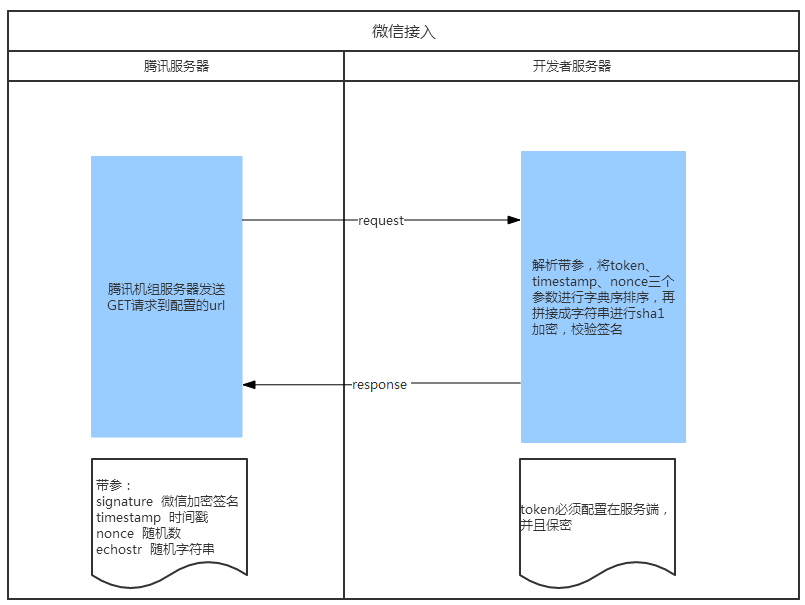
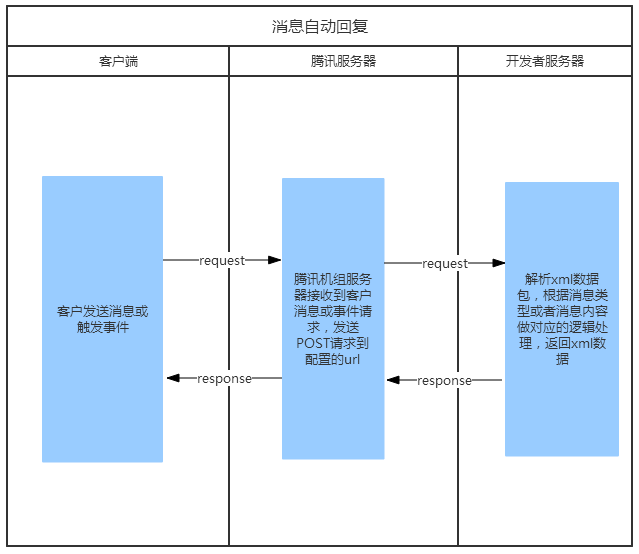
# 思路

和许多第三方服务器接入类似，微信服务器接入同样需要一个加解密的过程。当我们填好配置信息后微信服务器发起一个GET请求到我们在公众平台配置的服务器url地址，微信方将带上signature，timestamp，nonce，echostr四个参数，我们自己服务器通过拼接公众平台配置的token，以及传上来的timestamp，nonce进行SHA1加密后匹配signature，返回ture说明接入成功。



同理，消息自动回复则是用户发送消息或者触发的事件推送首先通过微信方服务器发起一个请求，通过POST请求发送到我们在公众平台配置的服务器url地址，同时带上了用户消息相关xml数据，我们通过解析xml中消息内容或者消息类型，做相关的逻辑处理，最后封装成xml的数据结构再返回给微信服务器，微信服务器再传达给用户。



# 配置

登录微信公众平台官网后，在公众平台官网的开发-基本设置页面，勾选协议成为开发者，点击“修改配置”按钮，填写服务器地址（URL）、Token和EncodingAESKey，其中URL是开发者用来接收微信消息和事件的接口URL。Token可由开发者可以任意填写，用作生成签名（该Token会和接口URL中包含的Token进行比对，从而验证安全性）。EncodingAESKey由开发者手动填写或随机生成，将用作消息体加解密密钥。



# 微信接入

## Controller

核心代码：

String signature = request.getParameter("signature");// 微信加密签名

String timestamp = request.getParameter("timestamp");// 时间戳

String nonce = request.getParameter("nonce");// 随机数

String echostr = request.getParameter("echostr");//随机字符串

// 通过检验signature对请求进行校验，若校验成功则原样返回echostr，表示接入成功，否则接入失败 if (SignUtil.checkSignature(DNBX\_TOKEN, signature, timestamp, nonce)) {

LOGGER.info("Connect the weixin server is successful.");

return echostr;

} else {

LOGGER.error("Failed to verify the signature!");

}

## 签名验证checkSignature

从上面的controller我们可以看到，我封装了一个工具类SignUtil，调用了里面的一个叫checkSignature，传入了四个值，DNBX\_TOKEN, signature, timestamp, nonce。这个过程非常重要，其实我们可以理解为将微信传过来的值进行一个加解密的过程，很多大型的项目所有的接口为保证安全性都会有这样一个验证的过程。DNBX\_TOKEN我们在微信公众平台配置的一个token字符串，注意保密哦！其他三个都是微信服务器发送get请求传过来的参数，我们进行一层sha1加密：

public class SignUtil {

/\*\*

\* 验证签名

\*

\* @param token 微信服务器token，在env.properties文件中配置的和在开发者中心配置的必须一致

\* @param signature 微信服务器传过来sha1加密的证书签名

\* @param timestamp 时间戳

\* @param nonce 随机数

\* @return

\*/

public static boolean checkSignature(String token,String signature, String timestamp, String nonce) {

String[] arr = new String[] { token, timestamp, nonce };

// 将token、timestamp、nonce三个参数进行字典序排序

Arrays.sort(arr);

// 将三个参数字符串拼接成一个字符串进行sha1加密

String tmpStr = SHA1.encode(arr[0] + arr[1] + arr[2]);

// 将sha1加密后的字符串可与signature对比，标识该请求来源于微信

return tmpStr != null ? tmpStr.equals(signature.toUpperCase()) : false;

}

}

## SHA1加密

/\*\*

\* 微信公众平台(JAVA) SDK

\*/

public final class SHA1 {

private static final char[] HEX\_DIGITS = {'0', '1', '2', '3', '4', '5',

'6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F' };

/\*\*

\* Takes the raw bytes from the digest and formats them correct.

\*

\* @param bytes the raw bytes from the digest.

\* @return the formatted bytes.

\*/

private static String getFormattedText(byte[] bytes) {

int len = bytes.length;

StringBuilder buf = new StringBuilder(len \* 2);

// 把密文转换成十六进制的字符串形式

for (int j = 0; j < len; j++) {

buf.append(HEX\_DIGITS[(bytes[j] >> 4) & 0x0f]);

buf.append(HEX\_DIGITS[bytes[j] & 0x0f]);

}

return buf.toString();

}

public static String encode(String str) {

if (str == null) {

return null;

}

try {

MessageDigest messageDigest = MessageDigest.getInstance("SHA1");

messageDigest.update(str.getBytes());

return getFormattedText(messageDigest.digest());

} catch (Exception e) {

throw new RuntimeException(e);

}

}

}

当你在公众平台提交保存，并且看到绿色的提示“接入成功”之后，恭喜你已经完成微信接入。这个过程需要细心一点，注意加密算法里的大小写，如果接入不成功，大多数情况都是加密算法的问题，多检查检查。

# 消息回复

## 引入jar包

消息自动回复的核心的xml的互相转换，为此我们需要引入两个jar

<!-- 解析xml -->

<dependency>

<groupId>dom4j</groupId>

<artifactId>dom4j</artifactId>

<version>1.6.1</version>

</dependency>

<dependency>

<groupId>com.thoughtworks.xstream</groupId>

<artifactId>xstream</artifactId>

<version>1.4.9</version>

</dependency>

## 核心service

// xml请求解析

Map<String, String> requestMap = MessageUtil.xmlToMap(request);

// 发送方帐号（open\_id）

String fromUserName = requestMap.get("FromUserName");

// 公众帐号

String toUserName = requestMap.get("ToUserName");

// 消息类型

String msgType = requestMap.get("MsgType");

// 消息内容

String content = requestMap.get("Content");

if (msgType.equals(MessageUtil.REQ\_MESSAGE\_TYPE\_EVENT)) {

String eventType = requestMap.get("Event");// 事件类型

// 订阅

if (eventType.equals(MessageUtil.EVENT\_TYPE\_SUBSCRIBE)) {

TextMessage text = new TextMessage();

text.setContent("欢迎关注，达农保险");

text.setToUserName(fromUserName);

text.setFromUserName(toUserName);

text.setCreateTime(new Date().getTime() + "");

text.setMsgType(MessageUtil.RESP\_MESSAGE\_TYPE\_TEXT);

respMessage = MessageUtil.textMessageToXml(text);

}

} else if(…){….}

有一个地方需要注意，上面标红的fromUserName和toUserName刚好相反。返回给微信服务器这时本身角色就变了，所以发送和接收方也肯定是相反的。

## xml转map集合对象

public static Map<String, String> xmlToMap(HttpServletRequest request) throws IOException{

Map<String, String> map = new HashMap<String, String>();

SAXReader reader = new SAXReader();

InputStream ins = null;

try {

ins = request.getInputStream();

} catch (IOException e1) {

e1.printStackTrace();

}

Document doc = null;

try {

doc = reader.read(ins);

Element root = doc.getRootElement();

List<Element> list = root.elements();

for (Element e : list) {

map.put(e.getName(), e.getText());

}

return map;

} catch (DocumentException e1) {

e1.printStackTrace();

}finally{

ins.close();

}

return null;

}

## 文本消息对象转换成xml

public static String textMessageToXml(TextMessage textMessage){

XStream xstream = new XStream();

xstream.alias("xml", textMessage.getClass());

return xstream.toXML(textMessage);

}

到此为止已经大功告成了，这个时候关注公众号，你会收到微信回复的“欢迎关注，达农保险”，当然你也可以发挥你的想象做所有你想做的事了，比如回复1查天气，2查违章等等....

微信公众号开发相关博文：<http://www.cnblogs.com/liliangel/category/865370.html>

欢迎指导交流