【济南中心】JavaEE就业班同步笔记第一阶段：注解与Servlet3.0

小鲁哥哥 • 2017-3-18 17:04:43

【济南中心】JavaEE就业班同步笔记第一阶段：JavaWeb之基础加强--注解与Servlet3.0

1. 案例一：使用自定义注解完成@Test注解功能类似的效果:

1.1 需求

使用Junit是单元测试的工具.在一个类中使用 @Test 对程序中的方法进行测试.

自定义一个注解@MyTest 也将这个注解加在类的方法上. 使这个方法得到执行.

1.2 分析：

1.2.1　技术分析：

【注解】

程序中有 注释 和注解

* 注释：给开发人员.

* 注解：给计算机看的.

注解使用：学习框架支持注解开发.

【JDK提供的注解】

@Override　　：描述方法的重写.

@SuppressWarnings　：压制警告.

@Deprecated　　：标记过时.

自定义注解：

回复帖子...　回帖

定义一个枚举:enum

定义一个注解:@interface

【自定义注解案例】

[mw_shl_code=java,true]@interface MyAnno1{

}[/mw_shl_code]
带有属性的注解:
[mw_shl_code=java,true]@interface MyAnno2{
    int a() default 1;
    String b();
    // 注解属性的类型：基本数据类型，字符串类型String,Class,注解类型,枚举类型,以及以上类型的一维数组.
    // Date d();
    Class clazz();
    MyAnno3 m3(); // 注解
    Color c(); // 枚举
    String[] arrs();

}[/mw_shl_code]

[mw_shl_code=java,true]@MyAnno4("aaa") // 如果属性名称为value 那么使用的时候 value可以省略(只出现这一个value的属性情况下).
public class AnnotationDemo3 {

}[/mw_shl_code]

[mw_shl_code=java,true]@interface MyAnno4{
    String value();
    int a() default 1;
}[/mw_shl_code]
1.2.2 步骤分析:
定义一个测试类
[mw_shl_code=java,true]public class AnnotationDemo3 {
    @MyTest
    public void demo1(){
        System.out.println("demo1执行了...");

```
    }
    @MyTest
    public void demo2(){
        System.out.println("demo2执行了...");
    }
    public void demo3(){
        System.out.println("demo3执行了...");
    }
}[/mw_shl_code]
```

定义核心运行类：

在核心运行类中有一个主函数：

获得测试类中的所有的方法.

获得每个方法,查看方法上是否有@MyTest注解.

如果有这个注解,让这个方法执行.

1.3 代码实现：

通过元注解定义注解存在的阶段.

* 元注解也是一个注解：修饰注解的 注解.

自定义一个注解:

核心运行类：

```
[mw_shl_code=java,true]public class CoreRunner {
    public static void main(String[] args) {
        // 反射：获得类的字节码对象.Class
        Class clazz = AnnotationDemo3.class;
        // 获得测试类中的所有的方法:
        Method[] methods = clazz.getMethods();
        // 遍历数组:
        for (Method method : methods) {
            // System.out.println(method.getName());
            // 判断方法上是否有@MyTest注解：
            boolean flag = method.isAnnotationPresent(MyTest.class);
            // System.out.println(method.getName()+"   "+flag);
            if(flag){
```

```
            // 让这个方法执行:
            try {
                method.invoke(clazz.newInstance(), null);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}
```

1.4 总结：

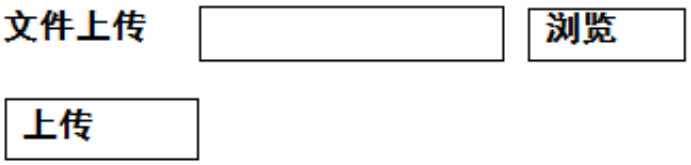
使用注解完成JDBC工具类的编写：（了解）

```
public class JDBCUtils {
    private static  String driverClass;
    private static  String url;
    private static  String username;
    private static  String password;
    @JDBCInfo
    public static Connection getConnection() throws Exception{
        // 反射：
        Class clazz = JDBCUtils.class;

        Method method = clazz.getMethod("getConnection", null);

        // 获得方法上的注解：
        JDBCInfo jdbcInfo = method.getAnnotation(JDBCInfo.class);
        driverClass = jdbcInfo.driverClass();
        url = jdbcInfo.url();
        username = jdbcInfo.username();
        password = jdbcInfo.password();

        Class.forName(driverClass);
```

```
        Connection conn = DriverManager.getConnection(url, username, password);

        return conn;
    }
}
```

2 案例二：使用Servlet3.0技术完成文件的上传:

2.1 需求:



2.2 分析:

2.2.1 技术分析：

【Servlet3.0】

Servlet3.0 与 Servlet2.5：

* Servlet3.0需要运行在tomcat7以上的服务器中.
* Servlet3.0以后web.xml就不是必须的.

1.Servlet3.0支持注解开发.

2.提供异步请求.

3.支持文件上传.

【Servlet3.0支持注解开发】

使用@WebServlet替换web.xml中配置的Servlet：

[mw_shl_code=java,true]@WebServlet(urlPatterns="/ServletDemo1",loadOnStartup=2,initParams=@WebInitParam(name="username",value="root"))[/mw_shl_code]

使用@WebListener替换web.xml中监听器的配置：

[mw_shl_code=java,true]@WebListener[/mw_shl_code]

使用@WebFilter替换web.xml中的过滤器的配置:

[mw_shl_code=java,true]@WebFilter(urlPatterns="/*")[/mw_shl_code]

【Servlet3.0支持异步请求-了解】

使用的是多线程的方式处理的,将传统的方式一次请求的非常耗时的操作,分成多个线程去处理.

需要在Servlet上添加一个属性:
[mw_shl_code=java,true]@WebServlet(urlPatterns="/ServletDemo2",asyncSupported=true)[/mw_shl_code]
[mw_shl_code=java,true]@WebServlet(urlPatterns="/ServletDemo2",asyncSupported=true)
public class ServletDemo2 extends HttpServlet{
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

        AsyncContext context = req.startAsync(req, resp);
        context.start(new MyRunner(context));

        for(int i=1;i<=20;i++){
            System.out.println(i);
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        doGet(req, resp);
    }
}[/mw_shl_code]

[mw_shl_code=java,true]class MyRunner implements Runnable{
    private AsyncContext context;
    public MyRunner(AsyncContext context) {
        this.context = context;

```
    }

    @Override
    public void run() {
        for(char i='a';i<='z';i++){
            try {
                context.getResponse().getWriter().println(i);
                Thread.sleep(100);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}
```
【Servlet3.0的文件上传】

文件上传：

文件上传：指的是将本地的文件 写到 服务器上.

文件上传的要素：

1.表单的提交的方式必须是POST.

2.表单中必须有一个文件上传项:<input type=”file”>,而且文件上传项必须有name属性和值.

```
[mw_shl_code=html,true]<input type=”file” name=”upload”/>[/mw_shl_code]
```

3.表单的enctype属性的值必须是multipart/form-data

文件上传的抓包分析：

未修改enctype属性的时候:

```
POST /WEB17_WEB/demo1/demo1.jsp HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
X-HttpWatch-RID: 22325-10011
Referer: http://localhost:8080/WEB17_WEB/demo1/demo1.jsp
Accept-Language: zh-Hans-CN,zh-Hans;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; rv:11.0) like Gecko
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
```

Host: localhost:8080
Content-Length: 47
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: JSESSIONID=99CD51DA9A47D29200168968AD983E9E
upload=C%3A%5CUsers%5Capple%5CDesktop%5Caaa.txt

已经修改了enctype属性:
POST /WEB17_WEB/demo1/demo1.jsp HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
X-HttpWatch-RID: 22325-10026
Referer: http://localhost:8080/WEB17_WEB/demo1/demo1.jsp
Accept-Language: zh-Hans-CN,zh-Hans;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; rv:11.0) like Gecko
Content-Type: multipart/form-data; boundary=---------------------------7e02e526160b66
Accept-Encoding: gzip, deflate
Host: localhost:8080
Content-Length: 224
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: JSESSIONID=99CD51DA9A47D29200168968AD983E9E

-----------------------------7e02e526160b66
Content-Disposition: form-data; name="upload"; filename="C:\Users\apple\Desktop\aaa.txt"
Content-Type: text/plain

Hello shouyi
-----------------------------7e02e526160b66—
【文件上传的原理】
根据分割线将请求体的部分分成几块:
* 判断 每块是 普通项还是文件上传项.
    * 普通项：获得名称和值.

* 文件上传项：获得文件名 和 文件内容输入流.

【文件上传的技术】

JspSmartUpload: jspSmartUpload组件是应用JSP进行B/S程序开发过程中经常使用的上传下载组件，它使用简单，方便。现在我又为其加上了下载中文名字的文件的支持，真个是如虎添翼，必将赢得更多开发者的青睐。-Model1年代的文件上传的工具.

FileUpload 是 Apache commons下面的一个子项目，用来实现Java环境下面的文件上传功能，与常见的SmartUpload齐名.应用在Model2年代了.

Servlet3.0 ：

Struts2 ：

2.2.2 步骤分析：

设计一个文件上传页面：

提交到Servlet：

接收普通项：request.getParameter();

接收上传项：Part：

通过Part对象中的方法完成文件的上传.

2.3 代码实现

[mw_shl_code=java,true]package com.itheima.servlet;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import javax.servlet.ServletException;

import javax.servlet.annotation.MultipartConfig;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.Part;

/**

* 文件上传的Servlet

*/

```
@WebServlet("/UploadServlet")
@MultipartConfig
public class UploadServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throw
s ServletException, IOException {
        // 接收普通项:
        request.setCharacterEncoding("UTF-8");
        String desc = request.getParameter("desc");
        System.out.println("文件描述:"+desc);
        Part part = request.getPart("upload");
        // 获得上传的文件的大小
        long size = part.getSize();
        System.out.println("文件大小"+size);
        String type = part.getContentType();
        System.out.println("文件类型"+type); // text/plain  image/jpeg
        String name = part.getName();
        System.out.println(name);
        // 获得文件名:
        String header = part.getHeader("Content-Disposition");
        System.out.println(header);

        int idx = header.lastIndexOf("filename=\"");
        String fileName = header.substring(idx+10, header.length()-1);
        System.out.println(fileName);

        // 获得文件内容:
        InputStream is = part.getInputStream();

        // 获得文件上传路径:
        String path = this.getServletContext().getRealPath("/upload");
```

```
        OutputStream os = new FileOutputStream(path+"/"+fileName);
        int len = 0;
        byte[] b = new byte[1024];
        while((len = is.read(b))!=-1){
            os.write(b, 0, len);
        }
        is.close();
        os.close();
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throw
s ServletException, IOException {
        doGet(request, response);
    }
}[/mw_shl_code]
```

2.4 总结:

2.4.1 文件名重名的问题:

UUID随机产生一个文件名.

2.4.2 文件上传的目录分离:

按用户分：一个用户创建一个或多个路径.

按时间分：按月，星期，天进行划分.

按个数分：一个路径中存3000个文件.

按分离算法分：按照一定的算法进行划分.

回帖

4条回帖

掬一束月光

小鲁哥哥太棒了 不过能发个汇总目录吗 不知道从哪里看起了

沙发 • 2017-3-24 18:01:32

回帖

扁舟

多线程不错

藤椅 • 2017-3-25 00:06:40

回帖

来自宇宙超级黑马专属苹果客户端

anyupeng

还是很有道理的

板凳 • 2017-3-26 14:42:12

回帖

来自宇宙超级黑马专属苹果客户端

achieve_IT

感谢分享！

报纸 • 2017-4-12 17:08:27

回帖