用户登录功能

小程序可以通过微信官方提供的登录能力方便地获取微信提供的用户身份标识，快速建立小程序内的用户体系。

**登录流程时序图**

****

**前端微信小程序**

**Js**

**/\*\***

**\* 生命周期函数--监听页面加载**

**\*/**

**onLoad: function (options) {**

**var that = this;**

**//检查用户登陆态是否过期**

**wx.checkSession({**

**success: function () {**

**//未过期**

**wx.getSetting({**

**success: function (res) {**

**if (res.authSetting['scope.userInfo']) {**

**wx.getUserInfo({**

**success: function (res) {**

**// that.queryUsreInfo();**

**var userInfo = wx.getStorageSync("userInfo");**

**if (userInfo) {**

**that.setData({**

**userInfo: userInfo,**

**hasUserInfo: true**

**});**

**}**

**}**

**});**

**}**

**}**

**})**

**},**

**fail: function () {**

**//过期**

**//移除缓存中的信息**

**wx.removeStorageSync("userInfo");**

**wx.removeStorageSync("loginFlag");**

**//将页面的值重置**

**that.setData({**

**userInfo: {},**

**hasUserInfo: false**

**});**

**}**

**})**

**}**

**getUserInfo: function (ee) {**

**console.log(ee);**

**if (ee.detail.userInfo) {**

**var that = this;**

**wx.login({**

**success: function (res) {**

**console.log();**

**//发送请求**

**wx.request({**

**url: 'http://localhost:8081/ehome/user/userLogin',**

**data: {**

**code: res.code, //临时登录凭证**

**rawData: ee.detail.rawData, //用户非敏感信息**

**signature: ee.detail.signature, //签名**

**encrypteData: ee.detail.encryptedData, //用户敏感信息**

**iv: ee.detail.iv //解密算法的向量**

**},**

**success: function (e) {**

**var data = e.data;**

**if (data.status == "1") {**

**//数据放入缓存**

**wx.setStorageSync('userInfo', ee.detail.userInfo);**

**wx.setStorageSync('loginFlag', data.userId);**

**that.setData({**

**userInfo: ee.detail.userInfo,**

**hasUserInfo: true**

**});**

**}**

**}**

**})**

**}**

**})**

**}**

**}**

**后端java代码**

**相关配置类：**

*小程序的appid和appSecret*

AppIdAndAppSecret.class

@PropertySource("classpath:wx.properties")  
@ConfigurationProperties(prefix = "wx")  
@Configuration  
@Data  
public class AppIdAndAppSecret {  
 private String jsCode2Session ;  
 private String appId;  
 private String appSecret;  
}

OpenIdAndSessionKey.class

*微信服务器返回的openid和session\_key*

public class OpenIdAndSessionKey {  
  
 private String session\_key;  
 private String openid;  
  
 public String getSession\_key() {  
 return session\_key;  
 }  
  
 public void setSession\_key(String session\_key) {  
 this.session\_key = session\_key;  
 }  
  
 public String getOpenid() {  
 return openid;  
 }  
  
 public void setOpenid(String openid) {  
 this.openid = openid;  
 }  
  
 @Override  
 public String toString() {  
 return "OpenIdAndSessionKey{" +  
 "session\_key='" + session\_key + '\'' +  
 ", openid='" + openid + '\'' +  
 '}';  
 }  
}

*服务端微信工具*

@Component  
public class WxTools {  
  
 @Autowired  
 private AppIdAndAppSecret appIdAndAppSecret;  
  
 */\*\*  
 \* 获取openid和session\_key  
 \** ***@param*** *code 用户登陆码  
 \** ***@return*** *OpenIdAndSessionKey  
 \*/* public OpenIdAndSessionKey getOpenidAndSessionkey(String code){  
 //创建请求实体类  
 RestTemplate restTemplate = new RestTemplate();  
 //封装数据  
 Map<String,String> map = new HashMap<>();  
 String url = appIdAndAppSecret.getJsCode2Session() + "?appid=" + appIdAndAppSecret.getAppId() +  
 "&secret=" + appIdAndAppSecret.getAppSecret() +  
 "&js\_code=" + code +  
 "&grant\_type=authorization\_code";  
 String openidAndSessionKey = restTemplate.getForObject(url, String.class);  
 Gson gson = new Gson();  
 OpenIdAndSessionKey json = gson.fromJson(openidAndSessionKey, OpenIdAndSessionKey.class);  
 return json;  
 }  
  
  
}

**controller层：**

package com.zyct.ehome.controller;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.zyct.ehome.config.weixin.OpenIdAndSessionKey;  
import com.zyct.ehome.config.weixin.RawData;  
import com.zyct.ehome.config.weixin.WxTools;  
import com.zyct.ehome.entity.Owner;  
import com.zyct.ehome.service.UserService;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.data.redis.core.HashOperations;  
import org.springframework.data.redis.core.StringRedisTemplate;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.RequestMapping;  
  
import java.io.IOException;  
import java.util.HashMap;  
import java.util.Map;  
import java.util.UUID;  
  
*/\*\*  
 \** ***@author*** *litianfu  
 \** ***@version*** *1.0  
 \** ***@email*** *1035869369@qq.com  
 \** ***@date*** *2019-12-03 17:00  
 \*/*@Controller  
@RequestMapping("/user")  
public class UserController {  
 @Autowired  
 private UserService UserService;  
 @Autowired  
 private StringRedisTemplate stringRedisTemplate;  
  
  
  
 @Autowired  
 WxTools wxTools;  
  
  
 @RequestMapping(value = "/userLogin")  
 public ResponseEntity<Map<String, String>> userLogin(String code, String rawData, String signature, String encrypteData, String iv){  
 ObjectMapper mapper = new ObjectMapper();  
 RawData data = null;  
 OpenIdAndSessionKey openidAndSessionkey = null;  
 String openid = null;  
 String sessionKey = null;  
  
 try {  
 //获取数据  
 if (rawData != null && !"".equals(rawData)){  
 data = mapper.readValue(rawData,RawData.class);  
 }  
 //调用工具获取openid和sessionkey  
 openidAndSessionkey = wxTools.getOpenidAndSessionkey(code);  
 openid = openidAndSessionkey.getOpenid();  
 sessionKey = openidAndSessionkey.getSession\_key();  
 } catch (IOException e) {  
 e.printStackTrace();  
  
 }  
 if (openid != null){  
 //插入用户  
 String userId = UserService.insertUser(openid);  
  
 //缓存openid, sessionKey, userId  
 redisCache(openid,sessionKey,userId);  
 //返回数据  
 Map<String,String> map = new HashMap<String,String>();  
 map.put("status","1");  
 map.put("userId",userId);  
 return ResponseEntity.*status*(HttpStatus.*OK*).body(map);  
 }else {  
 System.*out*.println("openId为空");  
 }  
  
 return null;  
  
 }  
  
 */\*\*  
 \* 缓存openid sessionKey userId  
 \** ***@param*** *openid  
 \** ***@param*** *sessionKey  
 \** ***@param*** *userId  
 \*/* private void redisCache(String openid,String sessionKey,String userId){  
 HashOperations<String, Object, Object> ops = stringRedisTemplate.opsForHash();  
 //根据openid查询用户的userId  
 String userIdRedis = (String) ops.get("WEXIN\_USER\_OPENID\_USERID", openid);  
 if(userIdRedis != null && !"".equals(userIdRedis)){  
 //如果存在  
 ops.delete("WEIXIN\_USER\_OPENID\_USERID",openid);  
 ops.delete("WEIXIN\_USER\_USERID\_OPENID", userIdRedis);  
 ops.delete("WEIXIN\_USER\_USERID\_SESSIONKEY",userIdRedis);  
 }  
 //缓存新的  
 ops.put("WEIXIN\_USER\_OPENID\_USERID",openid,userId);  
 ops.put("WEIXIN\_USER\_USERID\_OPENID",userId,openid);  
 ops.put("WEIXIN\_USER\_USERID\_SESSIONKEY",userId,sessionKey);  
  
 }  
  
 public StringRedisTemplate getStringRedisTemplate() {  
 return stringRedisTemplate;  
 }  
  
 public void setStringRedisTemplate(StringRedisTemplate stringRedisTemplate) {  
 this.stringRedisTemplate = stringRedisTemplate;  
 }  
  
  
}

**service层：**

package com.zyct.ehome.service.impl;  
  
import com.zyct.ehome.config.weixin.RawData;  
import com.zyct.ehome.dao.UserMapper;  
import com.zyct.ehome.entity.Owner;  
import com.zyct.ehome.service.UserService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import java.util.UUID;  
  
*/\*\*  
 \** ***@author*** *litianfu  
 \** ***@version*** *1.0  
 \** ***@email*** *1035869369@qq.com  
 \** ***@date*** *2019-12-03 17:07  
 \*/*@Service  
public class UserServiceImpl implements UserService {  
  
 @Autowired  
 private UserMapper userMapper;  
  
 @Override  
 public String insertUser(String openId) {  
 //查询用户是否存在  
 Owner owner = userMapper.selectUserByWeiXinId(openId);  
 //生成唯一标识  
 String uuid = UUID.*randomUUID*().toString();  
 //如果用户不存在就插入新用户  
 if (owner == null){  
 Owner newOwner = new Owner();  
 newOwner.setOwnerId(uuid);  
 newOwner.setOwnerWeixinId(openId);  
 userMapper.insertUser(newOwner);  
 }else {  
 return owner.getOwnerId();  
 }  
 return uuid;  
 }  
}

**Mapper:**

<?xml version="1.0" encoding="UTF-8" ?>  
<!DOCTYPE mapper  
 PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  
 "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  
<mapper namespace="com.zyct.ehome.dao.UserMapper">  
 <insert id="insertUser" parameterType="Owner">  
 insert into t\_owner(owner\_id,owner\_weixin\_id) values (#{ownerId},#{ownerWeixinId})  
 </insert>  
 <select id="selectUserByWeiXinId" resultType="Owner" parameterType="String">  
 select \* from t\_owner where owner\_weixin\_id = #{v}  
 </select>  
  
</mapper>