession>
    <clearDhtmlInstance enable="true" clearTime="7" />
    <saveLogs enable="true" bufSize="256" logFileMaxSize="204800" />
  </systemSet>
  + <connectors>
  + <mainHost>
  + <vHost number="1">
  + <vHost number="2">
</kangaroo-egg>

```

Picture 2-1-1

1. systemPsw

You can set server password use this element. Some operate need use server password, for example the operate at 10.6 need use server password. He password default set to 123456, so change it at first please.

2. regionSet

You can set server region use this element. Server information, language, data encoding will different by your set. Now support “cn” and “us” region.

3. urlEnc

Http request URL can attach datum, but no ascii data will encode, so server need decode those datum.

For example Chinese “小明” encode with UTF-8 character set will change to “%E5%B0%8F%E6%98%8E”. But it encode with GBK character set will change to “%D0%A1%C3%F7”.

So not encode URL is <http://127.0.0.1/run.dqm?name=小明>

Encode with UTF-8 URL is <http://127.0.0.1/run.dqm?name=%E5%B0%8F%E6%98%8E>

Encode with GBK URL is <http://127.0.0.1/run.dqm?name=%D0%A1%C3%F7>

So server need decode datum by different character set.

“%D0%A1%C3%F7” can decode to “小明” by use GBK character set. It can’t decode to “小明” by use UTF character set.

So you can set decode character use this element. You can use “default” value to set it, that means if you “regionSet” set to “cn” then this element will set to “GBK” automatically, and if you “regionSet” set to “us” then this element will set to “US-ASCII” automatically. Of course

you can set this element use explicit value also, example GBK, BIG5...

<urlEnc>BIG5</urlEnc>

Notice: Because http protocol recommend use UTF-8 character to decode URL, so kangaroo-egg will use UTF-8 to decode URL first, if decode unsuccessfully then use this element's value to decode URL again. So don't set this element to "UTF-8".

4. dataEnc

Chinese browser will encode post datum use GBK character default, Japanese browser will encode post datum use Shift_JIS character default. So you can set decode character use this element. You can use "default" value to set it, that means if you "regionSet" set to "cn" then this element will set to "GBK" automatically, and if you "regionSet" set to "us" then this element will set to "US-ASCII" automatically. Of course you can set this element use explicit value also, example GBK, BIG5...

<dataEnc>BIG5</dataEnc>

5. bufferSize

You can set server buffer size use this element. For example if this element's value set to 16, that means server buffer size is 16K. We set this element's value to 16 default.

6. threadPriority

You can set server thread priority use this element. Highest level is 10, lowest level is 1. We set this element's value to 5. Under some OS this is a useless element maybe.

7. clearTimeoutSession

Session will be expired, so need clear expired sessions to release memory periodically. (About session refer to Chapter 8 please.) You can set how long to clear time out sessions. We set this element's value to 60 default, that means server will clear time out sessions every 60 minutes.

If this value set too big then many expired sessions can't be cleared in time, so memory maybe will overflow.

If this value set too small then system will continually clear expired sessions, so server will become busy.

8. clearDhtmlInstance

DQM container like servlet container, first server translate from dynamic web page into java file, and compile java file into class file. If need execute class file, server will initialize class file to instance first, and save instance into DQM container. If another want to visit this page then server can get instance from DQM container direct, and execute instance.

The benefit of to do this is not initialize class when user visit it every time. This flow can improve server performance. But there are defects in this flow. If many instance save into DQM container then memory will overflow maybe. Other defect is if dynamic web page was deleted, but DQM container can't delete already exist instance automatically. So you can use this element clear all instances of DQM container periodically.

If "enable" attribute of this element set be "true", then this function enabled, else this function

disable. You can define how many days to clear instances use “clearTime” attribute of this element. Default it set to “7”, that means server will clear all instances every 7 days. The “clearTime” attribute can be avail when “enable” set to “true” only.

9. saveLogs

You can set visit logs save to file use this element. The element have 3 attributes:

1. The “enable” attribute’s value equal “true” then visit logs will save to file, else not save.
2. Server will save logs to buffer first, when buffer full then save all logs that in buffer to file. The “bufSize” attributes can set buffer size, if you set it to “256” that means buffer size is 256K.
3. You can set logs file maximum size use “logFileMaxSize” attributes. If you set it to “204800” that means logs file maximum size is 204800K (200M).

The “logFileMaxSize” attribute can be avail when “enable” attribute set to “true” only. If disable “saveLogs” function then visit logs will not save to buffer too. Please refer to section 10.6 to know how to view buffer logs.

Notice: If “bufSize” attribute’s value larger than “logFileMaxSize” attribute’s value then log file maximum size equal bufferSize attribute’s value.

Kangaroo-egg server use CLF format to record visit log. A entry of log file represent a HTTP request. Some request informations contained in every entry, use “|” flag to split it, it represent:

1. The thread of process current HTTP request.
2. Start time of process current HTTP request.
3. The remote IP address of current HTTP request.
4. The request method, request URI and request protocol of current HTTP request.
5. Server response code of current HTTP request, for example 200, 503.
6. Server response content length, if it equals -1 then mean content length unknown., for example if transmitted content use chunked code then can’t know content length.
7. End time of process current HTTP request.
8. If current web enable password protect then visit some page need enter username and password, if username ok then server will record it at current flag. About password protect please refer to section 2.3.

2.2 connectors element

You can modify “connectors” element to change kangaroo-egg server connection pool.

```

<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE kangaroo-egg (View Source for full doctype...)>
- <kangaroo-egg>
+ <systemSet>
- <connectors>
    <coreConnectionNumber>40</coreConnectionNumber>
    <maxConnectionNumber timeOut="20">200</maxConnectionNumber>
    <maxServerUnavailableNumber>5</maxServerUnavailableNumber>
    <maxPersistentConnectionsNumber>20</maxPersistentConnectionsNumber>
    <connectionTimeout>50</connectionTimeout>
    <HTTP enable="true" port="80" />
    <HTTPS enable="true" port="443" keystoreFile="c:\key.store"
        keystoreType="jks" keystorePass="123456" keyPass="123456"
        needClientAuth="false" />
    </connectors>
+ <mainHost>
+ <vHost number="1">
+ <vHost number="2">
</kangaroo-egg>

```

Picture 2-2-1

1. coreConnectionNumber

You can set server core connection pool size use this element. Connection will consume system memory. Core connection can't died never.

2. maxConnectionNumber

You can set server maximum connection pool size use this element. This value must larger than or equal coreConnectionNumber's value. The number of coreConnectionNumber larger than maxConnectionNumber are non-coreConnectionNumber connection pool size. See picture 2-2-1 that coreConnectionNumber's value is "40" and maxConnectionNumber's value is "200", so non-coreConnectionNumber's value is 160. But what is non-core ConnectionNumber connection pool? Non-core connection will died automatically when it's no activate and exceeded time limit. This element's "timeOut" attribute can set non-core connection time out value. For example in picture 2-2-1 that "timeOut" attribute is "20", that means non-core connection will died automatically when it's no activate and exceeded 20 minutes.

Server will use core connection first, if all of core connection used up then will use non-core connection. If non-core connection used up too then response 503 error.

If coreConnectionNumber's value equals maxconnectionNumber's value, then non-core connection pool can't be used. In this status all of the request will use core connection, if core connection used up then response 503 error.

No-core connection can be of use to the time of many requests, it's can offer more connectons for this status. Sever will automatically release time out non-core conntion when accept less requests.

Please notice if you set many connections then need many memory. You must assign more memory to kangaroo-egg server. For example I set "maxConnectionNumber" to 800, so I want to assign 512M memory to kangaroo-egg server, What can I do? Very easy modify run.bat :

```

c:\jdk1.5\bin\java -Xmx512m -cp .;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.StartServer

```

Picture 2-2-2

run.sh:

```
/jdk1.5/bin/java -Xmx512m -cp ./jdk1.5/lib/tools.jar com.kangaroo_egg.StartServer
```

Picture 2-2-3

3. maxServerUnavailableNumber

You can set server unavailable connection pool size use this element. If all of the connections used up then server will use unavailable connection response 503. If unavailable connection used up then closed client's request. Suggest set this element to a small value.

4. maxPersistentConnectionsNumber

You can set persistent connections maximum number use this element. For example if this value is ten that means client can connect ten number in a same connection. If set this element to "1" that means close function of persistent connections. About persistent connections please refer to rfc2616 document.

5. connectionTimeout

You can set time out of persistent connections use this element. Sets the length of time in seconds before the server disconnects an inactive user. This ensures that all connections are closed if the HTTP protocol fails to close a connection. For example in picture 2-2-1 that "connectionTimeout" element's value is "50", that means alive persistent connection will closed automatically when it's no activate and exceeded 50 seconds.

6. HTTP

You can enable HTTP service use this element. The element have 3 attributes:

1. The "enable" attribute's value equal "true" then enable HTTP service, else is disable HTTP service.
2. You can set HTTP listened port use "port" attribute. We default set it to "8080".

7. HTTPS

You can enable HTTP service use this element. The element have 7 attributes:

1. The "enable" attribute's value equal "true" then enable HTTPS service, else is disable HTTPS service.
2. You can set HTTPS listened port use "port" attribute. We default set it to "8443".
3. You can set certificate file of HTTPS use "keystoreFile" attribute. Please refer to addendum 6 to know how to create certificate file.
4. You can set type of certificate file use "keystoreType" attribute. If use JDK keytool to The certificate file that use JDK keytool created is "jks" type.
5. You can set certificate file's save password use "keystorePass" attribute.
6. You can set certificate file's password use "keyPass" attribute.
7. You can set server validate client certificate file use "needClientAuth" attribute. If this attribute set be "true", then need validate client certificate file, else not need validate.

All attributes can be avail when “enable” set to “true” only.

Notice: You must enable a service of HTTP or HTTPS at least.

2.3 mainHost element

You can modify “mainHost” element to config kangaroo-egg webserver main host.

```
+ <systemSet>
+ <connectors>
- <mainHost>
  <webPath>webapp</webPath>
  <defaultHttpFile>index.html</defaultHttpFile>
  <dhtmlExtName>dqm</dhtmlExtName>
  <maxPostLen>2000</maxPostLen>
  <session timeout="20" allowedMaxNumber="2000" />
  <listFile>true</listFile>
- <needPassword enable="true">
  <className>default</className>
  <parameter comment="title">mainHost</parameter>
  <parameter comment="userName">dunne</parameter>
  <parameter comment="userPsw">123456</parameter>
  <parameter comment="securityPath">/security/</parameter>
</needPassword>
- <needRedirect enable="true">
  <className>example.MaintenanceRedirect</className>
  <parameter comment="redirect url">/maintenance.htm</parameter>
</needRedirect>
  <needCompress>true</needCompress>
  <autoCompile>true</autoCompile>
- <userApps>
  - <userApp enable="true" number="1">
    <className>example.StartupTime</className>
    <parameter comment="prompt_info">Startup time =</parameter>
  </userApp>
  - <userApp enable="true" number="2">
    <className>example.PrintTime</className>
    <parameter comment="sleepTime(minute)">10</parameter>
  </userApp>
</userApps>
</mainHost>
+ <vHost number="1">
+ <vHost number="2">
```

Picture 2-3-1

1. webPath

You can set the location of main host home directory use this element.

2. defaultHttpFile

You can set main host default document use this element.

3. dhtmlExtName

You can set main host dynamic web page's extended name use this element. The kangaroo-egg webserver can define the dynamic web page's extended name in yourself. We default set it to "dqm", so the file that extended name is "dqm" can be executed.

Notice: The value of this element don't use dot sign, for example ".dqm" is incorrect value. And this element value case insensitive.

4. maxPostLen

Use this element you can set the datum size of main host allowed accept. The dynamic web page can accept data from client, for instance browser uplod file. Kangaroo-egg server will save accept data to temporary memory, so if client uploaded larger datum server maybe overflow. This element can solve this problem. For example if this element's value set to 16, that means server only can accept 16K upload datum, if upload datum size over 16K then server will prompt error information or close connection.

Notice: This element not set larger number please, because will cause memory overflow maybe.

5. session

You can config main host session use this element. About session refer to Chapter 8 please. This element has two attributes.

1. You can set the length of session timeout use "timeOut" attribute. For example set this attribute to "20" that mean session will overdue automatically when it's no activate and exceeded 20 minutes.
2. The session must need memory, so if server maintain large number sessions at same time then it maybe overflow memory. You can set session maximal number use "allowedMaxNumber" attribute. Sever will check current number of session before create a new session, if it exceeded session maximal numbe of your set then server will clear all existed sessions before create a new session. If this attribute set a negative then server session unlimited.

6. listFile

Use this element you can set allow the user to see a hypertext listing of the files and subdirectories in this host directory. If this element set be "true", then allow list files, else not allow.

Notice: Your Web server maybe will display an "Access Forbidden" error message in the user's Web browser if the user attempts to access a file or directory when listFile function disabled.

7. needPassword

You can enable your Web server's Basic authentication use this element. If this element set be

“true” that means enable web server’s authentication, else disable it. The browser will prompt the user input user name and the password when user visit host that enabled authenticate (to see picture 2-3-2).



Picture 2-3-2

This element has 5 sub-elements:

1. You can set web server’s authentication application use “className” element. See picture 2-3-1 that we set this element to “default”, it’s mean use kangaroo-egg’s default authentication application. We will introduce how to use custom authentication application later.
2. The “parameter” element has a “comment” attribute, it function is to remark, for example if you have many parameters then you can use this attribute to remark them. The “comment” attribute is not must need. See picture 2-3-1 the first “parameter” element’s “comment” attribute value is “title”. You maybe already guess the function of this element, it can set dialog box title (to see picture 2-3-2 red mark).
3. See picture 2-3-1 the next “parameter” element’s “comment” attribute value is “userName”. It’s function is set authentication application’s username.
4. See picture 2-3-1 the next “parameter” element’s “comment” attribute value is “userName”. It’s function is set authentication application’s username.
5. See picture 2-3-1 the next “parameter” element’s “comment” attribute value is “securityPath”. It’s function is set authentication area, for example if you want to protect all web site you can set this element to “/”, but if you want to protect resource that under “security” folder so you must set the element to “/security/”.

Use the default authentication application has a defect, that everyone can use a pair of username/password only. So you can use custom authentication application, for example use database to validate identity(Inquires the user name and the password from the database).

Kangaroo-egg has a interface that name is “com.kangaroo_egg.webserver.CheckServerPSW”, custom authentication class must need implement it. This interface is very simpleness, has two methods:

First method:

public String getPswTitle()

Kangaroo-egg server can get dialog box title (to see picture 2-3-2 red mark) from this method.

Second method:

public boolean checkPSW(String username, String password, String requestPath, String queryData)

Kangaroo-egg server use this method to validate identity. The parameter's value of "username" and "password" is user inputed when jump dialog box(to see picture 2-3-2). The "requestPath" parameter's value is path of user visited, but notice this path not encoded (please refer to section 2.1). The "queryData" parameter's value is characters of URL attached, if user request URL not attach any characters then is't value is null. And this method will return a boolean value, if it return true then successfully validate identity. If it return false then unsuccessfully validate identity.

Custom authentication application must implement this interface and it's two methods, then change "className" parameter's value to your authentication application's class name(must need full class name). Of course, custom authentication application's class maybe has constructor with parameters, so you can use "parameter" element to set constructor's parameters (if constructor not have parameters then user can not set "parameter" element). For example a custom authentication application class has two String type parameters, so you can define two "parameter" elements. The "parameter" element's value is constructor's parameter. The "parameter" element has a "comment" attribute, it function is to remark, for example if you have many parameters then you can use this attribute to remark them.

Ok, now let us research default authentication application, it's class full name is "com.kangaroo_egg.webserver.DefaultCheckServerPSW", so server has same function when "className" parameter's value set by "default" or "com.kangaroo_egg.webserver.DefaultCheckServerPSW". Because default class name too long, so server add this "default" value.

Now we see default authentication application source code, first this class implement "com.kangaroo_egg.webserver. CheckServerPSW" interface.

```
public class DefaultCheckServerPSW implements CheckServerPSW
```

Secondly this class constructor has four string type parameters, so need four "parameter" elements. (Four "parameter" elements see picture 2-3-1 please)

```
1 private String username, password, pswtitle, requestPath;
2 public DefaultCheckServerPSW(String ipswtitle, String iusername, String ipassword,
  String irequestPath) {
3     pswtitle = ipswtitle;
4     username = iusername;
5     password = ipassword;
6     requestPath = irequestPath;
7 }
```

This class constructor has four string type parameters, the "ipswtitle" parameter's value is

dialog box title (to see picture 2-3-2 red mark) that user use “parameter” element defined. (Source code line: 3)

The “username” parameter’s value is username that user use “parameter” element defined. (Source code line: 4)

The “ipassword” parameter’s value is password that user use “parameter” element defined. (Source code line: 5)

The “irequestPath” parameter’s value is authentication area that user use “parameter” element defined. (Source code line: 6)

Next we look default authentication application how to implement “getPswTitle()” method.

```
1 public String getPswTitle() {  
2     return pswtitle;  
3 }
```

It’s return “pswtitle” variable’s value (source code line: 2). The “pswtitle” variable is initialized by constructor.

Last we look default authentication application how to implement “checkPSW” method.

```
1 public boolean checkPSW(String iusername, String ipassword, String irequestPath, String  
  iqueryData) {  
2     if (!irequestPath.startsWith(requestPath)) {  
3         return true;  
4     }  
5     if (iusername.equals(username) && ipassword.equals(password)) {  
6         return true;  
7     }  
8     return false;  
9 }
```

First program check current visit area whether does need to protect, if does not need to protect then directly returns “true”, it’s express passed. (Source code line: 2 to 4)

If visits area needs to protected, then program will compare username and password that user inputed(iusername, ipassword) with the validated username and password which assigns (username, password). (Source code line: 5 to 7). If result identically then return “true” express passed confirmation, otherwise returns “false” express confirmation defeat. (Source code line: 8)

This default authentication application has not certainly used “iqueryData” parameter, of course you can use this parameter in your authentication application.

Above may see the custom authentication application also is so, you must element constructor, “getPswTitle” and “checkPSW” method.

Notice: The authentication application constructor’s parameters must match “parameter” elements, otherwise will has error when initialize it. The authentication application will initialized when server started, after the instance of authentication application will save to buffer. Server will

use the same that authentication application instance(in buffer) to validate identity every time. So please note the synchronized question for this. If username include colon will cause problem, because the http agreement reason, so please do not use colon. The authentication application can use common class loader to load it (refer to addendum 4 to know what is common class loader). You also can put authentication application into current host “/WEB-INF/classes” or “/WEB-INF/lib” folder (refer to section3.4), then server can load it too.

8. needRedirect

You can enable URL inexplicit redirect function use this element, but only can redirect to same host. For example, if a document name has been changed, but users still want to use the old address to access it, it can be to use this feature (let old address to redirect new address). And if website has a cumbersome URL then we can use this function to redirect it to a short URL.

If this element set be “true” that means enable redirect, else disable it. Next we introduce elements and attributes of “needRedirect”:

1. You can set redirect application use “className” element (must need full class name). We will introduce how to code redirect application later.
2. Of course, redirect application’s class maybe has constructor with parameters, so you can use “parameter” element to set constructor’s parameters. For example a user application class has two String type parameters, so you can define two “parameter” elements. The “parameter” element’s value is constructor’s parameter. But if constructor not have parameters then user can not set “parameter” element
3. The “parameter” element has a “comment” attribute, it function is to remark, for example if you have many parameters then you can use this attribute to remark them.

But how to code redirect application? Kangaroo-egg has a interface that name is “com.kangaroo_egg.webserver.RequestRedirect”, redirect class must need implement it. This interface is very simpleness, only has one method:

public URL redirectURL(URL oriUrl)

We can use this method to redirect URL of user request, the “oriUrl” is user original request URL, the URL of this method returned is redirect URL, but if this method returns null then 400 errors will be reported.

Next we give an example, if we maintaining our website now, we hope that user visit any web address always see maintenance information. We can use redirect function to do it. First we create a static web page “maintenance.htm”, its source code:

```
1 <html>
2
3 <head>
4 <meta http-equiv="Content-Language" content="zh-cn">
5 <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
6 <title>Maintenance information</title>
7 </head>
8
9 <body>
```

```

10
11 <p align="center"><b><font size="7" color="#FF0000">We are maintaining the website
    now, please visit later.</font></b></p>
12
13 </body>
14
15 </html>

```

Next we create redirect application, it's name is "example.MaintenanceRedirect", it's source code:

```

1 package example;
2
3
4 import java.net.MalformedURLException;
5 import java.net.URL;
6 import com.kangaroo_egg.webserver.RequestRedirect;
7
8 public class MaintenanceRedirect implements RequestRedirect {
9     private String redirectPath;
10
11     public MaintenanceRedirect(String redirectPath) {
12         this.redirectPath = redirectPath;
13     }
14
15     public URL redirectURL(URL oriUrl) {
16         try {
17             return new URL(oriUrl, redirectPath);
18         } catch (MalformedURLException e) {
19             return oriUrl;
20         }
21     }
22 }

```

Next we config the "needRedirect" element:

```

<needRedirect enable="true">
  <className>example.MaintenanceRedirect</className>
  <parameter comment="redirect url">/maintenance.htm</parameter>
</needRedirect>

```

We enable redirect function and set "MaintenanceRedirect" to redirect application, and this redirect application need a String parameter (source code line: 11 to 13), so we use "parameter" element to provide it. Now user visit any web address always will redirect to maintenance web page (source code line: 17), this maintenance web page defined by "parameter" element. But system maybe will throw exception when we construct URL instance, so if system throw

exception then we return original request URL (source code line: 18 to 20).

public URL redirectURL(URL oriUrl)

This method parameter is Java URL class, so we need to introduce it. Usually URL have 5 parts: protocol, host, port, path and query.

For example a URL: <http://www.kangaroo-egg.com:80/a.dqm?key=value>

It's protocol part is: http

It's host part is: www.kangaroo-egg.com

It's port part is: 80

It's path part is: /a.dqm

It's query part is: key=value

Please notice use this function only can redirect to same host, so system will ignore redirect URL's protocol, host and port. System actually use redirect URL's path and query. For example, we can revise "example.MaintenanceRedirect" source code as below:

```
1 package example;
2
3
4 import java.net.MalformedURLException;
5 import java.net.URL;
6 import com.kangaroo_egg.webserver.RequestRedirect;
7
8 public class MaintenanceRedirect implements RequestRedirect {
9     private String redirectPath;
10
11     public MaintenanceRedirect(String redirectPath) {
12         this.redirectPath = redirectPath;
13     }
14
15     public URL redirectURL(URL oriUrl) {
16         try {
17             return new URL("ftp://www.sun.com" + redirectPath);
18         } catch (MalformedURLException e) {
19             return oriUrl;
20         }
21     }
22 }
```

The final result has not change, system will not redirect to "sun" website's maintenance.htm, it still will redirect to current host's maintenance.htm.

Notice: The redirect application constructor's parameters must match "parameter" elements, otherwise will has error when initialize it. The redirect application will initialized when server

started, after the instance of redirect application will save to buffer. Server will use the same that redirect application instance(in buffer) to redirect every time. So please note the synchronized question for this. The redirect application can use common class loader to load it (refer to addendum 4 to know what is common class loader). You also can put redirect application into current host “/WEB-INF/classes” or “/WEB-INF/lib” folder (refer to section3.4), then server can load it too.

8. needCompress

You can enable compress web content use this element. Server can compress dynamic web content before it output, like this can improve performance of network bandwidth (but need use cpu resource).

Kangaroo-egg support the most mainstreams compression format (for example gzip), can fit the most browser. If client can't support receive compressed datum then kangaroo-egg will automatically disable compress function. If this value set be “true” then enable the function of compress, else disable it.

Notice: Dqm script language has independent sentence to disable or enable the compres of compress (to see section 4.3). If use dqm script language to control the function of compress then server will ignore the value of “needCompress” element.

9. autoCompile

You can enable automatically compile function use this element. We alread introduced that first server translate from dynamic web page into java file, next compile java file into class file. Only class file can be executed. But sometimes need re-compile, like dynamic web page modified. It's reasonable in the development time, because dynamic web page will often modified. But dynamic web page nearly dosen't modified in productive phase, so not need to consume the resources that monitor dynamic web page is changed.

If value of this element set be “true” then server will automatically compile dynamic web page if it modified. Other value then not automatically compile. (About automatically compile flow please refer to section F1.3)

10. userApps

You can setup own java program use this element, because this element has more content, so use section 2.4 to introduce this element. This element is not must need.

2.4 userApps element

You can modify “userApps” element to add user Java applications. For example you can add database connection pool application use this element. What are user Java applications? Very easy user Java applications will initialized when server starting.

Notice: kangaroo-egg not integrate database connection pool, but you can use third party products.

The “userApps” element can contain zero or more “userApp” elements. A “userApp” element represent a user application. Ok let me introduce all elements and attributes of “userApp”.

1. The “userApp” element has two attributes, “enable” and “number”. The “enable” attribute’s value equal “true” then enable current user application, else is disable it. The “number” attribute is not must need, it function is to remark, for example if you have many user applications then you can use this attribute to remark them.
2. You can set user application’s class use “className” element. This element must need full class name, server will initialize this class when server started.
3. Of course, user application’s class maybe has constructor with parameters, so you can use “parameter” element to set constructor’s parameters. For example a user application class has two String type parameters, so you can define two “parameter” elements. The “parameter” element’s value is constructor’s parameter. But if constructor not have parameters then user can not set “parameter” element
4. The “parameter” element has a “comment” attribute, it function is to remark, for example if you have many parameters then you can use this attribute to remark them.

Notice: You can use “parameter” element to set constructor’s parameters, but parametes must be String type only.

Ok now give a demo. “example.StartupTime” is a class, it can print current time when initialized, source code is below this.

```
1 public class StartupTime {
2     public StartupTime(String info) {
3         System.out.println(info + new java.util.Date());
4     }
5
6     public StartupTime() {
7         System.out.println(new java.util.Date());
8     }
9 }
```

So you can config userApps element to:

```
<userApps>
  <userApp enable="true" number="1">
    <className>example.StartupTime</className>
    <parameter comment="prompt_info">Startup time=</parameter>
  </userApp>
</userApps>
```

The “example.StartupTime” will be initialized when server started. Look the source code, this

class has two constructors, a constructor need a String type parameter(source code line:2-4). Ok we add a “parameter” element, it’s value is “Startup time=”, so when start server then console will print “Startup time=Fri Jan 06 16:10:04 GMT+08:00 2006”, but notice that information of “Startup time=” is our input via “parameter” element. And the time “Fri Jan 06 16:10:04 GMT+08:00 2006” will change to your current time.

But this class has another constructor that without any parameters(source code line:6-8), so how to use this constructor? Very easy not add “parameter” element at webconfig.xml, like below:

```
<userApps>
  <userApp enable="true" number="1">
    <className>example.StartupTime</className>
  </userApp>
</userApps>
```

The console will print “Fri Jan 06 16:10:04 GMT+08:00 2006” when server started. You can see that printed current time only.

Next we add second user application, it will regular print current time. It’s obvious what the class need use thread. The class name is “example.PrintTime”, source code is below this.

```
1 public class PrintTime implements Runnable {
2     long sleepTime;
3     public PrintTime(String isleepTime) {
4         sleepTime = Long.parseLong(isleepTime) * 1000 * 60; // minute
5         new Thread(this, "printTime").start();
6     }
7
8     public void run() {
9         while (true) {
10             try {
11                 Thread.sleep(sleepTime);
12                 System.out.println(new java.util.Date());
13             }
14             catch (InterruptedException ex) {
15                 // ignore exception
16             }
17         }
18     }
19 }
```

So you need add a new “userApp” element in “userApps”:

```
<userApps>
  <userApp enable="true" number="1">
    <className>example.StartupTime</className>
  </userApp>
  <userApp enable="true" number="2">
    <parameter comment="prompt_info">Startup time=</parameter>
  </userApp>
```

```
<className>example.PrintTime</className>
<parameter comment="sleepTime(minute)">10</parameter>
</userApp>
</userApps>
```

Application use a new thread to regular print current time(to see source code line: 5), at source code line 8 to 18 is thread main body. The constructor's parameter can decide how long to print current time(to see source code line: 4 and 11). So like above config that user application will print current every 10 minutes.

Notice: User application (like upper example) can use common class loader to load it (refer to addendum 4 to know what is common class loader). You also can put user application into current host “/WEB-INF/classes” or “/WEB-INF/lib” folder (refer to section3.4), then server can load it too.

2.5 vHost element

You can modify “vHost” element to config kangaroo-egg webserver virtual host. You can add more virtual host, only turns to increase “vHost” element. The “number” attribute of “vHost” element is not must need, it function is to remark number of current virtual host. Of course, you can change it to better understanding of the content, but still recommends using numeral. If you not config exactly this element then server will prompt number of that virtual host has error. (most of the top is No.1, from top to bottom, the increase).

The content of “vHost” element very like the content of “mainHost” element:


```

<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE kangaroo-egg (View Source for full doctype...)>
- <kangaroo-egg>
+ <systemSet>
+ <connectors>
+ <mainHost>
- <vHost number="1">
  <hostName>download.kangaroo-egg.com</hostName>
  <webPath>d:\webapp</webPath>
  <defaultHttpFile>default.html</defaultHttpFile>
  <dhtmlExtName>dqm</dhtmlExtName>
  <maxPostLen>3000</maxPostLen>
  <session timeout="30" allowedMaxNumber="2400" />
  <listFile>true</listFile>
- <needPassword enable="false">
  <className>default</className>
  <parameter comment="title">vHost1</parameter>
  <parameter comment="userName">admin</parameter>
  <parameter comment="userPsw">123456</parameter>
  <parameter comment="securityPath">/</parameter>
  </needPassword>
  <needCompress>true</needCompress>
  <autoCompile>true</autoCompile>
</vHost>
+ <vHost number="2">
</kangaroo-egg>

```

Picture 2-5-1

Seen from picture 2-5-1 that we distribute two virtual hosts on kangaroo-egg server, the second virtual host's elements same with first, only different point is element's value. We use second virtual host as example, please see picture 2-5-2. Please notice that first virtual host not config "userApps" and "needRedirect" element, this means that no user application and disable redirect function.

```

<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE kangaroo-egg (View Source for full doctype...)>
- <kangaroo-egg>
+ <systemSet>
+ <connectors>
+ <mainHost>
+ <vHost number="1">
- <vHost number="2">
    <hostName>www.kangaroo-egg.com|kangaroo-egg.com</hostName>
    <webPath>d:\kgweb</webPath>
    <defaultHttpFile>default.html</defaultHttpFile>
    <dhtmlExtName>dqm</dhtmlExtName>
    <maxPostLen>3000</maxPostLen>
    <session timeOut="40" allowedMaxNumber="1500" />
    <listFile>false</listFile>
- <needPassword enable="false">
    <className>default</className>
    <parameter comment="title">vHost2</parameter>
    <parameter comment="userName">admin</parameter>
    <parameter comment="userPsw">654321</parameter>
    <parameter comment="securityPath">/</parameter>
    </needPassword>
- <needRedirect enable="true">
    <className>test.Redirect</className>
    </needRedirect>
    <needCompress>true</needCompress>
    <autoCompile>true</autoCompile>
- <userApps>
    - <userApp enable="true" number="1">
        <className>test.Test</className>
    </userApp>
    </userApps>
    </vHost>
</kangaroo-egg>

```

Picture 2-5-2

1. hostName

The “mainHost” element not include this sub-element. You can set virtual host’s name(domain name, host head) use this element. Of course, a virtual server can set multi-hostname(use “|” flag to split values), see picture 2-5-2 that the second virtual host set two host name: “www.kangaroo-egg.com” and “kangaroo-egg.com”.

For example, if our server IP address is “210.52.216.34”. Meanwhile “www.kangaroo-egg.com” and “kangaroo-egg.com” two domain names are at this IP address. If user visit those two domain name then will see No.2 virtual host’s resource. See picture 2-5-1 that No.1 virtual host name is “download.kangaroo-egg.com”, and this domain name is at our server IP address, so user visit this domain name will see No.1 virtual host’s resource. So another question that user directly visit IP address will see main host’s resource. Why? Because if host name(domain name) of user visited was not defined in all “vHost” elements then server will automatically reset to main host. This is the reason that why “mainHost” element not include “hostName” sub-element.

2. webPath

You can set the location of virtual host home directory use this element.

3. defaultHttpFile

You can set main host default document use this element.

4. dhtmlExtName

You can set main host dynamic web page's extended name use this element. The function of this element please refer to section 2.3.

5. maxPostLen

Use this element you can set the datum size of main host allowed accept. The function of this element please refer to section 2.3.

6. session

You can config main host session use this element. The function of this element please refer to section 2.3.

7. listFile

Use this element you can set allow the user to see a hypertext listing of the files and subdirectories in this host directory. The function of this element please refer to section 2.3.

8. needPassword

You can enable your Web server's Basic authentication use this element. The function of this element please refer to section 2.3.

9. needRedirect

You can enable redirect function use this element. The function of this element please refer to section 2.3.

10. needCompress

You can enable compress web content use this element. The function of this element please refer to section 2.3.

11. autoCompile

You can enable automatically compile function use this element. The function of this element please refer to section 2.3.

12. userApps

You can setup own java program use this element. The function of this element please refer to section 2.3.

2.6 webfile folder

That has a webfile folder under the conf folder, all HTTP message files are saved at this folder. For example, a user visited a inexistent document then kangaroo-egg will output message file of code is 404 from this directory. You can customize the user message file to replace the default file. But notice message file must follow this norms: file extended name must be “html”, file main name must use underscore flag and region name, for example region of United States then message main file name is end by “_us”.

Chapter 3 DQM technology

Dynamic web page file can easy create and manage dynamic content, DQM technology of kangaroo-egg is a dynamic web page technology. This chapter will be introduce the syntax of dqm script language. About work flow of DQM please refer to addendum 1.

3.1 Introduce DQM

Add fragments of java program into traditional web HTML document then become dynamic web page(DQM web page). Dynamic web page can use database, redirects URL and other advanced features. Because dynamic web page executed at server, so network transmission to the client is only the result.

3.2 DQM directives

Directives content is inside “<%@” and “%>”, you can use it to set attribute of dynamic web. Now kangaroo-egg has three directives.

1. Directive of “page”

Syntax: <%@page attributeA="valueA" attributeB="valueB" %>

Attribute name is case insensitive. This directive can write at dqm file (dynamic web page) any line. If a same directive wrote at dqm file many times, then the last directive is available.

| Attribute of page directive | Function | Example |
|-----------------------------|--|---|
| contenttype | Set response result's MIME type. If not set this attribute then server use default value “text/html”. | <%@page contenttype= "text/html; charset=GB2312" %> |
| buffer | Whether to enable buffer function, the value can only set to “true” or “false”. If not set this attribute then server default to disable buffer function. | <%@page buffer = "true" %> |
| compress | Whether to enable content compress function, the value can only set to “true” or “false”. If not set this attribute then server will use current host “needCompress” element's value to decide enable or disable compress function. (About “needCompress” element please refer to section 2.3 and 2.5) | <%@page compress = "true" %> |
| import | Import java classes and packages, you can use comma or semicolon to split the classes that need to import. You can repeatedly used this directive to add different classes and packages. Even if not set this attribute the server also will default import “java.io.*” and “com.kangaroo_egg.dqm.*”. | <%@page import = "java.awt.*; java.awt.image.BufferedImage, javax.imageio.ImageIO" %> |
| session | Whether to enable session on current dqm file, the value can only set to “true” or “false”. If not set this attribute then server default to enable session function. (About session please refer to chapter 8. | <%@page session = "true" %> |
| command | Whether to enable “command” function on current | <%@page command = "true" %> |

| | | |
|--------------|--|-----------------------------------|
| | dqm file, the value can only set to “true” or “false”. If not set this attribute then server default to disable command function. (About command we will introduce later.) | |
| isthreadsafe | Whether to enable safe thread mode function on current dqm file, the value can only set to “true” or “false”. If this attribute set to “true”, then the current dqm file is safe when many user visit it at same time. If this attribute set to “false”, then the same time current dqm file can only be visited by a single user. If not set this attribute then the value default is “true”. | <% @page isthreadsafe = "true" %> |

Table 3-2-1

2. Directive of “include”

Dqm file (dynamic web page) can use “include” directives to include other files(dqm files, html files, or other type files). If the file (included use “include” directives) containing fragments of java program or directives, then these code were also executed.

Syntax: <% @ include file="filename"%>

In the development of web applications, if most dqm file contains the same content then you can let this part of the same content into a single file, and use “include” directives to include this file. This can increase performance while maintaining.

Notice: Include directive only can hold a single file attribute. For example:

```
<% @ include file="1.txt" file="2.txt"%>
```

Then server will inform user that error syntax.

Underside is correct syntax:

```
<% @ include file="1.txt"%><% @ include file="2.txt"%>
```

The space of at file’s head and end will be ignored. For example:

```
<% @ include file=" 1.txt  "%> equal <% @ include file="1.txt"%>
```

The include file can use relative path or absolute path. Relative path is relative to the current dqm file’s path. For example: “../xxx.txt”, “test/xxx.txt”. Absolute path’s example is “c:\xxx.txt”, “/xxx.txt”.

3. Directive of “bean”

It’s very similar to the JSP javabean, the function is repeated use components. Please refer to Chapter 11.

3.3 Fragments of java program

You can insert fragments of java program into dqm file. A simple example, hello.dqm, source code:

```
1 <%
2 String s = "hello!";
3 out.print(s);
4 %>
```

When user visit “hello.dqm” the result is “hello!”. You can see the fragments of java program of between “<%” and “%>”. However, you may have noticed that has a “out” object, it’s a built-in object. Kangaroo-egg server has six built-in object: out, request, response, session, application and command. Later, we will introduce these objects.

3.4 Dqm file released

How to release dqm files? Very easy, only put the dqm files into host or virtual host home directory (webpath). The home directory has specific structure.

| folder | description |
|---|--|
| home directory (root directory) use “webPath” element set it | The location of main host or virtual host home directory, all the dynamic web files (dqm files), html files and other resources are stored under this directory. |
| home directory/WEB-INF/classes | All the java class files that dqm file need used are stored under this directory. |
| home directory/WEB-INF/lib | All the java jar or zip files that dqm file need used are stored under this directory. For example, you can put JDBC Driver jar file under this folder. |
| home directory/WEB-INF/work | Kangaroo-egg will put dqm compiled files under this folder. |

Table 3-4-1

From table 3-4-1 you can seen that java file should put under “classes” and “lib” directory, but there is some difference. The java class file only can be stored under “classes” folder. The jar or zip file (many java class files can pack into a jar or zip file) only can be stored under “lib” folder.

If “classes” and “lib” folder containing the same name class file then server will priority load file that at “classes” folder.

Notice: The class file that at “classes” or “lib” folder only can loaded by current main host or virtual host.

The “WEB-INF” is not necessarily, if you not use dqm file(only use static html file) then this folder an not existed. If “classes” and “lib” folder (that under “WEB-INF” folder) not existed, so you can create it manually.

If you enable automatically compile dqm file then server will automatically create the “WEB-INF” folder when compile dqm file. About automatically compile please refer to section 2.3 and 2.5.

The dqm file name must conform to the java identifier. For example a dqm file name is “he-llo.dqm”, but this dqm file name is invalid. Because file name include “-” symbol, this symbol can’t be part of a Java identifier. And the folder that dqm file under it must conform to the java identifier too.

In windows OS that file name is case insensitive, but java is case sensitive, so maybe will happen the status of compiled folder confused. For example a web application with a “sd” folder, and a name of “hello.dqm” file under “sd” folder.

Folder structure: `web application root/sd/hello.dqm`

When user visit this dqm file then server will compile it frist, so you can find compiled file under “WEB-INF” folder.

Compiled folder structure: `WEB-INF/work/com/kangaroo_egg/workfile/sd/hello_dqm.class`

Next we change web application “sd” folder to “Sd”, and create a new dqm file “hello1.dqm” under it.

Now folder structure: `web application root/Sd/hello1.dqm`

`web application root/Sd/hello.dqm`

When user visit new file “hello1.dqm” then server will compile it first, so you can find new compiled file under “WEB-INF” folder.

Compiled folder structure: `WEB-INF/work/com/kangaroo_egg/workfile/sd/hello_dqm.class`

`WEB-INF/work/com/kangaroo_egg/workfile/sd/hello1_dqm.class`

You can seen that “sd” folder(under WEB-INF) not be changed, because windows think two directory(“sd” and “Sd”) is the same. But this time problem is coming, java is case sensitive, so mistakes will happen when execute “hello1_dqm.class”, because “hello1_dqm.class” file must under “Sd” folder.

Ok you maybe think I can chang the folder name “sd” to “Sd”:

`WEB-INF/work/com/kangaroo_egg/workfile/Sd`

But it’s not feasible, because “hello_dqm.class” must under “sd” folder.

Like this status we call it compiled folder confused, to solve it must delete this folder and all under it files: `WEB-INF/work/com/kangaroo_egg/workfile/sd`, and recompile.

Now kangaroo-egg server can detect this status and solve it automatically, but sometimes maybe can’t delete confused folder, then will prompt user delete it manually.

Please don’t change folder name that under “WEB-INF” folder.

Chapter 4 Built-in object : out

Earlier chapters introduced that dqm technology has six build-in objects. At this chapter we will go through these built-in objects, first introduced “out” object.

The “out” object can control sent to the client content.

4.1 Methof of print and println

The “out” object has two important methods: “print” and “println”. It’s function to transmit character data to client.

1. public void print(String content, String charsetName) throws IOException

Output character using the specified charset.

Parameters: content - character that need to output

charsetName - the name of a supported charset, for example BIG-5, UTF-8, GBK...

Throws: If happen problem when output then this method will throws IOException.

2. public void print(String content) throws IOException

Output character using default charset. The default charset is set by “webconfig.xml->systemSet->dataEnc”, please refer to section 2.1.

Parameters: content - character that need to output

Throws: If happen problem when output then this method will throws IOException.

3. public void print(boolean content) throws IOException

Output a value that type is Boolean.

Parameters: content – boolean type value that need to output

Throws: If happen problem when output then this method will throws IOException.

4. public void print(char content) throws IOException

Output a character.

Parameters: content – a character that need to output

Throws: If happen problem when output then this method will throws IOException.

5. public void print(char[] content) throws IOException

Output character array.

Parameters: content – character array that need to output

Throws: If happen problem when output then this method will throws IOException.

6. public void print(double content) throws IOException

Output a value that type is double.

Parameters: content – double type value that need to output

Throws: If happen problem when output then this method will throws IOException.

7. public void print(float content) throws IOException

Output a value that type is float.

Parameters: content – float type value that need to output

Throws: If happen problem when output then this method will throws IOException.

8. public void print(int content) throws IOException

Output a value that type is int.

Parameters: content – int type value that need to output

Throws: If happen problem when output then this method will throws IOException.

9. public void print(long content) throws IOException

Output a value that type is long.

Parameters: content – long type value that need to output

Throws: If happen problem when output then this method will throws IOException.

10. public void print(Object content) throws IOException

Output a object.

Parameters: content – object that need to output

Throws: If happen problem when output then this method will throws IOException.

The above method has a simple syntax.

`<%=content%>` it's equals `<%out.print(content);%>`

And “println” method is output value and then terminate the line.

For example a dqm file: print_test.dqm, source code:

```
1 <pre>
2
3 <%
4 out.println("Hello world", "US-ASCII");
5 out.println("Hello world ");
6 out.println(true);
7 out.println('H');
8 char[] chars = {'H','e'};
9 out.println(chars);
10 out.println(3.14d);
11 out.println(3.14f);
12 out.println(3);
13 out.println(314l);
14 Object obj = new Object();
15 out.println(obj);
16 %>
17
18 </pre>
```

Output result:

```
Hello world
Hello world
true
H
He
3.14
3.14
3
314
java.lang.Object@1bde4
```

4.2 Method of write

You can use “write” method to transmit binary data to client. For example we can use this method to output binary data(picture) of save in database.

1. **public void write(byte[] b, int off, int len) throws IOException**

Writes len bytes from the specified byte array starting at offset off to this output stream(client). The general contract for write(b, off, len) is that some of the bytes in the array b are written to the output stream in order; element b[off] is the first byte written and

b[off+len-1] is the last byte written by this operation.

Parameters: b - the data.

off - the start offset in the data.

len - the number of bytes to write.

Throws: If happen problem when output then this method will throws IOException.

2. public void write(byte[] b) throws IOException

Writes b.length bytes from the specified byte array to this output stream(client). The general contract for write(b) is that it should have exactly the same effect as the call write(b, 0, b.length).

Parameters: b - the data.

Throws: If happen problem when output then this method will throws IOException.

4.3 Method of compress web content

Earlier chapters introduced that main host or virtual host can use “needCompress” element to enable or disable compress web content function(Refer to section 2.3 or 2.5), but it only can control compress function of all dqm file. In section 3.2 we known that can use “compress” attribute of “page” directive to set single dqm file’s compress funtion. But maybe you will think that use directive to control compress function is not agility too. Ok, kangaroo-egg provide method to contrl single dqm file’s compress funtion.

1. public boolean setEnableCompress(boolean vaule)

Enable/disable compress output content of current dqm file. If not use this method in the dqm file, then kangaroo-egg will use current host “needCompress” element’s value to decide enable or disable compress funtion. About “needCompress” element please refer to section 2.3 and 2.5. If use this method many times then the last set is available. Only enable buffer function then can use this method. About buffer function refer to section 3.2 please. If disable buffer function then only can use “needCompress” element or “compress” attribute of “page” directive to control compress function. The current method can decide need to compress on real-time. For example you can enable compress function when a lot of content need to output. You can disable compress function when less content need to output. You can use “getBufSize” method to acquire output content size (“getBufSize” method can refer to section 4.4).

Parameters: value - a boolean indicating whether enable compress function.

Returns: If return true that mean is really enable compress function. Really enable? Yes, because in some cases such as the browser does not support decompression, then even you enable compress function but the fact is can not be compressed (This status will return false).

Notice: There is a greate difference between “compress” attribute of page directive and current method is that current method can use Boolean variable, “compress” attribute can’t use variable.

2. **public boolean getIsEnableCompress()**

Check really status of compress function. This method return value same as “setEnableCompress” method return value. Use this method can easy to check compression status at any time.

Return: If return true that mean is really enable compress function, else not.

3. **public String getCompressName()**

Get compression format. For example if the browser can support “gzip” format, the server can support “gzip” format, then return “gzip”. If the browser only can support “zip” format, the server can support “gzip” and “deflate”, then return “no”. Sometimes the browser and server can support many compression formats, then server will use default format. Now kangaroo-egg only support “gzip” and “deflate” formats, default is use “gzip”. So if browser can support “gzip” and “deflate” then kangaroo-egg will use “gzip” also. And notice the value of this method returned is unaffected by enable compress function.

Return: Compression format of client and server all can support.

For example “compress.dqm”, source code:

```
1  <% @ page buffer="true"%>
2
3  <pre>
4
5  Enable compress function
6  <%
7  out.setEnableCompress(true);
8  %>
9  Is enable compress function now: <%=out.getIsEnableCompress()%>
10 Compression format of client and server all can support: <%=out.getCompressName()%>
11
12 Disable compress function
13 <%
14 out.setEnableCompress(false);
15 %>
16 Is enable compress function now: <%=out.getIsEnableCompress()%>
17 Compression format of client and server all can support: <%=out.getCompressName()%>
18
19 </pre>
```

If your browser support “gzip” compression format then result:

```
1  Enable compress function
2
3  Is enable compress function now: true
4  Compression format of client and server all can support: gzip
```

```

5
6 Disable compress function
7
8 Is enable compress function now: false
9 Compression format of client and server all can support: gzip

```

Program will enable compress function at source code line 7, next program check whether really enable compress function(at source code line 9). Because we visit this dqm file by IE(IE browser support compress function), so result shown the compress function was really enabled(result line: 3).

At last program will disable compress function(source code line: 14), and check whether really disable compress function(at source code line 16). You can seen from result line 8 that was eally disabled.

You can seen from result line 4 and 9, that client support gzip compression format. Because we use IE browser and it unaffected by enable compress function.

The results maybe not same under different environments, because some browsers do not support compression format, even if it may not have support gzip compression format.

4.4 Method of buffer

In section 4.3 we again referred to the buffer, if you want to use method of “setEnableCompress” then must enable buffer function. But what is buffer function? Program will simultaneous execute and output when disable buffer function. If enable buffer function then first server will save all result to buffer until executive over, next output all results that in buffer to client. The benefits of buffer function is you can can modify the contents of the output when executing program, because result will first save in buffer. But the buffer needs temporarily to use the memory. Other if the program need long time to executed, then client get result need wait a long time too unit process of implementation complete.

Next we introduce two methods of operate buffer.

1. **public int getBufSize()**

Get current dqm file buffer size.

Return: If disable buffer function then return -1, else return the current dqm file buffer size.

2. **public void clear()**

Clear all of the current dqm file buffer data. If the buffer function disabled then this method do anything.

For example “buffer.dqm”, source code:

```

1 <% @ page buffer="true"%>
2

```

```

3
4 Content A
5
6 <%out.clear();%>
7
8 <pre>
9 Clear all of the current dqm file buffer data
10 current dqm file buffer size: <%=out.getBufSize()%>
11
12 Content B
13 current dqm file buffer size: <%=out.getBufSize()%>
14 </pre>

```

Output result:

Clear all of the current dqm file buffer data
current dqm file buffer size: 88

Content B
current dqm file buffer size: 135

The result not output “contentA”(source code line: 2 to 5), because we clear all buffer data at source code line 6. You can seen from result that the content(at source code line 7 to 10) size is 88 bytes, the content(at source code line 7 to 13) size is 135 bytes.

4.5 Directly output data

Built-in object “out” extends “java.io.OutputStream”, so base on polymorphism feature, you can think built-in “out” equal “java.io.OutputStream”.

For example “screen.dqm”, it can capture the server screen, then output screenshot to client.

Source code:

```

1 <% @page ContentType="image/png" buffer="true"
2 import="java.awt.*; java.awt.image.BufferedImage; javax.imageio.ImageIO"%><%
3
4 try {
5     Toolkit toolkit = Toolkit.getDefaultToolkit();
6     Dimension ScreenDim = toolkit.getScreenSize();
7     Robot rb = new Robot();
8     BufferedImage img = new BufferedImage(ScreenDim.width, ScreenDim.height,
9     BufferedImage.TYPE_BYTE_GRAY);
10    Graphics2D gImg = img.createGraphics();
    BufferedImage cutimg = rb.createScreenCapture(new

```

```

    Rectangle(ScreenDim.width,ScreenDim.height));
11     gImg.drawImage(cutimg, null, 0, 0);
12     ImageIO.write(img, "png", out);
13 }
14 catch (Exception e) {
15     e.printStackTrace();
16 }
17 out.setEnableCompress(false);    //Also    can    enable    compress    function.
    out.setEnableCompress(true);
18 %>

```

The execute result is server's screenshot. First program initialize java robot (At source code line 7). Next initialize a canvas that size same with server desktop size (At source code line 8 and 9). Next let java robot capture the server screen (source code line 10), and save screenshot to canvas (source code line 11). Last output canvas to client.

The "ImageIO.write" method at source line 12:

```

public static boolean write(RenderedImage im, String formatName, OutputStream output)
throws IOException

```

You can see No.3 parameter's type is "OutputStream". This place we send built-in "out" object to No.3 parameter, this can prove that built-in "out" equal "java.io.OutputStream".

Notice: The last flag at source code line 2 must be "%><%", not contain any blank like "%> <%". If contain blank then will output a invalid picture, because program will output blank character to binary picture file.

Other linux and unix OS must run this program under GUI.

Chapter 5 Built-in object : response

You can operate server response use this object, example set HTTP head attribute.

5.1 Redirect

You can add link into web page, then user can click the link to visit other page. But this operate need user click link, but sometimes we hope at server side can automatically guide (redirect) client to another page. For example, online examinations, system will automatically jump terminal page

when examination concluded. The “sendRedirect” method can do it.

public void sendRedirect(String url) throws IOException

Sends a temporary redirect response to the client using the specified redirect location URL. The URL can be absolute (for example, <https://hostname/path/file.html>) or relative (for example, ../path/file.html). Only enable buffer function then can use this method. And behind this method must follow “return” syntax.

Parameters: url- the redirect location URL

Throws: IOException - If an I/O error has occurred.

For example “redirect.dqm”, source code:

```
1 <% @page buffer="true" import="java.util.Random"%>
2 <%
3 String yahoo = "http://www.yahoo.com";
4 String sun = "http://www.sun.com";
5
6 Random rdom = new Random();
7 int rs = rdom.nextInt(10);
8 if (rs >=5) {
9     response.sendRedirect(yahoo);return;
10 }
11 else {
12     response.sendRedirect(sun);return;
13 }
14 %>
```

Visit this dqm file, maybe will open “yahoo” homepage, maybe will open “sun” homepage. If random number larger than five then redirect to “yahoo” website(source code line: 8 to 10), else redirect to “sun” website(source code line: 11 to 13).

Behind “sendRedirect” method must follow “return” syntax, so sometimes will happen status of unreachable statement. Example below source code, the unreachable statement at line 2.

```
1 response.sendRedirect(yahoo);return;
2 out.print("hello");
```

Please change code to:

```
1 f(true) {
2     response.sendRedirect(yahoo);return;
3 }
4 out.print("hello");
```

The compiler will check the dynamic document, if behind “sendRedirect” method not follow “return” syntax then will prompt error information. But if “sendRedirect” method in a bean, and use this bean to redirect, then compiler can’t check it. So please don’t forget add “return” too.

For example:

```
1  <% @page buffer="true" import="java.util.Random"%>
2  <%
3  String yahoo = "http://www.yahoo.com";
4  String sun = "http://www.sun.com";
5
6  Random rdom = new Random();
7  int rs = rdom.nextInt(10);
8  if (rs >=5) {
9      myRedirect(response, yahoo);return;
10 }
11 else {
12     myRedirect(response, sun);return;
13 }%><%!
14 private void myRedirect(DResponseIface response, String url) throws IOException {
15     response.sendRedirect(url);
16 }
17 %>
```

That “sendRedirect” method writed in declaration(source code line: 13 to 17). To add a declaration, you must use the <%! and %> sequences to enclose your declarations. So at source code line 9 and 12, if not add “return” then compiler also not prompt any errors. But please don’t forget add “return” too.

5.2 Set HTTP head’s attribute

Http content inclue “http head” and “http body”. Http head can define some parameters of transmit, so you can add or modify some parameters to enable some special function.

1. **public void setHeader(String name, String value, String charsetName) throws IOException**

Adds a field to the response header with the given name and value(using the specified charset encoding this value). If the field had already been set, the new value overwrites the previous one. The containsHeader method can be used to test for the presence of a header before setting its value. Only enable buffer function then can use this method.

Parameters: name - the name of the header field.

value - the header field's value.

charsetName - the name of a supported charset, for example BIG-5, UTF-8, GBK... Use it to encoding “value” parameter’s value.

Throws: If happen problem when add a field to the response header then this method will throws IOExeption.

2. **public void setHeader(String name, String value) throws IOException**

Adds a field to the response header with the given name and value(using default charset encoding this value, the default charset is set by “webconfig.xml->systemSet->dataEnc”, please refer to section 2.1). If the field had already been set, the new value overwrites the previous one. The containsHeader method can be used to test for the presence of a header before setting its value. Only enable buffer function then can use this method.

Parameters: name - the name of the header field.

value - the header field's value.

Throws: If happen problem when add a field to the response header then this method will throws IOException.

But what can we do use this method? Ok for example, when users visit a website, we do not want to allow users to directly see the results in browser, but want to download it, so we can add a new http head to achieves the goal. The dqm file “httphead.dqm”, souce code:

```
1 <% @page buffer="true"%>
2 <%response.setHeader("Content-Disposition",
  "attachment;filename=http_compression.txt");%>
3
4 Why Compress?
5
6 Most user's knowledge of compression is from compressing a group of files that they
  download, extract, and open. But compression can also be used passively to compress
  documents as they are being transferred to a client's browser. Because it's a passive
  process, the server can reduce the size of the pages sent, therefore reducing the download
  time for users and their bandwidth usage.
```

The browser will prompt to download "http_compression.txt" file when you visiting this dqm file. Not to directly show content in the browser. You can try to delete source code line 2, then visit this dqm file again, then you will find the content was directly shown in browser. You can seen we add a new http head “Content-Disposition” to achieves the goal, this http head’s value is “attachment;filename=http_compression.txt”, the “http_compression.txt” is download file’s name.

Section F2.3 will introduce how use “http head” to output international content.

5.3 Setup ContentType

Section 3.2 introduced that page directive can define MIME type, now response object can define it too.

public void setContentType(String contentType)

Sets the content type for this response. This type may later be implicitly modified by addition

of properties such as the MIME charset=<value> if the service finds it necessary, and the appropriate media type property has not been set. If use this method many times then the last set is available. Only enable buffer function then can use this method.

Parameters: contentType - the content's MIME type.

Notice: There is a great difference between “contentType” attribute of page directive and current method is that current method can use String variable, “contentType” attribute can’t use variable.

5.4 Method of encodeURL

Because session function is base on cookie function(session function please refer to chapter 8), so if browser disable cookie then session also expired. Even browse disable cookie function that session function will normal when use “encodeURL” method. The basic principle is that if the browser does not support cookie function then this method will rewrite the request URL, add session information to request URL.

public String encodeURL(String url)

Encodes the specified URL by including the session ID in it, or, if encoding is not needed, returns the URL unchanged. The implementation of this method should include the logic to determine whether the session ID needs to be encoded in the URL. For example, if the browser supports cookies, or session tracking is turned off, URL encoding is unnecessary.

Parameters: the url to be encoded.

Returns: the encoded URL if encoding is needed; the unchanged URL otherwise.

This method first will check the current dqm file whether enable session function(How to enable or disable session function please refer to section 3.2). If disabled session function then this method will return the URL unchanged.

For example, a URL before revision:

After uses the “encodeURL” method to revise:

<a href="<%=response.encodeURL("yourfile.dqm")%>">

When the browser does not support cookies, the URL will be automatically revised to “yourfile.dqm? DQMSESSIONID=XXX”, the “XXX” represents sessionID.

Notice: When cookie is not available, this method will automatically add sessionID to URL, sessionID parameter name is "DQMSESSIONID", it's a reservation name, so please don't use this name to transmit parameter or data.

5.5 Using cookie

HTTP cookies are used by Web servers to differentiate users and to maintain data related to the user during navigation, possibly across multiple visits. Using cookie that server can save some data into client, and server can read these data when client visit again.

Use cookie can do some special features, such as when user first visit may inquire the visitor name, then save the visitor name to user's browser. On next time this user visit again, server can take visitor's name from the browser, ok now you can use visitor's name to say hello.

1. **public void addCookie(DCookie cookie)**

Adds the specified cookie to the response. It can be called multiple times to set more than one cookie.

Parameters: the Cookie to return to the client

And how to use it and how to create DCookie we will introduce at chapter 7.

5.6 Insert static document

In section 3.2 we introduced “include” directive, here we will also introduce “includeFile” method. But these two usages also have the difference. Use “include” directive to insert a document, if this document's content has fragments of java program, then it's java program will executed too. And this method will check the document which inserts always, the dqm file that use “include” directive will be recompiled automatically if it changes.

The “includeFile” method don't check the document which inserts, the dqm file that use “includeFile” method will not be recompiled automatically if it changes. This method only for insert static document, the document content which inserts will be output intact.

1. **public void includeFile(String fileName) throws java.io.IOException**

To insert a static document, it's content will be output to client intact.

Parameters: filename – static document file name of need to insert. It can use relative path or absolute path. Relative path is relative to the current dqm file's path. For example: “../xxx.txt”, “test/xxx.txt”. Absolute path's example is “c:\\xxx.txt”, “/xxx.txt”.

Throws: If happen problem when output then this method will throws IOException.

For example, a dqm file name is “includeFile.dqm”, source code:

```
1 <pre>
2 Below is in the includeFile.txt document's content-----
3 <%response.includeFile("includeFile.txt");%>
```

```
4 | Above is in the includeFile.txt document's content-----
5 | </pre>
```

Output result:

Below is in the includeFile.txt document's content-----

The Sherlock Holmes Journal is published twice a year, usually in July and December. It is the official voice of the Society and contains its Transactions, news and reviews, letters and editorial notes. It is also home to the most erudite scholarship, publishing learned articles from Holmesians world-wide who have something to say on any aspect of Sherlock Holmes and his world. It has been appearing without a break since the first issue in May 1952.

The most recent issue, Vol 28 No 2 Summer 2007, contains the following articles, as well as the usual notes, reviews and letters

Above is in the includeFile.txt document's content-----

We can seen from result that dqm file output “includeFile.txt” document’s content intact(at source code line 3).

Notice: From the above example you can seen that use relative path(at source code line 3). The “includeFile.txt” file and “includeFile.dqm” file in the same directory.

5.7 Option output

In the BBS web application maybe happen this situation, the user can upload files to BBS, But downloads these files need some conditions, for example condition of user already sign in BBS. Then how to achieve this function? Of course we can use “out” method to code this program(use “out” method to do it please refer to section 4.2 and section 4.5). But code this program is not easy, if you want to support resume download then it’s a not small program. If you want to fully comply with the HTTP protocol then it will become a big project. So what better way? Next we will introduce “outBinFile” method.

1. **public void outBinFile(File out_bin_file) throws IOException**

Output the file to client.. Only enable buffer function then can use this method. And behind this method must follow “return” syntax.

Parameters: out_bin_file – the file need to be output.

Throws: If happen problem when output then this method will throws IOException.

First example, dqm file “outputfile.dqm”, it can output a file to client. If request url has a “type” parameter and it’s value is “download”, then can download file, other status will prompt can’t to download file. Source code:

```
1 | <% @ page buffer="true"%>
2 |
```

```

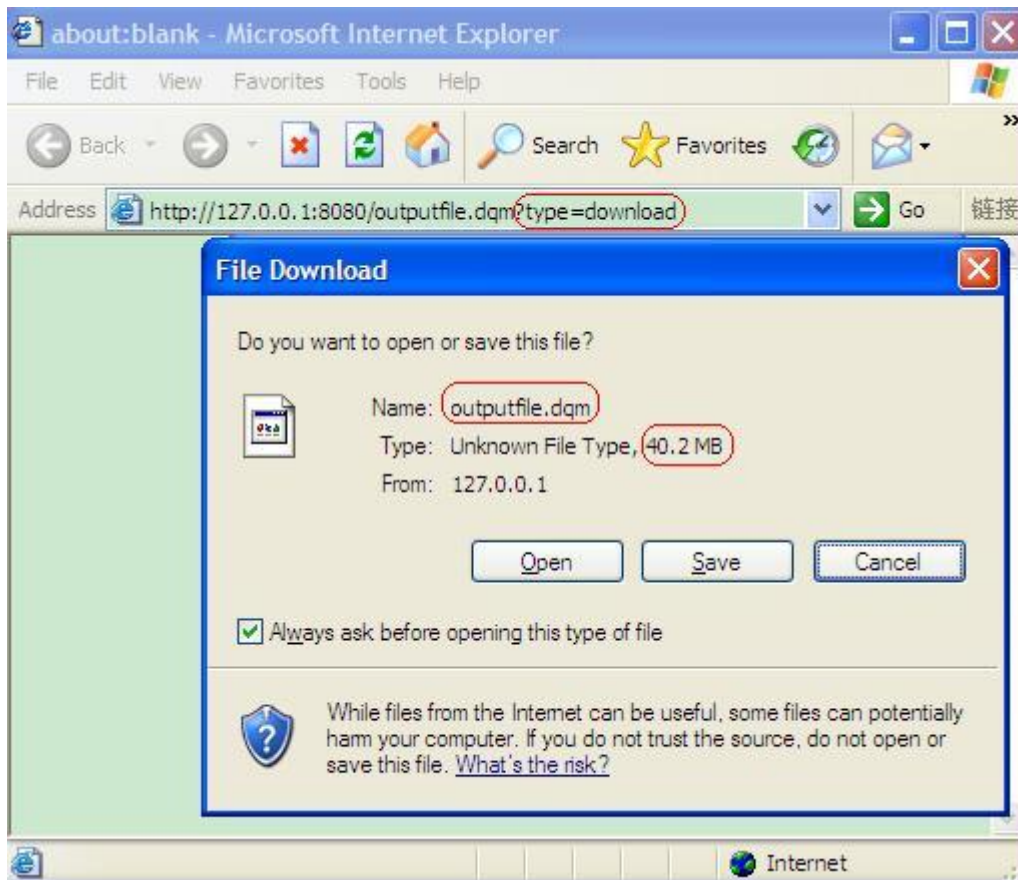
3 <html>
4
5 <head>
6 <meta http-equiv="Content-Language" content="zh-cn">
7 <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
8 <title>Download Page</title>
9 </head>
10
11 <body>
12
13 <%
14 String type = request.getParameter("type");
15
16 if ("download".equals(type)) { //can download file
17     File downFile = new File("c:\\kgweb.rar");
18     if (downFile.exists()) {
19         response.outBinFile(downFile);
20         return;
21     }
22 }
23 %>
24
25 <p align="center"><b><font color="#FF0000">Sorry, can not download!</font></b></p>
26
27 </body>
28
29 </html>

```

You can seen from source code line 16, if “type” parameter’s value is “download” then output the specified file(At source code line:17). Here we specified the output file is “c:\\kgweb.rar”, reader please change it when you test this program.

Notice: If use relative path at soure code line 17, for example chang it to `File downFile = new File("kgweb.rar")`, that mean the “kgweb.rar” file was located at kangaroo-egg installed folder(please refer to section 1.2). And please check the file is existed berfor output it (source code line 18).

When we use above parameter to visit “output.dqm”, then the browser will prompts the user to download files. Please refer to picuter 5-7-1.

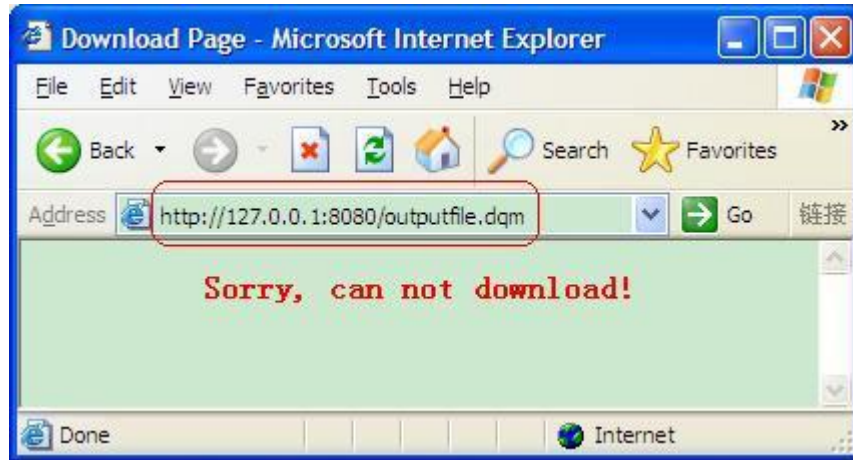


Picture 5-7-1

Seen picture 5-7-1 that URL has “type” parameter and it’s value is “download”, so that browser was prompted the user to download file. But notice that file name of need to download is “outputfile.dqm”, don’t worry, we will introduce how to change it later. Other we can seen that downloading file size is 40.2M, because c:\kgweb.rar file size is 40.2M.

You can use the URL (in picture 5-7-1) to test resume download, the results you will not be disappointed. This method not only support resume download, and all of the HTTP protocol specification transmission function will complete support, example ff the client requests has “if-modified-since” head tag, then it will be based on conditions to output the request document or output HTTP304 information.

If the visit URL hasn’t parameter or the file of need to output dosen’t exist , then program will promptthe file can’t be downloaded (source code line 25, and see picture 5-7-2).



Picture 5-7-2

Reference to the above example we can achieve different option output.

Next we have to solve the output filename problem, “response” object has another “outBinFile” method, use this method you can specify output file name

2. public void outBinFile(File out_bin_file, String saveAsNameStr) throws IOException

Output the file to client, and use default charset to reset output file’s name. The default charset is set by “webconfig.xml->systemSet->dataEnc”, please refer to section 2.1. Only enable buffer function then can use this method. And behind this method must follow “return” syntax.

Parameters: out_bin_file – the file need to be output.

saveAsNameStr – Output file’s name of reset by default charset.

Throws: If happen problem when output then this method will throws IOException.

3. public void outBinFile(File out_bin_file, String saveAsNameStr, String charsetName) throws IOException

Output the file to client, and use specified charset to reset output file’s name. Only enable buffer function then can use this method. And behind this method must follow “return” syntax.

Parameters: out_bin_file – the file need to be output.

saveAsNameStr – Output file’s name of reset by default charset.

charsetName - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

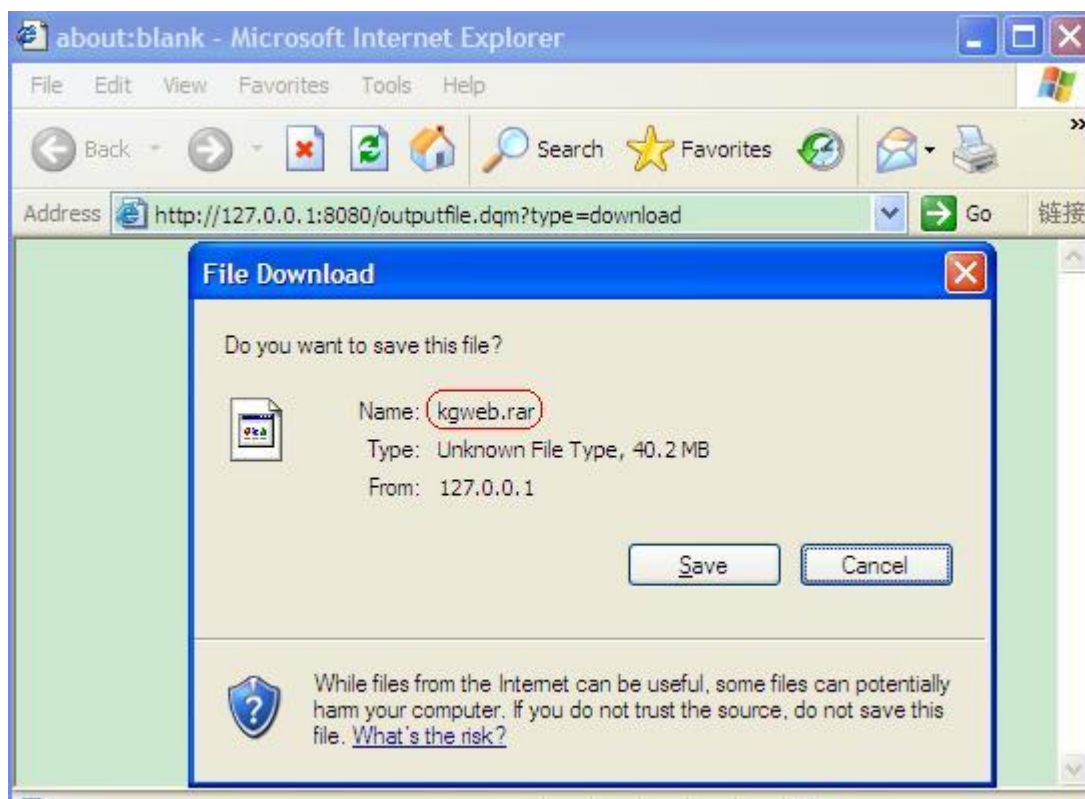
Throws: If happen problem when output then this method will throws IOException.

The principle of above No.2 method please refer to No.2 method of section 5.2. The principle of above No.3 method please refer to No.1 method of section 5.2.

So we changed “outputfile.dqm” file source code:

```
1  <% @ page buffer="true"%>
2
3  <html>
4
5  <head>
6  <meta http-equiv="Content-Language" content="zh-cn">
7  <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
8  <title>Download Page</title>
9  </head>
10
11 <body>
12
13 <%
14 String type = request.getParameter("type");
15
16 if ("download".equals(type)) { //can download file
17     File downFile = new File("c:\\kgweb.rar");
18     if (downFile.exists()) {
19         response.outBinFile(downFile, downFile.getName(), "GBK");
20         return;
21     }
22 }
23 %>
24
25 <p align="center"><b><font color="#FF0000">Sorry, can not download!</font></b></p>
26
27 </body>
28
29 </html>
```

Again visit this dqm file with parameter you will see download file's name has been changed. Please refer to picuter 5-7-3.



Picture 5-7-3

Behind “outBinFile” method must follow “return” syntax, so sometimes will happen status of unreachable statement. Example below source code, the unreachable statement at line 2.

```
1 response.outBinFile(new File("c:\\kgweb.rar"));return;
2 out.print("hello");
```

Please change code to:

```
1 f(true) {
2     response.outBinFile(new File("c:\\kgweb.rar"));return;
3 }
4 out.print("hello");
```

The compiler will check the dynamic document, if behind “outBinFile” method not follow “return” syntax then will prompt error information. But if “outBinFile” method in a bean, and use this bean to option output file, then compiler can’t check it. So please don’t forget add “return” too. For example:

```
1 <% @ page buffer="true"%>
2
3 <html>
4
5 <head>
6 <meta http-equiv="Content-Language" content="zh-cn">
7 <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
```

```

8 <title>Download Page</title>
9 </head>
10
11 <body>
12
13 <%
14 String type = request.getParameter("type");
15
16 if ("download".equals(type)) { //can download file
17     File downFile = new File("c:\\kgweb.rar");
18     if (downFile.exists()) {
19         myOutBinFile(response, downFile); return;
20     }
21 }
22 %>
23
24 <p align="center"><b><font color="#FF0000">Sorry, can not download!</font></b></p>
25
26 </body>
27
28 </html>
29 <%!
30 private void myOutBinFile(DResponseItface response, File outFile) throws IOException {
31     response.outBinFile(outFile);
32 }
33 %>

```

That “outBinFile” method writed in declaration(source code line: 30 to 33). So at source code line 19, if not add “return” then compiler also not prompt any errors. But please don’t forget add “return” too.

Chapter 6 Built-in object : request

Submits the information from client to the server is very common. For example, registration, user input information in browser, and will transmit information to server after submit.

the client via a Web browser, enter the person's name in the form and after the contents of the data can be transmitted to the server. So you can use “request” built-in object to get information of submit.

6.1 Get submit data

The “request” object has five methods can get submit data by name. The submit data can use

HTML form table to upload data, or use URL parameters to upload data.

1. public String getParameter(String key, String charsetName)

Returns a string containing the lone value of the specified parameter, or null if the parameter does not exist. The value of returns will be encoded by specified charset. You should use this method only when they are sure that there is only one value for the parameter, If the parameter has (or could have) multiple values please use “getParameterValues” method.

Parameters: key - the name of the parameter whose value is required. The “key” is case sensitive.

charsetName - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: a string containing the lone value of the specified parameter, or null if the parameter does not exist.

2. public String getParameter(String key)

Returns a string containing the lone value of the specified parameter, or null if the parameter does not exist. The value of returns will be encoded by default charset. The default charset please refer to section 2.1. You should use this method only when they are sure that there is only one value for the parameter, If the parameter has (or could have) multiple values please use “getParameterValues” method.

Parameters: key - the name of the parameter whose value is required. The “key” is case sensitive.

Returns: a string containing the lone value of the specified parameter, or null if the parameter does not exist.

3. public String[] getParameterValues(String key, String charsetName)

Returns the values of the specified parameter for the request as an array of strings, or null if the named parameter does not exist. This method would return the values of the specified query string or posted form as an array of strings. All values of returns will be encoded by specified charset.

Parameters: key - the name of the parameter whose value is required. The “key” is case sensitive.

charsetName - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: the values of the specified parameter for the request as an array of strings, or null if the named parameter does not exist.

4. public String[] getParameterValues(String key)

Returns the values of the specified parameter for the request as an array of strings, or null if the named parameter does not exist. This method would return the values of the specified query string or posted form as an array of strings. All values of returns will be encoded by default charset. The default charset please refer to section 2.1.

Parameters: key - the name of the parameter whose value is required. The “key” is case sensitive.

charsetName - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: the values of the specified parameter for the request as an array of strings, or null if the named parameter does not exist.

5. public Set<String> getParameterNameSet()

Returns: Returns the parameter names for this request as a set of strings, or an empty set if there are no parameters.

For example, “post.htm” is a static html file, you can use it to submit data to “post.dqm”.
Source code:

```
1 <form method="POST" action="post.dqm?key2=kangaroo-egg">
2   <p>key1: <input type="text" name="key1" size="20"></p>
3   <p>key1: <input type="text" name="key1" size="20"></p>
4   <p><input type="submit" value="Submit" name="B1"><input type="reset"
   value="Reset" name="B2"></p>
5 </form>
```

The “post.dqm” dqm file can get data of submit. Source code:

```
1 <% @page import="java.util.*"%>
2 <pre><%
3
4 //get all the parameter names of this request
5 Set nameSet = request.getParameterNameSet();
6 if (nameSet.size() == 0) {
7     out.println("No any parameters in current request");
8 }
9 else {
10     Iterator<String> nameItr = nameSet.iterator();
11     while (nameItr.hasNext()) {
12         String tempName = nameItr.next();
13         out.println("-----");
14         out.println("parameter name: " + tempName);
15
16         //use "getParameter" method
17         out.println("current parameter's value: " + request.getParameter(tempName));
18
19         //use "getParameterValues" method
20         String tempValues[] = request.getParameterValues(tempName);
21         out.println("Total value of current parameter: " + tempValues.length);
22         out.print("All values of current parameter: ");
23         for (int i = 0; i < tempValues.length; i++) {
24             out.print(tempValues[i] + ", ");
```

```

25     }
26     out.println("");
27 }
28 }
29
30 %></pre>

```

Visit post.htm and input “hello” and “world”, see picture 6-1-1.



The image shows a web form with a light green background. It contains two text input fields, both labeled 'key1:'. The first field contains the text 'hello' and the second field contains the text 'world'. Below the input fields are two buttons: 'Submit' and 'Reset'.

Picture 6-1-1

Result:

```

-----
parameter name: key1
current parameter's value: hello
Total value of current parameter: 2
All values of current parameter: hello, world,
-----
parameter name: B1
current parameter's value: Submit
Total value of current parameter: 1
All values of current parameter: Submit,
-----
parameter name: key2
current parameter's value: kangaroo-egg
Total value of current parameter: 1
All values of current parameter: kangaroo-egg,

```

From the results we can see, “getParameter” method only can get first value(source ode line 17), even “key1” parameter containing two values, but this method only can get it’s first value “hello”. The “getParameterValues” can get all values of the specified parameter(source code line 20).

Notice: Section2.3 and 2.5 introduced “maxPostLen” element, if submit data size over “maxPostLen” then server will prompt error information or close connection.

6.2 Temporary data

If you want to save temporary data when request a dqm file, you can use the following two methods.

1. **public void setAttribute(String name, Object value)**

This method stores a temporary data in the request context. The temporary data effective scope only in the same request.

Parameters: name - a String specifying the name of the attribute.

value - a temporary data stored with the key.

2. **public Object getAttribute(String name)**

Returns the value of the named attribute of the request temporary data, or null if the temporary data does not exist

Parameters: name - the name of the attribute whose value is required.

Returns: Temporary data by name.

Actually in the same page you can use a variable to achieve store temporary data. But if need to insert other dynamic web page, use this method can easy to transfer data. For example, a “setAttribute.txt” file, in this file we set two data “name” and “age” (source code line 2 to 3), source code:

```
1 <%  
2 request.setAttribute("name", "kangaroo-egg");  
3 request.setAttribute("age", "1");  
4 %>
```

Another dqm file “setAttribute.dqm”, it’s included “setAttribute.txt” file, and need to output data of set by “setAttribute.txt” (source code line 3 to 4). It’s source code:

```
1 <% @include file="setAttribute.txt"%>  
2 <pre>  
3 name: <%=request.getAttribute("name")%>  
4 age: <%=request.getAttribute("age")%></pre>
```

Result:

```
name: kangaroo-egg  
age: 1
```

6.3 Read client information

Sometimes the we need to get client visit information, for example client IP address, client version.

1. public String getRemoteAddr()

Returns the IP address of the agent that sent the request. Same as the CGI variable REMOTE_ADDR.

Returns: the IP address of the agent that sent the request.

2. public String getRemoteHost()

Returns the fully qualified host name of the agent that sent the request. Same as the CGI variable REMOTE_HOST.

Returns: the fully qualified host name of the agent.

3. public int getRemotePort()

Returns the port number of the agent that sent the request.

Returns: the port number of the agent that sent the request.

4. public String[] getAllHttpHead()

Gets the all header contents of this request.

Returns: the all header contents of this request as a array of strings.

5. public String getHeader(String name)

Gets the value of the requested header field of this request. The case of the header field name is ignored.

Parameters: name - the String containing the name of the requested header field.

Returns: the value of the requested header field, or null if not known.

For example, "clientInfo.dqm" dqm file, it will show client information of this request. Source code:

```
1 <pre>
2 Information of the client that sent current request
3
4 Client IP address: <%=request.getRemoteAddr()%>
5 Client host name: <%=request.getRemoteHost()%>
6 Client port number: <%=request.getRemotePort()%>
7
8 All header contents of current request:<%
9 String[] allHttpHead = request.getAllHttpHead();
10 for (int i = 0; i < allHttpHead.length; i++) {
11     out.println(allHttpHead[i]);
12 }
13 %>
14
15 The value of the "Accept-Language" field of current request:
16 <%=request.getHeader("accept-language")%>
17 </pre>
```

The results maybe not same under different environment. Our result is:

The result:

Information of the client that sent current request

Client IP address: 127.0.0.1

Client host name: 127.0.0.1

Client port number: 3032

All header contents of current request: GET /us/clientInfo.dqm HTTP/1.1

Accept: image/gif, image/x-bitmap, image/jpeg, image/pjpeg, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, application/x-shockwave-flash, */*

Referer: <http://127.0.0.1:8080/us/>

Accept-Language: en-us

Accept-Encoding: gzip, deflate

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.0.3705; InfoPath.1)

Host: 127.0.0.1:8080

Connection: Keep-Alive

The value of the "Accept-Language" field of current request: en-us

6.4 Read server information

Sometimes we need to get server response information.

1. **public String getLocalAddr()**

Returns the server IP address on which this request was received.

Returns: the server IP address on which this request was received.

2. **public String getLocalHost()**

Returns the server host name on which this request was received.

Returns: the server host name.

3. **public int getLocalPort()**

Returns the port number on which this request was received. Same as the CGI variable SERVER_PORT.

Returns: the port number on which this request was received.

For example, dqm file “serverInfo.dqm”, it will show server information of this request. Source code:

```
1 | <pre>
```

```
2 | Information of the server
3 |
4 | Server IP address: <%=request.getLocalAddr()%>
5 | Server host name: <%=request.getLocalHost()%>
6 | Client port number: <%=request.getLocalPort()%>
7 |
8 | </pre>
```

Result:

Information of the server

Server IP address: 127.0.0.1

Server host name: 127.0.0.1

Client port number: 8080

6.5 Read cookie

In section 5.6, we introduced how to save the cookie to the client, so this section will introduce how to read the cookie from client..

1. **public String getCookieValue(String cookieName)**

Returns the cookie's value by specified cookieName.

Parameters: cookieName - the specified name of the cookie.

Returns: cookie's value by specified cookieName or null if not found.

2. **public DCookie[] getCookies()**

Gets the array of cookies found in this request.

Returns: the array of cookies found in this request. If not found any cookie then return a empty array (array length is 0)

And how to use it we will introduce at chapter 7.

6.6 Read dqm file path information

Sometimes we need know dqm file absolute path, url path, so this section we will introduce how to get these information.

1. **public String getNowFolder()**

Returns the absolute file pathname string of current dqm file.

Returns: The absolute pathname string of current dqm file, here not return null unless the system error.

2. **public String getNowPath()**

Returns the absolute pathname string of current dqm file.

Returns: The string form of this absolute pathname.

3. **public File getNowFile()**

Returns the absolute form of this abstract file pathname.

Equivalent to `new File(this.getNowPath())`.

Returns: the absolute form of this abstract file pathname

4. **public String getWebPath()**

Get current host home directory. Its value is equivalent “webPath” item, about “webPath” refer to section 2.3 and 2.5.

Returns: the absolute pathname string of current host home directory.

5. **public String getNowUrlFolder()**

Gets, from the first line of the HTTP request, the part of this request's URI that is to the left of any query string.

Returns: this request's URI

6. **public String getNowUrlQuery()**

Gets any query string that is part of the HTTP request URI. Same as the CGI variable QUERY_STRING. Please notice the query string will be decoded (refer to “urlEnc” item in section 2.1).

Returns: query string that is part of this request's URI, or null if it contains no query string

For example, “getDqmFileInfo.dqm” dqm file can get its own information. We put this dqm file under the “test_folder” folder. Source code:

```
1  <% @page import="java.util.Date"%>
2
3  <pre>
4  Current host home directory: <%=request.getWebPath()%>
5
6  The absolute file pathname string of current dqm file: <%=request.getNowFolder()%>
7  The absolute pathname string of current dqm file: <%=request.getNowPath()%>
8  Current request's URI: <%=request.getNowUrlFolder()%>
9  Query string that is part of this request's URI: <%=request.getNowUrlQuery()%>
10
11 Use "request.getNowFile()" method can get more information-----
12 <%File file = request.getNowFile();%>
13 Current dqm file name: <%=file.getName()%>
```

```

14 Current dqm file size: <%=file.length()%>
15 Current dqm file last modified: <%=new Date(file.lastModified())%>
16 </pre>

```

The results maybe not same under different environment. Our result is:

```

1 Current host home directory: C:\kgserver\webapp
2
3 The absolute file pathname string of current dqm file: C:\kgserver\webapp\us\test_folder
4 The absolute pathname string of current dqm file:
  C:\kgserver\webapp\us\test_folder\getDqmFileInfo.dqm
5 Current request's URI: /us/test_folder/
6 Query string that is part of this request's URI: null
7
8 Use "request.getNowFile()" method can get more information-----
9
10 Current dqm file name: getDqmFileInfo.dqm
11 Current dqm file size: 686
12 Current dqm file last modified: Tue Sep 11 16:29:50 GMT+08:00 2007

```

We set current host home directory to “C:\kgserver\webapp”, so dqm absolute pathname was included home directory (result line 3 and 4). The request not include any query string so return null (result line 6). Use “getNowFile” method we can get more dqm file information, example file size (result line 8 to 12).

7. public boolean isSSL()

Get the type of current connection. Because the server can simultaneously enable http and https service (refer to section 2.2), so user can visit resources through http or https service.

Returns: If user visit by https service then return true, else visit by http service return false.

This method is also very useful, for example there is a “login.dqm” dqm file under “hr_system” folder, it’s a simulation login page of the HR system, source code:

```

1 <p align="center"><font color="#0000FF" size="7"><b>Welcome to the HR
  system</b></font></p>
2 <form method="POST" action="">
3 <p align="center">Username: <input type="text" name="T1" size="20"></p>
4 <p align="center">Password: <input type="text" name="T2" size="20"></p>
5 <p align="center"><input type="submit" value="Submit" name="B1"><input type="reset"
  value="Reset" name="B2"></p>
6 </form>
7
8 <%
9 //check the type of current connection (http or https)
10 boolean isSSL = request.isSSL();
11 String bgColor, link, linkInfo;

```

```

12 if (isSSL) { //If user visit by https service
13     //change the background color to pale yellow
14     bgColor = "#FFFF99";
15     //show http link url
16     link      =      "http://localhost:8080"      +      request.getNowUrlFolder()      +
request.getNowFile().getName();
17     //prompt user can use http service
18     linkInfo = "Normal connection";
19 }
20 else { //If user visit by http service
21     //change the background color to gray
22     bgColor = "#C0C0C0";
23     //show https link url
24     link      =      "https://localhost:8443"      +      request.getNowUrlFolder()      +
request.getNowFile().getName();
25     //prompt user can use https service
26     linkInfo = "Secure connection";
27 }
28
29 %>
30
31 <table border="0" width="100%" id="table1" bgcolor="<%=bgColor%>">
32     <tr>
33         <td>
34             <p align="right"><a href="<%=link%>"><%=linkInfo%></a></td>
35         </tr>
36     </table>

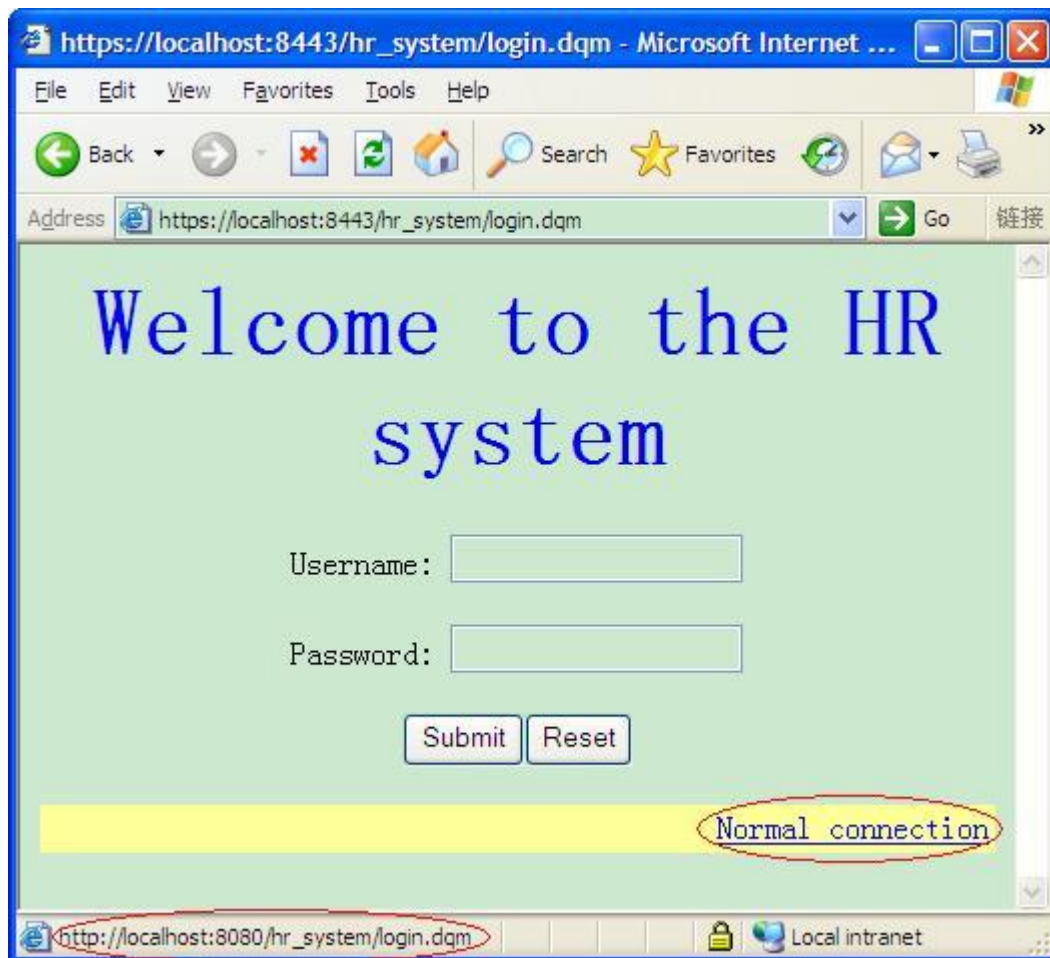
```

When user visit this dqm file by HTTP service, then page bottom will prompt can use secure connection to visit it. Please notice that bottom form's background color is gray, it prompt current visit by HTTP service, and provide HTTPS link url. See picture 6-6-1.



Picture 6-6-1

When user visit this dqm file by HTTPS service, then page bottom will prompt can use normal connection to visit it. Please notice that bottom form's background color is pale yellow, it prompt current visit by HTTPS service, and provide HTTP link url. See picture 6-6-2.



Picture 6-6-2

Source code line 12 will check the type of current connection. Source code line 16 and 24 is create HTTP and HTTPS url.

Notice: Only simultaneously enable HTTP and HTTPS service then can successfully execute this program. Please refer to section 2.2 to enable HTTP and HTTPS service. And HTTP service port must set be “8080”, HTTPS service port must set be “8443”.

6.7 GET multipart data

In section 6.1 we introduced how to get upload data, but it only can get TEXT data. So this section we will introduce multipart, use multipart we can upload binary file to server, for example we can upload picture, music files to server.

1. `public DMpFormData.FormDataEntry getFormDataEntry(String key)`

Returns a lone FormDataEntry(multipart object) by the specified parameter, or null if the parameter does not exist. You should use this method only when they are sure that there is

only one `FormDataEntry` for the parameter, If the parameter has (or could have) multiple values(multipart object) please use “`getFormDataEntryValues`” method.

Parameters: key - the name of the parameter whose `FormDataEntry` is required. The “key” is case sensitive.

Returns: a lone `FormDataEntry` of the specified parameter, or null if the parameter does not exist. About `FormDataEntry` class we will introduce later.

2. `public DMpFormData.FormDataEntry[] getFormDataEntryValues(String key)`

Returns the multipart objects of the specified parameter for the request as an array of `FormDataEntry` class, or null if the named parameter does not exist. This method would return the multipart objects of the specified posted form as an array of `FormDataEntry` class.

Parameters: key - the name of the parameter whose `FormDataEntry` is required. The “key” is case sensitive.

Returns: the `FormDataEntry` objects of the specified parameter for the request as an array of `FormDataEntry` class, or null if the named parameter does not exist.

3. `public Set<String> getFormDataNameSet()`

Returns: the multipart parameter names for this request as a set of strings, or null if there are no multipart parameters.

4. `public int getContentLength()`

Returns: the size of the request entity data. Same as the CGI variable `CONTENT_LENGTH`. If use multipart form to upload data then return the size of multipart post content. If use ordinary form to upload data then return this size of ordinary post content. The size of URL query string not be include.

Next we will talk about the “`FormDataEntry`” class, it includes multipart data and information. The “`FormDataEntry`” is inter class of “`com.kangaroo_egg.dqm.DMpFormData`”, now we introducing methods of this class.

1. `public String getValue(String charsetName) throws`

`UnsupportedEncodingException`

Returns a string containing the value of the current multipart object. The value of returns will be encoded by specified charset. This method can be applied to get multipart text data, if current multipart object’s value is binary data then this method also encoded binary data to text format.

Parameters: charsetName - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: a string containing the value of the current multipart object.

Throws: `UnsupportedEncodingException` - if charsetName is not supported

2. `public String getValue() throws UnsupportedEncodingException`

Returns a string containing the value of the current multipart object. The value of returns will be encoded by default charset, the default charset please refer to section 2.1. This method can

be applied to get multipart text data, if current multipart object's value is binary data then this method also encoded binary data to text format.

Returns: a string containing the value of the current multipart object.

Throws: `UnsupportedEncodingException` - if default charset is not supported

3. `public int getContentSize()`

Returns: the size of current multipart object's data. This value is the users upload data size.

4. `public boolean isFile()`

Judgment of the current multipart object's type is file or not, because the multipart form can upload TEXT data too.

Returns: If multipart object's data is file of user uploaded then return true, else return false.

5. `public String getAbsolutePath(String charsetName) throws`

`UnsupportedEncodingException`

If current multipart object's type is file then returns it's absolute pathname. The value of returns will be encoded by specified charset.

Parameters: `charsetName` - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: If current multipart object's type is file then returns it's absolute pathname. If current multipart object's type is not file then returns null. If current multipart object's type is file but user not upload any file then returns empty string, user can use "isFile()" method to judge current multipart object's type first.

Throws: `UnsupportedEncodingException` - if `charsetName` is not supported.

6. `public String getAbsolutePath() throws UnsupportedEncodingException`

If current multipart object's type is file then returns it's absolute pathname. The value of returns will be encoded by default charset, the default charset please refer to section 2.1.

Returns: If current multipart object's type is file then returns it's absolute pathname. If current multipart object's type is not file then returns null. If current multipart object's type is file but user not upload any file then returns empty string, user can use "isFile()" method to judge current multipart object's type first.

Throws: `UnsupportedEncodingException` - if default charset is not supported

7. `public String getFileName(String charsetName) throws`

`UnsupportedEncodingException`

If current multipart object's type is file then returns filename of uploaded. The value of returns will be encoded by specified charset.

Parameters: `charsetName` - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: If current multipart object's type is file then returns it's filename. If current multipart object's type is not file then returns null. If current multipart object's type is file but user not upload any file then returns empty string, user can use "isFile()" method to judge

current multipart object's type first.

Throws: `UnsupportedEncodingException` - if charsetName is not supported.

8. public String getFileName() throws UnsupportedEncodingException

If current multipart object's type is file then returns filename of uploaded. The value of returns will be encoded by default charset, the default charset please refer to section 2.1.

Returns: If current multipart object's type is file then returns it's filename. If current multipart object's type is not file then returns null. If current multipart object's type is file but user not upload any file then returns empty string, user can use "isFile()" method to judge current multipart object's type first.

Throws: `UnsupportedEncodingException` - if default charset is not supported

9. public String getFileExtName(String charsetName) throws

UnsupportedEncodingException

If current multipart object's type is file then returns Ext filename of uploaded. The value of returns will be encoded by specified charset.

Parameters: charsetName - specified charset name, the name of a supported charset, for example BIG-5, UTF-8, GBK...

Returns: If current multipart object's type is file then returns it's Ext filename. If current multipart object's type is not file then returns null. If current multipart object's type is file but user not upload any file then returns empty string, user can use "isFile()" method to judge current multipart object's type first.

Throws: `UnsupportedEncodingException` - if charsetName is not supported.

10. public String getFileExtName() throws UnsupportedEncodingException

If current multipart object's type is file then returns Ext filename of uploaded. The value of returns will be encoded by default charset, the default charset please refer to section 2.1.

Returns: If current multipart object's type is file then returns it's Ext filename. If current multipart object's type is not file then returns null. If current multipart object's type is file but user not upload any file then returns empty string, user can use "isFile()" method to judge current multipart object's type first.

Throws: `UnsupportedEncodingException` - if default charset is not supported.

11. public String getName()

Get parameter name of current multipart object. The returns value is equal to "request.getFormDataEntry(String key)" method's key parameter.

Returns: parameter name of current multipart object.

12. public String getContentType()

Get current multipart object's MIME type, This type determined according to the upload file's type.

Returns: current multipart object's MIME type. For example upload a picture file then returns MIME type is "image/jpeg", upload a text file then returns MIME type is "text/plain". If current multipart object's type is not file then returns null. If current

multipart object's type is file but user not upload any file then returns "application/octet-stream", user can use "isFile()" method to judge current multipart object's type first.

13. public String getContentDisposition()

Get current multipart object's disposition type.

Returns: current multipart object's disposition type. Usually it will returns "form-data".

14. public void saveToFile(String fileName, boolean append) throws IOException

Writing current multipart object's data to a File.

Parameters: fileName - the system-dependent file name. It can use relative path or absolute path. Relative path is relative to the current dqm file's path. For example: "../xxx.txt", "test/xxx.txt". Absolute path's example is "c:\\xxx.txt", "/xxx.txt".
append - if true, then bytes will be written to the end of the file rather than the beginning

Throws: IOException - if an I/O error occurs.

15. public boolean saveToFile(File file, boolean overwrite, boolean append)

Writing current multipart object's data to specified File.

Parameters: file - the file to be opened for writing
overwrite - If the target file exist, whether to allow it to overwrite
append - if true, then bytes will be written to the end of the file rather than the beginning

Returns: Is save the file successful.

If the target file existence:

If "overwrite" value was "false", this meaning is not allow to revise the file which already existed, so like this status "append" value will be ignored and current method will not to do anything, returns "false" to express the save file defeat.

If "overwrite" value was "true", this meaning is allow to revise the file which already existed, so need to judge based on the "append" value. If append or overwrite file successfully then returns "true", else returns "false".

If the target file does not exist:

This status the "append" value will be ignored, so need to judge based on the "append" value. If append or overwrite file successfully then returns "true", else returns "false".

Notice: The "saveToFile" method can use relative path, the relative path is relative to the kangaroo-egg installation directory.

For example: <%saveToFile(new File("save.txt"), true, false);%>

Then "save.txt" file will save in the installation directory.

If you need to save file in directory of current dqm file, then you can use the method of section 6.6, example:

<%saveToFile(new File(request.getNowFolder + "/save.txt"), true, false);%>

16. public int read()

Reads a byte of data from current multipart object's data. You can use current method in conjunction with "OutputStream. write(int b)" method.

Returns: the next byte of current multipart object's data, or -1 if the end of current multipart object's data is reached.

17. public void resetRead()

Reset "read()" method. After execute this method, "read()" method will from current multipart object data's first byte to read.

Next we will show a example. First we create a static html file "upfile.htm", it can upload data to dqm file. Source code:

```
1 <form method="POST" enctype="multipart/form-data" action="upfile.dqm">
2 <p>Self introduction:</p>
3 <p><textarea rows="7" name="intro" cols="31"></textarea></p>
4 <p>
5 Gender: <input type="radio" value="male" checked name="gender">male&nbsp;<input
   type="radio" name="gender" value="female">female
6 </p>
7 <p>
8 Hobbies: <input type="checkbox" name="favorite" value="football">football&nbsp;<
9 <input type="checkbox" name="favorite" value="cartoon">cartoon&nbsp;<
10 <input type="checkbox" name="favorite" value="Internet">Internet</p>
11
12 <input type="FILE" name="upfile" size="50"><br>
13 <p><input type="submit" value="Submit" name="button"><input type="reset"
   value="Reset" name="B2"></p>
14 </form>
```

Please notice string of **enctype="multipart/form-data"** at source code line 1, it meaning is upload data use multipart. And string of **action="upfile.dqm"** at source code line 1, so the "upfile.dqm" will process data of uploaded, it's source code:

```
1 <% @page import="java.util.*"%>
2
3 <pre>
4
5 Get the size of current request entity data: <%=request.getContentLength()%>
6
7 <%
8 //read all of the multipart parameter names for this request
9 Set nameSet = request.getFormDataSet();
10 %>
11
12 How many multipart of current request: <%if (nameSet != null) out.print(nameSet.size());
   else out.print("No multipart");%>
```

```

13
14 <%
15 if (nameSet != null) {
16     Iterator<String> nameItor = nameSet.iterator();
17     while (nameItor.hasNext()) {
18         String tempName = nameItor.next();
19         out.println("-----");
20
21         //read formDataEntry by the specified parameter
22         DMpFormData.FormDataEntry tempFDE = request.getFormDataEntry(tempName);
23
24         out.println("The parameter name of current multipart object: " +
tempFDE.getName()); //it's value is equal to current tempName's value
25
26         out.println("The MIME type of current multipart object: " +
tempFDE.getContentType());
27
28         out.println("The disposition type current multipart object: " +
tempFDE.getContentDisposition());
29
30         //output the size of current multipart object's data
31         out.println("The size of current multipart object's data: " +
tempFDE.getContentSize());
32
33         if (!tempFDE.isFile()) { //if current multipart object's type is not file, then output it's
content
34             out.println("Current multipart object's content: " + tempFDE.getValue());
35         }
36         else { //if current multipart object's type is file
37             out.println("Current multipart object's type is file.");
38             out.println("The absolute pathname of uploaded file: " +
tempFDE.getAbsolutePath());
39             out.println("The filename of uploaded file: " + tempFDE.getFileName());
40             out.println("The ext filename of uploaded file: " + tempFDE.getFileExtName());
41             out.println("Save the uploaded File.");
42             if (tempFDE.getContentSize() > 0) { //If has uploaded file
43                 tempFDE.saveToFile("uploaded_file.txt", false);
44             }
45         }
46
47     }
48 }
49 %>
50 </pre>

```

We start to upload the data, the operation like picture 6-7-1.

Self introduction:

I'm kangaroo-egg.

Gender: ☒ male ☐ female

Hobbies: ☐ football ☒ cartoon ☒ Internet

C:\myself.txt

Picture 6-7-1

The result is:

```
1  Get the size of current request entity data: 721
2
3
4
5  How many multipart of current request: 5
6
7  -----
8  The parameter name of current multipart object: upfile
9  The MIME type of current multipart object: text/plain
10 The disposition type current multipart object: form-data
11 The size of current multipart object's data: 8
12 Current multipart object's type is file.
13 The absolute pathname of uploaded file: C:\myself.txt
14 The filename of uploaded file: myself.txt
15 The ext filename of uploaded file: .txt
16 Save the uploaded File.
17 -----
18 The parameter name of current multipart object: gender
19 The MIME type of current multipart object: null
20 The disposition type current multipart object: form-data
21 The size of current multipart object's data: 4
22 Current multipart object's content: male
```



```

23 -----
24 The parameter name of current multipart object: intro
25 The MIME type of current multipart object: null
26 The disposition type current multipart object: form-data
27 The size of current multipart object's data: 17
28 Current multipart object's content: I'm kangaroo-egg.
29 -----
30 The parameter name of current multipart object: favorite
31 The MIME type of current multipart object: null
32 The disposition type current multipart object: form-data
33 The size of current multipart object's data: 7
34 Current multipart object's content: cartoon
35 -----
36 The parameter name of current multipart object: button
37 The MIME type of current multipart object: null
38 The disposition type current multipart object: form-data
39 The size of current multipart object's data: 6
40 Current multipart object's content: Submit

```

We have to analyze the results of the above output, this time we upload five multipart objects. We can see the “upfile” multipart object is file type, and if user upload the file then will show it’s filename, pathname (Source code line 38 to 40). The size of “upfile” multipart object is equal to size of uploaded file. Last save the uploaded file at current folder, file name is “uploaded_file.txt” (source code line 42 to 44).

We look at an interesting phenomenon, the result 1st line displayed that current upload the byte count is 721. But five multipart objects data size plus together is 42 bytes. (result line 11, 21, 27, 33 and 39). Why did they not equal? Because request entity data includes the description information, but kangaroo-egg server was processed(deleted) the description information.

Another interesting phenomenon, seen picture 6-7-1, in “hobbies” item we chose “carton” and “internet” two options. But results only show "carton", because we used “getFormDataEntry” method (source code line 22). Ok now we create two new files “upfile1.htm” and “upfile1.dqm”, it’s use “getFormDataEntryValues” method to read multipart object, upfile1.htm soure code:

```

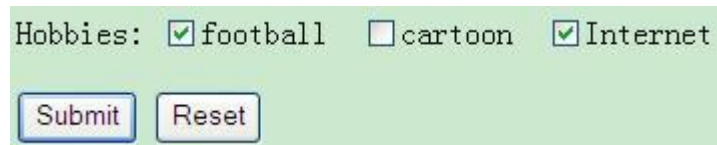
1  <form method="POST" enctype="multipart/form-data" action="upfile1.dqm">
2  <p>
3  Hobbies: <input type="checkbox" name="favorite" value="football">football&nbsp;
4  <input type="checkbox" name="favorite" value="cartoon">cartoon&nbsp;
5  <input type="checkbox" name="favorite" value="Internet">Internet
6  </p>
7  <p>
8  <input type="submit" value="Submit" name="button">
9  <input type="reset" value="Reset" name="B2">
10 </p>
11 </form>

```

Please notice string of **enctype="multipart/form-data"** at “upfile1.htm” source code line 1, it meaning is upload data use multipart. And string of **action="upfile1.dqm"** at “upfile1.htm” source code line 1, so the “upfile1.dqm” will process data of uploaded, it’s source code:

```
1 <% @page import="java.util.*"%>
2
3 <pre>
4
5 <%
6 DMpFormData.FormDataEntry[]          tempFDE          =
  request.getFormDataEntryValues("favorite");
7 if (tempFDE != null) {
8     out.println("Item of hobbies:");
9     for (int i = 0; i < tempFDE.length; i++) {
10         out.println("Content: " + tempFDE[i].getValue());
11     }
12 }
13 %>
14
15 </pre>
```

This time we selected “football” and “Internet”, seen picture 6-7-2.



Picture 6-7-2

The result:

```
Item of hobbies:
Content: football
Content: Internet
```

You can seen from the result that “football” and “Internet” all displayed (upfile1.dqm source code line 6).

Notice: Section2.3 and 2.5 introduced “maxPostLen” element, if submit data size over “maxPostLen” then server will prompt error information or close connection.

Chapter 7 Use cookie

This chapter we will introduce how to use cookie. Cookies are used to get user agents (web browsers etc) to hold small amounts of state associated with a user's web browsing. Common applications for cookies include storing user preferences, automating low security user signon facilities, and helping collect data used for "shopping cart" style applications.

7.1 DCookie object

If we need to read and write cookies then must create DCookie object first. DCookie class represents a "Cookie", as used for session management with HTTP and HTTPS protocols.

Cookies are named, and have a single value. They may have optional attributes, including a comment presented to the user, path and domain qualifiers for which hosts see the cookie, a maximum age, and a version. Current web browsers often have bugs in how they treat those attributes, so interoperability can be improved by not relying on them heavily.

Cookies are assigned by servers, using fields added to HTTP response headers. In this API, cookies are saved one at a time into such HTTP response headers, using the response.addCookie method (refer to section 5.5). User agents are expected to support twenty cookies per host, of at least four kilobytes each; use of large numbers of cookies is discouraged.

Cookies are passed back to those servers using fields added to HTTP request headers. In this API, HTTP request fields are retrieved using the cookie module's request.getCookies method (refer to section 6.5). This returns all of the cookies found in the request. Several cookies with the same name can be returned; they have different path attributes, but those attributes will not be visible when using "old format" cookies.

Constructor: **public DCookie(String iname, String ivalue)**

Defines a cookie with an initial name/value pair. Names must not contain whitespace, comma, or semicolons and should only contain ASCII alphanumeric characters.

Names starting with a "\$" character are reserved by RFC 2109. 创

Parameters: iname - name of the cookie

ivalue - value of the cookie

Methods:

1. **public String getName()**

Returns: the name of the cookie. This name may not be changed after the cookie is created.

2. **public String getValue()**

Returns: the value of the cookie.

3. public void setValue(String newValue)

Sets the value of the cookie. BASE64 encoding is suggested for use with binary values.

Parameters: newValue – the value of need to reset

4. public int getVersion()

Returns the version of the cookie. Version 1 complies with RFC 2109, version 0 indicates the original version, as specified by Netscape. Newly constructed cookies use version 0 by default, to maximize interoperability. Cookies provided by a user agent will identify the cookie version used by the browser.

5. public void setVersion(int v)

Sets the version of the cookie protocol used when this cookie saves itself. Since the IETF standards are still being finalized, consider version 1 as experimental; do not use it (yet) on production sites. Default set 0.

6. public void setComment(String purpose)

If a user agent (web browser) presents this cookie to a user, the cookie's purpose will be described using this comment. This is not supported by version zero cookies.

7. public String getComment()

Returns: the comment describing the purpose of this cookie, or null if no such comment has been defined.

8. public void setDomain(String pattern)

This cookie should be presented only to hosts satisfying this domain name pattern. Read RFC 2109 for specific details of the syntax. Briefly, a domain name begins with a dot (".foo.com") and means that hosts in that DNS zone ("www.foo.com", but not "a.b.foo.com") should see the cookie. By default, cookies are only returned to the host which saved them.

9. public String getDomain()

Returns: the domain of this cookie

10. public void setMaxAge(int expiry)

Sets the maximum age of the cookie. The cookie will expire after that many seconds have passed. Negative values indicate the default behaviour: the cookie is not stored persistently, and will be deleted when the user agent (web browser) exits. A zero value causes the cookie to be deleted.

11. public int getMaxAge()

Returns: the maximum specified age of the cookie. If none was specified, a negative value is returned, indicating the default behaviour described with setMaxAge.

12. **public void setPath(String uri)**

This cookie should be presented only with requests beginning with this URL. Read RFC 2109 for a specification of the default behaviour. Basically, URLs in the same "directory" as the one which set the cookie, and in subdirectories, can all see the cookie unless a different path is set.

13. **public String getPath()**

Returns: the prefix of all URLs for which this cookie is targetted

14. **public void setSecure(boolean flag)**

Indicates to the user agent that the cookie should only be sent using a secure protocol (https). This should only be set when the cookie's originating server used a secure protocol to set the cookie's value.

15. **public boolean getSecure()**

Returns: the value of the 'secure' flag

Notice: This implementation is not synchronized. If multiple threads access this DCookie concurrently, and at least one of the threads modifies the DCookie's value (for example use "setValue", "setMaxAge" method), it must be synchronized externally.

7.2 Example of use DCookie

For example, "writecookie.dqm" file can write cookie to client (web browsers etc), it's source code:

```
1  <% @ page buffer="true"%>
2  <%
3  DCookie ck=new DCookie("username", "dunne");
4  ck.setMaxAge(30*60); // Set the maximum age of the cookie to 30 minutes
5  response.addCookie(ck); // write cookie to client
6  out.print("Write cookie ok.");
7  %>
```

From up source code 5st line you can see need use addCookie method to write cookie (refer to section 5.6). We enable buffer function in source code line 1, because write cookie need to enable buffer.

Next we introduce how to read cookie, "readcookie.dqm" can read all cookies from client, and it will display those cookie's name. Source code:

```
1  <%
2  DCookie cookies[]=request.getCookies(); // read all cookies
3
```

```

4  if(cookies.length == 0) { // if not found any cookie
5      out.print("no any cookie");
6  }
7  else
8  {
9      out.print(cookies.length + "<br>");
10     for(int i = 0; i < cookies.length; i++) // loop read all cookies
11     {
12         out.println(cookies[i].getName() + "->" + cookies[i].getValue() + "<br>");
13     }
14 }
15 %>

```

First we execute “writecookie.dqm”, after execute “readcookie.dqm”, then result:

```

2
DQMSESSIONID->fe7a9b1a-0b24-4a47-8bae-9e377ef16141
username->dunne

```

From above results we can see that found two cookies, but we only set a cookie so why can get two cookies? Actually another cookie was automatically created by session, about session please refer to chapter 8, so please don't use “DQMSESSIONID” s cookie's name, it was reserved by server.

If you know cookie's name, then can directly read cookie, “directreadcookie.dqm” file demonstrated how to directly read cookie, source code:

```

1  <%
2  String value =request.getCookieValue("username");
3
4  out.println("cookie's value: " + value);
5  %>

```

First we execute “writecookie.dqm”, after execute “directreadcookie.dqm”, then result:

```

cookie's value: dunne

```

Chapter 8 Built-in object : session

Session management is a mechanism to maintain state about a series of requests from the same user across some period of time. That is, the term "session" refers to the time that a user is at a particular web site. The problem is, that HTTP has no mechanism to maintain state. Individual requests aren't related to each other. The Web server can't easily distinguish between single users and doesn't know about user sessions. Session management refers to the way that associate data with a user during a visit to a Web page. For example, a typical online shopping program's session might include logging in, putting an item into the shopping cart, going to the checkout page, entering address and credit card data, submitting the order, and closing the browser window. DQM includes native session management functions to ease the task of managing user sessions.

8.1 About session object

The session object is implemented by services to provide an association between an HTTP client and HTTP server. This association, or session, persists over multiple connections and/or requests during a given time period. Sessions are used to maintain state and user identity across multiple page requests. A session can be maintained either by using cookies or by URL rewriting.

Notice: Because the session will occupy resources, so if not need to use it you can close session function. The session can't be used when closed it. Open or closed session directives please see section 3.2.

8.2 Write and read data into session

1. `public void setAttribute(String name, Object value)`

Binds the specified object into the session's application layer data with the given name. Any existing binding with the same name is replaced. If

Parameters: name - the name to which the data object will be bound.

value - the data object to be bound, if it's null then equal to "removeAttribute(name)" method.

Throws: If value of parameter "name" is null then throws `IllegalArgumentException`.

2. `public Object getAttribute(String name)`

Returns the object bound to the given name in the session's application layer data. Returns

null if there is no such binding.

Parameters: name - the name of the binding to find.

Returns: the value bound to that name, or null if the binding does not exist.

3. public void removeAttribute(String name)

Removes the object bound to the given name in the session's application layer data. Does nothing if there is no object bound to the given name.

Parameters: name - the name of the object to remove

For example: a dqm file “sessionName.dqm”, it’s source code:

```
1 <html>
2
3 <head>
4 <meta http-equiv="Content-Language" content="zh-cn">
5 <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
6 <title>The names of visitor</title>
7 </head>
8
9 <body>
10
11 <form method="POST" action="addSessionName.dqm">
12     <p>Visitor's name: <input type="text" name="sessionName" size="20"></p>
13     <p><input type="submit" value="Submit" name="B1"><input type="reset"
14     value="Reset" name="B2">
15     <a href="delSessionName.dqm">Delete name</a></p>
16 </form>
17 <%String name = (String)session.getAttribute("name");
18 if (name == null) {
19     name = "Guest";
20 }
21 %>
22 <p>Welcome: <%=name%></p>
23 </body>
24
25 </html>
```

At above source code line 16, it will read name of visitor from current session. If has not read to the visitor’s name then set name to “Guest” (source code line 17 to 19). Last we display visitor’s name (source code line 21).

The dqm file “addSessionName.dqm” can add a name into current session, it’s source code:

```
1 <%
2 String name = request.getParameter("sessionName");
```



```

3  if (name != null) {
4      session.setAttribute("name", name);
5      out.print("Add the names of visitor ok!");
6  }
7  else {
8      out.print("Please input names of visitor!");
9  }
10 %>

```

The dqm file “delSessionName.dqm” can delete name from current session (if exist visitor’s name), source code:

```

1  <%
2  session.removeAttribute("name");
3  %>
4
5  Visitor's name has been deleted!

```

First execute “sessionName.dqm”, you will see “Welcome: Guest” (picture 8-2-1), because this time no name in the session.



Picture 8-2-1

Next we input “dunne” into “Visitor’s name” field and submit it, after execute “sessionName.dqm” again then you can see “Welcome: dunne” (if not see it please refresh). If at this time using another computer to visit “sessionName.dqm” then will still see that "You are Welcome: Guest". Because different customer with different session.

If click link of “Delete name” in “sessionName.dqm”, then visit “sessionName.dqm” again, you will see name has been deleted, it will be like the first show that "Welcome: Guest".

8.3 Session object's information

The session object itself also has many information, for instance the session survival time, sessionID...

1. public long getCreationTime()

Returns the time at which this session representation was created, in milliseconds since midnight, January 1, 1970 UTC.

Returns: the time when the session was created

2. public long getLastAccessedTime()

Returns the last time the client sent a request carrying the identifier assigned to the session. Time is expressed as milliseconds since midnight, January 1, 1970 UTC. Application level operations, such as getting or setting a value associated with the session, does not affect the access time.

Returns: the last time the client sent a request carrying the identifier assigned to the session

3. public void setMaxInactiveInterval(int interval)

Sets the maximum interval between requests that this session will be kept by the host server.

Parameters: the length of max inactive interval in seconds

4. public int getMaxInactiveInterval()

Returns: Current session's maximum inactive interval in seconds

5. public boolean isNew()

A session is considered to be "new" if it has been created by the server, but the client has not yet acknowledged joining the session.

Returns: true if the session has been created by the server but the client has not yet acknowledged joining the session; false otherwise

6. public String getId()

Returns the identifier assigned to this session. An HttpSession's identifier is a unique string that is created and maintained by DQM Context.

Returns: the identifier assigned to this session

Example, the dqm file "sessionInfo.dqm" using these methods, it's source code:

```
1 <pre>
2 Current session creation time: <%=new java.util.Date(session.getCreationTime())%>
3 Current session LastAccessedTime: <%=new
  java.util.Date(session.getLastAccessedTime())%>
4 Current session maximum interval: <%=session.getMaxInactiveInterval()%> second(s)
5 Set current session maximum interval to 5 minutes<%=session.setMaxInactiveInterval(60 *
  5);%>
6 Is a new session: <%=session.isNew()%>
7 Current session identifier: <%=session.getId()%>
8 </pre>
```

When first time visits this dqm file the result is approximately as follows:

```
1 Current session creation time: Fri Oct 19 13:46:17 GMT+08:00 2007
2 Current session LastAccessedTime: Fri Oct 19 13:46:17 GMT+08:00 2007
3 Current session maximum interval: 1200 second(s)
4 Set current session maximum interval to 5 minutes
```

```
5 | Is a new session: true
6 | Current session identifier: 83d89e2e-8d04-455a-b36f-e77358a9bc00
```

From the above results can be seen that a new session created when first visit (result line 5). The maximum interval of session is 1200 seconds (this value was set by default, result line 3). Session creation time is equal to last accessed time, because it's a new session (result line 1 and 2).

When visit it again, the result:

```
1 | Current session creation time: Fri Oct 19 13:46:17 GMT+08:00 2007
2 | Current session LastAccessedTime: Fri Oct 19 13:51:09 GMT+08:00 2007
3 | Current session maximum interval: 300 second(s)
4 | Set current session maximum interval to 5 minutes
5 | Is a new session: false
6 | Current session identifier: 83d89e2e-8d04-455a-b36f-e77358a9bc00
```

You can see this time session was not new created (second result line 5), and maximum interval of session become 300 seconds (refer to source code 5). The same reasoning that session creation time is not equal to last accessed time (second result line 1 and 2).

8.4 How to release session

We known the session will be expired, but system will not release the memory of expired session immediately, so system will release memory of expired session periodically (please refer to section 2.1). Of course, you can set max inactive interval to negative value, so session will expire immediate but this does not mean that system will release memory of expired session immediately. Ok if you use the method of this section then can release memory of expired session immediately.

1. **public void invalidate()**

Causes this representation of the session to be invalidated and removed from its context

For example, the dqm file “removeSession1.dqm” can add a name into current session. Source code:

```
<%session.setAttribute("name", "Shemin Dunne");%>
Add session ok.
```

The dqm file “removeSession2.dqm” can get name from current session, source code:

```
Name: <%=session.getAttribute("name")%>
```

The dqm file “removeSession3.dqm” used “invalidate” method to clear session, source code:

```
<%session.invalidate();%>
```

Clear session use "invalidate" method.

The dqm file "removeSession4.dqm" used "setMaxInactiveInterval" method to clear session, source code:

```
<%session.setMaxInactiveInterval(-1);%>
```

Clear session use "setMaxInactiveInterval" method.

First we visit "removeSession1.dqm", next visit "removeSession2.dqm" then you can see name's value is "Shemin Dunne". Next if we visit "removeSession3.dqm" or "removeSession4.dqm" then you will find that two dqm file all can clear session, but there is still a difference, use "invalidate" can clear current session and release it's memory immediately, use "setMaxInactiveInterval" can clear current session too but system will not release it's memory immediately.

Chapter 9 Built-in object : application

Application Object is used to share the data with all application pages. Thus, all users share information of a given application using the Application object. The session like private storage bin in the classroom, only the owner can use it. The application like public storage bin in the classroom, all the classmates can use it. So each host (virtual host) has its own application, but A virtual host can't visit B virtual host's application.

9.1 Write and read data into application

The application does not look like the session, it is always exist, do not expire, from server's start until server's stop, but if the server restart, then the application of information would be lost.

1. **public void setAttribute(String name, Object value)**

Store the object with the given object name in the application. Any existing binding with the same name is replaced.

Parameters: name - the name to which the data object will be bound.

value - the data object to be bound, if it's null then equal to "removeAttribute(name)" method.

Throws: If value of parameter "name" is null then throws IllegalArgumentException.

2. **public Object getAttribute(String name)**

Returns the object bound to the given name in the application's application layer data. Returns null if there is no such binding.

Parameters: name - the name of the binding to find.

Returns: the value bound to that name, or null if the binding does not exist.

3. **public void removeAttribute(String name)**

Removes the object bound to the given name in the application's application layer data. Does nothing if there is no object bound to the given name.

Parameters: name - the name of the object to remove

Notice: All of the above methods are synchronized, so multiple threads at the same time to use these methods are safe.

9.2 How to use application

The most typical application of "application" is chat room, put everybody's speech content into "application", so each other can see the speech content from "application".

Now look at a simple example of the chat room, this example use the upper and lower frame, total of three documents.

The "chat.htm" is framework file, source code:

```
1 <html>
2 <head>
3   <title>Chat room</title>
```

```

4 </head>
5 <frameset cols="72%,*">
6 <frame name = "message" src="chat1.dqm">
7 <frame name = "say" src="chat2.dqm">
8 </frameset>
9 <html>

```

From above source code line 6 and 7 you can see this framework file include two dqm files: chat1.dqm and chat2.dqm.

The dqm file “chat2.dqm” can put speech content into application, it’s source code:

```

1 <%@page import = "java.util.ArrayList;java.util.Date" %>
2
3 <html>
4 <head>
5 <title>Speak</title>
6 </head>
7
8 <body>
9 <form name="form1" method="post" action="">
10 Please speak: <input type="text" name="pronunciation" size="30">
11 <input type="submit" value="send">
12 </form>
13
14 <%
15 String tempLine;
16 if ((tempLine = request.getParameter("pronunciation")) != null) { //If has the speech
    content
17 ArrayList<String> content = (ArrayList)application.getAttribute("chatContent");
18 if (content == null) {
19 content = new ArrayList<String>();
20 application.setAttribute("chatContent", content);
21 }
22 int overNo = content.size() - 39;
23 if (overNo > 0) { //If application have too many speech contents, then delete it.
24 for (int i = 0; i < overNo; i++) {
25 content.remove(i);
26 }
27 }
28 Date date = new Date();
29 content.add(request.getRemoteAddr() + "<font color=\`#FF0000\`">(" + date +
    "</font><br>" + tempLine + "<br>");
30 }
31 %>

```

```
32
33 </body>
34 </html>
```

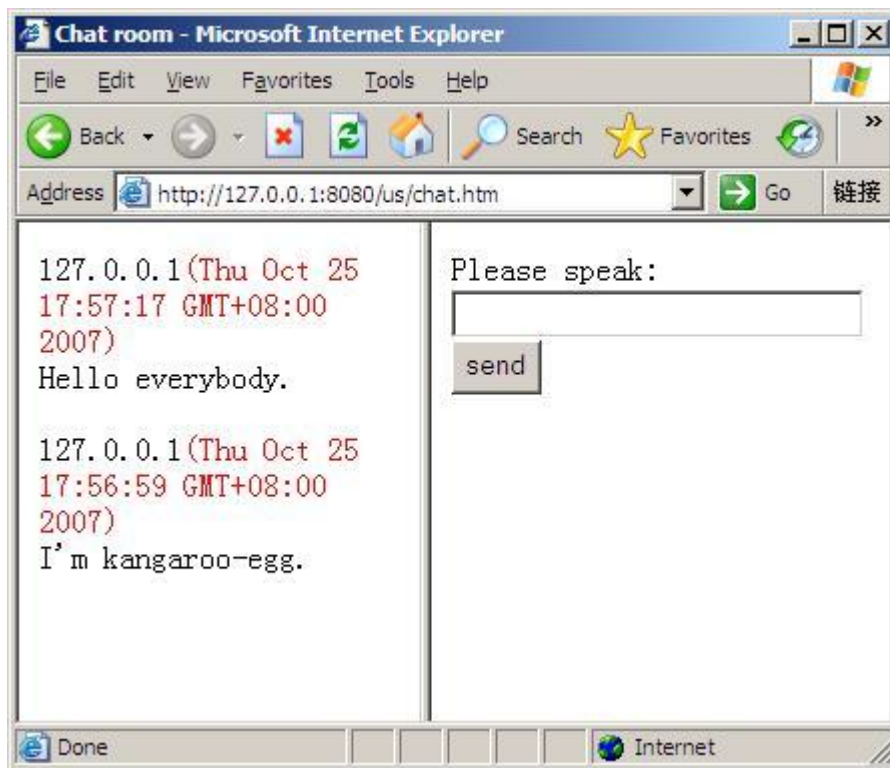
From above source code line 29 you can see speech content, speech time and IP of speaker all will save in a variable (this variable name is “content”, type is ArrayList). And “content” variable will put into “application” object, so id first execute this dqm then will create a new “content” variable into “application” (source code line 17 to 21). The “content” variable most preserves 40 records of speech (source code line 22 to 27).

The dqm file “chat1.dqm” can display speech content, source code:

```
1 <%@page import = "java.util.ArrayList" %>
2
3 <html>
4 <head>
5   <title>speech content</title>
6   <meta http-equiv="refresh" content="5">
7 </head>
8
9 <body>
10  <%
11   ArrayList<String> content = (ArrayList)application.getAttribute("chatContent");
12   if (content != null) {
13     for (int i = content.size() - 1; i >= 0; i--) {
14       out.println(content.get(i) + "<br>");
15     }
16   }
17   %>
18 </body>
19 </html>
```

This page will automatic refresh every 5 seconds (above source code line 6). Automatic refresh can constantly display the latest speech content (source code line 10 to 17).

The picture of implementation:



Picture 9-2-1

You can through different client to visit this chat room, it will like multi-people to chat.

Chapter 10 Built-in object : command

This chapter is going to introduce last built-in object “command”, other built-in objects are directed at the web procedure, but this chapter must introduce the object is directed at server's.

10.1 Introduce command object

The “command” is a special built-in object, only DQM container has this object. User can get or change server’s current information through this object, use this object you can apply new value to server, but not to need restart server.

Notice: Default command object is closed, we can not use it, if need to use it please enable command function (refer to section 3.2).

10.2 Use command to process session and application

1. **public int getSessionCount()**

Get the the quantity of current host’s sessions, including not expired and already expired(but had not cleared, refer to section 8.4) sessions.

Return: the current session quantity

2. **public void clearAllSession()**

Delete current host’s all sessions. All sessions will be cleared from memory, including not expired and already expired(but had not cleared) sessions.

For example, the dqm file “commandSession.dqm”, source code:

```
1 <% @page command = "true" %>
2
3 <%if (request.getParameter("clear") == null) { %>
4     Current host's session quantity, including not expired and already expired(but had not
      cleared) sessions: <%=command.getSessionCount()%>
5     <p><a href="?clear">Delete current host's sessions</a></p>
```

```

6  <% }
7  else {
8      command.clearAllSession();%>
9      Delete current host's sessions ok.
10 <% }%>

```

When visiting this dqm file, it will show current host's sessions quantity (source code line 4), next we clear current host's sessions (source code line 5 to 10), then you can try visit this dqm file again, you will find current host only has one session (That only session is you are using now).

3. **public int getApplicationCount()**

Return: how many items in current host's application.

4. **public void clearAllApplication()**

Removes all items from current host's application.

For example:

The dqm file "commandApplication1.dqm" can add five items into current host's application, source code:

```

1  Add five items into current host's application
2  <%
3  for (int i = 0; i < 5; i++) {
4      application.setAttribute("" + i, i);
5  }
6  %>

```

The dqm file "commandApplication2.dqm" can get the quantity of current application's items and clear all application's items, source code:

```

1  <% @page command = "true" %>
2
3  <%if (request.getParameter("clear") == null) { %>
4      How many items in current host's application: <%=command.getApplicationCount()%>
5      <p><a href="?clear">Delete all current application's items.</a></p>
6  <% }
7  else {
8      command.clearAllApplication();%>
9      Delete all current application's items ok.
10 <% }%>

```

First visit "commandApplication1.dqm" for add 5 items into current application (commandApplication1.dqm source code line 3 to 5). Next visit "commandApplication2.dqm" you will see current application has 5 items (if there are other programs also add item into application, then the quantity of application's items will bigger than 5, commandApplication2.dqm

source code line 4). You can clear current application's items (commandApplication2.dqm source code line 5 to 10), then you can try visit this dqm file again, you will find current application has zero item.

5. **public String getDhtmlExtName()**

Returns: dynamic web page's extended name of current host. If you set dynamic web page's extended name to "dqm" then will return ".dqm", how to set dynamic web page's extended name please refer to section 2.3 and 2.5.

Notice: Return of dynamic web page's extended name always lowercase and the front is the point symbol. The dynamic web page's extended name is case-insensitive. If you set dynamic web page's extended name to "dqm", then "A.Dqm" and "b.dQm" and "c.dqm" all are executable file, even if under linux and unix is also this. In this manual "dynamic web page" is equal to "dqm file".

10.3 Use command to process Dqm Instance Buffer Pool

DQM container has a pool to store instance of dqm file (please refer to addendum 1), this section we will introduce use command how to process "dqm instance buffer pool".

1. **public int getDhtmlInstanceCount()**

Return: the quantity of current dqm instance buffer pool's instances

2. **public void clearAllDhtmlInstance()**

Remove all instances from current "dqm instance buffer pool".

3. **public boolean clearDhtmlInstance(String http_file)**

Remove the specified dqm file instance from current "dqm instance buffer pool" if it is present.

Parameters: http_file - dqm file instance to be removed from current "dqm instance buffer pool", if present. For instance, if need to remove instance of "<http://youname/chat/main.dqm>", then "http_file" value should be: "/chat/main.dqm".

Returns: true if current "dqm instance buffer pool" contained the specified dqm file instance

We have to give an example, "testDhtmlInstancePool.dqm" can prove that dqm file instance is stored in the "dqm instance buffer pool", its source code:

```
<%! private int ct = 0;%> How many times this dqm file by visited: <%=++ct%>.
```

If using statement `<%!%>` that means all declaration will become class member. Here we declare a private static class member variable: “ct”, it’s type is integer. This variable uses in recording the times of visits this dqm file. But if re-new this dqm file then “ct” will become zero. In fact when you visit this dqm file you will see how many times this dqm file by visited. This proves that the dqm file only initialize single instance, and put the instance into “dqm instance buffer pool”.

The dqm file “clearDhtmlInstance.dqm” can check and delete dqm file instance from current “dqm instance buffer pool”, it’s source code:

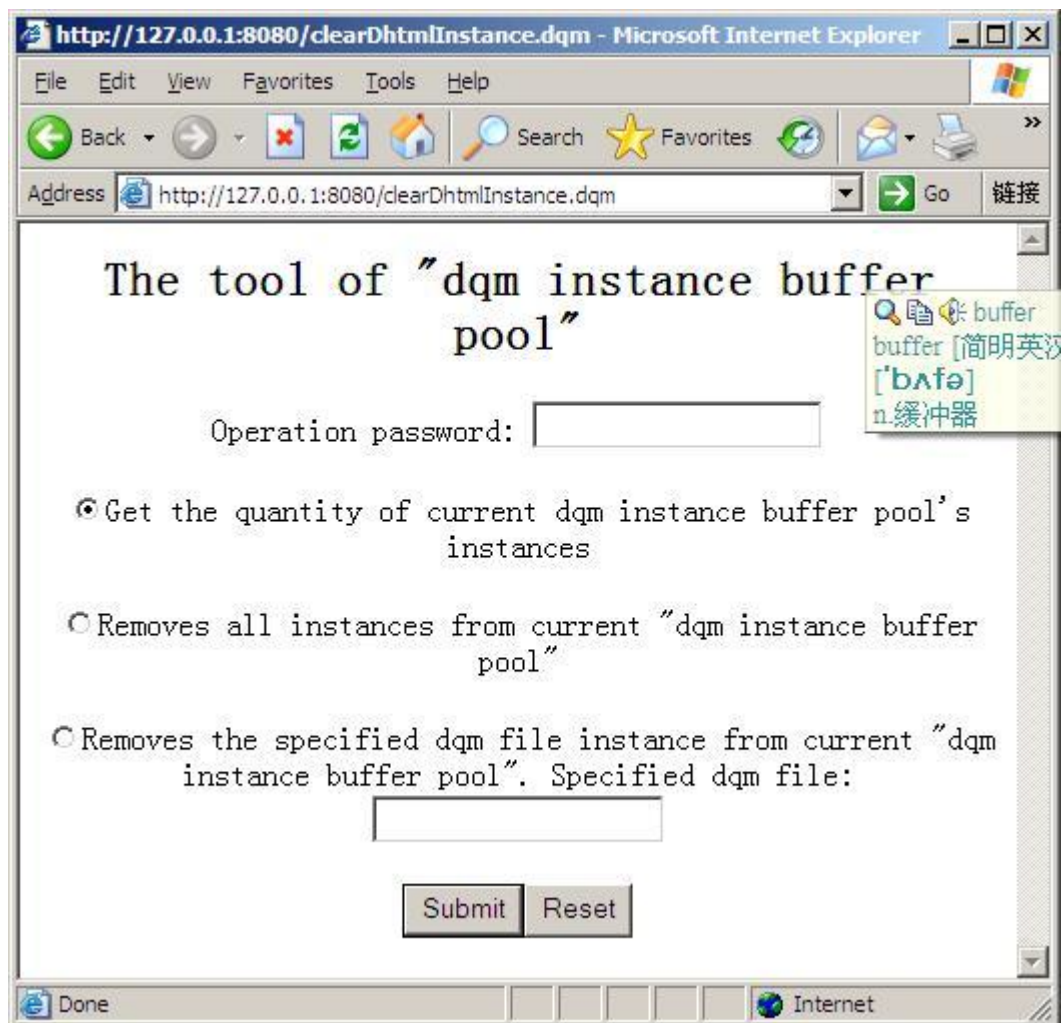
```
1  <% @page command="true"%>
2  <%
3  String password = "123456";
4  if (request.getParameter("B1") == null) {
5  %>
6
7  <form method="POST" action="">
8      <p align="center"><b><font size="5">The tool of "dqm instance buffer
pool"</font></b></p>
9      <p align="center">Operation password: <input type="password" name="psw"
size="20"></p>
10     <p align="center"><input type="radio" value="getSize" checked
name="system">Get the quantity of current dqm instance buffer pool's instances</p>
11     <p align="center"><input type="radio" name="system" value="clearAll">Removes
all instances from current "dqm instance buffer pool"</p>
12     <p align="center"><input type="radio" name="system" value="clear">Removes the
specified dqm file instance from current "dqm instance buffer pool". Specified dqm file:
<input type="text" name="dhtmlName" size="20"></p>
13     <p align="center"><input type="submit" value="Submit" name="B1"><input
type="reset" value="Reset" name="B2"></p>
14 </form>
15
16 <% }
17 else { %>
18
19 <p align="center"><a href="<%=request.getNowUrlFolder() +
request.getNowFile().getName()%>">Return</a></p>
20
21 <%
22 if (!password.equals(request.getParameter("psw"))) {
23     out.println("The operation password is wrong!!");
24     return;
25 }
```

```

26 //
27 String system = request.getParameter("system");
28
29 if ("getSize".equals(system)) {
30     out.println("How many dqm instances in the current \"dqm instance buffer pool\": " +
31         command.getDhtmlInstanceCount());
32     return;
33 }
34 if ("clearAll".equals(system)) {
35     command.clearAllDhtmlInstance();
36     out.println("Removed all instances from current \"dqm instance buffer pool\".");
37     return;
38 }
39
40 if ("clear".equals(system)) {
41     if (command.clearDhtmlInstance(request.getParameter("dhtmlName"))) {
42         out.println("Removes the specified dqm file instance from current \"dqm instance
43         buffer pool\", successful implementation.");
44     }
45     else {
46         out.println("Removes the specified dqm file instance from current \"dqm instance
47         buffer pool\", implementation failure.");
48     }
49     return;
50 }
51
52 %>
<% }%>

```

This program's interface:



Picture 10-3-1

This program has three functions, user can use three buttons to choose the three functions, but must first input the correct operation password. In here password is “123456”, it’s defined by “password” variable (source code line 3), if you want to change password please change this variable’s value.

We introduce the first function (source code line: 29 to 32), it can get the quantity of current dqm instance buffer pool's instances, this function was used “getDhtmlInstanceCount()” method. If restart the server to ensure that “dqm instance buffer pool” does not contain any instances, then visit “testDhtmlInstancePool.dqm” file, next visit “clearDhtmlInstance.dqm” file and use current function to check quantity of instance, you will see current has 2 dqm instance. Why it has 2 instances? First we visited “testDhtmlInstancePool.dqm” file, so put this dqm file’s instance into the “dqm instance buffer pool”. But what is another instance? Do not forget that we get instance quantity through “clearDhtmlInstance.dqm” file, so pool has 2 instances: “testDhtmlInstancePool.dqm” and “clearDhtmlInstance.dqm”.

Ok next we introduce the second function (source code line: 34 to 38) it can remove all instances from current “dqm instance buffer pool”, this function was used “clearAllDhtmlInstance()” method. We can try to visit many dqm files, then use first function to get current instance quantity, next we use current function to remove all dqm instances, next if we

use first function to get current instance quantity again, we will find current dqm instance pool only has one instance (it's your current visiting dqm file: "clearDhtmlInstance.dqm").

Sometimes we just want to remove the specified dqm file instance from current "dqm instance buffer pool", not need to remove all instances, this time we can use "clearDhtmlInstance" method. The third function can remove the specified dqm file instance (source code line: 40 to 48). We can try to visit many dqm files (include "testDhtmlInstancePool.dqm"), then use first function to get current instance quantity, next we try to remove instance of "testDhtmlInstancePool.dqm", only input "/testDhtmlInstancePool.dqm" into "Specified dqm file" item, because this dqm file under root folder. Ok now we use first function to get current instance quantity again, will find a instance was removed.

10.4 Use command to process Class Buffer Pool

In section 3.4 we introduced main host or virtual host has "/WEB-INF/classes" and "/WEB-INF/lib" folder, you can put jar and class files into these two folders, if the dqm file need to use the jar and class files then system will automatically load it from the two folders. In order to improve performance, class loader installed a "class buffer pool". First class loader will check "class buffer pool" when need to load a class, if the pool already has the class then load it from pool. If the pool does not have the class then load it into pool, later if other dqm file need to use the same class then can load it from "class buffer pool".

However, there is a question that "class buffer pool" can't check whether the jar or class file was updated, Why not check, because the jar or class files usually do not update, even if the jar or class files were updated, we still can manually reload new class file to "class buffer pool" without restart server (Use the following method).

1. **public boolean clearAllLoadedClass()**

Removes all classes from current "class buffer pool".

Return: true if remove successfully, else return false.

For example, first we create a java class, it's full name is "test.Test", it's source code:

```
1 package test;
2
3 public class Test {
4     public static String getInfo() {
5         return "one";
6     }
7 }
```

Please compile above java file and put it's class file into "/WEB-INF/classes" folder (notice the file's package name, full class path is "/WEB-INF/classes/test/Test.class"), next create a "testClassPool.dqm", this dqm file will use "test.Test", source code:

The content of "Test" class: <%=test.Test.getInfo()%>

Visit abovedqm file, you will find it will read information from "test.Test" class, the result is:
The content of "Test" class: one.

Next we change method return value, its source code:

```
1 package test;
2
3 public class Test {
4     public static String getInfo() {
5         return "two";
6     }
7 }
```

You can see the value of returned was changed to "two", and overwrite original class file after re-compile it, next visit "testClassPool.dqm" again (don't restart server), you will find that the results still show "one". At the begin of this section we already introduced, because "class buffer pool" does not detect the jar or class file has been updated, so class loader will still get old class from "class buffer pool", and use it, therefore the results still showed "one".

So we can use "clearAllLoadedClass" method to remove all classes from current "class buffer pool". Using this method does not need to restart the server, and "class buffer pool" not include "test.Test" class so it will reload this class again. Notice that to the "class buffer pool" no method can remove the specified class.

The dqm file "clearClassPool.dqm" can remove all classes from current "class buffer pool", its source code:

```
1 <% @page command="true"%>
2 <%
3 String password = "123456";
4 if (request.getParameter("B1") == null) {
5 %>
6
7 <form method="POST" action="">
8     <p align="center"><b><font size="5">The tool of "class buffer
9 pool"</font></b></p>
10     <p align="center">Operation password: <input type="password" name="psw"
11 size="20"></p>
12     <p align="center">Removes all classes from current "class buffer pool"</p>
13     <p align="center"><input type="submit" value="Submit" name="B1"><input
14 type="reset" value="Reset" name="B2"></p>
15 </form>
16
17 <% }
18 else { %>
```

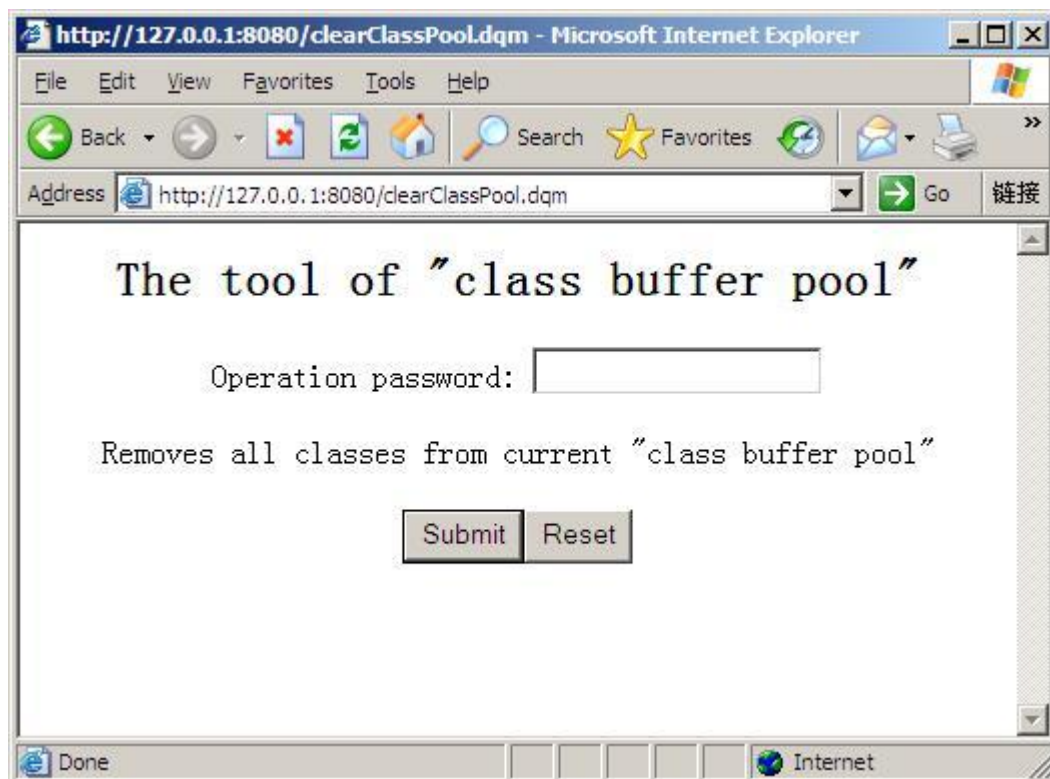


```

16
17 <p align="center"><a href="<%=request.getNowUrlFolder() +
request.getNowFile().getName()%>">Return</a></p>
18
19 <%
20 if (!password.equals(request.getParameter("psw"))) {
21     out.println("The operation password is wrong!!");
22     return;
23 }
24 //
25
26 if (command.clearAllLoadedClass()) {
27     out.println("Removes all classes from current \"class buffer pool\", successful
implementation.");
28 }
29 else {
30     out.println("Removes all classes from current \"class buffer pool\", implementation
failure.");
31 }
32
33 }%>

```

This program's interface:



Picture 10-4-1

You will find this program very like “clearDhtmlInstance.dqm”(at section 10.3), but this program only has one function: removes all classes from current “class buffer pool”. Similarly if want to execute this function you must first input the correct operation password. In here password is “123456”, it’s defined by “password” variable (source code line 3), if you want to change password please change this variable’s value.

Removes all classes from current “class buffer pool”, this function was used “clearAllLoadedClass ()” method (source code line 26).

Next use this program we can continue to the previous example, input operation password and execute function of remove classes (if execute not successful please try again). After remove all classes from current "class buffer pool" we visit “testClassPool.dqm” again, but you will find the execution result was still display “one”. Why? Old classes was already removed from pool, but don’t forget “dqm instance buffer pool”, so we must remove instance of “testClassPool.dqm” from "dqm instance buffer pool" too (use clearDhtmlInstance.dqm at section 10.3). Now visit “testClassPool.dqm” again, the results finally display "two."

Notice: JDK itself has a class called java.lang.Class, this section said class is this java.lang.Class, not say class’s instance. Only classes that under “/WEB-INF/classes” and “/WEB-INF/lib” folder will enter “class buffer pool”. The class that use other method loaded will not enter “class buffer pool” (refer addendum 5).

10.5 Soft restart the server

The command object has a method can soft restart kangaroo-egg server.

public void resetHost()

This method can restore condition that current host to server starting time condition. This method only can effect current host, equal to restart server, but not need to shutdown server. In fact this method equal to execute “clearAllApplication”, “clearAllDhtmlInstance”, “clearAllLoadedClass”, “clearAllSession” and “clearAllStaticPage” these five methods at same time. About “clearAllStaticPage” method we will introduce in section 12.2.

10.6 Log operation

At section 2.1, we introduced “saveLogs” element, you can save visit logs to file use this element. But server will save logs to buffer first, when buffer full then save all logs that in buffer to file. So below method of command object can get logs buffer’s content. Please notice that if disable “saveLogs” function then visit logs will not save to buffer too.

public String getLogBuf(String psw)

Read the log records from server log buffer,

Parameters: psw - Because the log records is based on the whole server, so need to

provide the password to get logs. About password set up please refer to “systemPsw” of section 2.1.

Returns: If provide the correct password then returns buffer’s log record, else return null.

10.7 Password protection

At section 2.3 and 2.5 we introduced “needPassword” item, use it can enable web server’s authentication, but user must input correct username and password before visit resource. But sometimes we need to get username which already logon, so Command has this method:

public String getAuthUser()

Get the username which already logon current host.

Returns: If enable web authentication and successful login then return the logon name, else return null.

Why needs to have this method? If we do not want to use dqm languages to code authentication program, then can use server authentication (refer to “needPassword” element at section 2.3 and 2.5). If use server authentication then you can use this method to get logon name, for example you can get logon name and put it into session, maybe other dqm program need to use logon name.

Chapter 11 Use JavaBean

JavaBeans components are Java classes that can be easily reused and composed together into applications. Any Java class that follows certain design conventions can be a JavaBeans component.

DQM technology directly supports using JavaBeans components with dqm language elements. You can easily create and initialize beans and get and set the values of their properties. This chapter provides information about JavaBeans components and the dqm language elements for accessing beans in your dqm pages.

11.1 Introduction JavaBean

JavaBeans are usual Java classes which adhere to certain coding conventions. Following are the coding conventions I am talking about:

1. Must public class
2. Implements `java.io.Serializable` interface
3. Provides no argument constructor
4. Provides getter and setter methods for accessing it's properties

Use JavaBean has two advantages:

1. Causes HTML and the java procedure separates, can advantageous for the maintenance code.
2. DQM scripting language focused on generating dynamic web page, JavaBean focused on transaction processing, like this can fully use the reusability of JavaBean component, improve the efficiency of the development web site.

We give an example of JavaBean, “mybean.Calculate.java” is a JavaBean, it’s used to calculate the data, it’s source code:

```
1 package mybean;  
2
```

```

3 public class Calculate {
4     public Calculate() {
5     }
6
7     //Adder calculation
8     public double add() {
9         return num1 + num2;
10    }
11
12    //Subtraction calculation
13    public double minus() {
14        return num1 - num2;
15    }
16
17    public void setNum1(double inum1) {
18        num1 = inum1;
19    }
20
21    public void setNum2(double inum2) {
22        num2 = inum2;
23    }
24
25    private double num1;
26    private double num2;
27 }

```

In this JavaBean we can set two float numbers use “setNum1” and “setNum2” methods, execute “add” and “minus” methods can get calculated results.

Please compile above java file and put it’s class file into “/WEB-INF/classes” folder (notice the file’s package name, full class path is “/WEB-INF/classes/mybean/Calculate.class”)

11.2 DQM JavaBean tags

In dqm file we can use java code or DQM JavaBean tags to declare and instantiate the JavaBean class. We recommend use DQM JavaBean tags, this can reduce the java code in dqm file. Next we will introduce DQM JavaBean tags.

DQM JavaBean tags syntax:

```
<%@bean id="object-name" class="classname" scope="object scope"%>
```

Let us now see what are the different attributes :

id - name of the object e.g.

```
String myName = null;
```

In the above code, “myName” is the 'id' we are talking about.

scope - an optional attribute by which you can control when your JavaBean object will be destroyed. Default is page, which means every page view will create a new JavaBean. About scope we will introduce in the next section.

class - a class name e.g.

```
Date d = new java.util.Date();
```

In the above code, “java.util.Date” is the class we are talking about. If not import package path then need to write fully qualified class name.

We now give an example, “useJavaBean.dqm” used the above “mybean.Calculate” JavaBean, source code as follows:

```
1  <% @bean id="myBean" class="mybean.Calculate" scope="page"%>
2
3  <%
4  //set two float numbers which need to calculate
5  myBean.setNum1(12.23);
6  myBean.setNum2(0.88);
7  %>
8
9  <pre>
10 Result of add: <%=myBean.add()%>
11 Result of subtration: <%=myBean.minus()%>
12 </pre>
```

At source code 1 program used DQM JavaBean tag to declare and instantiate the “mybean.Calculate” JavaBean. Next we set two float numbers which need to calculate (source code line: 5 ,6), then calculate these two numbers and display result (source code line: 10, 11).

Result of add: 13.110000000000001

Result of subtration: 11.35

Although this JavaBean looks like very not necessary, such a simple calculation program we can write directly in the dqm file, why use javaBean? Certainly I also have such idea, but here just an example. But if you need to process the massive complicated calculations then this has the advantage when using JavaBean. It can causes HTML and the java procedure separates, can advantageous for the maintenance code. If the calculation needs to change that basically only revises the code in JavaBean, in dqm file's code only needs very little change or even without any change.

11.3 JavaBean object Scope

Every JavaBean class object or any other class object we create will have a scope. Scope means the length of time this object will remain in memory. There are three kinds of scopes :

1. page - it means a new object will be created and destroyed for every page view. This is the default for DQM JavaBean tag when you don't explicitly give it any.
2. session - the newly created object will be bound to the session object. What this means is that every visitor coming to your site will have a separate session for it, so you will not have to create a new object every time for it. You can just retrieve that object later again from the session object when you want it.
3. application - an object bound to application object means that your object will stay as long as the application remains loaded. This can be useful when for instance you want to count page views or daily sessions for your site.

For example, first we create a javaBean: "mybean.CounterBean", in this bean we declare a value, it's name is "value", and we create methods of set and get this value. Source code:

```
1 package mybean;
2
3 public class CounterBean {
4     public CounterBean() {
5     }
6
7     public int getCount() {
8         return count;
9     }
10
11    public void setCount(int count) {
12        this.count = count;
13    }
14
15    private int count = 0;
16 }
```

Next we create a new dqm file "counterBean.dqm", this dqm file will use "CounterBean, it's source code:

```
1 <%@page import = "mybean.CounterBean"%>
2 <%@ bean id="myBean" scope="page" class="mybean.CounterBean"%>
3
4 <pre>
5 The current counter's value is: <%=myBean.getCount()%>
6
7 <%//Increase count value
8 myBean.setCount(myBean.getCount() + 1);
9
10 //Next judgment myBean object scope
11 CounterBean obj = null;
12 String scope = null;
13
```

```

14 //If object is session scope
15 obj = (CounterBean)session.getAttribute("myBean");
16 if (obj != null)
17     scope = "session";
18
19 //If object is application scope
20 obj = (CounterBean)application.getAttribute("myBean");
21 if (obj != null)
22     scope = "application";
23
24 //If object scope is not session or application then must is page
25 if (scope == null)
26     scope = "page";
27 %>
28
29 scope=<%=scope%>
30 </pre>

```

At above source code line 2, you can see that JavaBean object scope is “page”, we visit this dqn file you will see result:

```

The current counter's value is: 0
scope=page

```

You can try to refresh this page, but you will find “count” variable’s value always display 0. Because JavaBean object will be created and destroyed for every page view.

Next we change JavaBean object scope to “session”, change source code line 2 to

```
<%@ bean id="myBean" scope="session" class="mybean.CounterBean"%>
```

We visit it again, first you will see “count” variable’s value is 0, but if you refresh it then “count” variable’s value will plus 1 every time. But if you close browser and open browser visit it again then “count” variable’s value become 0 again. Because created JavaBean object will be bound to the session object. By the way if use session scope then we can directly get JavaBean object from session too (source code line 15).

Finally we change JavaBean object scope to “application”, change source code line 2 to

```
<%@ bean id="myBean" scope="application" class="mybean.CounterBean"%>
```

If you refresh this dqn file then “count” variable’s value will plus 1 every time. If you close browser and open browser visit it again then “count” variable’s value not will reset to 0 and it will continue to increase. Because created JavaBean object will be bound to application object. By the way if use application scope then we can directly get JavaBean object from application too (source code line 20).

Chapter 12 Create static web page

Uses the DQM technology we can create dynamic web page easily, but some cases, the result of execute dqm file will remain unchanged longtime. For example, a notice system, it maybe publish a notice in several months (write notice into database), but many users will read notices every day (read notice from database when user visit it). Because the frequency of update notice is very low, so so most of the time users read the contents of the notice is the same. In this kind of situation, if we generated a static web page when publish new notices every time, then the user read this static web page can know the latest notices. Use this method system will not access database when users read notices via dqm file, so performance improved. Of course, we can manually generate these static web pages, but it will be very troublesome and maintain with difficulty. So next we will introduce use kangaroo-egg server generated the static web page automatically.

12.1 The ID of static web page

If dqm file need to generate static web page then first must provide an unique static web page ID (for short: static ID), kangaroo-egg will check static web page use static ID, if the static web page which is corresponding to the static ID exists then directly output this static web page, else then server will first generate static web page by this static ID, next output static web page which newly generated. May see in the static web page function this static ID is very important, so how we define static ID? The DQM container has provided a method for dqm file, if user need to define static ID then must override(overwrite) this method.

public String getStaticPageId(

DRequestItf request, DSessionItf session, DApplicationItf application)

Define a static ID according to the current visit's condition.

Parameters: request - request bulid-in object in current dqm file, refer to chapter 6.

session - session bulid-in object in current dqm file, refer to chapter 8.

application - application bulid-in object in current dqm file, refer to chapter 9.

Returns: a static web page ID, if return null then means disable static web page function.

Ok next we need to solve several questions.

First question, “request”, “session” and “application” of these built-in variables can be used directly, so why we still transmit these built-in variables into “getStaticPageId” method? Because it’s a class method, class method must write in `<%! %>` block. But build-in object only can be used in `<% %>` block.

The method of “getStaticPageId” is a reserved by kangaroo-egg server, therefore please do not re-definition return type of this method. For example if you definid below method then server will throw error.

```
public int getStaticPageId(  
    DRequestItface request, DSessionItface session, DApplicationItface application)
```

Are there will happen problem if we not override “getStaticPageId” method in dqm file? Of course not, if not override it then will use default provides method. The default method will directly return null, that is disable the static web pages function.

What is use of three variables: request, session and application? You may also very puzzled about it. When we generate static ID usually need to use these three variables, for example, a bbs thread page has many pages, user need click “next_page url” to see next page, so the first page’s content and the second page’s content is not the same, then we must generate an unique static ID for each page. But how did we know that which page the user visiting now? This time we may use these three variables to get the page number which use visiting now.

We give an example, create a dqm file “/ test_folder/mpage.dqm”, it can has many pages, use URL parameter “page” to control display which page’s content. If we visit first page then dqm file will display “Content of page 1”, if we visit second page then it will display “Content of page 2”, and so on.

Below is rule of URL parameter “page”:

“/test_folder/mpage.dqm?page=1” or “/test/viewnews.dqm” Display page 1.

“/test_folder/mpage.dqm?page=2” Display page 2.

“/test_folder/mpage.dqm?page=hd” If value of “page” is not a digit then display page 1 too.

Therefore if we need to generate static ID for “mpage.dqm”, we must consider these situations.

Below is source code of “mpage.dqm”:

```
1  <%  
2  int pageid;  
3  String pageIdStr = request.getParameter("page");  
4  if (pageIdStr != null) {  
5      try {  
6          //Get page number which user current visiting from parameter "page"  
7          pageid = Integer.parseInt(pageIdStr);  
8      }  
9      catch(NumberFormatException e) { //If page number is not a digit  
10         pageid = 1;  
11     }  
12 }  
13 else { //If URL not include parameter "page" then default set first page  
14     pageid = 1;  
15 }  
16  
17 out.println("Content of page "+ pageid);  
18 %>  
19
```

```

20 <%!
21 public String getStaticPageId(DRequestItface request, DSessionItface session,
   DApplicationItface application) {
22     String pageIdStr = request.getParameter("page");
23     if (pageIdStr != null) {
24         try {
25             return "/test_folder/mpage.dqm_page" + Integer.parseInt(pageIdStr);
26         }
27         catch(NumberFormatException e) { //If page number is not a digit
28             return "/test_folder/mpage.dqm_page" + 1;
29         }
30     }
31     else { //If URL not include parameter "page" then default set first page
32         return "/test_folder/mpage.dqm_page" + 1;
33     }
34 }
35 %>

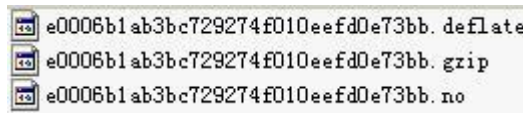
```

At source code line 4, it check parameter “page”, if URL include parameter “page” then check it’s value, if it’s value is a digit then get it (source code line: 7).

If URL not include parameter “page” (source code line: 13 to 15) or parameter’s value is not a digit (source code line: 9 to 11) then default set page number to 1. At source code line 17, it output current page’s content.

At source code line 20 to 35, we override the “getStaticPageId” method to enable static web page function. At source code line 22, we get parameter which name is “page” from “request” variable. If we can get parameter “page” from “request” variable then check it’s value, if it’s value is a digit then return static ID of current page number (source code line: 25). If we can’t get parameter “page” from “request” variable (source code line: 31 to 33) or parameter’s value is not a digit (source code line: 27 to 29) then return first page’s static ID.

Next we discuss how to assign static pages ID. At source code line 25, 28 and 32, all will return static ID, first we must know the usefulness of static ID before know how to assign static ID. In actually the mean of static ID is very simple, the static ID represent static page web page, you may understand that static ID is the static web file’s filename. In the present example, when user visit first page then program will return “/test_folder/mpage.dqm_page1” as static ID (source code line: 28 and 32), so server will seek static web page file which corresponds this ID, if static web page file exists then output it, if it not exist first generate static web page by this static ID, next time user visit it again can directly output the static web file which already generated. Users can try to visit the above dqm file’s first page, you will find server will automatically create “static_page” folder under “WEB-INF” directory. At the same time you will find that there are three files under “static_page” folder, see below:



Picture 12-1-1

We can see that these three files' main name is the same, all are "e0006b1ab3bc729274f010eefd0e73bb".

Why can be that strange name? In fact, this strange name is MD5 code by "/test_folder/mpage.dqm_page1". You can understand that MD5 code is another kind of written format. About static id: "/test_folder/mpage.dqm_page1", it returns from source code line 28 or 32.

Next another question, why are there three files? In section 4.3 we know that the server can support two kinds of compression output, gzip and deflate. So server will create three static files, two are the compressed file (extension name is "gzip" and "deflate"), one is the uncompressed file (extension name is "no"). If current browser support gzip compression, then static file which extension name is "gzip" will be output.

Next we listed all possible situations of visit "viewnews.dqm" first page:

| Browser supports compression format | Static ID | Static file name(code static ID to MD5) |
|-------------------------------------|------------------------------|--|
| gzip | /test_folder/mpage.dqm_page1 | e0006b1ab3bc729274f010eefd0e73bb.gzip |
| deflate | /test_folder/mpage.dqm_page1 | e0006b1ab3bc729274f010eefd0e73bb.deflate |
| not support the above compression | /test_folder/mpage.dqm_page1 | e0006b1ab3bc729274f010eefd0e73bb.no |

Table 12-1-2

From the above explanation readers should know why server will generate three static page files. If static page file's extension name is "gzip" then means that its content to be compressed on gzip format. If static page file's extension name is "deflate" then means that its content to be compressed on deflate format. If static page file's extension name is "no" then means that its content not to be compressed. Each static ID have three static files, it looks really complicated enough, but for users who do not need to care about this, because the system will automatically handle these documents, for users these static files are all transparent, so we introduced it only for user can better understanding static page function.

Ok next we return to discuss the static ID. From source code of "viewnews.dqm", we can see this dqm file every page's content is not the same, so we must base on each page to generate static pages. Will server generate 100 static web files if this dqm file has 100 pages? Yes, because the content of these 100 pages are not same, so we need to provide 100 unique static IDs for generate 100 static web pages. What will happen when we provide reduplicate static ID? Ok, we give an example, if we provide a same static ID for viewnews.dqm's first page and second page, then if server first generate static file of first page, next even visit this dqm file's second page but you will find it will still display the content of first static page. If server first generate static file of second page, then even visit this dqm file's first page but you will find it will still display the content of second static page.

From above introduced that we can see the static ID is core of static web page. From “viewnews.dqm” source code line 25, 28 and 32, you can see we use dqm file’s absolute path and visit page number to generate unique static ID. We recommend use this method to generate static ID, because use dqm file’s absolute path can guarantee unique static ID to be generated. In fact, we can automatically obtain the absolute path, see below source code:

```
1  <%
2  int pageid;
3  String pageIdStr = request.getParameter("page");
4  if (pageIdStr != null) {
5      try {
6          //Get page number which user current visiting from parameter "page"
7          pageid = Integer.parseInt(pageIdStr);
8      }
9      catch(NumberFormatException e) { //If page number is not a digit
10         pageid = 1;
11     }
12 }
13 else { //If URL not include parameter "page" then default set first page
14     pageid = 1;
15 }
16
17 out.println("Content of page "+ pageid);
18 %>
19
20 <%!
21 public String getStaticPageId(DRequestItface request, DSessionItface session,
22 DApplicationItface application) {
23     String pageIdStr = request.getParameter("page");
24     if (pageIdStr != null) {
25         try {
26             return request.getNowPath() + "_page" + Integer.parseInt(pageIdStr);
27         }
28         catch(NumberFormatException e) { //If page number is not a digit
29             return request.getNowPath() + "_page" + 1;
30         }
31     }
32     else { //If URL not include parameter "page" then default set first page
33         return request.getNowPath() + "_page" + 1;
34     }
35 }
36 %>
```

Of course, users can generate static ID according to your actual situation. The “viewnews.dqm”

dqm file only need to use “request” build-in object when generating static ID.

12.2 Regist static web page

The previous section introduced how to assign static ID in order to open the static web page function, but how many static web pages in current host, we can know? Of course, user can know how many static page current has produced? Because static ID and it page’s information will be registered in the system, so user can read these registration information from system at any time.

But how to read these registration information from system? Do you remember command object which introduced at chapter 10. Command object include methods which operate static web page.

1. **public int getStaticPageCount()**

Returns the number of static web pages in current host.

Returns: the number of static web pages in current host.

2. **public boolean clearStaticPage(String staticPageId)**

Removes the static page of the specified staticPageId from current host, if it is present (optional operation).

Parameters: Static page to be removed from current host, if present

Returns: true if current host contained the specified static web page.

3. **public void clearAllStaticPage()**

Removes all of the static web page from current host.

The first method is to be understood easy, it can get the number of static web pages in current host. But what is the usefulness of second and third methods? Because the system will not inspect the dqm file whether already expired, so we must use these 2 methods to manually delete static web pages, like this system will re-created static web page when we visit dqm file again.

For example, the dqm file “/test_folder/readfile.dqm” will load “content.txt” file and output it’s content, source code as below:

```
1 <pre>
2 <%
3 File readFile = new File(request.getNowFolder() + File.separatorChar + "content.txt");
4 if (readFile.exists()) {
5     BufferedReader br = null;
6     try {
7         br = new BufferedReader(new InputStreamReader(new FileInputStream(readFile),
8 "GBK"));
9         String line;
10        out.println("The following is the content which reads from content.txt:");
```

```

10     while ((line = br.readLine()) != null) {
11         out.println(line, "GBK");
12     }
13 }
14 finally {
15     if (br != null) {
16         br.close();
17     }
18 }
19 }
20 else {
21     out.print("File does not exist!");
22 }
23 %>
24 </pre>
25 <%!
26 public String getStaticPageId(DRequestItface request, DSessionItface session,
27     DApplicationItface application) {
28     return "/test_folder/readfile.dqm";
29 }
29 %>

```

At source code line 3 we define the “content.txt” file which need to load, at line 4 we inspect the “content.txt” file whether to exist, if it dose not exist then inform user(line 21), else then output file’s content(line 4 to 19).

At source code line 25 to 29, we override “getStaticPageId” method, that means enable static web page function.

Next we show the content of “content.txt” file:

```

Shemin Dunne
Hirota, Yoshinobu
James Gao

```

So we visit readfile.dqm, the result is:

The following is the content which reads from content.txt:

```

Shemin Dunne
Hirota, Yoshinobu
James Gao

```

From the above result, we can see the program correctly output the content. Ok next we start to do the experiment, add a new line into “content.txt”, as follow showed revised content:

```

Shemin Dunne
Hirota, Yoshinobu
James Gao
Li Dong

```

Next we visit “readfile.dqm” again, but the result has not changed, why? Because “readfile.dqm” enable static web page function, but the system will not inspect the static page which earlier generated whether already expired, therefore, system will return the static page which earlier generated when visit this dqm again. How can we solve the problem? We can manually delete static web pages, like this system will re-created static web page when we visit dqm file again.

The dqm file “clearStaticPage.dqm” can operate the static page which already generated, it source code:

```
1  <% @page command="true"%>
2  <%
3  String password = "123456";
4  if (request.getParameter("B1") == null) {
5  %>
6
7  <form method="POST" action="">
8      <p align="center"><b><font size="5">The tool of remove static web
pages</font></b></p>
9      <p align="center">Operation password: <input type="password" name="psw"
size="20"></p>
10     <p align="center"><input type="radio" name="system" value="getSize"
checked>Return the number of static web pages in current host</p>
11     <p align="center"><input type="radio" name="system" value="clearAll">Removes
all of the static web page from current host</p>
12     <p align="center"><input type="radio" name="system" value="clear">Removes the
static page of the specified staticPageId from current host. The specified staticPageId is:
<input type="text" name="staticID" size="20"></p>
13     <p align="center"><input type="submit" value="Submit" name="B1"><input
type="reset" value="Reset" name="B2"></p>
14 </form>
15
16 <% }
17 else { %>
18
19 <p align="center"><a href="<%=request.getNowUrlFolder() +
request.getNowFile().getName()%>">Return</a></p>
20
21 <%
22 if (!password.equals(request.getParameter("psw"))) {
23     out.println("The operation password is wrong!!");
24     return;
25 }
26 //
```

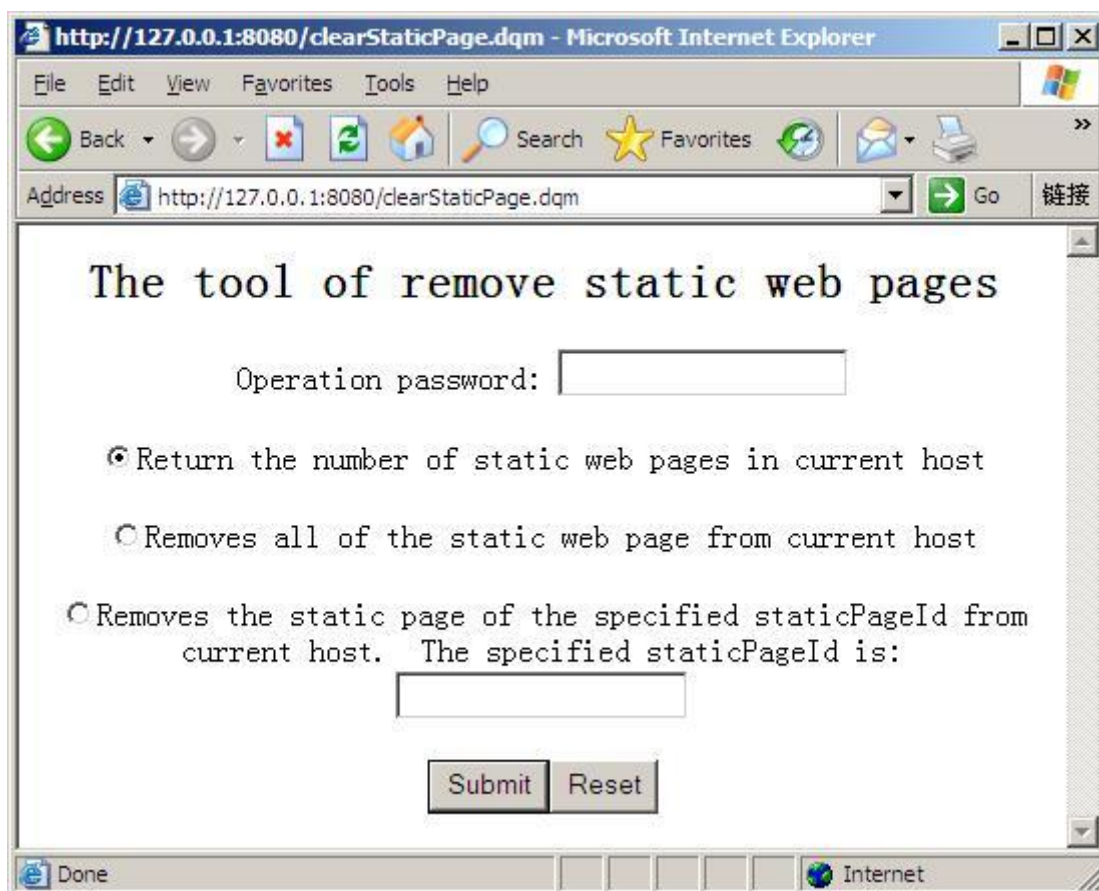


```

27 String system = request.getParameter("system");
28
29 if ("getSize".equals(system)) {
30     out.println("How many static web pages in the current host: " +
command.getStaticPageCount());
31     return;
32 }
33
34 if ("clearAll".equals(system)) {
35     command.clearAllStaticPage();
36     out.println("Removed all static web pages from current host.");
37     return;
38 }
39
40 if ("clear".equals(system)) {
41     if (command.clearStaticPage(request.getParameter("staticID"))) {
42         out.println("Removes the specified static web page from current host, successful
implementation.");
43     }
44     else {
45         out.println("Removes the specified static web page from current host, successful
failure.");
46     }
47     return;
48 }
49
50 %>
51
52 <% }%>

```

The interface of running the program as follows:



Picture 12-2-1

This program has three functions, user can use three buttons to choose the three functions, but must first input the correct operation password. In here password is “123456”, it’s defined by “password” variable (source code line 3), if you want to change password please change this variable’s value.

We introduce the first function (source code line: 29 to 32), it can get the quantity of current host's static web pages, this function was used “getStaticPageCount()” method, the result is quantity of registered static web pages in current host after executed this function.

If restart the server to ensure that current host does not contain any static web pages, then visit “readfile.dqm” file, next visit “clearStaticPage.dqm” file and use current function to check quantity of static web pages, you will see current has 1 static web page, it’s explained that static web page of “readfile.dqm” has been generated and registered. If this time we visit “readfile.dqm” again then system will directly to return static web page which already generated. User also can try to visit “mpage.dqm” (refer to section 12.1), then will find that quantity of static web page increased.

Ok next we introduce the second function (source code line: 34 to 38) it can remove all static web pages from current host, this function was used “clearAllStaticPage()” method.

We can try to visit “readfile.dqm” and “mpage.dqm”, then use first function to get current static web pages quantity, next we use current function to remove all static web pages, next if we use first function to get current static web pages quantity again, we will find current host has zero static web page. This shows that all of the static web pages have been deleted. Now if the content of “content.txt” has been revised, then we can use second function to manually remove old static web page.

Sometimes we just want to remove the specified static web page from current host, not need to remove all static web pages, this time we can use “clearStaticPage(String staticPageId)” method. The third function can remove the specified static web page (source code line: 40 to 48). We make a test, first we need to remove all static web pages in current host, then we visit “readfile.dqm” and “mpage.dqm”. Ok this time we can use first function to get quantity of static web pages, next we try to remove static web page of “readfile.dqm”, only input “/test_folder/readfile.dqm” into “The specified staticPageId” item. Ok finally we use first function to get current quantity of static web page again, will find a static web page was removed.

From the above presentation, we know how to renew the static web page which already expired, but need to manually renew, not automatic. If in the operating process will change the original static web page’s content then need to manually delete that original static web page. For example, the “a.dqm” will need to read datum from database “A” and display these datum. By the way “a.dqm” enable static web page function. The “b.dqm” will modify record of database “A”, then need to add operation of delete a.dqm’s static web page into “b.dqm”. The purpose of the to do this is to generate newest static page of “a.dqm” after modify record of database “A”.

We can remove the static page by specified staticPageId, but if there are too many static web pages then maybe we can’t remember all static IDs, therefore next we will introduce how to read all static IDs from current host.

4. public String[] getAllStaticID()

Read all static IDs from current host.

Returns: all static IDs from current host.

5. public String getStaticPageFileName(String staticPageId)

Get static web page’s full name by specified staticPageId from current host.

Parameters: Static ID whose associated static web page’s full name is to be returned

Returns: the static web page’s full name by specified static ID. Returns null if static ID not exist.

For example, “allStaticID.dqm” will display current host’s all static IDs and corresponding static web page’s name. It’s source code:

```
1  <% @page command="true"%>
2  <html>
3
4  <head>
5  <meta http-equiv="Content-Language" content="zh-cn">
6  <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
7  <title>Static ID</title>
```

```

8 </head>
9
10 <body>
11
12 <table border="0" width="100%" id="table1">
13     <tr>
14         <td bgcolor="#C0C0C0" width="275" align="center"><b>Static ID</b></td>
15         <td bgcolor="#C0C0C0" align="center"><b>Name of static web page</b></td>
16     </tr>
17     <%
18         String[] staticIds = command.getAllStaticID();
19         for (int i = 0; i < staticIds .length; i++) {
20             %>
21             <tr>
22                 <td width="275"><%=staticIds[i]%></td>
23                 <td><%=command.getPageFileName(staticIds[i])%></td>
24             </tr>
25             <% } %>
26
27 </table>
28
29 </body>
30
31 </html>

```

First we use “getAllStaticID” method to read all static IDs (source code line 19 to 25), next use static ID to get static web page file name (source code line 23, use “getPageFileName” method). Following is my results of this program:

| Static ID | Name of static web page |
|--|---|
| C:\kgserver\webapp\test_folder\mpage.dqm_page1 | C:\kgserver\webapp\WEB-INF\static_page\c74eee8f52fffe5ecb2456e543117739 |
| /test_folder/readfile.dqm | C:\kgserver\webapp\WEB-INF\static_page\f93a44cecf93190e897596d02a90c9f |
| C:\kgserver\webapp\test_folder\mpage.dqm_page2 | C:\kgserver\webapp\WEB-INF\static_page\a97b4e3a2999a0339436340e60a50fd9 |

Picture 12-2-2

From above picture we can seen there are three static ID and corresponding static web page’s name. But please note that static web page filename not include the extension.

Example, above picture first static web page file name is:

C:\kgserver\webapp\WEB-INF\static_page\c74eee8f52fffe5ecb2456e543117739

So the actual existence of the static page file should have three, there are:

C:\kgserver\webapp\WEB-INF\static_page\c74eee8f52fffe5ecb2456e543117739.gzip

C:\kgserver\webapp\WEB-INF\static_page\c74eee8f52fffe5ecb2456e543117739.deflate

C:\kgserver\webapp\WEB-INF\static_page\c74eee8f52fffe5ecb2456e543117739.no

User can find these files from above path, but because server installation directory different so different environmental results will be different.

Perhaps some users will ask that the usefulness of static web page filename, very simple we can directly delete corresponding three static web pages, it's effect is same to `clearStaticPage(String staticPageId)`.

12.3 The flow of static web page

This section we will be presenting the implementation process of static web pages, so users can better understand static web page function. We introduced override “`getStaticPageId`” method can return the static ID, if static ID which is null then mean disable static web page function. In fact, this has already summarized the static web page function flow, here we will detailed to explain this process again.

When the user requests a dqm file, server first will execute this method:

```
public String getStaticPageId(  
DRequestItface request, DSessionItface session, DApplicationItface application)
```

If this method return null then server consider static web page function disabled, so next execute dqm file and output result.

But if this method not return null then server consider static web page function enabled, so next corresponding flow also changed. First server will inspect the static ID whether already to register, if yes then next inspect the corresponding static web page whether to exist, only these 2 conditions were satisfied then server will directly output the static page. So this is the result of why delete static web page file can let the server re-generate static page.

If the above two conditions even one not be satisfied then corresponding flow will be changed. At this time server will continue to implement the dqm file which current visited, next output the result, but please note that server will register the static ID and save the result to corresponding three static web pages at same time. Then, when the user visit this dqm again server will directly output the static page by first flow.

By the way that all static web pages registration information will be lost after restarted server, so all static web pages will be re-generated and re-registered.

12.4 Matters needing attention

This section we will introduce static web page function matters needing attention.

In section 5.1 and 5.7 we introduced “redirect” and “option output” function, if the dqm file used these two functions then don't enable static web page function, otherwise might get incorrect results.

In section 5.4 we introduced “encodeURL” method, if the dqm file used this method then please don’t enable static web page, because maybe URL which in static web page file’s content will with initial sessionID.

If enable static web page function (return to non-null static ID), then even if you have disabled buffer function, server will force to enable the buffer function of current dqm file (buffer function please refer to section 3.2). Another point, even if you have enabled compress function, server will force to disable compress function of current dqm file when first generating static web page files (compress function please refer to section 4.3)

Under certain circumstances, the performance of static web page function maybe will lower than dqm file to be directly executed, therefore the user must measure it.

附录 1 DQM 容器介绍

F1.1 编译动态文件

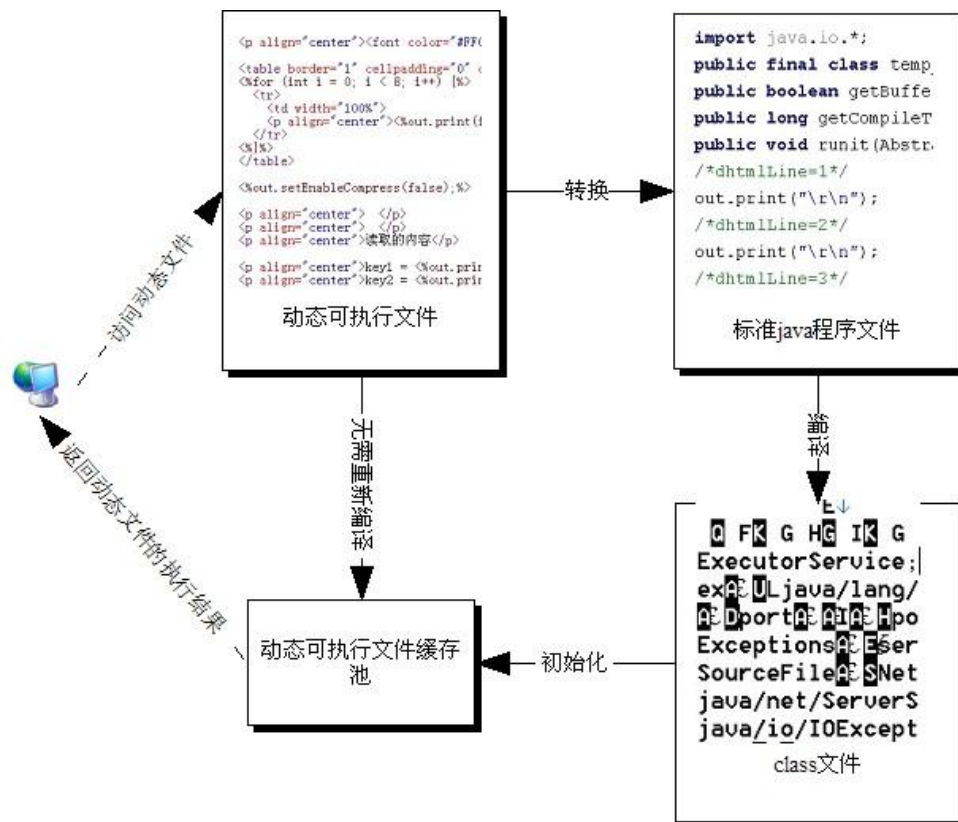


图 F1-1-1

DQM 容器类似于 JSP 容器，其主要作用就是用于执行动态文件（默认扩展名.dqm），因为是编译后执行，所以理所当然第一步是如何先编译动态文件。那么如何编译呢？因为动态文件是在 `html` 语言中嵌入 `dqm` 脚本语言所以编译的第一步是将动态文件转换成标准的 `java` 文件，转换后的 `java` 文件将保存在当前主机或虚拟主机的 `web` 根目录下的 `WEB-INF\work\com\kangaroo_egg\workfile` 目录中。当前主机或虚拟主机的 `web` 根目录请参见 2.3 和 2.5 节的 `webPath` 项。第二步是将刚才已经转换且保存的 `java` 文件编译成 `class` 文件，编译后的 `class` 文件保存在与 `java` 文件的同一目录下。编译后的 `class` 就可以被 `java` 虚拟机执行了。介绍了编译动态文件过程，那么服务器会在什么时候编译动态文件呢？在下一节介绍流程中就会有所介绍。

F1.2 DQM 容器流程

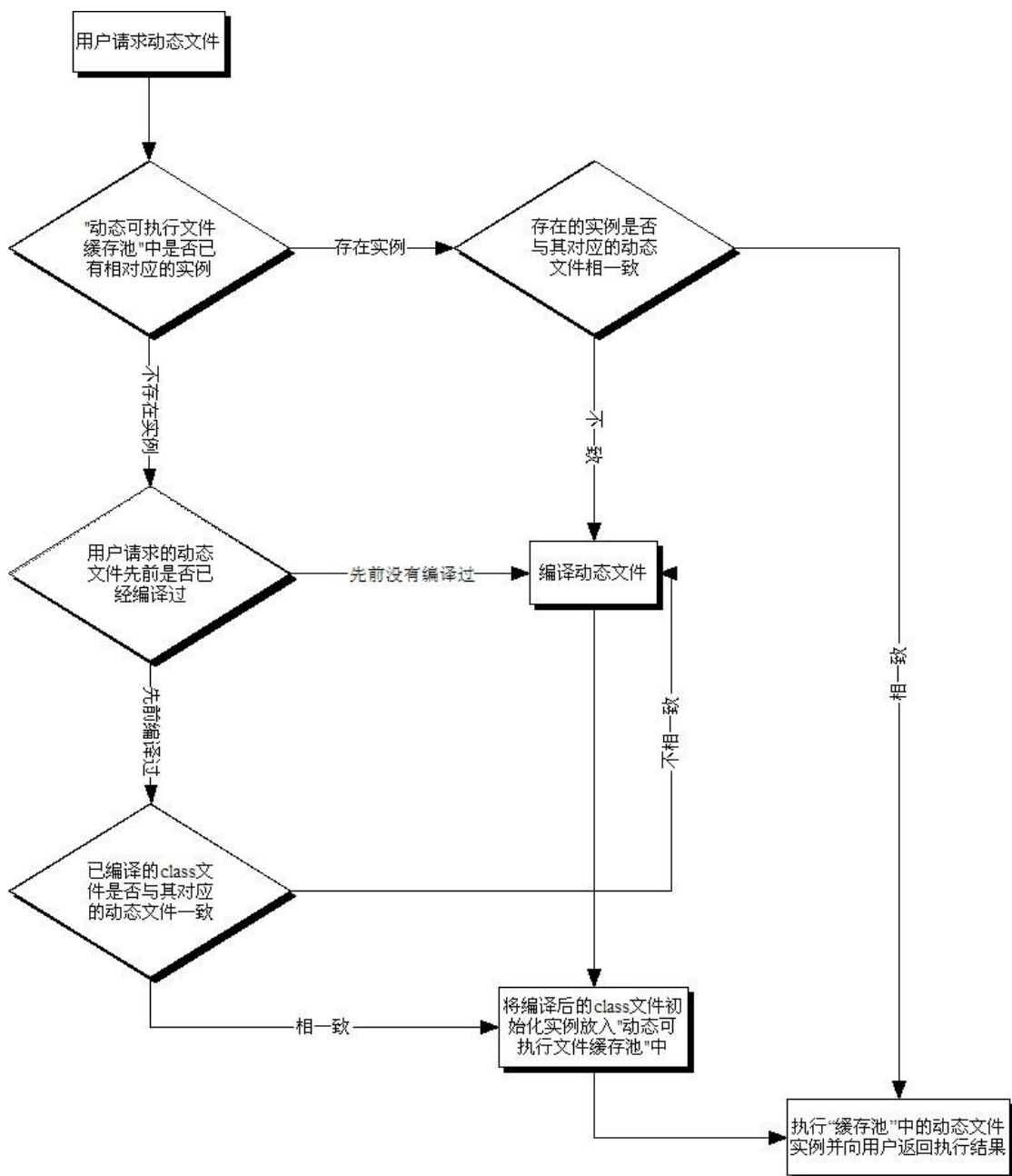


图 F1-2-1

当用户访问一个动态文件时 DQM 容器首先去检查“动态可执行文件缓存池”中是否已有相对应的实例，如果有的话则检查实例是否与其对应的动态文件一致，如果一致的就执行实例返回结果，如果不一致（比如动态文件已经修改了，而实例是动态文件修改前的）则如上一节所说的进行编译动态文件，再将编译后的 class 文件初始化实例放入“动态可执行文件缓存池”中执行返回结果。

“动态可执行文件缓存池”中如没有动态文件相对应的实例的话则判断此动态文件是否已经被编译过（相对应的目录中存在已编译过的 class 文件），如果编译过则检查已编译的 class 文件是否与其对应的动态文件一致，如果一致则将已编译的 class 文件初始化实例放入

“缓存池”中执行返回结果，如果不一致则进行编译而后初始化实例放入“缓存池”执行返回结果。如果动态文件没有被编译过（相对应目录中不存在已编译的 class 文件）则进行编译而后初始化实例放入“缓存池”执行返回结果。

注意：通常情况下如果修改了动态文件，DQM 容器会重新编译动态文件，并把编译生成的新 class 文件覆盖 WEB-INF\work\com\kangaroo_egg\workfile 目录下原来的旧文件。在少数情况下更新动态文件后还是看到旧的内容，请删除 work 目录下相关的文件，这样能够确保重新编译。

F1.3 关闭自动编译时 DQM 容器流程

webconfig.xml 中主机和虚拟主机有一个配置项 autoCompile（参见 2.3 和 2.5 节），此项主要用于是否允许服务器自动编译动态可执行文件。为什么需要这个功能呢？一个原因是编译动态文件很占用系统资源，所以在正式运行的服务器上可以禁止对动态文件编译，用户可以在本地计算机预先编译好后再将编译后的 class 文件上传至服务器相应目录（web 根目录下的 WEB-INF\work\com\kangaroo_egg\workfile 目录中）。另一个原因是如果开启了自动编译则服务器就会监视动态文件有无修改，如果修改了则先前编译的 class 文件就与修改后的动态文件不一致了，所以必须重新编译，在开发阶段这是合理的，因为要不断修改动态文件且查看修改后结果，但是在运行阶段动态文件很少或者几乎不改变，于是就无需消耗资源监视动态文件是否改变从而重新编译。我们来看一下关闭自动编译后的流程，您可以与未关闭前的流程做比较。

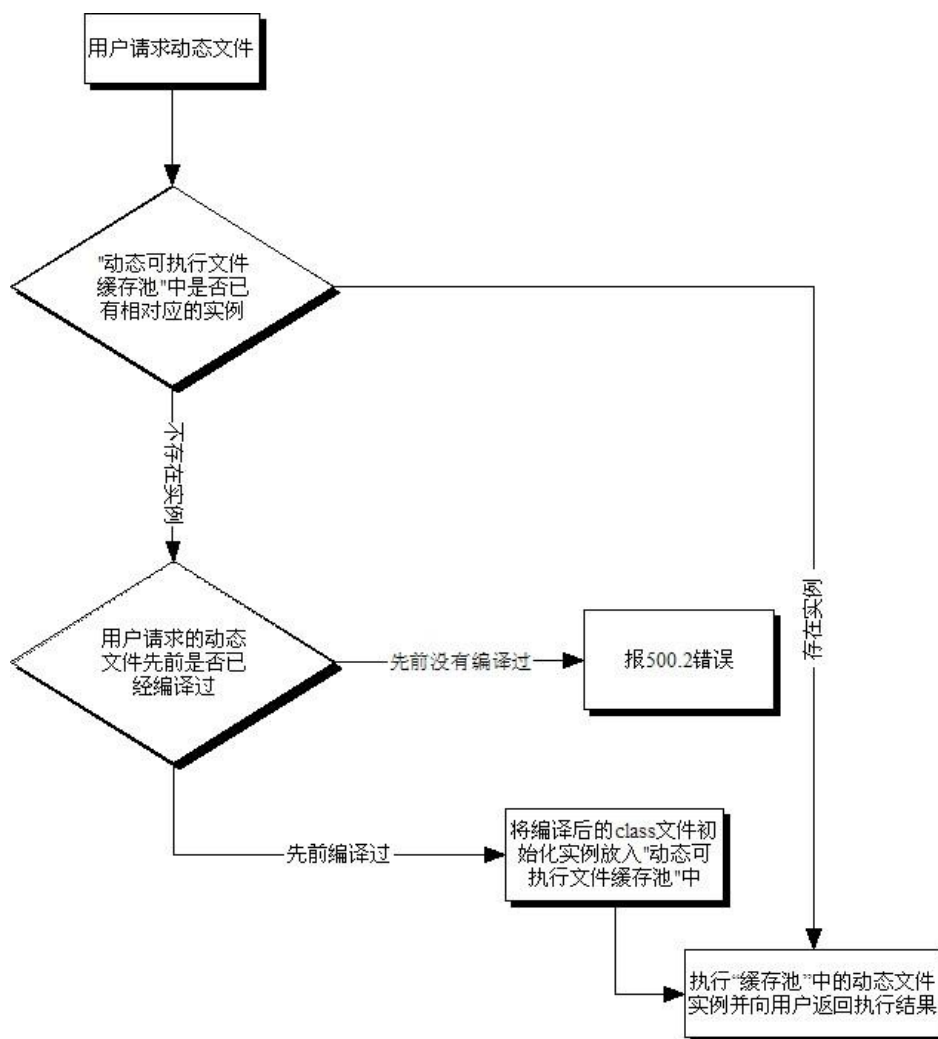


图 F1-3-1

从流程图中看出此时当用户访问一个动态文件时，DQM 容器首先去检查“动态可执行文件缓存池”中是否已有相对应的实例，如果有的则不会再检查实例是否与其对应的动态文件一致而直接执行“缓存池”中实例且返回结果。

如“动态可执行文件缓存池”中没有动态文件相对应实例的话，则判断此动态文件是否已经被编译过（即相对应的目录中存在已编译过的 class 文件），如果编译过则不会再检查已编译的 class 文件是否与其对应的动态文件一致，而是直接将已编译的 class 文件初始化实例后放入“缓存池”中执行返回结果。

如果动态文件没有被编译过（即相对应的目录中不存在已编译的 class 文件）则报 500.2 错误，提示所访问的动态文件需要编译后才能执行，而目前服务器禁止编译。

细心的读者在看完上面的流程后可能会有二个疑惑，首先只要“动态可执行文件缓存池”中有相对应的实例则直接执行而不管实例是否与其对应的动态文件一致，那么即使修改了动态文件和动态文件对应的 class 文件，而“缓存池”中含有相对应的实例还是旧的，不过即使“缓存池”中实例是旧的过期的可是还是会直接执行，那么岂不是修改后的动态文件永远不会被执行？可是有时候我们还是会对已运行 web 应用进行一些小的修改，那么在关闭了自动编译的这时候该怎么办呢？其实无非就是将“缓存池”中的这个实例清除，只要重启服务器则“缓存池”中的所有实例就清除了，不过运行中的服务器是不太易于重新启动的，而且重启后“缓存池”中的实例都没有了，那些原本就无需改动的动态文件也必须重新实例化

放入缓存池中，因此增加了无谓的开销。于是有更好的方法便是前面 10.3 节提到过的。

首先将修改后的动态文件和编译后的文件覆盖原文件（编译后的文件在 web 根目录下的 WEB-INF\work\com\kangaroo_egg\workfile 目录中），然后用 10.3 节的方法将“动态可执行文件缓存池”中此动态文件实例删除，这样的话当再次访问此动态文件时服务器会发现“缓存池”中没有相对应实例，于是判断此动态文件是否已经被编译过，因为我们刚才已经将重新编译的文件覆盖了原来的，所以服务器发现动态文件已编译且将编译后的 class 文件初始化放入“缓存池”中执行返回结果，而此时载入的就是已经重新编译过的 class 文件了，于是就会返回修改后的结果。其实最主要是覆盖编译后的 class 文件，而修改后的动态文件覆盖不覆盖原文件是无所谓的，为什么呢？前面流程中已经介绍过了当初始化 class 文件时不会检测 class 文件是否与其对应的动态文件一致，而是直接将已编译的 class 文件初始化实例后放入“缓存池”中执行返回结果。所以只要已编译的 class 文件修改了就可以了。不过为了保持一致还是最好同时覆盖动态文件。

如果要清除类缓存也请参阅前面 10.4 节。

另外一个疑惑是用户访问的动态文件对应的实例在“缓存池”中不存在，同时此动态文件也没有编译过（即此动态文件编译后的 class 文件不存在），则服务器会报 500.2 错误，为什么会报这个错误呢？因为此时自动编译功能关闭，所以在既无缓存实例又无编译过的动态文件时当然无法执行，那么此时该怎么做呢？只要先在其它机器上用打开自动编译的 kangaroo-egg 服务器编译（F3.1 节也介绍了编译工具用于编译动态文件），然后将编译后的文件上传至那台关闭自动编译的服务器上，这样再次访问此动态文件时 DQM 容器就会将编译后的 class 初始化放入缓存池执行。至于编译后的 class 文件存放目录在 web 根目录下的 WEB-INF\work\com\kangaroo_egg\workfile 目录中。

附录 2 国际化问题

目前很多网站都需要同时提供多种语言，以供不同地区的用户访问。kangaroo-egg 从一开始就考虑到这样的国际化问题，为此提供了适用于国际化的方法。在介绍如何编写国际化的网站前有必要了解一下编译时的字符集。

F2.1 编译和执行时的字符集

附录 1 详细介绍了动态文件编译原理，本节则补充介绍一下编译时字符集。如图 F1-1-1 所示编译时动态文件首先要转换成标准的 java 程序文件，那么在转换过程中又是采用何种字符集呢？其实转换过程中采用是由 dataEnc（参见 2.1 节）所指定的字符集。比如 dataEnc 所指定的字符集为 GBK，则 DQM 容器首先按 GBK 字符集读取动态文件，按 GBK 字符集转换完后再按 GBK 字符集保存转化完后的标准 java 程序文件。但是如果此时动态文件中含有超出 GBK 字符则在读取动态文件时就会将那些超出的字符转换成问号，所以最终编译后执行的结果也是那些超出 GBK 字符集的字符变成了问号。

我们来做个实验，首先将 webconfig.xml 的 systemSet 元素中的 dataEnc 值改为 US-ASCII，这样在编译过程中只能识别英文字符，而所有的中文字符将被转换成问号。

如源程序如下的动态文件：

```
1 <pre>
2 <%= "你好"%>
3 <%= "Hello"%>
4 </pre>
```

在 dataEnc 值为 US-ASCII 的情况下编译后执行结果为：

```
????
Hello
```

可以看到上面的执行结果中原本应该显示“你好”字符，可是显示了四个问号。

而在 dataEnc 值为 GBK 的情况下编译后执行结果为：

```
你好
Hello
```

到这里又有一个疑问，如果编译的时候 dataEnc 指定的字符集为 GBK，而执行的时候再将 dataEnc 指定的字符集改称 US-ASCII，那么执行结果会是怎么样的呢？其实也是会变成问号的，为什么呢，4.1 节中介绍过 print(String content)方法也是按 dataEnc 指定的字符集输出的，所以即使编译的时候是 GBK 字符集，然而在执行时改为了 US-ASCII，则在执行

时其实就是用 `out.print("你好", "US-ASCII")` 这个方法输出，中文字符强行被转换成英文字符自然变成了问号。

所以如果动态文件源代码如下：

```
1 <pre>
2 <%out.print("你好", "GBK");%>
3 <%= "Hello"%>
4 </pre>
```

那么只要在编译的时候 `dataEnc` 指定的字符集为 `GBK`，即使在执行时 `dataEnc` 指定的字符集为 `US-ASCII`（或者其他任何字符集）也能正确显示“你好”。

可是在运行需要多国语言支持的网站时也不可能将 `dataEnc` 的值改来改去，也只能指定一种字符集，于是我们假定我们将 `dataEnc` 指定在 `US-ASCII` 上，那么如何才能编写出支持中文的动态文件呢？为了能使在 `US-ASCII` 字符集下也能正确编译出中文字符我们可以将中文字符变成 `unicode` 表示格式，当然不只中文字符，任何非 `ASCII` 字符都应改为它们等价的 `unicode` 表示格式。比如“你”的 `unicode` 表示格式为“`\u4f60`”，而“好”的 `unicode` 表示格式为“`\u597d`”，于是动态文件源程序就改成了如下：

```
1 <pre>
2 <%out.print("\u4f60\u597d", "GBK");%>
3 <%= "Hello"%>
4 </pre>
```

这样不管 `dataEnc` 指定的是何种字符集，编译和执行都可以显示出正确的“你好”。不过我怎么知道哪些字符应该转换成 `unicode` 格式呢，毕竟有些字符是不需要转换的，同时我该如何转换呢？`JDK` 本身提供了一个很好的工具 `native2ascii`，利用这个工具可以很容易转换。它的使用方法为：

`native2ascii a.dqm b.dqm`

`a.dqm` 是指包含普通字符的文件，`b.dqm` 是指将 `a.dqm` 中可以转换的字符转换成 `unicode` 格式后保存的文件。利用这个工具就可以轻松将字符转换成 `unicode` 格式了。

你也可以用 `-reverse` 选项来进行逆向转换：

`native2ascii -reverse b.dqm a.dqm`

并且可以用 `-encoding` 选项来设定另一种编码：

`native2ascii -encoding BIG5 a.dqm b.dqm`

F2.2 读取客户端请求时的字符集

当客户端向服务器发出请求时，服务器必须首先读取客户端请求数据，之后才能给与回应，于是问题就来了，客户端种类很多而且还分地区，比如中国大陆地区的某些浏览器发送请求的数据中如含有中文字符，则这些中文字符是按 `GBK` 字符集编码后发送的，那么服务器必须以 `GBK` 字符集读取才能准确获得请求中的中文字符，那么台湾地区呢？含有繁体中文字符的请求会按 `BIG-5` 字符集编码后发送，那么作为服务器必须以 `BIG-5` 字符集读取才能准确获得请求中的繁体中文字符。那么 `kangaroo-egg` 服务器在读取客户请求时是使用哪种

字符集呢？是使用 webconfig.xml 的 systemSet 元素中的 urlEnc 值指定的字符集来读取的（urlEnc 参见 2.1 节），而 urlEnc 只能设定一种字符集，所以不可能同时能够读取多种字符集。那么岂不是无法支持国际化了？其实 HTTP 协议也考虑到了这个问题，为此建议客户端在发出请求的 HTTP 头中如含有非 ASCII 字符，则将这些字符按 UTF-8 字符集进行 URL 编码（URL 编码请参见 2.1 节的 urlEnc 项）。可是 HTTP 协议只是建议这样，各种客户端并不需要完全遵循，比如中文版的 IE 默认就会将非 ASCII 字符按 UTF-8 字符集进行 URL 编码后发送，但中文版的 FireFox 默认会将非 ASCII 字符按 GBK 字符集进行 URL 编码后发送，而有些客户端比如下载工具更本就不进行 URL 编码，直接将非 ASCII 字符以某种字符集编码后发送。那么我们该怎么办呢？幸好客户端一般不会发送含有非 ASCII 字符的请求信息，那么我们就来看看什么时候会发送这些非 ASCII 字符呢？通常有以下几种情况：

1. 当请求一个非 ASCII 字符的资源时。比如用户在服务器上有“file 你好.dqm”这个文件，那么当访问这个文件时客户端就会含有中文字符，如果此时你的 urlEnc 值为 GBK 字符集那么是没有问题的，可是服务器上如果还有 BIG5 的繁体中文的文件呢？这个时候该怎么办，你不可能为 urlEnc 同时设置多个字符集。为此请在国际化的环境中使用只包含 ASCII 字符的文件名。

2. 服务器端给客户端设置的信息中含有非 ASCII 字符，并且以后还需要从客户端读取的。比如 cookie 就是一个例子（cookie 请参见第 7 章），当向客户端写入一个 cookie，那么以后客户端就会将这个 cookie 内容包含在其请求中以便服务器端再次读取。如果向客户端写入非 ASCII 字符，那么服务器端读取客户端返回的这些非 ASCII 字符就有可能遇到问题，因为国际化有可能需要同时向客户端写入中文、韩文等，但是服务器端只能以一种字符集读取客户端请求中的字符。那么我们该如何进行类似于 cookie 的国际化读写呢？还记得 2.1 节的 urlEnc 项中提到的 URL 编码吗？URL 编码可以很容易将非 ASCII 字符编码成只含有 ASCII 字符的形式，这样不管服务器端设置为哪种字符集读取请求都没问题，因为所有字符集都能够读取 ASCII 字符。我们来举个例子，首先将 urlEnc 设置为 Shift_JIS，即日文字符集，至于 dataEnc 你可任意设置为一种字符集，接下来所要做的就是写一个含有中文字符的 cookie，之后再读取这个 cookie 的值。因为 urlEnc 已经设置为日文字符集，所以为了要读取中文字符我们就必须先 will 中文字符先 url 编码，再 url 解码读取。

首先看写入 cookie 的动态文件 write_nonascii_cookie.dqm，源程序如下：

```
1 | <% @ page buffer="true"%>
2 | <%
3 |   DCookie ck=new DCookie("ourset", java.net.URLEncoder.encode("\u4f60\u597d",
   |   "GBK"));
4 |   ck.setMaxAge(30*60); // 设置 Cookie 的存活时间为 30 分钟
5 |   response.addCookie(ck); // 写入客户端硬盘
6 |   out.print("Write cookie ok!");
7 | %>
```

从上面第 3 行可以看到 jdk 本身就提供了 url 编码的静态方法 java.net.URLEncoder.encode，此方法有 2 个参数，第一个是所要进行编码的字符串，这里我们是对\u4f60\u597d这个字符串进行 url 编码（\u4f60\u597d是“你好”的 unicode 表示格式）。第 2 个参数是对第一个参数采用哪种字符集进行 url 编码，我们看到这里是用 GBK 字符集。

当写入 cookie 后我们所要做的就是再正常读取此 cookie，read_nonascii_cookie.dqm 就是

用来读取 cookie 的动态文件，源程序如下：

```
1 <%String charset = "GBK";%>
2 <html>
3 <head><title>
4 read non ascii cookie
5 </title>
6 <meta http-equiv=Content-Type content="text/html; charset=<%=charset%>">
7 </head>
8
9 <%
10 String value =request.getCookieValue("ourset");
11
12 if (value == null) {
13     out.println("cookie not found!");
14 }
15 else {
16     out.println("cookie value: " + java.net.URLDecoder.decode(value, charset), charset);
17 }
18 %></html>
```

从上面第 16 行可以看到 jdk 也提供了 url 解码的静态方法 `java.net.URLDecoder.decode`，此方法也有 2 个参数，第一个是所要进行解码的字符串，这里我们是对读取的 cookie 值进行 url 解码。第 2 个参数是对第一个参数采用哪种字符集进行 url 解码，这里当然还是用 GBK 字符集，因为 url 编码的时候是采用 GBK 字符集的。

当读取 cookie 程序执行后你就可以看到结果是：

```
cookie value: 你好
```

从上面的结果看到中文字符被正确的读取，而这是在 `urlEnc` 设置为 `Shift_JIS` 字符集的情况下。你可以试着将 `write_nonascii_cookie.dqm` 中的 url 编码和解码语句去掉，运行的话就会发现结果是乱码。

注意：在初始化 cookie 时需要设置 2 个参数，cookie 的名字和其对应的值，从上面例子中看到我们只对 cookie 的值进行了 url 编码，因为值中含有中文字符。那么我们是否也能将 cookie 的名字设置为含有中文等非 ASCII 的字符呢？如果这样做的话当根据名字来读取 cookie 的值时你会发现很多问题，所以我们推荐对于 cookie 的名字只包含 ASCII 字符。

3. 请求资源时附带的数据含有非 ASCII 字符。我们知道 url 可以附带数据，比如这个 <http://localhost:8080/getName.dqm?name=dunne>，其中问号后面的“name=dunne”就是附带的数据，这种传输数据的方式通常称之为 GET 方式，可以通过 6.1 节介绍的方法来获取。通过这种方式上传的数据是包含在 HTTP 请求头中的，而前面已经介绍过服务器读取 HTTP 头所在用的字符集是由 `urlEnc` 指定的，于是老问题又来了，`urlEnc` 只能指定一种字符集，而 GET 方式在国际化时有可能需要同时传输中文、韩文等非 ASCII 字符，那该怎么办，哈哈聪明的你一定想到了用前面介绍的 url 编码，再来个例子，就像前面一样首先将 `urlEnc` 设

置为 Shift_JIS, dataEnc 你任意, 于是我们所要做的就是 在 url 中附带中文字符数据, 看看能否被正确读取。

首先看写一个附带中文数据的链接, write_nonascii_url.dqm 源程序如下:

```
1 <html>
2 <body>
3 <a
  href="read_nonascii_url.dqm?name=<%=java.net.URLEncoder.encode("\u888b\u9f20\u86cb", "GBK");%>">
4 <%out.print("\u53d1\u9001url\u9644\u5e26\u7684\u6570\u636e", "GBK");%>
5 </a>
6 </body>
7 </html>
```

从上面第 3 行可以看到我们又用了 java.net.URLEncoder.encode 这个方法, 这里我们是对 \u888b\u9f20\u86cb 这个字符串进行 url 编码 (\u888b\u9f20\u86cb 是“袋鼠蛋”的 unicode 表示格式)。\u53d1\u9001url\u9644\u5e26\u7684\u6570\u636e 是“发送 url 附带的数据”的 unicode 表示格式。

当打开 write_nonascii_url.dqm 后单击链接后就会指向 read_nonascii_url.dqm, 而这个动态文件就是用来读取 url 附带的数据, 源程序如下:

```
1 <%String charset = "GBK";%>
2 <html>
3 <head><title>
4 read non ascii url
5 </title>
6 <meta http-equiv=Content-Type content="text/html; charset=<%=charset%;%>">
7 </head>
8
9 <%
10 String value =request.getParameter("name", charset);
11
12 if (value == null) {
13     out.println("name not found!");
14 }
15 else {
16     out.println("name value: " + value, charset);
17 }
18 %></html>
```

从上面第 10 行可以看到用 request 的 getParameter 带字符集参数的方法就可以直接读取 url 编码的数据了, 这里我们当然还是采用 GBK 字符集来读取。

执行后你就可以看到结果是:

```
name value: 袋鼠蛋
```

从上面的结果看到中文字符被正确的读取, 而这是在 urlEnc 设置为 Shift_JIS 字符集的

情况下。你可以试着将 `write_nonascii_url.dqm` 中的 `url` 编码语句去掉，运行的话就会发现结果是乱码。

F2.3 读取数据时的字符集

在上一节中第 3 点已经介绍了通过 `url` 附带数据的方法，这种传输数据的方式通常称之为 `GET` 方式，然而对于量比较大的数据则可以通过 `POST` 方式上传，本节我们将介绍 `POST` 方式的国际化问题。

国际化的客户端通过 `POST` 方式可以上传多种字符集的数据，服务器必须以其对应的字符集读取才能准确获得字符。那么 `kangaroo-egg` 服务器在读取 `POST` 方式的数据时是使用哪种字符集呢？是使用 `webconfig.xml` 的 `systemSet` 元素中的 `dataEnc` 值指定的字符集来读取的（`dataEnc` 参见 2.1 节），就像上一节也遇到过的，`dataEnc` 只能设定一种字符集，所以不可能同时能够读取多种字符集，不过我们可以使用带指定字符集的方法读取。

我们来看个例子，首先将 `dataEnc` 设置为 `US-ASCII`，即英文字符集，至于 `urlEnc` 你可任意设置为一种字符集，然后运行 6.1 节曾使用过的 `post.htm` 和 `post.dqm`，你会发现 `post.htm` 没有问题，而当通过 `post.htm` 向 `post.dqm` 传送数据时 `post.dqm` 显示的是乱码，首先我们按 F2.1 节介绍的方法将所有中文字符变成 `unicode` 形式后按 `GBK` 字符集输出，改变后的 `post.dqm` 源代码如下：

```
1  <%@page import="java.util.*"%>
2  <pre><%
3  String charset = "GBK";
4  //读取所有非多分数数据项的名称
5  Set nameSet = request.getParameterNameSet();
6  if (nameSet.size() == 0) {
7
8  out.println("\u6ca1\u6709\u4e0a\u4f20\u4efb\u4f55\u975e\u591a\u5206\u6570\u636e\u5b9a\u4e4e", "charset");
9  }
10 else {
11     Iterator<String> nameItr = nameSet.iterator();
12     while (nameItr.hasNext()) {
13         String tempName = nameItr.next();
14         out.println("-----");
15         out.println("\u4e0a\u4f20\u975e\u591a\u5206\u9879\u540d\u79f0\u5b9a\u4e4e" +
16 tempName, charset);
17
18         //getParameter 方法
19         out.println("\u672c\u975e\u591a\u5206\u9879\u7684\u503c\u5b9a\u4e4e" +
20 request.getParameter(tempName), charset);
21
22         //getParameterValues 方法
```

```

20     String tempValues[] = request.getParameterValues(tempName);

21     out.println("\u672c\u975e\u591a\u5206\u9879\u603b\u5171\u6709\u51e0\u4e2a\u503c\u51fa" + tempValues.length, charset);
22     out.print("\u672c\u975e\u591a\u5206\u9879\u7684\u6240\u6709\u503c\u51fa",
charset);
23     for (int i = 0; i < tempValues.length; i++) {
24         out.print(tempValues[i] + ", ", charset);
25     }
26     out.println("");
27 }
28 }
29
30 %></pre>

```

再次通过 post.htm 向 post.dqm 传送数据时你会发现信息字符已经没有问题了，但是如果传送的是中文则显示为问号，英文就没有问题。比如我们在 post.htm 中 key1 栏中写“data”，在 key2 栏中写“数据”，那么执行结果如下：

```

-----
上传非多项名称: key1
本非多项的值: data
本非多项总共有几个值: 2
本非多项的所有值: data, ????,

-----
上传非多项名称: B1
本非多项的值: ???
本非多项总共有几个值: 1
本非多项的所有值: ???,

-----
上传非多项名称: key2
本非多项的值: kangaroo-egg
本非多项总共有几个值: 1
本非多项的所有值: kangaroo-egg,

```

从上面结果可以看到，英文的“data”正常显示了，而中文的“数据”变成了四个问号，为什么呢？因为 request.getParameter 方法如果不指定字符集则按 dataEnc 设置的字符集读取，而 dataEnc 此时被我们设置为了 US-ASCII，所以当然无法正常读取中文字符，于是我们要使用 request.getParameter 带有字符集的方法（request.getParameter 方法参见 6.1 节），post.dqm 的源程序在前面的基础上再修改如下：

```

1  <%@page import="java.util.*"%>
2  <pre><%
3  String charset = "GBK";
4  //读取所有非多项数据项的名称

```

```

5 Set nameSet = request.getParameterNameSet();
6 if (nameSet.size() == 0) {
7     out.println("\u6ca1\u6709\u4e0a\u4f20\u4efb\u4f55\u975e\u591a\u5206\u6570\u636e\u53c3\u6570", "charset");
8 }
9 else {
10     Iterator<String> nameItr = nameSet.iterator();
11     while (nameItr.hasNext()) {
12         String tempName = nameItr.next();
13         out.println("-----");
14         out.println("\u4e0a\u4f20\u975e\u591a\u5206\u9879\u540d\u79f0\u53c3\u6570" +
tempName, charset);
15
16         //getParameter 方法
17         out.println("\u672c\u975e\u591a\u5206\u9879\u7684\u53c3\u6570" +
request.getParameter(tempName, charset), charset);
18
19         //getParameterValues 方法
20         String tempValues[] = request.getParameterValues(tempName, charset);
21         out.println("\u672c\u975e\u591a\u5206\u9879\u603b\u5171\u6709\u51e0\u4e2a\u53c3\u6570" + tempValues.length, charset);
22         out.print("\u672c\u975e\u591a\u5206\u9879\u7684\u6240\u6709\u53c3\u6570",
charset);
23         for (int i = 0; i < tempValues.length; i++) {
24             out.print(tempValues[i] + ", ", charset);
25         }
26         out.println("");
27     }
28 }
29
30 %></pre>

```

再次执行你会发现中文字符已经没有任何问题了。从上面源代码红色部分看出，我们在读取数据时使用了 `request` 的 `getParameter` 和 `getParameterValues` 带字符集参数的方法。在前面 6.7 中介绍的多分数据中也有指定字符集的读取方法，用法和功能也是相同的，这里就不再复述了。

F2.4 设置 http 头时的字符集

前面 5.2 节中曾介绍 `response.setHeader` 方法，其中还举了一个 `httphead.dqm` 的例子，当执行 `httphead.dqm` 后浏览器会提示下载“http 压缩.txt”这个文件，不过并不是所有情况下都会如此正常执行，现在我们就将 `webconfig.xml` 的 `systemSet` 元素中的 `dataEnc` 值改为 US-ASCII (`dataEnc` 参见 2.1 节)，再次访问 `httphead.dqm` 后浏览器还是会提示下载文件，只不过提示下载的文件名不是“http 压缩.txt”，而是变成了其它乱码，这是为什么呢？5.2 节中已经提到过了 `setHeader(String tag, String value)` 方法将会采用 `systemSet` 元素中的 `dataEnc` 的值进行编码，而现在 `dataEnc` 的值被改成了 US-ASCII，所以当执行

`response.setHeader("Content-Disposition", "attachment;filename=http 压缩.txt")`

这个方法时，所有的中文字符都会按 US-ASCII 字符集输出，这样中文字符变成了乱码，所以浏览器提示下载的文件名也变成了乱码。

那么在提供多种语言浏览时该怎么办呢？`systemSet` 元素中的 `dataEnc` 值只能设置为一种字符集，于是我们就要用到 5.2 节的介绍到的

`setHeader(String tag, String value, String charsetName)`

方法，这个方法能在输出 http 头时不默认采用 `dataEnc` 值编码，而是由用户指定编码字符集。

于是将 5.2 节中的 `httphead.dqm` 修改成：

```
1 <% @page buffer="true"%>
2 <%response.setHeader("Content-Disposition", "attachment;filename=http 压 缩 .txt",
   "GBK");%>
3
4 什么是 HTTP 压缩？
5
6 HTTP 压缩（或叫 HTTP 内容编码）作为一种网站和网页相关的标准，存在已久了，
   只是最近几年才引起大家的注意。HTTP 压缩的基本概念就是采用标准的 gzip 压缩或者
   deflate 编码方法，来处理 HTTP 响应，在网页内容发送到网络上之前对源数据进行压缩。
   有趣的是，在版本 4 的 IE 和 NetScape 中就早已支持这个技术，但是很少有网站真正使用它。
   Port80 软件公司做的一项调查显示，财富 1000 强中少于 5% 的企业网站在服务器端采用了
   HTTP 压缩技术。不过，在具有领导地位的网站，如 Google、Amazon、和 Yahoo!等，HTTP 内容
   编码技术却是普遍被使用的。考虑到这种技术会给大型的网站们带来带宽上的极大节省，用于
   突破传统的系统管理员都会积极探索并家以使用 HTTP 压缩技术。
```

于是无论 `dataEnc` 设置为何值，此条 http 头都会按 GBK 字符集输出，不过这样是不是就没有问题了呢？的确在某些情况下还是有问题的（请参见 F2.1 节），那么怎么办呢，最保险的方法就是将非 ASCII 字符都改为它们等价的 unicode 表示格式，在这里“压缩”的 unicode 表示格式为“\u538b\u7f29”（如何转换请参见 F2.1 节），于是我们修改 `httphead.dqm` 为：

```
1 <% @page buffer="true"%>
2 <%response.setHeader("Content-Disposition",
   "attachment;filename=http\u538b\u7f29.txt", "GBK");%>
3
```

4 什么是 HTTP 压缩?

5

6 HTTP 压缩（或叫 HTTP 内容编码）作为一种网站和网页相关的标准，存在已久了，只是最近几年才引起大家的注意。HTTP 压缩的基本概念就是采用标准的 gzip 压缩或者 deflate 编码方法，来处理 HTTP 响应，在网页内容发送到网络上之前对源数据进行压缩。有趣的是，在版本 4 的 IE 和 NetScape 中就早已支持这个技术，但是很少有网站真正使用它。Port80 软件公司做的一项调查显示，财富 1000 强中少于 5% 的企业网站在服务器端采用了 HTTP 压缩技术。不过，在具有领导地位的网站，如 Google、Amazon、和 Yahoo! 等，HTTP 内容编码技术却是普遍被使用的。考虑到这种技术会给大型的网站们带来带宽上的极大节省，用于突破传统的系统管理员都会积极探索并家以使用 HTTP 压缩技术。

这样就能确保浏览器提示下载文件名是正确的。同时 5.7 节所介绍的

```
public void outBinFile(File out_bin_file, String saveAsNameStr, String charsetName)
```

方法也是使用这个原理。

附录 3 重新编译和隐藏源代码

F3.1 重新编译动态文件

前面附录 1 已经详细介绍了动态文件执行的流程，动态文件必须先转换成 java 文件再编译后才能执行，但是编译动态文件很耗资源，为此提供了 autoCompile 这个设置（参见 2.3 和 2.5 节）。当关闭了 autoCompile 时就必须手动上传已编译的 class 文件了（附录 1 中详细介绍了关闭 autoCompile 时的流程），于是问题就来了当一个 web 应用很大，有很多动态文件，要将这些动态文件编译成 class 文件的方法就是在其它开启 autoCompile 的服务器上逐个访问动态文件。这似乎很麻烦，于是服务器提供了一个重新编译某个主机或虚拟主机的工具。

工具类全名为 com.kangaroo_egg.tools.Rebuild，如果服务器安装在 c:\webserver 目录下则工具类位于 c:\webserver\classes\com\kangaroo_egg\tools\Rebuild.class。在 classes 目录下会看到 rebuild.bat 和 rebuild.sh，其中 rebuild.bat 是用于在 windows 下重新编译的脚本命令，而 rebuild.sh 是用于在 redhat linux 和 Solaris 下的脚本命令。不过现在还不能执行，首先必须配置一下这二个脚本命令。

1. 在 windows 下运行则打开 rebuild.bat 文件，你会看到如下内容：



```
XXX\bin\java -cp .;XXX\lib\tools.jar com.kangaroo_egg.tools.Rebuild
```

图 F3-1-1

注意红框内的 XXX，请将此 XXX 换成 JDK 安装的目录，比如你的 JDK 安装在 c:\jdk1.5 下面则替换成如下内容：



```
c:\jdk1.5\bin\java -cp .;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.tools.Rebuild
```

图 F3-1-2

之后就可以执行 rebuild.bat，执行后控制台屏幕会显示“请输入需要编译的主机或虚拟主机号”的信息，并且等待用户输入，此时需要用户输入要编译的主机或虚拟主机号，这个是在 webconfig.xml 中 mainHost 和 vHost 中定义的（参见 2.3 和 2.5 节），比如我们要编译主机中所有动态文件那我们就输入 0，输入后又提示“是否需要隐藏源代码（按 y 需要隐藏，其它键无需隐藏）？”，这个用于隐藏源代码功能，本章后面会有介绍，在这里我们先不需要，按其它键继续，工具提示隐藏代码功能为关闭的状态，最后就开始编译所选定主机的所有动态文件了。

所有编译后的文件都会存在主机或虚拟主机根目录下的 WEB-INF\work\com\kangaroo_egg\workfile 目录下（参见 F1.1 节），如果编译前此目录中已有文件（以前已编译的文件）则会全部删除后再重新编译。需要编译的动态文件是以

webconfig.xml 中 mainHost 和 vHost 的 dhtmlExtName 所定义的为扩展名。本编译工具也是多国语言的，所显示的语言是依据 webconfig.xml 中 systemSet 的 regionSet 设定（参见 2.1 节）。

2. 在 linux 和 solaris 下运行则打开 rebuild.sh 文件，你会看到如下内容：

```
XXX/bin/java -cp ./XXX/lib/tools.jar com.kangaroo_egg.tools.Rebuild
```

图 F3-1-3

同样注意红框内的 XXX，请将此 XXX 换成 JDK 安装的目录，比如你的 JDK 安装在 /jdk1.5 下面则替换成如下内容：

```
/jdk1.5/bin/java -cp ./jdk1.5/lib/tools.jar com.kangaroo_egg.tools.Rebuild
```

图 F3-1-4

之后就可以执行 rebuild.sh，流程和前面介绍的一样。

注意：rebuild.sh 属性必须是可执行，请在 linux 和 unix 下修改（修改方法参见 linux 和 unix 相关命令），否则将无法执行。

rebuild 工具需要使用配置文件 webconfig.xml，请确保配置正确。

F3.2 隐藏源代码的原理

许多用户希望保护自己的 web 应用，为此就需要不让其他人看到源代码，kangaroo-egg 服务器从一开始就考虑到了这个问题，为此提供隐藏源代码的方法。前面附录 1 已经详细介绍了动态文件执行的流程，动态文件只有最终编译成 java 的 class 文件后才能执行，因为 class 文件是编译后的文件所以看不到源代码，而含有源代码的只有动态文件和动态文件转换成标准的 java 文件，于是只要将这二个文件中源代码隐藏即可。那 we 来看一下隐藏着二个文件会有什么后果。

隐藏动态文件代码（即将代码删除或改成其它的非代码信息）后服务器会发现动态文件已被修改，为此会重新编译动态文件，这是 we 不想看到的，因为我们其实真正想用的是修改前已经编译过的动态文件。那么服务器是如何判断动态文件已被修改了呢？其实很简单，是看动态文件最后修改的时间，隐藏代码就会造成最后修改时间改动，所以只要在隐藏代码后将动态文件的最后修改时间再改为隐藏代码前的即可。

接下来就是隐藏动态文件转换成标准的 java 文件代码，服务器不会检测转换后 java 文件是否被修改，因此可以不用隐藏代码后修改最后修改时间，不过转换后 java 文件中会含有对应转换前的动态文件代码行数，因此在隐藏代码时需要保留这些行数信息，否则执行动态文件出错时就无法显示动态文件源代码出错的行数，如果无需提示出错行则直接删除转换后 java 文件也是可以的。不过我们还是建议保留，因为程序如在运行时出错则用户也能看到相对于源程序出错行，这样的话用户就可以将出错信息和出错行反馈给开发人员，以方便开发人员调试。

所以只要用上述介绍的原理去隐藏源代码就不会有任何问题,在隐藏源代码的同时又能保证程序运行正常。为了方便用户, rebuild 工具提供了自动用上述方法隐藏源代码的功能,下一节将介绍使用方法。

F3.3 隐藏源代码的工具

如 F3.1 节中介绍的执行 rebuild.bat 或 rebuild.sh, 执行后控制台屏幕会显示“请输入需要编译的主机或虚拟主机号”的信息, 输入需要编译的主机或虚拟主机号, 输入后就会提示“是否需要隐藏源代码(按 y 需要隐藏, 其它键无需隐藏)?”, 这时就输入 y, 之后再次提示“警告: 隐藏源代码操作会使源代码消失, 请确认已经备份了源代码。是否继续(按 y 继续, 其它键退出)?”的信息, 因为启用这个功能会使源代码全部删除为此需要确认已备份了含有源代码的动态文件, 输入 y 就会继续, 输入其它内容就会退出。我们输入 y 后工具提示隐藏代码功能为开启的状态, 最后就开始编译动态文件并且隐藏源代码。

F3.4 tools.jar 文件

rebuild.bat 和 rebuild.sh 和 run.bat 和 run.sh (参见 1.2 节) 的内容很相似, 其中一个就是都会使用到 tools.jar 这个文件, 那么这个文件有什么用呢? 要了解这个 jar 文件首先要了解 java 通常有二个版本 JRE 和 JDK, JRE 是运行 java 所需要的环境而 JDK 除了可以运行 java 外还可以开发 java 程序, 所以 JDK 包括了 JRE。只有 JDK 才含有 tools.jar 这个文件, 因为此 jar 文件中含有将 java 程序编译成 class 文件的工具, 所以 JRE 中不含有这个文件(因为 JRE 只能运行已编译的 java 程序而不能开发 java 程序, 当然就不需要这个编译工具了)。

kangaroo-egg 服务器需要使用 JDK 环境, 因为需要编译动态文件所以需要 tools.jar。rebuild.bat 和 rebuild.sh 必须要用到这个文件, 因为这个工具就是用来编译的。

然而启动服务器的脚本文件 run.bat 和 run.sh 是否必须用到此文件呢? 换句话说 kangaroo-egg 服务器一定要用到 tools.jar 吗? 其实不是的, 当 autoCompile (参见 2.3 和 2.5 节) 关闭时 DQM 容器将永远不会编译动态文件, 既然不编译那么也用不到 tools.jar 这个文件了, 所以当确定 autoCompile 关闭的情况下启动脚本就可以改成如下:

在 windows 下 run.bat 改成如下内容:



```
XXX\bin\java com.kangaroo_egg.StartServer
```

图 F3-4-1

注意红框内的 XXX, 请将此 XXX 换成 JDK 安装的目录, 比如你的 JDK 安装在 c:\jdk1.5 下面则替换成如下内容:



```
c:\jdk1.5\bin\java com.kangaroo_egg.StartServer
```

图 F3-4-2

linux 和 unix 下请参照上面的修改方法修改 run.sh 脚本文件。

再深入想想，关闭 autoCompile 情况下是无需编译而只需要运行的，那么是不是可以只用 JRE 呢，答案是肯定的，如果你确认将来永远不需要开启 autoCompile 功能，那么你可以只安装 JRE。

附录 4 类的载入

F4.1 公用类的载入

在前面 3.4 节和 10.4 节都介绍了主机或虚拟主机的 /WEB-INF/classes 和 /WEB-INF/lib 目录下可以存放各种 class 或 jar 文件, 如果动态文件需要这些类则服务器会自动从这二个目录中寻找后加载。但是存放在这些目录下的类只能被当前所处的主机或虚拟主机所读取到, 其它主机和虚拟主机是无法读取到的。不过有的时候还是需要载入的类能够被主机和所有虚拟主机都访问到, 比如 JDBC 的驱动类等, 那么这个时候该如何做呢? 当然你完全可以将类似 JDBC 驱动类放入一个个主机或虚拟主机的 /WEB-INF/classes 和 /WEB-INF/lib 目录下。不过这样似乎有点麻烦, 也不高效, 因为 10.4 节提到过在这二个目录下的类会载入到类缓存中, 而主机和每个虚拟主机都有自己独立的类缓存, 这样就会造成多个类缓存中存放着同一个需要的类型。

接下来所介绍的就是如何载入公用类。还记得 1.2 节中介绍的 run.bat 和 run.sh 这二个启动脚本吗? 我们可以通过这二个服务脚本来加载公用类, 首先来看 windows 平台下, 例如公用类是以 class 文件形式存在于 c:\commclasses 目录下, 那么要加载此目录下的这些公用类, 我们只需在原有的 run.bat 中增加如图 4-1-1 中的红框内容。

```
c:\jdk1.5\bin\java -cp .;c:\commclasses;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.StartServer
```

图 4-1-1

而如果公用类是已打包的 jar 或者 zip 文件, 则必须在脚本中一个个写明, 例如需载入的文件位于 c:\a.jar 和 d:\b.zip, 则修改如下:

```
c:\jdk1.5\bin\java -cp .;c:\a.jar;d:\b.zip;c:\jdk1.5\lib\tools.jar com.kangaroo_egg.StartServer
```

图 4-1-2

再来看 linux 和 unix 平台下, 如果公用类是以 class 文件形式存在于 /commclasses 目录下, 那么要加载这些公用类, 我们只需在原有的 run.sh 中增加如图 4-1-3 中的红框内容。

```
/jdk1.5/bin/java -cp ./commclasses:/jdk1.5/lib/tools.jar com.kangaroo_egg.StartServer
```

图 4-1-3

而如果公用类是已打包的 jar 或者 zip 文件, 则必须在脚本中一个个写明, 例如需载入的文件位于 /a.jar 和 /b.zip, 则修改如下:

```
/jdk1.5/bin/java -cp ./a.jar:/b.zip:/jdk1.5/lib/tools.jar com.kangaroo_egg.StartServer
```

图 4-1-4

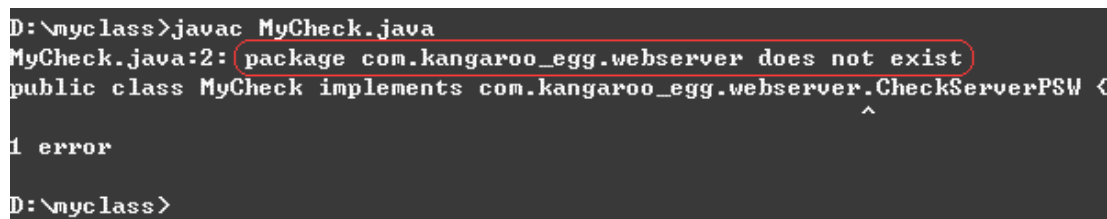
注意：windows 平台下载入多个类路径时分隔符为分号，而 linux 和 unix 平台下为冒号。通过以上方法载入的类是不会进入类缓存中的。

因为服务器会自动载入安装目录下的 classes 主程序目录(参见 1.2 节),所以如果以 class 文件形式存在于此目录下是会被自动载入的,即不需要在启动脚本中再加入此类路径,不过如果是 jar 或 zip 打包文件的话就像前面所说的还是要在启动脚本中一个个加入。

前面 2.4 节中所提到的用户可以编写自己的 java 程序用于和服务一同启动,而这些用户程序的载入也必须用上面介绍的方法,不能放在某个主机或虚拟主机的 classes 或 lib 目录下。同样 2.3 节介绍的 needPassword 部分中自己编写的验证密码程序也必须是这样载入。

F4.2 服务器类的载入

在前面 2.3 节的 needPassword 部分中有介绍到自己编写的验证密码程序必须要继承 com.kangaroo_egg.webserver.CheckServerPSW 这个接口,于是问题来了当我们编写自己的密码保护类时去哪里载入 CheckServerPSW 这个接口? 其实很简单只要将服务器安装的 classes 目录也设置为类搜索路径即可(服务器安装的 classes 目录请参见 1.2 节)。例如我么自己所编写的密码验证类是 d:\myclass\MyCheck.java,而服务器 classes 的路径是 c:\webserver\classes。那么如果直接去编译 MyCheck.java 会发生什么结果呢? 如下图所看到的会提示找不到 CheckServerPSW 这个接口。



```
D:\myclass>javac MyCheck.java
MyCheck.java:2: package com.kangaroo_egg.webserver does not exist
public class MyCheck implements com.kangaroo_egg.webserver.CheckServerPSW {
^
1 error
D:\myclass>
```

图 4-2-1

于是我们将 c:\webserver\classes 加入类路径,再次编译就会看到成功了。



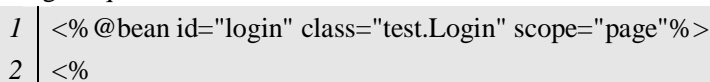
```
D:\myclass>javac -cp c:\webserver\classes MyCheck.java
D:\myclass>
```

图 4-2-2

所以遇到编译时要用到服务器类就可以采用上面的方法。如果使用 JBuilder、eclipse 这些 IDE 工具进行编写,那么请参考这些 IDE 工具说明用以设置需要的类路径。

在 web 程序中很多会用到 javaBean,而在 javaBean 中处理内置对象也是很必要的,如下面的例子。

login.dqm 的源代码如下:



```
1 | <% @bean id="login" class="test.Login" scope="page"%>
2 | <%
```

```

3  if (session.getAttribute("username") == null) {
4      if (login.action(request, session)) {
5          out.print("Login ok.");
6      } else {%>
7          <form method="POST" action="">
8              username: <input type="text" name="username" size="20"></p>
9              password: <input type="password" name="password" size="20"></p>
10             <input type="submit" value="Submit" name="B1">
11             </form>
12 <%
13     }
14 }
15 else {
16     out.print("Login ok.");
17 }%>

```

Login.java 的源代码如下：

```

1  package test;
2
3  public class Login {
4      public Login() {
5      }
6
7      public boolean action(com.kangaroo_egg.dqm.DRequestItface request,
8          com.kangaroo_egg.dqm.DSessionItface session) {
9          //判断用户输入的用户名和密码是否正确
10         String username = request.getParameter("username");
11         String password = request.getParameter("password");
12         //如果密码正确则将用户名增加到 session 中
13         if ("admin".equals(username) && "123456".equals(password)) {
14             session.setAttribute("username", username);
15             return true;
16         }
17         return false;
18     }
19 }

```

login.dqm 用于登录界面，如果先前没有登录过则会显示登录界面（如图 4-2-3），如果登录成功则会提示"Login Ok."。



图 4-2-3

那么首先来看 Login.java 的源程序，此程序功能就是进行密码检测，这里我们使用固定的用户名密码（源程序 13 行），如果验证通过则在 session 中写用户名的同时返回 true 值（源程序 14、15 行），如果验证不通过则直接返回 false 值。而用户输入的用户名和密码是从 request 中读取的（源程序 10 和 11 行），程序中需要使用到的 request 和 session 对象是从 login.dqm 中传进来的（Login.java 源程序 7 和 8 行，login.dqm 源程序第 4 行）。因为 Login.java 中包含了 request 和 session 这二个对象（Login.java 源程序 7 和 8 行），所以需要用到服务器类，而载入的方法就是前面介绍过的。

我们再来看 login.dqm，如果用户输入的用户名和密码正确则显示"Login Ok."（源程序 4 到 6 行）。如果不正确则显示登录界面（源程序 6 到 11 行）。如果 session 中含有用户名则说明先前成功登录过，则也显示"Login Ok."（源程序 15 到 17 行）。可以看到将内置对象传入 JavaBean 的做法可以使 web 程序更清晰且易于维护。读者可以用用户名 admin 和密码 123456 可以测试本例子。

表 4-2-1 列出了 6 个内部对象的类型。

| 内置对象 | 类型 |
|-------------|---|
| out | com.kangaroo_egg.dqm.AbstractMyOut |
| request | com.kangaroo_egg.dqm.DRequestItface |
| response | com.kangaroo_egg.dqm.DResponseItface |
| session | com.kangaroo_egg.dqm.DSessionItface |
| application | com.kangaroo_egg.dqm.DApplicationItface |
| command | com.kangaroo_egg.dqm.DCommand |

表 4-2-1

附录 5 内部变量

内置变量是 kangaroo-egg 所保留的变量，用以完成特殊的功能。用户自定义的变量名不能与内部变量名重复，不过内部变量名都是以`ke`开头的，所以一般不会与用户自定义的变量名相同。接下来我们就来看看内置变量的用途。

F5.1 `kesourceLineNumber`

`kesourceLineNumber` 变量的类型是 `int` 型。

此内部变量记录了当前在动态文件中的行数，比如此变量在动态文件第 4 行出现，那么此变量就是 4。可是这个变量有什么用呢？我们来看一下 `ke_sourceLineNumber.dqm` 这个例子：

```
1 <form method="POST" action="">
2   <p>只能输入除了（50 到 60、100 到 110）以外的数字:
3   <input type="text" name="inputVar" size="20">
4   <input type="submit" value="Submit" name="B1"></p>
5 </form>
6
7
8 <%
9 String input = request.getParameter("inputVar");
10 if (input != null && input.length() > 0) {
11     int inputInt;
12     try {
13         inputInt = Integer.parseInt(input);
14     }
15     catch(NumberFormatException ex) {
16         outErr(out, 1);
17         return;
18     }
19
20     if (inputInt >= 50 && inputInt <= 60) {
21         outErr(out, 2);
22         return;
23     }
24 }
```

```

25     if (inputInt >= 100 && inputInt <= 110) {
26         outErr(out, 3);
27         return;
28     }
29
30     out.print("输入正确");
31 }
32
33 %>
34 <%!
35 private static void outErr(AbstractMyOut out, int errId) throws IOException {
36     out.print("输入的数字不符合规则 ErrCode = " + errId);
37 }
38 %>

```

上面的程序运行界面如下：

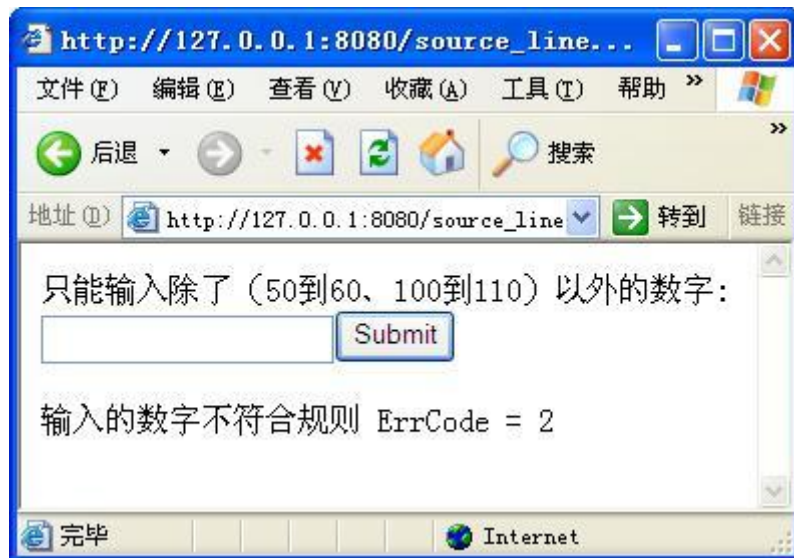


图 F5-1-1

此程序要求用户只能输入除了 50 到 60（源程序 20 到 23 行）和 100 到 110（源程序 25 到 28 行）以外的任何数字，如果输入了非数字以及 50 到 60 和 100 到 110 以内的数字则会报错，但报错信息只是提示“输入的数字不符合规则”（源程序 35 到 37 行），为此加上出错编号以区分具体错误。

于是又一个问题，如果有很多错误源，那么你就要手动添加删除错误编号了。比如上面程序需要再增加一个非法数集 75 到 85，且要增加在检查 100 到 110 错误前，那么你就需要将检查 100 到 110 的错误编号改为 4，同时增加检查 75 到 85 的错误编号 3，修改后的源程序如下：

```

1  <form method="POST" action="">
2  <p>只能输入除了（50 到 60、75 到 85 和 100 到 110）以外的数字:
3  <input type="text" name="inputVar" size="20">
4  <input type="submit" value="Submit" name="B1"></p>

```

```

5 </form>
6
7
8 <%
9 String input = request.getParameter("inputVar");
10 if (input != null && input.length() > 0) {
11     int inputInt;
12     try {
13         inputInt = Integer.parseInt(input);
14     }
15     catch(NumberFormatException ex) {
16         outErr(out, 1);
17         return;
18     }
19
20     if (inputInt >= 50 && inputInt <= 60) {
21         outErr(out, 2);
22         return;
23     }
24
25     if (inputInt >= 75 && inputInt <= 85) {
26         outErr(out, 3);
27         return;
28     }
29
30     if (inputInt >= 100 && inputInt <= 110) {
31         outErr(out, 4);
32         return;
33     }
34
35     out.print("输入正确");
36 }
37
38 %>
39 <%!
40 private static void outErr(AbstractMyOut out, int errId) throws IOException {
41     out.print("输入的数字不符合规则 ErrCode = " + errId);
42 }
43 %>

```

或许你会说上面的修改很简单，但是请试想一下如果有很多出错源会怎样，你要将那些位于新增出错编号后的老出错编号重新整理，这样是不是很麻烦？不过有了 `kesourceLineNumber` 这个变量事情就变得简单多了，请看下面的源程序：

```

1 <form method="POST" action="">

```



```

2    <p>只能输入除了（50 到 60、75 到 85 和 100 到 110）以外的数字:
3    <input type="text" name="inputVar" size="20">
4    <input type="submit" value="Submit" name="B1"></p>
5  </form>
6
7
8  <%
9  String input = request.getParameter("inputVar");
10 if (input != null && input.length() > 0) {
11     int inputInt;
12     try {
13         inputInt = Integer.parseInt(input);
14     }
15     catch(NumberFormatException ex) {
16         outErr(out, $ke$sourceLineNumber);
17         return;
18     }
19
20     if (inputInt >= 50 && inputInt <= 60) {
21         outErr(out, $ke$sourceLineNumber);
22         return;
23     }
24
25     if (inputInt >= 75 && inputInt <= 85) {
26         outErr(out, $ke$sourceLineNumber);
27         return;
28     }
29
30     if (inputInt >= 100 && inputInt <= 110) {
31         outErr(out, $ke$sourceLineNumber);
32         return;
33     }
34
35     out.print("输入正确");
36 }
37
38 %>
39 <%!
40 private static void outErr(AbstractMyOut out, int errId) throws IOException {
41     out.print("输入的数字不符合规则 ErrCode = " + errId);
42 }
43 %>

```

当再次运行上面的程序，如果出错你会发现出错编号已经变成了源代码抛出错误的行

数，同时以后不管在哪里要增加出错源，也不需要关心以前定义的出错编号，因为编译器会帮你重新定义 `kesourceLineNumber` 的值，你所做的就是写一个固定的内部变量 `kesourceLineNumber`。

那么接下来我们需要讨论另一个问题，上面的程序为什么不直接提示具体错误而非要用数字作为提示信息呢？似乎这个问题很难回答，不同的开发者有不同的考虑，不过有一点可以肯定如果提示信息中含有容易辨别的编号，当发生的错误需要回馈给开发人员，则使用编号更易于定位错误区域，而根据出错的行号定位出错区域则再简单也不过了。

现在再来看一下 `kesourceLineNumber` 实现的原理，其实很简单，在将动态文件编译成 class 文件前编译器会将 `kesourceLineNumber` 替换成其所在行的行数，之后再编译。

附录 6 生成证书

F6.1 创建证书的 2 种方法

如果要启用 SSL 连接（参见 2.2 节 https 项）则必须要使用证书，因为 SSL 连接使用证书来进行验证。对于需要使用 SSL 来保证通信安全的客户端和服务端，都必须创建证书。证书的创建有 2 种方法，一种是通过证书由权威认证机构 (CA) 来创建证书，目前知名的权威认证机构有 Verisign, Entrust 和 Thawte 等，虽然由 CA 来创建证书最权威，不过需要收费，所以另一种方法是使用 JDK 自带的工具来创建证书。不过因为主流浏览器默认都只认 CA 创建的证书是安全的，为此如果用 JDK 自带工具创建的证书时浏览器会提示用户服务器证书不是 CA 认证的，并询问用户是否接受（参见图 F6-1-1）。

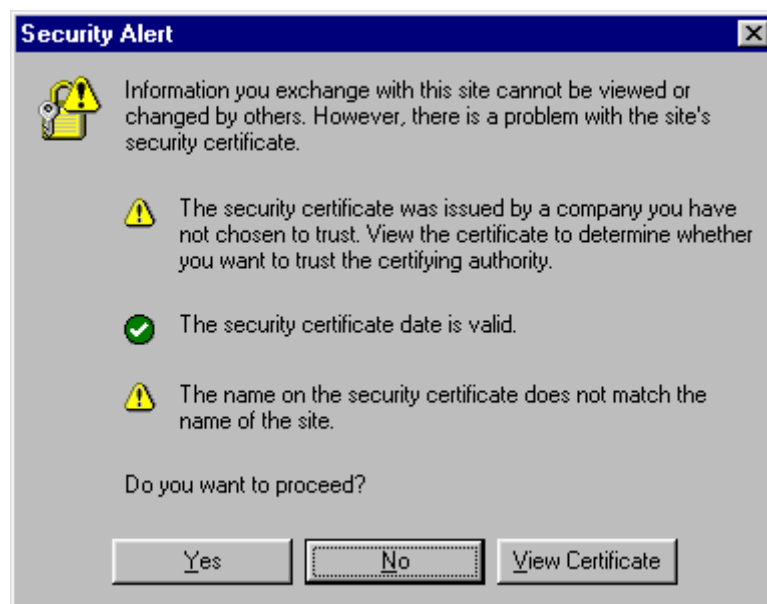


图 F6-1-1

所以在内部的私有系统中产生你自己的证书是个很好的主意。但在公共系统中，最好从知名的 CA 处获得证书，以避免浏览器的安全警告。如果你接受证书，你就可以看到安全连接之后的页面。以后访问同一个网站的时候浏览器就不再会弹出安全警告了。有许多网站使用 HTTPS，而证书是自己产生或者由不知名的 CA 产生的。例如我们的网站 <https://www.kangaroo-egg.com>。如果你没访问过我们的网页，你会看到像图 F6-1-1 一样的安全警告。

F6.2 使用 JDK 工具创建证书

JDK 提供了一个创建证书的工具，使用这个工具可以很容易生成自己的证书。这个工具在 JDK 安装目录下的 bin 目录下，名字为 keytool。下面我们就来介绍次工具如何使用，我们假设在 windows 环境下的 c:\，那么用下列命令来为 HTTP 服务器创建一个证书：

```
c:\jdk1.5\bin\keytool -genkey -keystore key.store -alias kangaroo-egg
```

这个命令会产生一个由别名 kangaroo-egg 引用的证书，并将其保存在一个名为 key.store 的文件中。产生证书的时候，工具会提示我们输入一些信息，如下面的信息，其中黑体内容为用户输入的。

Enter keystore password: **123456**

What is your first and last name?

[Unknown]: **Shemin Dunne**

What is the name of your organizational unit?

[Unknown]: **Software Department**

What is the name of your organization?

[Unknown]: **Kangaroo-egg Software**

What is the name of your City or Locality?

[Unknown]: **ShangHai**

What is the name of your State or Province?

[Unknown]: **ShangHai**

What is the two-letter country code for this unit?

[Unknown]: **CN**

Is CN=Shemin Dunne, OU=Software Department, O= Kangaroo-egg Software, L=ShangHai, ST=ShangHai, C=CN correct?

[no]: **y**

Enter key password for

(RETURN if same as keystore password): **654321**

完成后就会在当前目录下生成 key.store 这个文件，所以这个文件绝对路径为：c:\jdk1.5\bin\key.store。于是我们配置 webconfig.xml 的 https 项（参见 2.2 节），配置后内容如下：

```
- <connectors>
  <coreConnectionNumber>5</coreConnectionNumber>
  <maxConnectionNumber timeOut="2">40</maxConnectionNumber>
  <maxServerUnavailableNumber>1</maxServerUnavailableNumber>
  <maxPersistentConnectionsNumber>20</maxPersistentConnectionsNumber>
  <connectionTimeout>50</connectionTimeout>
  <HTTP enable="true" port="8080" />
  <HTTPS enable="true" port="8443" keystoreFile="c:\jdk1.5\bin\key.store"
    keystoreType="jks" keystorePass="123456" keyPass="654321" needClientAuth="false" />
</connectors>
```

图 F6-2-1

这样启动服务器后就可以用 https 访问了，如果将上面 needClientAuth 设置为 true，则需要客户端提供合法的证书，否则会拒绝连接。您可以参见其它相关书籍以进一步了解相关内容。

Copyright © 2005, 2006 Kangaroo-egg. All rights reserved.

License

本软件为免费且开源软件，源代码环境为JBui lder2005。但在使用本软件及源代码时请遵循以下原则：

1. 您可以分发复制本软件及源代码，但必须完整包含本手册。
2. 您可以修改本程序的源代码，但必须在与之发布的同时注明修改的地方，同时说明源程序的原始出处。
3. 在延伸的代码中（修改和有源代码衍生的代码中）需要带有本协议。
4. 商业版或牟利性软件（包含无形资产牟利）不得使用本源代码的全部或部分，除非有作者书面授权。
5. 不可以用本软件的作者/机构名字和本产品的名字做市场推广。
6. 本协议将有可能不断扩充，使用者必须遵循将来扩充协议。