

天御业务安全防护

验证码服务

产品文档



腾讯云

【版权声明】

©2013-2018 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

验证码服务

验证码服务产品简介

验证码服务购买指南

验证码服务开发指引

验证码开发指引

PC页面开发指引

H5开发指引

APP开发指引

验证码服务API文档

Android客户端API

IOS客户端API

H5网页API

PC网页API

后台获取验证码js地址

后台验证票据API

验证码服务

验证码服务产品简介

最近更新时间：2018-05-30 15:55:08

1. 天御验证码是什么

天御验证码是腾讯将多年积累的验证码技术通过天御平台开放的验证码服务，我们的验证码在腾讯多年业务发展过程中积累了大量的验证码对抗经验，无论从验证码的数量还是类型上都是领先的，希望通过验证码的技术开放能够为广大企业提供有效的恶意对抗能力。

2. 天御验证码类型

天御验证码的验证方式目前提供最新型的交互式验证码-滑块验证码。天御验证码服务为开发者提供由云端控制的验证页面，开发者只需要在云端配置验证方式，验证页面就可以实时生效，自由切换验证方式，无需APP或者Web站点的任何改动。

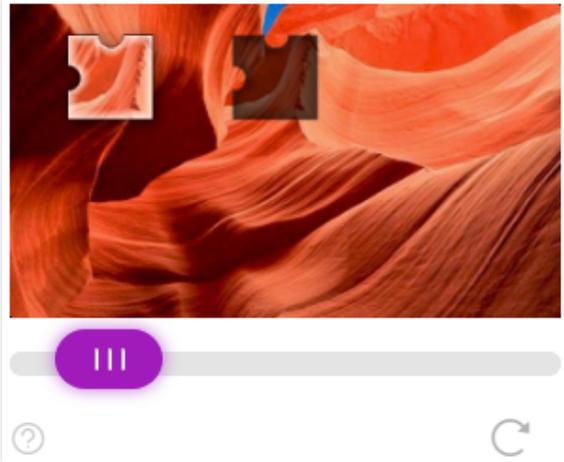
2.1 滑块验证码

滑块验证码是一种创新的交互式验证码，由策略制成、抗破解性更好，且用户只需轻轻一滑就可快速完成验证。



2.2 验证码参数

验证码类型	验证码类型参数	干扰程度参数	示例
-------	---------	--------	----

验证码类型	验证码类型参数	干扰程度参数	示例
滑块验证码 (新UI)	captchaType=9	disturbLevel=1	

3. 天御验证码支持哪些平台

目前天御验证码已支持在PC网页、手机网页和android/ios等平台上使用。手机上的验证码效果图：

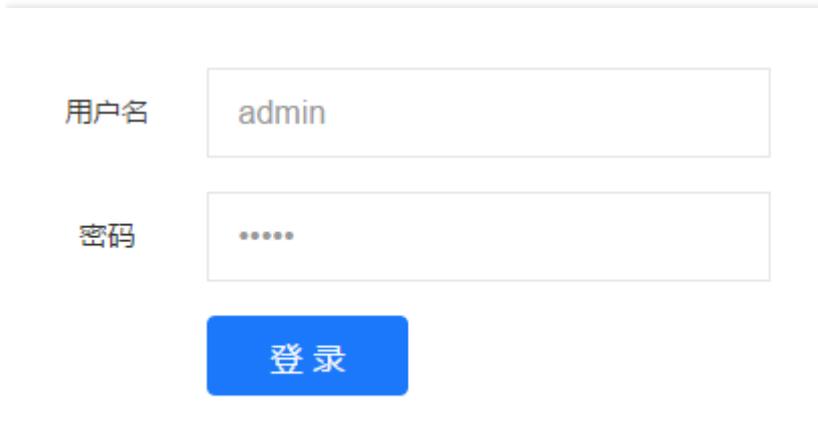


4. 天御验证码的优势

4.1 交互型验证码，支持后台切换验证类型

天御验证码为企业提供完整的验证页面，验证码类型由后台控制，切换时前端页面无需改动。支持多种页面显示方式。可以快速适用于各种业务场景，不影响原页面的排版布局和美观。交互型验证码和传统验证码相比，更灵活、验证方式更加多样化，能支持多种验证方式，不仅能更好的进行恶意对抗，用户体验上还更流畅和简单。网站接入交互型验证码后，页面只需显示验证区域让用户进行单击，验证码的验证过程将由天御验证码完成：

• 弹窗式：



The image shows a login form with two input fields: '用户名' (Username) containing 'admin' and '密码' (Password) containing six dots. Below the fields is a blue '登录' (Login) button. A white modal window is overlaid on top of the form, containing a security verification interface.

用户单击登录后弹出验证码：



• 嵌入式：

用户名

密码

验证码 

请向右拖动滑块完成验证

4.2 基于腾讯安全大数据的天御验证码策略

天御验证码除了提供验证码功能外，还为企业提供验证码下发策略。腾讯的安全大数据系统拥有十多年安全防护和恶意对抗经验沉淀，天御验证码策略运用安全大数据系统，能实时准确分析业务当前访问的恶意情况，让好人免验证或进行轻量的验证，对恶意用户下发高难度验证码，保证好人有良好体验的同时，达到对抗恶意的效果。

验证码服务购买指南

最近更新时间：2018-09-25 16:19:56

每日防护上限	包月价格 (包年价格 = 包月价格 * 10)
2 万次	1200 元
4 万次	1800 元
10 万次	3000 元
20 万次	5000 元
60 万次	12000 元

天御各服务对已企业认证的用户提供 7 天免费体验 (包含 1000 次使用)。体验服务结束后,您可以通过购买对应套餐继续享受安全服务。若线上套餐仍不能满足您的需求,可通过 [工单](#) 联系我们提供定制报价。

验证码服务开发指引

验证码开发指引

最近更新时间：2018-05-30 15:56:30



开发者需要通过调用云API的天御相关接口完成接入，接入API步骤如下：

1. 获取腾讯云密钥

如果您已经具有了腾讯云密钥则可以跳过这一步。进入[云API密钥](#)，选择"API密钥"，单击"新建密钥"。

个人 API 密钥 [云 API 使用文档](#)

使用 云 API 请在调用前使用 API 密钥获取签名，否则将无法调用 API，详见[鉴权签名算法](#)。

API 密钥是构建腾讯云 API 请求的重要凭证，使用腾讯云 API 可以操作您名下的所有腾讯云资源，为了您的财产和服务安全，请妥善保管和定期更换密钥，当您更换密钥后，请及时删除旧密钥。

+新建密钥 → 点击“新建密钥”

密钥	创建时间	状态	操作
SecretId: XXXXXXXXXX SecretKey: XXXXXXXXXX	2016-09-13 00:33:41	已启用	禁用

→ 获取腾讯云密钥

2. 进入天御业务安全防护管理中心

进入管理中心->云产品->天御业务安全防护。



3. 申请权限

在天御业务安全防护管理中心中单击“验证码服务”免费体验，确定开通。



4. 选择验证码接入模式，按照指引编写代码

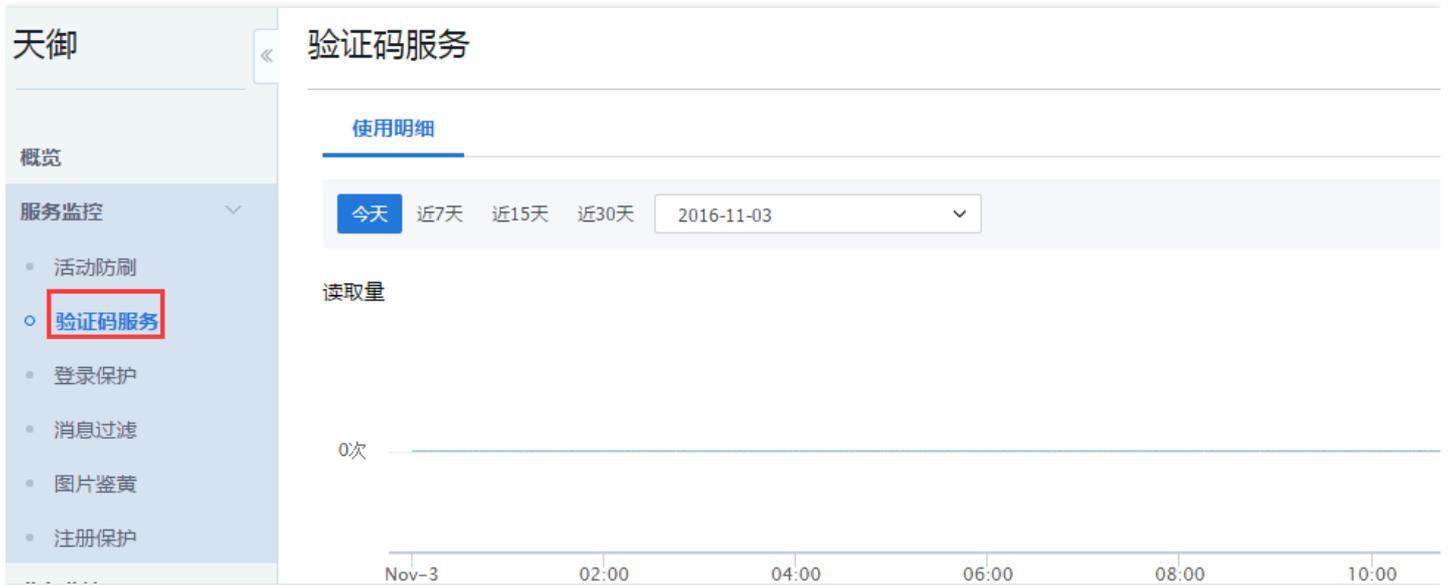
[PC客户端开发指引](#)

[H5开发指引](#)

[APP开发指引](#)

5. 查询调用数据

进入[天御业务安全防护管理中心](#)，选择“服务监控”，即可查询到对应服务的调用数据。



PC页面开发指引

最近更新时间：2017-03-09 02:23:49

1.接口调用流程



- 1) 后台通过调用天御的CaptchaIframeQuery接口获取验证码的js地址。
- 2) 把获取到的js地址回传给网页客户端。
- 3) 客户端依据获取到的回传的js地址加载和校验验证码。
- 4) 用户验证完成后提交天御返回的票据到后台。
- 5) 后台调用天御的CaptchaCheck接口来验证票据是否通过验证。

2.后台获取验证码js地址接口

[后台获取验证码API](#)

3.客户端加载验证码和获取票据接口

[PC页面API](#)

4.获取验证码票据

用户依据第3步获取的用户验证票据，提交到后台。

```
function cbfn(retJson){
    if(retJson.ret==0)    {
        //用户验证成功
        retJson.ticket;//用户票据

    }    else
    {
        //用户关闭验证码页面，没有验证
    }
}
```

5.后台校验验证码票据

后台依据第三步获取的用户票据，提交天御验证

[后台验证票据API](#)

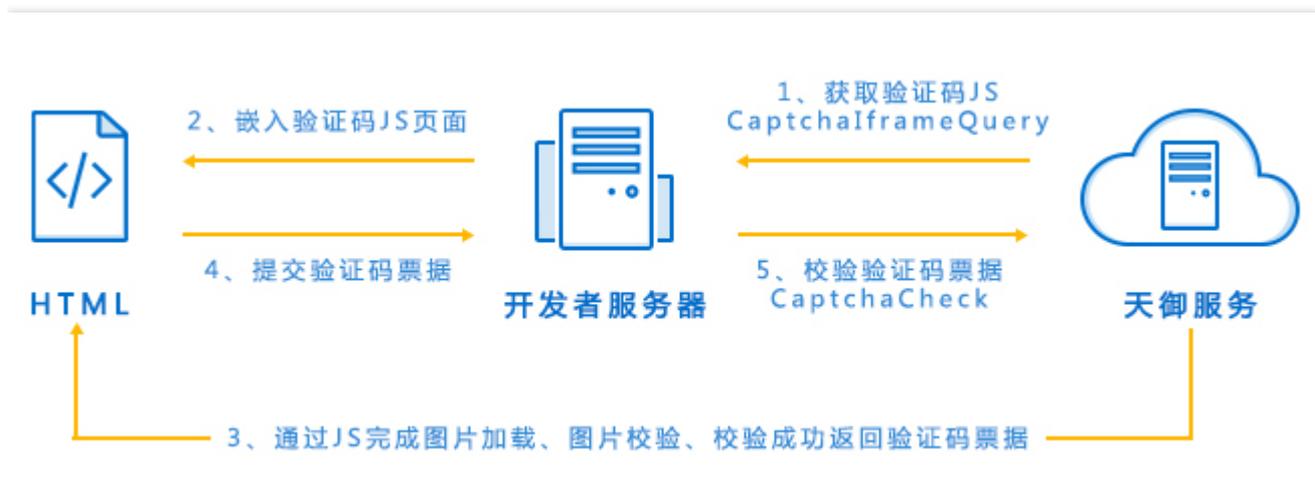
6.使用注意

- 1) 请不要使用iframe页面嵌入验证码。验证码弹出的iframe框大小会变化，如果业务使用iframe会导致验证码iframe页面显示不全。
- 2) PC预留给验证码展示的地方尺寸不能小于300px（宽）*310px（高），否则会导致验证码显示异常而影响用户使用。
- 3) PC页面必须设置验证码显示页面初始宽高。
- 4) 手机验证码页面要全屏显示，否则验证码页面会显示异常，影响用户使用。

H5开发指引

最近更新时间：2017-03-09 02:23:50

1.接口调用流程



- 1) 后台通过调用天御的CaptchaIframeQuery接口获取验证码的js地址。
- 2) 把获取到的js地址回传给网页客户端。
- 3) 客户端依据获取到的回传的js地址加载和校验验证码。
- 4) 用户验证完成后提交天御返回的票据到后台。
- 5) 后台调用天御的CaptchaCheck接口来验证票据是否通过验证。

2.后台获取验证码js地址接口

[后台获取验证码API](#)

3.客户端加载验证码和获取票据接口

[H5客户端API](#)

4.获取验证码票据

用户依据第3步获取的用户验证票据，提交到后台。

```
(retJson){  
    if(retJson.ret==0)    {  
        //用户验证成功  
        retJson.ticket;//用户票据  
  
    }    else  
    {  
        //用户关闭验证码页面，没有验证  
    }  
}
```

5.后台校验验证码票据

后台依据第三步获取的用户票据，提交天御验证

[后台验证票据API](#)

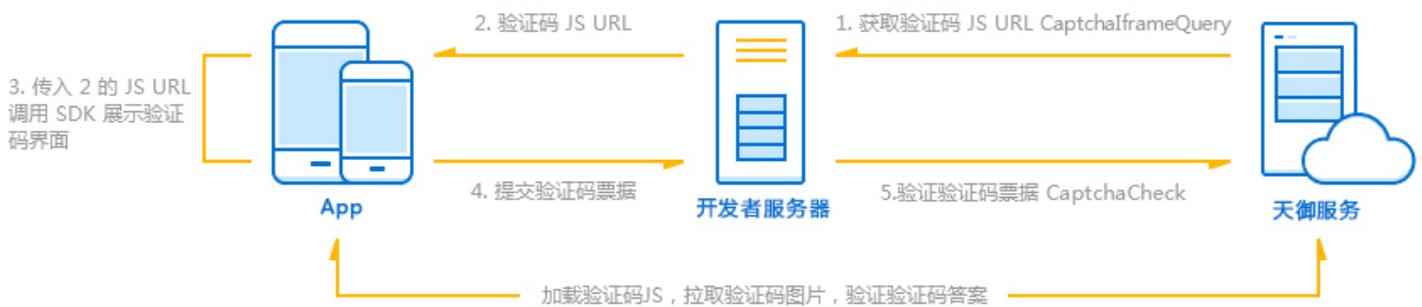
6.使用注意

手机验证码页面要全屏显示，否则验证码页面会显示异常，影响用户使用。

APP开发指引

最近更新时间：2017-03-09 02:23:51

1.接口调用流程



- 1) 后台通过调用天御的CaptchaIframeQuery接口获取验证码的js地址。
- 2) 把获取到的js地址回传给网页客户端。
- 3) 客户端依据获取到的回传的js地址加载和校验验证码。
- 4) 用户验证完成后提交天御返回的票据到后台。
- 5) 后台调用天御的CaptchaCheck接口来验证票据是否通过验证。

2.后台获取验证码js地址接口

[后台获取验证码API](#)

3.客户端加载验证码和获取票据接口

[IOS客户端API](#)

[Android客户端API](#)

4.获取验证码票据

用户依据第3步获取的用户验证票据，提交到后台。

5.后台校验验证码票据

后台依据第三步获取的用户票据，提交天御验证

[后台验证票据API](#)

6.使用注意

手机验证码页面要全屏显示，否则验证码页面会显示异常影响用户使用。

验证码服务API文档

Android客户端API

最近更新时间：2018-06-11 15:24:08

Android 应用

接入要求

Android 系统 2.3 以上。

开发步骤

1.将 VerifySDK.jar 拷贝到 libs 目录下关联到工程。

[Android-SDK 下载](#)

2.如 AndroidManifest.xml 未声明以下权限，则添加声明。

```
<uses-permission android:name="android.permission.INTERNET" />
```

3.AndroidManifest.xml 中添加声明。

```
<!--说明com.example.verifydemo业务可替换成自己包名，VerifyFullScreenActivity为全屏显示验证码，VerifyPopupActivity为弹框显示验证码-->
<activity android:name="com.example.verifydemo.VerifyFullScreenActivity"></activity>
<activity android:name="com.example.verifydemo.VerifyPopupActivity" android:theme="@style/dialog"></activity>
```

4.需要下发验证码前从后台获取 jsurl（参考服务端开发获取验证码 JSURL 的接口）。

5.根据业务需要实现全屏验证码界面 VerifyFullScreenActivity 或弹框验证码界面 VerifyPopupActivity。

```
//VerifyPopupActivity onCreate实现实例：
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    this.requestWindowFeature(Window.FEATURE_NO_TITLE);
    String jsurl = getIntent().getStringExtra("jsurl");
    if (jsurl == null) {
        finish();
    }
    return;
}
```

```
}

WindowManager manager = getWindowManager();
DisplayMetrics metrics = new DisplayMetrics();
manager.getDefaultDisplay().getMetrics(metrics);
mDensity = metrics.density;
int windowWidth = metrics.widthPixels;

/*
 * 以滑动拼图弹框验证码为例，取弹框验证码宽度为屏幕宽度0.7
 * 滑动拼图标准宽18.2*16dp，标准高16.1*16dp,最大缩放比例2 ----capType=7
 * 图中点字标准宽18.2*16dp，标准高19.6*16dp,最大缩放比例2 ----capType=4,6
 */
int iframeWidthPX = (int) (windowWidth * mScale);
int iframeWidthDP = (int) (iframeWidthPX/mDensity);
if (iframeWidthDP >= (int) (F_DEFAULT_POPUP_IFRAME_WIDTH*F_MAX_IFRAME_WIDTH_SCALE)){
    iframeWidthDP = (int) (F_DEFAULT_POPUP_IFRAME_WIDTH*F_MAX_IFRAME_WIDTH_SCALE);
    iframeWidthPX = (int) (iframeWidthDP*mDensity);
}
//根据验证码类型和弹框宽度，获取验证码弹框高度
int iframeHeightDP = VerifyCoder.getPopupIframeHeightByWidthAndCapType(iframeWidthDP,F_CAP_TYPE_SLIDE_PUZZLE);
int iframeHeightPX = (int) (iframeHeightDP * mDensity);

//设置主题色，弹框验证码，弹框宽度
VerifyCoder verifyCoder = VerifyCoder.getVerifyCoder();
verifyCoder.setJson("themeColor:'ff0000',type:'popup',fwidth:" +iframeWidthDP);
mWebView = verifyCoder.getWebView(getApplicationContext(), jsurl, mListener);
mWebView.requestFocus();
mWebView.forceLayout();

//业务可根据自己需要实现不同的loading展现
setContentView(R.layout.activity_verify_popup);
mContainer = (RelativeLayout)findViewById(R.id.container);
mProgressBar = (ProgressBar)findViewById(R.id.progressBar);
mWebView.setVisibility(View.INVISIBLE);
mContainer.addView(mWebView);
android.view.WindowManager.LayoutParams attributes = getWindow().getAttributes();
attributes.width = iframeWidthPX;
attributes.height = iframeHeightPX;
getWindow().setAttributes(attributes);
}

//回调VerifyListener实例
private VerifyListener mListener = new VerifyListener() {
```

@Override

```
public void onVerifySucc(String ticket, String randstr) {  
    //验证成功回调  
    Intent it = new Intent();  
    it.putExtra("ticket", ticket);  
    it.putExtra("randstr", randstr);  
    setResult(Activity.RESULT_OK, it);  
    finish();  
}
```

@Override

```
public void onVerifyFail() {  
    //验证不成功回调，如用户单击返回或关闭按钮  
    setResult(Activity.RESULT_CANCELED);  
    finish();  
}
```

@Override

```
public void onframeLoaded(int state, String info) {  
    //收到验证码页面(包括图片)加载完成回调时，把Loading隐藏，WebView显示  
    mProgressBar.setVisibility(View.INVISIBLE);  
    mWebView.setVisibility(View.VISIBLE);  
}
```

@Override

```
public void onFrameResize(float width, float height) {  
    //验证码弹框宽度，高度发生变化时回调  
    android.view.WindowManager.LayoutParams attributes = getWindow().getAttributes();  
    attributes.width = (int)(width*mDensity);  
    attributes.height = (int)(height*mDensity);  
    getWindow().setAttributes(attributes);  
}  
  
};
```

6. 获取到 jsurl 后调用 startVerifyActivityResult(Context context,String jsurl,int requestCode) 并实现 onActivityResult 来接收是否验证成功的通知。

```
Intent intent = new Intent(this,VerifyFullScreenActivity.class);  
intent.putExtra("jsurl", jsurl);  
startActivityResult(it,requestCode);
```

//onActivityResult实现实例：

@Override

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode == 1) {//此处对应startVerifyActivityForResult的参数值  
        if(resultCode==Activity.RESULT_OK){  
            Log.e("onActivityResult", "verifysucc");  
            Toast.makeText(MainActivity.this, "验证成功",2000).show();  
        }  
        else{  
            Toast.makeText(MainActivity.this, "未验证成功",2000).show();  
        }  
    }  
}
```

7.如有混淆，需要添加脚本。

```
<arg value="-libraryjars ${lib}/VerifySDK.jar"/>  
<arg value="-keep public class com.token.verifysdk{*;}"/>
```

其它接口说明

```
public static VerifyCoder getVerifyCoder() //获取单例  
public void release() //重置参数，释放资源  
public void setShowtitle(boolean showtitle) //是否显示验证码页面标题栏  
public void setJson(String json) //用于扩展参数，如实现自定义样式等  
public WebView getWebView(Context context,String jsurl,VerifyListener listener) //获取验证码WebView  
  
public static int getPopuiframeHeightByWidth(int width)//根据滑动拼图弹框验证码宽度(单位dp)获取弹框验证码高度  
public static int getPopuiframeHeightByWidthAndCapType(int width, int capType)//根据弹框验证码宽度(单位dp)和验证码类型获取弹框验证码高度，图中点字类型4或6，滑动拼图类型7
```

IOS客户端API

最近更新时间：2017-08-31 17:28:19

IOS 应用

1 概述

SDK工具包目录结构说明:

- TCWebCodesSDK.framework:包含TCWebCodesSDK.framework
- TCWebCodesSDKDemo:示例工程,演示了如何使用TCWebCodesSDK.framework

本SDK运行环境与项目要求:

适用于iOS6.0及以上的系统版本

[IOS-SDK下载](#)

2 接口说明

```
/**
 设置回调
  @note 成功/失败可以通过 resultJSON["@ret"] 判断，0为成功，非0为失败
  @warning 为了避免引用，内部再回调完成后会重置回调为空，请记得每次都需要设置回调
 */
@property (nonatomic, copy) void (^callback)(NSDictionary *resultJSON, UIView *webView);
```

主要函数，设置验证结果的callback回调；

```
/**
 开始加载H5
  @param url，业务服务器通过腾讯云拉取的验证码URL
  @param frame, webView控件大小
  @return 返回H5的webView控件
  @note 该函数用于开始加载，并且返回H5的webView控件，用于显示
  @warning 请不要设置返回webView控件的回调
 */
- (UIView*)startLoad:(NSString*)url webFrame:(CGRect)frame;
```

主要函数，通过腾讯云返回的验证码URL创建一个用于显示的webView控件；

```
/**  
  返回单例  
*/  
+ (instancetype)sharedBridge;
```

返回 TCWebCodesBridge 全局对象；

```
/**  
  设置是否显示H5的内部导航头部  
*/  
@property (nonatomic) BOOL showHeader;
```

配置函数，设置是否显示H5页面的头部，默认不显示；

```
/**  
  设置其余属性  
  @note 该接口是为了后续扩展  
  @warning 请设置value为基本类型: string, number  
*/  
- (void)setCapValue:(id)aValue forKey:(id<NSCopying>)aKey;
```

可扩展函数，用于设置后续显示H5界面的一些样式；

H5网页API

最近更新时间：2017-10-30 16:19:00

1. 说明

本节主要描述 H5 页面的开发方法

2. H5 页面开发步骤

2.1 页面头部引入 JS

```
<script type="text/javascript" src="xxx"></script>
```

提示：[JS地址的获取方法](#)

2.2 添加验证码展示控件

```
<div id="TCaptcha" style="" ></div>
```

2.3 初始化并显示验证码

```
<script type="text/javascript">
var capOption = {callback:cbfn, showHeader:true};
capInit(document.getElementById("TCaptcha"), capOption);
//回调函数：验证码页面关闭时回调
function cbfn(retJson) {
if (retJson.ret == 0) {
// 用户验证成功
}
else {
//用户关闭验证码页面，没有验证
}
}
</script>
```

3 完整页面代码示例

```

<!DOCTYPE html>
<html>
<head>
<script type="text/javascript" src="xxx"></script>
<meta name="viewport" content="initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no, width=device-width">
</head>

<body>
<form action="xxx" id="myform" method="post">
<input type="hidden" id="ticket" name="ticket" value="">
<div id="TCaptcha" style="" ></div>
</form>
<script type="text/javascript">
var capOption = {callback:cbfn, showHeader:true};
capInit(document.getElementById("TCaptcha"), capOption);
//回调函数：验证码页面关闭时回调
function cbfn(retJson) {
if (retJson.ret == 0) {
// 用户验证成功
document.getElementById("ticket").value = retJson.ticket;
document.getElementById("myform").submit();
}
else {
//用户关闭验证码页面，没有验证
}
}
</script>
</body>
</html>
    
```

4. javascript 接口说明

函数名	描述
capInit(iframe_div, options)	初始化并显示验证码,参数如下： <ol style="list-style-type: none"> iframe_div (必填)：嵌入验证码 iframe 的元素。 options：{callback:xxx,showheader:xxx, themeColor:xxxxxx,type:"embed"}，json 格式对象

函数名	描述
	<p>callback：验证码页面关闭回调函数。用户验证之后，会调用该函数，传入json格式验证参数。</p> <pre data-bbox="453 405 1444 694"> {ret:xxx,ticket:"xxx"} ret=0 表示用户验证完成，业务可以校验 ticket； ret=1 表示用户未验证验证码，此时没有 ticket 参数。 参数 ticket 需要提交给业务后台，具体填哪个字段参考后面后台 server 开发部分。 </pre> <p>themeColor：设置页面的主题色彩，值为 16 进制色彩，比如 ff572d。设置后页面里的按钮和图标会变成设置的颜色</p> <p>showHeader：显示验证码页面的 header (返回和帮助，只对手机页面有效)</p> <pre data-bbox="453 902 1444 1025"> false：不显示 </pre> <p>type：PC 端可选项，配置验证码的样式。具体样式表现可以查看验证码官网</p> <pre data-bbox="453 1149 1444 1357"> "point"：触发式（默认） "embed"：嵌入式 "popup"：弹窗式 </pre> <p>pos：设置弹框验证码的位置属性，该参数只对 PC 弹框验证码有效</p> <pre data-bbox="453 1480 1444 1731"> absolute: 绝对定位 fixed：相对于浏览器窗口的绝对定位 static：静态定位 relative：相对定位 </pre> <p>keepOpen：设置验证通过页面属性</p> <pre data-bbox="453 1854 1444 2018"> false：验证通过刷新（默认） true：保持显示，不刷新 </pre>

函数名	描述
	<p>lang : 设置验证码语言类型</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"><p>简体中文 : 2052 (默认)</p><p>繁体中文 : 1028</p><p>英文 : 1033</p></div>
capGetTicket()	获取验证码验证结果 返回 Josn 格式数据{ret: 0, randstr: "xxx"}其中 ticket 是验证码验证成功票据, 如果票据为空表示验证码验证没通过
capRefresh()	刷新验证码 要求用户重新验证当票据验证失败时也可调用该接口刷新验证码
capDestroy()	无参数, 当 dom 被销毁需要重新使用 capInit 的时候, 在 capInit 之前调用

5.接入规范

手机验证码页面要全屏显示, 否则验证码页面会显示异常影响用户使用。

PC网页API

最近更新时间：2018-08-21 16:29:34

1. 说明

本节主要描述PC页面的开发方法

2. PC 页面开发步骤

2.1 页面头部引入 JS

```
<script type="text/javascript" src="xxx"></script>
```

[JS地址的获取方法](#)

2.2 添加验证码展示控件

```
<!--显示验证码的地方预留的最小空间，小于该值会导致显示异常-->  
<!--触发式：300px*40px（宽*高）-->  
<!--嵌入式：300px*270px-->  
<!--弹窗式：300px*310px-->  
<!--通过width和height设置验证码初始宽高，未设置会导致显示异常，默认触发式-->  
<div id="TCaptcha" style="width:300px;height:40px;" ></div>
```

2.3 初始化并显示验证码

```
<script type="text/javascript">  
var capOption={callback :cbfn};  
capInit(document.getElementById("TCaptcha"), capOption);  
//回调函数：验证码页面关闭时回调  
function cbfn(retJson)  
{  
if(retJson.ret==0)  
{  
// 用户验证成功,需要校验签名  
}  
else  
{  
//用户关闭验证码页面，没有验证  
}}
```

```

}
}
</script>
    
```

3. 完整页面代码示例

```

<!DOCTYPE html>
<html>
<head>
<script type="text/javascript" src="xxx"></script>
</head>

<body>
<form action="xxx" id="myform" method="post">
<input type="hidden" id="ticket" name="ticket" value="">
<div id="TCaptcha" style="width:310px;height:40px;" ></div>
</form>
<script type="text/javascript">
var capOption = {callback:cbfn, themeColor:"a11bbb"};
capInit(document.getElementById("TCaptcha"), capOption);
//回调函数：验证码页面关闭时回调
function cbfn(retJson) {
if (retJson.ret == 0) {
// 用户验证成功
document.getElementById("ticket").value = retJson.ticket;
document.getElementById("myform").submit();
}
else {
//用户关闭验证码页面，没有验证
}
}
</script>
</body>
</html>
    
```

4. javascript 接口说明

函数名	描述
capInit(iframe_div,	初始化并显示验证码,参数如下：

options) 函数名	描述
	<p>1. <code>iframe_div</code> (必填) : 嵌入验证码 <code>iframe</code> 的元素。</p> <p>2. <code>options</code> : {<code>callback:xxx,showheader:xxx, themeColor:xxxxxx,type:"embed"</code>} , json 格式对象</p> <div style="background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p>callback : 验证码页面关闭回调函数。用户验证之后, 会调用该函数, 传入json 格式验证参数。</p> <pre style="background-color: #fff; padding: 5px; border: 1px solid #ccc;">{ret:xxx,ticket:"xxx"}</pre> <p><code>ret=0</code> 表示用户验证完成, 业务可以校验 <code>ticket</code> ; <code>ret=1</code> 表示用户未验证验证码, 此时没有 <code>ticket</code> 参数。 参数 <code>ticket</code> 需要提交给业务后台, 具体填哪个字段参考后面后台 <code>server</code> 开发部分。</p> </div> <p>themeColor : 设置页面的主题色彩, 值为 16 进制色彩, 比如 <code>ff572d</code>。设置后页面里的按钮和图标会变成设置的颜色</p> <p>showHeader : 显示验证码页面的 <code>header</code> (返回和帮助, 只对手机页面有效)</p> <div style="background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p><code>false</code> : 不显示</p> </div> <p>type : PC 端可选选项, 配置验证码的样式。具体样式表现可以查看验证码官网</p> <div style="background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p><code>"point"</code> : 触发式 (默认) <code>"embed"</code> : 嵌入式 <code>"popup"</code> : 弹窗式</p> </div> <p>pos : 设置弹框验证码的位置属性, 该参数只对 PC 弹框验证码有效</p> <div style="background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p><code>absolute</code>: 绝对定位 <code>fixed</code> : 相对于浏览器窗口的绝对定位 <code>static</code> : 静态定位 <code>relative</code> : 相对定位</p> </div> <p>keepOpen : 设置验证通过页面属性</p>

函数名	描述
	<p>false : 验证通过刷新 (默认) true : 保持显示, 不刷新</p> <p>lang : 设置验证码语言类型</p> <p>简体中文 : 2052 (默认) 繁体中文 : 1028 英文 : 1033</p>
capGetTicket()	获取验证码验证结果 返回 Josn 格式数据{ret: 0, randstr: "xxx"}其中 ticket 是验证码验证成功票据, 如果票据为空表示验证码验证没通过
capRefresh()	刷新验证码 要求用户重新验证当票据验证失败时也可调用该接口刷新验证码
capDestroy()	无参数, 当 dom 被销毁需要重新使用 capInit 的时候, 在 capInit 之前调用

5. 接入规范

- 1) 请不要使用 iframe 页面嵌入验证码。验证码弹出的 iframe 框大小会变化, 如果业务使用 iframe 会导致验证码 iframe 页面显示不全。
- 2) PC 预留给验证码展示的地方尺寸不能小于300px (宽) * 310px (高), 否则会导致验证码显示异常而影响用户使用。
- 3) PC 页面必须设置验证码显示页面初始宽高。

后台获取验证码js地址

最近更新时间：2017-06-06 15:13:40

1.接口描述

协议：HTTPS

域名：csec.api.qcloud.com

接口名: CaptchalframeQuery

获取验证码的JavaScript连接，通过将验证码的JavaScript嵌入页面实现验证码的刷新和验证操作。

2.输入参数

以下请求参数列表仅列出了接口请求参数，正式调用时需要加上公共请求参数，见[公共请求参数](#)页面。

其中，此接口的Action字段为CaptchalframeQuery。

参数名称	是否必须	类型	描述
captchaType	必选	UInt	验证码类型
disturbLevel	必选	UInt	验证码干扰程度
isHttps	必选	UInt	返回的JavaScript中是否使用HTTPS 0：HTTP 1：HTTPS
clientType	必选	UInt	客户端类型 1：手机Web页面 2：PCWeb页面 4：APP
accountType	必选	UInt	用户账号类型 0：其他账号 1：QQ开放帐号 2：微信开放帐号 4：手机账号 6：手机动态码 7：邮箱账号

appId	建议	String	accountType是QQ或微信开放账号时，该参数必填，表示QQ或微信分配给给网站或应用的appId，用来唯一标识网站或应用
uid	建议	String	用户IDaccountType不同对应不同的用户ID。如果是QQ或微信用户则填入对应的openId
businessId	建议	UInt	业务ID 网站或应用多个业务中使用此服务，通过此ID区分统计数据
registerTime	可选	UInt	注册时间戳，单位秒
userIp	可选	String	用户操作来源的外网IP
xForwardedFor	可选	String	用户Http请求中的x_forward_for
macAddress	可选	String	mac地址或设备唯一标识
imei	可选	String	手机设备号
associateAccount	可选	String	accountType是QQ或微信开放账号时，用于标识QQ或微信用户登录后关联业务自身的账号ID
sceneId	可选	UInt	场景ID 网站或应用的业务下有多个场景使用此服务，通过此ID区分统计数据

3.输出参数

参数名称	类型	描述
code	Int	公共错误码，0表示成功，其他值表示失败。详见错误码页面的 公共错误码
codeDesc	String	业务侧错误码。成功时返回Success，错误时返回具体业务错误原因。
message	String	模块错误信息描述，与接口相关
url	String	验证码JavaScript地址，该链接单次有效

4.示例代码

代码下载：[java](#) [php](#) [Python](#)

一个完整的请求需要两类请求参数：公共请求参数和接口请求参数。这里只列出了接口请求参数，并未列出公共请求参数，有关公共请求参数的说明可见[公共请求参数](#)小节。

请求示例：

```
https://csec.api.qcloud.com/v2/index.php?Action=CaptchalframeQuery
```

&<公共请求参数>

```
&secretId=AKIDmQtAxYTAB2iBS8s2DCzazCD2g7OUq4Zw
```

```
&captchaType=1
```

```
&disturbLevel=1
```

```
&isHttps=1
```

```
&clientType=1
```

5. 响应示例

```
{  
  "code":0,  
  "message":"No Error",  
  "url":"https://captcha.guard.qcloud.com/template/TCapIframeApi.js?appid=1251001047&clienttype=1  
&lang=2052&asig=-DhJtUkDwLzJpmlfAmasXFn1Y6zCkRQUn8WERrs4IVNmUDcuoDiYYLmoKqd-Ev77  
Eogpq97Dpb69_MrwGjWXKmTGg9y9iW7wjdrTu_y6WBN4qGsHn6VRk0W1hLB6ZWvqHqw2E5IFCRUcG  
rHBzMF7A**"  
}
```

后台验证票据API

最近更新时间：2017-06-06 13:00:37

1.接口描述

协议：HTTPS

域名：csec.api.qcloud.com

接口名: CaptchaCheck

用户输入验证码之后会获取API返回的票据，必须将此票据通过本接口进行校验，以确认票据是从安全API返回的，否则将可能导致验证码功能被绕过

2.输入参数

以下请求参数列表仅列出了接口请求参数，正式调用时需要加上公共请求参数，见[公共请求参数](#)页面。其中，此接口的Action字段为CaptchaCheck。

参数名称	是否必须	类型	描述
ticket	必选	String	API返回给用户的票据
captchaType	必选	Int	验证码类型
userIp	必选	String	用户操作来源的外网IP
accountType	必选	UInt	用户账号类型 0：其他账号 1：QQ开放帐号 2：微信开放帐号 4：手机账号 6：手机动态码 7：邮箱账号
appId	建议	String	accountType是QQ或微信开放账号时，该参数必填，表示QQ或微信分配给给网站或应用的appId，用来唯一标识网站或应用
businessId	可选	UInt	业务ID，网站或应用在多个业务中使用此服务，通过此ID区分统计数据

sceneId	可选	UInt	场景ID，网站或应用的业务下有多个场景使用此服务，通过此ID区分统计数据
uid	可选	String	用户ID，accountType不同对应不同的用户ID。如果是QQ或微信用户则填入对应的openId
associateAccount	可选	String	accountType是QQ或微信开放账号时，用于标识QQ或微信用户登录后关联业务自身的账号ID
registerTime	可选	UInt	注册时间戳，单位秒
xForwardedFor	可选	String	用户Http请求中的x_forward_for
macAddress	可选	String	mac地址或设备唯一标识
imei	可选	String	手机设备号

3.输出参数

参数名称	类型	描述
code	Int	公共错误码，0表示成功，其他值表示失败。详见错误码页面的[公共错误码] (https://cloud.tencent.com/document/product/295/7285)
codeDesc	String	业务侧错误码。成功时返回Success，错误时返回具体业务错误原因。
message	String	模块错误信息描述，与接口相关

4.示例代码

代码下载：[java](#) [php](#) [Python](#)

一个完整的请求需要两类请求参数：公共请求参数和接口请求参数。这里只列出了接口请求参数，并未列出公共请求参数，有关公共请求参数的说明可见[公共请求参数](#)小节。

请求示例：

```
https://csec.api.qcloud.com/v2/index.php?Action=CaptchaCheck
&<公共请求参数>
&ticket=1111
&captchaType=1
&disturbLevel=1
&userIp=127.0.0.1
```

5. 响应示例

```
{  
  "code":0,  
  "message":"No Error",  
  "is_right":0  
}
```