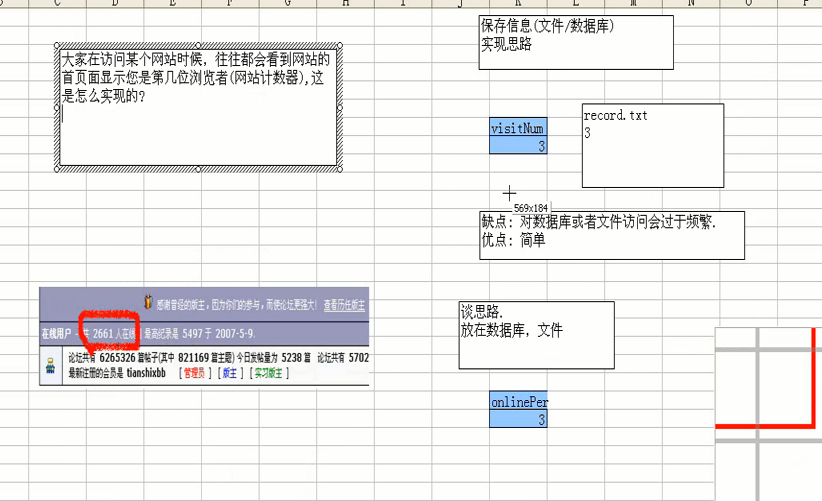
1. 为什么需要ServletContext?

一个接口

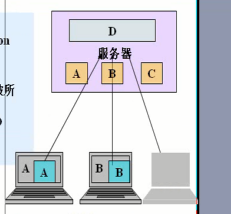
需求:访问次数/在线人数



解决技术

ServletContext

原理图:



D为ServletContext

快速入门:

1. ServletContext实在服务器端的
2. ServletContext是被所有客户端共享的
3. ServletContext是当我们的web应用启动的时候自动创建,当web应用/Tomcat/web应用reload的时候,造成ServletContext销毁

ServletContext类似于一张表

ServletContext用法小结:

获取

ServletContext servletContex=this.getServletContext();

ServletContext servletContext2=this.getServletConfig().getServletContext();

添加属性:

servletContex.setAttribute("uname", "魏少轩");

得到属性

String uname=(String) servletContext.getAttribute("uname");

删除属性

servletContext.removeAttribute("uname");

* ServletContext应用

1. 获取整个web应用的初始化参数

<!--如果希望所有的Servlet都能访问该配置 -->

<context-param>

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

<param-value>sa</param-value>

</context-param>

如何获取

String name=this.getServletContext().getInitParameter("name");(整个web应用都可以访问的)

1. 使用ServletContext实现跳转

this.getServletContext().getRequestDispatcher("/资源URL").forward(request, response);

和request.getRequestDispatcher("/资源URL").forward(request, response);

效果一样

1. 利用ServletContext读取资源文件

从webRoot下面读取文件

//首先读取到文件

InputStream inputstream=**this**.getServletContext().getResourceAsStream("dbinfo.properties");

//创建properties对象

Properties pp=**new** Properties();

pp.load(inputstream);

out.print("username="+pp.getProperty("username")+"<br/>");

如果dbinfo.properties文件在src目录下要使用类加载器

//如果文件放在src目录下,我们要使用类加载器读取

InputStream is=Servlet4.**class**.getClassLoader().getResourceAsStream("com.wsx.dbinfo.properties");

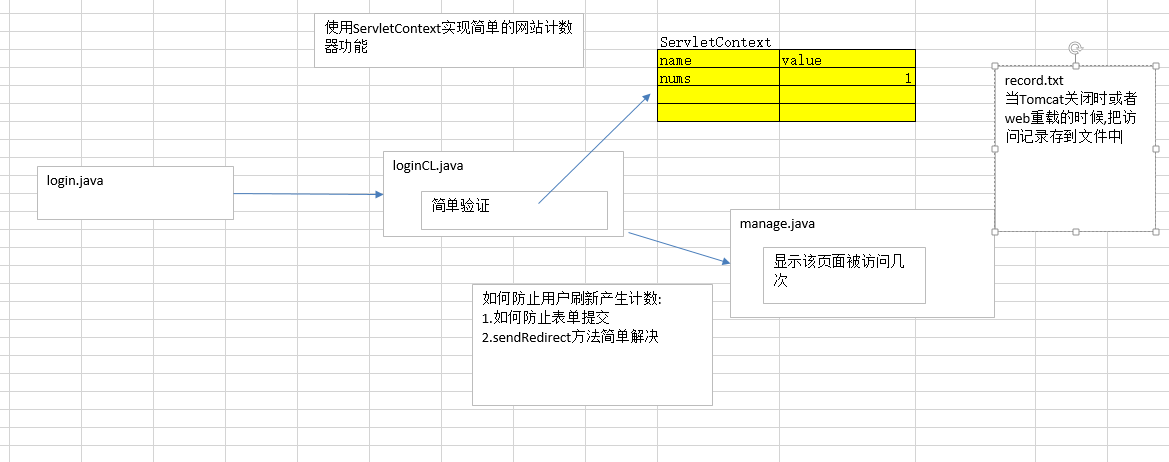
获取文件的全路径

//得到文件的全路径

String path=**this**.getServletContext().getRealPath("/images/curry1.jpg");

out.print("path="+path);

* 涉及到用户共享数据,而这些数据量又不大,不想写入数据库,我们就考虑用ServletContext来实现
* 网站计数器思考



代码:

我们建立了一个文件,record.txt用来保存访问量,这样可以保证稳定增长.

实现方法:

建立一个InitServlet,用于初始化ServletContext和关闭Tomcat的时候保存访问量

package com.wsx;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class InitServlet extends HttpServlet {

/\*\*

\* Destruction of the servlet. <br>

\*/

public void destroy() {

super.destroy(); // Just puts "destroy" string in log

// Put your code here

System.out.println("init Servlet 被调用...");

FileWriter fileWriter=null;

BufferedWriter bufferedWriter=null;

//向record.txt中写入nums

//1.得到文件路径

String filePath=this.getServletContext().getRealPath("record.txt");

//2.打开文件

try {

fileWriter=new FileWriter(filePath);

//为了写入先要得到nums的值

String nums=(String) this.getServletContext().getAttribute("nums");

//为了写入转换用BufferedWriter

bufferedWriter=new BufferedWriter(fileWriter);

//写入

bufferedWriter.write(nums);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

try {

bufferedWriter.close();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

try {

fileWriter.close();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

/\*\*

\* Initialization of the servlet. <br>

\*

\* @throws ServletException if an error occurs

\*/

public void init() throws ServletException {

// Put your code here

//从record.txt中读取浏览量

//1.得到文件的真实路径

FileReader fileReader=null;

BufferedReader bufferedReader=null;

String filePath=this.getServletContext().getRealPath("record.txt");

System.out.println(filePath);

//2.打开文件

try {

fileReader=new FileReader(filePath);

//为了读取我们转换为BufferedReader

bufferedReader=new BufferedReader(fileReader);

String nums=bufferedReader.readLine();

//把读到的值放入ServletContext

//System.out.println(nums);

this.getServletContext().setAttribute("nums", nums);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}finally{

//一定要关闭流,后打开的先关闭

try {

bufferedReader.close();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

try {

fileReader.close();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

}

当用户登陆一次,我们取出访问量,并+1.

String nums=(String) **this**.getServletContext().getAttribute("nums");

**this**.getServletContext().setAttribute("nums", (Integer.*parseInt*(nums)+1)+"");

在Manage页面显示

String nums=(String) **this**.getServletContext().getAttribute("nums");

out.print("<h2>主页</h2>");

out.print("您是第"+nums+"个访问者");

问题:如果我们的Tomcat异常退出,怎么办

使用线程,定时把ServletContext的值,刷新到record.txt中,such as 10 min

最后说:

针对工具类SQLHelper.java

1. 我们连接数据库的变量都是静态的,这样会有潜在危险,如果访问量过大,,可能造成一些用户等待超时.
   1. 解决方法:把static变量改为非static

在调用SQLHelper的时候,先创建一个SQLHelper对象,再调用相应的方法

1. 我们的SqlHelper在查询数据的时候,resultset没有在SQLHelper本类中关闭,不太好,解决方案如下:

