



第4章 分布式软件体系结构风格

4.5 浏览器/服务器风格 (Browser/Server Architecture)



刘其成

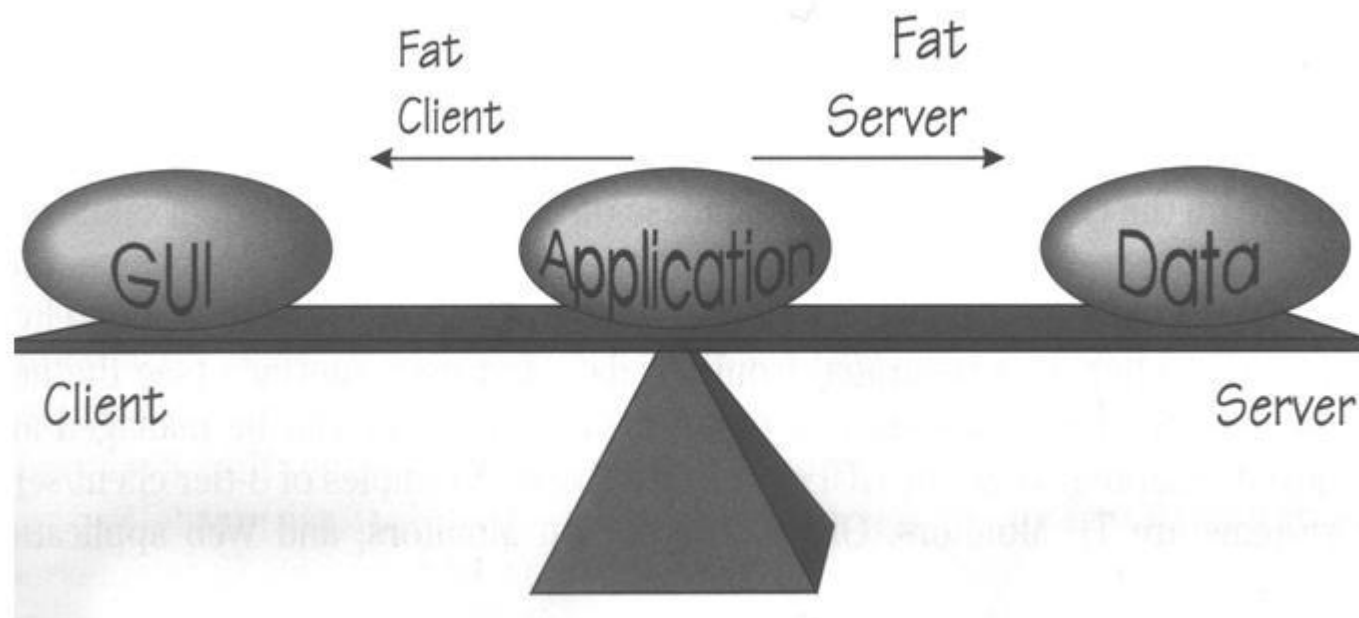
计算机与控制工程学院

ytliuqc@163.com

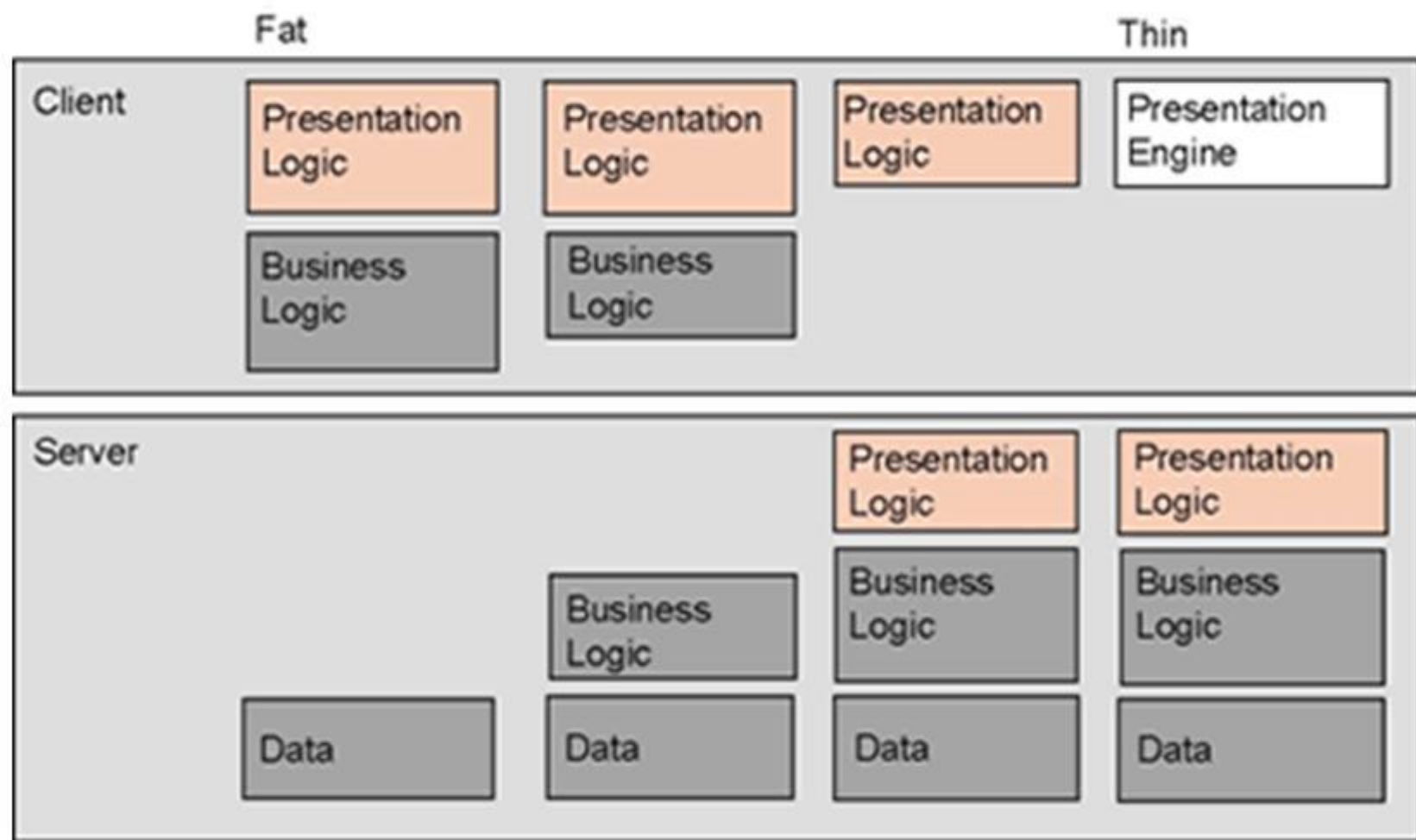
2018-09

胖客户端还是瘦客户端

- 业务逻辑的划分比重：在客户端多一些还是在服务器端多一些？
 - 胖客户端：客户端执行大部分的数据处理操作
 - 瘦客户端：客户端具有很少或没有业务逻辑



胖客户端还是瘦客户端 考虑二者的优缺点？



胖客户端还是瘦客户端 考虑二者的优缺点？

胖客户端	瘦客户端
较低的网络带宽	管理成本低，升级容易
较低的服务器需求	安全
更好的计算性能	客户端价值低

- **B/S**体系结构主要是利用不断成熟的**WWW**浏览器技术，结合浏览器的多种脚本语言，用通用浏览器就实现了原来需要复杂的专用软件才能实现的强大功能，并节约了开发成本。
- 在三层**C/S**体系结构中，
 - 表示层负责处理用户的输入和向客户的输出(出于效率的考虑，它可能在向上传输用户的输入前进行合法性验证)。
 - 功能层负责建立数据库的连接，根据用户的请求生成访问数据库的**SQL**语句，并把结果返回给客户端。
 - 数据层负责实际的数据库存储和检索，响应功能层的数据处理请求，并将结果返回给功能层。
- **浏览器/服务器(B/S)**是三层**C/S**风格的一种实现方式。

浏览器/服务器

- **浏览器/服务器(B/S)**是三层**C/S**风格的一种实现方式

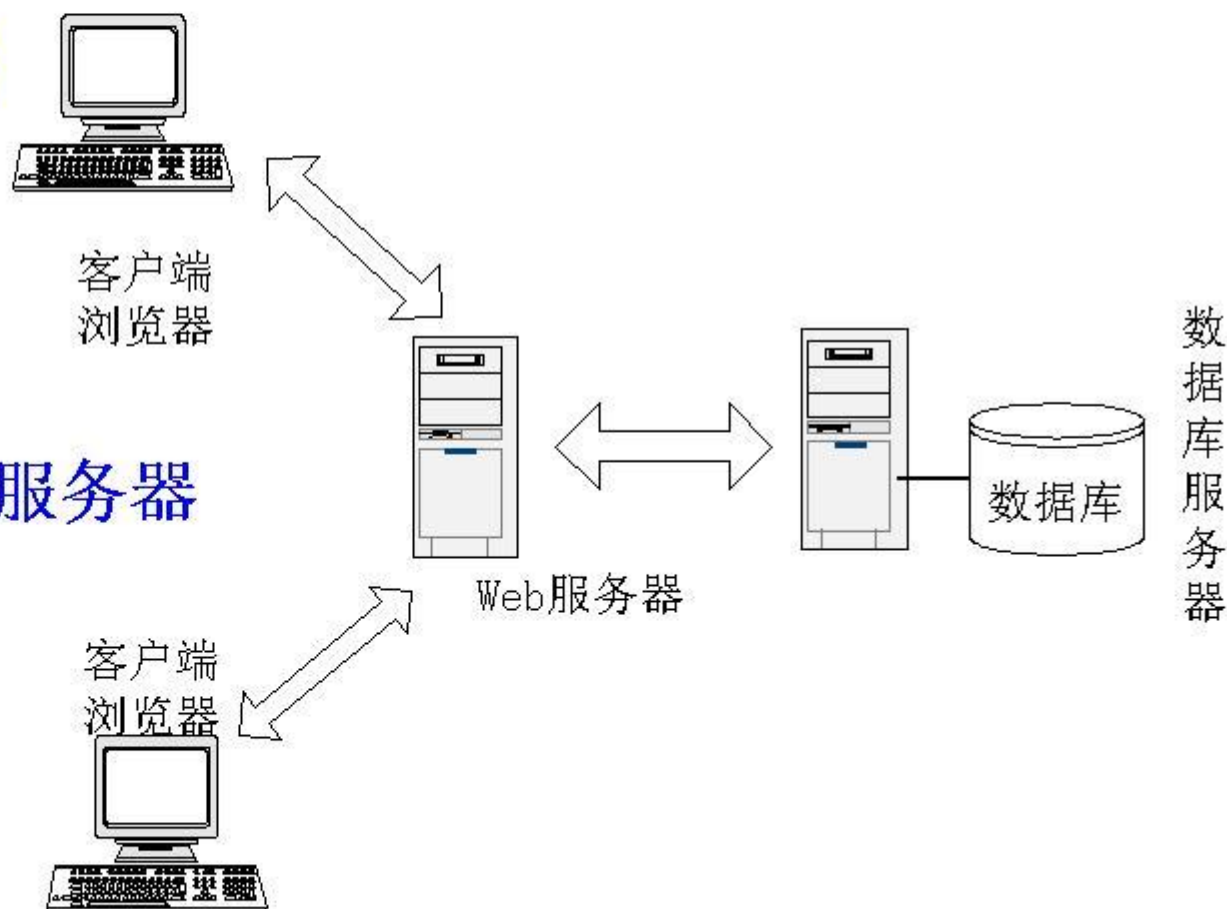
—表现层：浏览器

—逻辑层：

- **Web服务器**

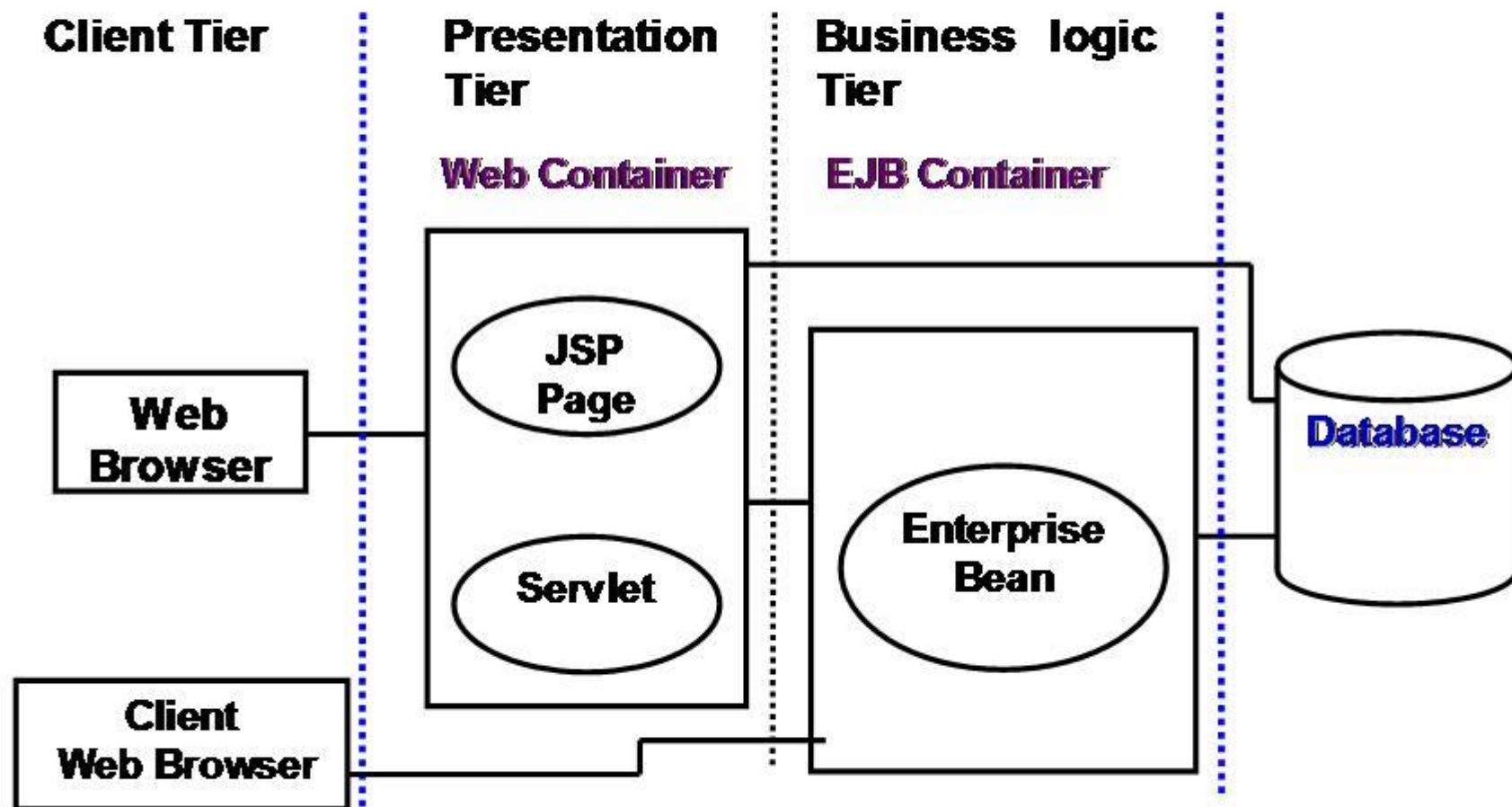
- **应用服务器**

—数据层：数据库服务器



- 在**B/S**结构中，应用程序以网页形式存放于**Web**服务器上，用户运行某个应用程序时，只要在**客户端**上的**浏览器**中**键入**相应的**网址(URL)**，**调用Web服务器上的应用程序并对数据库进行操作**，完成相应的数据处理工作，最后将**结果通过浏览器显示给用户**。
- 浏览器只有在**接收到用户请求后**，**才和Web服务器进行连接**，**Web服务器**马上与数据库通信并生成结果，然后**Web服务器**再把数据库的返回结果转发给浏览器，**浏览器在收到返回信息后断开连接**。
- 由于**真正的连接时间很短**，因此**Web服务器**能够为更多的用户提供服务。

J2EE平台典型B/S结构的实现方式



B/S结构的优点(一)

- 基于B/S体系结构的软件，系统安装、修改和维护全在服务器端解决，**系统维护成本低**：
 - **客户端无任何业务逻辑**，用户在使用系统时，仅仅需要一个浏览器就可运行全部的模块，真正达到了“零客户端”的功能，很容易在运行时自动升级。
 - **良好的灵活性和可扩展性**：对于环境和应用条件经常变动的情况，只要对业务逻辑层实施相应的改变，就能够达到目的。
- B/S体系结构还提供了**异种机、异种网、异种应用服务的联机、联网、统一服务**的最现实的开放性基础。

B/S结构的优点(二)


- **较好的安全性**：在这种结构中，客户应用程序不能直接访问数据，应用服务器不仅可控制哪些数据被改变和被访问，而且还可控制数据的改变和访问方式。
- 三层模式成为真正意义上的“**瘦客户端**”，从而具备了很高的稳定性、延展性和执行效率。
- 三层模式可以将服务集中在一起管理，统一服务于客户端，从而具备了良好的**容错能力**和**负载平衡能力**。

B/S结构的优点(三)

- 扩大了组织计算机应用系统功能覆盖范围，可以更加充分利用网络上的各种资源，同时应用程序维护的工作量也大大减少
 - B/S结构出现之前，管理信息系统的功能覆盖范围主要是组织内部。
 - B/S结构“零客户端”方式使组织的供应商和客户（这些供应商和客户有可能是潜在的，也就是说可能是事先未知的）的计算机方便地成为管理信息系统的客户端，进而在限定的功能范围内查询组织相关信息，完成与组织的各种业务往来的数据交换和处理工作。
- B/S结构的计算机应用系统与Internet的结合也使新近提出的一些新的企业计算机应用(如电子商务，客户关系管理)的实现成为可能。

B/S结构的缺点

- 客户端浏览器以同步的请求/响应模式交换数据，每请求一次服务器就要刷新一次页面；
- 受HTTP协议“基于文本的数据交换”的限制，在数据查询等响应速度上，要远远低于C/S体系结构；
- 数据提交一般以页面为单位，数据的动态交互性不强，不利于在线事务处理(OLTP)应用；
- 受限于HTML的表达能力，难以支持复杂GUI (如报表等)。

- 
- 因此，虽然**B/S**结构的计算机应用系统有如此多的优越性，但由于**C/S**结构的成熟性且**C/S**结构的计算机应用系统网络负载较小。
 - 因此未来一段时间内将是**B/S**结构和**C/S**结构共存的情况。
 - 但计算机应用系统计算模式的发展趋势是向**B/S**结构转变。

AJAX

- 传统的**Web**应用允许用户端填写表单(**form**)，当提交表单时就向**Web**服务器发送一个请求。服务器接收并处理传来的表单，然后送回一个新的网页。
- 这个做法浪费了许多带宽，因为在前后两个页面中的大部分**HTML**代码往往是相同的。由于每次应用的交互都需要向服务器发送请求，应用的响应时间就依赖于服务器的响应时间。
- 这导致了用户界面的响应比本地应用慢得多。

AJAX

- 与此不同，**AJAX**应用可以**仅向服务器发送并取回必需的数据**，它使用**SOAP**或其它一些基于**XML**的页面服务接口，并在客户端采用**JavaScript**处理来自服务器的响应。
- 因为在**服务器和浏览器之间交换的数据**大量**减少**（大约只有原来的**5%**），结果就能看到**响应更快的应用**。
- 同时**很多的处理工作**可以在发出请求的**客户端机器上完成**，所以**Web服务器的处理时间也减少**了。

AJAX

- 对应用**Ajax**最主要的批评就是，它可能**破坏浏览器后退按钮的正常行为**。
- 在动态更新页面的情况下，**用户无法回到前一个页面状态**，这是因为浏览器仅能记下历史记录中的静态页面。
- 而不是一个已经被动态修改过的页面

AJAX

- 不过开发者已想出了种种办法来解决这个问题，当中大多数都是在用户单击后退按钮访问历史记录时，通过建立或使用一个隐藏的IFRAME来重现页面上的变更。
- 例如，当用户在 **Google Maps** 中单击后退时，它在一个隐藏的IFRAME中进行搜索，然后将搜索结果反映到**Ajax**元素上，以便将应用程序状态恢复到当时的状态。

- 
- JSP
 - ASP
 - PHP


1. JSP直接连接数据库的程序


- `<%@page contentType="text/html;charset=gb2312"%>`
- `<%@page import="java.sql.*"%>`
- `<%`
- `Connection conn = null;`
- `Statement stmt = null;`
- `ResultSet rs = null;`
- `try{`
- `Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");`
- `}catch(ClassNotFoundException ce){`
- `out.println(ce.getMessage());`
- `}`

- try{
- conn=DriverManager.getConnection("jdbc:odbc:myDB","liu","123");
- stmt=conn.createStatement();
- rs=stmt.executeQuery("SELECT * FROM chengjibiao");
- while(rs.next()){
- out.print(rs.getString(1)+" ");
- out.print(rs.getString(2)+" ");
- out.print(rs.getString(3)+" ");
- out.print(rs.getInt(4)+" ");
- out.print(rs.getInt(5)+" ");
- out.print("
");
- }
- }catch(SQLException e){
- System.out.println(e.getMessage());
- }finally{
- stmt.close();
- conn.close();
- }
- %>
- 上面程序的存放位置为tomcat安装目录的webapps\ROOT子目录中。

2. JSP+JavaBean直接连接数据库的程序

- (1) JSP程序如下
- `<%@page contentType="text/html;charset=gb2312"%>`
- `<%@page import="java.sql.*"%>`
- `<%@page import="db.dbConn"%>`
- `<jsp:useBean id="connDbBean" class="db.dbConn" scope="page">`
- `</jsp:useBean>`
- `<%`
- `ResultSet rs=connDbBean.executeQuery(`
- `"SELECT * FROM chengjibiao");`
- `while(rs.next()){`
- `out.print(rs.getString(1)+" ");`
- `out.print(rs.getString(2)+" ");`
- `out.print(rs.getString(3)+" ");`
- `out.print(rs.getInt(4)+" ");`
- `out.print(rs.getInt(5));`
- `out.print("
");`
- `}`
- `%>`

- 
- (2) **JavaBean**程序如下
 - **package db;**
 - **import java.sql.*;**
 - **public class dbConn {**
 - **String sDBDriver = "sun.jdbc.odbc.JdbcOdbcDriver";**
 - **String sConnStr = "jdbc:odbc:myDB";**
 - **Connection conn = null;**
 - **ResultSet rs = null;**




```
▪ public dbConn() {  
▪     try {  
▪         Class.forName(sDBDriver);  
▪     }catch(java.lang.ClassNotFoundException e) {  
▪         System.err.println( e.getMessage());  
▪     }  
▪ }
```


- ```
public ResultSet executeQuery(String sql) {
 try {
 conn = DriverManager.getConnection(sConnStr,"liu","123");
 Statement stmt = conn.createStatement();
 rs = stmt.executeQuery(sql);
 }catch(SQLException ex) {
 System.err.println(ex.getMessage());
 }
 return rs;
}
```
- 将上述Java文件保存为dbConn.java，并编译，得到字节码文件dbConn.class，
- 存放位置为tomcat安装目录的webapps\ROOT\WEB-INF\classes\db子目录中。



## ■ 组件

- 数据库服务器 (**SQL Server**等)
- **Web**服务器 (\*.jsp、\*.java)
- 浏览器 (**IE**等)



## ■ 连接件

- 浏览器(**IE**等)向**Web**服务器发出查询学生信息的请求, **Web**服务器接受浏览器的请求后, 调用相关程序(\*.jsp、\*.java), 向数据库服务器(**SQL Server**等)发送请求。
- 数据库服务器(**SQL Server**等)根据**Web**服务器相关程序(\*.jsp、\*.java)发送的请求进行操作, 并将操作后的结果返回给**Web**服务器以及相关程序(\*.jsp、\*.java)。
- **Web**服务器相关程序(\*.jsp、\*.java)将结果发送给浏览器(**IE**等), 浏览器(**IE**等)最终将结果显示出来。



## 思考题

- **B/S体系结构：**组件、连接件、工作机制、特点。相关程序，语言不限。



# 谢谢

---

**2018年10月29日**