后台管理界面

第一个阶段  *附带 权限管理*

计划使用：DWZ前端框架实现

研究使用DWZ框架作为后台管理页面的UI实现。

*分析：虽然第一阶段的核心任务是研究dwz框架的使用。但是由于和后续任务关联度很强，所以要先花时间做好设计*

###### 包括

后台管理

微信公众号

App

###### 功能

后台管理：

新闻的发布管理，用户权限管理，供需发布管理（为了微信公众号和app和后台互动起来）

预留 文件上传，图片上传，视频，聊天、等接口

微信公众号：

自助聊天

供需查看 供需发布（需后台审核）

新闻浏览

红色字体均在web页面完成

App：

供需查看 发布

新闻浏览

图片上传，视频播放。

###### 服务器端

框架：Spring springmvc mybatis

数据库：mysql

服务器：tomcat

选择mysql数据库的原因是因为这次项目的定位是轻量级的。

###### 表结构

权限管理



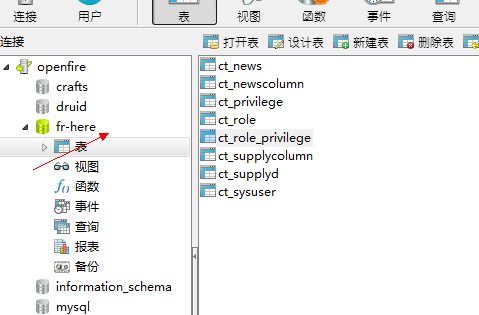
新闻发布



供需发布



建表到mysql



##### 服务器端搭建

参考上次组织部的项目，整个框架很快搭建起来了。

由于这次试用mysql，所以需要简单了解下mybatis操作oracle和mysql的不同的地方， 有点后悔使用mysql数据库了。

其实主要的原因是 cubp绑定oracle，公司大小项目只要和cubp有关全部要使用oracle数据库，而这次的开发试验使用 j-ui作为后台管理。将抛弃cubp这厮，既然抛弃了cubp，那就顺便换个小鲜肉吧----mysql。

###### Mysql mybatis 学习资料

<http://limingnihao.iteye.com/blog/781671> 基础搭建

<http://limingnihao.iteye.com/blog/781878> 关联 一对一 一对多 直观感觉会消耗效率，暂时放弃，可能需要多一点sql

<http://limingnihao.iteye.com/blog/781911> 增删改查 参数 缓存

<http://limingnihao.iteye.com/blog/782190> 动态sql语句

需要注意：

插入一体记录时，自动处理id mysql

<insert id="saveUserInfo" parameterType="UserInfo" useGeneratedKeys="true" keyProperty="id">

insert into

userinfo(userName,phone,age,birthday,remark)

values(#{userName},#{phone},#{age},#{birthday},#{remark})

</insert>

<http://chenzhou123520.iteye.com/blog/1849881>

这一系列的mysql mybatis 的博客特别值得看，把我这个半路出家做ibatis oracle的人感动的一塌糊涂。

良心博客

本来是重点研究j-ui后台框架的使用的。现在带着 服务器端 这个拖油瓶有甩不开。看来要整天不务正业了。

###### 给配置文件中数据库密码加密的方法

<!-- 引入配置文件 -->

<bean id=*"propertyConfigurer"* class=*"org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"*>

<property name=*"locations"*>

<list>

<value>/WEB-INF/config/application.properties</value>

</list>

</property>

</bean>

改为：

<bean id= "configReader" class="org.springframework.beans.factory.config.PreferencesPlaceholderConfigurer">      
        <property name = " *locations* ">  
           <list>  
               <!-- 里面保存了加密的信息 -->  
               <value>file:D:/database.properties</value>   
           </list>  
        </property>  
        <property name="propertiesPersister">  
            <bean class="com.yihaomen.propertiesextend.MyPropertiesPersist" />  
        </property>  
        <property name ="ignoreResourceNotFound" value="true"/>  
    </bean>

***MyPropertiesPersist***

import java.io.ByteArrayInputStream;  
import java.io.ByteArrayOutputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.OutputStream;  
import java.util.Properties;  
  
import org.springframework.util.DefaultPropertiesPersister;  
  
public class MyPropertiesPersist extends DefaultPropertiesPersister {  
      
    public void load(Properties props, InputStream is) throws IOException{  
      
    Properties properties = new Properties();   
    properties.load(is);   
      
    if ( (properties.get("password") != null) ){  
        /\*这里通过解密算法，得到你的真实密码，然后写入到properties中\*/  
//        String password = getRealPassword( decrypter , properties.getProperty("password") );      
//        properties.setProperty("password" , password);       
    }      
    OutputStream outputStream = null;   
    try {  
        outputStream = new ByteArrayOutputStream();   
        properties.store(outputStream, "");   
        is = outStream2InputStream(outputStream);   
        super.load(props, is);  
    }catch(IOException e) {   
        throw e;  
    }finally {  
        outputStream.close();  
    }  
    }  
      
      
    private InputStream outStream2InputStream(OutputStream out){  
        ByteArrayOutputStream bos = new ByteArrayOutputStream();   
        bos = (ByteArrayOutputStream) out ;  
        ByteArrayInputStream swapStream = new ByteArrayInputStream(bos.toByteArray());   
        return swapStream;  
    }  
}

###### SpringJUnit4ClassRunner对DAO层进行单元测试

@RunWith(value = SpringJUnit4ClassRunner.**class**)

///fr-here/WebRoot/WEB-INF/config/applicationContext.xml

@ContextConfiguration(locations = "classpath:applicationContext.xml")

**public** **class** TestSysUserDao {

@Autowired

**private** SysUserMapper sysUserMapper;

@Test

@Transactional

**public** **void** getStudentTest(){

SysUser entity = sysUserMapper.findUserByID(1);

System.*out*.println("" + entity.getAccount());

}

}

摆弄了好久

遇到问题：

1. spring-test.jar 不兼容 ，导致异常。

原因：随手找一个jar文件掉进去，导致的。 细心

解决方法：去官网下载兼容版本spring-test-4.0.6.RELEASE.jar

1. 路径问题

@ContextConfiguration(locations = "classpath:applicationContext.xml")

原因：需要加载spring的配置，因为我的spring配置文件并没有在classpath下面，然后测试各种写法都失败了。

解决方法：妥协解决，拷一份spring配置到classpath目录下面

#### 总结

这两天的工作基本上都属于铺垫型的

很多东西真正做了，用了才会出现更多问题。

引入单元测试的目的也是为了是开发更加的规范化

即使她是一个demo