网站的静态页面生成方案

yanrong79@gmail.com

提升网站性能的方式有很多，例如有效的使用缓存，生成静态页面等等。今天要说的就是生成静态页面的方式。

什么叫生成静态页面呢？很简单，例如我们如果访问一个链接 http://xyz.com/index.do，那么服务器就会解析这个请求，让特定的 Action 去处理。这其中的缺点是显而易见的：如果访问的人，那么就会加重应用服务器的压力，最恶劣的后果就是应用服务器 down 掉了。那么如何去避免呢？如果我们把对 index.do 请求后的结果保存成一个 html 文件，然后每次用户都去访问 http://xyz.com/index.html ，这样应用服务器的压力不就减少了？

好吧，我已经学会如何生成静态页面了。打开我的浏览器，输入 http://xyz.com/index.do，然后再使用“另存为”功能保存一个 html 文件，然后上传到服务器，这样就 ok 了！

我不能说这不是解决方案，但是我可以说这不是好的解决方案。我们需要的是自动的生成静态页面，当用户访问 http://xyz.com/index.do，会自动生成 index.html ，然后显示给用户。

1、 基础—URL Rewrite

什么是 URL Rewrite 呢？请 google it ☺ 。不过这里简单的啰嗦一下。从字面上我们就可以理解它的意思：URL 重写。用一个简单的例子来说明问题：输入网址 http://www.xyz.com/index.do ，但是实际上访问的却是 http://www.xyz.com/index.action，那我们就可以说 URL 被重写了。这项技术应用广泛，有许多开源的工具可以实现这个功能。

2、 基础—Servlet web.xml

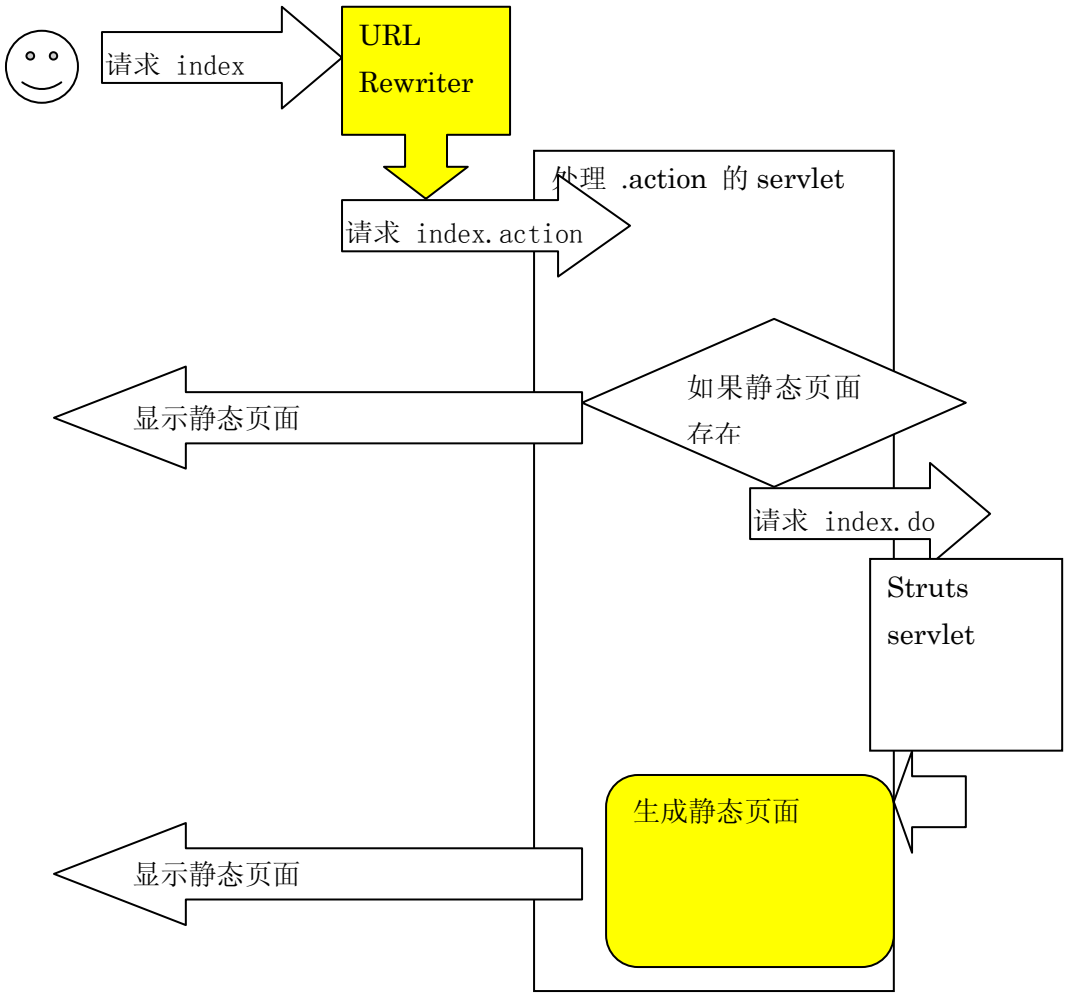
如果你还不知道 web.xml 中一个请求和一个 servlet 是如何匹配到一起的，那么请搜索一下 servlet 的文档。这可不是乱说呀，有很多人就认为 /xyz/*.do 这样的匹配方式能有效。

如果你还不知道怎么编写一个 servlet ，那么请搜索一下如何编写 servlet。这可不是说笑呀，在各种集成工具漫天飞舞的今天，很多人都不会去从零编写一个 servlet 了。

3、 谈谈方法

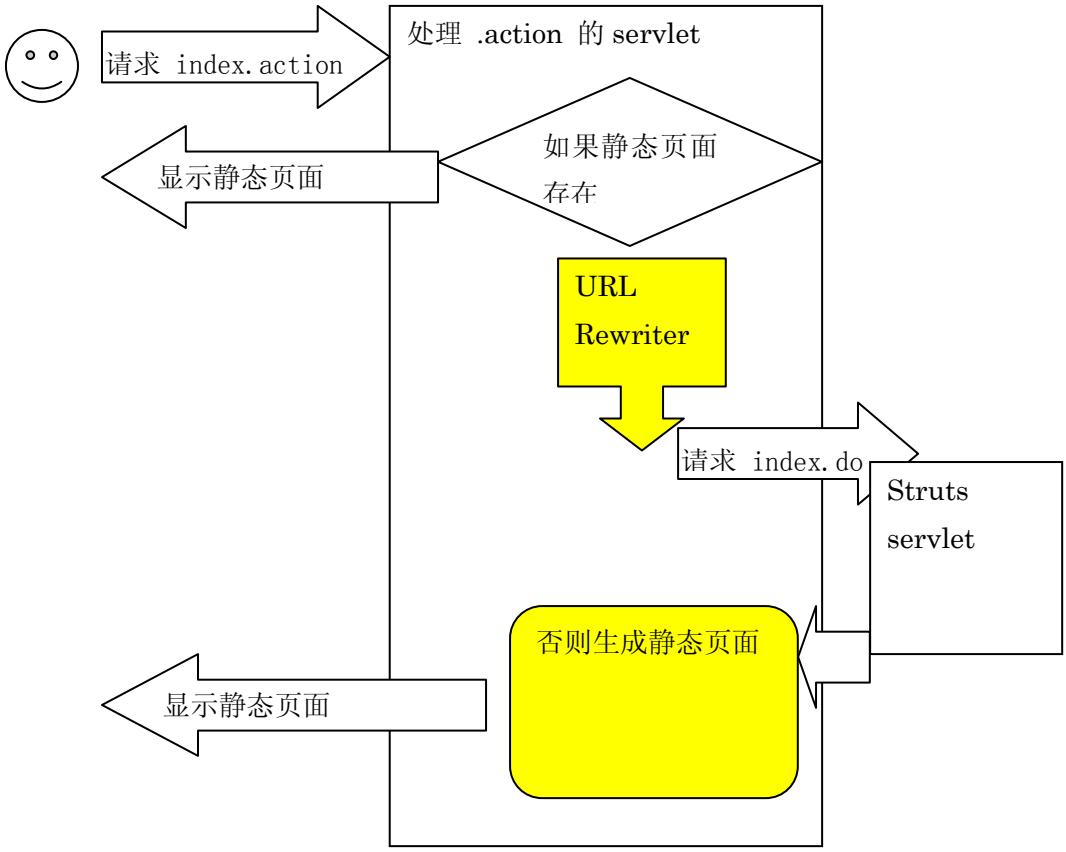
好了，现在步入正题。谈谈生成静态页面的方法(还是不要说成“原理”，给人一种

很高深的感觉)。方法就是这样：



方法 1

其中，对于 URL Rewriter 的部分，可以使用收费或者开源的工具来实现，如果 url 不是特别的复杂，可以考虑在 servlet 中实现，那么就是下面这个样子：



方法 2

图画的有些乱，不过方法还是非常简单，其中最关键的有 2 个地方：1 是 url rewriter，2 是生成静态页面的具体代码。

4、 从一个简单的方式入手

现在我们从一个简单的方式入手来看看怎么生成静态页面。假如现在要显示第一页的商品，那么通常的链接会是这个样子：http://xyz.com/product.do?pageNumber=1 。好，我们说了使用最简单的方式，那么我们就不使用 url rewriter 工具了，我们来做一个约定：当我们访问 http://xyz.com/product_pageNumber_1.shtml 的时候，就是在访问 http://xyz.com/product.do?pageNumber=1 。规律就是请求的 action，和参数的名称，参数的值，都用下滑线分开，而且请求的网页是 .shtm 。仅仅有这样的约定还

不够，我们还得编写一个 servlet，去处理 .shtm 的请求：

```
<servlet>
   <servlet-name>htmlCreator</servlet-name>
   <servlet-class>web.hc.HtmlCreatorServlet</servlet-class>
   <load-on-startup>2</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>htmlCreator</servlet-name>
  <url-pattern>*.shtm</url-pattern>
</servlet-mapping>
```

下面就是 servlet 的内容了：

```
public class HtmlCreatorServlet extends HttpServlet {

        private static final long serialVersionUID = 1L;

        private Logger logger=Logger.getLogger(HtmlCreatorServlet.class);

        public void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                doPost(request, response);
        }

        public void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
                //  编码方式，可以配置到 web.xml 里。
                String encoding = “UTF-8”;
                //  得到真实的请求地址
                String templatePath = simpleURLReWrite(request);
                String realPath=
                        request.getSession().getServletContext().getRealPath("/");
                //  想要生成的静态html文件的名字
                String htmlName = getHtmlFileName(request);
```

```
// 静态html的名字，包含绝对路径
String cachFileName = realPath + File.separator + htmlName;

logger.debug("cachFileName = " + cachFileName);
File cacheFile = new File(cachFileName);

boolean load = true;
//  如果静态html 存在，就直接显示html，否则，我们就生成它。
if(cacheFile.exists()) {
         load = false;
}
if(load) {
         final ByteArrayOutputStream os = new ByteArrayOutputStream();
         final ServletOutputStream stream = new ServletOutputStream() {
            public   void   write(byte[]   data,   int   offset,
                                        int   length)   {
               os.write(data,   offset,   length);
            }

            public   void   write(int   b)   throws   IOException   {
               os.write(b);
            }
         };

         final PrintWriter pw =
               new PrintWriter(new OutputStreamWriter(os, encoding));

        HttpServletResponse rep =
               new HttpServletResponseWrapper(response)   {

               public ServletOutputStream getOutputStream() {
                   return stream;
                }

               public PrintWriter getWriter() {
                  return pw;
```

```
                    }
                };

        logger.debug("HtmlCreatorServlet RequestDispatcher = " + templatePath);
                //  使用 RequestDispatcher 去处理真正的请求。
                // 例如 index.shtm ，则转发到 index.do
                RequestDispatcher  dispatcher =
                   getServletContext().getRequestDispatcher(templatePath);
                dispatcher.include(request, rep);
                pw.flush();
                FileOutputStream fos = null;
                try {

                        if(os.size() == 0) {
                             // 如果请求的地址无效，那么就发送一个404错误。
                            response.sendError(
                               HttpServletResponse.SC_NOT_FOUND, "");
                        } else {
                                 // 生成静态文件，并且显示这个静态文件
                                fos = new FileOutputStream(cachFileName);
                                 os.writeTo(fos);
                                 dispatcher =
                                      getServletContext().
                                      getRequestDispatcher("/"+htmlName);
                                 dispatcher.include(request, response);
                        }

                } finally {
                        if(fos != null) {
                                 fos.close();
                        }
                }

          } else {
                   RequestDispatcher dispatcher =
                      getServletContext().getRequestDispatcher("/"+htmlName);
```

```
                        dispatcher.include(request, response);
                }
        }
        // 这个方法就是把 http://xyz.com/product_pageNumber_1.shtm
        // 变成 http://xyz.com/product.do?pageNumber=1
        protected String simpleURLReWrite(HttpServletRequest request)
                        throws ServletException, IOException {
                String uri = request.getRequestURI();
                String contextPath = request.getContextPath();
                logger.debug("HtmlCreator contextPath = " + contextPath);
                if (contextPath != null && contextPath.length() > 0)
                        uri = uri.substring(contextPath.length());

                uri = uri.substring(0, uri.length()-5);

                String[] urls = uri.split("_");

                uri = urls[0] + ".do";
                if(urls.length > 1) {
                        for(int i = 1; i < urls.length; i += 2) {
                                if(i==1) {
                                        uri += "?" + urls[i] + "=" + urls[i+1];
                                } else {
                                        uri += "&" + urls[i] + "=" + urls[i+1];
                                }
                        }
                }

                logger.debug("HtmlCreatorServlet get uri = " + uri);

                return uri;
        }
        // 这个方法就是根据 http://xyz.com/product_pageNumber_1.shtm
        // 来得到生成的html文件名字，也就是 product_pageNumber_1.html
        private String getHtmlFileName(HttpServletRequest request) throws
ServletException, IOException{
```

```
                String uri = request.getRequestURI();
                String contextPath = request.getContextPath();
                if (contextPath != null && contextPath.length() > 0)
                        uri = uri.substring(contextPath.length());

                uri = uri.substring(1, uri.length()-5);
                uri += ".html";
                return uri;
        }
```

真是简单的不能再简单了！我们这个 servlet 其实什么也没有做，因为真正的请求我们已经转发给 struts 了。

5、 请遵守访问方式？

上面我们已经知道，如果用户访问 product_pageNumber_1.shtm，那么我们可以保证让用户访问静态页面，从而减轻应用服务器的压力。可是如果客户直接访问 index.do?pageNumber=1 呢？难道我们要在页面上告诉访问者“请遵守访问方式，否则您将被如何如何”吗？

看来我们需要在 index.do 中做点手脚。如果 index.do 中能够知道当前的请求是从 index.shtm 传过来，而不是直接从浏览器中传过来不就好了？我们可以在 index.shtm 的 servlet 里设置一个变量，然后在 index.do 的 action 中检查这个变量是否存在。实现方式就不罗嗦了，因为实在是太容易了！

6、 为什么没有使用 URL Rewriter？

上面的例子没有使用 URL Rewriter！但是请看看“simpleURLReWrite”这个方法，如果网站的应用不复杂，那么自己编写一个满足需求的 url rewriter 也是个不错的主意。大家完全可以参考上面的实现方式，然后再增加 url rewriter 的工具。

7、 怎么刷新这些静态的 html？

刷新这些静态 html，可以考虑这么几种方式。

1、手动刷新。我想这个功能是必须的。管理员增加了一个商品，他当然可以立刻刷新静态页面了。

2、定时刷新。定时刷新可以 2 种方式。一是定时刷新全部页面。这也没什么难的，例如要刷新全部商品的页面，你可以得到全部的商品，分页的规则，然后就分别发送 http 请求就可以了。二是乐观刷新（我自己编造的词）。我仅仅刷新已经存在的 html。

例如如果目录里存在 index.html ，那么我就发送 index.shtm 请求。
实现的代码？实在太简单了，搜索 java 中如何发送 http 请求就可以了。