

💎 💎 2021级web考试复习总结 💎 💎

FROM:CWidsomj
20230511 Yifu Building

📌 个人总结，仅供参考，未考勿怪，考到则灵。

😍 😍 考试题型

1. 简答 3个 概念题 18分 主流概念
2. 选择 6道 2分 很难 有代码 输出结果
3. 编程 40分 (核心是前端内容)
 1. 给定界面 写出来 html css
 2. 给定代码, 界面和描述写出来
 3. 数据库, 连接代码写出来
4. 综合编程 20分
 - 1道, 前后端, 前后端全写出来
 - 链接数据库代码, servlet和spring接口代码, 做一个简单的登录界面效果
5. 问答题 谈谈什么看法 10分

1. 简答题 - 文字类

😍 J2EE

- J2EE是什么? 它包括哪些技术

从整体上讲, J2EE是使用Java技术开发企业级应用的工业标准, 它是Java技术不断适应和促进企业级应用过程中的产物。适用于企业级应用的J2EE, 提供一个平台独立的、可移植的、多用户的、安全的和基于标准的企业级平台, 从而简化企业应用的开发、管理和部署。J2EE是一个标准, 而不是一个现成的产品。

主要包括以下这些技术:

1)Servlet

Servlet是Java平台上的CGI技术。Servlet在服务器端运行, 动态地生成Web页面。与传统的CGI和许多其它类似CGI的技术相比, Java Servlet具有更高的效率并更容易使用。对于Servlet, 重复的请求不会导致同一程序的多次转载, 它是依靠线程的方式来支持并发访问的。

2)JSP

JSP(Java Server Page)是一种实现普通静态HTML和动态页面输出混合编码的技术。从这一点来看, 非常类似Microsoft ASP、PHP等技术。借助形式上的内容和外观表现的分离, Web页面制作的任务可以比较方便地划分给页面设计人员和程序员, 并方便地通过JSP来合成。在运行时态, JSP将会被首先转换成Servlet, 并以Servlet的形态编译运行, 因此它的效率和功能与Servlet相比没有差别, 一样具有很高的效率。

3)EJB

EJB定义了一组可重用的组件：Enterprise Beans。开发人员可以利用这些组件，像搭积木一样建立分布式应用。

4) JDBC

JDBC(Java Database Connectivity, Java数据库连接)API是一个标准SQL(Structured Query Language, 结构化查询语言)数据库访问接口，它使数据库开发人员能够用标准Java API编写数据库应用程序。JDBC API主要用来连接数据库和直接调用SQL命令执行各种SQL语句。利用JDBC API可以执行一般的SQL语句、动态SQL语句及带IN和OUT参数的存储过程。Java中的JDBC相当于Microsoft平台中的ODBC(Open Database Connectivity)

HTML5新特性 (重点)

- 请简述html5的新特性，至少写出4种

语义特性：html5赋予网页更好的意义和结构，使用新的语义定义标签，可以更好地了解html文档含义，使得创建网站也更简单。

本地存储特性：能够不需要第三方插件而保存数据到用户的浏览器中。

连接特性：具有更有效的连接工作效率，使得基于网页的实时聊天、游戏和在线交流得到更优化实现。同时，html5拥有更有效的服务器推送技术。

网页多媒体特性：支持网页端的Audio、Video等多媒体功能，使访问视频、音频资源更加简单。

三维、图形及特效特性：基于SVG、Canvas、WebGL及CSS3的3D功能，呈现良好的视觉效果。

get和post (重点)

- 简述http请求get和post的区别

Form中的get和post方法，在数据传输过程中分别对应了HTTP协议中的GET和POST方法。

二者主要区别如下：

- 1) Get是用来从服务器上获得数据，而Post是用来向服务器上传数据；
- 2) Get将表单中数据按照variable=value的形式，添加到action所指向的URL后面，并且两者使用“?”连接，而各个变量之间使用“&”连接；Post是将表单中的数据放在form的数据体中，按照变量和值相对应的方式，传递到action所指向URL；
- 3) Get是不安全的，因为在传输过程，数据被放在请求的URL中；Post的所有操作对用户来说都是不可见的；
- 4) Get传输的数据量小，这主要是因为受URL长度限制；而Post可以传输大量的数据，所以在上传文件只能使用Post；
- 5) Get限制Form表单的数据集必须为ASCII字符，而Post支持整个ISO10646字符集；
- 6) Get是Form的默认方法。

IOC和AOP (重点)

- 说明反转控制 (IOC) 和面向方向编程 (AOP) 在spring中的应用

Spring 核心容器 (Core) 提供Spring框架的基本功能。核心容器的主要组件是BeanFactory，它是工厂模式的实现。BeanFactory使用控制反转 (Ioc) 模式将应用程序的配置和依赖性规范与实际的应用代码程序分开。Spring的声明式事务基于AOP实现，却并不需要程序开发者成为AOP专家，亦可轻易使用Spring的声明式事务管理。

- 什么是IOC和AOP?

IOC (Inverse of Control, 控制反转) 和AOP (Aspect-Oriented Programming, 面向切面编程) 是两个重要的编程思想。

IOC是一种设计模式，它将应用程序的控制权从应用程序代码中移出来，交给容器来管理。在IOC模式下，应用程序中的对象不再创建和管理自己所依赖的对象，而是将控制权转移给容器，由容器来负责创建和管理对象之间的依赖关系。这样，应用程序的对象之间的耦合度就降低了，代码的可维护性和可扩展性也得到了提高。Spring框架就是一个典型的IOC框架。

AOP是一种编程思想，它可以将应用程序中的关注点 (aspect) 从业务逻辑中分离出来，从而使业务逻辑更加清晰简洁。在AOP中，一个应用程序可以被看作是由多个方面 (aspect) 组成的，这些方面可以横跨整个应用程序，如日志记录、性能监控等。使用AOP可以将这些方面从应用程序代码中分离出来，从而提高了代码的可重用性和可维护性。AOP可以通过“切面” (aspect) 来实现，切面是一组跨越应用程序多个对象的行为，例如事务管理、安全检查等。Spring框架也支持AOP编程。

综上所述，IOC和AOP都是为了提高代码的可维护性和可重用性而设计的编程思想。IOC通过将对象的控制权交给容器来降低对象之间的耦合度，AOP则通过将应用程序的关注点从业务逻辑中分离出来来提高代码的可重用性和可维护性。

mvc (或许是重点)

MVC是一种设计模式，它将应用程序分为三个部分：模型 (Model)、视图 (View) 和控制器 (Controller)。MVC设计模式的目的是使应用程序的各个部分之间的耦合度降低，同时提高应用程序的可扩展性和可维护性。

- Model (模型)：负责封装应用程序的业务逻辑和数据存储。在MVC中，模型通常是一个类或一组类，它们与数据库或其他数据存储系统交互，负责数据的读取、处理和存储。
- View (视图)：负责展示应用程序的用户界面。在MVC中，视图通常是一个包含HTML、CSS和JavaScript的网页，它通过向控制器请求数据来动态地渲染页面。
- Controller (控制器)：负责处理视图和模型之间的交互，将用户的请求转发给模型，并根据模型返回的数据来选择相应的视图。在MVC中，控制器通常是一个处理HTTP请求的类或一组类，它们将请求路由到正确的模型和视图，并处理任何与请求相关的逻辑。

MVC的优点在于它能够使应用程序的各个部分彼此独立，从而提高了应用程序的可维护性、可扩展性和可重用性。此外，MVC还使开发人员能够更加专注于各个部分的开发，从而提高了开发效率。

B/S和C/S

- 什么是B/S结构,C/S结构

C/S是Client/Server的缩写。服务器通常采用高性能的PC、工作站或小型机，并采用大型数据库系统，如Oracle、Sybase、Informix或SQL Server。客户端需要安装专用的客户端软件。

B/S是Browser/Server的缩写，客户机上只要安装一个浏览器(Browser)，如Netscape Navigator或Internet Explorer，服务器安装Oracle、Sybase、Informix或SQL Server等数据库。在这种结构下，用户界面完全通过WWW浏览器实现，一部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。浏览器通过Web Server 同数据库进行数据交互。

JDBC

- Java数据库编程包含哪些类? Java数据库编程的基本过程是什么?

用到的类：Connection、ResultSet、PreparedStatement、Statement

Java中访问数据库的步骤如下：

- 1) 注册驱动；
- 2) 建立连接；
- 3) 创建Statement；
- 4) 执行sql语句；
- 5) 处理结果集（若sql语句为查询语句）；
- 6) 关闭连接。

- 简述Statement、PreparedStatement、CallableStatement的区别

Statement是PreparedStatement和CallableStatement的父类；2) Statement是直接发送Sql语句到数据库，事先没有进行预编译。PreparedStatement会将sql进行预编译，当sql语句要重复执行时，数据库会调用以前预编译好的sql语句，所以PreparedStatement在性能方面会更好；3) PreparedStatement在执行sql时，对传入的参数可以进行强制的类型转换。以保证数据格式与底层的数据库格式一致。4) CallableStatement 适用与存储过程的查询表达语句

- 什么是数据库系统

数据库系统是储存、管理、处理和维护数据的软件系统，它由数据库、数据库管理员和有关软件组成。数据库系统的结构框架由外部层（单个用户的视图）、概念层（全体用户的公共视图）和内部层（存储视图）组成

cookie和session（感觉是重点）

- 什么是cookie?

cookie是小段的文本信息，通过使用cookie可以标识用户身份、记录用户名及密码、跟踪重复用户。cookie在服务器端生成并发送给浏览器，浏览器将cookie的key/value保存到某个指定的目录中，服务器的名称与值可以由服务器端定义。

- Session的基本原理是什么？

Session对象的原理在于，服务器可以为客户端创建并维护一个所谓的Session对象，用于存放数据。在创建Session对象的同时，服务器将会为该Session对象产生一个唯一编号，这个编号称之为SessionID，服务器以Cookie的方式将SessionID存放在客户端。当浏览器再次访问该服务器时，会将SessionID作为Cookie信息带到服务器，服务器可以通过该SessionID检索到以前的Session对象，并对其进行访问。需要注意的是，此时的Cookie中仅仅保存了一个SessionID，而相对较多的会话数据保存在服务器端对应的Session对象中，由服务器来统一维护，这样一定程度保证了会话数据安全性，但增加了服务器端的内存开销。存放在客户端的用于保存SessionID的Cookie会在浏览器关闭时清除。我们把用户打开一个浏览器访问某个应用开始，到关闭浏览器为止交互过程称为一个“会话”。在一个“会话”过程中，可能会向同一个应用发出了多次请求，这些请求将共享一个Session对象，因为这些请求携带了相同的SessionID信息。Session对象的正常使用要依赖于Cookie。如果考虑到客户端浏览器可能出于安全的考虑禁用了Cookie，应该使用URL重写的方式使Session在客户端禁用Cookie的情况下继续生效。

- JavaScript能否操作cookie和session?

JavaScript可以操作cookie，但是不能操作session

Filter接口

- Filter接口有哪些方法？这些方法的用途是什么

`public void init(FilterConfig filterConfig)`过滤器的初始化方法，容器调用此方法完成过滤的初始化。对于每一个Filter实例，此方法只被调用一次。

`public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)` 此方法与Servlet的`service()`方法相类似，当请求及响应交给过滤器时，过滤器调用此方法进行过滤处理。

`public void destroy()`在过滤器生命周期结束时调用此方法，用于释放过滤器所占用的资源。



PS: 有时间的话可以了解一下:

1. 输入一个网址URL到呈现一个页面会发生的变化？

在输入一个网址 URL 到呈现一个页面的过程中，会发生以下一系列变化：

1. DNS 解析：输入 URL 后，浏览器首先会解析 URL 中的域名，将其转换为对应的 IP 地址，以便浏览器能够找到该网站的服务器。
2. 建立 TCP 连接：一旦浏览器获得了服务器的 IP 地址，它会尝试通过 Internet 连接建立一个与服务器的 TCP 连接，以便在客户端和服务器之间建立一个数据通道。
3. 发送 HTTP 请求：一旦 TCP 连接建立成功，浏览器会向服务器发送一个 HTTP 请求，该请求包含了请求页面的相关信息，比如请求方法、请求头、cookies 等。
4. 服务器处理请求：服务器接收到 HTTP 请求后，会解析请求，处理请求所对应的页面并返回结果。
5. 返回 HTTP 响应：服务器会将处理结果打包成一个 HTTP 响应，包括状态码、响应头和响应体等，并发送回浏览器。
6. 浏览器渲染页面：浏览器接收到服务器返回的 HTTP 响应后，会根据 HTML、CSS 和 JavaScript 等页面元素解析和渲染页面，最终呈现在用户的屏幕上。
7. 断开 TCP 连接：一旦页面渲染完毕并成功呈现在用户屏幕上，浏览器会关闭与服务器的 TCP 连接，页面加载过程结束。

2. MVVM（前端概念）

MVVM 是一种前端架构模式，它由 Model、View、ViewModel 三个部分组成。

Model 表示应用程序中用于处理数据的领域模型，包括数据的获取、存储和处理等。

View 表示用户界面，通常是由 HTML、CSS 和 JavaScript 组成，用于向用户展示数据和接收用户的交互操作。

ViewModel 是连接 Model 和 View 的桥梁，它主要负责将 Model 中的数据转换为 View 中的展示形式，并将用户在 View 中的交互操作转换为 Model 中的数据操作，从而实现数据的双向绑定。

在 MVVM 架构中，View 与 ViewModel 之间的通信通常采用数据绑定的方式，ViewModel 会监听 Model 中数据的变化，并将变化反映到 View 上，同时，ViewModel 也会监听 View 中用户的交互操作，并将操作转化为 Model 中的数据操作，从而保证了 Model、View 和 ViewModel 之间的解耦。

MVVM 架构的优点是可以将业务逻辑和用户界面的展示逻辑分离，使得代码更加易于维护和扩展。同时，MVVM 架构也支持数据的双向绑定，可以实现数据和界面的实时同步，提高了用户体验。

在前端开发中，MVVM 架构常常被应用于框架和库的设计中，比如 Angular、Vue.js 等。

2. 基础大题 - 代码类

HTML

1. 如何将HTML页面的标题设置为“数字天堂”？

```
<html>

<head><title>数字天堂</title></head>

<body>body部分</body>

</html>
```

2. 请描述下面代码的页面显示

```
<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

</head>

<body>

<p>输出结果如下。</p>

<button onclick="myFunction()">点我</button>

<p id="demo"></p>

<script>

function myFunction(){

var x="",i;

for (i=0;i<5;i++){

x=x + "这个数字是" + i + "<br>";

}

document.getElementById("demo").innerHTML=x;

}

</script>

</body>
```

```
</html>
```

答案解析

知识点

HTML, JavaScript

答案

[点我](#)

这个数字是0
这个数字是1
这个数字是2
这个数字是3
这个数字是4

解析

略

3. 请说明下列代码的页面显示(画图加文字说明)

```
<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

</head>

<body>

<div onmouseover="mOver(this)" onmouseout="mOut(this)" style="background-
color:#D94A38; width:120px; height:20px; padding:40px;">Mouse Over Me</div>

<script>

function mOver(obj){

obj.innerHTML="Thank You"

}

function mOut(obj){

obj.innerHTML="Mouse Over Me"

}

</script>

</body>

</html>
```

答案解析

知识点

CSS, JavaScript, HTML

答案



红色背景块，鼠标进入显示“Thank you”；离开显示“Mouse Over Me”。

解析

略

4. 请用html语言写出以下效果，其中湖南科技大学超链接到学校网站 (2020级考过)

湖南

湘潭

湖南科技大学

```
<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

</head>

<body>

<p>湖南</p>

<p>湘潭</p>

<a href="https://www.hnust.com">湖南科技大学</a>

</body>

</html>
```

5. 请用html、js实现以下效果：页面上有个按钮，点击按钮，按钮下边显示“这个月是*月”。注意，显示的月份是根据电脑时间显示 (2020级考过)

```
<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

</head>

<body>
```



```

<button onclick="myFunction()">点击这里</button>

<p id="demo"></p>

<script>

function myFunction(){

var x;

var d=new Date().getDay();

x="这个月是"+(d+1).toString()+"月";

document.getElementById("demo").innerHTML=x;

}

</script>

</body>

</html>

```

6. 将所有标签中的超链接字体改成“红色15号宋体字”

```

a{
  font-family:宋体;
  font-size:15px;
  color:red;
}

```

7. 将下文中的三个标题中的文字内容改成“红色宋体字”，字体大小分别是24、16、12。

```

<body>
  <h2 class="one"> 应用了选择器one </h2><!--定义样式后页面会自动加载样式-->
  <p> 正文内容1 </p>
  <h2 class="two">应用了选择器two</h2>
  <p>正文内容2 </p>
  <h2 class="three">应用了选择器three </h2>
  <p>正文内容3 </p>
</body>

```

```

<style>
.one{
  font-family:宋体;
  font-size:24px;
  color:red;
}
.two{
  font-family:宋体;
  font-size:16px;
  color:red;
}

```

```
.three{
    font-family:宋体;
    font-size:12px;
    color:red;
}
</style>
```

🔔 html实现一个表格、css实现一个样式效果

Servlet

1. 使用jQuery将 中的值改成abc

答: `$("#txt").attr("value", "abc");`

2. web.xml文件中配置<context-param>元素初始化参数如下:

```
<context-param>

<param-name>url</param-name>

<param-value>jdbc:mysql://localhost:3306/db_database15</param-value>

</context-param>

<context-param>

<param-name>name</param-name>

<param-value>root</param-value>

</context-param>

<context-param>

<param-name>password</param-name>

<param-value>111</param-value>

</context-param>
```

请使用application对象来获取这些参数

答:

```
String url = application.getInitParameter("url");
String name = application.getInitParameter("name");
String password = application.getInitParameter("password");
```

3. 配置User的JavaBean

```
public class User {  
    private String name;  
    private Integer age;  
    private String sex;  
    /* 省略的Setter和Getter方法 */  
}
```

答:

```
<bean name="user" class="com.mr.user.User">  
  <property name="name">  
    <value>无语</value>  
  </property>  
  <property name="age">  
    <value>30</value>  
  </property>  
  <property name="sex">  
    <value>女</value>  
  </property>  
</bean>
```

SSM

1. 创建一个名为“mrCookInfo”的cookie，在cookie中写入用户名称、用户生日和用户电子邮箱地址

```
String name = "username";  
String birthday = "19900101";  
String mail = "tom@126.com";  
Cookie myCook = new Cookie("mrCookInfo", name+"#+birthday+"#+mail);  
myCook.setMaxAge(60*60*24*365); //设置cookie有效期  
response.addCookie(myCook);
```

2. 在配置JDK的环境变量时，假如已经配置了JAVA_HOME变量，那么PATH变量的值应该是?

```
%JAVA_HOME%\bin;
```

3. 综合编程大题

要求有这样一个模块：前端进行用户登录(如图)，输入用户密码进行登录；后端对于输入的用户密码进行查询，判断是否可以登录，如果可以登录则输出“登陆成功”并跳转页面，如果不行则输出“登录失败”。请写出前端发出请求的表单代码，和后端数据库操作和servlet处理的主要代码。(2020级考过)


登录

Servlet接口: "/login"

数据库连接参数:

```
private String drive="com.mysql.jdbc.Driver";
private String url="jdbc:mysql://localhost:3306/javaweb";
private String user="root";
private String password="123456";
```

User表:

字段	索引	外键	触发器	选项	注释	SQL 预览				
名					类型	长度	小数点	不是 null	键	注
userId					int	30	0	<input checked="" type="checkbox"/>	 1	
password					varchar	255	0	<input type="checkbox"/>		
userName					varchar	255	0	<input type="checkbox"/>		

答案:

- 前端

```
<form name="login" action="login" method="post">
<input type="text" name="name" placeholder="用户名">
<input type="password" name="password" placeholder="密码">
<div class="submit">
    <input type="submit" value="登录">
</div>
</form>
```

- 数据库

```
public Connection getConnection() {
    Connection con = null;
```

```

try {

Class.forName(drv);

con=DriverManager.getConnection(url,user,password);

System.out.println("connection!");

}

catch(ClassNotFoundException e){

e.printStackTrace();

}catch(SQLException e) {e.printStackTrace();}

return con;

}

public User validate(String name,String password) { //登录验证身份

User user = new User();

Connection conn = null;

PreparedStatement stmt = null;

ResultSet res = null;

String sql ="select * from user where username='"+name+"'and
password='"+password+"'";

try{

conn = getConnection();

stmt = conn.prepareStatement(sql);

res = stmt.executeQuery();

if(res.next()) {

user.setUserid(res.getInt(1));

user.setPassword(res.getString(2));

user.setUsername(res.getString(3));

user.setState(res.getInt(4));

}

else {

```

```

user.setState(0);

}

res.close();

stmt.close();

conn.close();

}catch(SQLException e) {e.printStackTrace();}

return user;

}

```

- servlet

```

@WebServlet("/login")

public class LoginServlet extends HttpServlet {

public void doPost(HttpServletRequest request,HttpServletResponse response)
throws ServletException,IOException{

String name=request.getParameter("name");

String password=request.getParameter("password");

User user= new User();

UserDAO userdao = new UserDAOImpl();

user = userdao.validate(name, password);

state= user.getState();

if(state.equals("1")) {

System.out.println("成功登录");

}

else {

System.out.println("登录失败");

}

}

}

public void doGet(HttpServletRequest request,HttpServletResponse response)throws
ServletException,IOException{

```

```
doPost(request, response);
```

```
}
```

```
}
```