

CameraSDK 接入指南

开发者下载版 · 内容与开放平台在线文档保持一致

SDK 接入

简介

主要特性

- 高清拍照：拍照分辨率支持 1440×1920
- 丰富水印：支持多种水印模板（自定义、时间地点、打卡、工程、巡逻、物业等）
- 横竖屏支持：自动检测设备方向，支持横竖屏拍摄
- 防伪码生成：自动生成带防伪码的图片，支持平台验真
- 位置信息：集成地址、经纬度、海拔等信息
- 天气信息：显示实时天气数据
- 内容编辑：支持自由添加编辑水印条目

核心优势

时间防篡改+照片验真技术 双重校验保证照片真实性：

- 真实时间：中科院国家授时中心授时，自动检测照片时间，发现篡改立即修正
- 真实地点：通过Wi-Fi定位/基站定位等多种方式辅助定位保证定位准确

照片验真中心：

对 照片分辨率 / 防伪码 / 异常情况 / 时间 / 经纬地点 进行5道程序校验，结果真实可信

丰富的水印模版：

6个水印模版，用量角度覆盖率92%。包含：时间地点天气、工程水印、考勤打卡水印、自定义水印、执勤水印、物业水印

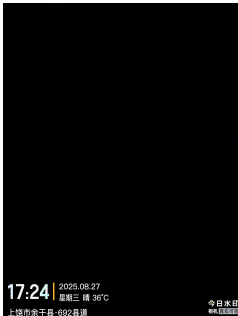
领先行业的拍摄体验：

支持广角与智能极速拍照，内置高效水印合成，实现更快成片。

SDK效果展示

图片为微信小程序插件截图



拍摄效果1	拍摄效果2
	

H5 版本

前提：必须在 https 环境中使用，否则无法打开相机、无法获取定位！

在 html 中引用 js 文件

```
<script src="https://static.xhey.top/sdk/prod/xhey-camera-sdk.v1.0.2.min.js"></script>
```

向商务获取 appid 和 secretkey，签名生成方法参考当前文档鉴权部分；

初始化

初始化相机实例，如果 eslint 报错，添加 // eslint-disable-next-line no-undef

```
// eslint-disable-next-line no-undef
let camera = new XHeyCamera({
  // 【可选】是否需要关闭拍照确认页
  disablePhotoConfirm: false,

  // 【可选】开启自定义水印编辑，用户可以自己选择水印和编辑水印条目
  // 开启后，下面的watermarkId、title、logoUrl、customInputItems设置不生效
  enableCustomWatermark: false,

  // 【可选】手机型号，比如SM-S918U1
  // 如果可以获取到，最好传入，可以解决一些手机兼容性问题
  deviceModel: "your device model",

  // 【可选】手机品牌比如samsung
  // 如果可以获取到，最好传入，可以解决一些手机兼容性问题
  deviceBrand: "your device brand",

  // 【必须】应用 ID (替换为你的应用 ID)
  appid: "your appid",

  // 【必须】时间戳 (单位：秒)
  timestamp: 1736928042,

  // 【必须】随机字符串
  noncestr: "65cdfefecaf3a0c4",

  // 【必须】签名 (生成的签名)，最好在服务端做，防止密钥泄漏
  signature: "your generated signature",

  // 【可选】水印 ID
  watermarkId: watermarkId,

  // 【可选】自定义水印标题
  title: "自定义会议标题",

  // 【可选】自定义水印中的 Logo 图片 URL
  logoUrl: "",

  // 【可选】自定义水印中的项内容
  customInputItems: {
    "会议名称": "这是自定义的会议名称",
    "会议类型": "这是自定义的会议类型",
  },

  // 【可选】最大拍摄照片张数
```

```

maxImageCount: 1,

// 【可选】输出log回调
enableLogMessage: true,

// 【可选】关闭右下角官方水印
disableOfficialWatermark: false,

// 【可选】禁止水印拖动
disableWatermarkDragging: false,
});

```

设置回调

```

// maxImageCount为1时回调，拍照成功回调
camera.onSuccess((imageBase64Data, userCommentObject) => {
  // imageBase64Data 是图片的 Base64 数据
  // userCommentObject 有水印信息
  this.imageUrl = imageBase64Data;
  console.log("拍照成功");
});

// 取消操作回调
camera.onCancel(() => {
  console.log("取消操作");
});

// 错误回调
camera.onError((error) => {
  console.error("发生错误:", error);
});

// maxImageCount大于1时回调
camera.onCaptureStillImages((imageBase64DataArray) => {
  console.log("拍照成功:", imageBase64DataArray);
  this.splitLogos = imageBase64DataArray.map((imageBase64Data) => [
    { url: imageBase64Data },
  ]);
});

// logInfo回调
camera.onLogInfo((message) => {
  console.log(message);
});

// logError回调
camera.onLogError((message) => {
  console.error(message);
});

// logWarn回调
camera.onLogWarn((message) => {
  console.warn(message);
});

```

开始拍照

```
camera.takePhoto();
```

主动关闭拍照页

一般情况下不需要手动调用下面的方法

```

// 关闭拍照页
camera.cancel()

// 彻底释放相机页面资源，在某些环境下，再次打开相机拍照页可能会需要再次授权相机权限；
camera.dispose()

```

Flutter 版本

请向商务获取最新版本 SDK、appid、secretkey；

添加依赖：

```

YAML
dependencies:
  flutter:

```

```

    sdk: flutter
    plugin_platform_interface: ^2.0.2

xheycamerasdk:
  # SDK 路径
  path: ../xheycamerasdk

```

在 AndroidManifest.xml 里声明权限：

XML

```

<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-feature android:name="android.hardware.camera" android:required="true" />

```

iOS info.plist 里声明权限：

XML

```

<key>NSLocationAlwaysAndWhenInUseUsageDescription</key>
<string>我们需要精确的位置信息来提供准确的服务</string>
<key>NSLocationAccuracy</key>
<string>我们需要精确的位置信息来提供准确的服务</string>
<key>NSLocationWhenInUseUsageDescription</key>
<string>我们需要在后台获取您的位置来提供服务</string>
<key>NSCameraUsageDescription</key>
<string>我们需要使用您的相机来拍摄照片</string>

```

iOS Podfile 配置（按 SDK example）：

RUBY

```

target 'Runner' do
  # use_frameworks!

  flutter_install_all_ios_pods File.dirname(File.realpath(__FILE__))
  target 'RunnerTests' do
    inherit! :search_paths
  end
end

post_install do |installer|
  installer.pods_project.targets.each do |target|
    flutter_additional_ios_build_settings(target)

    target.build_configurations.each do |config|
      config.build_settings['GCC_PREPROCESSOR_DEFINITIONS'] ||= [
        '(inherited)',
      ]
      config.build_settings['ENABLE_BITCODE'] = 'NO'
      config.build_settings['SWIFT_VERSION'] = '4.2'
    end
  end
end

```

拍照，创建 XheyCameraSdk 实例并跳转到拍照页面（参考 example/lib/main.dart）：

DART

```

void takePhoto(BuildContext context) {
  _xheyCameraSdk = XheyCameraSdk(
    appid: "your app id",
    secretKey: "your secret key",
    // 【可选】不传的话，用户可以自己选水印，传入的话就用配置的团队水印
    groupWatermarkId: "",
    delegate: MyXheyCameraSdkDelegate(state: this),
  );
  _xheyCameraSdk?.takePhoto();
}

```

设置结果回调：

DART

```

class MyXheyCameraSdkDelegate implements XheyCameraSdkDelegate {
  final _MyAppState state;

  MyXheyCameraSdkDelegate({required this.state});

  @override
  void onCancel() {

```

```

    print("[MyApp]onCancel");
}

@override
void onSuccess(List<Image> images) {
  if (images.isEmpty) {
    print("[MyApp]onSuccess: No images");
    return;
  }

  state.setState(() {
    state._image = images.first;
  });
}

@override
void onError(Error error) {
  print("[MyApp]error: $error");
}
}

```

设置 log 回调：

DART

```

class MyLoggerDelegate implements LoggerDelegate {
  static String _getCurrentTime() {
    var now = DateTime.now();
    var formatter = DateFormat('yyyy-MM-dd HH:mm:ss.SSS');
    return formatter.format(now);
  }

  @override
  void logInfo(String message) {
    var currentTime = _getCurrentTime();
    print("[${currentTime}][INFO]$message");
  }

  @override
  void logError(String message) {
    var currentTime = _getCurrentTime();
    print("[${currentTime}][ERROR]$message");
  }

  @override
  void logWarn(String message) {
    var currentTime = _getCurrentTime();
    print("[${currentTime}][WARN]$message");
  }
}

Logger.setDelegate(MyLoggerDelegate());

```

iOS 版本

[请向商务获取最新版本 SDK、appid、secretkey](#)；

以下示例基于 XheyCameraSDK-v1.2.12.6c73e6eb.iOS.zip。

将 XheyCameraSDK.xcframework 和 XheyCameraSDKResource.bundle 拖入工程 (勾选 Copy items if needed)。

iOS info.plist 里声明权限：

XML

```

<key>NSLocationAlwaysAndWhenInUseUsageDescription</key>
<string>我们需要精确的位置信息来提供准确的服务</string>
<key>NSLocationAccuracy</key>
<string>我们需要精确的位置信息来提供准确的服务</string>
<key>NSLocationWhenInUseUsageDescription</key>
<string>我们需要在后台获取您的位置来提供服务</string>
<key>NSCameraUsageDescription</key>
<string>我们需要使用您的相机来拍摄照片</string>

```

打开拍照页进行拍照：

OBJECTIVE-C

```

- (IBAction)takePhotoButtonClicked:(UIButton *)sender {
    XHCameraViewConfig *config = [[XHCameraViewConfig alloc] init];
    config.appid = @"your app id";
    config.secretKey = @"your secret key";
    // 如果 needPhotoConfirm = YES，拍完会回调 willDismiss，业务处理完后需手动调用 dismiss
    config.needPhotoConfirm = NO;
    config.maxImageCount = 9;
    // 资源 bundle 路径
    config.bundlePath = [[NSBundle mainBundle] pathForResource:@"XheyCameraSDKResource" ofType:@"bundle"];
    // 可选：指定团队水印
    // config.groupWatermarkId = @"your group watermark id";

    self.cameraViewController = [[XHCameraViewController alloc] initWithConfig:config
                                delegate:self];
    self.cameraViewController.modalPresentationStyle = UIModalPresentationFullScreen;
    [self presentViewController:self.cameraViewController animated:YES completion:nil];
}

```

设置拍照结果回调（didCaptureStillImages 返回 NSArray<NSData>，内容是 JPEG 二进制）：

OBJECTIVE-C

```

- (void)cameraViewController:(XHCameraViewController *)cameraViewController
    didCaptureStillImages:(NSArray<NSData *>)images {
    NSLog(@"%s %d", __FUNCTION__, __LINE__);
    for (NSInteger i = 0; i < self.imageViews.count; ++i) {
        if (i >= images.count) {
            self.imageViews[i].image = nil;
            continue;
        }
        self.imageViews[i].image = [UIImage imageWithData:images[i]];
    }
}

- (void)cameraViewControllerDidCancel:(XHCameraViewController *)cameraViewController {
    NSLog(@"%s %d", __FUNCTION__, __LINE__);
    self.cameraViewController = nil;
}

- (void)cameraViewController:(XHCameraViewController *)cameraViewController
    didFailWithError:(NSError *)error {
    NSLog(@"%s %d error: %@", __FUNCTION__, __LINE__, error);
    self.cameraViewController = nil;
}

- (void)cameraViewControllerWillDismiss:(XHCameraViewController *)cameraViewController {
    NSLog(@"%s %d", __FUNCTION__, __LINE__);
    [self.cameraViewController dismiss];
}

```

设置 log 回调：

OBJECTIVE-C

```

@interface MyCameraLogger : NSObject <XHCameraLoggerProtocol>
@end

@implementation MyCameraLogger

- (NSString *)currentTimestamp {
    NSDateFormatter *formatter = [[NSDateFormatter alloc] init];
    [formatter setDateFormat:@"yyyy-MM-dd HH:mm:ss.SSS"];
    return [formatter stringFromDate:[NSDate date]];
}

- (NSString *)currentThreadID {
    uint64_t tid;
    pthread_threadid_np(NULL, &tid);
    return [NSString stringWithFormat:@"%llu", tid];
}

- (void)logInfo:(NSString *)info {
    NSLog(@"[%@][%@][INFO] %@", [self currentTimestamp], [self currentThreadID], info);
}

- (void)logError:(NSString *)error {
    NSLog(@"[%@][%@][ERROR] %@", [self currentTimestamp], [self currentThreadID], error);
}

- (void)logWarn:(NSString *)warn {
    NSLog(@"[%@][%@][WARN] %@", [self currentTimestamp], [self currentThreadID], warn);
}

```

```

}

@end

[XHCameraLogger registerLogger:[MyCameraLogger new]];

```

Android 版本

请向商务获取最新版本 SDK、appid、secretkey；

以下示例基于 XheyCameraSDK-v1.2.12.6c73e6eb.Android.zip。

将 xheycamerasdk-release.aar 放到 app/libs，将 XheyCameraSDKAssets 目录拷贝到 app/src/main/assets/。

Gradle 依赖示例：

```

GROOVY
android {
    // ...
}

repositories {
    flatDir {
        dirs 'libs'
    }
}

dependencies {
    implementation(name: 'xheycamerasdk-release', ext: 'aar')

    def camerax_version = "1.3.4"
    implementation "androidx.camera:camera-core:${camerax_version}"
    implementation "androidx.camera:camera-camera2:${camerax_version}"
    implementation "androidx.camera:camera-lifecycle:${camerax_version}"
    implementation "androidx.camera:camera-view:${camerax_version}"
}

```

AndroidManifest.xml 申请权限：

```

XML
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-feature android:name="android.hardware.camera" android:required="true" />

```

拍照：

```

JAVA
private void takePhoto() {
    Intent intent = new Intent(this, CameraActivity.class);
    intent.putExtra(CameraViewConfig.kAppid, "your app id");
    intent.putExtra(CameraViewConfig.kSecretKey, "your secret key");
    intent.putExtra(CameraViewConfig.kMaxImageCount, 9);
    intent.putExtra(CameraViewConfig.kNeedPhotoConfirm, true);
    // 资源路径 ( 对应 app/src/main/assets/XheyCameraSDKAssets )
    String resourceDir = "file:///android_asset/XheyCameraSDKAssets";
    intent.putExtra(CameraViewConfig.kResourceDir, resourceDir);
    // 可选：指定团队水印
    // intent.putExtra(CameraViewConfig.kGroupWatermarkId, "your group watermark id");

    EventBus.registerEvent(EventBus.EventName.kCameraActivityWillDismiss, eventBusDelegate);
    someActivityResultLauncher.launch(intent);
}

```

如果 kNeedPhotoConfirm 为 true，需要注册 kCameraActivityWillDismiss 事件并在完成业务逻辑后手动关闭拍照页：

```

JAVA
private final EventBus.EventBusDelegate eventBusDelegate = new EventBus.EventBusDelegate() {
    @Override
    public void onEvent(String event, Object data) {
        Log.d("MainActivity", "onEvent: " + event);
        EventBus.unregisterEvent(EventBus.EventName.kCameraActivityWillDismiss, eventBusDelegate);
        EventBus.EventData.CameraActivityWillDismiss eventData =
            (EventBus.EventData.CameraActivityWillDismiss) data;
    }
}

```

```

        CameraActivity cameraActivity = eventData.cameraActivity;
        CameraActivity.Result result = eventData.result;
        // 业务处理完后手动关闭
        cameraActivity.dismiss();
    }
};

```

获取拍照结果 (CameraActivity.Result.images 类型为 List<byte[]>) :

JAVA

```

ActivityResultLauncher<Intent> someActivityResultLauncher = registerForActivityResult(
    new ActivityResultContracts.StartActivityForResult(),
    result -> {
        Log.d("MainActivity", "onActivityResult: " + result.getResultCode());
        if (result.getResultCode() == Activity.RESULT_OK) {
            CameraActivity.Result myResult = CameraActivity.getResult();
            if (myResult.error != null) {
                Log.e("MainActivity", "capture failed", myResult.error);
                return;
            }
            for (byte[] imageData : myResult.images) {
                Bitmap bitmap = BitmapFactory.decodeByteArray(imageData, 0, imageData.length);
                // TODO: 使用 bitmap ( 展示、上传、保存 )
            }
            return;
        }
        if (result.getResultCode() == Activity.RESULT_CANCELED) {
            Log.d("MainActivity", "RESULT_CANCELED");
        }
    });

```

设置 log 回调 :

JAVA

```

Logger.setDelegate(new Logger.LoggerDelegate() {
    @Override
    public void logInfo(String tag, String message) {
        Log.i(tag, message);
    }

    @Override
    public void logError(String tag, String message) {
        Log.e(tag, message);
    }

    @Override
    public void logWarn(String tag, String message) {
        Log.w(tag, message);
    }
});

```

Uni-app 版本

请向商务获取最新版本 SDK、appid、secretkey ;

以下示例基于 XCCameraModule 2.zip (插件版本 1.0.0) 。

安装原生插件

1. 解压 XCCameraModule 2.zip , 将 XCCameraModule 目录放到项目 nativeplugins/ 目录下。
2. 在 HBuilderX 中重新生成自定义基座 (或云打包) , 确保原生插件被编入安装包。

权限配置

manifest.json 中确保包含相机和定位权限。

iOS plist 示例 :

JSON

```

{
    "NSCameraUsageDescription": "用于拍照",
    "NSLocationWhenInUseUsageDescription": "用于获取拍照地点",
    "NSLocationAlwaysAndWhenInUseUsageDescription": "用于获取拍照地点"
}

```

Android 权限示例：

XML

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

调用方式

JAVASCRIPT

```
const xcCameraModule = uni.requireNativePlugin("XCCameraModule");

xcCameraModule.takePhotoWithAppId(
  "your app id",
  "your secret key",
  9, // maxImageCount
  "", // groupWatermarkId, 可选
  true, // needPhotoConfirm, 建议传 true
  (res) => {
    // imageDatas 是 JSON 字符串；解析后每项为 base64 图片（通常带 data:image/jpg;base64, 前缀）
    console.log("capture success:", JSON.parse(res?.imageDatas || "[]"));

    // 失败场景可关注：
    // res.error / res.errorCode
  },
);
```

微信小程序版本

请向商务获取插件信息：appid、secretkey、最新插件版本号等；签名生成方法参考当前文档鉴权部分

安装插件

在 app.json 中声明插件：

JSON

```
{
  "plugins": {
    "XCameraWXMiniPlugin": {
      "version": "1.0.60",
      "provider": "wx2978946aed99cf6f"
    }
  }
}
```

权限配置

在 app.json 中配置必要权限：

JSON

```
{
  "permission": {
    "scope.camera": {
      "desc": "用于拍照功能"
    },
    "scope.userLocation": {
      "desc": "用于获取位置信息"
    }
  }
}
```

基础库要求

- 微信小程序基础库 2.19.4+
- 支持插件开发模式

Page基本调用

JAVASCRIPT

```
// 打开相机页面
wx.navigateTo({
  url: "plugin://XCameraWXMiniPlugin/camera-page",
});
```

```

events: {
  onPluginResult: (data) => {
    console.log("拍照结果:", data.result);
    console.log("照片信息:", data.userCommentObject);
  },
},
});

```

完整示例

JAVASCRIPT

```

Page({
  data: {
    photoResult: "",
  },

  // 打开相机
  openCamera() {
    wx.navigateTo({
      url: "plugin://XCameraWXMiniPlugin/camera-page",
      events: {
        onPluginResult: this.handlePhotoResult.bind(this),
      },
    });
  },

  // 处理拍照结果
  handlePhotoResult(data) {
    if (data.result) {
      this.setData({
        photoResult: data.result,
        userComment: data.userCommentObject,
      });
      wx.showToast({
        title: "拍照成功！",
        icon: "success",
      });
    } else {
      wx.showToast({
        title: "拍照取消",
        icon: "none",
      });
    }
  },
});

```

带参数调用

JAVASCRIPT

```

const params = {
  appid: "",
  noncestr: "",
  timestamp: 1752804199,
  signature: encodeURIComponent(""), //signature可能存在特殊符号，需要提前处理encodeURIComponent
  groupWatermarkId: "", // 使用平台配置的水印
  disableOfficialWatermark: true, // 禁用官方水印
  disableWatermarkDragging: true, // 禁用水印拖拽
  maxImageCount: 3, // 最多拍摄3张
  customInputItems: encodeURIComponent(
    JSON.stringify([
      { title: "会议名称", content: "自定义会议名称" },
      { title: "会议类型", content: "自定义会议类型" },
      { title: "负责员工", content: "ABC" },
    ])
  ),
};

const query = Object.entries(params)
  .map(([key, value]) => `${key}=${value}`)
  .join("&");

wx.navigateTo({
  url: `plugin://XCameraWXMiniPlugin/camera-page?${query}`,
  events: {
    onPluginResult: this.handlePhotoResult.bind(this),
  },
});

```

Component基本调用

带参数调用

XML

```
<camera-component
  appid="{{appid}}"
  noncestr="{{noncestr}}"
  timestamp="{{timestamp}}"
  signature="{{signature}}"
  takingPhoto="{{takingPhoto}}"
  disable-photo-confirm="{{false}}"
  enable-custom-watermark="{{false}}"
  watermark-title="会议记录"
  logo-url=""
  flashMode="on"
  cameraPosition="front"
  watermark-scale="1.0"
  disable-official-watermark="{{false}}"
  disable-watermark-dragging="{{false}}"
  custom-input-items="{{customInputItems}}"
  group-watermark-id="{{groupWatermarkId}}"
  bind:result="onXcameraResult"
/>
```

新增交互控制属性：

属性名	类型	默认值	取值/格式	行为说明
takingPhoto	Boolean	false	true/false	设为 true 将触发一次拍照流程；组件内部完成后会自动复位为 false（由宿主决定是否复位，建议置回 false 以便下次触发）
cameraPosition	String	'back'	'back' 或 'front'	控制前后置摄像头；动态变更立即生效
flashMode	String	'off'	'off'/'on'/'auto'	控制闪光灯模式；动态变更立即生效

其他参数说明

认证参数（必填）

参数名	类型	默认值	说明
appid	String	-	应用ID
noncestr	String	-	随机字符串
timestamp	Number	-	时间戳
signature	String	-	签名

水印配置参数

参数名	类型	默认值	说明
groupWatermarkId	String	-	在今日水印相机平台上配置的水印模板ID
disableOfficialWatermark	Boolean	false	是否禁用官方水印
disableWatermarkDragging	Boolean	false	是否禁用水印拖拽
enableCustomWatermark	Boolean	false	是否支持用户自选水印

拍照功能参数

参数名	类型	默认值	说明
maxImageCount	Number	1	最大拍摄数量

自定义功能参数

参数名	类型	默认值	说明
customInputItems	String	-	自定义输入项（JSON字符串，若没有设置groupWatermarkId，则仅自定义水印模版支持展示，内容会直接作为条目。若设置了groupWatermarkId，则customInputItems的内容会根据title内容替换到水印里）

回调机制

通过页面引入的相机模块通过事件通道返回拍照结果，支持实时回调：

```
JAVASCRIPT
wx.navigateTo({
  url: "plugin://XCameraWXMiniPlugin/camera-page",
  events: {
    onPluginResult: (data) => {
      console.log("拍照结果:", data.result);
      console.log("照片信息:", data.userCommentObject);
      // data.result 是 Base64 格式的图片数据
    },
  },
});
```

通过组件引入的相机模块通过组件绑定事件返回拍照结果：

```
XML
<camera-component
  bind:result="onXCameraResult"
/>

JS
Page({
  onXCameraResult(e) {
    const { result, userCommentObject } = e.detail || {};
  },
});
```

成功回调

```
JAVASCRIPT
{
  result: "data:image/jpeg;base64,/9j/4AAQSkZJRgABAQAAQ... // Base64图片数据
  userCommentObject: {
    "data": {
      "baseInfo": {
        "altitude": "102.3",
        "decibel": "",
        "captureTimeClock": 0,
        "frontRearCam": "rear",
        "imageDirection": "",
        "isSelfLocation": false,
        "latitude": "39.6937632",
        "location": "北京市通州区·老李草莓采摘园",
        "locationDetail": "",
        "locationPoi": "",
        "locationType": "4",
        "longitude": "116.7036319",
        "photoNumber": "",
        "screenType": "2",
        "speed": "0",
        "time": "2025:09:02 11:00:24",
        "timeType": "1",
        "timeZone": "-480",
        "ua": {
          "appName": "XCamera-web",

```

```

      "appVersion": "1.0.1",
      "manufacturer": "",
      "model": "",
      "os": "",
      "osVersion": ""
    },
    "weather": {
      "type": "阴",
      "temperature": 30
    }
  },
  "fileName": "23f827f8-3376-4d51-b29c-dc58f14cf424.jpg",
  "groupLocation": {
    "groupID": "",
    "locationID": ""
  },
  "groupWatermarkID": "74542b94-ef29-4fff-a9f3-68c5f2823c71",
  "userID": "",
  "watermarkBaseID": 10,
  "watermarkContent": [

  ],
  "watermarkContentExtension": {
    "antiFakeCode": "Y62GPC2CEUR6X4"
  },
  "watermarkFromGroupID": "",
  "watermarkFromGroupID": "",

  "watermarkID": "10"
},
"ver": "20220209"
}
}

```

失败回调：

JAVASCRIPT

```

{
  result: null; // 拍照失败或用户取消
}

```

功能说明

maxImageCount 功能：

- 当达到最大数量时，会自动显示提示并返回上一页
- 支持任意正整数设置（建议 1-10）
- 拍照失败时也会计入计数
- 每次拍照后都会发送回调，无论成功或失败

customInputItems 功能：

- 必须使用 encodeURIComponent(JSON.stringify()) 进行编码
- 每个项目必须包含 title 和 content 字段
- title 需要对应水印条目的标题
- 无数量限制，建议不超过10个项目

常见问题

Q1: 插件无法加载怎么办？A: 检查以下几点：

1. 确认 app.json 中插件配置正确
2. 插件使用申请是否已经通过(请联系商务获取支持)
3. 检查基础库版本是否支持（2.19.4+）
4. 确认插件提供者ID正确

5. 检查网络连接是否正常

Q2: 无法获得地点/时间等信息？ A：请检查：

1. 鉴权信息是否正确？请参考下文“鉴权”
2. signature需要经过 encodeURIComponent() 处理，signature中可能存在特殊符号

鉴权

由于安全性的考虑，H5 版本需要业务方在自己的服务端搭建鉴权服务，Flutter、iOS、Android 不需要额外的鉴权服务。

鉴权流程：

/img/sdk/camera_sdk_img1.svg

鉴权算法

签名生成规则如下：

参与签名的字段包括 appid(申请的 appid)，noncestr (随机字符串)，timestamp (时间戳) 对所有待签名参数按照字段名的 ASCII 码从小到大排序 (字典序) 后，使用 URL 键值对的格式 (即 key1=value1&key2=value2...) 拼接成字符串 string1，这里需要注意的是所有参数名均为小写字符。对 string1 采用 HMAC-SHA256 签名，经过 Base64 编码 得到 signature

签名有效期：24h

示例：

加签数据

```
appid=123455&noncestr=U5YJHgUFrN&timestamp=1736919106
```

获取签名

```
// data=appid=123455&noncestr=U5YJHgUFrN×tamp=1736919106
```

```
// secret=1ecfaaa70d389eb87731e41036560282
```

```
// 得到 signature=StlLozltCtcp28CQZbQX1myPwwjk4626aRt3/83R1dQ=
```

```
// go代码示例 HMAC-SHA256 签名，输出格式 Base64
func HmacSign(data string, secret string) (ret string) {
    □ hmacObj := hmac.New(sha256.New, []byte(groupSecret))
    □ hmacObj.Write([]byte(data))
    □ ret = base64.StdEncoding.EncodeToString(hmacObj.Sum(nil))
    □ return ret
}
```

鉴权代码参考

JS 版本 1

```
generateSignature(params, secret) {
  // 1. 对参数按照 ASCII 字典序排序
  const sortedKeys = Object.keys(params).sort();

  // 2. 拼接成 URL 键值对的格式
  const data = sortedKeys.map((key) => `${key}=${params[key]}`).join("&");

  // 3. 使用 HMAC-SHA256 进行签名
  const encoder = new TextEncoder();
  const keyData = encoder.encode(secret);
  const messageData = encoder.encode(data);

  return crypto.subtle
    .importKey(
      "raw",
      keyData,
      { name: "HMAC", hash: { name: "SHA-256" } },
      false,
      ["sign"]
    )
    .then((key) => {
      return crypto.subtle.sign("HMAC", key, messageData);
    })
    .then((signature) => {
      // 4. 签名结果转换为 Base64
      return btoa(String.fromCharCode(...new Uint8Array(signature)));
    });
}

const appid = "your appid";
const noncestr = "65cdfefecaf3a0c4"; // 可以随机生成一个字符串
const timestamp = Math.floor(Date.now() / 1000).toString();
const params = {
  appid: appid,
  noncestr: noncestr,
  timestamp: timestamp,
};

const secret = "secret";
const signature = await this.generateSignature(params, secret);
```

JS 版本 2

```
<!-- 引入 CryptoJS 库 -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/crypto-js/4.1.1/crypto-js.min.js"></script>
<script>
function generateSignature(params, secret) {
  // 1. 对参数按照 ASCII 字典序排序
  const sortedKeys = Object.keys(params).sort();

  // 2. 拼接成 URL 键值对格式
  const data = sortedKeys.map((key) => `${key}=${params[key]}`).join('&');

  // 3. 使用 CryptoJS 计算 HMAC-SHA256 签名
  const hash = CryptoJS.HmacSHA256(data, secret);

  // 4. 将签名结果转换为 Base64
  const signature = CryptoJS.enc.Base64.stringify(hash);
  return signature;
}
</script>
```

Python 版本

```

import hmac
import hashlib
import base64
import argparse
import time
import random
import string

# 生成签名的函数
def generate_signature(params, secret):
    # 1. 对参数按照 ASCII 字典序排序
    sorted_keys = sorted(params.keys())

    # 2. 拼接成 URL 键值对的格式
    data = "&".join(f"{key}={params[key]}" for key in sorted_keys)

    # 3. 使用 HMAC-SHA256 进行签名
    secret_bytes = secret.encode('utf-8')
    data_bytes = data.encode('utf-8')

    # 创建 HMAC 对象并进行签名
    signature = hmac.new(secret_bytes, data_bytes, hashlib.sha256).digest()

    # 4. 签名结果转换为 Base64
    signature_base64 = base64.b64encode(signature).decode('utf-8')

    return signature_base64

# 生成随机字符串
def generate_random_string(length=16):
    return ''.join(random.choices(string.ascii_lowercase + string.digits, k=length))

# 主函数
def main():
    appid = "your app id"
    secret = "your secret key"

    # 生成随机字符串和时间戳
    noncestr = generate_random_string()
    timestamp = str(int(time.time())) # 获取当前时间戳 ( 秒 )

    # 构造参数字典
    params = {
        "appid": appid,
        "noncestr": noncestr,
        "timestamp": timestamp
    }

    # 生成签名
    signature = generate_signature(params, secret)

    # 输出参数和签名
    print(f"appid: {appid}")
    print(f"noncestr: {noncestr}")
    print(f"timestamp: {timestamp}")
    print(f"signature: {signature}")

if __name__ == "__main__":
    main()

```

Java 版本

```

package com.xhey.xheycamerasdk;

import android.util.Base64;

import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Random;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public class SignatureUtil {

    /**
     * 封装签名结果的实体类
     */
    public static class SignatureResult {
        public int timestamp; // 修改为 int 类型
        public String noncestr;
        public String signature;
    }

    /**
     * 根据 appid 和 secretKey 生成 timestamp、noncestr 和 signature
     *
     * @param appid 应用 ID
     * @param secretKey 密钥
     * @return 签名结果，包括 timestamp、noncestr 和 signature
     */
    public static SignatureResult generateSignature(String appid, String secretKey) {
        SignatureResult result = new SignatureResult();

        // 获取当前 Unix 时间戳 ( 秒 ) 并转成 int 类型
        result.timestamp = (int) (System.currentTimeMillis() / 1000);

        // 生成随机字符串 ( noncestr )
        result.noncestr = generateRandomString(16);

        // 构造参数字典 ( 键值对 )，注意需要按照 ASCII 字典序排序
        // 由于参数只有三个，可先放入一个 List 中，然后排序
        List<String> keys = new ArrayList<>();
        keys.add("appid");
        keys.add("noncestr");
        keys.add("timestamp");
        Collections.sort(keys); // 按 ASCII 顺序排序

        // 根据排序后的 key 拼接成 URL 键值对格式
        // 例如：appid=xxx&noncestr=xxx&timestamp=xxx
        StringBuilder dataBuilder = new StringBuilder();
        for (int i = 0; i < keys.size(); i++) {

            String key = keys.get(i);
            String value;
            switch (key) {
                case "appid":
                    value = appid;
                    break;
                case "noncestr":
                    value = result.noncestr;
                    break;
                case "timestamp":
                    value = String.valueOf(result.timestamp);
                    break;
                default:
                    value = "";
            }
            dataBuilder.append(key).append("=").append(value);
            if (i != keys.size() - 1) {
                dataBuilder.append("&");
            }
        }
        String data = dataBuilder.toString();

        try {
            // 使用 HMAC-SHA256 算法计算签名

```

```

SecretKeySpec signingKey = new SecretKeySpec(secretKey.getBytes(StandardCharsets.UTF_8), "HmacSHA256");
Mac mac = Mac.getInstance("HmacSHA256");
mac.init(signingKey);
byte[] rawHmac = mac.doFinal(data.getBytes(StandardCharsets.UTF_8));

// 将签名结果进行 Base64 编码
result.signature = Base64.encodeToString(rawHmac, Base64.NO_WRAP);
} catch (Exception e) {
    e.printStackTrace();
    result.signature = null;
}

return result;
}

/**
 * 生成指定长度的随机字符串 (只包含小写字母和数字)
 *
 * @param length 随机字符串长度
 * @return 随机字符串
 */
private static String generateRandomString(int length) {
    String chars = "abcdefghijklmnopqrstuvwxyz0123456789";
    StringBuilder sb = new StringBuilder(length);
    Random random = new Random();
    for (int i = 0; i < length; i++) {
        sb.append(chars.charAt(random.nextInt(chars.length())));
    }

    return sb.toString();
}
}

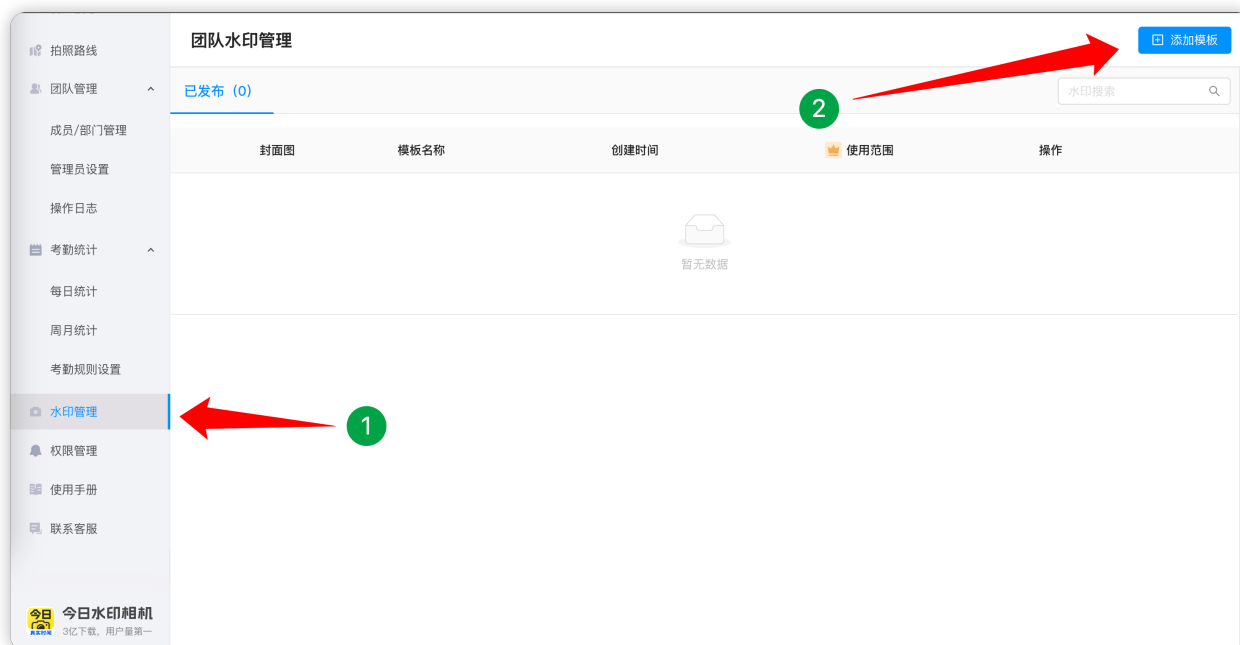
```

水印模版

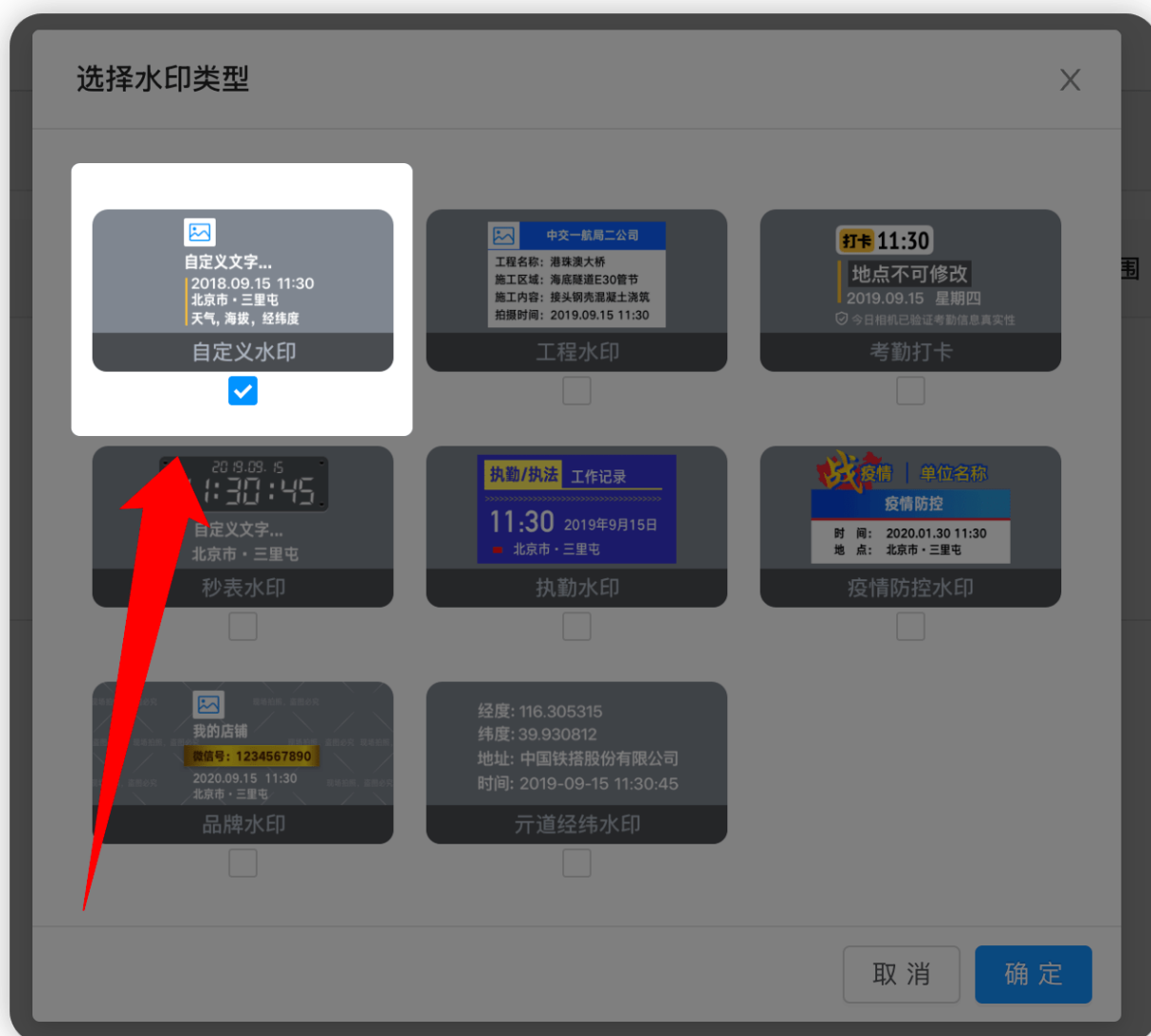
水印后台配置水印流程

进入今日水印相机 web 端：<https://www.5181688.com/>

左侧 Tab 选择水印管理后，点击添加模版



目前只支持第一款水印(如有其他水印选择需求，向商务同学沟通进行支持)，选择，点击确定：



title

上面 H5 的 sdk 实例创建中, 如果 title 字段不为空的话, 会覆盖自定义文字中的内容

主题风格

品牌图片

自定义文字

编号

时间

天气

地点

海拔

方位角

速度

经纬度

拍摄人

自定义项

请输入水印模板名称...

> 主题风格

< 自定义文字

内容类型：☒ 文本 ☐ 选项 ☐ 多级选项

水印里的信息需拍照人手动填写

☐ 内容要求员工必填 限免

确保员工填写正确，拍照必须填写内容，定期自动清空已输入内容

☐ 每5分钟清空 ☐ 始终不清空

内容：

这是配置的Title的内容

☐ 员工不可修改内容

提示文字：

选填，可为员工输入范围或注意事项

logoUrl

logoUrl 内容不为空的话，会覆盖掉水印中设置的品牌图

主题风格

品牌图片

自定义文字

编号

时间

天气

地点

海拔

方位角

速度

经纬度

拍摄人

自定义项

< 品牌图片



图片大小： 23%

透明度： 0%

customInputItems

customInputItems 不为空的话，会覆盖掉水印中对应的自定义内容

主题风格 >

品牌图片 ☒

自定义文字 ☒

编号 ☐

时间 ☐

天气 ☐

地点 ☐

海拔 ☐

方位角 ☐

速度 ☐

经纬度 ☐

拍摄人 ☐

会议名称 ☒

会议类型 ☒

+ 自定义项

会议名称

标题:

会议名称

内容类型: ☒ 文本 ☐ 选项 ☐ 多级选项

水印里的信息需拍照人手动填写

☐ 内容要求员工必填 限免
 确保员工填写正确, 拍照必须填写内容, 定期自动清空已输入内容
☐ 每5分钟清空 ☐ 始终不清空

内容:

这是会议名称的内容

☐ 员工不可修改内容

提示文字:

选填, 可为员工输入范围或注意事项

会议类型

标题:

会议类型

内容类型: ☒ 文本 ☐ 选项 ☐ 多级选项

水印里的信息需拍照人手动填写

☐ 内容要求员工必填 限免
 确保员工填写正确, 拍照必须填写内容, 定期自动清空已输入内容
☐ 每5分钟清空 ☐ 始终不清空

内容:

这是会议类型的内容

☐ 员工不可修改内容

自定义项设置示例

假设在后台配置了以下自定义项的标题和内容：

- 自定义项标题 1，自定义项内容 1
- 自定义项标题 2，自定义项内容 2
- 自定义项标题 3，自定义项内容 3
- 自定义项标题 4，自定义项内容 4
- 自定义项标题 5，自定义项内容 5

在 customInputItems 中设置如下：

```
{
  "自定义项标题2": "这是自定义项内容2",
  "自定义项标题5": "这是自定义项内容5"
}
```

```
}

```

最终，在 H5 页面中的水印显示效果如下：

- 自定义项标题 1，自定义项内容 1
- 自定义项标题 2，这是自定义项内容 2
- 自定义项标题 3，自定义项内容 3
- 自定义项标题 4，自定义项内容 4
- 自定义项标题 5，这是自定义项内容 5

QA

浏览器中相机预览黑屏

确保是在 https 环境中打开相机，否则打开相机报错；

时间、地点、天气显示获取异常

保证鉴权正确，鉴权码生成的 appid、noncestr、时间戳跟传给 XHeyCamera 的值要一致；

如果手机时间落后服务端时间一天或者手机时间超过服务端时间 10 分钟，会鉴权失败；

小米手机预览卡顿

点击拍照页右下角切换分辨率按钮，由 2K 切换到 4K

目前这个分辨率按钮影响的是预览分辨率，不影响拍照，是个为适配特殊机型做的折中方案。纯 H5 中无法获取机型信息，需要客户传 deviceModel&deviceBrand，传入正确的值后 SDK 根据机型进行相机适配处理；

OPPO Reno5 Pro 4K 下会预览白屏

点击拍照页右下角切换分辨率按钮，由 4K 切换到 2K

Nova 13 分辨率切换到 4K 后再切换到前置，无法切换

点击拍照页右下角切换分辨率按钮，由 4K 切换到 2K

荣耀 10 青春版、荣耀 9 青春版后置无法打开相机

需要传入正确的 deviceModel&deviceBrand，SDK 根据机型进行相机适配；

iPhone12 某个手机在企业微信浏览器环境中无法打开相机

打开 https://test-sdk-h5.xhey.top，点击拍照，然后点击右下角 vConsole，看一下 Log 以及 Error，看看 Log 中下面关键字中内容：

```
navigator.getUserMedia:
navigator.webkitGetUserMedia:
navigator.mozGetUserMedia:
navigator.mozGetUserMedia:
navigator.msGetUserMedia:
navigator.mediaDevices:
navigator.mediaDevices.getUserMedia:
```

如果全为 undefined，说明浏览器环境异常，无法获取打开浏览器 api；

微信小程序使用 H5 打开调用拍照无响应

需要把 https://sdk-h5.xhey.top 加到微信小程序的安全域名中，下载对应的证书给我们放到我们根目录下。

Android SDK 无法收到 EventBus 发送的 Event

保证 CameraActivity 跟调用方所在的 Activity 处于同一个进程中，处在不同的进程中会无法收到回调事件；

H5 拍照 UI 变得特别小

外面业务方设置了 initial-scale 小于 1 导致的

```
<meta name="viewport" content="width=device-width, initial-scale=0.5" />
```