关注并使用过微信“飞常准”公众号的朋友们都有过如下体验：查询一个航班情况后，这个航班的checkin、登机、起降等信息都会在后续陆续异步发给你，这个服务就是通过微信公众平台的客服消息实现的。

[微信公众平台开发文档](http://mp.weixin.qq.com/wiki)中关于客服消息的解释如下：“当用户主动发消息给公众号的时候（包括发送信息、点击自定义菜单、订阅事件、扫描二维码事件、支付成功 事件、用户维权），微信将会把消息数据推送给开发者，开发者在一段时间内（目前修改为48小时）可以调用客服消息接口，通过POST一个JSON数据包来 发送消息给普通用户，在48小时内不限制发送次数。此接口主要用于客服等有人工消息处理环节的功能，方便开发者为用户提供更加优质的服务”。

这篇文章我们就来说说如何用[golang](http://tonybai.com/tag/golang)实现发送文本客服消息。

**一、获取access\_token**

access\_token是公众号的全局唯一票据，公众号调用微信平台各接口时都需使用access\_token。我们要主动给微信平台发送客服消息，该access\_token就是我们的凭证。在构造和下发客服消息前，我们需要获取这个access\_token。

access\_token的有效期为2小时（7200s），我们获取一次，两小时内均可使用。微信公众平台开发文档也给出了access\_token获取、保存以及刷新的技术建议。但我们这里仅是Demo，无需考虑这么多。

通过https GET请求，我们可以得到属于我们的access\_token，请求line为：

https://api.weixin.qq.com/cgi-bin/token?grant\_type=client\_credential&appid=APPID&secret=APPSECRET

golang提供了默认的http client实现，通过默认的client实现我们可以很容器的获取access\_token。

const (  
        token               = "wechat4go"  
        appID               = "wx8e0fb2659c2eexxx"  
        appSecret           = "22746009b0162fe50cb915851c53fyyy"  
        accessTokenFetchUrl = "https://api.weixin.qq.com/cgi-bin/token"  
)

func fetchAccessToken() (string, float64, error) {  
        requestLine := strings.Join([]string{accessTokenFetchUrl,  
                "?grant\_type=client\_credential&appid=",  
                appID,  
                "&secret=",  
                appSecret}, "")

        resp, err := http.Get(requestLine)  
        if err != nil || resp.StatusCode != http.StatusOK {  
                return "", 0.0, err  
        }

        defer resp.Body.Close()  
        body, err := ioutil.ReadAll(resp.Body)  
        if err != nil {  
                return "", 0.0, err  
        }

        fmt.Println(string(body))  
        … …  
}

无论成功与否，微信平台都会返回一个包含json数据的应答：

如果获取正确，那么应答里的Json数据为：

{"access\_token":"0QCeHwiRtPRUCiM5MM0cSPYIP5QOUNYdb8usRSgVZcsFuVF6mu3vQq41OIifJdrtJPGn7b1x90HdvUanpb7eZHxg40B6bU\_Sgszh2byyF40","expires\_in":7200}

如果获取错误，那么应答里的Json数据为：

{"errcode":40001,"errmsg":"invalid credential"}

和xml数据包一样，golang也提供了json格式数据包的Marshal和Unmarshal方法，且使用方式相同，也是将一个json数据包与一 个struct对应起来。从上面来看，通过http response，我们无法区分出是否成功获取了token，因此我们需要首先判断试下body中是否包含某些特征字符串，比 如"access\_token"：

if bytes.Contains(body, []byte("access\_token")) {  
    //unmarshal to AccessTokenResponse struct  
} else {  
    //unmarshal to AccessTokenErrorResponse struct  
}

针对获取成功以及失败的两种Json数据，我们定义了两个结构体：

type AccessTokenResponse struct {  
        AccessToken string  `json:"access\_token"`  
        ExpiresIn   float64 `json:"expires\_in"`  
}

type AccessTokenErrorResponse struct {  
        Errcode float64  
        Errmsg  string  
}

Json unmarshal的代码片段如下：

//Json Decoding  
if bytes.Contains(body, []byte("access\_token")) {  
        atr := AccessTokenResponse{}  
        err = json.Unmarshal(body, &atr)  
        if err != nil {  
            return "", 0.0, err  
        }  
        return atr.AccessToken, atr.ExpiresIn, nil  
} else {  
        fmt.Println("return err")  
        ater := AccessTokenErrorResponse{}  
        err = json.Unmarshal(body, &ater)  
        if err != nil {  
            return "", 0.0, err  
        }  
        return "", 0.0, fmt.Errorf("%s", ater.Errmsg)  
}

我们的main函数如下：  
func main() {  
        accessToken, expiresIn, err := fetchAccessToken()  
        if err != nil {  
                log.Println("Get access\_token error:", err)  
                return  
        }  
        fmt.Println(accessToken, expiresIn)  
}

编译执行，成功获取access\_token的输出如下：

0QCeHwiRtPRUCiM5MM0cSPYIP5QOUNYdb8usRSgVZcsFuVF6mu3vQq41OIifJdrtJPGn7b1x90HdvUanpb7eZHxg40B6bU\_Sgszh2byyF40 7200

失败时，输出如下：

2014/12/30 12:39:56 Get access\_token error: invalid credential

**二、发送客服消息**

平台开发文档中定义了文本客服消息的body格式，一个json数据：

{  
    "touser":"OPENID",  
    "msgtype":"text",  
    "text":  
    {  
         "content":"Hello World"  
    }  
}

其中的touser填写的是openid。之前的文章中提到过，每个微信用户针对某一个订阅号/服务号都有唯一的OpenID，这个ID可以在微信订阅号 /服务号管理页面中看到，也可以在收到的微信平台转发的消息中看到(FromUserName)。比如我个人订阅的我的测试体验号后得到的OpenID 为：

BQcwuAbKpiSAbbvd\_DEZg7q27QI

我们要做的就是构造这样一个json数据，并放入HTTP Post包中，发到：

https://api.weixin.qq.com/cgi-bin/message/custom/send?access\_token=ACCESS\_TOKEN

从平台开发文档给出的json数据包样例来看，这是个嵌套json数据包，我们通过下面方法marshall：

type CustomServiceMsg struct {  
        ToUser  string         `json:"touser"`  
        MsgType string         `json:"msgtype"`  
        Text    TextMsgContent `json:"text"`  
}

type TextMsgContent struct {  
        Content string `json:"content"`  
}

func pushCustomMsg(accessToken, toUser, msg string) error {  
        csMsg := &CustomServiceMsg{  
                ToUser:  toUser,  
                MsgType: "text",  
                Text:    TextMsgContent{Content: msg},  
        }

        body, err := json.MarshalIndent(csMsg, " ", "  ")  
        if err != nil {  
                return err  
        }  
        fmt.Println(string(body))  
        … …  
}

如果单纯输出上面marshal的结果，可以看到：

{  
   "touser": "oBQcwuAbKpiSAbbvd\_DEZg7q27QI",  
   "msgtype": "text",  
   "text": {  
     "content": "你好"  
   }  
 }

接下来将marshal后的[]byte放入一个http post的body中，发送到指定url中：

var openID = "oBQcwuAbKpiSAbbvd\_DEZg7q27QI"

func pushCustomMsg(accessToken, toUser, msg string) error {  
        … …

        postReq, err := http.NewRequest("POST",  
                strings.Join([]string{customServicePostUrl, "?access\_token=", accessToken}, ""),  
                bytes.NewReader(body))  
        if err != nil {  
                return err  
        }

        postReq.Header.Set("Content-Type", "application/json; encoding=utf-8")

        client := &http.Client{}  
        resp, err := client.Do(postReq)  
        if err != nil {  
                return err  
        }  
        resp.Body.Close()

        return nil  
}

我们在main函数中加上客服消息的发送环节：

func main() {  
        // Fetch access\_token  
        accessToken, expiresIn, err := fetchAccessToken()  
        if err != nil {  
                log.Println("Get access\_token error:", err)  
                return  
        }  
        fmt.Println(accessToken, expiresIn)

        // Post custom service message  
        msg := "你好"  
        err = pushCustomMsg(accessToken, openID, msg)  
        if err != nil {  
                log.Println("Push custom service message err:", err)  
                return  
        }  
}

编译执行，手机响起提示音，打开观看，微信公众平台测试号发来消息：“你好”。

上述Demo完整代码在[这里](https://github.com/bigwhite/experiments/tree/master/wechat_examples/public/4-customservicetextmsg)可以看到，别忘了appID，appSecret改成你自己的值。

**目前客服接口仅提供给认证后的订阅号以及服务号，对于未认证的订阅号，无法发送客服消息。**

文章地址：

https://tonybai.com/2014/12/30/send-custom-service-text-msg-for-wechat-public-platform-dev-in-golang/