GoEasy 使用说明 v0.1.18

Copyright © 2015-2018 GoEasy 版权所有

1. GoEasy介绍 **3**

2. 获得开发者账号 **5**

3. 获取SDK **5**

3.1. JavaScript 5

3.2. Java 6

4. 服务器端发送（发布）消息 **7**

4.1. RESTful API 7

4.2. SDK 8

5. 浏览器端发送（发布）和接收消息 **8**

5.1. 发送（发布）消息 8

5.2. 接收（订阅）消息 8

5.3. 取消订阅 9

6.微信小程序 **9**

7.客户端在线状态 **9**

8. 事件和监听器 **9**

8.1. 网络状态 9

8.2. 消息发送（发布） 10

8.3. 接收消息（订阅Channel） 11

8.4. 取消订阅Channel 11

9. GoEasy-OTP **12**

9.1. 什么是GoEasy-OTP？有什么用途呢? 12

9.2. GoEasy-OTP是如何工作的呢？ 12

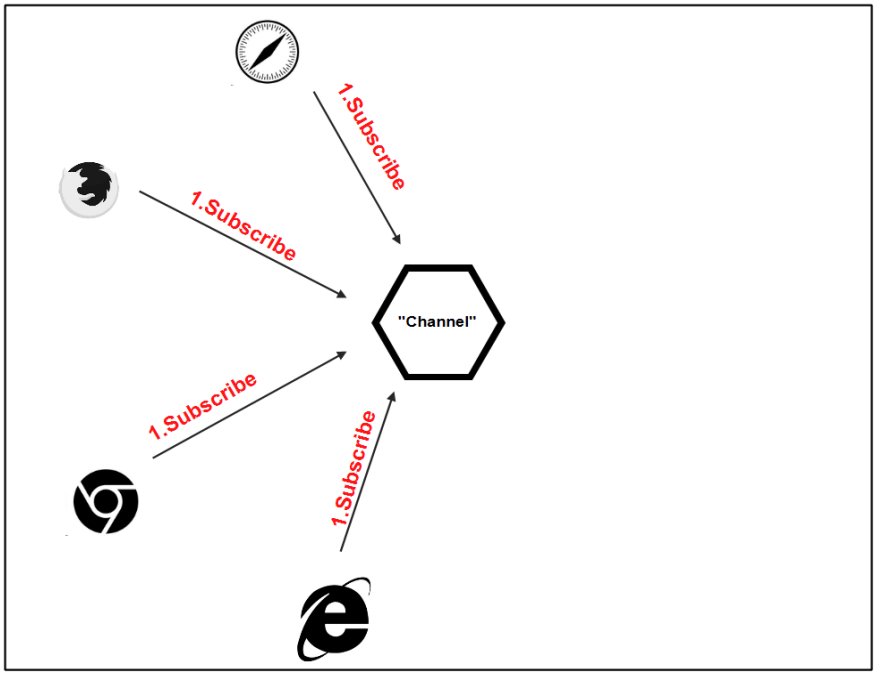
9.3. 我想使用GoEasy-OTP来保护我的消息，要如何开始呢？ 12

9.4. 生成GoEasy-OTP参考代码 13

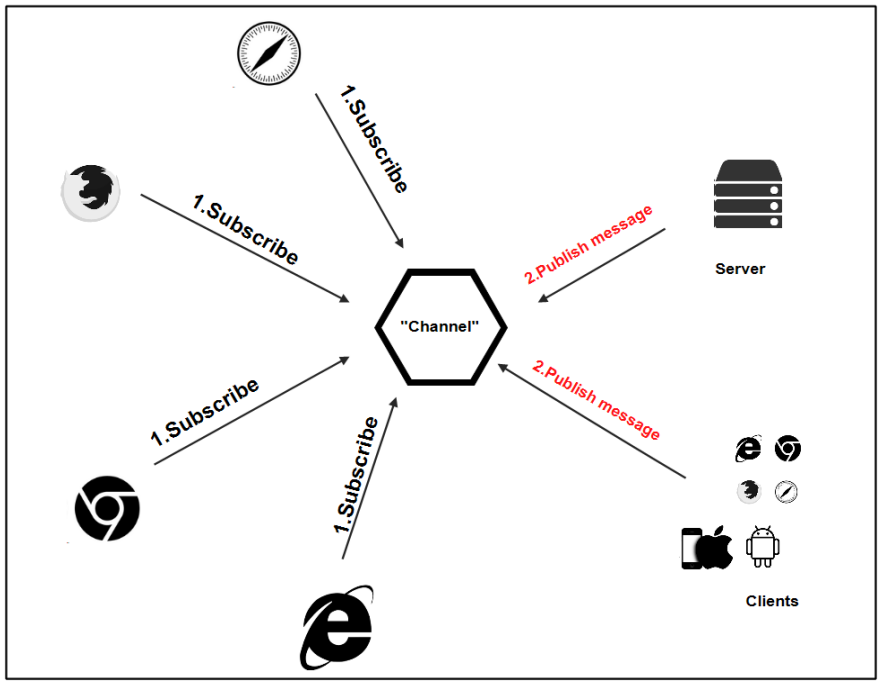
附录： **15**
A.返回值说明 15

1. GoEasy介绍

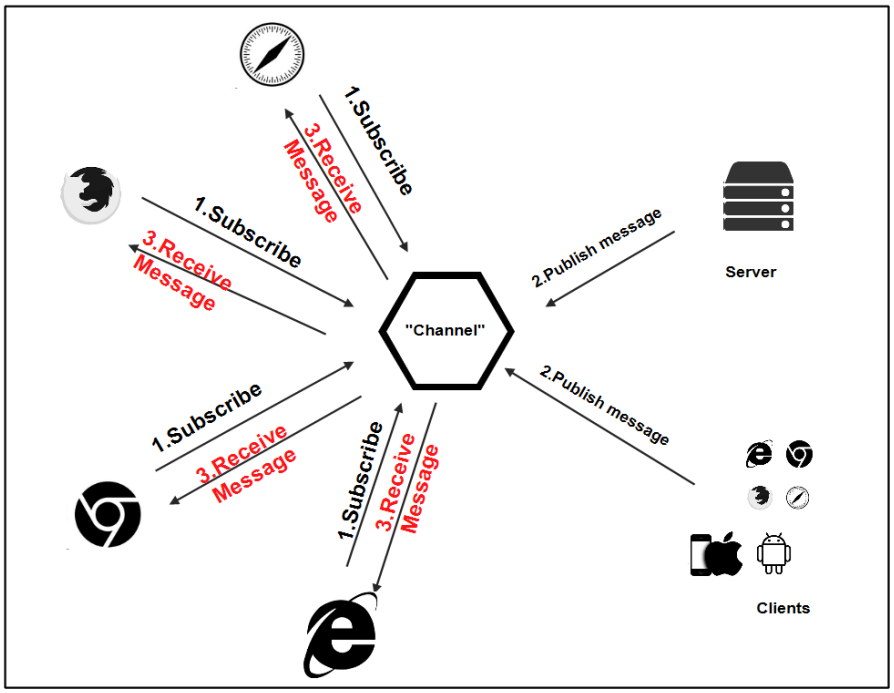
GoEasy专注于服务器与浏览器，浏览器与浏览器之间消息推送，完美兼容世界上的绝大多数浏览器，包括IE6, IE7之类的非常古老的浏览器。
GoEasy采用 发布/订阅 的消息模式，帮助您非常轻松的实现一对一，一对多的通信。
工作流程：
1. 浏览器订阅channel。



2. 向channel发布消息，不论是服务器还是客户端均可以向channel发布消息。



3. 当客户端或者服务器向channel上发布一条消息时，所有订阅该channel的浏览器均可以收到该消息。



2. 获得开发者账号

1. 在GoEasy官网（http://goeasy.io）注册账号
2. 验证手机号码
3. 登录，添加application，获得appkey

3. 获取SDK

3.1. JavaScript

<script src="http(s)://<CDN Host>/goeasy.js"></script>

注意：

1. 请不要将goeasy.js下载到本地，GoEasy动态为不同浏览器提供不同内容的goeasy.js，使用下载到本地的goeasy.js，将会导致某些浏览器不能发送和接收消息。
2. 如果您的应用程序需要支持**Windows XP用户**，**请务必使用HTTP**来获取js文件，GoEasy将于**2016年9月22日**升级所有SSL证书至SHA-2，届时起**Windows XP用户**将无法通过**HTTPS**连接GoEasy。

3.2. Java

方式一：手动下载sdk

goeasy-sdk.jar

下载依赖包：gson-2.3.1.jar slf4j-api-1.7.2.jar

方式二：Maven

添加goeasy仓库到您的pom.xml文件：

```
<repositories>
  …
  <repository>
      <id>goeasy</id>
      <name>goeasy</name>
      <url>
          http://maven.goeasy.io/content/repositories/releases/
      </url>
  </repository>
</repositories>
```

然后添加依赖：

```
<dependencies>
  …
  <dependency>
```

```
        <groupId>io.goeasy</groupId>
        <artifactId>goeasy-sdk</artifactId>
        <version>0.3.8</version>
    </dependency>
</dependencies>
```

4. 服务器端发送（发布）消息

4.1. RESTful API

URL:

http(s)://<REST Host>/publish

Method:

Post

参数：

名称	必要	描述
appkey	是	您的appkey
channel	是	channel
content	是	您要发送的消息内容

返回值：

```
{
        "code" : 200,
        "content" : "OK"
}
```

返回值的详细说明，请参考附录A。

4.2. SDK

对于服务器端，目前GoEasy仅提供Java 的SDK, 未来会提供更多其他语言的SDK，敬请期待。

4.2.1. 用Java SDK在服务器端发送消息

```
GoEasy goEasy = new GoEasy( "http(s)://<REST Host>”, "my_appkey");
goEasy.publish("my_channel", "Hello, GoEasy!");
```

5. 浏览器端发送（发布）和接收消息

5.1. 发送（发布）消息

5.1.1. Java

```
GoEasy goEasy = new GoEasy( "http(s)://<REST Host>”, "my_appkey");
goEasy.publish("my_channel","Hello, GoEasy!");
```

5.1.2. JavaScript

```
var goEasy = new GoEasy({
    appkey: "my_appkey"
});
//GoEasy-OTP可以对appkey进行有效保护，详情请参考 7.GoEasy-OTP

goEasy.publish({
   channel: "my_channel",
   message: "Hello, GoEasy!"
});
```

5.2. 接收（订阅）消息

5.2.1. JavaScript

```
var goEasy = new GoEasy({
    appkey: "my_appkey"
});
goEasy.subscribe({
    channel: "my_channel",
    onMessage: function (message) {
      alert("Channel:" + message.channel + " content:" + message.content);
    }
});
```

5.3. 取消订阅

5.3.1. JavaScript

```
goEasy. unsubscribe ({
   channel: "my_channel"
});
```

6.微信小程序

网页版goeasy.js不能应用于微信小程序的开发，该特性目前尚处于试用阶段，详情请联系GoEasy。

7.客户端在线状态

该特性目前尚处于试用阶段，详情请联系GoEasy。

8. 事件和监听器

8.1. 网络状态

在初始化GoEasy对象的时候，可以注册相应的事件来监听与网络状态相关的事件。错误信息，可根据错误编码参阅附录A。

8.1.1. JavaScript

```
var goEasy = new GoEasy({
    appkey: "my_appkey",
    onConnected: function () {
        alert("成功连接GoEasy。");
    },
    onDisconnected: function () {
       alert("与GoEasy连接断开。");
    },
    onConnectFailed: function (error) {
        alert("与GoEasy连接失败，错误编码："+error.code+"错误信息："+error.content);
    }
});
```

8.2. 消息发送（发布）

当您发布一条消息的同时，您可以注册一个监听器，来监听本条发送的结果。错误信息，可根据错误编码参阅附录A。

8.2.1. Java

```
goEasy.publish("my_channel","Hello, GoEasy!", new PublishListener(){
    @Override
    public void onSuccess() {
      System.out.print("消息发布成功。");
    }

    @Override
    public void onFailed(GoEasyError error) {
      System.out.print("消息发布失败, 错误编码：" + error.getCode() + " 错误信息： " +                    error.getContent());
    }
});
```

8.2.2. JavaScript

```
goEasy.publish({
    channel: "my_channel",
    message: "Hello GoEasy!",
     onSuccess:function(){
        alert("消息发布成功。");
     },
    onFailed: function (error) {
        alert("消息发送失败，错误编码："+error.code+" 错误信息："+error.content);
    }

});
```

8.3. 接收消息（订阅Channel）

当您订阅一个Channel的同时，您可以注册一个监听器来监听订阅相关的结果。错误信息，可根据错误编码参阅附录A。

8.3.1. JavaScript

```
goEasy.subscribe({
    channel: "my_channel",
    onMessage: function (message) {
        alert("您有新消息：channel：" + message.channel + " 内容：" + message.content);
    },
    onSuccess: function () {
        alert("Channel订阅成功。");
    },
    onFailed: function (error) {
        alert("Channel订阅失败, 错误编码：" + error.code + " 错误信息：" + error.content)
    }
});
```

8.4. 取消订阅Channel

您可以取消订阅一个Channel，您就可以停止继续收取来自这个channel的消息。您可以注册一个监听器来监听取消订阅相关的结果。错误信息，可根据错误编码参阅附录A。

8.4.1. JavaScript

```
goEasy.unsubscribe({
    channel: "my_channel",
    onSuccess: function () {
      alert("订阅取消成功。");
    },
    onFailed: function (error) {
      alert("取消订阅失败，错误编码：" + error.code + " 错误信息：" + error.content)
    }
});
```

9. GoEasy-OTP

9.1. 什么是GoEasy-OTP？有什么用途呢?

GoEasy-OTP(One-time password)是GoEasy对app key的安全保护措施，安全可靠，简单易用！
将会有效杜绝他人通过在页面上获取app key的方式，进行非法操作。

9.2. GoEasy-OTP是如何工作的呢？

```
var goEasy = new GoEasy({
    appkey: '您的client key',
    otp:'您服务器端生成的OTP'
});
```

1）当您在Javascript代码中初始化一个goEasy对象的时候，除了需要app key做为参数，还需要传入一个新参数：otp。
2）GoEasy-OTP是一个在您的服务器端按照GoEasy OTP的算法规则来生成的加密字符串，每个页面都需要一个独一无二的GoEasy-OTP。
3）GoEasy将会对OTP进行合法性验证，因为每个OTP只可以使用一次，即使他人从页面代码中获得了app key，也无法推送或接收消息。

9.3. 我想使用GoEasy-OTP来保护我的消息，要如何开始呢？

9.3.1. 登录GoEasy，找到“Professional keys”,您会发现3个key：

Client key: 仅用于客户端，必须和OTP一起使用，既可以推送，也可以订阅，服务器端无法使用。

Restful key: 仅用于调用Restful api，无法用于客户端订阅和推送。
Secret key：用于在生成GoEasy-OTP时，作为加密的秘钥。

9.3.2. 在server端生成GoEasyOTP，规则：

1） 声明一个字符串，内容为"000"+当前系统毫秒数
2） 将Secret key作为秘钥，用AES(ECB)算法对字符串进行加密
3） 使用Base64对加密结果进行编码，结果就是GoEasyOTP
4） 验证自己的OTP算法是否工作：
测试参数：
secret key：86726e4356dce2d3
系统毫秒数：0001490325990593
测试结果：+rOKqbTZioistsdMrhon0A==

9.3.3. 修改Javascript代码，使用Client key作为appkey，同时传入服务器端生成的OTP

```
var goEasy = new GoEasy({
    appkey: '您的client key',
    otp:'您服务器端生成的OTP'
});
```

9.3.4. 结束

9.4. 生成GoEasy-OTP参考代码

9.4.1. Java

```
public static String goEasyOTP(String secretKey) {
  try {
     String otp = "000" + System.currentTimeMillis();
     SecretKeySpec key = new SecretKeySpec(secretKey.getBytes(), "AES");
     Cipher cipher = Cipher.getInstance("AES/ECB/NoPadding");
     byte[] otpBytes = otp.getBytes();
     cipher.init(Cipher.ENCRYPT_MODE, key);
     byte[] encryptedOTP = cipher.doFinal(otpBytes);
     otp = new BASE64Encoder().encode(encryptedOTP);
     return otp;
```

```
    } catch (Exception e) {
        throw new IllegalStateException("Failed to generate GoEasy-OTP.", e);
    }
}
```

9.4.2. Python

```
from Crypto.Cipher import AES

def goEasyOTP(secretKey):
    otp = "000" + str(int(round(time.time() * 1000)))
    cipher = AES.new(secretKey, AES.MODE_ECB)
    encryptedOtp = cipher.encrypt(otp)
    encryptedOtp = base64.b64encode(encryptedOtp)
    return encryptedOtp
```

9.4.3. PHP

```
public  function  goEasyOTP($secretKey){
    $key = $secretKey;
    //$key=86726e4356dce2d3;
    list($t1, $t2) = explode(' ', microtime());
    $text=(float)sprintf('%.0f',(floatval($t1)+floatval($t2))*1000);
    $text = "000".$text;
    //$text = "0001490325990593";
    $iv_size = mcrypt_get_iv_size(MCRYPT_RIJNDAEL_128, MCRYPT_MODE_ECB);
    $iv = mcrypt_create_iv($iv_size, MCRYPT_RAND);
    $crypttext =base64_encode(mcrypt_encrypt(MCRYPT_RIJNDAEL_128, $key, $text, MCRYPT_MODE_ECB, $iv));
    return  $crypttext;
}
```

9.4.4. C#

```
public string goEasyOTP(string secretKey)
{
   string otp = null;
   byte[] encrypted = null;
```

```
    string currentTimeMills=string.Format("000{0}",(long)(DateTime.UtcNow-new
DateTime(1970,1,1,0,0,0,DateTimeKind.Utc)).TotalMilliseconds);
    //"0001490325990593"
    byte[] byteSecretKey = Encoding.ASCII.GetBytes(secretKey);
    using (AesManaged aesAlg = new AesManaged(){Key = byteSecretKey,Mode = CipherMode.ECB,Padding = PaddingMode.None})
    {
        ICryptoTransform encryptor = aesAlg.CreateEncryptor();
        using (MemoryStream msEncrypt = new MemoryStream())
       {
           using(CryptoStream csEncrypt = new CryptoStream(msEncrypt,encryptor,CryptoStreamMode.Write))
           {
               using(StreamWriter swEncrypt = new StreamWriter(csEncrypt))
               {
                   swEncrypt.Write(currentTimeMills);
               }
               encrypted = msEncrypt.ToArray();
           }
       }
    }
    otp = Convert.ToBase64String(encrypted);
    return otp;
}
```

9.4.5. Ruby

Coming...

附录：

A.返回值说明

Code	Content	状态

200	OK	成功
401	Unauthorized	app key 错误或者过期
400	<具体错误信息>	参数错误或不完整
408	Unreachable or timeout	发送数据到GoEasy失败，可能是因为网络原因，不能与GoEasy建立连接
500	Internal Server Error	GoEasy服务器错误
900	Over max concurrent connections	超过最大连接数
901	No remain messages	消息数量已用完
902	Please encode parameters with utf8	restful请求参数没有使用utf8进行编码
999	Your GoEasy application is expired, please contact your administrator to renew it	您的GoEasy application已经过期，请联系 您们的管理人员对其进行续费