C/C++Linux服务器开发

高级架构师课程

三年课程沉淀 | 五次精益升级 | 十年行业积累 | 百个实战项目 | 十万内容受众

讲师:darren/326873713

扫一扫 升职加薪

班主任:柚子/2690491738

讲师介绍--专业来自专注和实力

King老师

系统架构师，曾供职著名创业公司系统架构师，微软亚洲研究院、创维集团全球研发中心。国内第一代商业Paas平台开发者。著有多个软件专利，参与多个开源软件维护。在全球化，高可用的物联网云平台架构与智能硬件设计方面有丰富的研发与实战经验。

Darren老师

曾供职于国内知名半导体公司（珠海扬智/深圳联发科），曾在某互联网公司担任音视频通话项目经理。主要从事音视频驱动、多媒体中间件、流媒体服务器的开发，开发过即时通讯+音视频通话的大型项目，在音视频、C/C++/GO Linux服务器领域有丰富的实战经验。

微信公众号号后台开发

1 微信公众号开发逻辑
2 HTTP服务
3 token机制
4 XML解析
5 你问我答

1 微信公众号开发逻辑

1.1 注册公众号
1.2 开发者权限
1.3 微信公众号后台接口权限
1.4 公众号消息回复
1.5 服务器配置

1.1 注册公众号

注册地址： https://mp.weixin.qq.com/cgi-bin/registermidpage?action=index&lang=zh_CN&token=

1.2 开发者权限

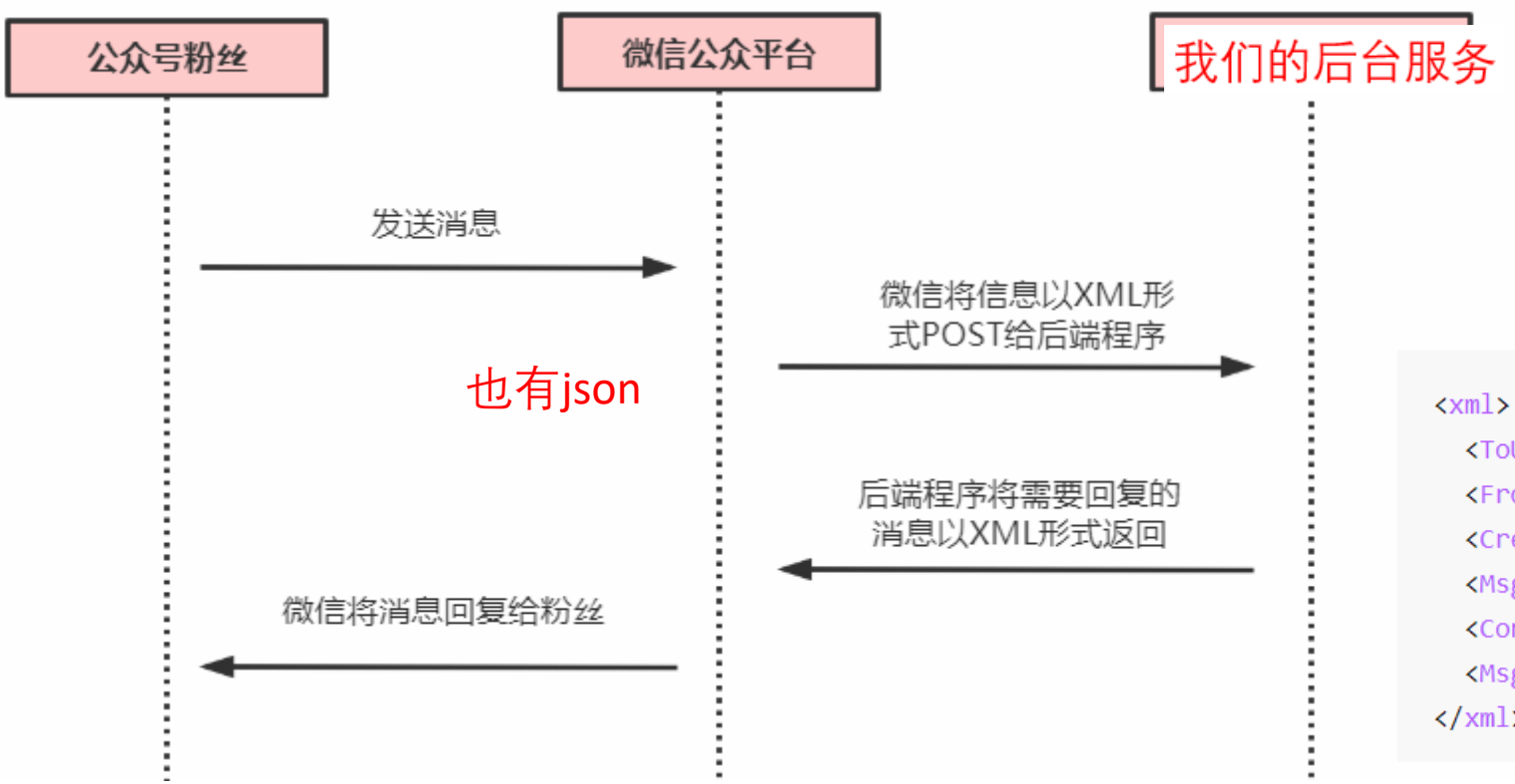
进入公众号管理页面，下拉左边侧

1.3 微信公众号后台接口权限

普通用户只要是接收消息和自动回复消息的权限

类目	功能	接口	每日实时调用量/上限(次)	接口状态
对话服务	基础支持	获取access_token	0/2000	已获得
		获取微信服务器IP地址		已获得
	接收消息	验证消息真实性	无上限	已获得
		接收普通消息	无上限	已获得
		接收事件推送	无上限	已获得
		接收语音识别结果 (已关闭)	无上限	已获得
	发送消息	自动回复	无上限	已获得
		客服接口		未获得
		群发接口		未获得
		模版消息（业务通知）		未获得
		一次性订阅消息		未获得

1.4 公众号消息回复




```
<xml>
  <ToUserName><![CDATA[toUser]]></ToUserName>
  <FromUserName><![CDATA[fromUser]]></FromUserName>
  <CreateTime>1348831860</CreateTime>
  <MsgType><![CDATA[text]]></MsgType>
  <Content><![CDATA[this is a test]]></Content>
  <MsgId>1234567890123456</MsgId>
</xml>
```

可以微信搜索：**Linux服务器开发** 查找我测试的公众号

1.5 服务器配置

服务器配置(已启用)

修改配置

服务器地址(URL)	http://111.229.231.225
令牌(Token)	liaoqingfu
消息加解密密钥(EncodingAESKey)	5JB5UsMqbFyzZNNYtbIs2RiOr1Cc6YJvhDjVo213dl3
消息加解密方式	明文模式

2 HTTP服务

我们先使用原生的http接口来处理，后续改用gin来处理

我们这里主要处理Get和Post方法，见代码

Get：处理token验证　　处理token的验证

Post：处理消息回复　　**处理消息**

工程: wechat

3 token机制

解析请求中的GET参数

微信公众号签名验证的方法

参考:

https://developers.weixin.qq.com/doc/offiaccount/Getting_Started/Getting_Started_Guide.html

源码: 3-1-token.go

3.1 token算法

按照字母排列顺序

参数	描述
signature	微信加密签名，signature结合了开发者填写的token参数和请求中的timestamp参数、nonce参数。
timestamp	时间戳
nonce	随机数
echostr	随机字符串 如果服务器校验成功，返回echostr 如果校验失败，返回""字符串

验证方法

1.服务器端获取token,nonce,timestamp组成列表
2.列表排序
3.排序后的元素进行摘要
4.摘要比对signature
5.响应echostr

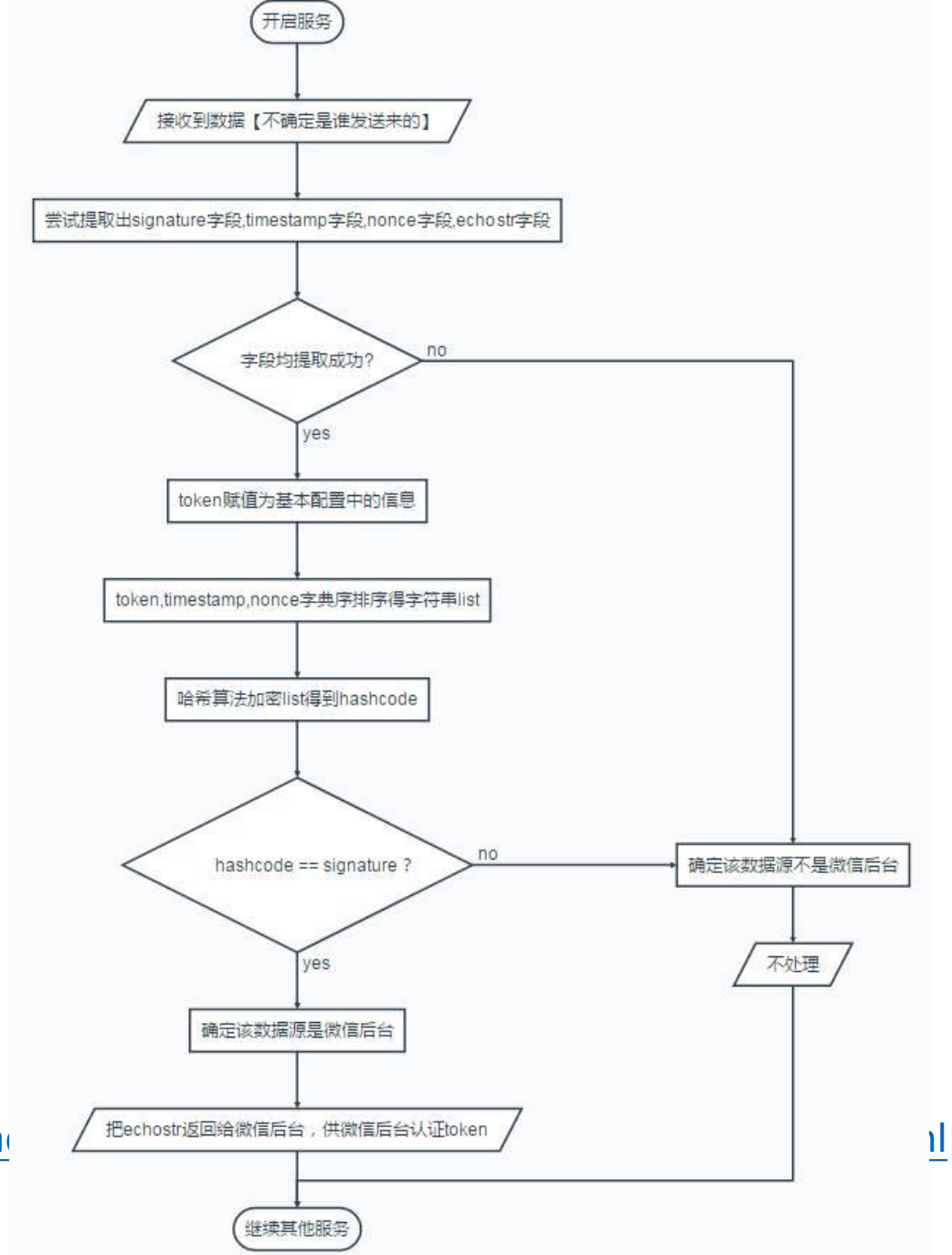
参考：

https://developers.weixin.qq.com/doc/offiaccount/Getting_Started/Getting_Started_Guide.html

3.2 token算法-流程图

验证方法

1.服务器端获取token,nonce,timestamp组成列表
2.列表排序
3.排序后的元素进行摘要
4.摘要比对signature
5.响应echostr



参考:

https://developers.weixin.qq.com/doc/offia

4 XML解析

微信消息采用XML进行封装，所以我们需要先学习XML内容解析

```
<xml>
  <ToUserName><![CDATA[toUser]]></ToUserName>
  <FromUserName><![CDATA[fromUser]]></FromUserName>
  <CreateTime>1348831860</CreateTime>
  <MsgType><![CDATA[text]]></MsgType>
  <Content><![CDATA[this is a test]]></Content>
  <MsgId>1234567890123456</MsgId>
</xml>
```

4.1 XML解析-解析XML

在代码里，先针对xml的格式，创建对应的struct结构体

```go
type SConfig struct {
    XMLName       xml.Name    `xml:"config"`      // 指定最外层的标签为config
    SmtpServer    string      `xml:"smtpServer"` // 读取smtpServer配置项，并
    SmtpPort      int         `xml:"smtpPort"`
    Sender        string      `xml:"sender"`
    SenderPasswd string      `xml:"senderPasswd"`
    Receivers     SReceivers `xml:"receivers"` // 读取receivers标签下的内容
}

type SReceivers struct {
    Age    int      `xml:"age"`
    Flag   string   `xml:"flag,attr"` // 读取flag属性
    User   []string `xml:"user"`      // 读取user数组
    Script string   `xml:"script"`    // 读取 <![CDATA[ xxx ]]> 数据
}
```

```xml
<config>
    <smtpServer>smtp.0voice.com</smtpServer>
    <smtpPort>25</smtpPort>
    <sender>youzi@0voice.com</sender>
  <senderPasswd>123456</senderPasswd>
    <receivers flag="true">
      <user>king@0voice.gom</user>
      <user>darren@0voice.gom</user>
      <script>
      <![CDATA[
          function matchwo(a,b) {
              if (a < b && a < 0) then {
                  return 1;
              } else {
                  return 0;
              }
          }
          ]]>
      </script>
  </receivers>
</config>
```

4-1-xml.go

4.2 XML解析-解析CDATA

```
<xml> <ToUserName><![CDATA[toUser]]></ToUserName>
<FromUserName><![CDATA[fromUser]]></FromUserName>
<CreateTime>1348831860</CreateTime>
<MsgType><![CDATA[text]]></MsgType>
<Content><![CDATA[this is a test]]></Content>
<MsgId>1234567890123456</MsgId> </xml>
```

XML 文档中的所有文本均会被解析器解析。
只有 CDATA 区段中的文本会被解析器忽略。

术语 CDATA 是不应该由 XML 解析器解析
像 "<" 和 "&" 字符在 XML 元素中都是非法的。
"<" 会产生错误，因为解析器会把该字符解释为新元素的开始。
"&" 会产生错误，因为解析器会把该字符解释为字符实体的开始。
某些文本，比如 JavaScript 代码，包含大量 "<" 或 "&" 字符。为了避免错误，可以将脚本代码定义为 CDATA。
CDATA 部分中的所有内容都会被解析器忽略。
CDATA 部分由 " **<![CDATA[** " 开始，由 "**]]>**" 结束：

```
<script>
<![CDATA[
function matchwo(a,b)
{
if (a < b && a < 0) then
{
return 1;
}
else
{
return 0;
}
}
]]>
</script>
```

4-2-CDATA.go

5 你问我答

1）理解被动消息的含义 2）理解收\发消息机制 预实现功能： 粉丝给公众号一条文本消息，公众号立马回复一条文本消息给粉丝，不需要通过公众平台网页操作。

5.1 你问我答–接收消息协议

```
<xml> <ToUserName><![CDATA[toUser]]></ToUserName>
<FromUserName><![CDATA[fromUser]]></FromUserName>
<CreateTime>1348831860</CreateTime>
<MsgType><![CDATA[text]]></MsgType>
<Content><![CDATA[this is a test]]></Content>
<MsgId>1234567890123456</MsgId> </xml>
```

参数	描述
ToUserName	开发者微信号
FromUserName	发送方帐号（一个OpenID）
CreateTime	消息创建时间 （整型）
MsgType	消息类型，文本为text
Content	文本消息内容
MsgId	消息id，64位整型

参考:
https://developers.weixin.qq.com/doc/offiaccount/Message_Management/Receiving_standard_messages.html

5.2 你问我答–被动回复消息协议

```xml
<xml>
  <ToUserName><![CDATA[toUser]]></ToUserName>
  <FromUserName><![CDATA[fromUser]]></FromUserName>
  <CreateTime>12345678</CreateTime>
  <MsgType><![CDATA[text]]></MsgType>
  <Content><![CDATA[你好]]></Content>
</xml>
```

参数	是否必须	描述
ToUserName	是	接收方帐号（收到的OpenID）
FromUserName	是	开发者微信号
CreateTime	是	消息创建时间 （整型）
MsgType	是	消息类型，文本为text
Content	是	回复的消息内容（换行：在content中能够换行，微信客户端就支持换行显示）

参考：
https://developers.weixin.qq.com/doc/offiaccount/Message_Management/Passive_user_reply_message.html#0

6 go语言之进阶篇正则表达式

参考官网： https://studygolang.com/pkgdoc

范例： https://www.cnblogs.com/nulige/p/10260149.html

[...]	字符集中的字符可以逐个列出，也可以给出范围，如[abc]或[a-c]。第一个字符如果是^则表示取反，如[^abc]表示不是abc的其他字符。 所有的特殊字符在字符集中都失去其原有的特殊含义。在字符集中如果要使用]、-或^，可以在前面加上反斜杠，或把]、-放在第一个字符，把^放在非第一个字符。	a[bcd]e	abe ace ade
预定义字符集（可以写在字符集[...]中）			
\d	数字：[0-9]	a\dc	a1c
\D	非数字：[^\d]	a\Dc	abc
\s	空白字符：[<空格>\t\r\n\f\v]	a\sc	a c
\S	非空白字符：[^\s]	a\Sc	abc
\w	单词字符：[A-Za-z0-9_]	a\wc	abc
\W	非单词字符：[^\w]	a\Wc	a c
数量词（用在字符或(...)之后）			
*	匹配前一个字符0或无限次。	abc*	ab abccc
+	匹配前一个字符1次或无限次。	abc+	abc abccc
?	匹配前一个字符0次或1次。	abc?	ab abc
{m}	匹配前一个字符m次。	ab{2}c	abbc
{m,n}	匹配前一个字符m至n次。 m和n可以省略：若省略m，则匹配0至n次；若省略n，则匹配m至无限次。	ab{1,2}c	abc abbc
*? +? ?? {m,n}?	使 * + ? {m,n}变成非贪婪模式。	示例将在下文中介绍。	
边界匹配（不消耗待匹配字符串中的字符）			
^	匹配字符串开头。 在多行模式中匹配每一行的开头。	^abc	abc
$	匹配字符串末尾。 在多行模式中匹配每一行的末尾。	abc$	abc
\A	仅匹配字符串开头。	\Aabc	abc
\Z	仅匹配字符串末尾。	abc\Z	abc
\b	匹配\w和\W之间。	a\b!bc	a!bc
\B	[^\b]	a\Bbc	abc
逻辑、分组			