### Java Web复习指南

题目类型

单项选择（10道）、填空（5道）、简答（5道）、编程题

##### 考试范围

###### Web基础

1. HTML基础概念

超文本标记语言，是W3C组织推荐使用的一个国际标准。

1. HTTP协议

超文本传输协议，万维网的数据通信的基础。用户代理程序通过中间层访问源服务器。



HTTP请求/响应的步骤

* 客户端连接到WEB服务器
* 发送HTTP请求
* 服务器接受请求并返回HTTP响应
* 释放连接TCP连接
* 客户端浏览器解析HTML内容

HTTP请求方法

GET：请求指定的页面信息，并返回实体主体

HEAD：类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头

POST：向指定资源提交数据进行处理请求，数据被包含在请求体中，post请求可能导致新的资源的建立或已有资源的修改

PUT：从客户端向服务器传送的数据取代指定的文档的内容

另外还有DELETE\CONNECT\OPTIONS\TRACE

GET和POST请求的区别

* GET提交，请求的数据会附在URL后，以‘？’分割URL和传输数据，多个参数用‘&’连接；POST提交，把提交的数据放置在是HTTP包的包体内。
* GET提交，特定浏览器和服务器对URL长度有限制；POST提交，由于不是URL传值，理论上数据不受限。
* POST的安全性比GET高。

1. Tomcat的基础概念、配置、部署

基本概念

Tomcat是一个web应用服务器，功能是处理来自客户端浏览器发出的请求。

目录结构

Bin存放tomcat启动、关闭的脚本和启动依赖的jar包

Conf存放系统的配置文件

Lib存放jar库

Logs存放访问日志

Webapps存放web应用的文件

Work存放jsp编译后的class文件

###### Servlet

1. Servlet基本概念

是用java编写的服务器端程序。其主要功能在于交互式地浏览和修改数据，生成动态web内容。狭义的servlet是指java语言实现的一个接口，广义的servlet是指任何实现了这个servlet接口的类。

1. HttpServlet类

继承httpservlet类，重写init、doGet、doPost、destroy类

1. ServletConfig接口和ServletContext接口

Servletconfig是servlet的配置对象，作用是获取servlet的初始化参数，

Servletcontext是在servlet容器启动一个web应用时，会为该应用创建一个唯一的servletcontext对象供该应用中的所有servlet对象共享。

1. HttpServletResponse对象和 HttpServletRequest对象

Httpservletrequest对象代表客户端的请求，当客户端通过http访问服务器时，http请求头中的所有信息都封装在这个对象中，开发人员通过这个对象中的方法，可以获得客户的这些信息。

Httpservletresponse对象是指服务器的响应，这个对象中封装了客户端发送数据，发送响应头，发送响应状态码的方法。

1. doGet()和doPost()
2. RequestDispatch对象
3. Cookie对象和Session对象

Cookie通过在客户端极了信息确认用户身份，session通过在服务器端记录信息确定用户身份。

1. Servlet的生命周期

Servlet的生命周期始于将它装入web服务器的内存时，并在终止或重新装入servlet时结束。

初始化、请求处理、终止。

1. Filter过滤器

Filter是对客户端访问资源的过滤，符合条件放行，不符合条件不放行，并且可以对目标资源访问前后进行逻辑处理。

10、Listener监听器

监听器用于监听web应用中某些对象、信息的创建、销毁、增加、修改、删除等动作的

发生，然后做出响应的响应处理。

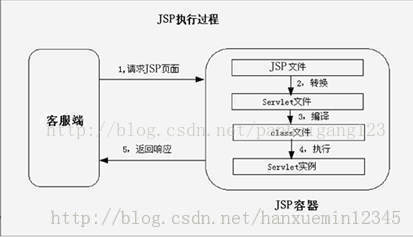
###### 三、JSP

1. JSP基本概念

动态页面，包括html,css,javascript,java,el

1. JSP的语法
2. JSP的执行过程

当第一次访问jsp页面时，会向一个servlet容器（tomcat）发出请求，servlet容器先把jsp页面转化为servlet（.java），再编译成.class文件再进行调用。当再次访问jsp页面时跳过翻译和编译的过程直接调用。



1. 与Servlet的区别

* Jsp经编译后变成了servlet。
* Jsp更擅长表现于页面显示，servlet更擅长于逻辑控制。
* Servlet没有内置对象，jsp中的内置对象都是必须通过httpservletrequest对象，httpservletresponse对象以及httpservlet对象得到。
* Jsp是servlet的一种简化，使用jsp只需要完成程序员需要输出到客户端的内容，jsp中的java脚本如何镶嵌到一个类中，由jsp容器完成。而servlet则是个完整的java类，这个类的service方法用于生成对客户端的响应。

1. JSP指令

|  |  |
| --- | --- |
| **指 令** | **说 明** |
| page | [page 指令](http://c.biancheng.net/view/1435.html)用于定义 JSP 页面的各种属性。page 是最复杂的一个指令，它的属性很多，常用的属性有 import、language、pageEncoding 等。 |
| include | 静态包含指令，用于引入其它 JSP 文件。使用 [include 指令](http://c.biancheng.net/view/1440.html)引入的其它 JSP 文件，JSP 引擎将把这两个 JSP 文件翻译成一个 servlet，所以将 include 称为静态包含。 |
| taglib | 用于在 JSP 页面中导入标签库（JSP 标准标签库、第三方标签库、自定义标签库）。 |

1. JSP的内置对象

1、request对象

request 对象是 javax.servlet.httpServletRequest类型的对象。 该对象代表了客户端的请求信息，主要用于接受通过HTTP协议传送到服务器的数据。（包括头信息、系统信息、请求方式以及请求参数等）。request对象的作用域为一次请求。

### 2、response对象

response 代表的是对客户端的响应，主要是将JSP容器处理过的对象传回到客户端。response对象也具有作用域，它只在JSP页面内有效。

### 3、session对象

session 对象是由服务器自动创建的与用户请求相关的对象。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，跟踪用户的操作状态。session对象内部使用Map类来保存数据，因此保存数据的格式为 “Key/value”。 session对象的value可以使复杂的对象类型，而不仅仅局限于字符串类型。

### 4、application对象

 application 对象可将信息保存在服务器中，直到服务器关闭，否则application对象中保存的信息会在整个应用中都有效。与session对象相比，application对象生命周期更长，类似于系统的“全局变量”。

### 5、out 对象

out 对象用于在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。在使用 out 对象输出数据时，可以对数据缓冲区进行操作，及时清除缓冲区中的残余数据，为其他的输出让出缓冲空间。待数据输出完毕后，要及时关闭输出流。

### 6、pageContext 对象

pageContext 对象的作用是取得任何范围的参数，通过它可以获取 JSP页面的out、request、reponse、session、application 等对象。pageContext对象的创建和初始化都是由容器来完成的，在JSP页面中可以直接使用 pageContext对象。

### 7、config 对象

config 对象的主要作用是取得服务器的配置信息。通过 pageContext对象的 getServletConfig() 方法可以获取一个config对象。当一个Servlet 初始化时，容器把某些信息通过 config对象传递给这个 Servlet。 开发者可以在web.xml 文件中为应用程序环境中的Servlet程序和JSP页面提供初始化参数。

### 8、page 对象

page 对象代表JSP本身，只有在JSP页面内才是合法的。 page隐含对象本质上包含当前 Servlet接口引用的变量，类似于Java编程中的 this 指针。

### 9、exception 对象

exception 对象的作用是显示异常信息，只有在包含 isErrorPage="true" 的页面中才可以被使用，在一般的JSP页面中使用该对象将无法编译JSP文件。excepation对象和Java的所有对象一样，都具有系统提供的继承结构。exception 对象几乎定义了所有异常情况。在Java程序中，可以使用try/catch关键字来处理异常情况； 如果在JSP页面中出现没有捕获到的异常，就会生成 exception 对象，并把 exception 对象传送到在page指令中设定的错误页面中，然后在错误页面中处理相应的 exception 对象。

1. JSP动作

jsp:include 动态包含；

jsp:forward 转发；

jsp:useBean 实例化bean对象；

jsp:setProperty 设置一个属性值

jsp:getProperty 获取一个属性值

jsp:param     动态传参数；

jsp:plugin      生成一个插件

1. JSP8个隐含对象(page，pageContext,request，response，session，application,config，out)
2. include与forward

###### 四、EL和JSTL

1、Javabean的基本概念

可重用的组件，在jsp程序中常用来封装业务逻辑和数据库操作的对象，实际上是一个java类，一般满足以下要求（实体类）

* 是一个公有类，并提供无参的构造方法
* 属性私有
* 具有公有的get和set方法

2、EL的基本概念，标识符、保留字、变量、常量、运算符

基本概念

Expression language表达式语言，是一种在jsp页面获取数据的简单方式（只能获取数据，不能设置数据）语法格式${expression}

3、JSTL的基本概念，标签

Jstl标签是基于jsp页面的，这些标签可以插入在jsp代码中，本质上jstl也是提前定义好的一组标签，这些标签封装了不同的功能，在也米娜上调用标签时，就等于调用了封装起来的功能。

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<c:out>](https://www.runoob.com/jsp/jstl-core-out-tag.html) | 用于在JSP中显示数据，就像<%= ... > |
| [<c:set>](https://www.runoob.com/jsp/jstl-core-set-tag.html) | 用于保存数据 |
| [<c:remove>](https://www.runoob.com/jsp/jstl-core-remove-tag.html) | 用于删除数据 |
| [<c:catch>](https://www.runoob.com/jsp/jstl-core-catch-tag.html) | 用来处理产生错误的异常状况，并且将错误信息储存起来 |
| [<c:if>](https://www.runoob.com/jsp/jstl-core-if-tag.html) | 与我们在一般程序中用的if一样 |
| [<c:choose>](https://www.runoob.com/jsp/jstl-core-choose-tag.html) | 本身只当做<c:when>和<c:otherwise>的父标签 |
| [<c:when>](https://www.runoob.com/jsp/jstl-core-choose-tag.html) | <c:choose>的子标签，用来判断条件是否成立 |
| [<c:otherwise>](https://www.runoob.com/jsp/jstl-core-choose-tag.html) | <c:choose>的子标签，接在<c:when>标签后，当<c:when>标签判断为false时被执行 |
| [<c:import>](https://www.runoob.com/jsp/jstl-core-import-tag.html) | 检索一个绝对或相对 URL，然后将其内容暴露给页面 |
| [<c:forEach>](https://www.runoob.com/jsp/jstl-core-foreach-tag.html) | 基础迭代标签，接受多种集合类型 |
| [<c:forTokens>](https://www.runoob.com/jsp/jstl-core-foreach-tag.html) | 根据指定的分隔符来分隔内容并迭代输出 |
| [<c:param>](https://www.runoob.com/jsp/jstl-core-param-tag.html) | 用来给包含或重定向的页面传递参数 |
| [<c:redirect>](https://www.runoob.com/jsp/jstl-core-redirect-tag.html) | 重定向至一个新的URL. |
| [<c:url>](https://www.runoob.com/jsp/jstl-core-url-tag.html) | 使用可选的查询参数来创造一个URL |

###### 五、JDBC

1、JDBC的基本概念

是java语言中用来规范客户端程序如何访问数据库的应用程序接口。

2、JDBC的常用API

Connection

Statement

Preparedstatement

Resultset

1. JDBC的程序实现

* **public** **class** Demo1 {
* **private** **static** **final** String URL = "jdbc:mysql://位置";
* **private** **static** **final** String USER = "用户名";
* **private** **static** **final** String PASSWORD = "密码";
* **public** **static** Connection conn = **null**;
* **public** **static** **void** main(String[] args) **throws** ClassNotFoundException, SQLException, ParseException {
* //1、加载驱动，需要提前把 jar 包添加到 classpath 中
* Class.forName("com.mysql.jdbc.Driver");
* //2、创建应用程序与数据库连接的 Connection 对象
* conn = DriverManager.getConnection(URL, USER, PASSWORD);
* //3、要执行的 sql 语句：name,password,email,status通过占位符填数,create\_date 自动为当前时间
* String sql = " INSERT INTO users"+
* "(name,password,birthday,email,create\_date,status) "+
* "VALUES("+
* "?,?,?,?,current\_date(),?)";
* //4、创建执行 SQL 语句的 PreparedStatement 对象
* PreparedStatement ptmt = conn.prepareStatement(sql);
* ptmt.setString(1, "小美");
* ptmt.setString(2, "123456");
* ptmt.setDate(3,**new** Date((**new** SimpleDateFormat("yyyy-MM-dd").parse("2011-10-1")).getTime()) );
* ptmt.setString(4, "xiaomei@126.com");
* ptmt.setInt(5, 1);
* //5、真正执行 sql 语句，并返回影响的行数
* **int** x = ptmt.executeUpdate();
* System.out.println("影响行数：" + x); //返回1
* }
* }

###### 六、MVC设计模式

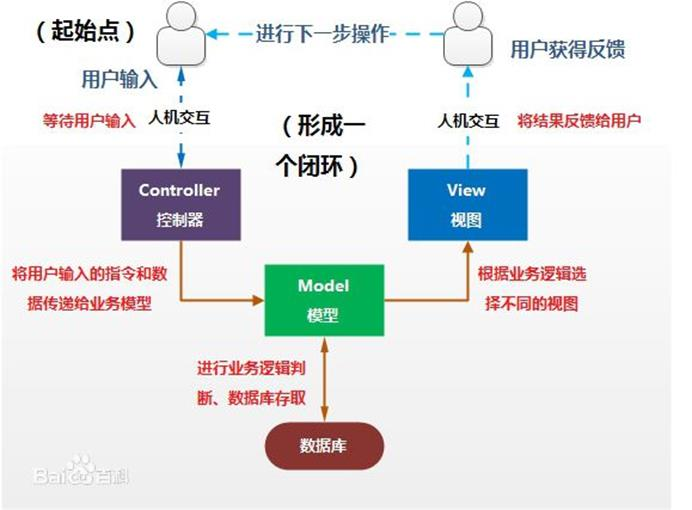
1、MVC设计模式的基本概念

模型-视图-控制器

V即View视图是指用户看到并与之交互的界面。比如由html元素组成的网页界面，或者软件的客户端界面。MVC的好处之一在于它能为应用程序处理很多不同的视图。在视图中其实没有真正的处理发生，它只是作为一种输出数据并允许用户操纵的方式。

M即model模型是指模型表示业务规则。在MVC的三个部件中，模型拥有最多的处理任务。被模型返回的数据是中立的，模型与数据格式无关，这样一个模型能为多个视图提供数据，由于应用于模型的代码只需写一次就可以被多个视图重用，所以减少了代码的重复性。

C即controller控制器是指控制器接受用户的输入并调用模型和视图去完成用户的需求，控制器本身不输出任何东西和做任何处理。它只是接收请求并决定调用哪个模型构件去处理请求，然后再确定用哪个视图来显示返回的数据。



2、两种常用的设计模式

###### 七、Spring

1、spring的基本概念

Spring为简化企业级开发而生，使用Spring开发可以将Bean对象，Dao组件对象，Service组件对象等交给Spring容器来管理，这样使得很多复杂的代码在Spring中开发却变得非常的优雅和简洁，有效的降低代码的耦合度，极大的方便项目的后期维护、升级和扩展。

2、依赖注入

是指依赖的对象不需要手动调用setXX方法去设置，而是通过配置赋值。

3、面向切面的编程

AOP 是一个概念，一个规范，本身并没有设定具体语言的实现，可以通过[预编译](https://baike.baidu.com/item/%E9%A2%84%E7%BC%96%E8%AF%91)方式和运行期动态代理实现在不修改[源代码](https://baike.baidu.com/item/%E6%BA%90%E4%BB%A3%E7%A0%81)的情况下给程序动态统一添加功能的一种技术。