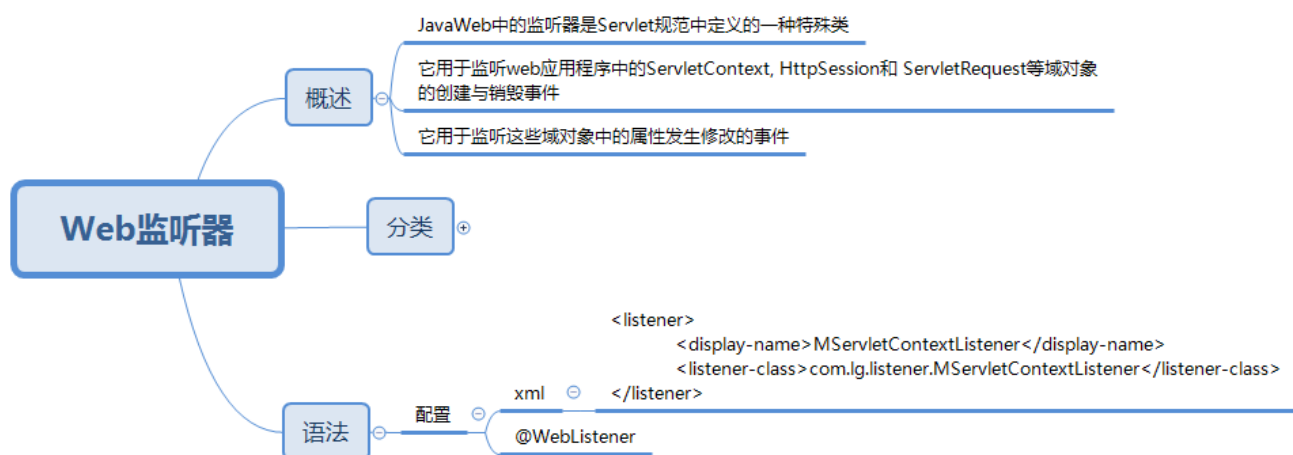


* 学习目标

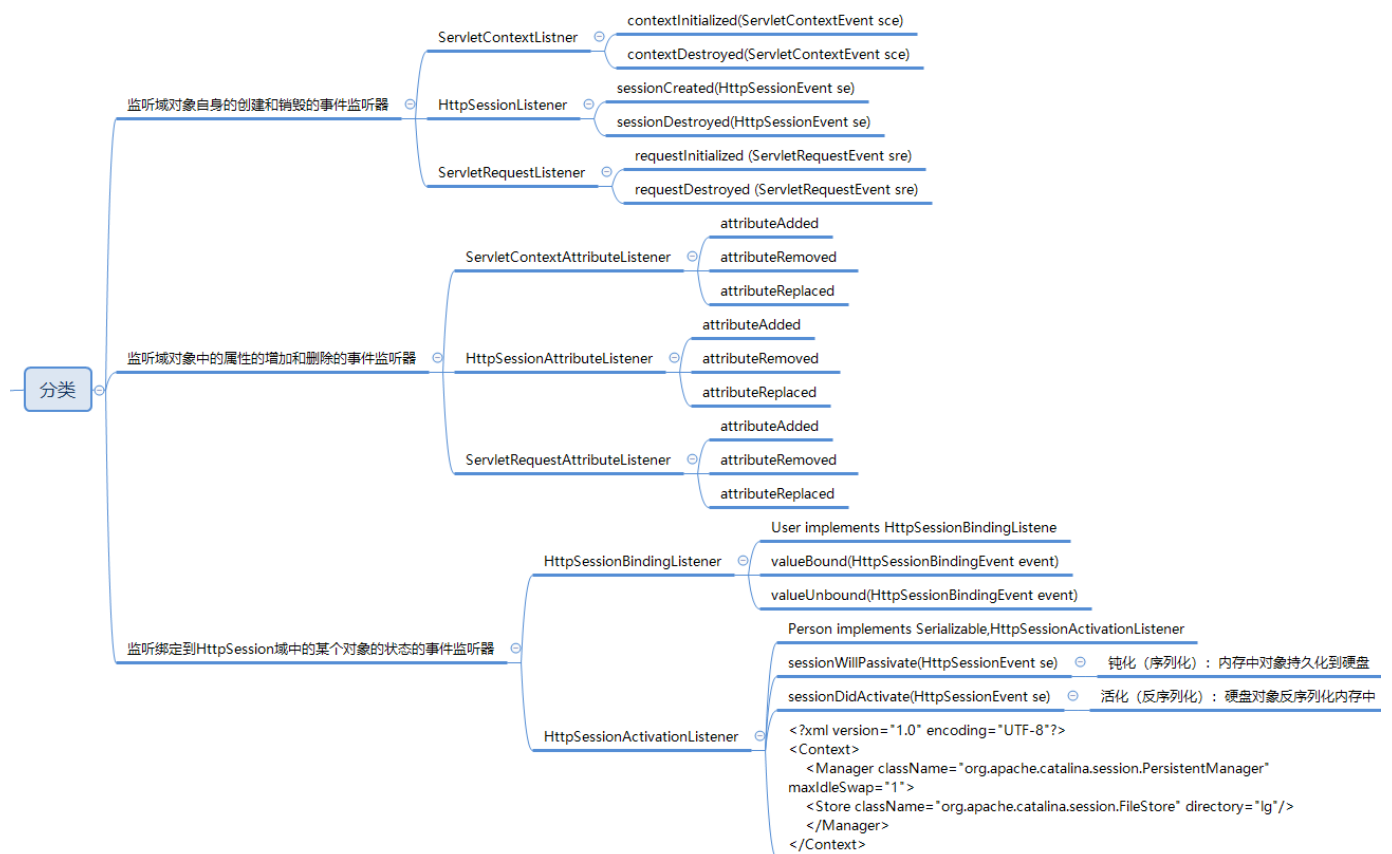
- * 能够掌握Web监听器
 - * 能够使用Web监听器开发简单的案例
 - * 能够掌握Ajax的概述
 - * 能够掌握Ajax的开发
 - * 能够结合BootStrap实现列表分页显示的效果
-
-

* 能够掌握Web监听器

* Web监听器的概述



* 分类



```

1  * 开发案例
2  @WebListener()
3  public class MListener implements ServletContextListener,
4      HttpSessionListener, HttpSessionAttributeListener {
5      public MListener() {
6      }
7
8      public void contextInitialized(ServletContextEvent sce) {
9          System.out.println("MListener:contextInitialized");
10     }
11
12     public void contextDestroyed(ServletContextEvent sce) {
13         System.out.println("MListener:contextDestroyed");
14     }
15
16     public void sessionCreated(HttpSessionEvent se) {
17         HttpSession session = se.getSession();
18         System.out.println("MListener:sessionCreated");
  
```

```

19         System.out.println(session+"被创建");
20     }
21
22     public void sessionDestroyed(HttpSessionEvent se) {
23         HttpSession session = se.getSession();
24         System.out.println("MListener:sessionDestroyed");
25         System.out.println(session+"被创建销毁");
26     }
27
28     public void attributeAdded(HttpSessionBindingEvent sbe) {
29         System.out.println("HttpSessionAttributeLitener:attributeAdded:"+sbe.get
30     }
31
32     public void attributeRemoved(HttpSessionBindingEvent sbe) {
33         System.out.println("HttpSessionAttributeLitener:attributeRemoved:"+sbe.g
34     }
35
36     public void attributeReplaced(HttpSessionBindingEvent sbe) {
37         System.out.println("HttpSessionAttributeLitener:attributeReplaced:"+sbe.
38     }
39 }
40

```

* 能够使用Web监听器开发简单的案例

```

1 * 开发案例:
2 * 统计当前在线人数
3 * 当用户访问我们网站: 分配一个Session
4 * 当Session被创建的时候, 计数器加一, 当Session被销毁的时候, 计数器减一
5 * 计数器存在ServletContext域中
6 @WebListener
7 public class OnLineCountLinstener implements HttpSessionListener {
8     private ReentrantLock lockCreated=new ReentrantLock();
9     private ReentrantLock lockDestroyed=new ReentrantLock();
10    @Override
11    public void sessionCreated(HttpSessionEvent se) {
12        lockCreated.lock();

```

```

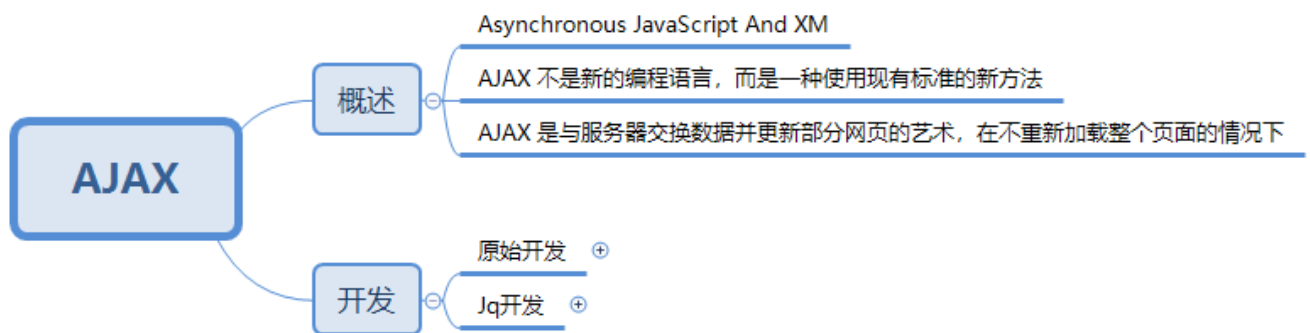
13     HttpSession session = se.getSession();
14     ServletContext context = session.getServletContext();
15     Integer count= (Integer) context.getAttribute("count");
16     if(count==null){
17         context.setAttribute("count",1);
18     }else{
19         count++;
20         context.setAttribute("count",count);
21     }
22     lockCreated.unlock();
23 }
24
25 @Override
26 public void sessionDestroyed(HttpSessionEvent se) {
27     lockDestroyed.lock();
28     HttpSession session = se.getSession();
29     ServletContext context = session.getServletContext();
30     Integer count= (Integer) context.getAttribute("count");
31     if(count==null){
32         context.setAttribute("count",1);
33     }else{
34         count--;
35         context.setAttribute("count",count);
36     }
37     lockDestroyed.unlock();
38 }
39 }
40
41 * 在首页加上
42 * <h1 style="color: maroon;">当前在线人数: ${count }</h1>

```

当前在线人数：3

亮哥教育,做教育我们是认真的。

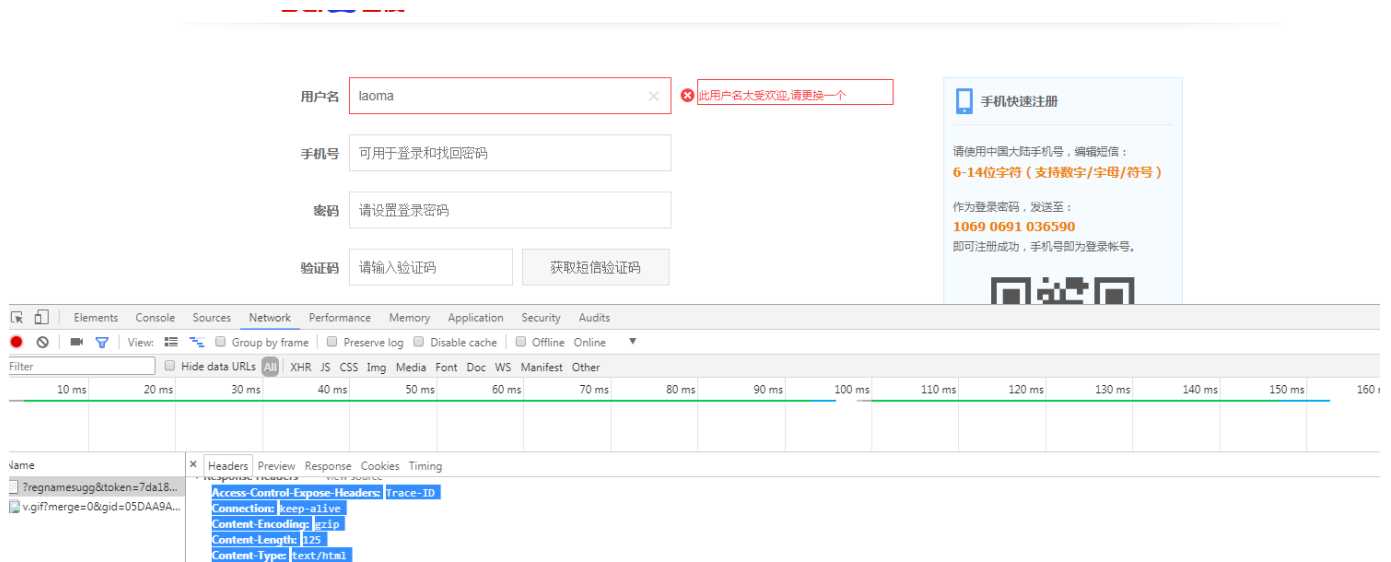
* 能够掌握Ajax的概述



xiaohei

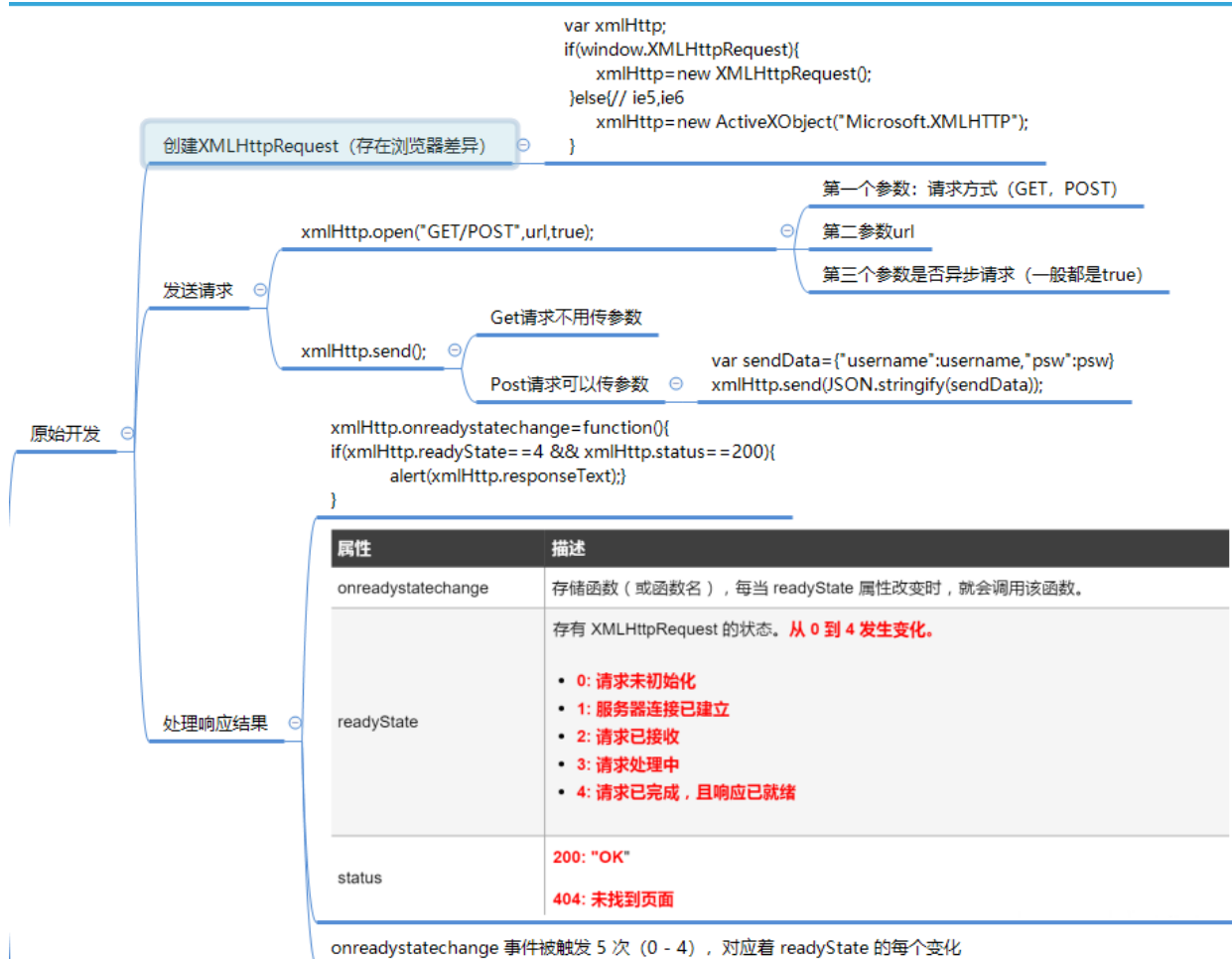
注册

校验规则，注册用户已经存在（文本框失去焦点的时候），部分更新网页技术



* 能够掌握Ajax的开发

* 原始开发



1 * 案例

2 * 注册页面

3 * Get方式提交

4 <!DOCTYPE html>

```
5 <html lang="en">
6 <head>
7   <meta charset="UTF-8">
8   <title>注册</title>
9   <script>
10     function checkName() {
11       // 1 创建XMLHttpRequest (存在浏览器差异)
12       var xmlhttp;
13       if(window.XMLHttpRequest){
14         xmlhttp=new XMLHttpRequest();
15       }else{//ie5,ie6
16         xmlhttp=new ActiveXObject("Microsoft.XMLHTTP")
17       }
18       // 2 发送请求
19       var param=document.getElementById("iname").value;
20       var url="/lgweb02/reg?username="+param;
21       xmlhttp.open("GET",url,true);
22       xmlhttp.send();
23       // 3 处理结果
24       xmlhttp.onreadystatechange=function (ev) {
25         if(xmlhttp.readyState==4 && xmlhttp.status==200){
26           document.getElementById("sname").innerHTML=xmlhttp.response
27         }
28       }
29     }
30   </script>
31 </head>
32 <body>
33   <form action="/lgweb02/reg" method="get">
34     <input id="iname" type="text" name="username" onblur="checkName();" />&r
35     <span id="sname"></span>
36   </form>
37 </body>
38 </html>
39
40 * Post 提交
41 <!DOCTYPE html>
42 <html lang="en">
43 <head>
44   <meta charset="UTF-8">
```

```

45     <title>注册</title>
46     <script>
47         function checkName() {
48             // 1 创建XMLHttpRequest（存在浏览器差异）
49             var xmlHttp;
50             if(window.XMLHttpRequest){
51                 xmlHttp=new XMLHttpRequest();
52             }else{//ie5,ie6
53                 xmlHttp=new ActiveXObject("Microsoft.XMLHTTP")
54             }
55             // 2 发送请求
56             var param="username="+document.getElementById("iname").value;
57             var url="/lgweb02/reg";
58             xmlHttp.open("POST",url,true);
59             xmlHttp.setRequestHeader("content-type","application/x-www-form-urlencoded");
60             xmlHttp.send(param);
61             // 3 处理结果
62             xmlHttp.onreadystatechange=function (ev) {
63                 if(xmlHttp.readyState==4 && xmlHttp.status==200){
64                     document.getElementById("sname").innerHTML+xmlHttp.responseText;
65                 }
66             }
67         }
68     </script>
69 </head>
70 <body>
71     <form action="/lgweb02/reg" method="post">
72         <input id="iname" type="text" name="username" onblur="checkName();"/>&#20135;
73         <span id="sname"></span>
74     </form>
75 </body>
76 </html>
77 * Servlet
78 @WebServlet("/reg")
79 public class RegServlet extends HttpServlet {
80     @Override
81     protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
82         String username = req.getParameter("username");
83         if ("xiaohai".equals(username)) {
84             resp.getWriter().write("<font color='red'>您注册用户名，太受欢迎</font>");
85         } else {
86             resp.getWriter().write("<font color='red'>用户名或密码错误，请重新输入</font>");
87         }
88     }
89 }

```



```

85         } else {
86             resp.getWriter().write("<font color='green'>可以放心注册</font>");
87         }
88     }
89
90     @Override
91     protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
92         doGet(req, resp);
93     }
94 }
95
96 * 案例二（JSON数据提交和响应数据）：
97 <!DOCTYPE html>
98 <html lang="en">
99 <head>
100     <meta charset="UTF-8">
101     <title>注册</title>
102     <script>
103         function checkName() {
104             // 1 创建XMLHttpRequest（存在浏览器差异）
105             var xmlHttp;
106             if(window.XMLHttpRequest){
107                 xmlHttp=new XMLHttpRequest();
108             }else{//ie5,ie6
109                 xmlHttp=new ActiveXObject("Microsoft.XMLHTTP")
110             }
111             // 2 发送请求
112             var username=document.getElementById("iname").value;
113             var sendData={"username":username}
114             var url="/lgweb02/check";
115             xmlHttp.open("POST",url,true);
116             xmlHttp.setRequestHeader("content-type","application/json");
117             xmlHttp.send(JSON.stringify(sendData));
118             // 3 处理结果
119             xmlHttp.onreadystatechange=function (ev) {
120                 if(xmlHttp.readyState==4 && xmlHttp.status==200){
121                     var content= JSON.parse(xmlHttp.responseText);
122                     document.getElementById("sname").innerHTML=content.msg;
123                 }
124             }

```

```

125     }
126     </script>
127 </head>
128 <body>
129     <form action="/lgweb02/reg" method="post">
130         <input id="iname" type="text" name="username" onblur="checkName();" />&r
131         <span id="sname"></span>
132     </form>
133 </body>
134 </html>
135
136 * 代码
137 @Data
138 @AllArgsConstructor
139 @NoArgsConstructor
140 public class RespResult {
141     private Object msg;
142     private int code;
143 }
144
145 @WebServlet("/check")
146 public class ChekcServlet extends HttpServlet {
147     @Override
148     protected void doGet(HttpServletRequest req, HttpServletResponse resp) thro
149         resp.setContentType("application/json;charset=utf-8");
150         // 获取json数据
151         ServletInputStream is = req.getInputStream();
152         byte[] buffer=new byte[1024];
153         StringBuilder sb=new StringBuilder();
154         while((is.read(buffer))!=-1) {
155             sb.append(new String(buffer));
156         }
157         String json=sb.toString().trim();
158         System.out.println(json);
159         Gson gson=new Gson();
160         User user=gson.fromJson(json, User.class);
161         RespResult result=new RespResult();
162         if ("xiaohei".equals(user.getUsername())) {
163             result.setCode(401); // 401代表用户名已经存在
164             result.setMsg("<font color='red'>您注册用户名，太受欢迎</font>");

```

```

165         } else {
166             result.setCode(200); // 200 代表用户不存在
167             result.setMsg("<font color='green'>可以放心注册</font>");
168         }
169         String resultStr = gson.toJson(result);
170         resp.getWriter().write(resultStr);
171     }
172
173     @Override
174     protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException {
175         doGet(req, resp);
176     }
177 }
178

```

* 效果图

可以放心注册

您注册用户名，太受欢迎

* JQ开发



```

1  * 案例
2  * 引入jquery库
3  * Get的请求
4  <!DOCTYPE html>
5  <html lang="en">
6  <head>
7      <meta charset="UTF-8">
8      <title>注册</title>
9      <script src="/lgweb02/js/jquery-3.3.1.min.js"></script>
10     <script>
11         $(function () {
12             $("#iname").blur(function () {
13                 var param=$("#iname").val();
14                 var url="/lgweb02/reg?username="+param;
15                 $.get(url,function (data) {
16                     $("#sname").html(data);
17                 });
18             });
19         })
20     </script>
21 </head>

```

[illegible]

员工编号	员工姓名	薪水
1	xiaobai1	2001
2	xiaobai2	2002
3	xiaobai3	2003
4	xiaobai4	2004
5	xiaobai5	2005
6	xiaobai6	2006
7	xiaobai7	2007
8	xiaobai8	2008
9	xiaobai9	2009
10	xiaobai10	2010

首页

上一页

1

2

3

4

5

6

7

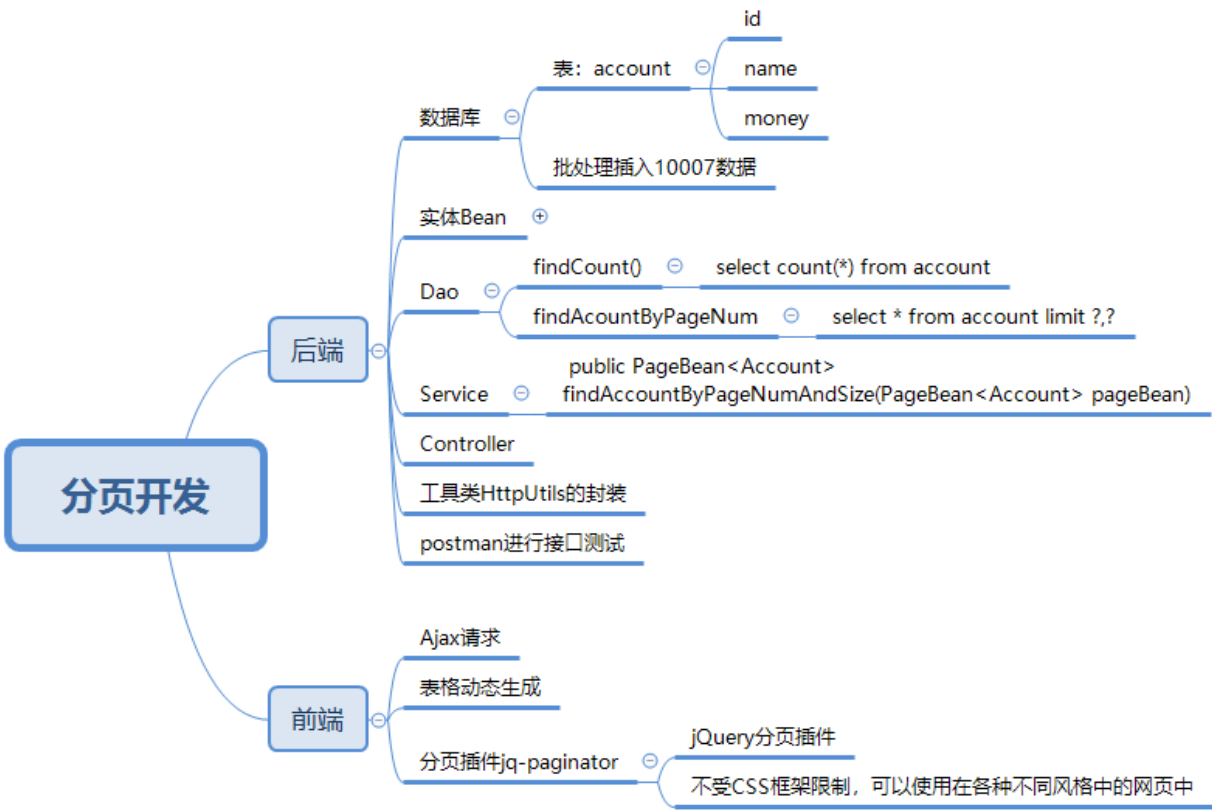
8

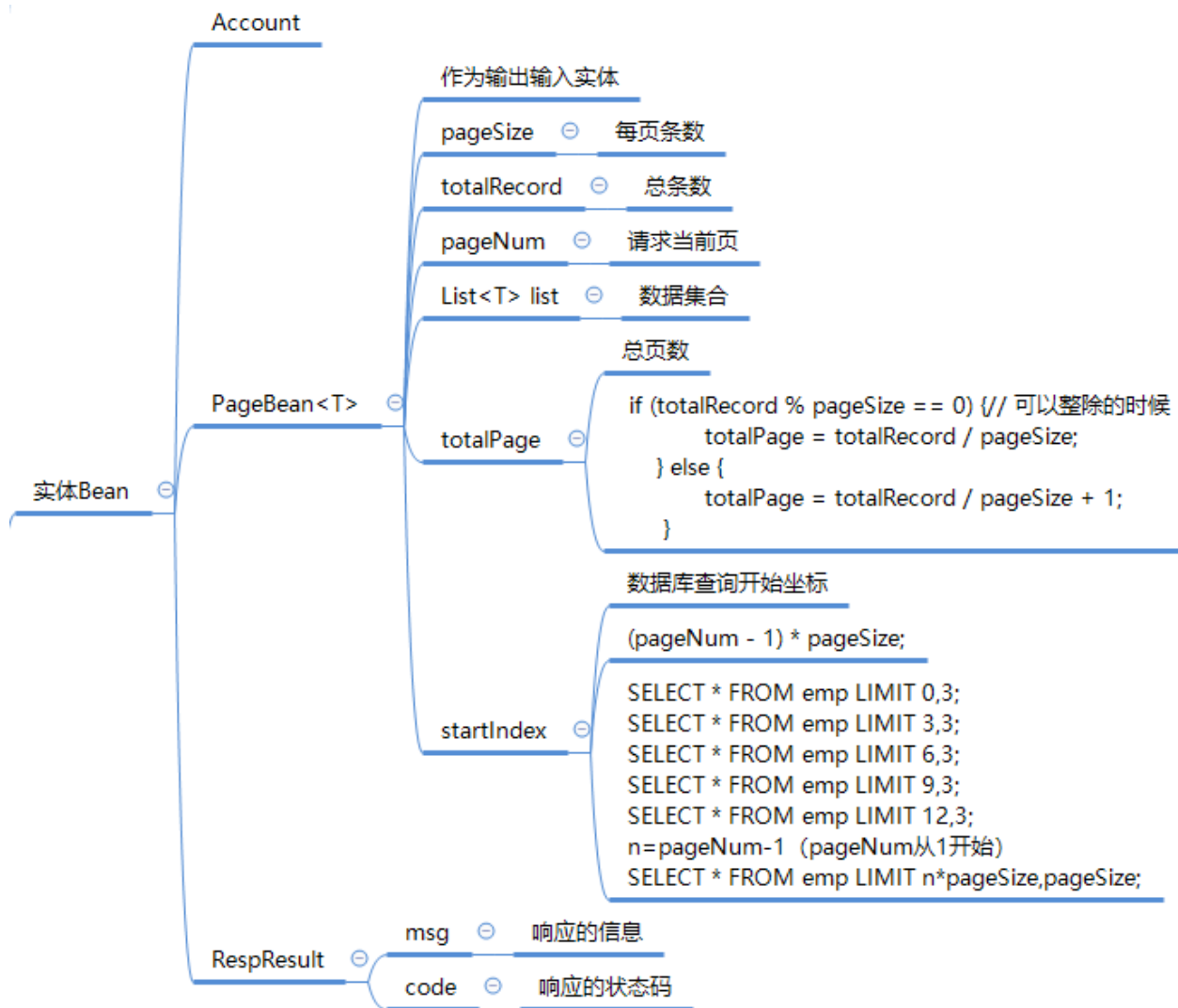
9

10

下一页

末页





```

1 * 开发案例
2 * Account表的创建 (id,name,money), 批量插入数据
3 * 后端:
4 * 持久层
5 @Data
6 @AllArgsConstructor
7 @NoArgsConstructor
8 public class Account {
9     private int id;
10    private String name;
11    private double money;
12 }
13 public interface AccountDao {
14     // 查询所有记录
15     int findCount() throws SQLException;
16     // 根据下标和每页条数, 查询账户列表
17     List<Account> findAccountByPageNum(int startIndex, int pageSize) throws SQL
  
```



```

18 }
19 public class AccountDaoImpl implements AccountDao {
20     @Override
21     public int findCount() throws SQLException {
22         QueryRunner runner=new QueryRunner(DataSourceUtils.getDataSource());
23         String sql = "select count(*) from account";
24         Long count = runner.query(sql, new ScalarHandler<Long>());
25         return count.intValue();
26     }
27
28     @Override
29     public List<Account> findAccountByPageNum(int startIndex, int pageSize) throws
30         QueryRunner runner=new QueryRunner(DataSourceUtils.getDataSource());
31         String sql="select * from account limit ?,?";
32         List<Account> accounts = runner.query(sql, new BeanListHandler<Account>());
33         return accounts;
34     }
35 }
36 public class AccountTest {
37     @Test
38     public void test1() throws SQLException {
39         AccountDao accountDao=new AccountDaoImpl();
40         int count = accountDao.findCount();
41         System.out.println(count);
42     }
43     @Test
44     public void test2() throws SQLException {
45         AccountDao accountDao=new AccountDaoImpl();
46         List<Account> accounts = accountDao.findAccountByPageNum(0, 10);
47         System.out.println(accounts);
48     }
49
50 }
51 * 服务层
52 @Data
53 @NoArgsConstructor
54 public class PageBean<T> {
55     // 已知
56     // 每页条数: pageSize
57     private int pageSize;

```

```

58 // 总条数: totalRecord
59 private int totalRecord;
60 // 请求当前页: pageNum
61 private int pageNum;
62 // 集合
63 private List<T> list;
64 // 计算出来
65 // 总页数: totalPage
66 private int totalPage;
67 // 起始坐标: startIndex
68 private int startIndex;
69 public void setTotalRecord(int totalRecord) {
70     this.totalRecord = totalRecord;
71     // * 总页数: totalPage
72     // * 当可以整除的时候: totalPage=totalRecord/pageSize
73     // * 当不可以整除的时候: totalPage=totalRecord/pageSize+1
74     if (totalRecord % pageSize == 0) { // 可以整除的时候
75         totalPage = totalRecord / pageSize;
76     } else {
77         totalPage = totalRecord / pageSize + 1;
78     }
79 }
80 public int getStartIndex(){
81     return (pageNum - 1) * pageSize;
82 }
83 }
84 public interface AccountService {
85     PageBean<Account> findAccountByPageNumAndSize(PageBean<Account> pageBean)
86 }
87 public class AccountServiceImpl implements AccountService {
88     @Override
89     public PageBean<Account> findAccountByPageNumAndSize(PageBean<Account> pageBean) {
90         AccountDao accountDao=new AccountDaoImpl();
91         pageBean.setTotalRecord(accountDao.findCount());
92         pageBean.setList(accountDao.findAccountByPageNum(pageBean.getStartIndex(), pageBean.getPageSize()));
93         return pageBean;
94     }
95 }
96 * 控制层
97 @Data

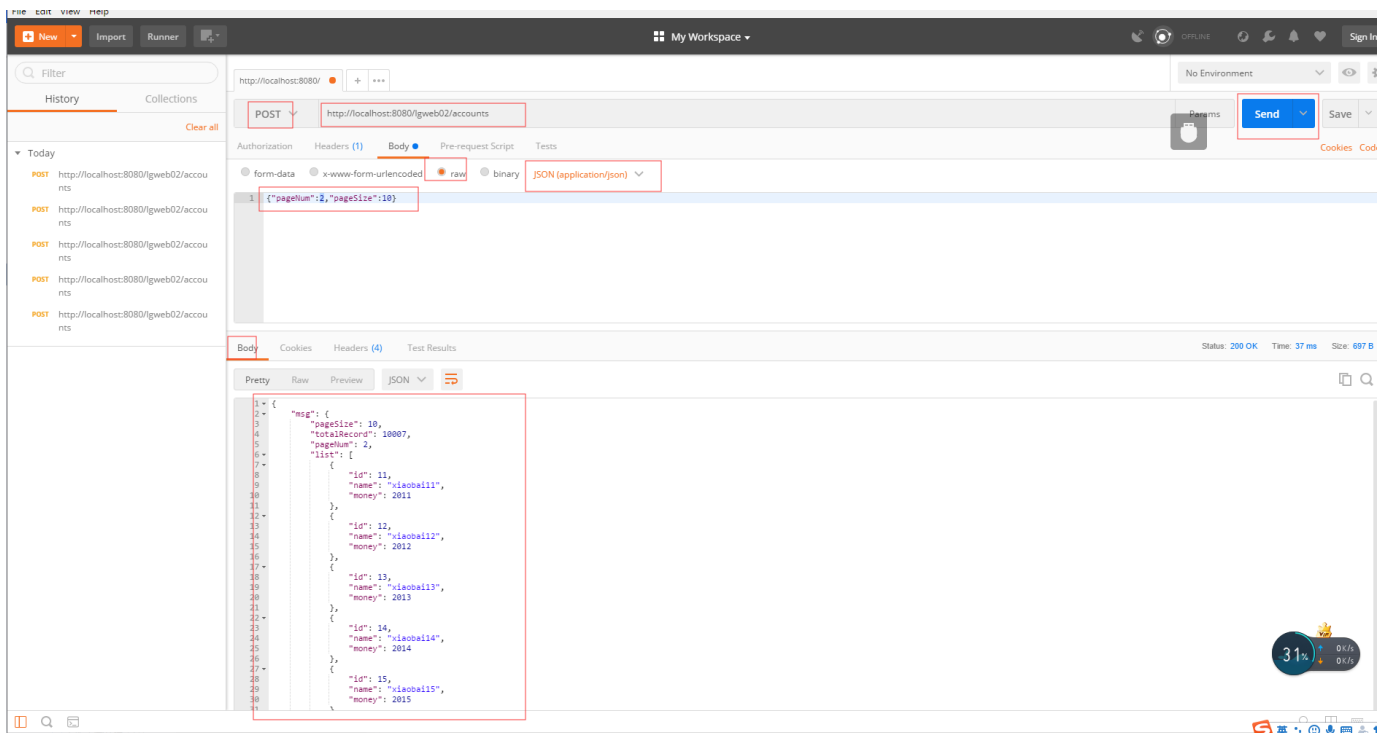
```

```

98 @AllArgsConstructor
99 @NoArgsConstructor
100 public class RespResult {
101     private Object msg;
102     private int code;
103 }
104 public class HttpUtils<T> {
105     public String getJson(HttpServletRequest req, HttpServletResponse resp) throws IOException {
106         resp.setContentType("application/json;charset=utf-8");
107         // 获取json数据
108         ServletInputStream is = req.getInputStream();
109         byte[] buffer=new byte[1024];
110         StringBuilder sb=new StringBuilder();
111         while((is.read(buffer))!=-1) {
112             sb.append(new String(buffer));
113         }
114         String json=sb.toString().trim();
115         return json;
116     }
117     public T getBean(HttpServletRequest req, HttpServletResponse resp,Class<T> clazz) {
118         String json = getJson(req, resp);
119         Gson gson=new Gson();
120         T t = gson.fromJson(json, clazz);
121         return t;
122     }
123
124     public T getBean(HttpServletRequest req, HttpServletResponse resp,Type type) {
125         String json = getJson(req, resp);
126         Gson gson=new Gson();
127         T t = gson.fromJson(json, type);
128         return t;
129     }
130
131     public String toJson(Object obj){
132         return new Gson().toJson(obj);
133     }
134 }
135
136 @WebServlet("/accounts")
137 public class AccountServlet extends HttpServlet {

```

```
138  @Override
139  protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
140      HttpUtils<PageBean<Account>> httpUtils=new HttpUtils<>();
141      PageBean<Account> pageBean = httpUtils.getBean(req, resp, new TypeToken<
142      AccountService accountService=new AccountServiceImpl();
143      RespResult result=new RespResult();
144      try {
145          PageBean<Account> page = accountService.findAccountByPageNumAndSize
146          result.setCode(200);
147          result.setMsg(page);
148      } catch (SQLException e) {
149          e.printStackTrace();
150          result.setCode(500);
151          result.setMsg("服务器出错");
152      }
153      String resultStr = httpUtils.toJson(result);
154      resp.getWriter().write(resultStr);
155  }
156
157  @Override
158  protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
159      doGet(req, resp);
160  }
161 }
162 * 使用postman测试接口
```



```
1 * 前端开发
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5   <meta charset="UTF-8">
6   <title>分页显示案例</title>
7   <link rel="stylesheet" href="/lgweb02/css/bootstrap.css">
8   <script src="/lgweb02/js/jquery-3.3.1.js"></script>
9   <script src="/lgweb02/js/jq-paginator.js"></script>
10  <script src="/lgweb02/js/popper.min.js"></script>
11  <script src="/lgweb02/js/bootstrap.js"></script>
12  <script>
13    var url="/lgweb02/accounts";
14    var sendData={"pageNum":1,"pageSize":10};
15    $(function () {
16      // 当前页: 1, 第一次发送
17      sendRequest(1,true)
18    });
19    function sendRequest(num,isfirst) {
20      sendData.pageNum=num;
21      $.ajax({
22        type:"POST",
```

```

23     url:url,
24     contentType:"application/json;charset=utf-8",
25     dataType:"json",
26     data:JSON.stringify(sendData),
27     processData:false,//默认是true, 是否转换为查询字符串
28     success:function(result,status){
29         var content = JSON.parse(JSON.stringify(result));
30         if(content.code==200){
31             console.log(content.msg.totalRecord);
32             var tbody=document.getElementsByTagName("tbody")[0];
33             tbody.innerHTML="";
34             for(var index in content.msg.list){
35                 var account=content.msg.list[index];
36                 console.log(account);
37                 tbody.innerHTML+="|\n" +
38                     "                <td>"+account.id+"</td>\n" +
39                     "                <td>"+account.name+"</td>\n" +
40                     "                <td>"+account.money+"</td>\n"
41                     "                </tr>";
42             }
43             if(isfirst){
44                 $('#pagination1').jqPaginator({
45                     totalPages: content.msg.totalPage,
46                     visiblePages: 10,
47                     currentPage: num,
48                     first:'<li class="page-item"><a class="page-lin
49                     prev: '<li class="page-item"><a class="page-lin
50                     next: '<li class="page-item"><a class="page-lin
51                     last: '<li class="page-item"><a class="page-lin
52                     page: '<li class="page-item"><a class="page-lin
53                     onPageChange: function (num) {
54                         sendRequest(num,false);
55                     }
56                 });
57             }
58         }
59     },
60     error:function(result){
61         alert("请求数据失败")
62     }

|  |

```

```
63         });
64     }
65     </script>
66 </head>
67 <body>
68     <div class="container-fluid">
69         <h1>亮哥教育</h1>
70         <table class="table table-bordered table-striped">
71             <thead>
72                 <tr>
73                     <th>员工编号</th>
74                     <th>员工姓名</th>
75                     <th>薪水</th>
76                 </tr>
77             </thead>
78             <tbody>
79                 <!--<tr>-->
80                 <!--<td>100001</td>-->
81                 <!--<td>刘备</td>-->
82                 <!--<td>2000</td>-->
83                 <!--</tr>-->
84             </tbody>
85         </table>
86         <ul class="pagination" id="pagination1">
87
88         </ul>
89     </div>
90 </body>
91 </html>
92
```

亮哥教育

员工编号	员工姓名	薪水
9991	xiaobai9991	11991
9992	xiaobai9992	11992
9993	xiaobai9993	11993
9994	xiaobai9994	11994
9995	xiaobai9995	11995
9996	xiaobai9996	11996
9997	xiaobai9997	11997
9998	xiaobai9998	11998
9999	xiaobai9999	11999
10000	xiaobai10000	12000

首页

上一页

992

993

994

995

996

997

998

999

1000

1001

下一页

末页