0. 概述

我们在创建Servlet时会覆盖service()方法，或doGet()/doPost(),这些方法都有两个参数，一个为代表请求的request和代表响应response。service方法中的request的类型是ServletRequest，而doGet/doPost方法的request的类型是HttpServletRequest，HttpServletRequest是ServletRequest的子接口，功能和方法更加强大。

1. 获得请求行

1.1 获得请求方式

```
13          //1、获得请求方式
14          String method = request.getMethod();
15          System.out.println("method:"+method);
16          //2、获得请求的资源相关的内容
```

1.2 获得请求url及uri

```
16          //2、获得请求的资源相关的内容
17          String requestURI = request.getRequestURI();
18          StringBuffer requestURL = request.getRequestURL();
19          System.out.println("uri:"+requestURI);
20          System.out.println("url:"+requestURL);
```

```
uri:/WEB15/line
url:http://localhost:8080/WEB15/line
```

1.3 获得web应用的名称

```java
        //获得web应用的名称
        String contextPath = request.getContextPath();
        System.out.println("web应用："+contextPath);
```

```
web应用：/WEB15
```

1.4 获得地址后参数的字符串

```java
        //地址后的参数的字符串
        String queryString = request.getQueryString();
        System.out.println(queryString);
```

显然，post方式访问得到null值，get方式才能得到后面的参数值。

```
username=zhangsan2&passwd=1234
```

1.5 获得客户机的IP信息

```java
        //3、获得客户机的信息---获得访问者IP地址
        String remoteAddr = request.getRemoteAddr();
        System.out.println("IP:"+remoteAddr);
```

2. 获取请求头

2.1 获得指定头的键值对

```java
        //1、获得指定的头
        String header = request.getHeader("User-Agent");
        System.out.println(header);
```

2.2 获得所有的头键名并遍历

```java
        //2、获得所有的头的名称
        Enumeration<String> headerNames = request.getHeaderNames();
        //遍历所有的头取出其键值
        while(headerNames.hasMoreElements()){
            String headerName = headerNames.nextElement();
            String headerValue = request.getHeader(headerName);
            System.out.println(headerName+":"+headerValue);
        }
```

3. 根据请求头设计防盗链技术

如果资源被别的资源以链接技术访问，在发送给本资源的请求会有一个referer的请求头。但是链接所指向的内容是有价值的，不经允许不能在别的网站跳转访问的需求是存在的，为此，我们需要设

计防盗链技术。
比如我们在一个html文档中放一个链接标签指向我们的referer资源，而我们的refer资源设计了防盗链技术。如下：

```
13          //对该新闻的来源的进行判断
14          String header = request.getHeader("referer");
15          response.setContentType("text/html;charset=UTF-8");
16          if(header!=null&&header.startsWith("http://localhost")){
17              //是从我自己的网站跳转过来的 可以看新闻
18              response.getWriter().write("中国确实已经拿到100块金牌....");
19          }else{
20              response.getWriter().write("你是盗链者，可耻！！");
21          }
```

通过判断referer头的值是否以某个地址开头，否则访问失败。

4. 获取请求体

4.1 获取单个参数的值

```
16          //1、获得单个表单值
17          String username = request.getParameter("username");
18          System.out.println(username);
19          String password = request.getParameter("password");
20          System.out.println(password);
```

4.2 获取一对多的参数值

```
21          //2、获得多个表单的值
22          String[] hobbys = request.getParameterValues("hobby");
23          for(String hobby:hobbys){
24              System.out.println(hobby);
25          }
```

4.3 获取所有请求参数名称

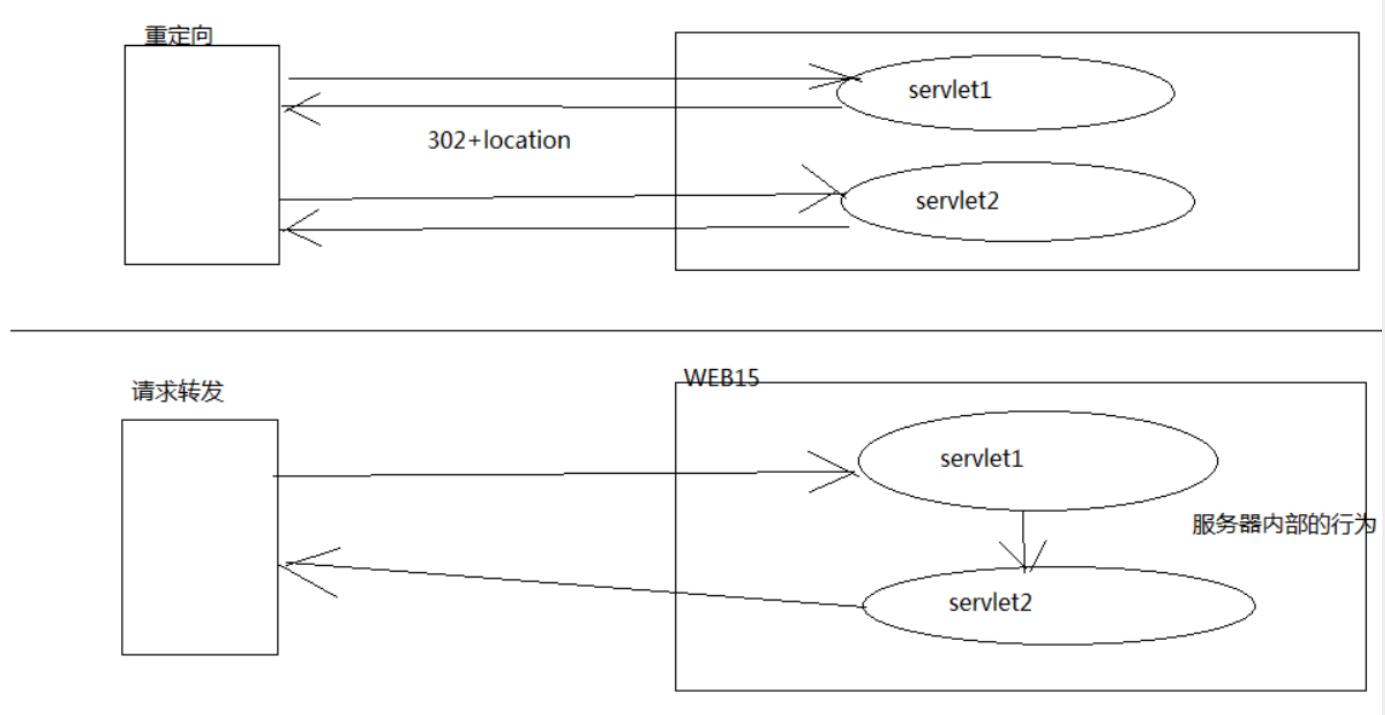
```
26          //3、获得所有的请求参数的名称
27          Enumeration<String> parameterNames = request.getParameterNames();
28          while(parameterNames.hasMoreElements()){
29              System.out.println(parameterNames.nextElement());
30          }
```

4.4 获取所有请求参数的名称及值

```java
        //4、获得所有的参数 参数封装到一个Map<String,String[]>
        Map<String, String[]> parameterMap = request.getParameterMap();
        for(Map.Entry<String, String[]> entry:parameterMap.entrySet()){
            System.out.println(entry.getKey());
            for(String str:entry.getValue()){
                System.out.println(str);
            }
            System.out.println("--------------------------");
        }
```

再次提醒：**getParameter是得到请求体中的参数，getAttribute是得到域中的数据**

5. 转发与重定向的区别



注意：转发与重定向的区别：

1. 重定向两次请求，转发一次请求
2. 重定向地址栏的地址变化，转发地址不变
3. 重新定向可以访问外部网站, 转发只能访问内部资源
4. 转发的性能要优于重定向

6. Request的其它作用

6.1 Request域对象

request对象也是一个存储数据的区域对象，所以也具有如下方法：

setAttribute(String name, Object o)

getAttribute(String name)

removeAttribute(String name)

注意：request域的作用范围：一次请求中

6.2 转发

```
16          //想request域中存储数据
17          request.setAttribute("name", "tom");
18
19
20          //servlet1 将请求转发给servlet2
21          RequestDispatcher dispatcher = request.getRequestDispatcher("/servlet2");
22          //执行转发的方法
23          dispatcher.forward(request, response);
```

将servlet1的请求发给servlet2，在转发过程中，request对象保持不变，任意被转发的过程中都可以被取到

Servlet2处理，但浏览器地址栏地址还是Servlet1上的。

```
16          //从request域中取出数据
17          Object attribute = request.getAttribute("name");
18
19          response.getWriter().write("hello ..."+attribute);
```

7.注意事项

7.1 ServletContext域与Request域的生命周期比较

ServletContext：
创建：服务器启动
销毁：服务器关闭
域的作用范围：整个web应用
request：
创建：访问时创建request
销毁：响应结束request销毁
域的作用范围：一次请求中

7.2 服务器地址与客户端地址

客户端地址：
是客户端去访问服务器的地址，服务器外部的地址，特点：写上web应用名称
直接输入地址：重定向
服务器端地址：
服务器内部资源的跳转的地址，特点：不需要写web应用的名称
比如：转发