第一、二章

1. 什么是 C / S 模式？什么是 B / S 模式？试简述两种模式各层的作用并比较其优缺点

1. C / S 模式是客户机/服务器模式。客户机负责用户端业务逻辑的处理，且可以根据不同的用户的需求进行定制。服务器仅对重要的过程和数据库进行处理和存储。
2. B / S 模式是浏览器/服务器模式，是一种从传统的两层 C / S 模式发展起来的新的网络结构模式，其本质为三层结构的 C / S 模式。三层为表示层、业务层和数据层。表示层向客户提供数据，业务层实施业务和数据规则，数据层定义数据访问标准。B / S 系统统一了客户端，无需特殊安装，拥有 Web 浏览器即可。
3. 优缺点比较：
    1. 网络环境不同

        C / S 一般是建立在局域网的基础上的。

        B / S 一般是建立在广域网的基础上的。

    2. 硬件环境不同

        C / S 一般建立在专用的网路上，小范围的网络环境，局域网之间再通过专门服务器提供连接和数据交换服务。

        B / S 一般建立在广域网上，不必有专门的网络硬件环境，比 C / S 更强的适应范围，一般只要有操作系统和浏览器就行。

    3. 对安全要求不同

C / S 一般面向相对固定的用户群，对信息安全的控制能力很强。一般高度机密的信息系统采用 C / S 结构。

B / S 建立在广域网上，对安全的控制能力相对较弱。

4. 软件重用不同

   C / S 程序不可避免的考虑整体性，构件的重用性不如在 B / S 要求下的构件的重用性要好。

   B / S 的多重结构，要求构件相对独立的功能，能够相对较好的重用。

5. 用户接口不同

   C / S 多是建立在 Windows 平台，表现方式有限，对程序员普遍要求较高

   B / S 建立在浏览器上，通过 Web 服务或其他公共可识别描述语言可跨平台，使用更灵活。不仅可应用在 Windows 平台上，还可以应用于 Unix/Linux 平台。

2. 什么是静态网站？什么是动态网站？试比较他们之间的区别。

1. 纯粹的 HTML 格式的网页通常被称为“静态网页”。
2. 动态网页是基本的 HTML 语法规范与 PHP、Java、Python 等程序语言、数据库等多种技术的融合，以实现网站内容和风格的高效、动态、交互式的管理。
3. 区别

   动态网页在服务器端运行，客户机看到的只是它的返回结果，不可能看到它的源文件。而静态网页则只能通过服务器把网页文件原封不动地传给客户机，不进行任何的处理。

动态网页只有经过客户浏览时才会返回一个完整的网页，而其本身并不是一个独立存在于服务器的网页文件。

静态网页内容相对固定，容易被搜索引擎检索，且不需要连接数据库，因此响应速度较快。动态网页涉及到数据的连接访问和查询等一系列过程，所以响应速度相对较慢。

静态网页由于很多内容都是固定的，在功能方面有很大的限制，所以交互性较差。动态网页则可以实现更多的功能，如用户的登录、注册、查询等。

3. Web 应用服务器的用途是什么?

1. 向浏览器等 Web 客户提供文档
2. 放置网站文件让网络用户浏览
3. 提供可以下载的数据

4. 简要说明表格与框架在网页布局时的区别

1. 表格是用于划分页面区域的，而框架是用于分割浏览器窗口的，即使用框架结构的页面可以将一个浏览器窗口划分为多个相互独立而又互相联系的小窗口。
2. 表格在它的单元格内可以放置具体的内容。
3. 框架可以在同一个浏览器窗口中显示多个网页。

5. 如何在网页中设置字体？有哪些字体可以使用?

1. 设置字体：

   在具体的标签中设置其 `style` 属性。例如：

   ```
   <h1 style="font-family:verdana">
   ```

2. 可以使用的字体

SimSun：宋体

FangSong：仿宋

Microsoft YaHei：微软雅黑

KaiTi：楷体

...

6. 如何定义跨行的表格？如何将表格的字体与边框的具体加大？

1. 使用 `<td>` 元素的 `rowspan` 属性来实现单元格的跨行操作
2. `style` 属性设置为 `margin-left/right/top/bottom:数字px` ，如果要设置多个方向，那么里面的内容用封号隔开

7. 如何引入一张图片？如何给图片加上边框

1. 图片引入：

   HTML：

   `<img src="picture url">`

   JavaScript：

   ```
   <script>
       var image = document.createElement("img");
       image.src = "picture url";
       document.body.appendChild(image);
   </script>
   ```

2. 加上边框：

   1. 将图片放到 `<div>` 中

2. 在 `<style>` 中设置 `<div>` 的属性：

`border:dashed/solid 数字px 颜色;`

8. 框架有几种基本形式？如何使用？

1. 基本语法：

```
<frameset rows="行划分方式" cols="列划分方式">
    <frame src="HTML 文件1" name="框架名1">
    ...
</frameset>
```

第三章

1. Java 和 JavaScript 的区别

Java 是面向对象的程序设计语言，用于开发企业级应用程序，而 JavaScript 是在浏览器中执行，用户开发客户端浏览器的应用程序，能够实现用户与浏览器的动态交互，是一个解释性语言。

Java 是强类型变量，所有变量在编译之前都必须做声明；而 JavaScript 中变量声明采用弱类型，即变量在使用前不需做声明，而是解释器在运行时检查其数据类型。

2. JavaScript 变量命名规范

1. 严格区分大小写
2. 只能包含字母、数字和下划线，不允许包含空格和其他标点符号
3. 必须以字母或者下划线开头，不能以数字开头
4. 不能使用关键字、保留字作为变量名
5. 尽量避免使用没有意义的命名

3. 简述列举文档对象模型 DOM 里 document 的常用的查找访问节点的方法并作简单说明

1. Document.getElementById：根据元素 id 查找元素
2. Document.getElementByName：更具元素 name 查找元素
3. Document.getElementTagName：根据指定的元素名查找元素

第四章

1. 简述 JSP 的工作原理

1. 当用户访问一个 JSP 页面时，会向一个 Servlet 容器（Tomcat 等）发出请求；
2. 如果页面有所改动，则 Servlet 容器首先要把 JSP 页面（.jsp）转化为 Servlet 代码（.java），再将其转化为 class 文件（.class），然后其就会被 Servlet 容器装载并解释执行
3. 执行结束后，将结果返回到客户端（返回 html 格式的文件流）

2. 简述 page 指令、include 指令、taglib 指令的作用

1. page 指令：用于设置 JSP 页面的属性，这些属性将用于和 JSP 容器通信，控制所生成的 servlet 结构。Page 指令作用于整个 JSP 页面，可以将这个指令放在文档中任何地方。
2. include 指令：在页面转换期间将文件包含到 jsp 页面。
3. taglib 指令：声明用户使用的自定义标签

3. application 对象有什么特点？它与 session 对象有什么联系和区别

1. application 对象实现了用户间的数据共享，可存放全局变量。在其存在期间，不同用户可以对此对象的同一属性进行操作，并且在任何地方对此对象属性的操作，都会影响其他用户对此对象的访问。
2. 联系：

   数据都存储在服务端，而且都可以保留一段时间。

   区别：

   每个用户都有自己的 session，而 application 是共用的

4. JSP 常用的基本动作有哪些？简述其作用

1. `<jsp:include>`：动态包含在页面中，被请求时引入一个文件
2. `<jsp:param>`：在动作组件中引入参数信息
3. `<jsp:forward>`：把请求转到一个新的页面
4. `<jsp:useBean>`：寻找或实例化一个 JavaBean
5. `<jsp:setProperty>`：设置 JavaBean 的属性
6. `<jsp:getProperty>`：输出某个 JavaBean 的属性
7. `<jsp:plugin>`：用于指定在客户端运行的插件

5. 简述 `include` 指令（静态包含）和 `<jsp:include>` 动作（动态包含）的异同

1. 同：

   将外部文档包含到 JSP 文档中。

2. 异：

    1. 根本区别：被调用时间的不同。`include` 指令在页面转换期间被激活，而 `<jsp:include>` 动作在请求期间被激活
    2. `include` 指令先把 JSP 代码合并进来然后编译。而 `<jsp:include>` 动作是当执行到该处时先编译，把编译的结果与源文件的编译结果合并。
    3. `include` 指令包含时不能带参数，而 `<jsp:include>` 可以。

6. 有几种方式实现页面的跳转，如何实现？

1. html：

   `<a href="跳转 URL">跳转链接</a>`

2. JavaScript：

   `window.location.href='跳转 URL'`

   `window.location.reload('跳转 URL')`

3. JSP：

   `<% response.sendRedirect("xxx.jsp"); %>`

   `<jsp:forward page="xxx.jsp" />`

7. JSP 内置对象有哪些？它们的作用是什么？

1. request 对象（httpServletRequest 的对象）：

   代表客户端的请求信息，主要用于获取客户端的参数和流

2. response 对象：

   代表客户端的响应

3. session 对象：

   服务器为每个用户都生成一个 session 对象，用于保存该用户的信息并跟踪用户的操作状态。

4. application 对象：

   保存用户信息，比 session 生命周期更长且被共用。

5. out 对象：

   在 Web 浏览器内输出信息，并且管理应用服务器上的输出缓冲区。

6. pageContext 对象：

   取得任何范围的参数，可以获得前 5 个对象。

7. config 对象：

   取得服务器的配置信息。

8. page 对象：

   代表 JSP 本身，只有在 JSP 页面内才合法。本质上包含当前 Servlet 接口引用的变量，类似 this 指针。

9. exception 对象：

   显示异常信息，只有当 `isErrorPage="true"` 的页面中才能使用。

第五章

1. JDBC 连接数据库代码

```
Connection connection = null;
PreparedStatement ps = null;
ResultSet rs = null;
try{
    Class.forName(JDBC 驱动类);
```

```
    connection = DriverManager.getConnection("jdbc://localhost:3306/database", "username", "password");
    ps = connection.prepareStatement("sql 语句");
    ps.setstring(1, "xxxx");
    ...;
    rs = ps.executeQuery();
    while(rs.next()){
        ...
    }
}
catch(Exception e){
    e.printStackTrace();
}
finally{
    if(rs != null){
        rs.close();
    }
    if(ps != null){
        ps.close();
    }
    if(rs != null){
        connection.close();
    }
}
```

2. JDBC 常用的包、类和接口

1. 包：

   java.sql

   javax.sql

2. 类：

   DriverManager：装入所需要的 JDBC 驱动程序，调用它来创建连接

ResultSet：负责保存 Statement 执行过后所产生的查询结果

3. 接口：

   Connection：创建 Statement 对象

   Statement：得到 ResultSet 对象

3. JDBC 的作用

1. 与数据库建立连接
2. 向数据库发送 SQL 语句并执行这些语句
3. 处理数据返回的结果

4. JDBC 连接数据库的步骤

1. 加载 JDBC 驱动程序
2. 创建数据库的连接
3. 创建 prepareStatement
4. 执行 SQL语句
5. 遍历结果集
6. 处理异常、关闭 JDBC 对象资源

第六章

1. 什么是 JavaBean？使用 JavaBean 的优点是什么?

1. JavaBean 是一种 Java 类，是通过封装属性和方法成为具有某种功能或者处理某个具体业务的对象。
2. 优点：
    1. 可以实现代码的重复利用，因此可以缩短开发时间。

2. 易编写，易维护，易维护。
3. 可以在任何安装了 Java 运行环境的平台上使用，而不需要重新编译。这为 JSP 的应用带来了更多的可扩展性。

2. 一个标准的 JavaBean 需要具备哪些条件

1. 对象的属性：ID、name
2. 对象的 get 方法
3. 对象的 set 方法

第七章

1. 什么是 Servlet？Servlet 的技术特点是什么？Servlet 与 JSP 有什么区别？

1. Servlet 是使用应用程序设计接口（API）以及相关类和方法的 Java 程序，是服务端的一种扩展技术，运行并部署在 Servlent 容器中。
2. 技术特点：

   由 Web 服务器进行加载和运行，动态生成 Web 页面，为 Servlet 程序提供驻留环境并执行它。
3. 区别：
    1. JSP 是 Servlet 技术的扩展，本质上就是 Servlent 的简易模式，JSP 编译后是 “类 Servlet”
    2. Servlet 的应用逻辑是在 Java 文件中，并且完全从表示层中的 HTML 里分离开，通过 HttpServletResponse 动态输出 HTML 内容。而

JSP 是 Java 和 HTML 可以组合成一个扩展名为 .jsp 的文件。

2. 创建一个 Servlet 的步骤

1. 导入 Servlet 需要的基本包。
2. 继承 Servlet 接口，重写 `service()` 方法。
3. 在 web.xml 中配置 Servlet。

3. 运行 servlet 需要在 web.xml 中进行哪些配置?

1. `<servlet>` 先于 `<servlet-mapping>` 定义。
2. 在 `<servlet>` 中，`<servlet-name>` 配置该 servlet 的名字，`<servlet-class>` 定义 Servlet 程序实际的包名和类名。
3. 在 `<servlet-mapping>` 中， `<url-pattern>` 中定义 URL 的调用模式。

4. 简述 Servlet 的生命周期

1. 不存在 Servlet 实例 --> 容器装载 Servlet --> 创建 Servlet 实例 --> 初始化 Servlet，即对应的 Servlet 的 init() 方法 --> 调用对应 Servlet 的 service() 方法 --> 结束 Servlet，调用对应 Servlet 的 destory() 方法

5. 简述 HttpSession 接口的功能和使用方法

1. 这个接口被 Servlet 引擎用来实现在 HTTP 客户端和 HTTP 会话两者的关联。session 用来在无状态的 HTTP 下超过多个请求页面来维持状态和识别用户。
2. 过程：

1. 使用 HttpServletRequest 的 `getSession` 方法获得当前存在的 Session，没有就创建新的。
2. 写 Session 变量，可以使用 `HttpSession.setAttribute(name, value)` 来向 session 存储一个信息。
3. 读 Session 变量，可以使用 `HttpSession.getAttribute(name)` 来向 session 读取一个信息，读出来的类型是 Object 类
4. 关闭 Session，使用 `session.invalidate()` 来关闭（并非严格需要）。

6. 什么是过滤器？它的作用？

1. 过滤器是一个程序，它先于与之相关的 Servlet 或 JSP 页面运行在服务器上。
2. 作用：
    1. 以常规的方式调用资源（即调用 Servlet 或 JSP 页面）
    2. 利用修改过的请求信息调用资源
    3. 调用资源，但在发送响应到客户机前对其修改
    4. 阻滞该资源的调用，代之转到其他的资源，返回一个特定的状态码或生成替换输出。
3. 好处：
    1. 可以以一种模块化的或者可重用的方式封装公共的行为
    2. 可以利用过滤器将高级访问决策与表现代码相分离
    3. 通过使用过滤器能够对许多不同的资源进行批量性的更改。

第八章

1. ORM 的三个关键映射

1. 类与数据库中表的映射
2. 对象与表中的记录的映射
3. 类的属性与数据库中表的字段的映射

2. Hibernate 框架体系结构

Hibernate 是连接应用程序与数据库之间的一个中间件，在应用程序中通过创建持久化类来使用 Hibernate。这样应用程序不再关心后台所用的是什么数据库，实现了应用程序的业务逻辑与数据库之间的解耦。Hibernate 通过配置文件（hibernate.cfg.xml 或 hibernate.properties）和映射文件（*.hbm.xml）把持久化对象（Persistent Object，PO）映射到数据库中的表，程序员编程通过操作 PO 对表进行各种操作来进行。

3. Hibernate 共有 6 个核心接口，请简述每个接口的作用（网上说是 5 个，书上也是 5 个）

1. Configuration 接口

   其负责配置 Hibernate。

2. SessionFactory 接口

   一个 SessionFactory 实例对应一个数据库连接，Web 应用从 SessionFactory 中获得 Session 实例。

3. Session 接口

   持久化管理器，提供了和持久化相关的操作，例如数据库的增删改查。

4. Transaction 接口

进行事务操作，用来管理 Hibernate 事务，对底层的事务接口做了封装。

5. Query 接口

   其为 Hibernate 的查询接口，主要用于数据库对象查询，以及控制执行查询的过程。

4. HQL 查询的主要步骤

1. 获取 Session 对象
2. 以 HQL 语句作为参数，调用 Session 的 createQuery 方法创建查询对象。
3. 如果 HQL 语句本身包含参数，则调用 Query 的 setXxx() 方法为参数赋值。
4. 调用 Query 对象的 list 等方法返回查询结果列表。

5. MVC 是什么，它的优点，Model 2 架构

1. MVC 是一种程序开发设计模式，它实现了显示模块与功能模块的分离，其主要分为模型、视图、控制器三层。

   模型是应用程序的主体部分，为多个视图提供数据。

   视图是用户与之交互的界面。

   控制器是接收来自界面的请求，并将其交给模型层进行处理，起到连接作用。

2. 优点：

   1. 降低代码耦合性
   2. 有利于分工合作，各尽其职
   3. 有利于组件的重用

3. 缺点：

   1. 增加的系统接口的实现的复杂性

    2. 视图与控制器间过于紧密，妨碍两者的独立重用
    3. 视图对模型数据的低效率访问
4. Model2 架构

   其本质上就是 JSP 与 Servlet 联合使用来实现动态内容服务的方法，就是 JSP + Servlet + JavaBean 模式。使用 JSP 生成表示层的内容，使用 Servlet 完成深层次的处理内容，使用 JavaBean 进行逻辑处理。