

第三天

了解Servlet运行和tomcat容器创建，关于xml语法的讲解。

```
1 //写在<web-app>中。
2 <servlet>
3     <servlet-name>hello</servlet-name>
4     //servlet名
5     <servlet-class>web.HelloServlet</servlet-class>
6     //反射的Java类，指定到本地文件位置。
7 </servlet>
8 <servlet-mapping>
9     <servlet-name>hello</servlet-name>
10    <url-pattern>/hello</url-pattern>
11    //导向到对应网址下
12 </servlet-mapping>
13
```

接受前端浏览器传递的参数：

```
1 request.getParameter()//request对象，请求对象
```

参数传递的方式，浏览器传递给服务器。

通讯协议

协议：TCP、IP、UDP

http协议(超文本传输协议)--应用层协议，定义了web和服务之间的通讯。

数据包的结构：请求数据包结构（浏览器->服务器）

1. 请求行（数据包第一行）

- a) 请求方式（get/post） [刷题记录](#)（和之前的CTF-web有关联）
- b) 请求资源路径
- c) 协议的类型和版本、

2. 若干消息头（www命名的一些有特殊含义的键值对如：content-type= text/html）服务器和浏览器都会遵守这些消息头，消息头一般由浏览器和服务器自动生成，也可以手动编程。

响应数据包结构：第一部分：状态行、协议的类型与版本、状态码（数字）404、200、500.

第二部分：若干消息头

第三部分：实体内容，服务器返回给浏览器的处理结果。

输出传输中的中文乱码问题（重点）

(ASCII) 当表单出现乱码时，在html文档中添加，背后的原因是因为编码方式不同。

```
1 <meta http-equiv="content-type"
2 content="text/html;charset=utf-8">
```

在servlet添加:

```
1 request.setCharacterEncoding("utf-8")
```

Java连接数据库类型

```
1 Statement
2 PreparedStatement
3 //实战PreparedStatement比Statement更安全可以有效的防止SQL注入攻击
4 //而且PreparedStatement比Statement运行的效率
```

Statement实例:

```
1 import java.sql.Connection;
2 import java.sql.ResultSet;
3 import java.sql.Statement;
4 import java.util.Scanner;
5
6 public class DBUtilTest {
7     public static void main(String[] args) {
8         //System.out.println(DBUtil.getConnection());
9         //DBUtil是JDBC中的一个工具
10        Scanner sc=new Scanner(System.in);
11        //获取用户名
12        String name=sc.nextLine();
13        //获取用户密码
14        String pwd=sc.nextLine();
15        //System.out.println(name+":"+pwd);
16        //获取数据库链接
17        Connection conn=DBUtil.getConnection();
18        try {
```

```

19         //获取数据库语句操作对象
20         Statement state=conn.createStatement();
21         //定义sql语句
22         String sql="select * from emp where uname='"+name+"' and pwd='"+pwd+"'";
23         //让Statement对象向数据库发送上面的sql语句 并获取查询语句 查询的结果集对象
24         ResultSet rs=state.executeQuery(sql);
25         //从结果及中查询出对应的数据信息
26         //判断是否查询到了结果
27         while(rs.next()){
28             System.out.println("有结果");
29         }
30     } catch (Exception e) {
31         // TODO Auto-generated catch block
32         e.printStackTrace();
33     }
34 }
35 }

```

PreparedStatement实例

```

1  import java.sql.Connection;
2  import java.sql.PreparedStatement;
3  import java.sql.ResultSet;
4  import java.sql.SQLException;
5  import java.util.Scanner;
6
7  public class DBUtilTest2 {
8      public static void main(String[] args) {
9          Scanner sc=new Scanner(System.in);
10         //获取用户名
11         String name=sc.nextLine();
12         //获取用户密码
13         String pwd=sc.nextLine();
14         //System.out.println(name+":"+pwd);
15         //获取数据库链接
16         Connection conn=DBUtil.getConnection();
17         //定义sql语句
18         String sql="select * from emp where uname=? and pwd=?";
19         //获取数据库语句操作对象PreparedStatement

```

```
20     try {
21         PreparedStatement prep=conn.prepareStatement(sql);
22         //为两个占位符传递参数
23         prep.setString(1, name);
24         prep.setString(2, pwd);
25         //PreparedStatement 向数据库发送sql语句并执行  获取返回的结果集对象
26         ResultSet rs=prep.executeQuery();
27         while(rs.next()){
28             System.out.println("有数据");
29         }
30     } catch (Exception e) {
31         // TODO Auto-generated catch block
32         e.printStackTrace();
33     }
34 }
35 }
```