

Tencent Effect SDK

API Documentation

Product Documentation



Copyright Notice

©2013-2024 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

API Documentation

iOS

Android

Flutter

Web

API Documentation

iOS

Last updated : 2024-03-19 15:50:12

XMagic.h, the core interface class of the Tencent Effect SDK, is used to initialize the SDK, update the beautification value, call the animation effect and other functions.

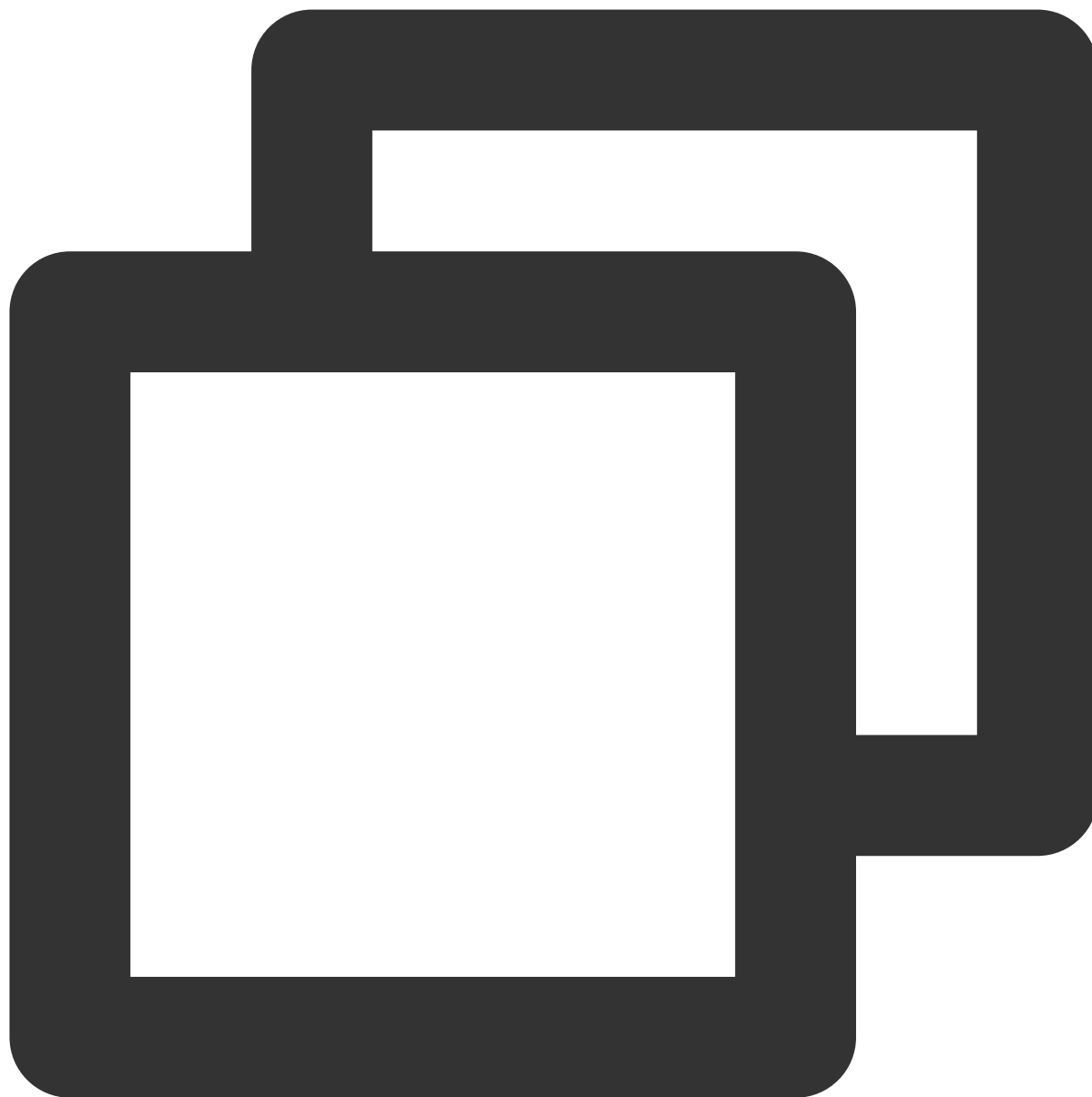
Public Member APIs

API	Description
initWithRenderSize	Initialization API
initWithGITexture	Initialization API
configPropertyWithType	Configures effects.
setEffect	Configuration of various beauty filter effects (Added in 3.5.0.2)
emitBlurStrengthEvent	Setting post-processing blur strength (Applies to all blur components)
setRenderSize	Sets the render size.
deinit	Releases resources.
process:	Image data processing interface: inputs image before beautification, returns image after beautification.
process:withOrigin:withOrientation:	Image data processing interface: inputs image before beautification, returns image after beautification. This interface has two additional parameters compared to the previous one.
process	Processes data.
processUIImage	Processes an image.
getConfigPropertyWithName	Gets effect information.
registerLoggerListener	Registers a log listener.
registerSDKEventListener	Register a listener for SDK events.

clearListeners	Removes listeners.
getCurrentGLContext	Gets the current OpenGL context.
onPause	Pauses the SDK.
onResume	Resumes the SDK.
setAudioMute	Whether to turn on mute when the dynamic effect material is used (new in V2.5.0)Parameters: YES means mute, NO means no mute
enableEnhancedMode	<p>Enable enhanced beauty filter pattern (Added in V2.5.1). By default, it is not activated.</p> <p>When it's disabled, the application layer can set the intensity range of each beauty option from 0 to 1 or -1 to 1. If it exceeds this range, the SDK will take the boundary value. For example, if the application layer sets the face-slimming component to 1.2, SDK will limit it to the maximum value of 1.0, then it corrects the face-slimming value to 1.0 internally, and then it corrects the face-slimming value to 1.0 internally.</p> <p>After enabling the enhanced pattern, the application layer can set a larger range of values. For example, if you want greater face slimming, you can set the face slimming value to 1.2. The SDK will accept and use this value and will not correct it to 1.0.</p> <p>Note:</p> <p>After enabling the enhanced pattern, the application layer needs to manage the maximum value that each beauty item can set, and let users adjust the value within this range. We provide a set of reference values, which you can adjust according to product requirements. However, we do not recommend exceeding our recommended values, otherwise the beauty effect may worsen.</p>
Overlaying of materials	If you want to overlay a certain animation/beauty/segmentation material on the current material, when setting up the material, set 'mergeWithCurrentMotion' to true in the dictionary of 'withExtraInfo'
High performance mode	During SDK initialization, add @ "setDowngradePerformance":@(YES) to the dictionary. With high performance mode enabled, the beauty feature will consume less system CPU/GPU resources, reducing overheating and lagging on the phone, making it more suitable for long-term use on low-end devices.

initWithRenderSize

This API is used to configure effects.



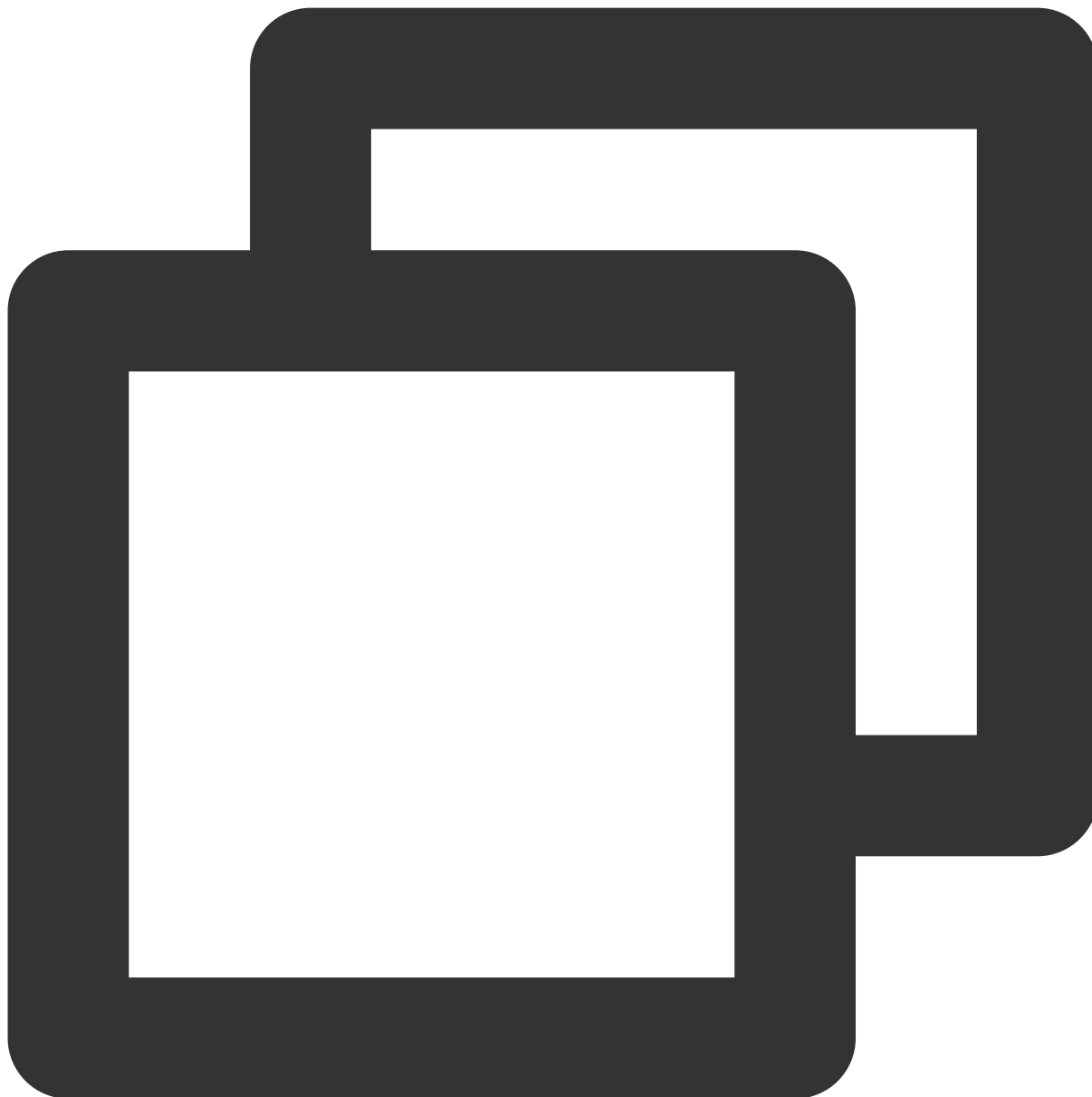
```
- (instancetype _Nonnull)initWithRenderSize:(CGSize)renderSize  
    assetsDict:(NSDictionary* _Nullable)assetsDict;
```

Parameters

Parameter	Description
renderSize	The render size.
assetsDict	The resource dictionary.

initWithGLTexture

This API is used to configure effects.



```
- (instancetype _Nonnull) initWithGLTexture: (unsigned) textureID
    width: (int) width
    height: (int) height
    flipY: (bool) flipY
    assetsDict: (NSDictionary* _Nullable) assetsDict;
```

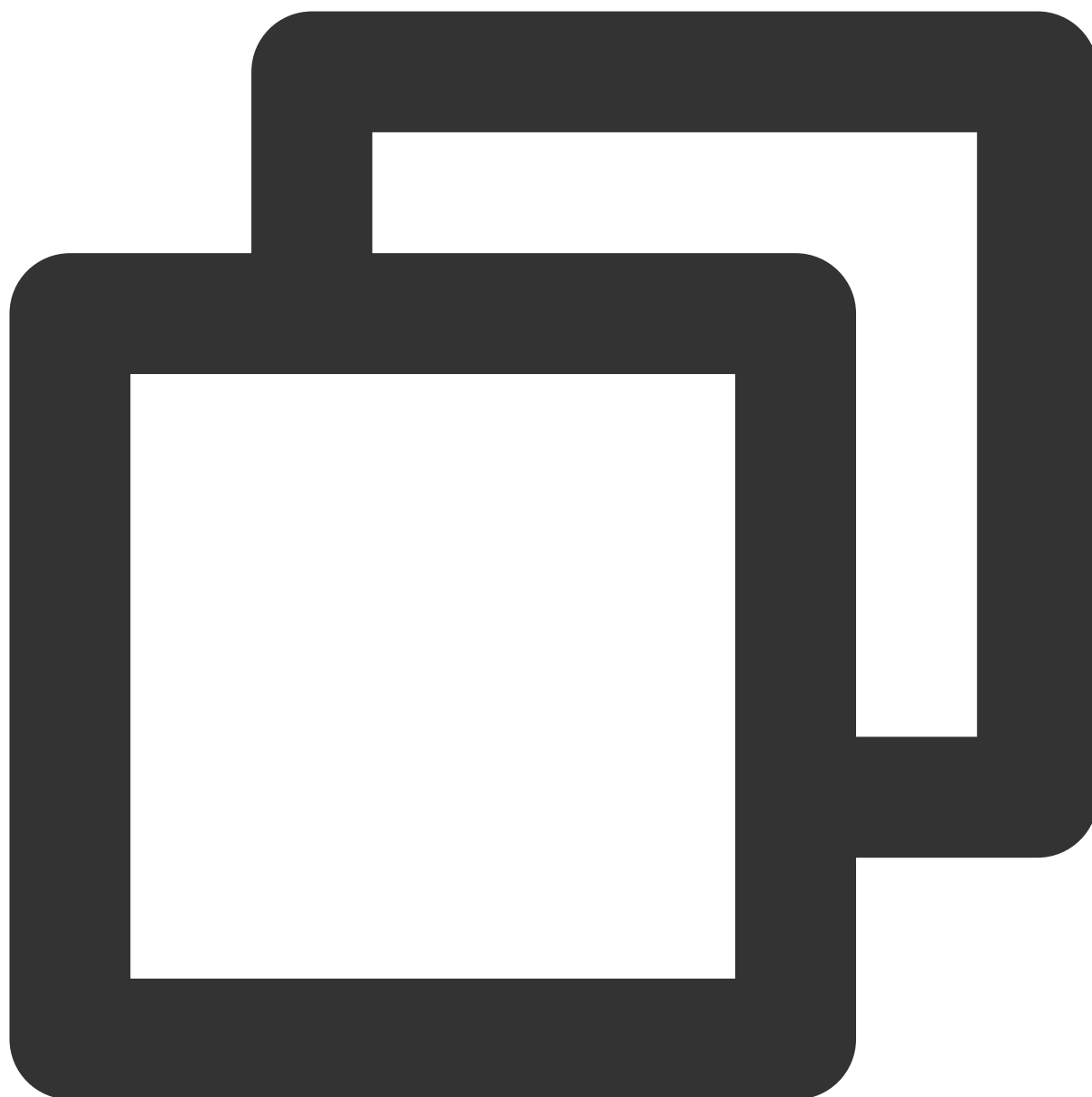
Parameters

--	--

Parameter	Description
textureID	The texture ID.
width	The render size.
height	The render size.
flipY	Whether to flip the image.
assetsDict	The resource dictionary.

configPropertyWithType

This API is used to configure effects.



```
- (int)configPropertyWithType:(NSString *_Nonnull)propertyType withName:(NSString *)
```

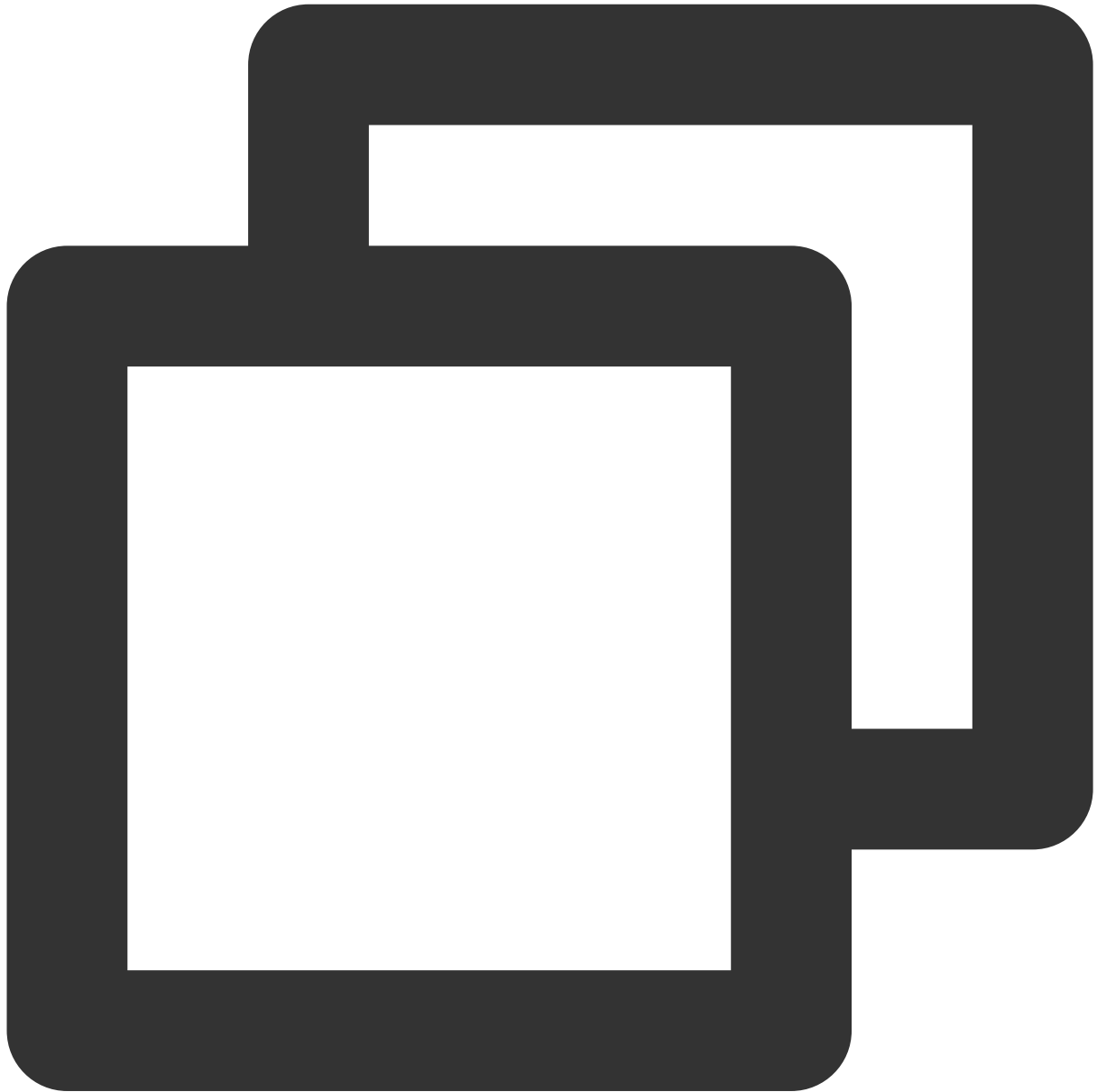
Parameters

Parameter	Description
propertyType	The effect type.
propertyName	The effect name.
propertyValue	The effect value.

extraInfo	A reserved parameter, which can be used for dictionary configuration.
-----------	---

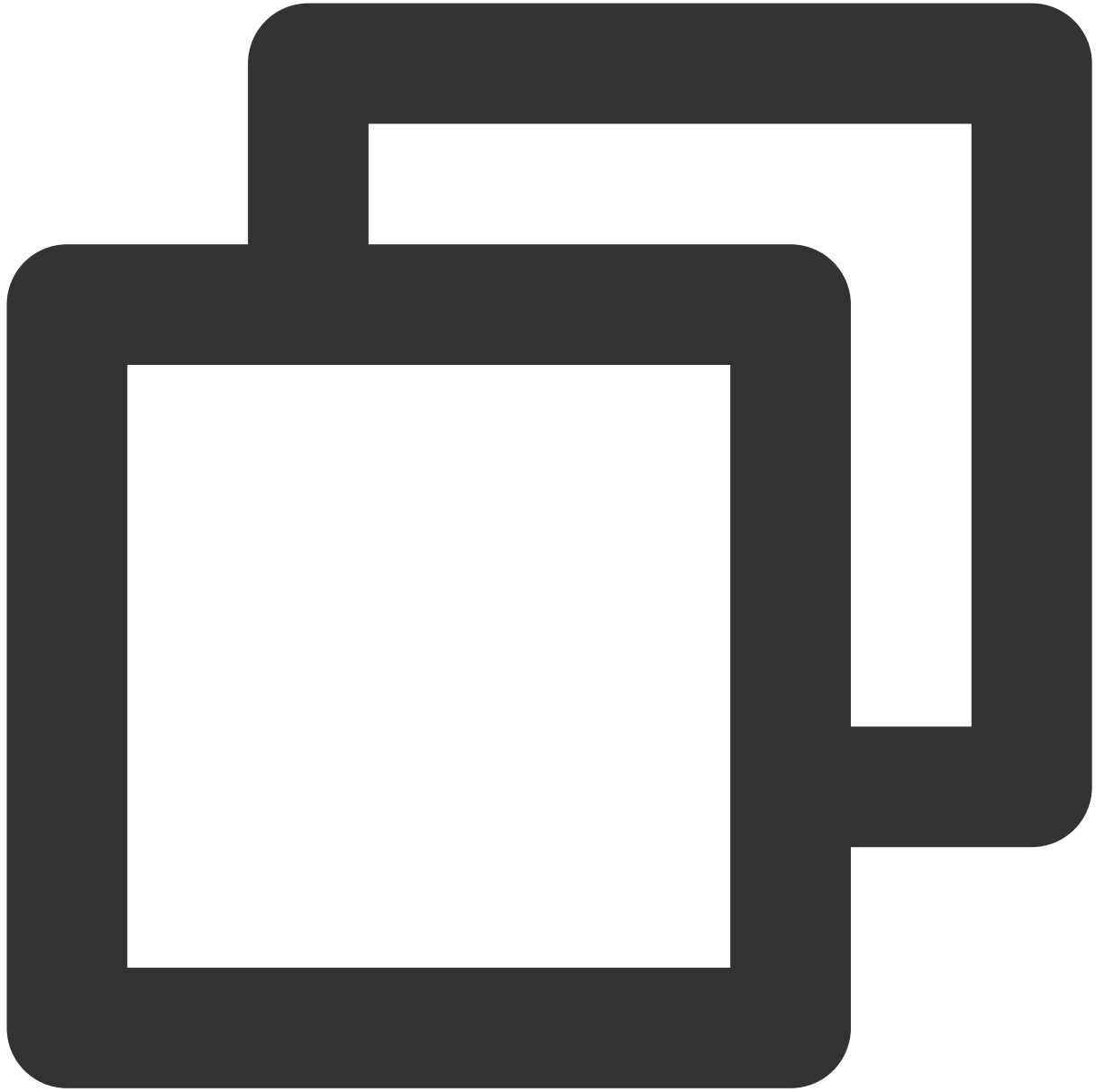
Examples

Beautification: Configuring the skin brightening effect



```
NSString *propertyType = @"beauty";           //Set the effect type, take beauty as an
NSString *propertyName = @"beauty.whiten";   //Specify the effect name, take the skin
NSString *propertyValue = @"60";              //Set the effect value.
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
traInfo:nil];
```

Filter: Configuring the Allure filter



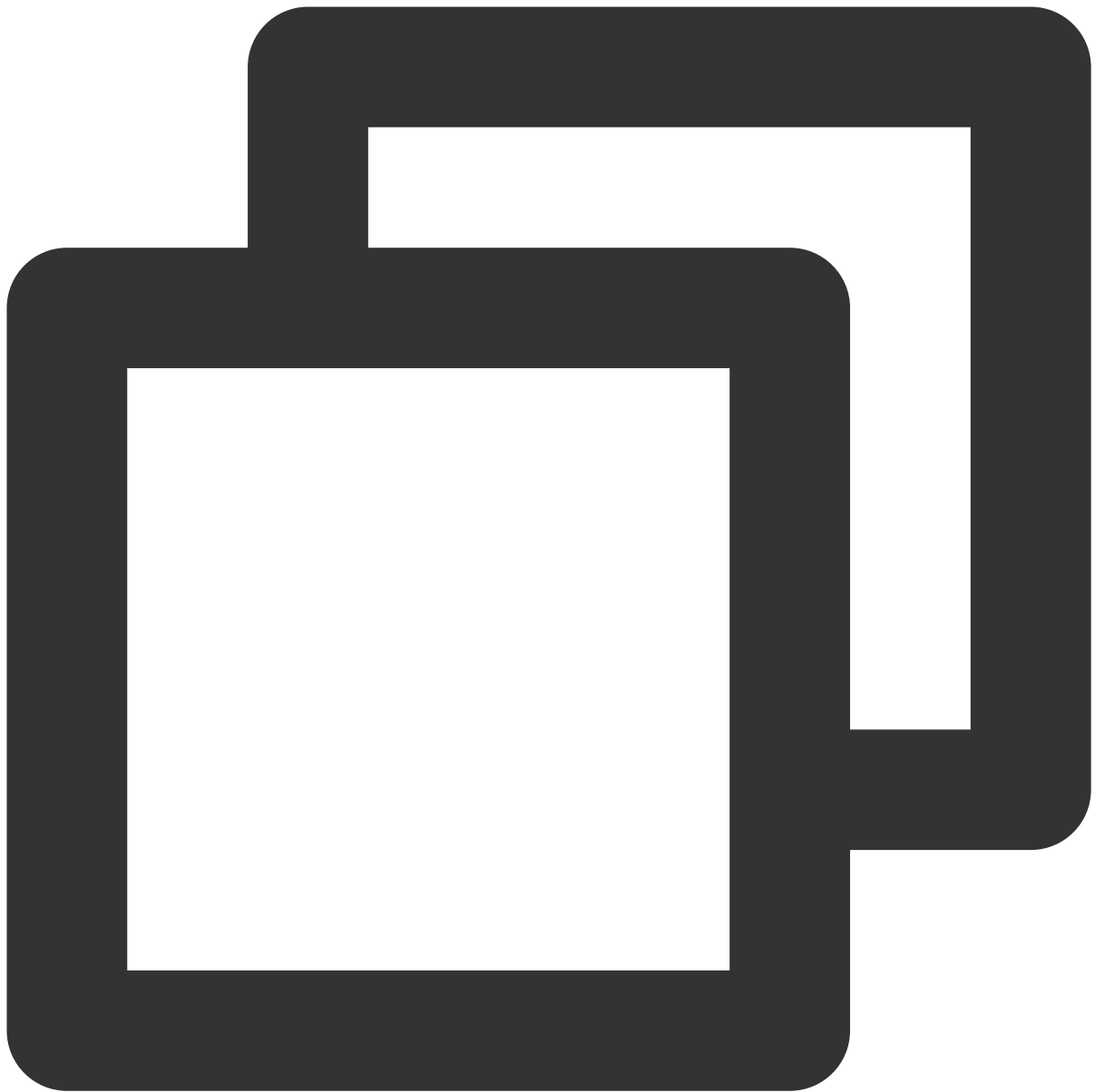
```
NSString *propertyType = @"lut";           //Set the effect type, take filter as an ex
NSString *propertyName = [@"lut.bundle/" stringByAppendingString:@"xindong_l
NSString *propertyValue = @"60";           //Set the effect value.
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
```

Body retouch: Configuring the long leg effect



```
NSString *propertyType = @"body";           //Set the effect type, take body retouch a
NSString *propertyName = @"body.legStretch"; //Specify the effect name, take the lo
NSString *propertyValue = @"60";             //Set the effect value.
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
```

Animated effect: Configuring the animated 2D cute effect



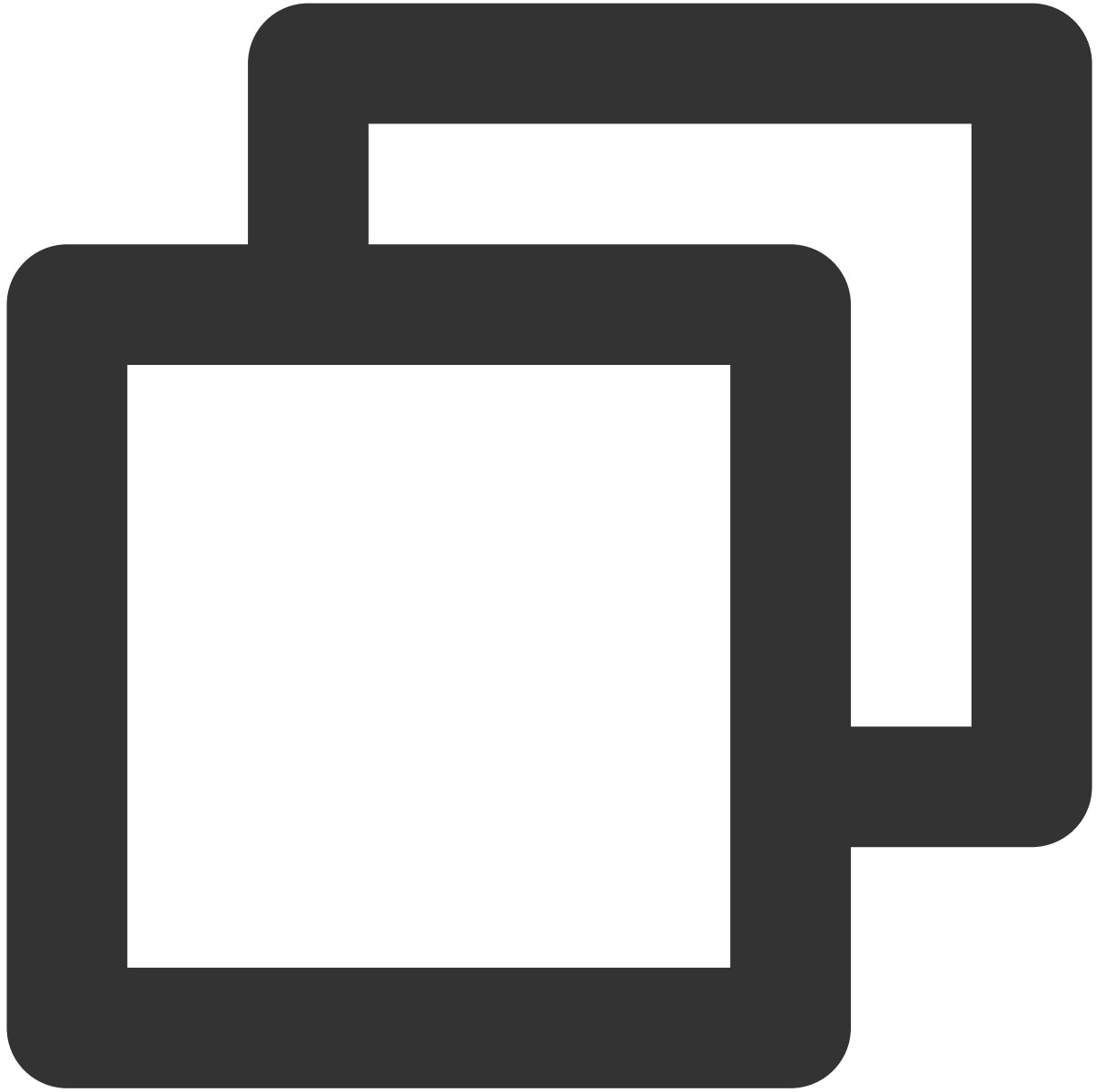
```
NSString *motion2dResPath = [[NSBundle mainBundle] pathForResource:@"2dMotionRes"  
NSString *propertyType = @"motion";           //Set the effect type, take animated eff  
NSString *propertyName = @"video_keaituya"; //Specify the effect name, take animate  
NSString *propertyValue = motion2dResPath;  //Set the path of the animated effect.  
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
```

Makeup: Configuring the girl group makeup effect



```
NSString *motionMakeupResPath = [[NSBundle mainBundle] pathForResource:@"makeupMoti
NSString *propertyType = @"motion";          //Set the effect type, take makeup as an
NSString *propertyName = @"video_nvtuanzhuang"; //Specify the effect name, take the
NSString *propertyValue = motionMakeupResPath; //Set the path of the animated effe
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
//Below are settings for the makeup effect (you only need to configure the above pa
NSString *propertyTypeMakeup = @"custom";      //Set the effect type, take makeu
NSString *propertyNameMakeup = @"makeup.strength"; //Specify the effect name, take
NSString *propertyValueMakeup = @"60";         //Set the effect value.
[self.xmagicApi configPropertyWithType:propertyTypeMakeup withName:propertyNameMake
```

Keying: Configuring the background blurring effect (strong)



```
NSString *motionSegResPath = [[NSBundle mainBundle] pathForResource:@"segmentMotion
NSString *propertyType = @"motion";           //Set the effect type, take keying as a
NSString *propertyName = @"video_segmentation_blur_75"; //Specify the effect name,
NSString *propertyValue = motionSegResPath; //Set the path of the animated effect.
NSDictionary *dic = @{@"bgName":@"BgSegmentation.bg.png", @"bgType":@0, @"timeOffse
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
```

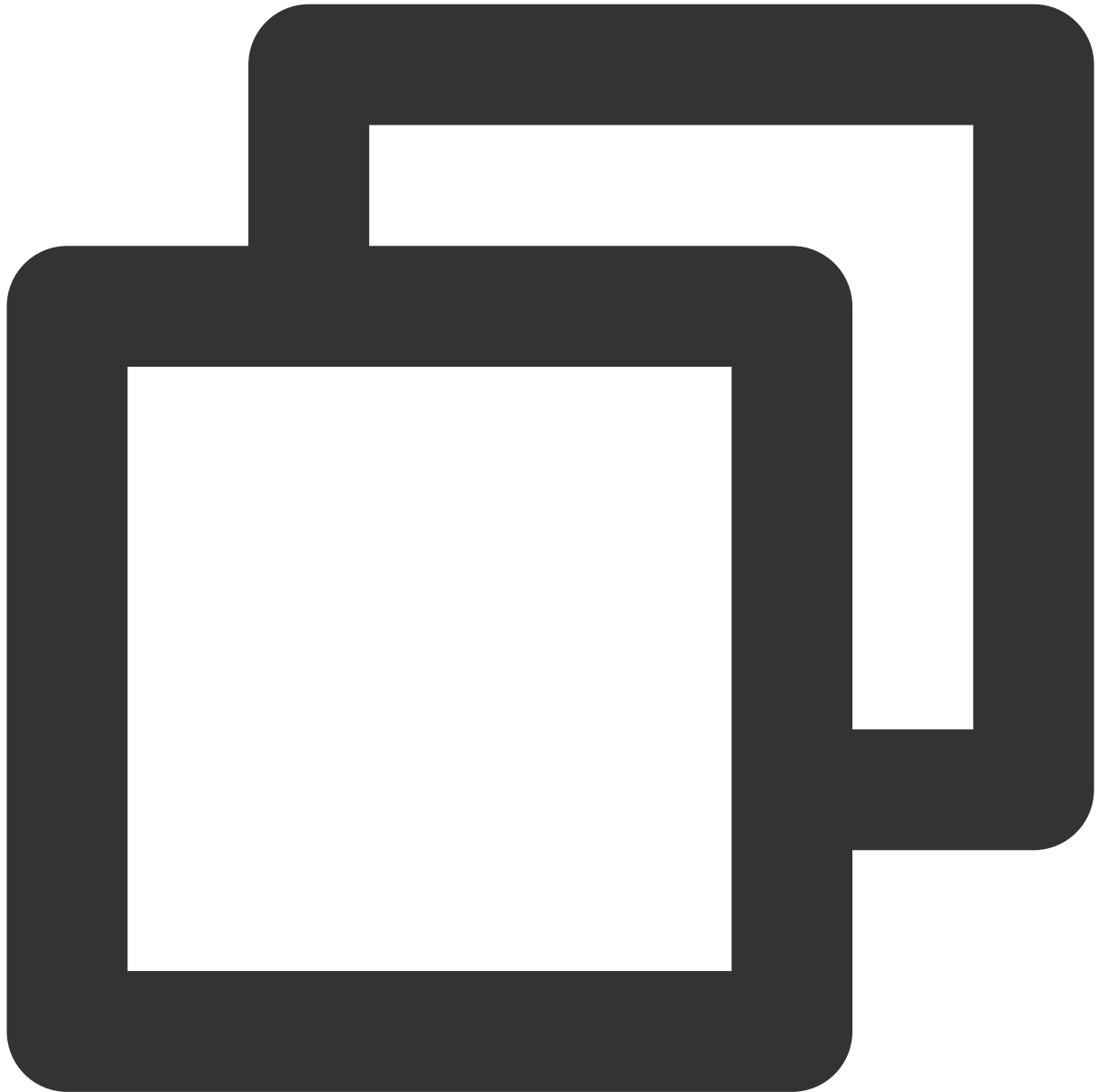
Custom background:



```
NSString *motionSegResPath = [[NSBundle mainBundle] pathForResource:@"segmentMotion
NSString *propertyType = @"motion";           //Set the effect type, take keying as a
NSString *propertyName = @"video_empty_segmentation"; //Specify the effect name, ta
NSString *propertyValue = motionSegResPath; //Set the path of the animated effect
NSString *imagePath = @"/var/mobile/Containers/Data/Application/06B00BBC-9060-450F-
int bgType = 0; //The background type. 0: image; 1: video.
int timeOffset = 0; //The duration. If an image is used as the background, its value
NSDictionary *dic = @{@"bgName":imagePath, @"bgType":@(bgType), @"timeOffset": @(ti
[self.xmagicApi configPropertyWithType:propertyType withName:propertyName withData:
```


setEffect (Added in 3.5.0.2)

For the configuration of various beauty effects, see [Effect Parameters](#) for specific usage examples.



```
- (void)setEffect:(NSString * _Nullable)effectName
    effectValue:(int)effectValue
resourcePath:(NSString * _Nullable)resourcePath
    extraInfo:(NSDictionary * _Nullable)extraInfo;
```

Parameter

--	--

Parameter	Meaning
effectName	Effect type.
effectValue	Effect value.
resourcePath	Material path.
<u>extraInfo</u>	Reserved for expansion and additional configuration.

emitBlurStrengthEvent

Setting post-processing blur intensity (which applies to all blur components).



```
- (void)emitBlurStrengthEvent:(int)strength;
```

Parameter

Parameter	Meaning
strength	Effect value.

setRenderSize

This API is used to set the render size.



```
- (void) setRenderSize: (CGSize) size;
```

Parameters

Parameter	Description
size	The render size.

deinit

This API is used to release resources.



```
- (void)deinit;
```

process

Processing data interface, the input data formats include

`YTImagePixelData`, `YTTextureData`, `YTImageRawData`, `YTUIImageData` , outputting the corresponding data formats. The pixel format in `YTImagePixelData` is `RGBA` , and the texture format in `YTTextureData` is `OpenGL 2D` .



```
/// @brief Process input 4 choose 1
@interface YTProcessInput : NSObject
/// Camera data object
@property (nonatomic, strong) YTImagePixelData * _Nullable pixelData;
/// Texture object
@property (nonatomic, strong) YTTextureData * _Nullable textureData;
/// Raw data object
@property (nonatomic, strong) YTImageRawData * _Nullable rawData;
/// UIImage object
@property (nonatomic, strong) YTUIImageData * _Nullable UIImageData;
/// Input data type
```

```
@property (nonatomic) enum YTProcessDataType dataType;
@end

/// @brief Process output
@interface YTProcessOutput : NSObject
/// Texture output object (always guaranteed)
@property (nonatomic, strong) YTTextureData * _Nullable textureData;
/// Camera output object (if the input is camera acquisition data)
@property (nonatomic, strong) YTImagePixelData * _Nullable pixelData;
/// Raw output object (if the input is raw data)
@property (nonatomic, strong) YTImageRawData * _Nullable rawData;
/// UIImage output object (if the input is a UIImage object)
@property (nonatomic, strong) YTUIImageData * _Nullable UIImageData;
/// Output data type
@property (nonatomic) enum YTProcessDataType dataType;
@end

- (YTProcessOutput* _Nonnull)process:(YTProcessInput * _Nonnull)input;
```

Parameter

Parameter	Meaning
input	Input data processing information, one of four input formats can be chosen from (YTImagePixelData, YTTextureData, YTImageRawData, YTUIImageData).

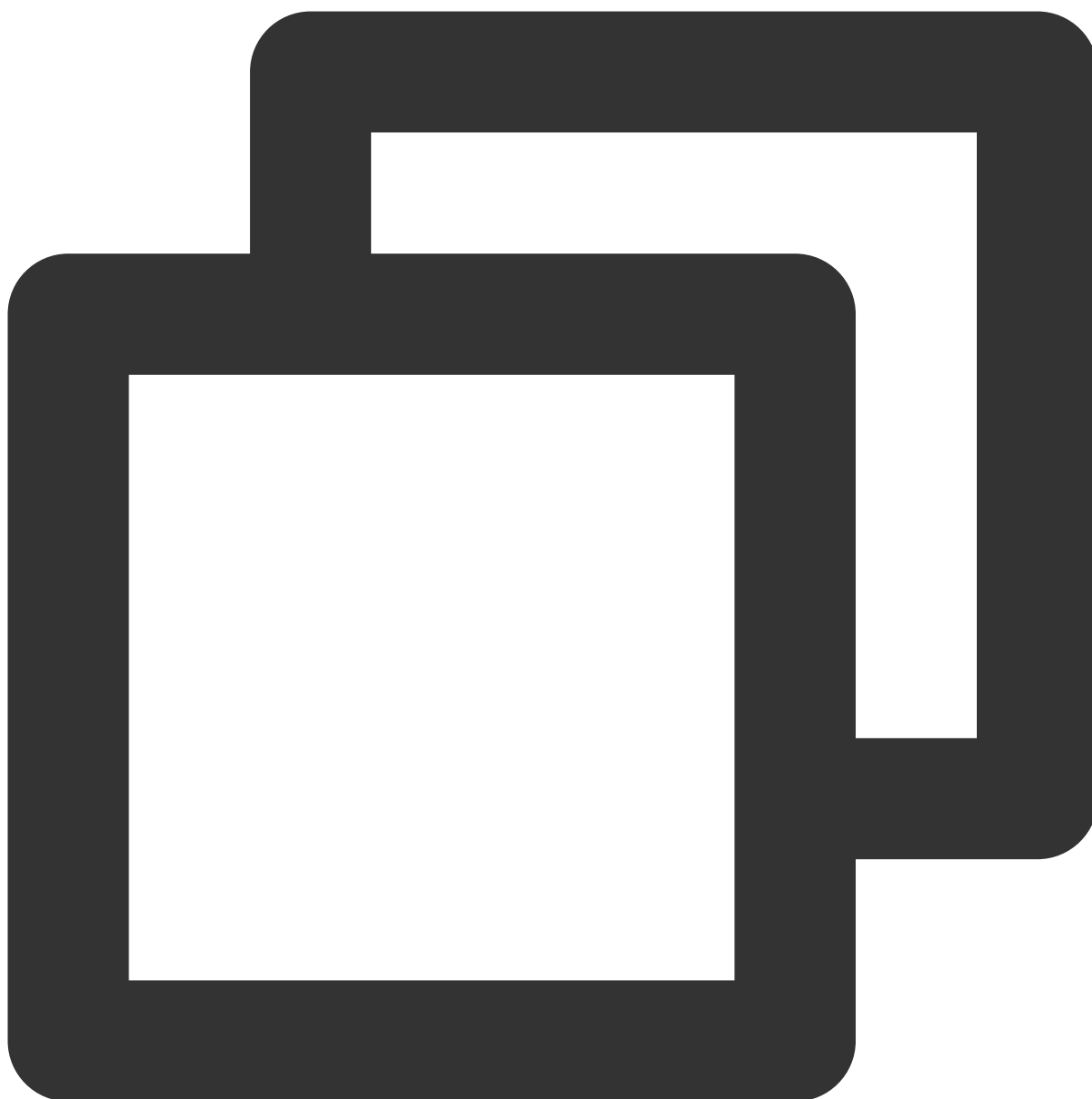
YTProcessInput Input data types and descriptions

When invoking the [process](#) interface, the output data type matches the input data type, YTTextureData type is always output.

Type	Meaning
YTImagePixelData	Camera data object, with the pixel format, is RGBA.
YTTextureData	Texture object, with the texture format, is OpenGL 2D.
YTImageRawData	Raw data object.
YTUIImageData	UIImage object.

process:withOrigin:withOrientation:

Data processing interface. The input and output data formats match [process](#). withOrigin: Setting whether to flip the image vertically. withOrientation: Setting the image rotation direction.



```
- (YTProcessOutput* _Nonnull)process:(YTProcessInput* _Nonnull)input withOrigin:(Yt
```

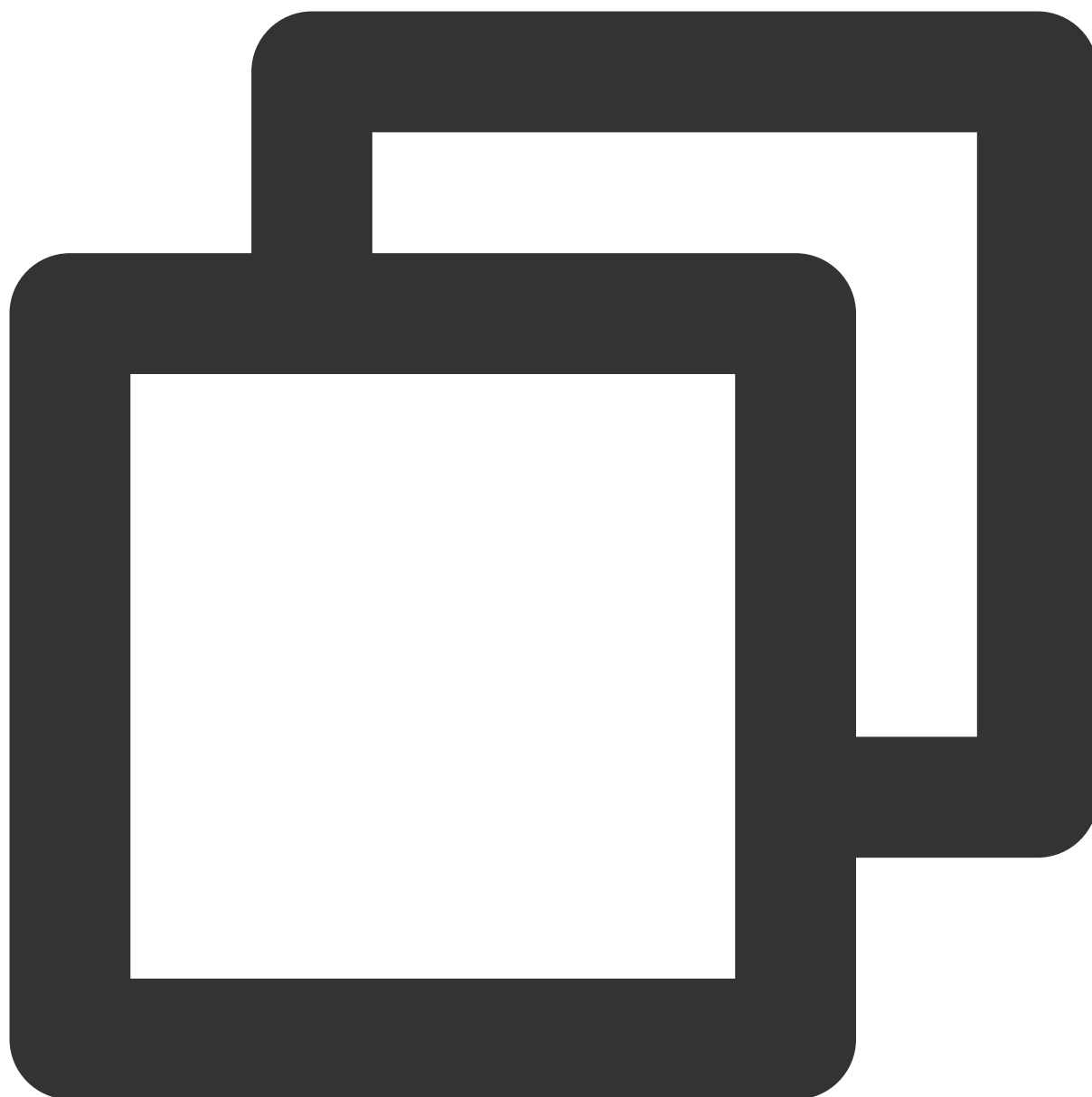
Parameter

Parameter	Meaning
input	Input data processing information.
withOrigin	Enumeration value (YtLightImageOriginTopLeft and

	YtLightImageOriginBottomLeft), when set to YtLightImageOriginBottomLeft, the image is flipped vertically.
withOrientation	Enumeration value: Image rotation angle, setting the angle will change the output image angle.

TEImageTransform tool class

Image Process tool class, input/output data formats include CVPixelBufferref and texture id. It supports mutually converting CVPixelBufferref data's bgra<-->yuv format, rotation, and vertical/horizontal mirroring. It also supports rotation and vertical/horizontal mirroring of the input in texture id format.



```
/// @param context If you are using the OpenGL interface of this class, we recommend  
- (instancetype)initWithEAGLContext:(EAGLContext *)context;
```

Parameter	Meaning
context	Using the context environment of OpenGL ES, you can pass [xMagic getCurrentGLContext].

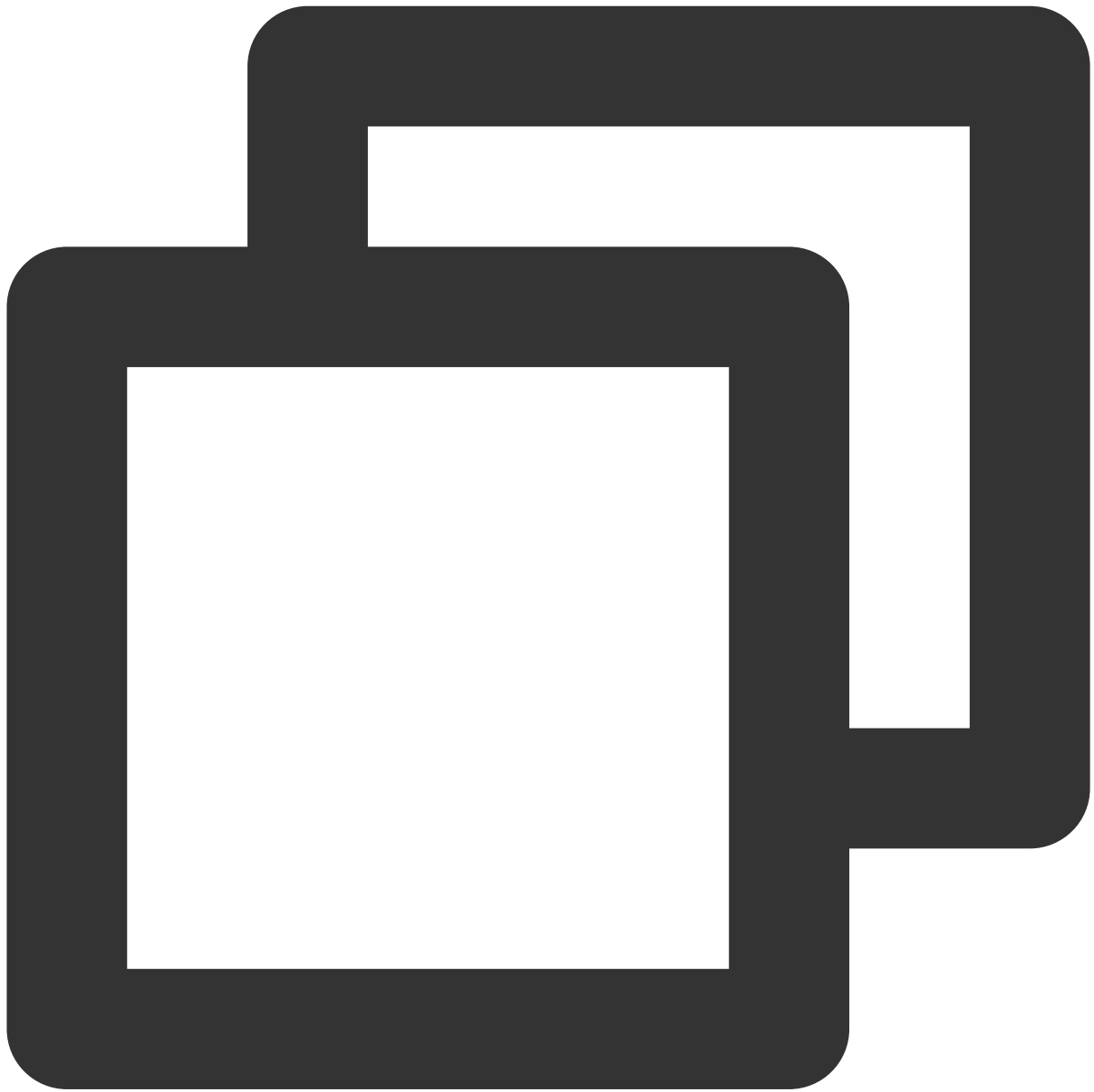


```
/// @brief CVPixelBufferRef yuv/rgb interconversion interface, currently, only supp
/// @param pixelBuffer Input pixelBuffer      input pixelBuffer
/// @param outputFormat Specify the type of output pixelBuffer      output pixelBuffs
- (CVPixelBufferRef)transformCVPixelBufferToBuffer:(CVPixelBufferRef)pixelBuffer ou
```

Parameter	Meaning
pixelBuffer	Input pixelBuffer data

outputFormat

Output pixelBuffer format, supports BGRA, NV12F(kCVPixelFormatType_420YpCbCr8BiPlanarFullRange), and NV12V(kCVPixelFormatType_420YpCbCr8BiPlanarVideoRange).

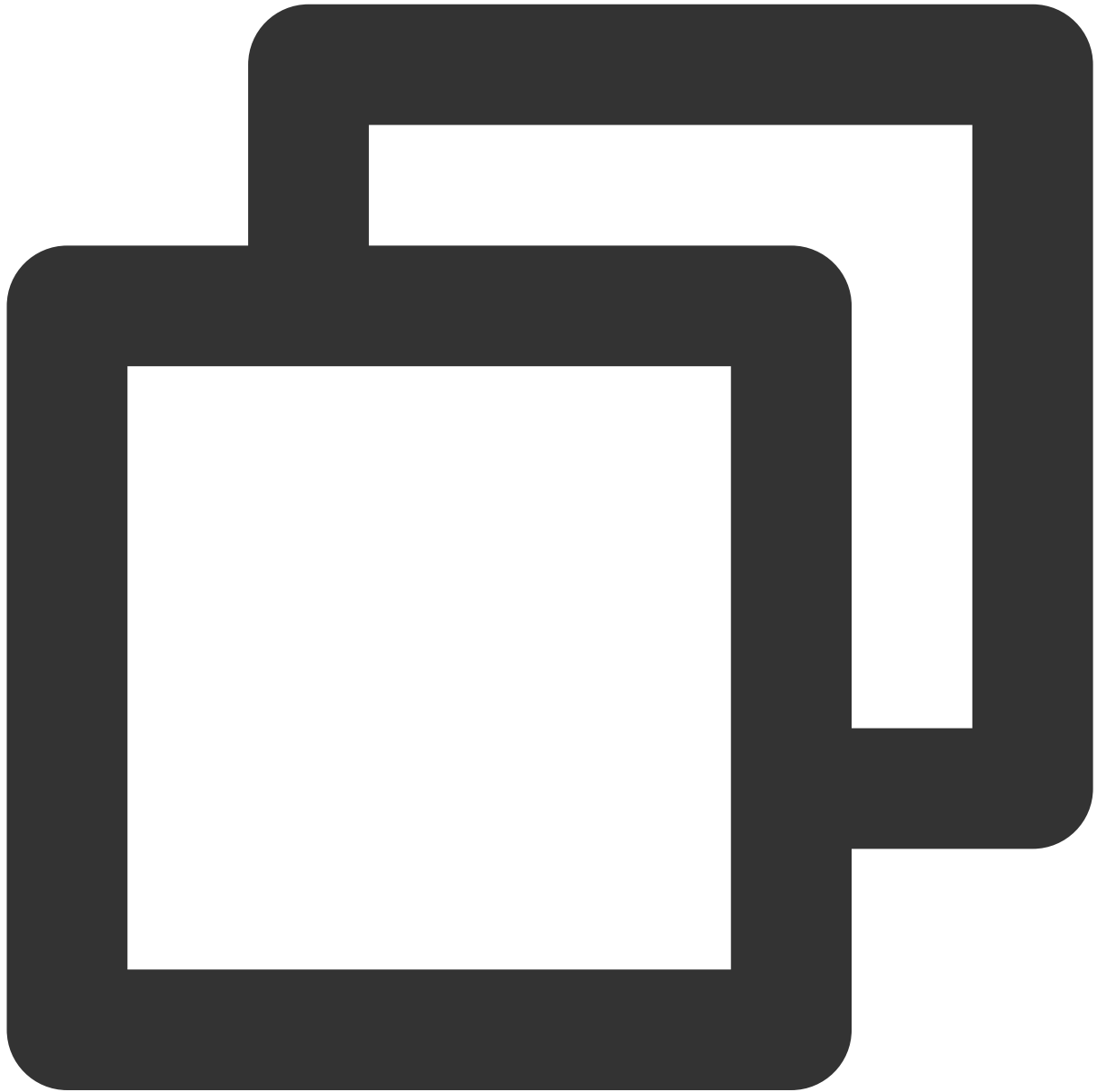


```
/// Convert yuv/rgb pixelBuffer to bgra format texture id
/// @param pixelBuffer Input pixelBuffer
- (GLuint)transformPixelBufferToBGRATexture:(CVPixelBufferRef)pixelBuffer;
```

Parameter

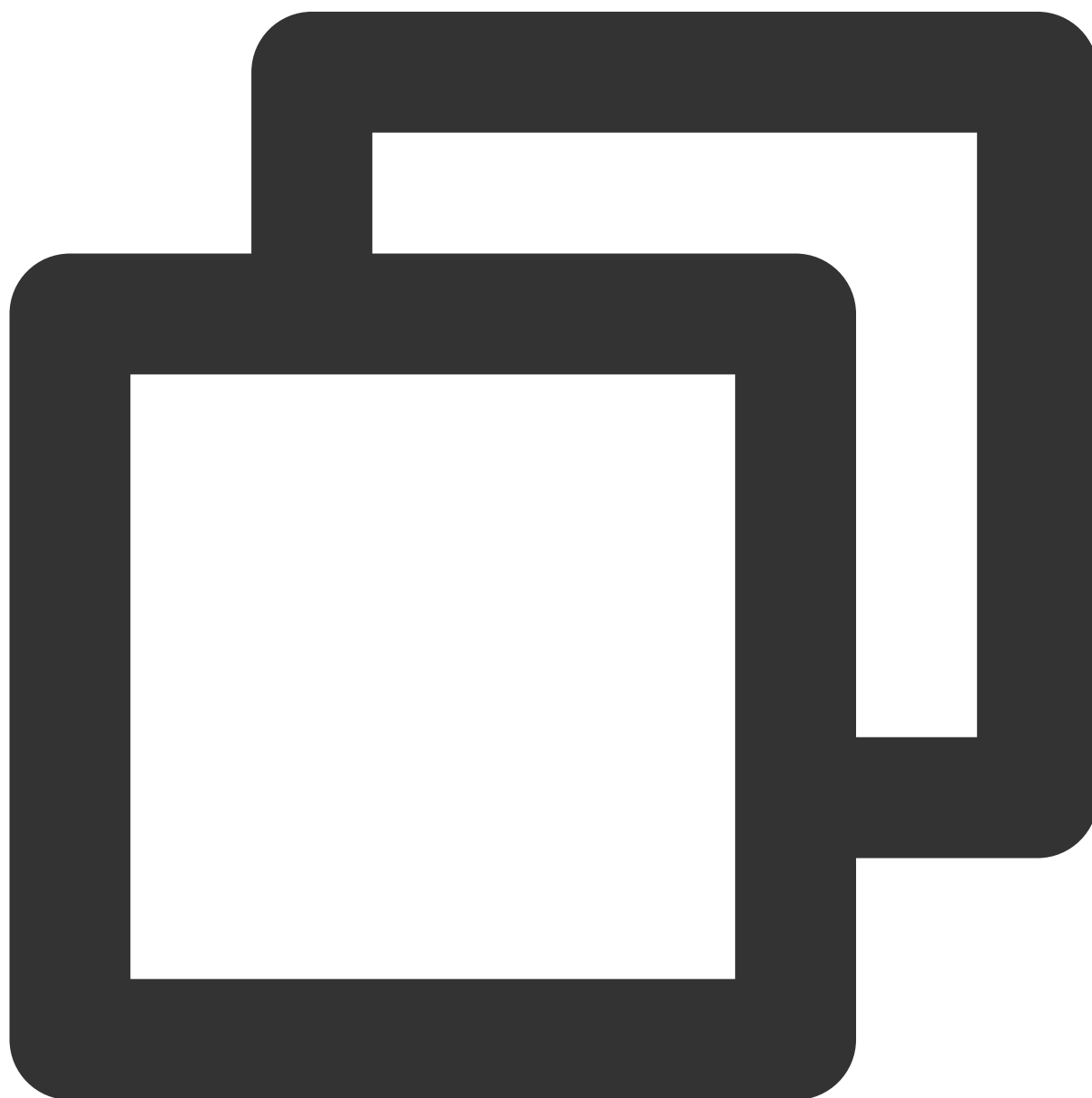
Meaning

pixelBuffer	Input pixelBuffer data, supports BGRA, NV12F(kCVPixelFormatType_420YpCbCr8BiPlanarFullRange), and NV12V(kCVPixelFormatType_420YpCbCr8BiPlanarVideoRange).
-------------	---



```
/// Rotate the CVPixelBufferRef and flip it. If you pass rotation and flipping at t  
- (CVPixelBufferRef)convertCVPixelBuffer:(CVPixelBufferRef)pixelBuffer rotaion:(YtL
```

Parameter	Meaning
pixelBuffer	Input pixelBuffer data
rotation	The counterclockwise rotation angle, supports 0 degree, 90 degrees, 180 degrees, and 270 degrees.
flipType	Mirror type, horizontal mirroring, or vertical mirroring. If both rotation and flipping are passed, the processing logic is to mirror first and then rotate.



```
/// Rotate/flip the texture Id, if both rotation and flipping are passed, the proce  
- (GLuint)convert:(GLuint)srcId width:(int)width height:(int)height rotaion:(YtLigh
```

Parameter	Meaning
srcId	Input Texture ID.
width	Texture width.
height	Texture height.
rotation	The Counterclockwise rotation angle, supports 0 degree, 90 degrees, 180 degrees, and 270 degrees.
flipType	Mirror type, horizontal mirroring, or vertical mirroring. If both rotation and flipping are passed, the processing logic is to mirror first and then rotate.

processUIImage

This API is used to process an image.



```
- (UIImage* _Nullable)processUIImage:(UIImage* _Nonnull)inputImage needReset:(bool)
```

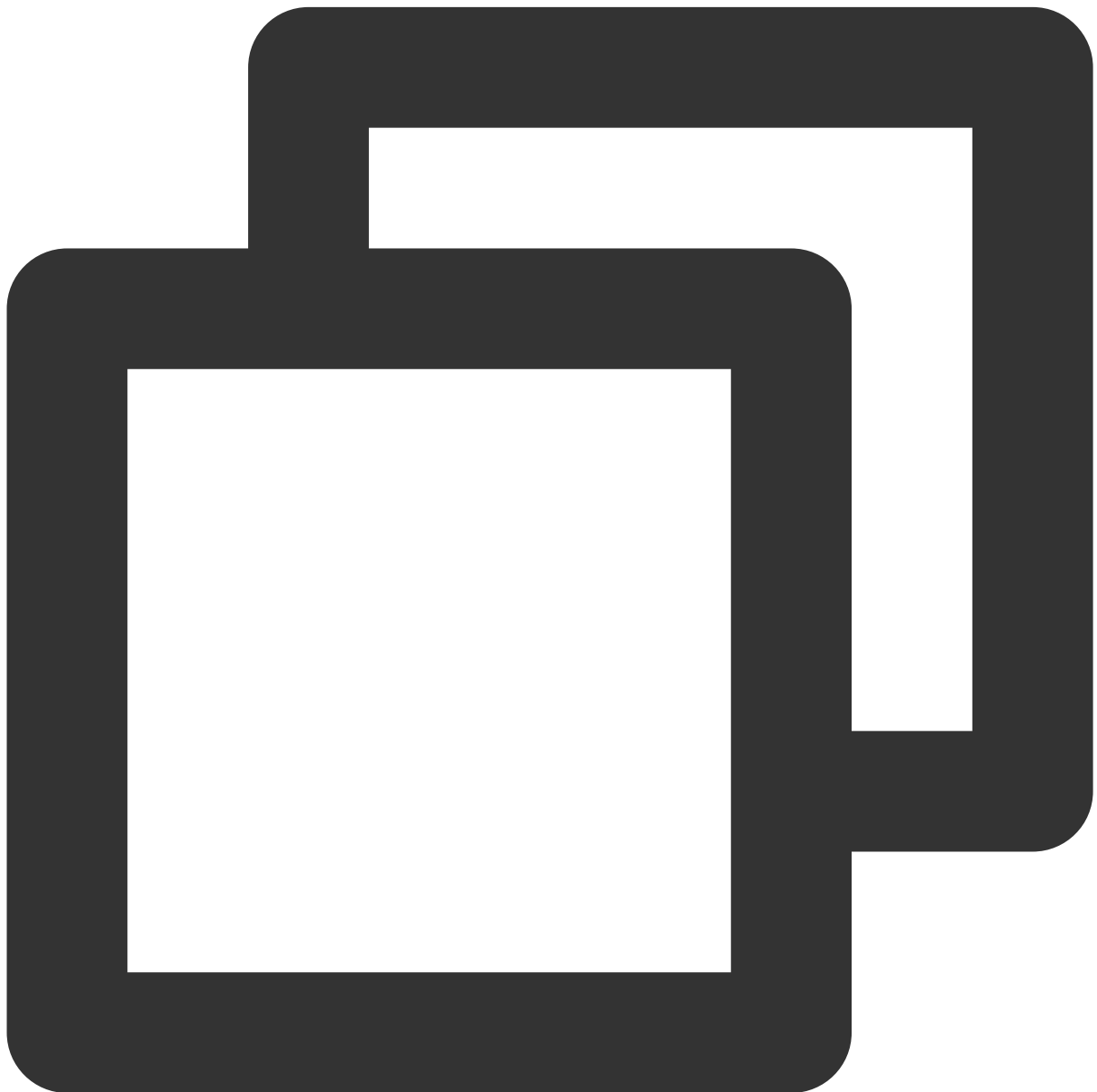
Parameters

Parameter	Description
inputImage	The input image. If your image is larger than 2160 x 4096, we recommend you reduce its size before passing it in; otherwise, face recognition may fail or may be inaccurate. It may also cause an OOM error.
needReset	This parameter must be set to true in the following cases:

The image processed is changed
The first time a keying effect is used
The first time an animated effect is used
The first time a makeup effect is used

getConfigPropertyWithName

This API is used to get effect information.



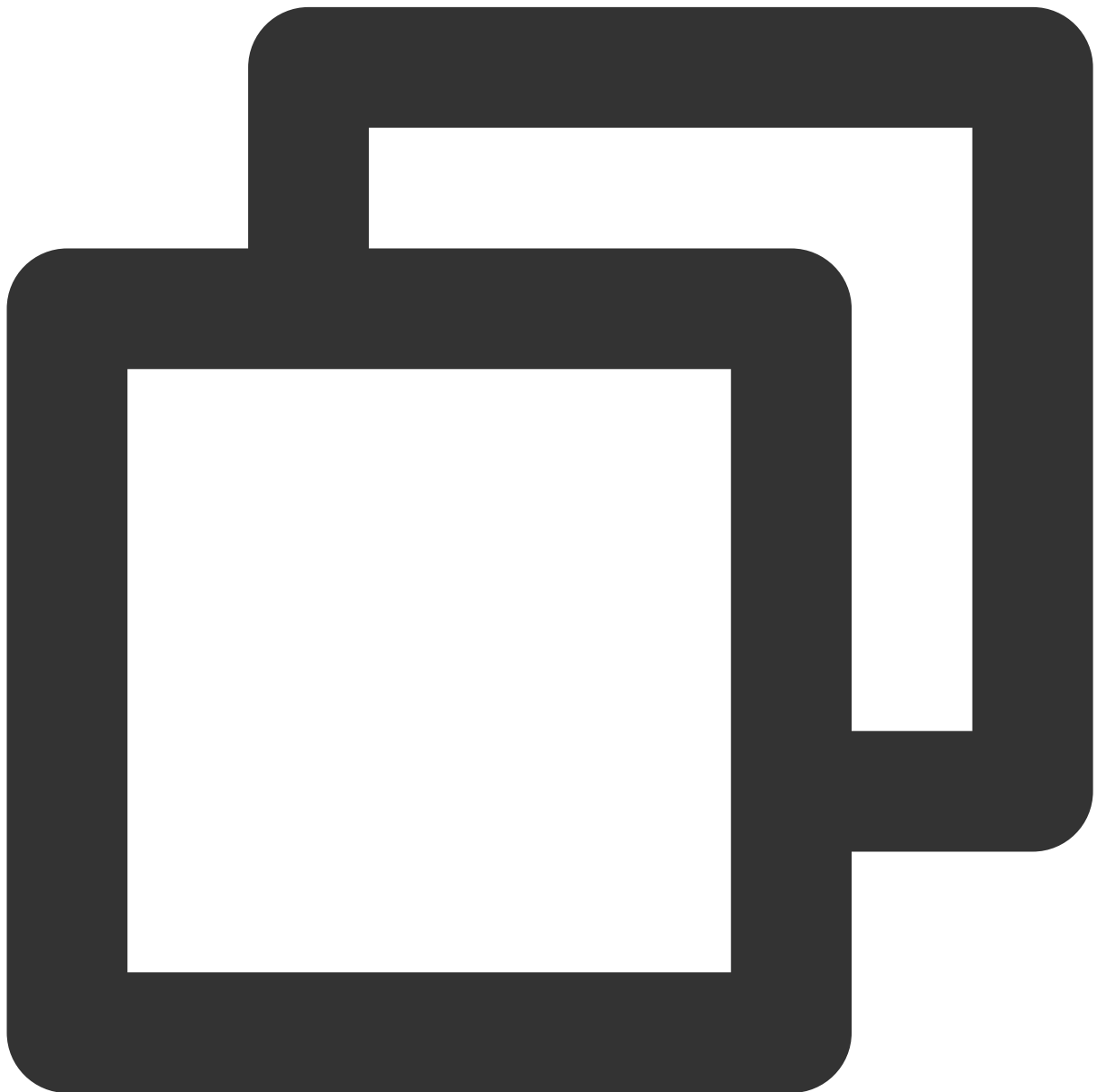
```
- (YTBeautyPropertyInfo * _Nullable)getConfigPropertyWithName:(NSString *_Nonnull)p
```

Parameters

Parameter	Description
propertyName	The effect name.

registerLoggerListener

This API is used to register a log listener.



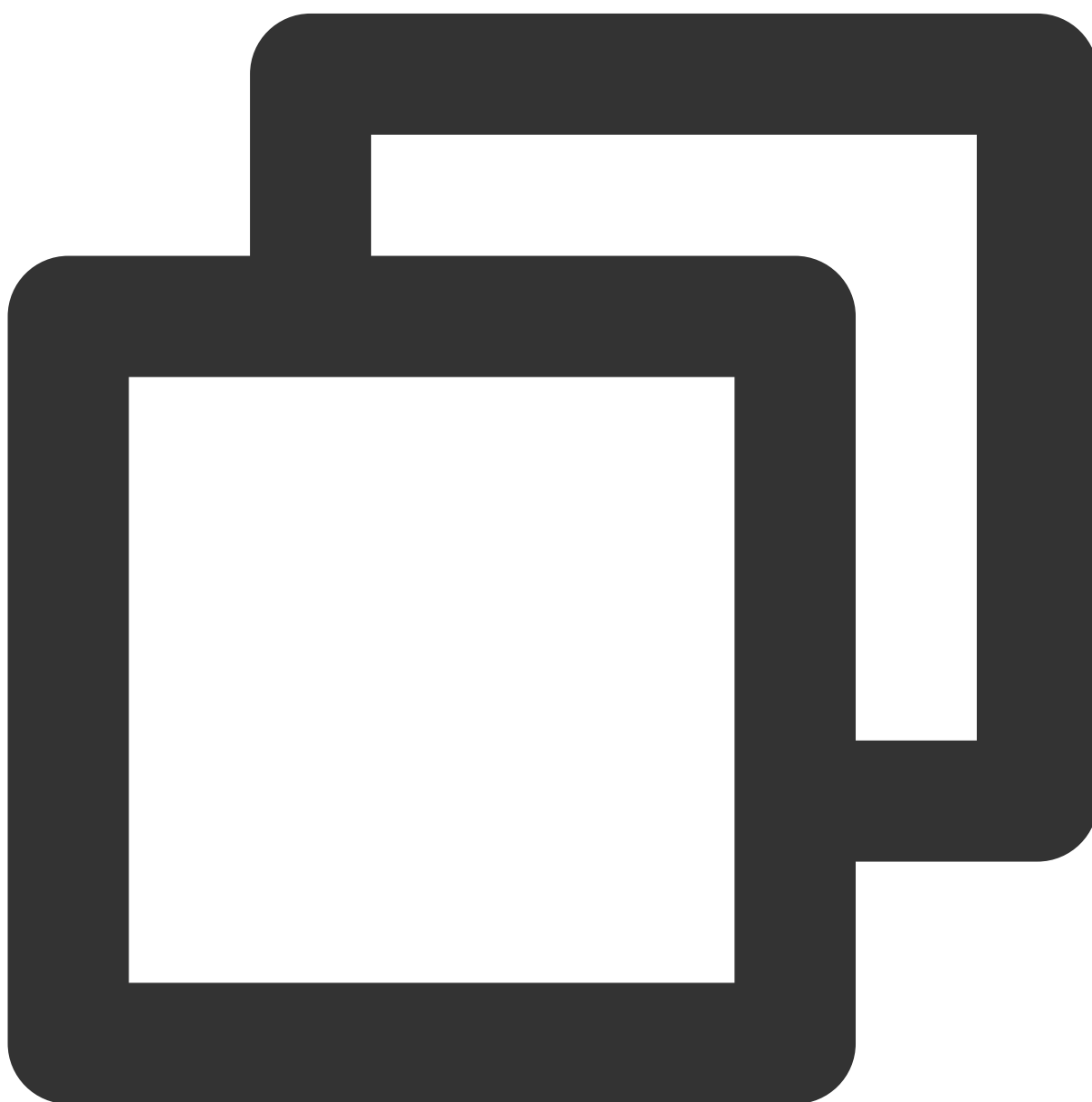
```
- (void)registerLoggerListener:(id<YTSDKLogListener> _Nullable)listener withDefault
```

Parameters

Parameter	Description
listener	The log callback.
level	The log output level, which is ERROR by default.

registerSDKEventListener

This API is used to register a listener for SDK events.



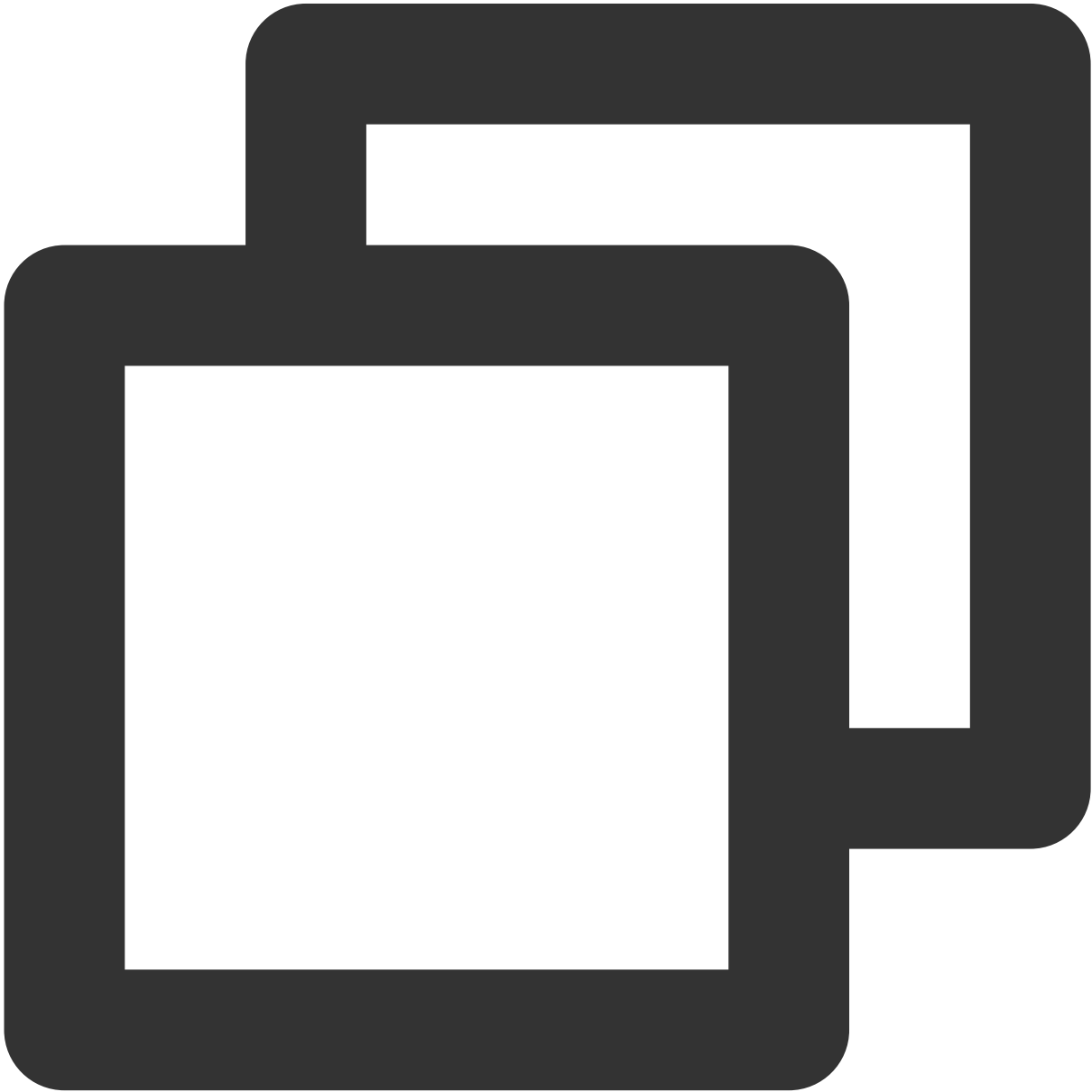
```
- (void)registerSDKEventListener:(id<YTSDKEventListener> _Nullable)listener;
```

Parameters

Parameter	Description
listener	The listener for SDK events, including AI events, tips, and resource events.

clearListeners

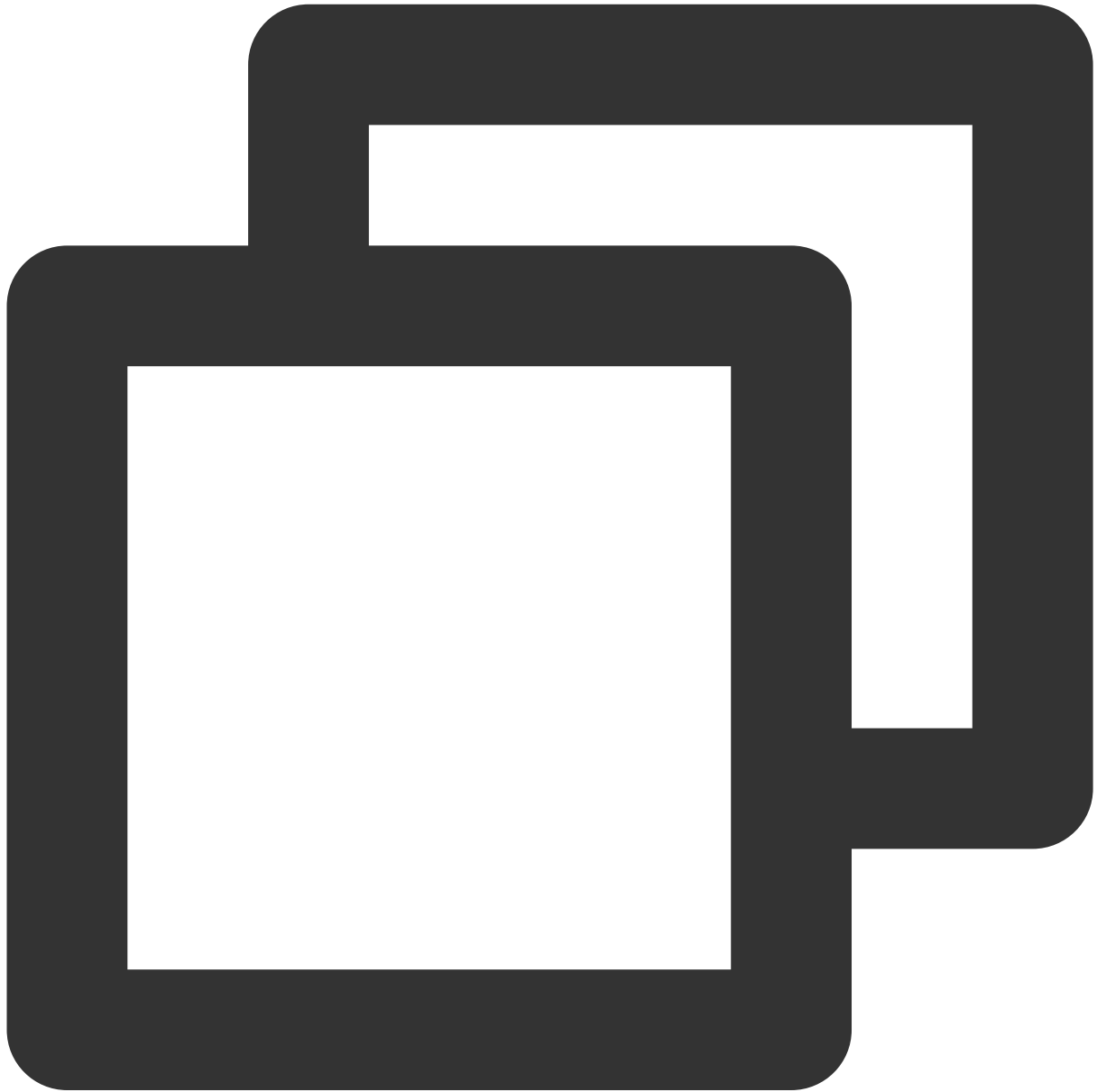
This API is used to remove listeners.



```
- (void)clearListeners;
```

getCurrentGLContext

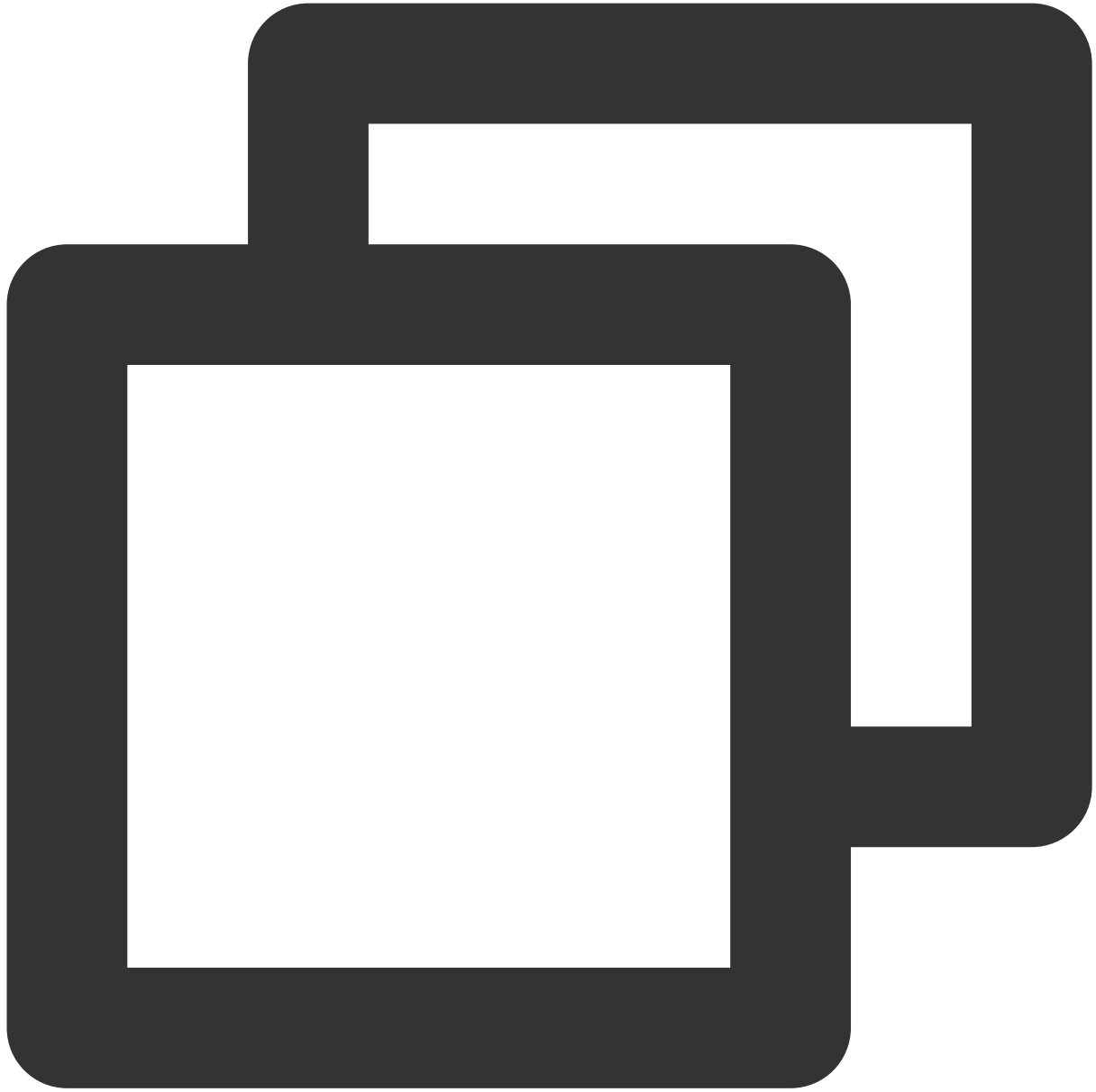
This API is used to get the current OpenGL context.



```
- (nullable EAGLContext*)getCurrentGLContext;
```

onPause

This API is used to pause the SDK.



```
/// @brief When your app is switched to the background, you need to call this API to  
- (void) onPause;
```

onResume

This API is used to resume the SDK.



```
/// @brief When your app is switched back to the foreground, you need to call this  
- (void)onResume;
```

setAudioMute

Whether to turn on mute when the dynamic effect material is used (new in V2.5.0)



```
/// @brief set mute  
- (void)setAudioMute:(BOOL)isMute;
```

enableEnhancedMode



```
/// @brief Activate the beautification enhancing pattern  
- (void)enableEnhancedMode;
```

Enable enhanced beauty filter pattern (Added in V2.5.1). By default, it is not activated.

When it's disabled, the application layer can set the intensity range of each beauty option from 0 to 1 or -1 to 1. If it exceeds this range, the SDK will take the boundary value. For example, if the application layer sets the face-slimming component to 1.2, SDK will limit it to the maximum value of 1.0, then it corrects the face-slimming value to 1.0 internally.

After enabling the enhanced pattern, the application layer can set a larger range of values. For example, if you want greater face slimming, you can set the face slimming value to 1.2. The SDK will accept and use this value and will not correct it to 1.0.

After enabling the enhanced pattern, the application layer needs to manage the maximum value that each beauty item can set, and let users adjust the value within this range. We provide a set of reference values, which you can adjust according to product requirements. However, we do not recommend exceeding our recommended values, otherwise the beauty effect may worsen. See the reference values below:

Beauty Item Name	In enhanced pattern, recommended maximum value (magnification factor)
Whitening, shortening the face, V-face, eye distance, nose position, removal of laugh lines, lipstick, three-dimensional appearance	1.3
Eye lightening	1.5
Blush	1.8
Other	1.2

High performance mode (V3.1.0.1)

With high performance mode enabled, the beauty feature will consume less system CPU/GPU resources, reducing overheating and lagging on the phone, making it more suitable for long-term use on low-end devices.

Note that the following beauty options will be unavailable when high performance mode is enabled:

- 1.Eye: Eye width, Eye height, Eye bags.
- 2.Eyebrow: Eyebrow angle, Eyebrow distance, Eyebrow Height, Eyebrow Length, Eyebrow Thickness, Eyebrow ridge.
- 3.Mouth: Smile face.
- 4.Facial: Thin face (Natural, Woman, Man), Slim jaw, Wrinkles, Smile lines. It is recommended to use Face shape to achieve a comprehensive effect of big eyes and thin "face".



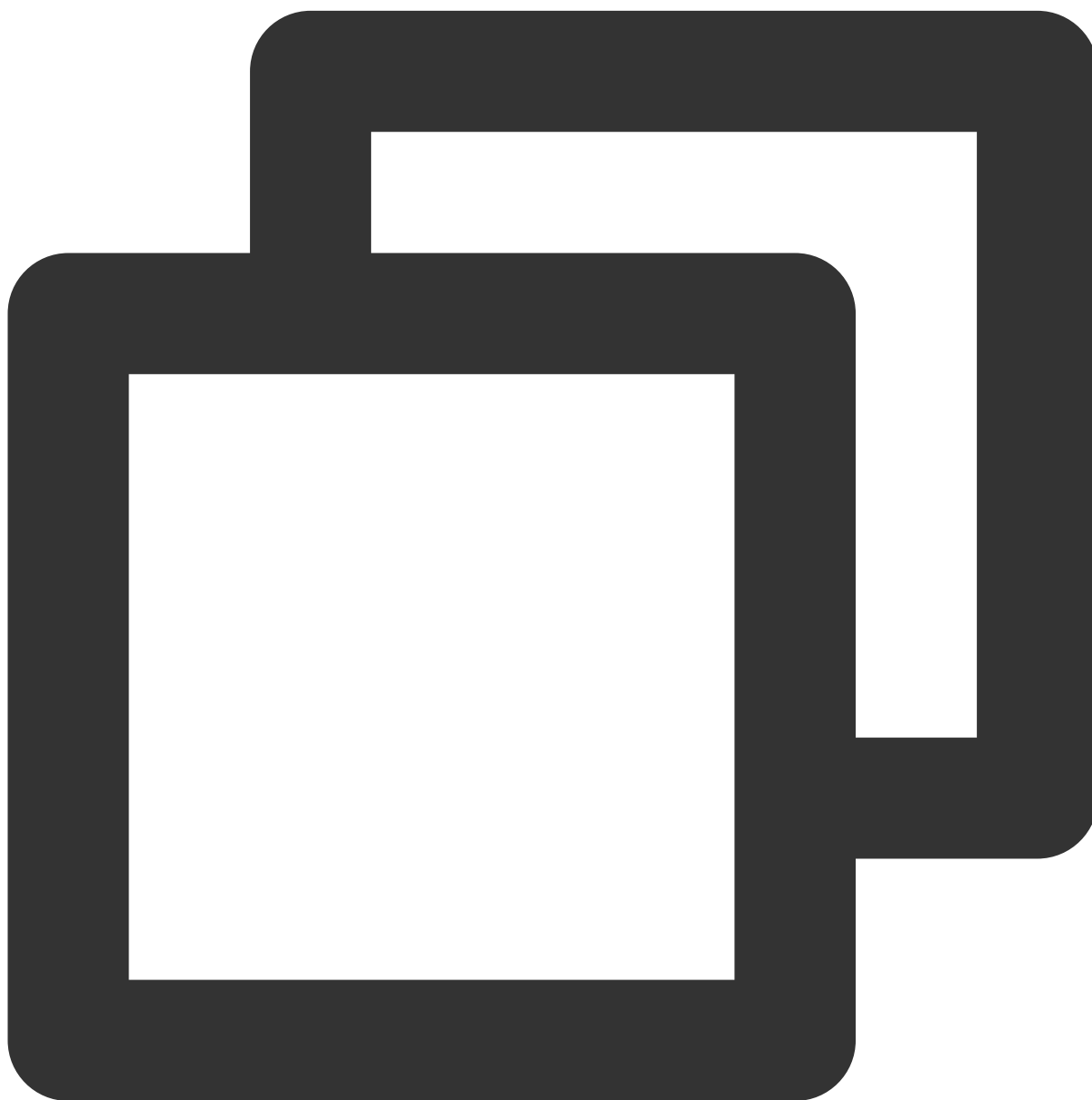
```
NSDictionary *assetsDict = @{@"core_name":@"LightCore.bundle",
                              @"root_path":[[NSBundle mainBundle] bundlePath],
                              @"setDowngradePerformance":@"(YES) //Enable high pe
};
self.beautyKit = [[XMagic alloc] initWithRenderSize:previewSize assetsDict:asse
```

Static APIs

API	Description
isBeautyAuthorized	Gets the authorization information of an effect parameter.

isBeautyAuthorized

This API is used to get the authorization information of the effect parameter (only supports the parameter of beauty and body retouch).



```
/// @param featureId: The effect parameter.  
/// @return: The authorization information of the effect parameter.
```

```
+ (BOOL)isBeautyAuthorized:(NSString * _Nullable)featureId;
```

Callback

API	Description
YTSDKEventListener	The SDK event callback.
YTSDKLogListener	The log callback.

YTSDKEventListener

The callback for the internal events of the SDK.



```
@protocol YTSDKEventListener <NSObject>
```

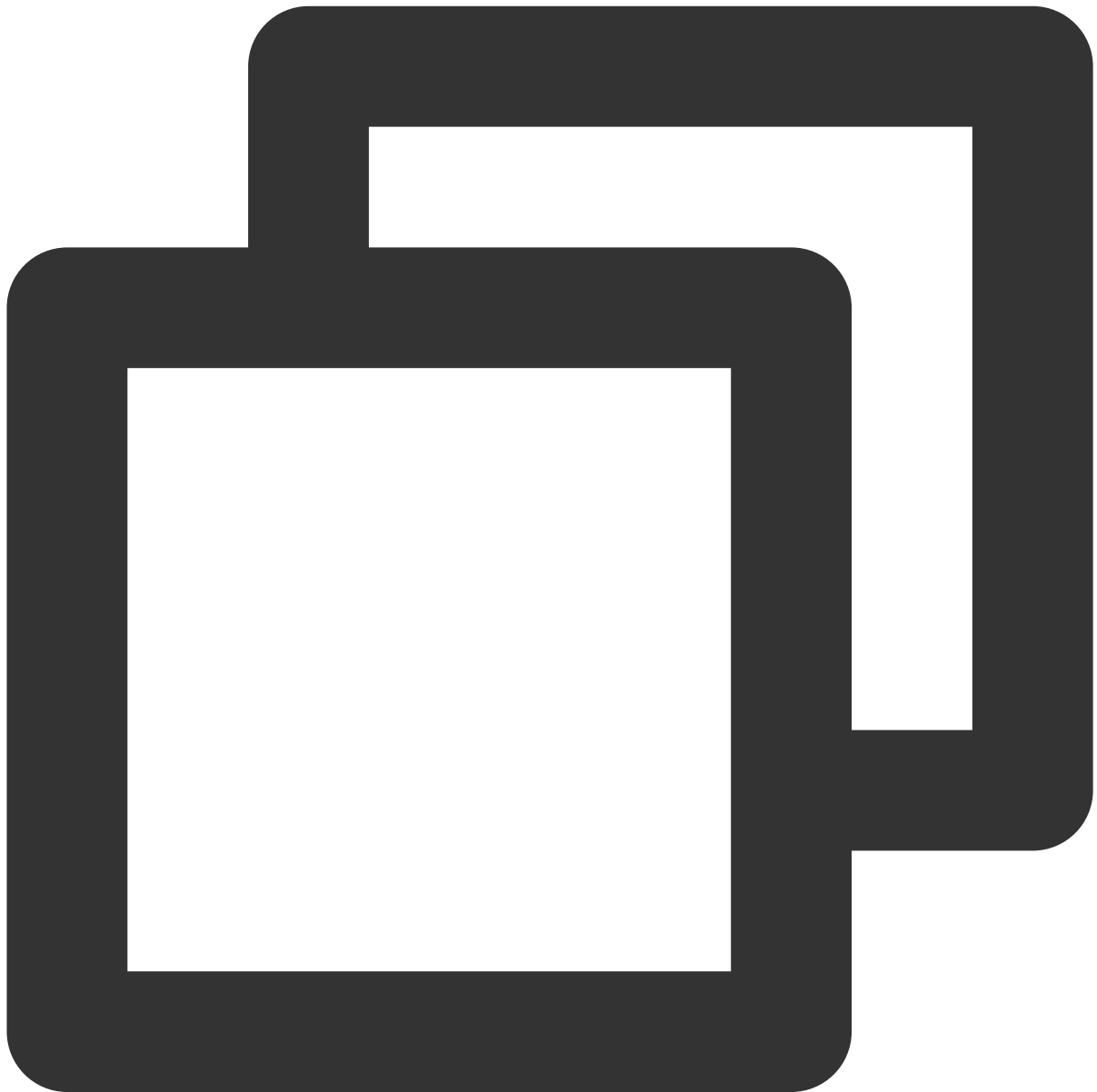
Member callback APIs

Return Type	Callback
void	onAIEvent
void	onTipsEvent
void	onAssetEvent

Callback description

onAIEvent

The YTDataUpdate event callback.



```
/// @param event Callback in dict format  
- (void)onAIEvent:(id _Nonnull)event;
```

The information of up to five faces is returned as JSON strings.



```
{
  "face_info": [{
    "trace_id": 5,
    "face_256_point": [
      180.0,
      112.2,
      ...
    ],
    "face_256_visible": [
      0.85,
      ...
    ]
  }]
```



```

    ],
    "out_of_screen":true,
    "left_eye_high_vis_ratio":1.0,
    "right_eye_high_vis_ratio":1.0,
    "left_eyebrow_high_vis_ratio":1.0,
    "right_eyebrow_high_vis_ratio":1.0,
    "mouth_high_vis_ratio":1.0
  },
  ...
]
}

```

Field Description

Field	Type	Value Range	Remarks
trace_id	int	[1,INF)	The face ID. If the faces obtained continuously from a video stream have the same face ID, they belong to the same person.
face_256_point	float	[0,screenWidth] or [0,screenHeight]	512 values in total for 256 facial keypoints. (0,0) is the top-left corner of the screen.
face_256_visible	float	[0,1]	The visibility of the 256 facial keypoints.
out_of_screen	bool	true/false	Whether only part of the face is captured.
left_eye_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the left eye.
right_eye_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the right eye.
left_eyebrow_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the left eyebrow.
right_eyebrow_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the right eyebrow.
mouth_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the mouth.

onAIEvent

The callback for AI events.



```
/// @param event: Callback in dict format  
- (void)onAIEvent:(id _Nonnull)event;
```

onTipsEvent

The callback for tips.



```
/// @param event: Callback in dict format  
- (void)onTipsEvent:(id _Nonnull)event;
```

onAssetEvent

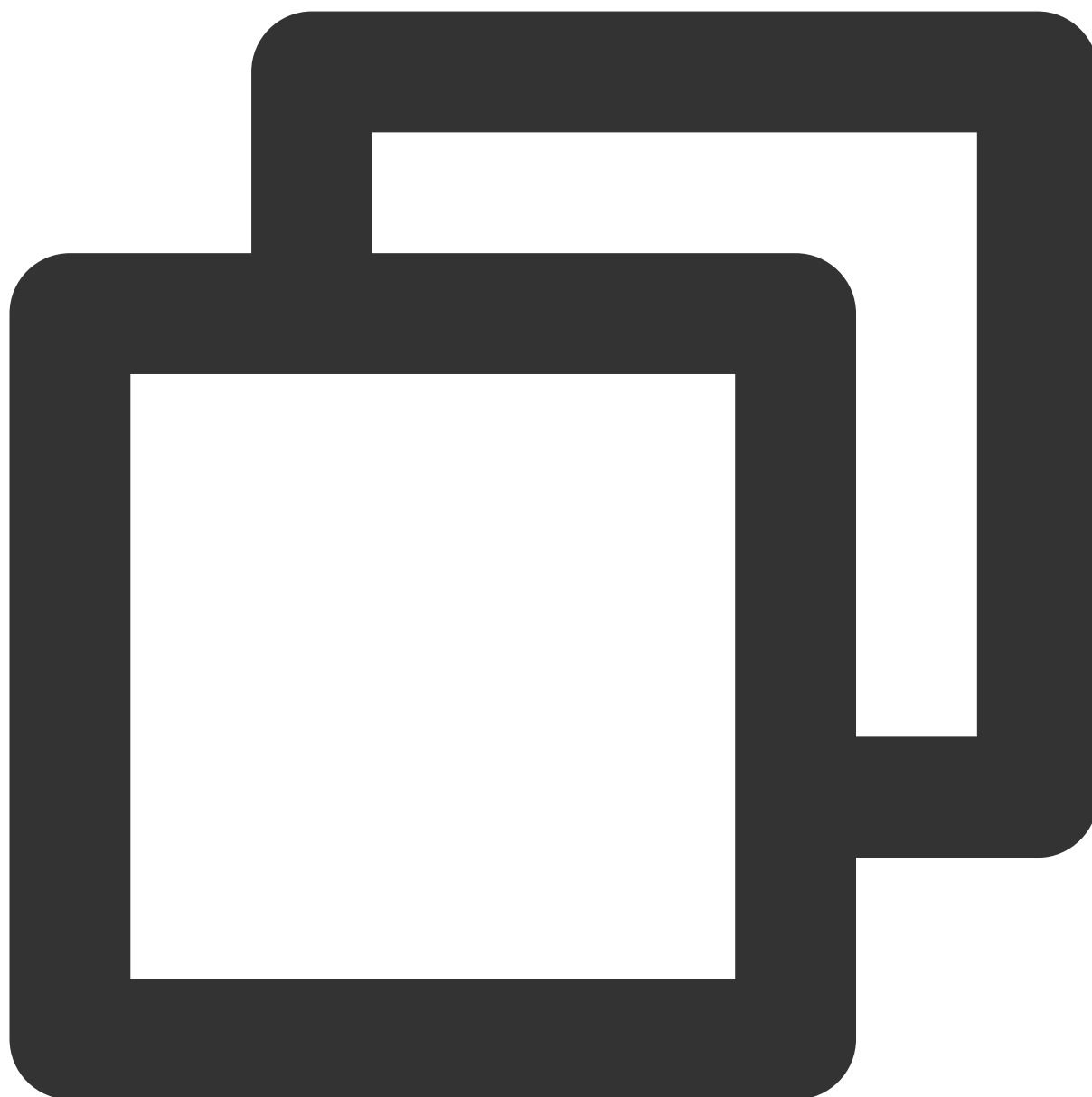
The callback for resource events.



```
/// @param event: Callback in string format  
- (void)onAssetEvent:(id _Nonnull)event;
```

YTSDKLogListener

The log callback.



```
@protocol YTSDKLogListener <NSObject>
```

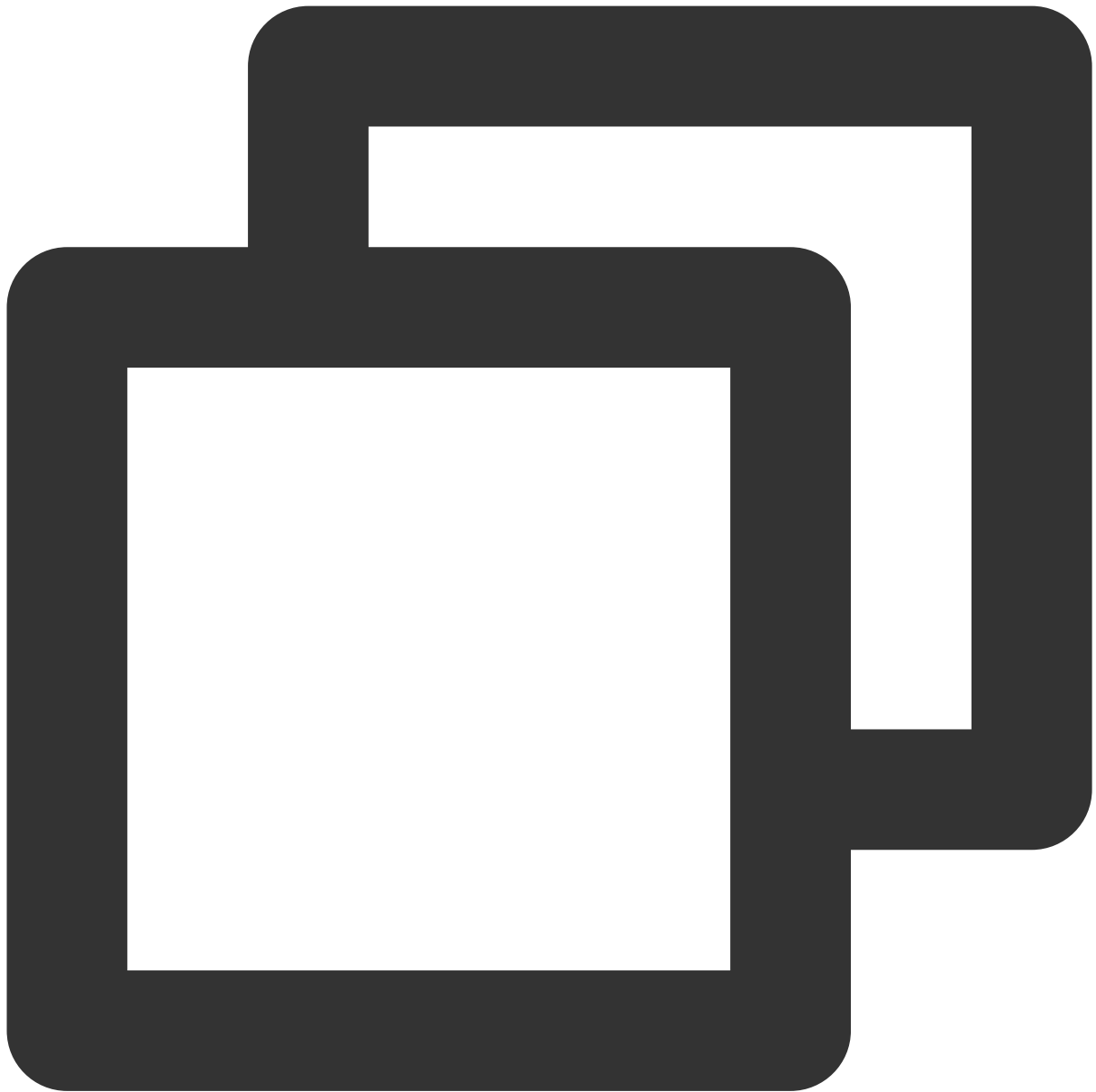
Member callback APIs

Return Type	API
void	onLog

Callback description

onLog

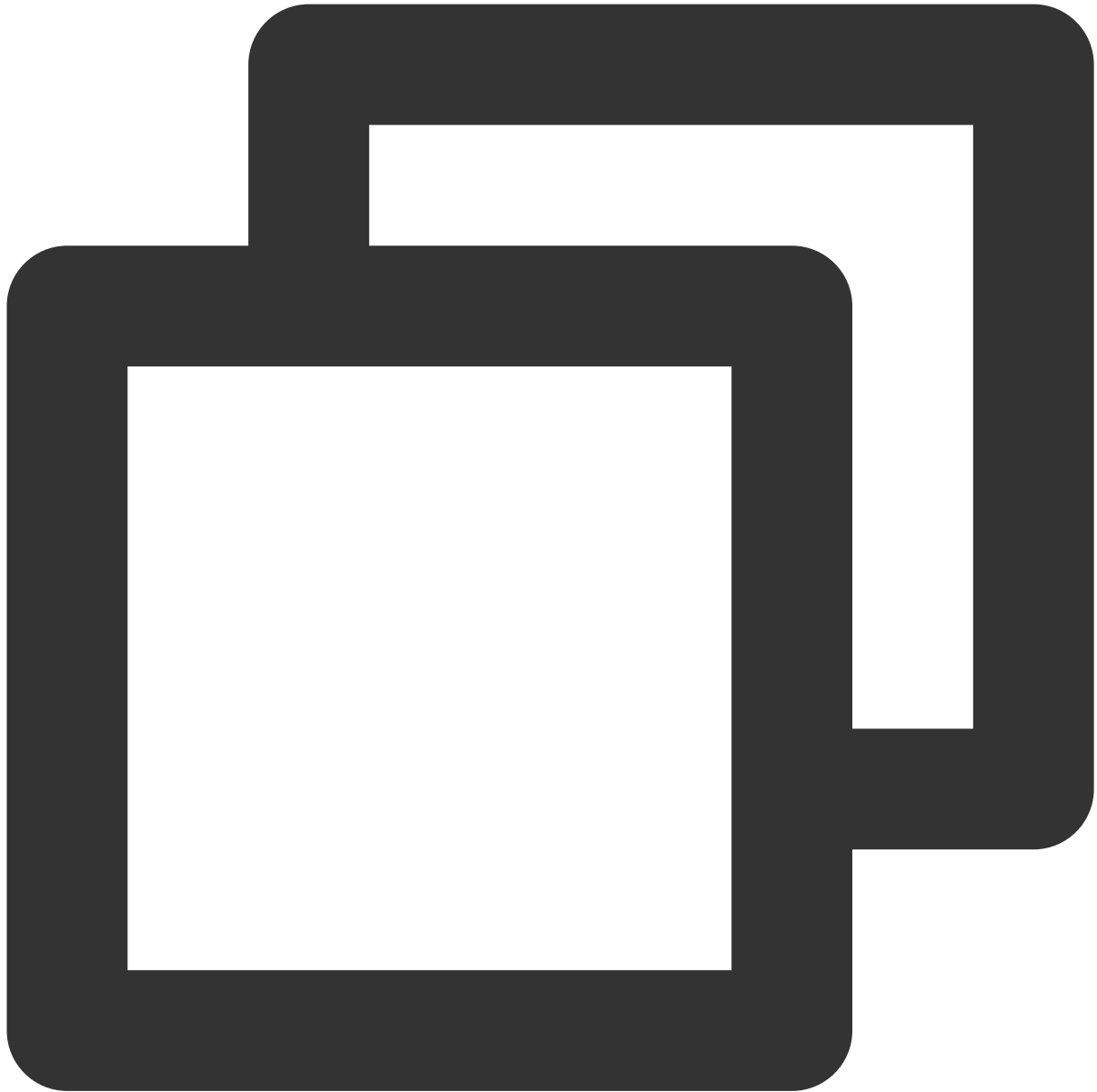
The log callback.



```
/// @param loggerLevel: The current log level.  
/// @param logInfo: The log information.  
- (void)onLog:(YtSDKLoggerLevel) loggerLevel withInfo:(NSString * _Nonnull) logInfo
```

Material overlay (added in version 3.0.1.5)

If you want to overlay a certain animation/beauty/segmentation material on the current material, when setting up the material, set 'mergeWithCurrentMotion' to true in the dictionary of 'withExtraInfo', An example is shown below:



```
NSString *key = _xmagicUIProperty.property.Id;
NSString *value = [[NSBundle mainBundle] pathForResource:@"makeupMotionRes" ofType:
NSDictionary* extraInfo = @{@"mergeWithCurrentMotion":@(true)};
[self.beautyKitRef configPropertyWithType:@"motion" withName:key withData:[NSStrin
```

The precautions for material overlay are :

1. The client needs to manage whether the materials are suitable for overlaying. Here are two examples:

Example 1: Effect A is to turn into a noblewoman's face, and effect B is to turn into a fairytale face. Overlaying these two effects may result in a very awkward image.

Example 2: Effect A is a pair of rabbit ears, and effect B is a pair of pig ears. Overlaying these two effects will result in two types of ears.

These two cases are not suitable for overlaying. If effect A is a pair of rabbit ears and effect B is blowing a kiss, these two effects will not conflict and are suitable for overlaying.

2. Only simple materials can be overlaid. Simple materials refer to those with only one animation effect, makeup effect, or background removal effect, while complex materials refer to those that contain multiple effects. There is no clear boundary between simple and complex materials, so it is recommended that clients fully test and manage which materials can be overlaid and which cannot.

3. When overlaying, effects triggered by actions (such as triggering an effect by reaching out or smiling) are complex effects and need to be placed in front, with simple effects overlaid on top.

4. Usage example: The anchor uses effect A, and then the audience sends gift effect B, which needs to be overlaid on A. After a period of time, B disappears and returns to effect A. The settings are as follows:

4.1. Set effect A, and set `mergeWithCurrentMotion` to false.

4.2. Set effect B, and set `mergeWithCurrentMotion` to true.

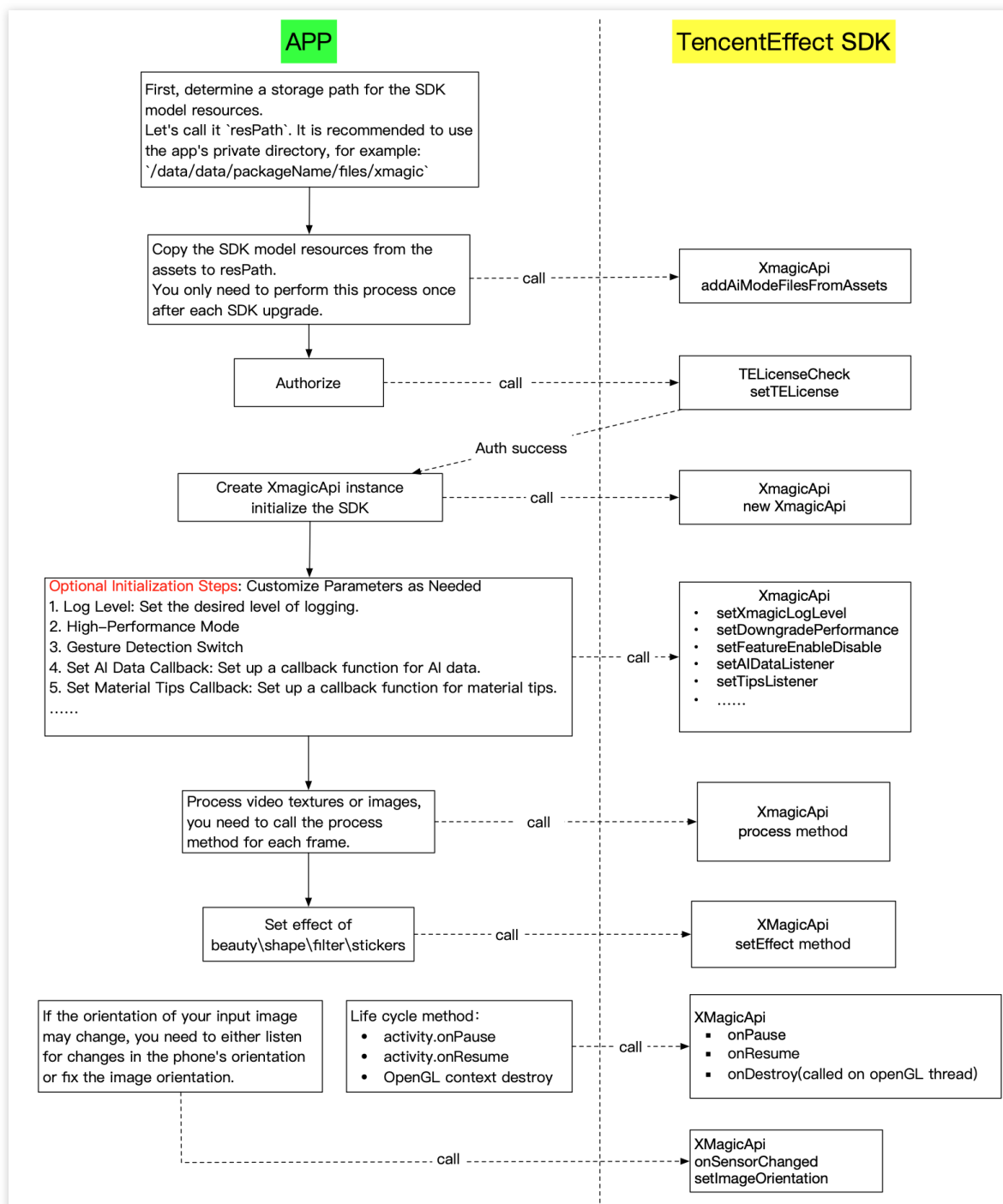
4.3. After a short period of time, set A again, and set `mergeWithCurrentMotion` to false.

Android

Last updated : 2024-08-08 19:55:43

Tencent Effect SDK Core Interface Class `XmagicApi.java` , utilized for initializing the SDK, updating beauty metric values, invoking animated effects, amongst other features.

The overall calling process of each API is as follows:



Public Member Functions

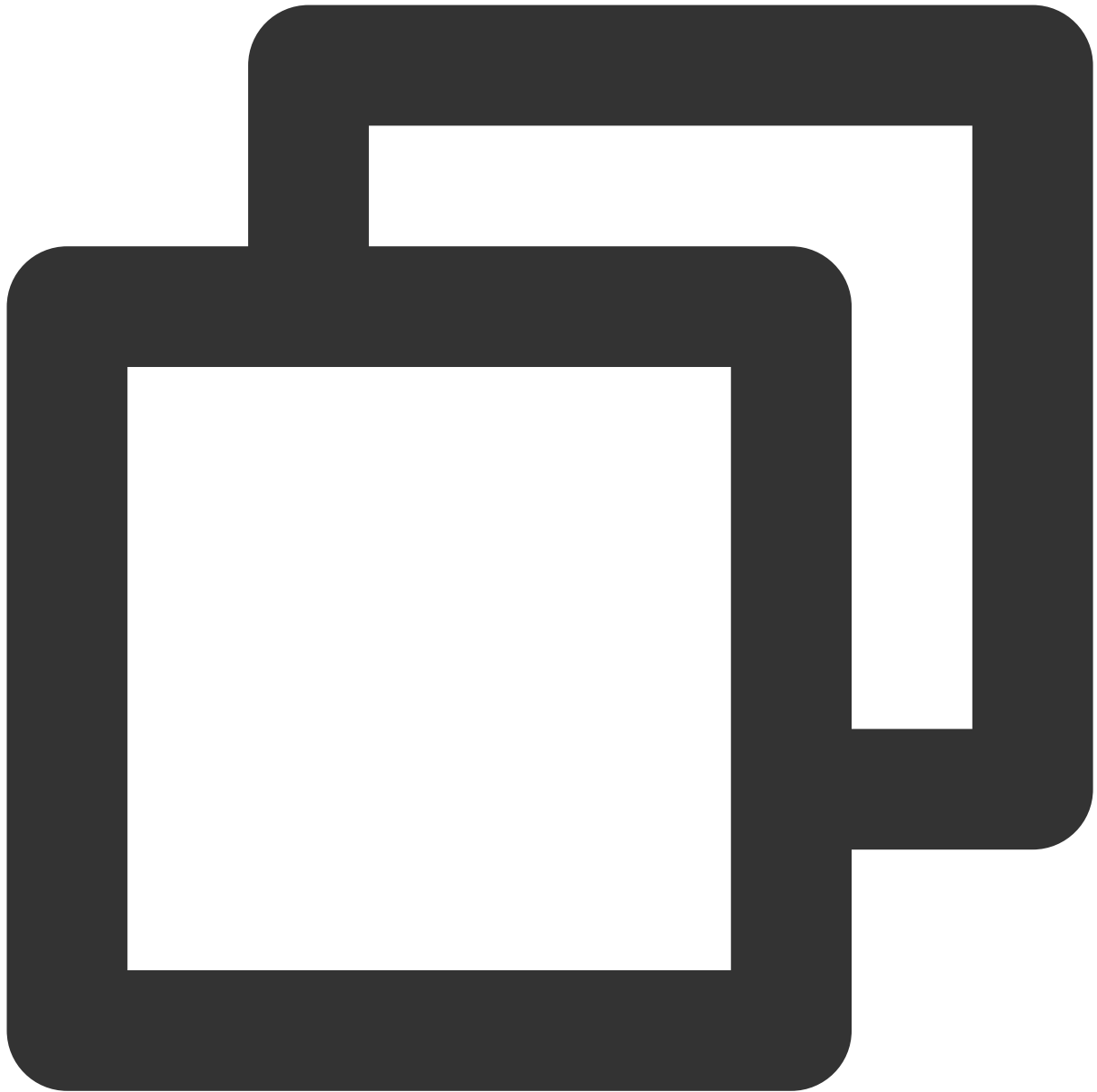
API	Description
XmagicApi	constructor.
@Deprecated updateProperty	Updates the attribute, it can be invoked in any thread (V3.3.0 and before).
@Deprecated updateProperties	Performing bulk attribute updates can be invoked in any thread (V3.3.0 and previous).
setEffect	Establishing beauty enhancements, styling, filters, makeup applications, stickers, division and other effects can be invoked in any thread (New in V3.5.0 version)
setTipsListener	Setting up callback functions for animated hint text, designated to display hints on the frontend page.
@Deprecated setYTDDataListener	Configure the callback of facial keypoints and other data, callback will only be available with the License authorization required to acquire facial keypoints (such as Atomic Capability X102).
setAIDataListener	Configure the callback for face, gesture, and body detection statuses.
onPause	Pause audio playback, which can be associated with the `onPause` lifetime affinity of `Activity`.
onResume	Resume rendering, can be paired with Activity onResume lifetime affinity.
onDestroy	Terminate `xmagic`, which necessitates its invocation within the `GL` thread.
process	The method for SDK rendering to accept data can be used within the camera data callback function.
@Deprecated onPauseAudio	Invoke this function when only the cessation of the audio is necessitated, without the need to release the GL thread. (This interface has been deprecated)
sensorChanged	It serves to establish the present angle of rotation of the mobile phone, thus adjusting the angle for AI to recognize faces based on this.
isDeviceSupport	The list of dynamic effect resources is introduced into the SDK for examination, executing afterwards the <code>XmagicProperty.isSupport</code> field represents whether the atomic capability is applicable. According to <code>XmagicProperty.isSupport</code> , the UI layer can implement click restrictions, or directly eliminate from the resource list.

@Deprecated getPropertyRequiredAbilities	Pass in a list of dynamic effect resources and return the list of SDK atomic capabilities used by each resource. (This interface has been deprecated)
getDeviceAbilities	Return the atomic capability list supported by the current device.
isSupportBeauty	Determine whether the current device supports refinement (OpenGL3.0).
isBeautyAuthorized	Determine which beauties the current lic authorization supports. It only supports the detection of BEAUTY and BODY_BEAUTY types of beauty items. The detection result will be assigned to the XmagicProperty.isAuth field of each beauty object.
setXmagicStreamType	Set the input data type, the default is Android camera data stream.
setXmagicLogLevel	Establish the log level of the SDK, the default is <code>WARN</code> . During the development and debugging stage, if needed, it can be set to <code>Log.DEBUG</code> . Be sure to set to <code>Log.WARN</code> or <code>Log.ERROR</code> upon official release, otherwise excessive logs will affect performance. Invoke after new XmagicApi().
setAudioMute	Whether to mute when using animation materials (Added in V2.5.0): Parameter: true indicates mute, false denotes non-mute.
enableEnhancedMode	Activate the beautification enhancing pattern (Added in V2.5.1). By default, it is not activated. When it's disabled, the application layer can set the strength range for each beauty field between 0 to 1 or -1 to 1. If it exceeds the boundary, SDK will fixate to the boundary value. For example, if the application layer sets face-slimming component to 1.2, SDK will limit it to the maximum value of 1.0. After enabling the enhanced pattern, the application layer can set a broader range of values. For instance, if a greater face slimming effect is desired, you can set the slimming value to 1.2. The SDK will accept and use this value of 1.2 without correcting it to 1.0. Note: Upon enabling the enhancement pattern, the application layer must manage the maximum settable value for each beauty field, allowing users to adjust the value within such boundary parameters. We offer a reference value that you can adjust in accordance to product demand. However, we do not recommend exceeding our advised estimates as this could potentially deteriorate the beauty effect.
@Deprecated setDowngradePerformance	Invoke this method to enable the high-performance pattern. Upon the activation of the high-performance pattern, the system CPU/GPU resources occupied by beauty filters are minimized, thereby reducing heat generation and latency issues in the mobile device. It is particularly suitable for prolonged use on low-end devices. (This interface has been deprecated)

<code>enableHighPerformance</code>	Invoke this method to enable the high-performance pattern. Upon the activation of the high-performance pattern, the system CPU/GPU resources occupied by beauty filters are minimized, thereby reducing heat generation and latency issues in the mobile device. It is particularly suitable for prolonged use on low-end devices.
<code>exportCurrentTexture</code>	Get the picture on the current texture
<code>setFeatureEnableDisable</code>	Enable or disable a certain capability.

XmagicApi

constructor.



```
XmagicApi(Context context, String resDir)  
XmagicApi(Context context, String resDir, OnXmagicPropertyErrorListener xmagicProper
```

Parameter

Parameter	Type	Meaning
context	Context	context.
resDir	String	Resource directory.

		<p>If the SDK resources are built into assets, before the SDK, the resources need to be copied to the assets directory. The resource path is set first via</p> <pre>XmagicResParser.setResPath(new File(getFilesDir(), "xmagic").getAbsolutePath());</pre> <p>and then the resource copying is completed via</p> <pre>XmagicResParser.copyRes(getApplication().getAssets());</pre> <p>See the <code>TEMenuActivity.java</code> document for detailed information.</p> <p>If the SDK resources are downloaded from the internet, after a successful download, the resource path can be set via</p> <pre>XmagicResParser.setResPath(validResPath);</pre> <p>Access the previously set path via</p> <pre>XmagicResParser.getResPath()</pre>
xmagicPropertyErrorListener	OnXmagicPropertyErrorListener	Error callback interface.

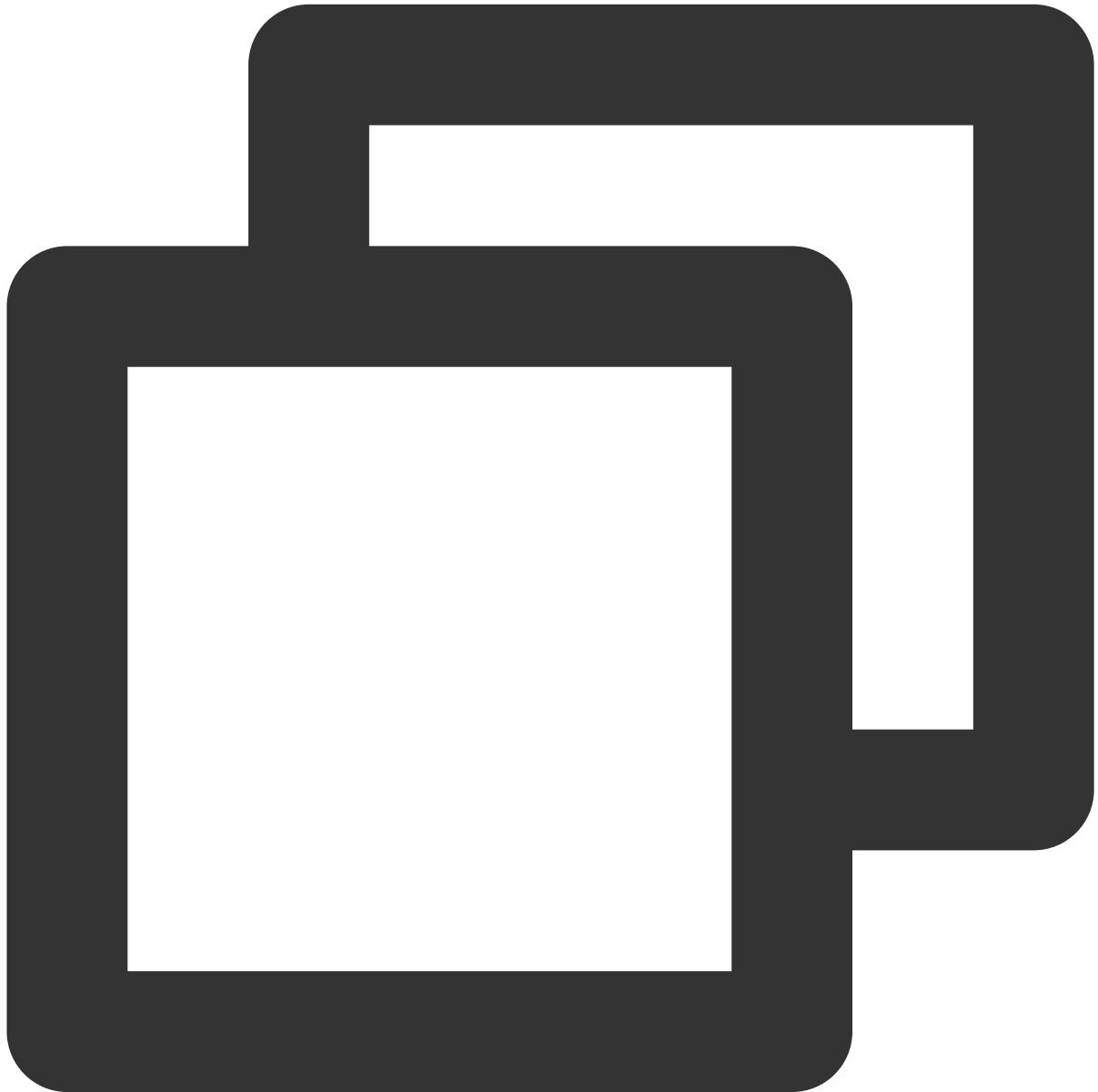
Return the reference table for the meanings of error codes:

Error code	Meaning
-1	Unknown error.
-100	3D engine resource initialization failed.
-200	GAN materials are not supported.
-300	This device does not support this material component.
-400	The template JSON content is empty.
-500	The SDK version is too low.
-600	Splitting is not supported.
-700	OpenGL is not supported.
-800	Scripting is not supported.
5000	The resolution of the split background image exceeds 2160x3840.
5001	Insufficient memory required to segment the background image.
5002	Failed to parse the video segmentation of the background.
5003	Background video segment exceeds 200 seconds.
5004	Background video segment format unsupported.

5005	Background image segment possesses rotation angle
------	---

updateProperty,updateProperties (This interface has been deprecated)

Modifies either a single beauty effect or multiple beauty effects, dynamic effects, filters in bulk, and can be invoked from any thread.



```
void updateProperty(XmagicProperty<?> p)
void updateProperties(List<XmagicProperty<?>> properties)
```

Parameter

Parameter	Meaning
XmagicProperty<?>p	<p>Tencent Effect data entity class.</p> <p>Taking "Skin Smoothing" as an example, an instance can be created as follows: <code>new XmagicProperty<>(Category.BEAUTY, null, null, BeautyConstant.BEAUTY_SMOOTH, new XmagicPropertyValues(0, 100, 50, 0, 1));</code></p> <p>To take '2D Animated Effect Bunny Paste' as an example, you can instantiate a new instance in the following manner: <code>new XmagicProperty<>(Category.MOTION, "video_tutujiang", "path of the effect file", null, null);</code></p> <p>If you want a certain animation/beauty/masking material to be overlaid on the current material, set the <code>mergeWithCurrentMotion</code> of that material's <code>XmagicProperty</code> object to <code>true</code>. For a detailed explanation of material overlay, see Material Overlay.</p> <p>For more examples, please refer to the configuration information in the <code>assets/beauty_panel</code> folder of the Demo project.</p>

XmagicProperty ([beauty parameter description](#), this interface has been deprecated)

Beauty filter

Attribute Field	Description
category	Category.BEAUTY
ID	<p>null</p> <p>Special case:</p> <p>The respective ID values for Natural, Goddess, and Handsome in the Face Slimming feature are:</p> <p><code>BeautyConstant.BEAUTY_FACE_NATURE_ID,</code> <code>BeautyConstant.BEAUTY_FACE_FEMALE_GOD_ID,</code> <code>BeautyConstant.BEAUTY_FACE_MALE_GOD_ID</code></p> <p>The ID value in the lipstick is:</p> <p><code>XmagicConstant.BeautyConstant.BEAUTY_LIPS_LIPS_MASK</code></p> <p>The ID value in the blush</p> <p>is: <code>XmagicConstant.BeautyConstant.BEAUTY_MAKEUP_MULTIPLY_MULTIPLY_MASK</code></p> <p>The ID value in 3D</p> <p>is: <code>XmagicConstant.BeautyConstant.BEAUTY_SOFTLIGHT_SOFTLIGHT_MASK</code></p>
resPath	<p>null</p> <p>Special Case: Lipstick, blush, and three-dimensional resPath represent the paths of resource images. For details, please refer to the <code>`assets/beauty_panel/advanced_beauty.json`</code> file in the Demo.</p>

effkey	Mandatory, refer to the Demo Example: Brightening BeautyConstant.BEAUTY_WHITEN
effValue	Mandatory, refer to the assets/beauty_panel/advanced_beauty.json file in the Demo, the demo project parses this file to construct an XmagicPropertyValues object, for XmagicPropertyValues attribute values, see beauty parameter description

Body beautification

Attribute Field	Description
category	Category.BODY_BEAUTY
ID	null
resPath	null
effkey	Mandatory, refer to the Demo Example: Long-legged BeautyConstant.BODY_LEG_STRETCH
effValue	Mandatory, refer to the assets/beauty_panel/beauty_body.json file in the Demo. In the demo project, parse this file to construct an XmagicPropertyValues object. The values of XmagicPropertyValues' properties can be seen in Beauty Parameters Explanation

Filters

Attribute Field	Description
category	Category.LUT
ID	Image name, required Sample: dongjing_lf.png The "none" ID corresponds to XmagicProperty.ID_NONE
resPath	Filter image path, mandatory, "none" setting as null
effkey	null
effValue	Mandatory, "None" Setting as null. It is an XmagicPropertyValues object, for the property values of XmagicPropertyValues see Beauty parameters explanation

Animated Effects

Attribute Field	Description
category	Category.MOTION

ID	Resource folder name, required Example: video_lianliancaomei The "none" ID corresponds to XmagicProperty.ID_NONE
resPath	Mandatory, refer to the Demo
effkey	null
effValue	null

Makeup

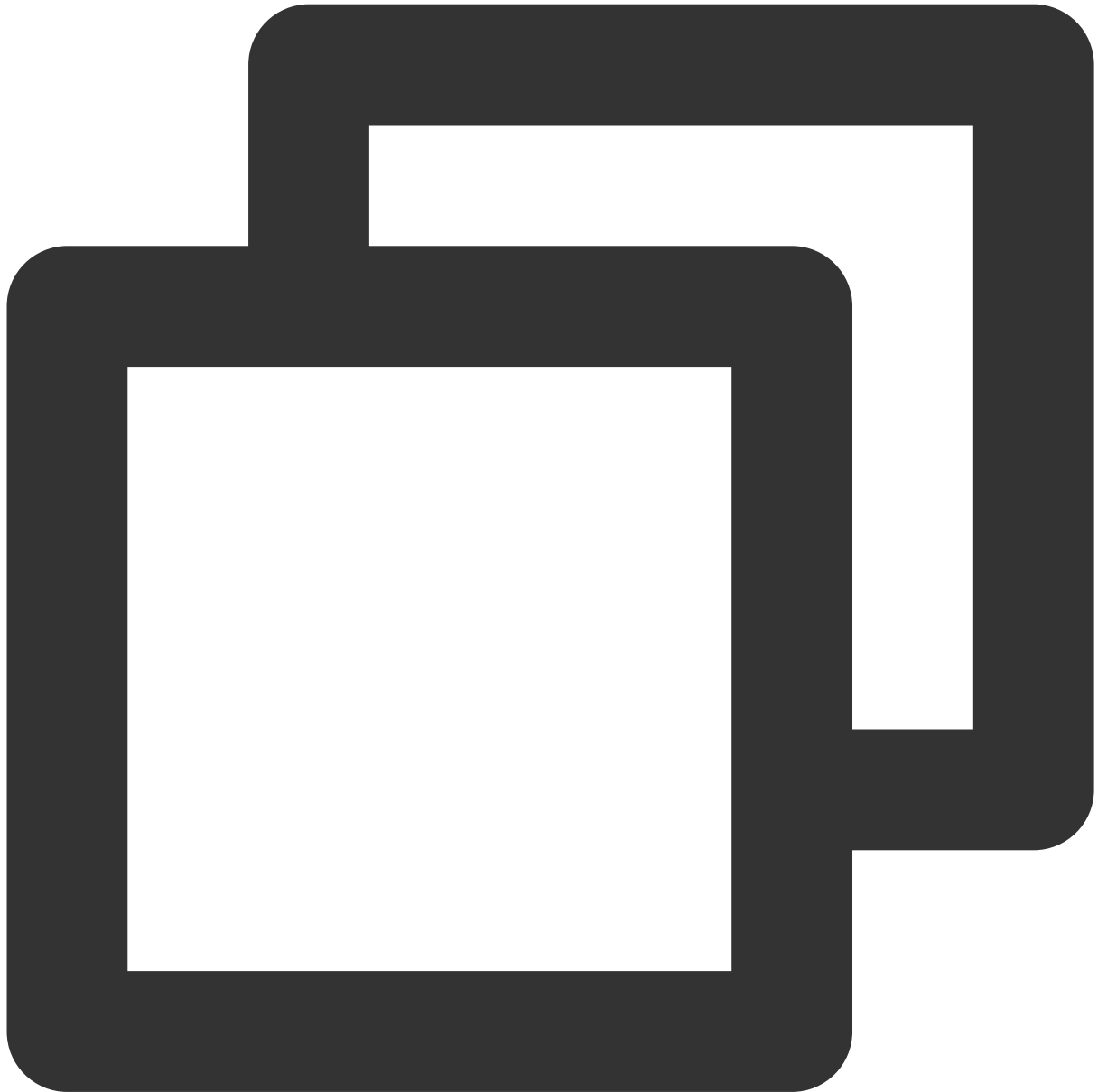
Attribute Field	Description
category	Category.MAKEUP
ID	Resource folder name, required Example: video_xuejiezhuang The "none" ID corresponds to XmagicProperty.ID_NONE
resPath	Mandatory, refer to the Demo
effkey	Mandatory. The value is: makeup.strength"None" Setting is null
effValue	Mandatory, "None" Setting is null. This is an XmagicPropertyValues object, for the values of various properties of XmagicPropertyValues, see Beauty Parameter Explanation

Keying

Attribute Field	Description
category	Category.SEGMENTATION
ID	Resource folder name, required Example: video_segmentation_blur_45 The "none" ID corresponds to XmagicProperty.ID_NONE From the Definition, the ID value must use: <code>XmagicConstant.SegmentationId.CUSTOM_SEG_ID</code>
resPath	Mandatory, refer to the Demo
effkey	null (excluding custom definition background), the value of the custom definition background is the selected resource path
effValue	null

setEffect(Added in V3.5.0)

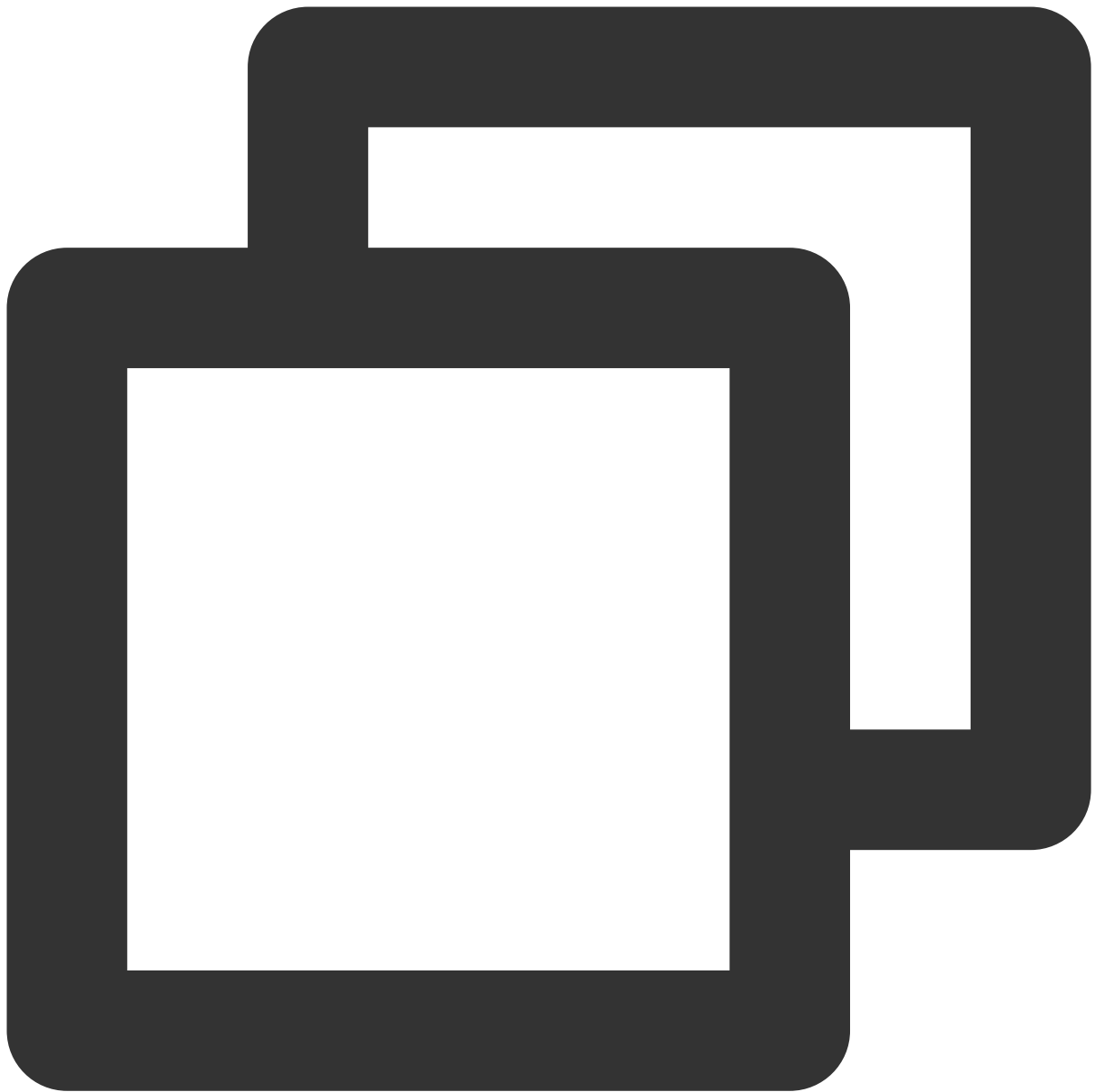
You can configure beautification, shaping, filtering, makeup, stickers, partitioning, and other effects on any thread. For specific parameters, please refer to [Beautification Parameters Explanation](#).



```
void setEffect(String effectName, int effectValue, String resourcePath, Map<String,
```

setTipsListener

Establish the callback for animated effect cues, utilized for displaying prompts onto the frontend page. For instance, certain materials might instruct users to nod their heads, extend their palms, or make a heart shape.

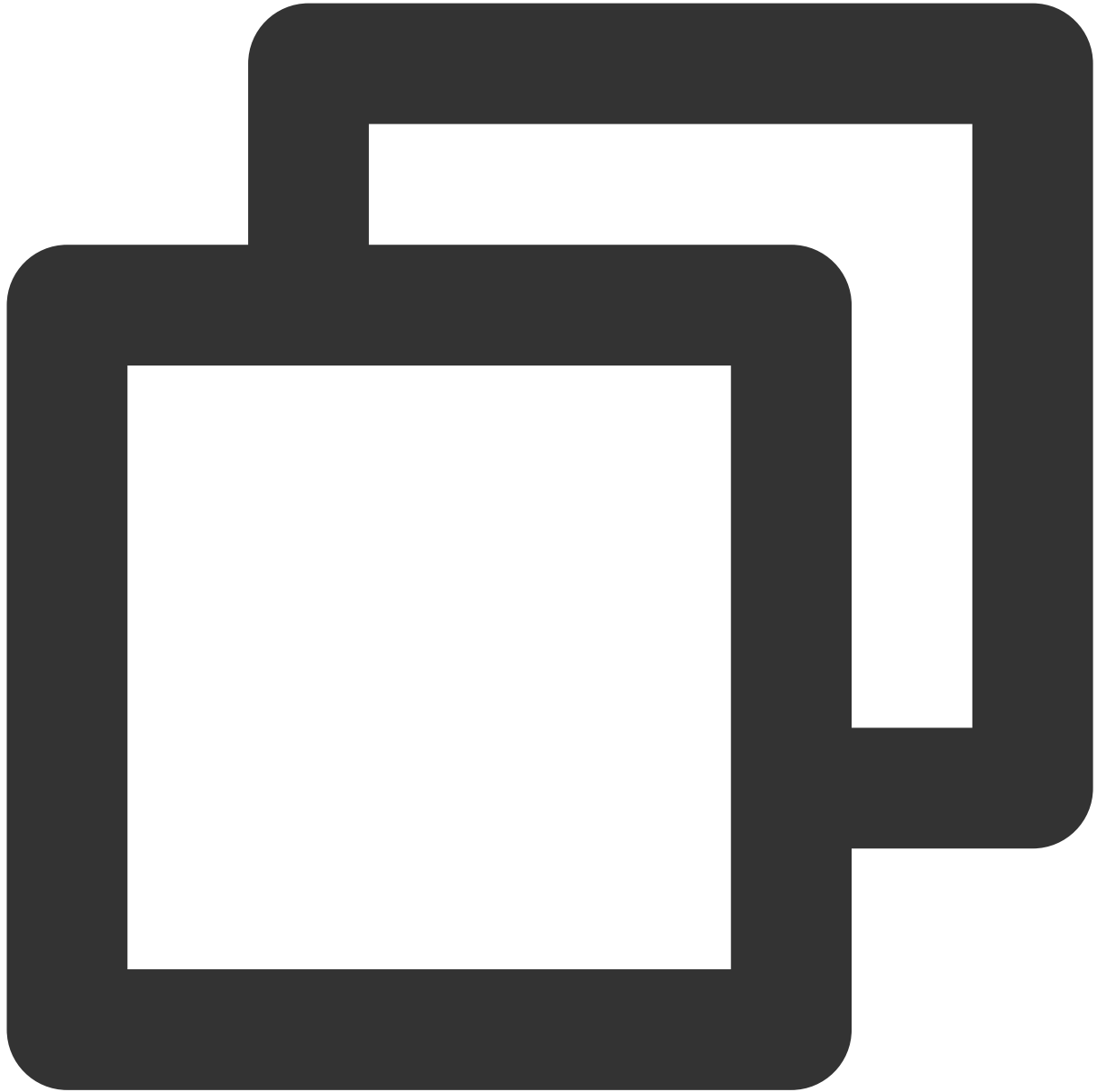


```
void setTipsListener(XmagicApi.XmagicTipsListener effectTipsListener)
```

Parameter

Parameter	Meaning
XmagicApi.XmagicTipsListener effectTipsListener	Implementation class of the callback function, callbacks are not necessarily executed in the main thread.

XmagicTipsListener Includes the following methods:

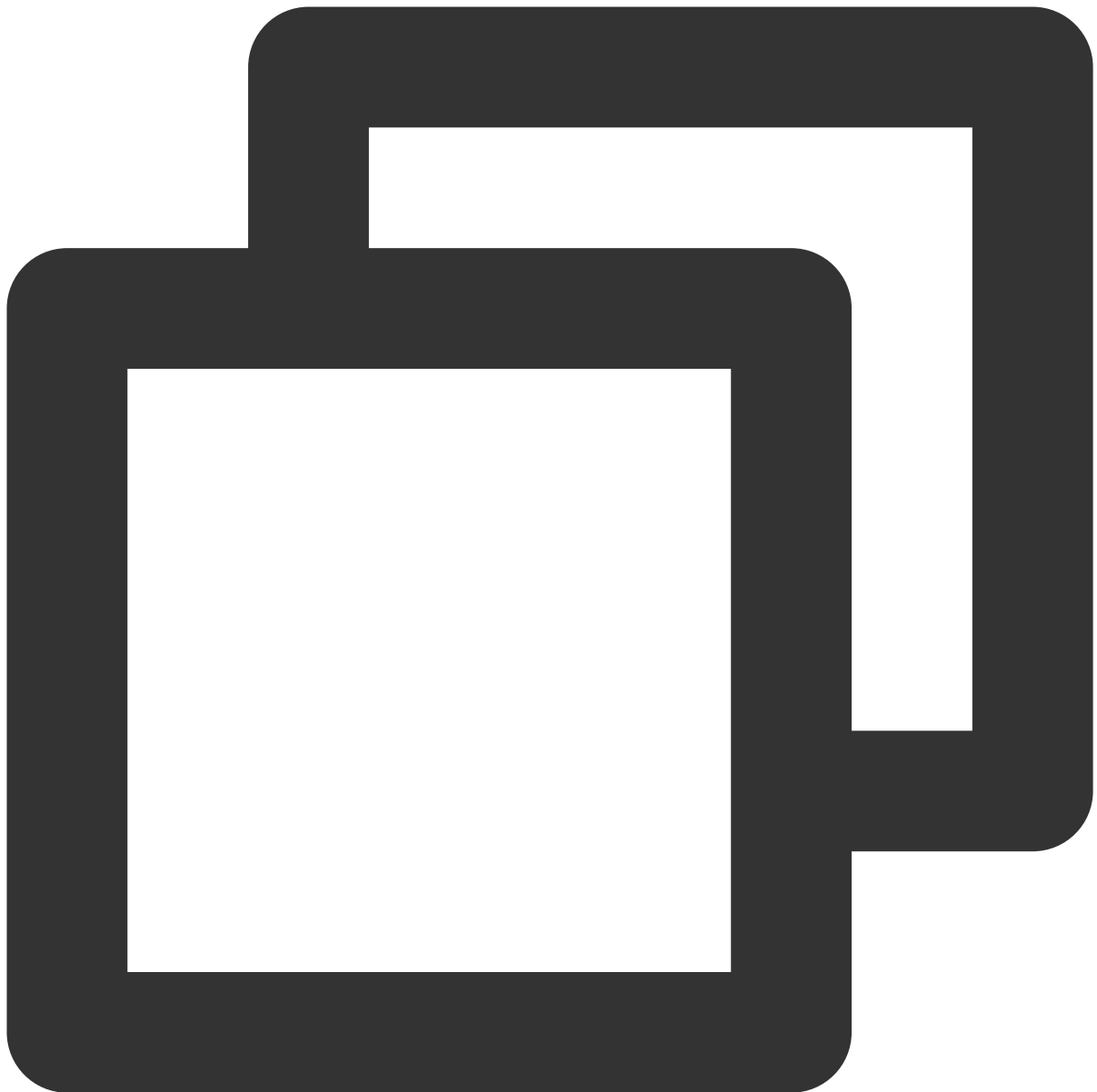


```
public interface XmagicTipsListener {  
  
    /**  
     * Display tips.  
     * @param tips The text information of the returned tips.  
     * @param tipsIcon The file path of the tips' icon, which can be used to resolve  
     *                  Note: tipsIcon may be empty if it is not configured in the  
     * @param type The category of tips. 0 indicates that both tips and tipsIcon ha  
     * @param duration The duration of tips display in milliseconds.  
     */  
}
```

```
    */
    void tipsNeedShow(String tips, String tipsIcon, int type, int duration);

    /**
     * Hide tips.
     * @param tips The text information of the returned tips.
     * @param tipsIcon The file path of the tips' icon, which can be used to resolve
     * @param type The category of tips. 0 indicates that both tips and tipsIcon ha
     */
    void tipsNeedHide(String tips, String tipsIcon, int type);
}
```

The example code is as follows :



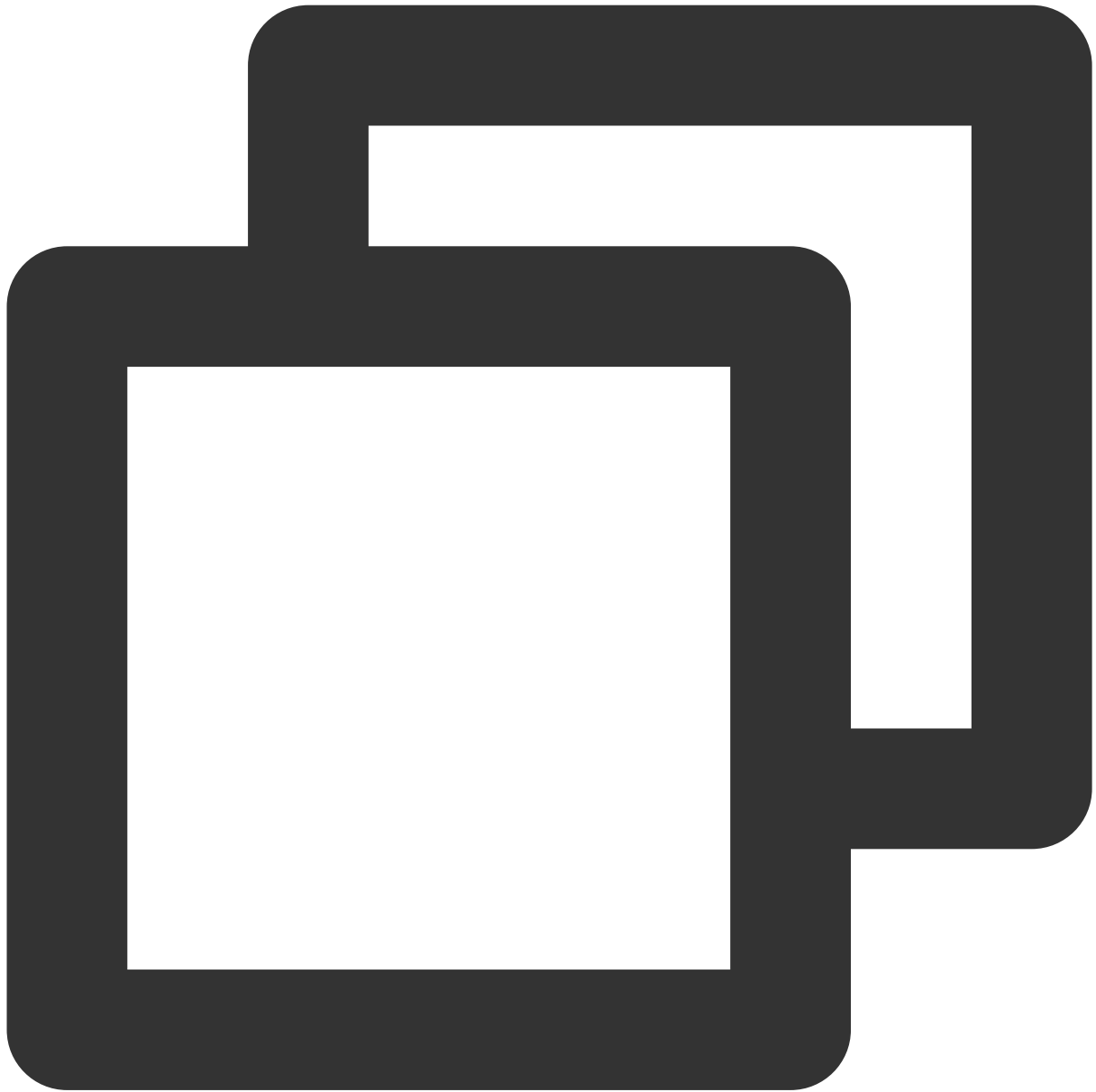
```
public void tipsNeedShow(final String tips, String tipsIcon, int type, int duration) {
    final int tipsType = type;
    final String tipsIconPath = tipsIcon;
    mMainHandler.post(new Runnable() {
        @Override
        public void run() {
            if (tipsType == 0) {
                if (!TextUtils.isEmpty(tipsIconPath) && !mImageTipsIsShow) {
                    mTipsImageView.setVisibility(View.VISIBLE);
                    Bitmap bitmap = BitmapUtils.decodeSampleBitmap(AEModule.getCont
                    mTipsImageView.setImageBitmap(bitmap);
                }
            }
        }
    });
}
```

```
        mImageTipsIsShow = true;
    } else {
        mTipsImageView.setVisibility(View.GONE);
    }
    mTipsTextView.setText(tips);
} else {
    pagFile = PAGFile.Load(tipsIconPath);
    tipsPAGView.post(new Runnable() {@Override
        public void run() {
            if (pagFile != null) {
                tipsPAGView.setRepeatCount(-1);
                tipsPAGView.setComposition(pagFile);
                tipsPAGView.setProgress(0);
                tipsPAGView.play();
                tipsPAGView.setVisibility(View.VISIBLE);
            }
        }
    });
}
});
}

public void tipsNeedHide(String tips, String tipsIcon, int type) {
    final int tipsType = type;
    final String tipsIconPath = tipsIcon;
    mMainThreadHandler.post(new Runnable() {
        @Override
        public void run() {
            if (tipsType == 0) {
                mTipsContainer.setVisibility(View.GONE);
                mImageTipsIsShow = false;
            } else {
                tipsPAGView.post(new Runnable() {
                    @Override
                    public void run() {
                        tipsPAGView.stop();
                        tipsPAGView.setVisibility(View.GONE);
                    }
                });
            }
        }
    });
}
```


setYTDataListener (this interface was removed in version 3.0.0, the feature has been migrated to the **onAIDataUpdated** method in **XmagicAIDataListener**)

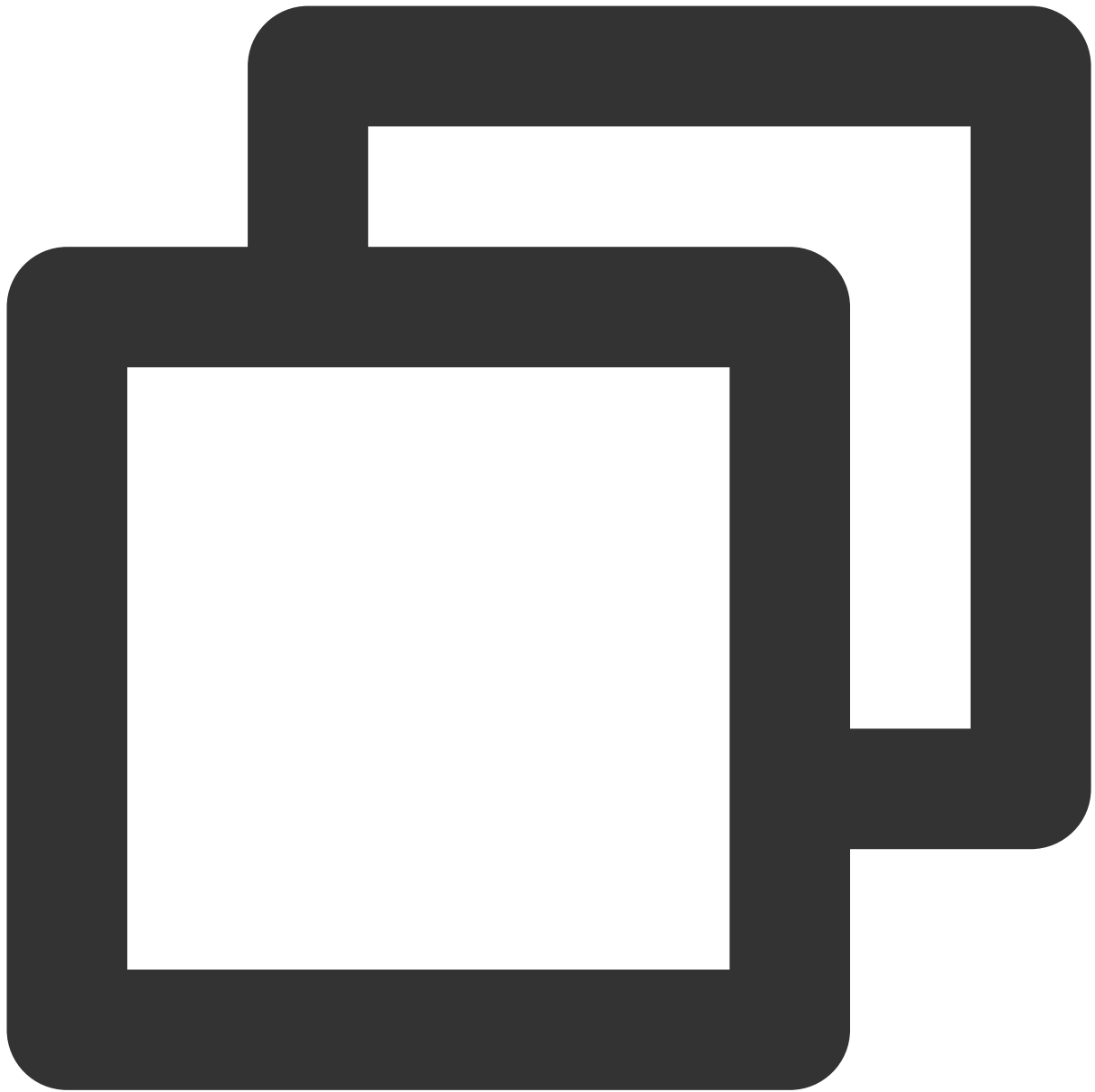
Configure callbacks for facial landmark information and other data.



```
void setYTDataListener(XmagicApi.XmagicYTDataListener ytDataListener)  
Configuring the callback for face information and other data
```

```
public interface XmagicYTDataListener {  
    void onYTDataUpdate(String data)  
}
```

onYTDataUpdate returns a JSON string structure, providing a maximum of five facial information:



```
{
  "face_info": [{
    "trace_id": 5,
    "face_256_point": [
      180.0,
      112.2,
      ...
    ],
    "face_256_visible": [
```

```

    0.85,
    ...
  ],
  "out_of_screen":true,
  "left_eye_high_vis_ratio":1.0,
  "right_eye_high_vis_ratio":1.0,
  "left_eyebrow_high_vis_ratio":1.0,
  "right_eyebrow_high_vis_ratio":1.0,
  "mouth_high_vis_ratio":1.0
},
...
]
}

```

Field Description

Field	Type	Range	Description
trace_id	int	[1,INF)	Face ID. The same ID points to the same face in the process of continuous stream fetching.
face_256_point	float	[0,screenWidth] or [0,screenHeight]	512 values in total for 256 facial keypoints. (0,0) is the top-left corner of the screen.
face_256_visible	float	[0,1]	Visibility of the 256 facial keypoints.
out_of_screen	bool	true/false	Whether the face is out of the screen.
left_eye_high_vis_ratio	float	[0,1]	Percentage of the highly visible points of the left eye.
right_eye_high_vis_ratio	float	[0,1]	Percentage of the highly visible points of the right eye.
left_eyebrow_high_vis_ratio	float	[0,1]	Percentage of the highly visible points of the left eyebrow.
right_eyebrow_high_vis_ratio	float	[0,1]	Percentage of the highly visible points of the right eyebrow.
mouth_high_vis_ratio	float	[0,1]	Percentage of the highly visible points of the mouth.

Parameter

Parameter	Meaning

XmagicApi.XmagicYTDataListener ytDataListener	Callback function implementation class.
--	---

setAIDataListener

Upon detection of faces, bodies, and gestures, the spatial information of these parts will be returned via a callback:

onFaceDataUpdated: Once beauty mode is enabled, there will be a callback. When a face is detected, a List<FaceData> is returned. If no face is detected, an empty List is returned.

onHandDataUpdated: A callback is established when gestures are recognized after setting the gesture animation. No callbacks occur under any other circumstances.

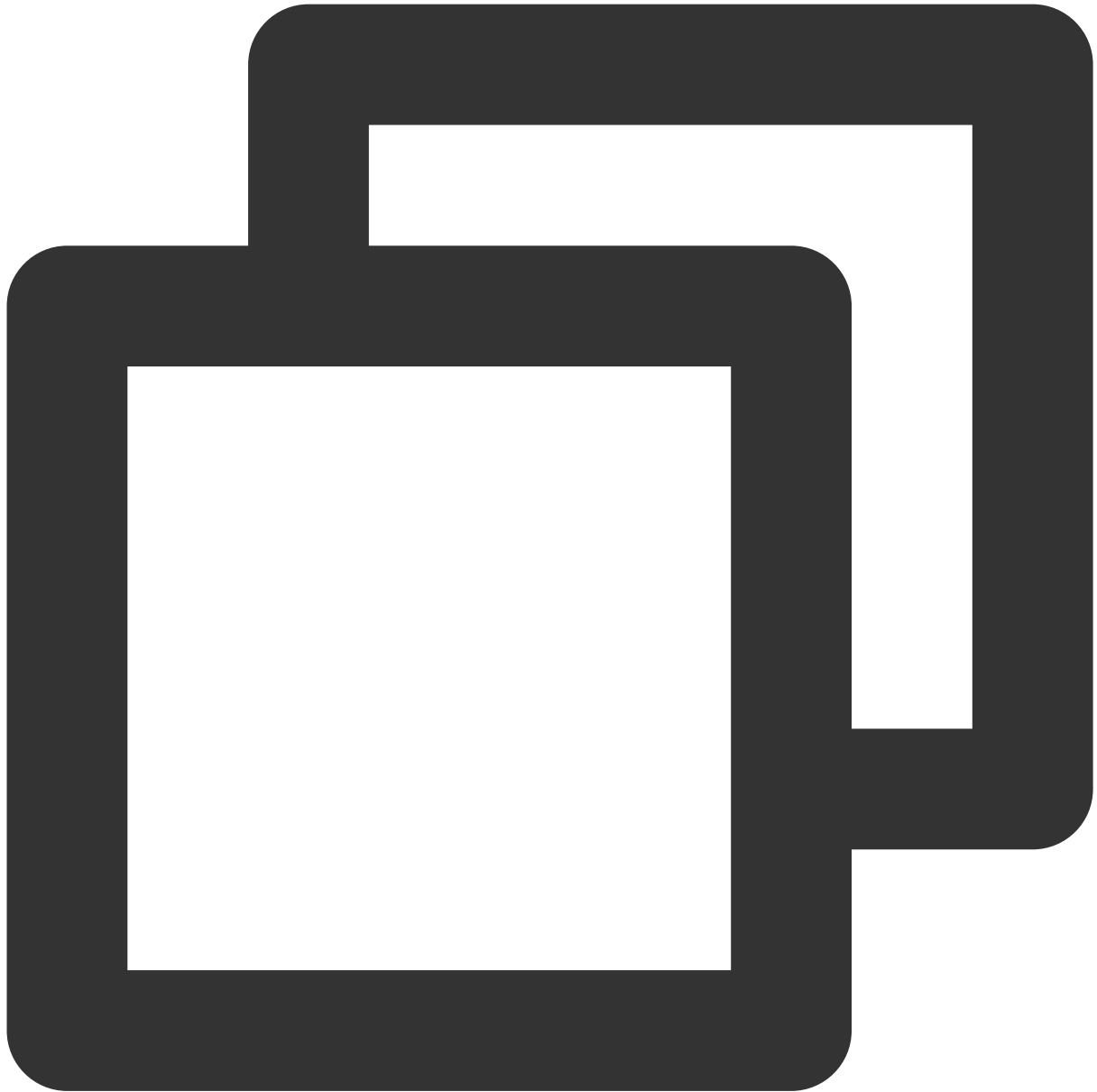
onBodyDataUpdated: Callback when the body shaping properties are set and the body is recognized. It does not callback in other situations.



```
public interface OnAIDataListener {  
  
    void onFaceDataUpdated(List<TEFaceData> faceDataList);  
    void onHandDataUpdated(List<TEHandData> handDataList);  
    void onBodyDataUpdated(List<TEBodyData> bodyDataList);  
  
    void onAIDataUpdated(String jsonString);    // This method is a new addition in  
}
```

onPause

Pause rendering, which can be bound with Activity's onPause lifetime affinity. Currently, only `onPauseAudio` is called internally.



```
void onPause()
```

onResume

Resume rendering, can be paired with Activity onResume lifetime affinity.



```
void onResume()
```

onDestroy

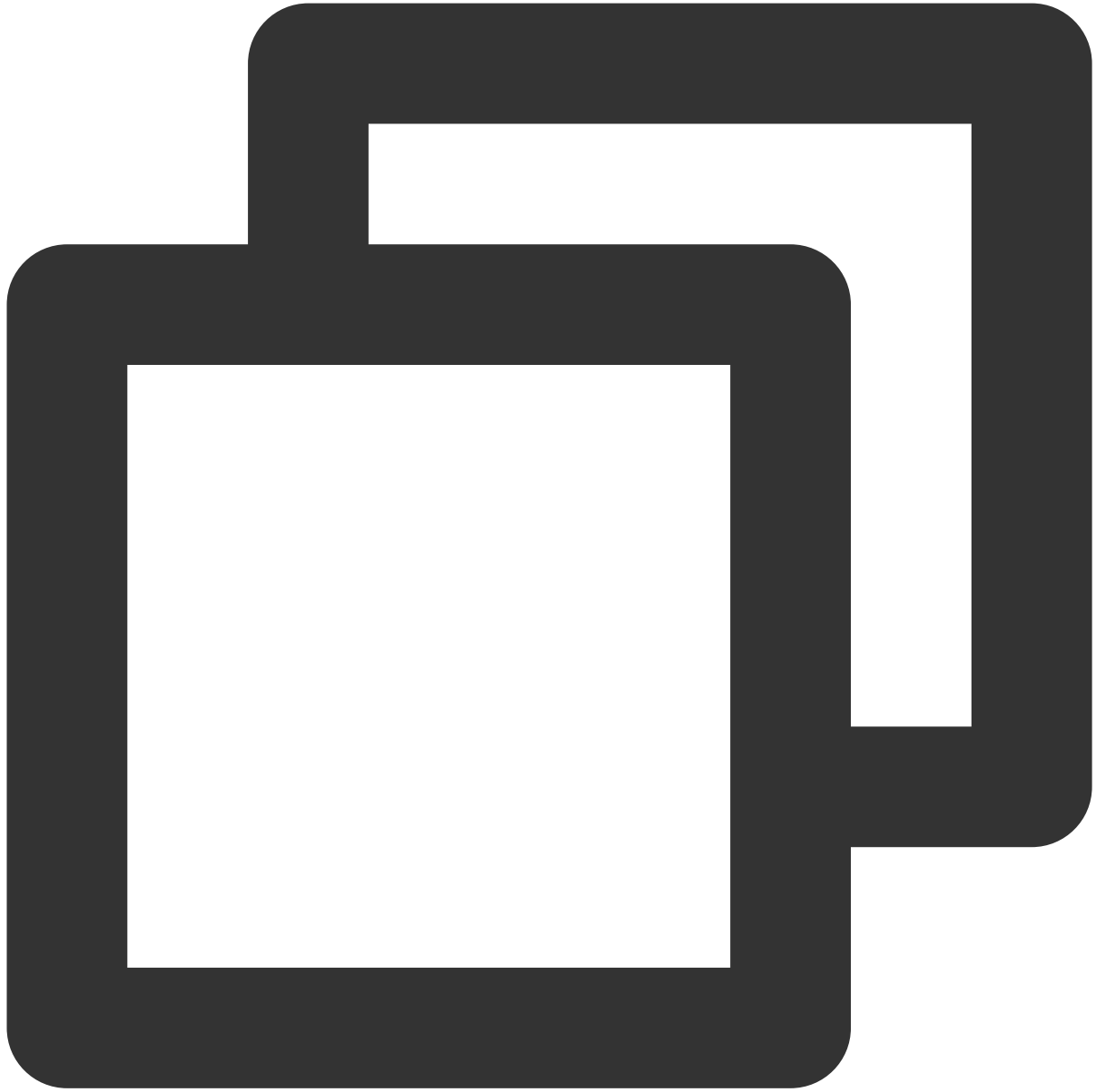
Clears GL thread resources and needs to be called within the GL thread. Sample code:



```
// Refer to the sample code in `TECameraBaseActivity.java`  
public void onGLContextDestroy() {  
    if (this.mXMagicApi != null) {  
        this.mXMagicApi.onDestroy();  
        this.mXMagicApi = null;  
    }  
}
```

process

The method for SDK rendering to accept data can be used within the camera data callback function.



```
// Render the texture  
int process(int srcTextureId, int srcTextureWidth, int srcTextureHeight)  
// Render the bitmap  
Bitmap process(Bitmap bitmap, boolean needReset)
```

Parameter

Parameter	Meaning

int srcTextureId	The texture that needs to be rendered. The type is: OpenGL 2D texture format, and the pixel format is RGBA.
id int srcTextureWidth	Width of the texture that needs to be rendered.
int srcTextureHeight	Height of the texture that needs to be rendered.
Bitmap bitmap	The recommended maximum size is 2160 x 4096. Larger images have poor face recognition results or cannot get faces recognized and are likely to cause OOM problems. Shrink such images first before passing them in.
boolean needReset	Switch the image. First time using partition. Initial use of animated effects. First-time usage of makeup. For these scenarios, the needReset Setting should be set to `true`.

onPauseAudio (this interface has been deprecated)

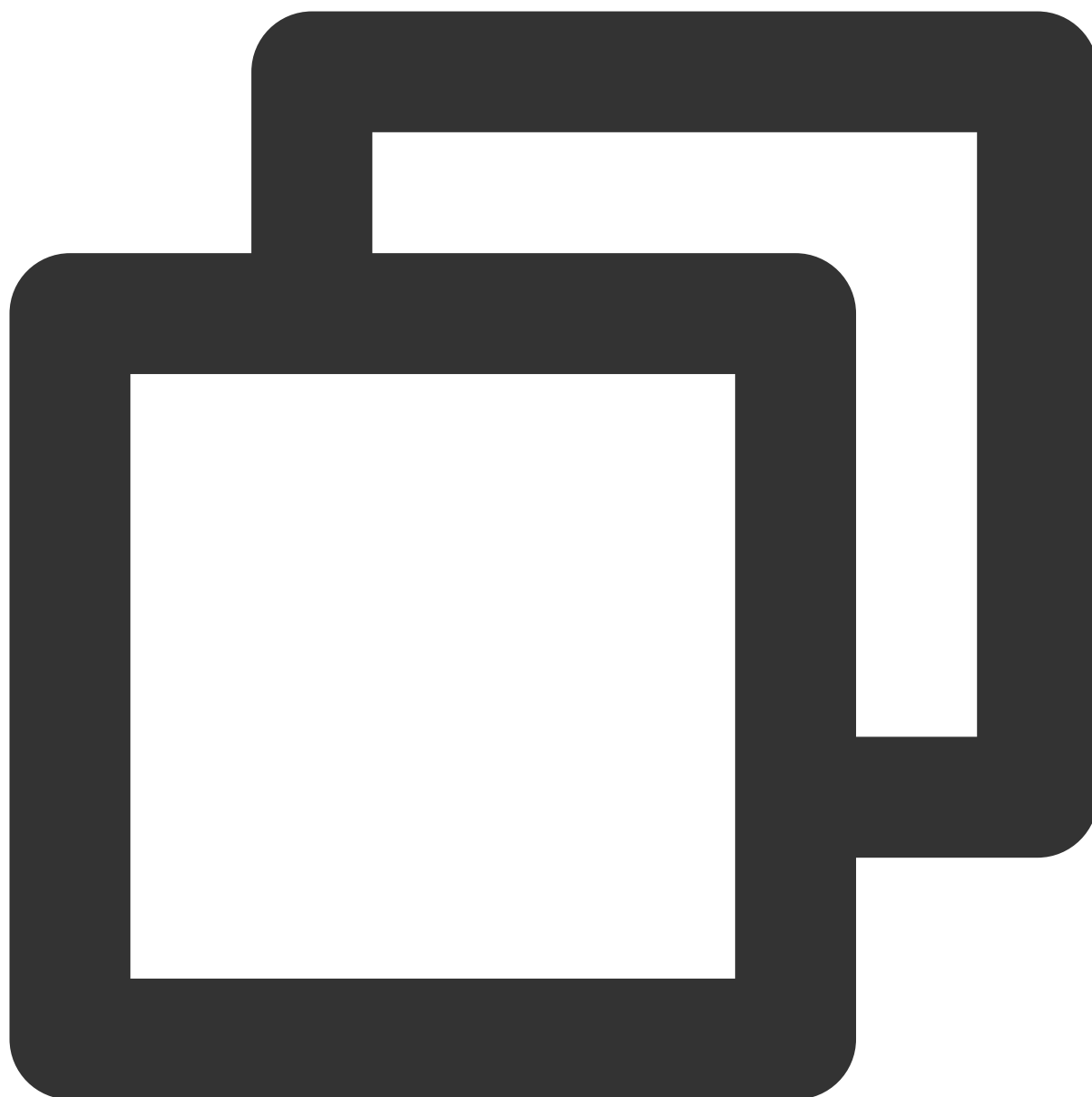
Invoke this function when only the cessation of the audio is necessitated, without the need to release the GL thread.



```
void onPauseAudio()
```

sensorChanged

Used to determine the current rotation angle of the phone, so as to adjust the basis for AI to recognize the angle of the face. This needs to be called in the callback function of the G-sensor.



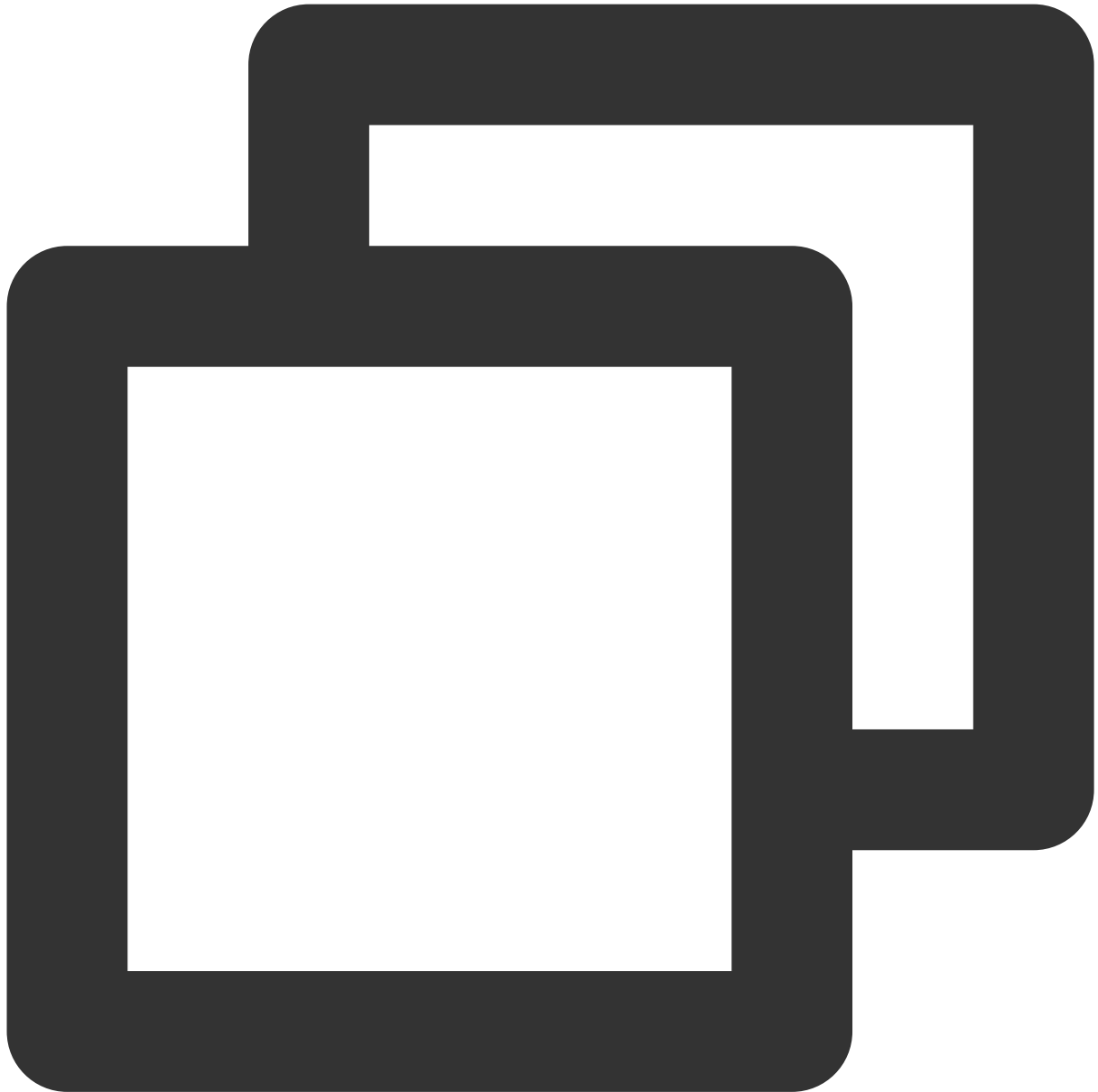
```
void sensorChanged(SensorEvent event, Sensor accelerometer)
```

Parameter

Parameter	Meaning
SensorEvent event	Event entity class returned by the gyroscope sensor callback function <code>onSensorChanged</code> .
Sensor accelerometer	Sample G-sensor.

isDeviceSupport

version 3.5.0 and later

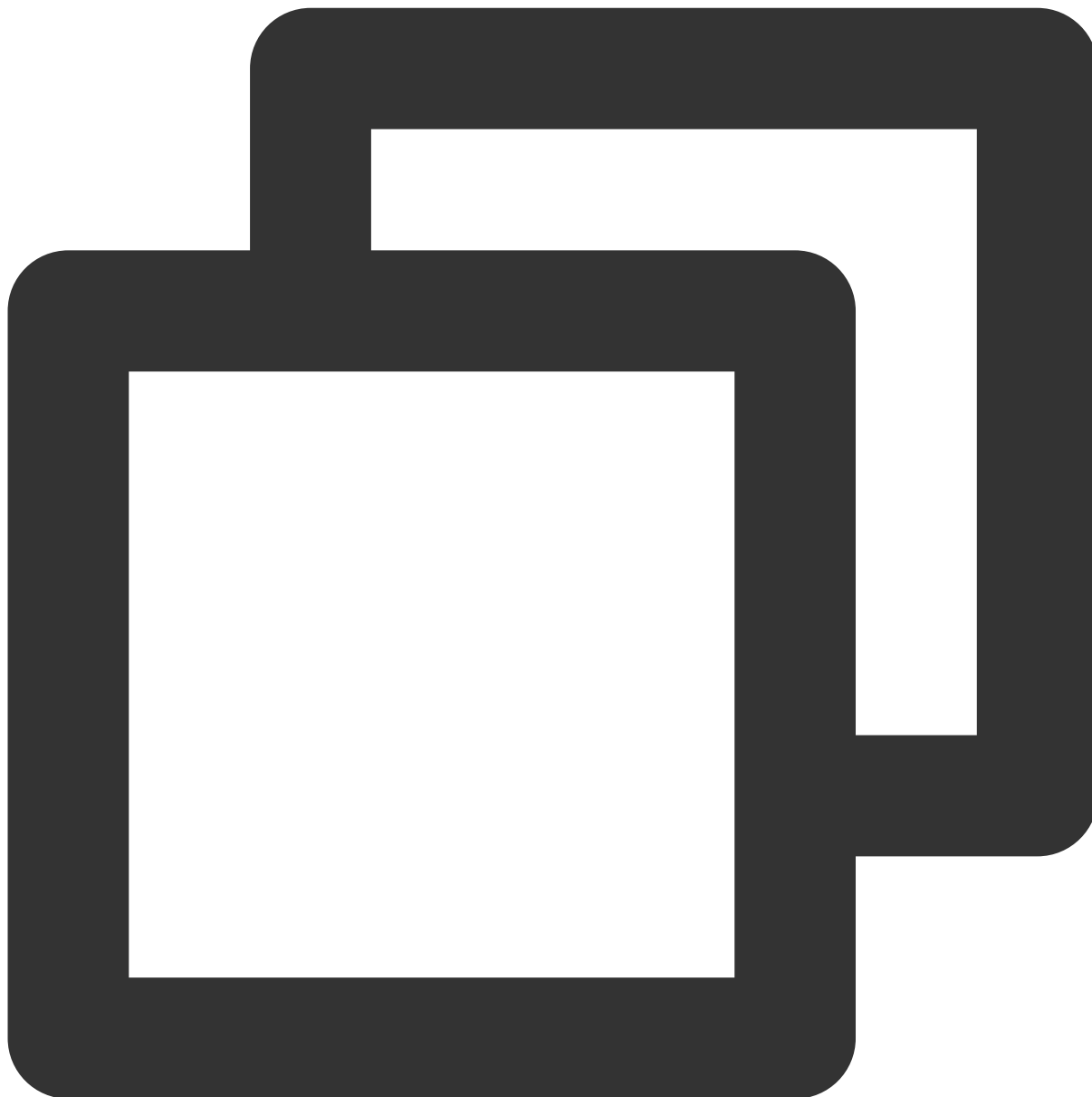


```
/**
 * Check whether the current device supports this material
 * @param motionResPath The path of the material file
 * @return true means supported, false means not supported
 */
boolean isDeviceSupport(String motionResPath)
```

version 3.3.0 and earlier

Transmit the list of animated effect resources into the SDK for verification. After execution, the

`XmagicProperty.isSupport` field signifies if the resource is serviceable. Based on `XmagicProperty.isSupport`, click control can be enacted at the UI level, or it can be directly expunged from the resource list.



```
void isDeviceSupport(List<XmagicProperty<?>> assetsList)
```

Parameter

Parameter	Meaning
-----------	---------

List<XmagicProperty<?>> assetsList	List of animated effect materials to be checked.
---------------------------------------	--

getPropertyRequiredAbilities (this interface has been deprecated)

Inputs an animated effect resources list, returns the SDK atomic abilities list used for each resource.

The usage scenario of this method:

If you have purchased or created a number of animated effect materials, by invoking this method, it will return a list of atomic abilities required for each material. For instance, material 1 requires abilities A, B, C while material 2 needs abilities B, C, D. Subsequently, you maintain such a list of abilities on your server. Later on, when a user wants to download the animated effect materials from the server, the user first accesses the list of atomic abilities that his mobile has through the `getDeviceAbilities` method (for example, this phone possesses abilities A, B, C, but lacks D), transmits this abilities list to the server. The server, on determining the device does not have ability D, therefore does not issue material 2 to the user.

Parameter

Parameter	Meaning
List<XmagicProperty<?>> assets	List of animated effect resources for which to check the atomic capabilities.

Return

Returned value `Map<XmagicProperty<?>, ArrayList<String>>` :

key: animated effect resource material entity class.

value: list of used atomic capabilities.

getDeviceAbilities

It returns the atomic capability table that the current device supports. To be used in conjunction with the `getPropertyRequiredAbilities` method. For more details, please refer to the description of the `getPropertyRequiredAbilities`.



```
Map<String, Boolean> getDeviceAbilities()
```

Return

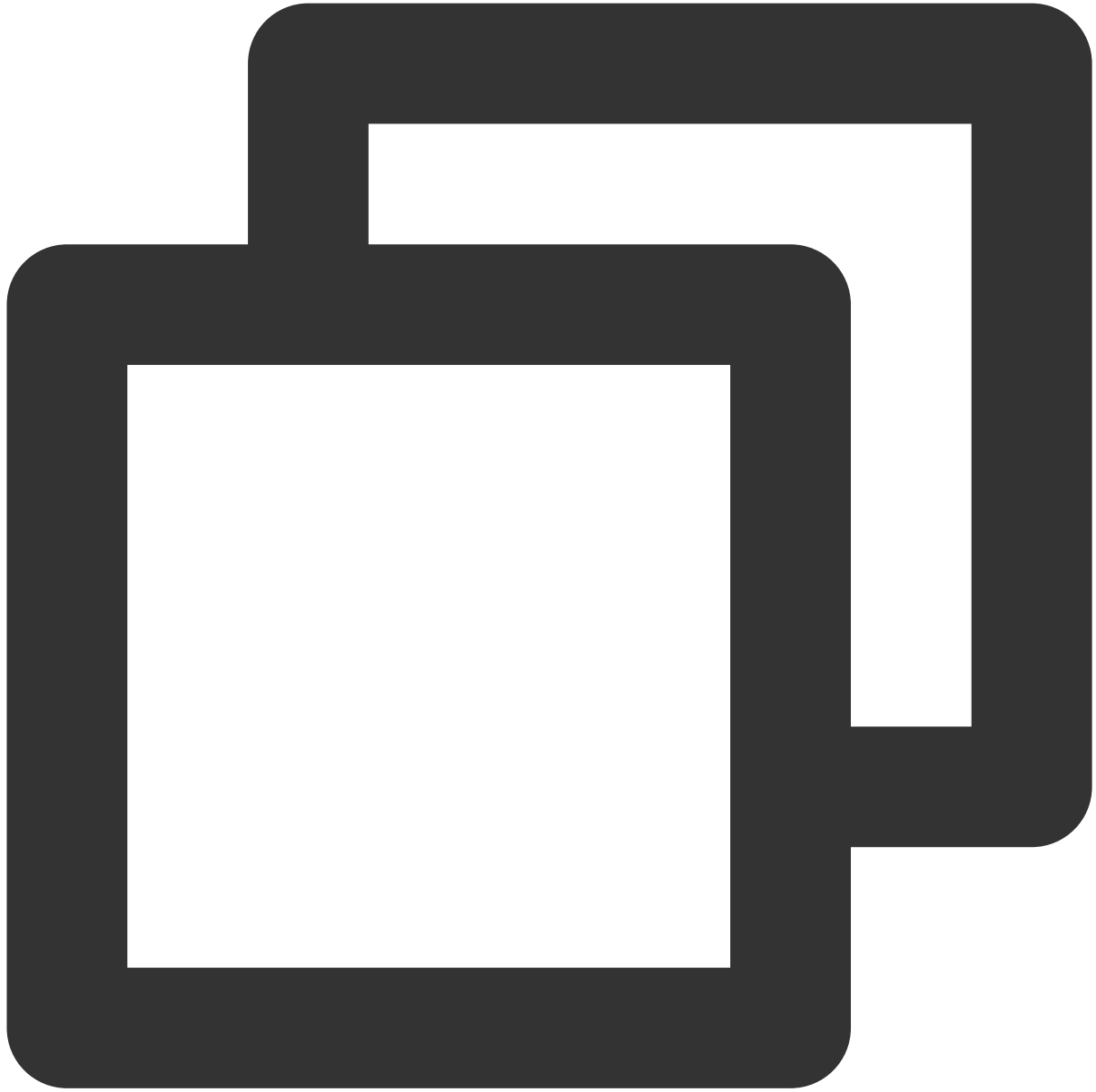
Return value `Map<String, Boolean>` :

key: atomic capability name (corresponding to the material capability name).

value: whether it is supported by the current device.

isSupportBeauty

Determine whether the current device supports refinement (OpenGL3.0).



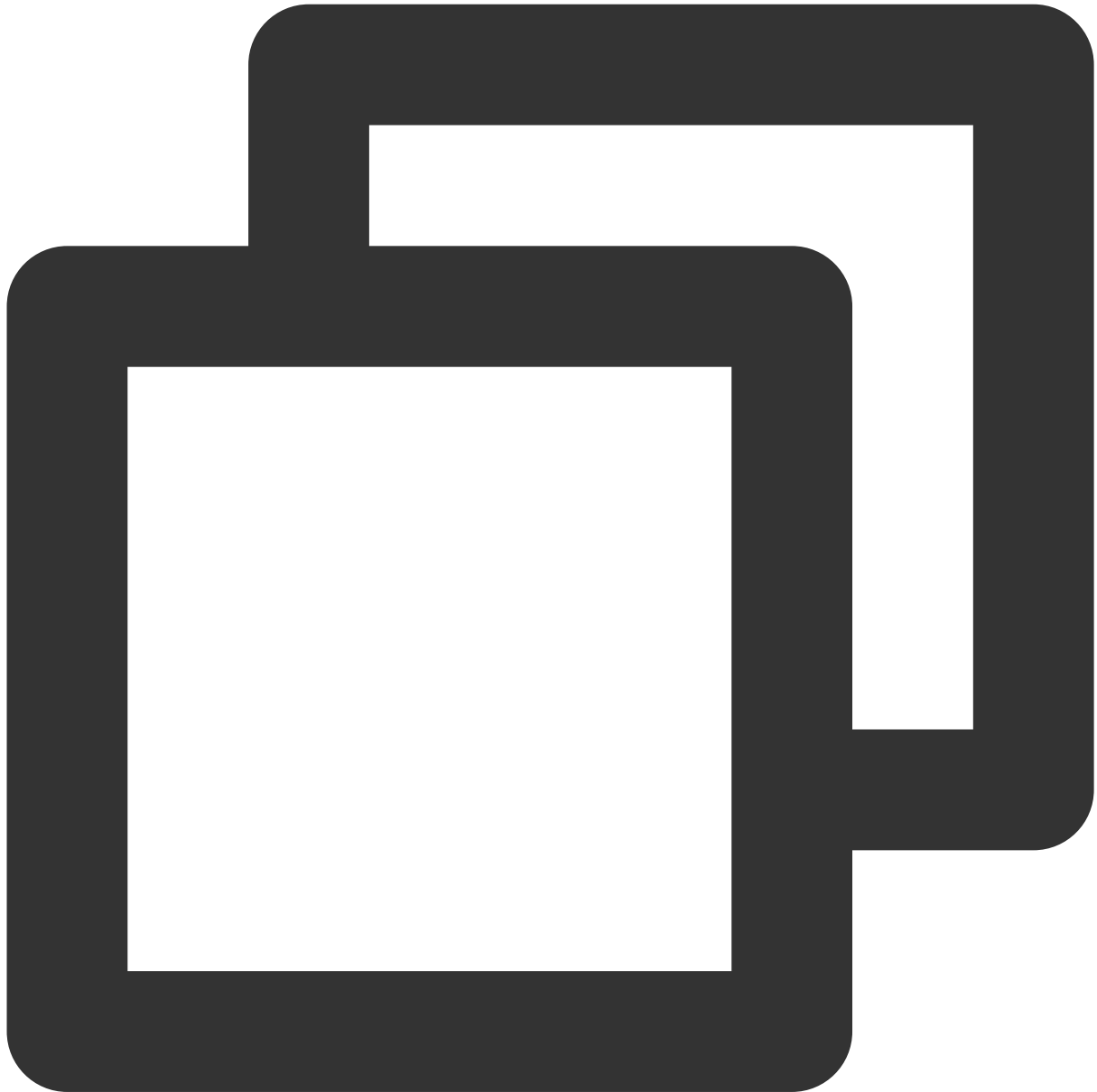
```
boolean isSupportBeauty()
```

Return

Return value boolean: Determines whether the beauty filter is supported.

isBeautyAuthorized

Determine which beauty or body modifier features the current License authorization supports. Only detection of BEAUTY and BODY_BEAUTY types are supported. The resultant evaluation will be allocated to each beauty object's `XmagicProperty.isAuth` field. If the `isAuth` field returns false, the entrances for these features can be concealed in the UI.



```
void isBeautyAuthorized(List<XmagicProperty<?>> properties)
```

Parameter

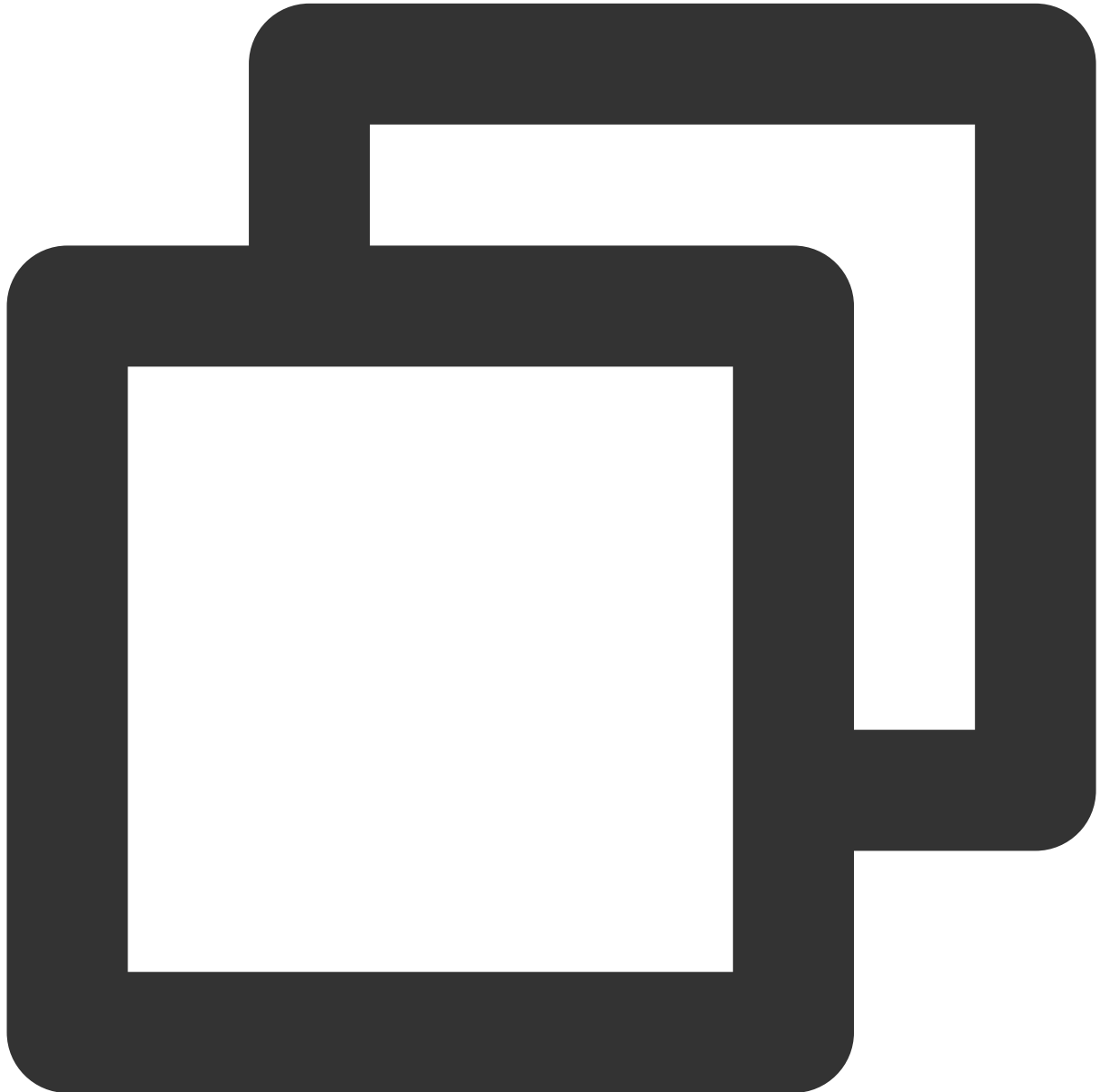
Parameter	Meaning
-----------	---------

```
List<XmagicProperty<?>>  
properties
```

Beauty filters that need to be checked.

setXmagicStreamType

Sets the input data type, which is Android camera data stream (XmagicApi.PROCESS_TYPE_CAMERA_STREAM) by default.



```
void setXmagicStreamType(int type)
```

Parameter

Parameter	Meaning
int type	Data source type, there are two options: <code>XmagicApi.PROCESS_TYPE_CAMERA_STREAM</code> : camera data source. <code>XmagicApi.PROCESS_TYPE_PICTURE_DATA</code> : Image data source.

setXmagicLogLevel

Setting the log level of the SDK, the default is `WARN`. During the development debugging stage if necessary, it can be set to `Log.DEBUG`. When officially released, it must be set to `Log.WARN` or `Log.ERROR` otherwise, an immense amount of logs will adversely affect performance.

Invoke after new `XmagicApi()`.

setAudioMute

Whether to mute when using animation materials (Added in V2.5.0):

Parameter: true indicates mute, false denotes non-mute.

enableEnhancedMode

Activate the beautification enhancing pattern (Added in V2.5.1). By default, it is not activated.

When inactive, the strength range of each beauty option in the application layer can be set from 0 to 1 or -1 to 1. If this range is exceeded, the SDK will take the boundary value. For instance, if the application layer sets the face-thinning to 1.2, the SDK determines that this surpasses the maximum value of 1.0, and internally adjusts the face-thinning value to 1.0.

After enabling the enhanced pattern, the application layer can set a broader range of values. For instance, if a greater face slimming effect is desired, you can set the slimming value to 1.2. The SDK will accept and use this value of 1.2 without correcting it to 1.0.

Upon activation of the enhanced pattern, it becomes the responsibility of the application layer to manage the maximal values of each beauty setting, allowing users to adjust within this range. We provide a set of reference values which you may adjust freely according to product requirements, though we advise not exceeding our recommended values to avoid potential degradation of beauty effects. The reference values are as follows:

Beauty Item Name	In enhanced pattern, recommended maximum value (magnification factor)
Whitening, shortening the face, V-face, eye distance, nose position, removal of laugh lines, lipstick, three-dimensional appearance	1.3
Eye lightening	1.5

Blush	1.8
Other	1.2

Material Overlapping (Added in 3.0.1.2)

If you wish to stack a certain dynamic effect, make-up or division material on the current material, then set the **XmagicProperty** object of said material 'mergeWithCurrentMotion' to true. For other property Settings see [Beauty Parameter Settings](#).

Points to Note on Material Overlay:

1. Clients are required to manage the compatibility of their materials for overlaying. Here are two examples:

Example 1: Effect A turns the face into an imperial concubine's visage, and Effect B transforms it into a fairytale-like appearance. The overlay of these two effects may result in an unnaturally distorted image.

Example 2: Effect A represents rabbit ears, while Effect B represents pig ears. When combined, two types of ears are presented.

Case 1 and Case 2 are not conducive to superimposition. If Effect A is a pair of rabbit ears, and Effect B is blowing a kiss, these two effects won't conflict and are hence suitable for superimposition.

2. Only the overlay of simple materials is supported. Simple materials refer to single animation capabilities, or single makeup effects, or single background removal, etc. Complex materials refer to those that contain multiple effects. There is no clear definition between simple and complex materials. It is recommended that customers thoroughly test and manage which materials can be overlaid and which cannot.

3. When combined, effects triggered by actions (such as stretching out a hand to trigger a certain effect, smiling to instigate a specific effect, and so on) are classified as complex effects. These need to be placed in the forefront, with simpler effects overlaid behind them.

4. Usage example: The anchor uses effect A, then the audience gifts effect B. Effect B is applied on top of effect A. After a period of time, effect B disappears, returning to effect A. The setting process is as follows:

4.1 Setting special effect A, mergeWithCurrentMotion Setting is false.

4.2 Setting special effect B, mergeWithCurrentMotion Setting is true.

4.3 After a brief period, proceed with Setting A, ensuring mergeWithCurrentMotion Setting is set to false.

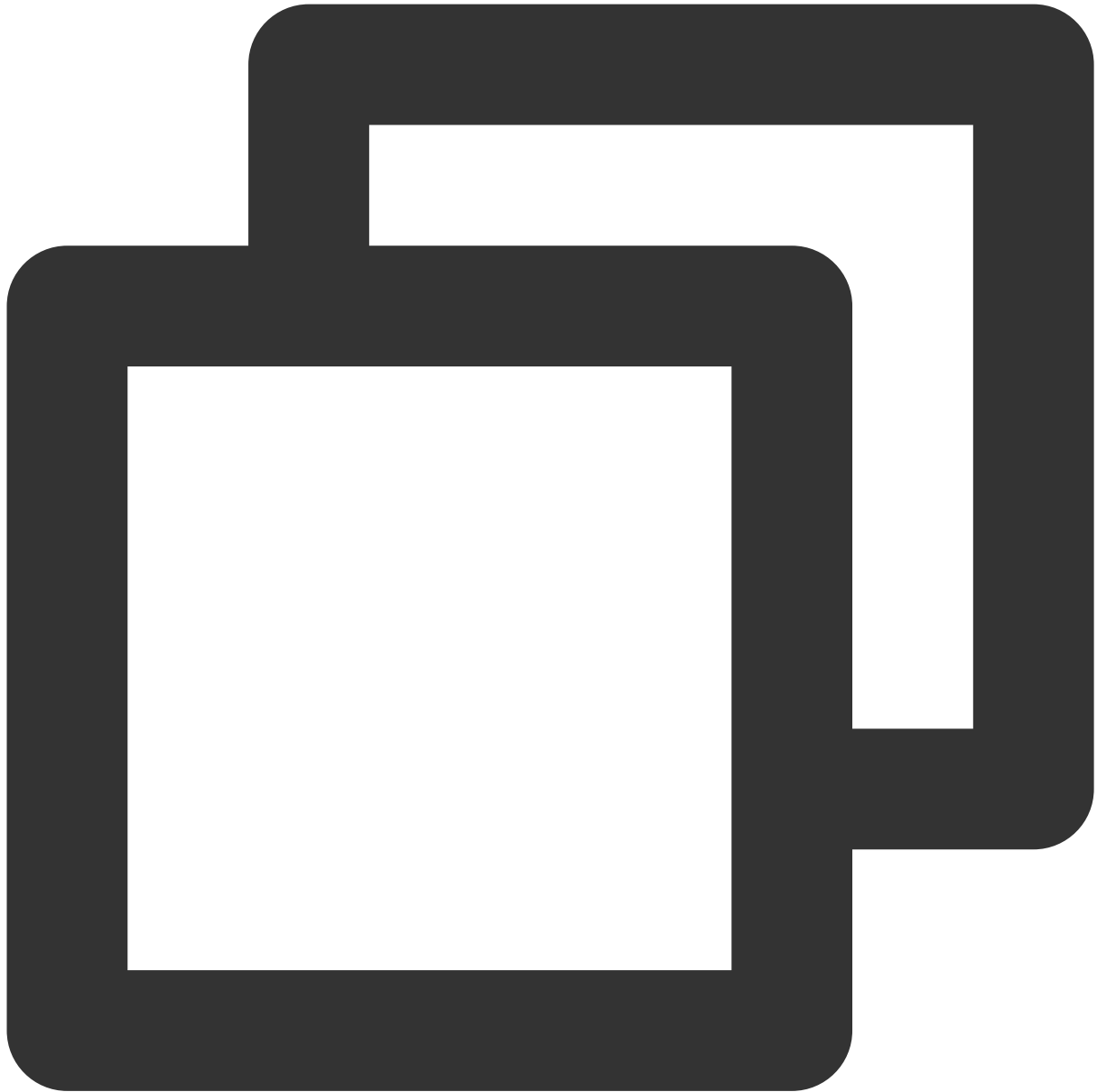
setDowngradePerformance (Added in V3.1.0)

Upon activating the advanced performance pattern, the resources consumed by the beautification function on your system's CPU/GPU are diminished, effectively mitigating the heating and lagging issues of your mobile device, making it particularly beneficial for extended utilization on lower-end devices.

Note: After enabling the high performance pattern, the following beauty options will be unavailable:

1. Ocular region: Eye width, eye height, removal of eye bags .
2. Eyebrows: Angle, Distance, Height, Length, Thickness, Eyebrow Peak.
3. Mouth: The Smiling Lips.

4. Facial: Slimming (natural, goddess, handsome), jaw reduction, wrinkle removal, and nasolabial fold removal. It is recommended to use "face shape" to achieve the comprehensive effect of large eyes and slim face.



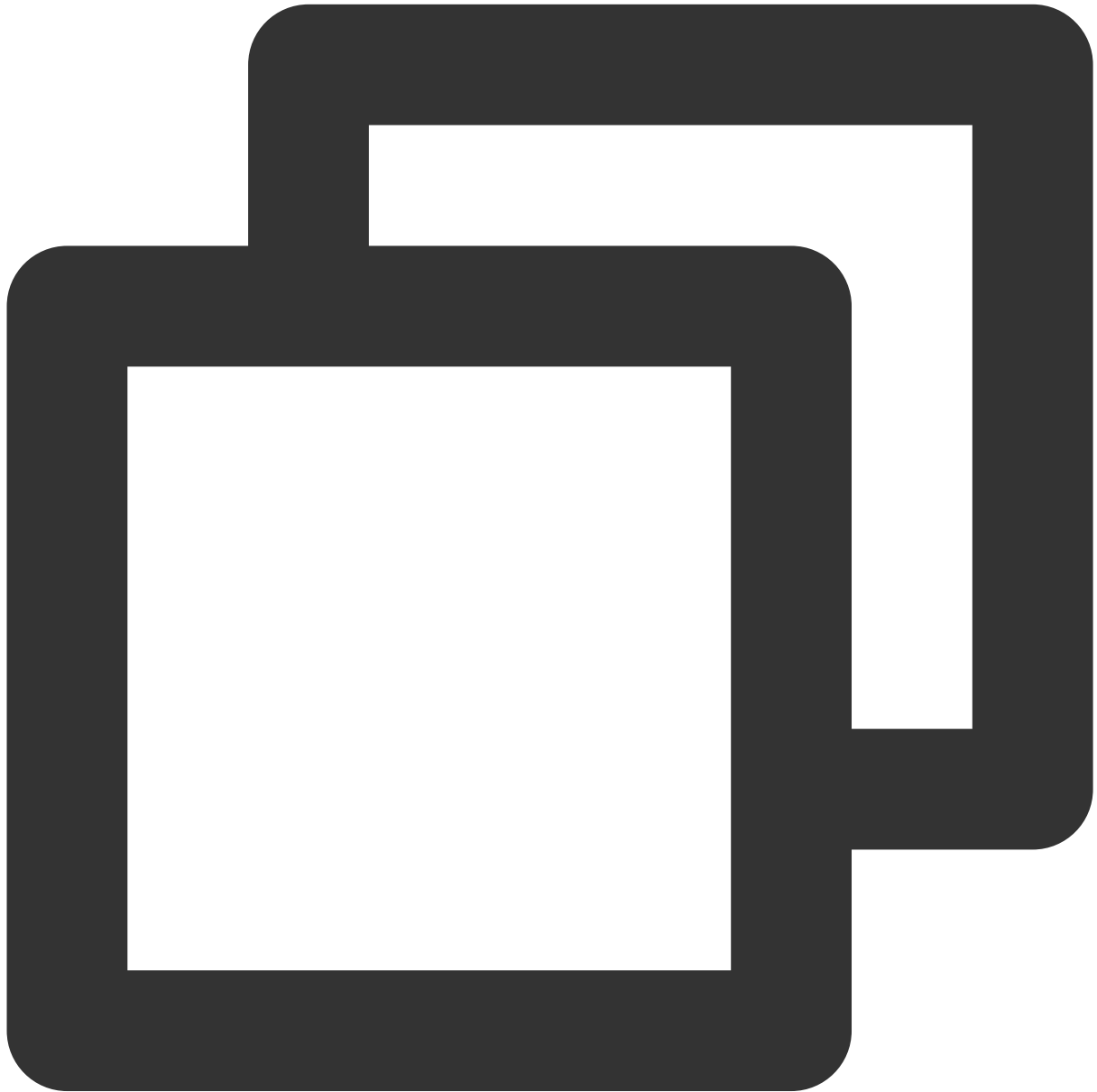
```
void boolean setDowngradePerformance()
```

enableEnhancedMode (Added in V3.7.0)

Upon activating the advanced performance pattern, the resources consumed by the beautification function on your system's CPU/GPU are diminished, effectively mitigating the heating and lagging issues of your mobile device, making it particularly beneficial for extended utilization on lower-end devices.

Note: After enabling the high performance pattern, the following beauty options will be unavailable:

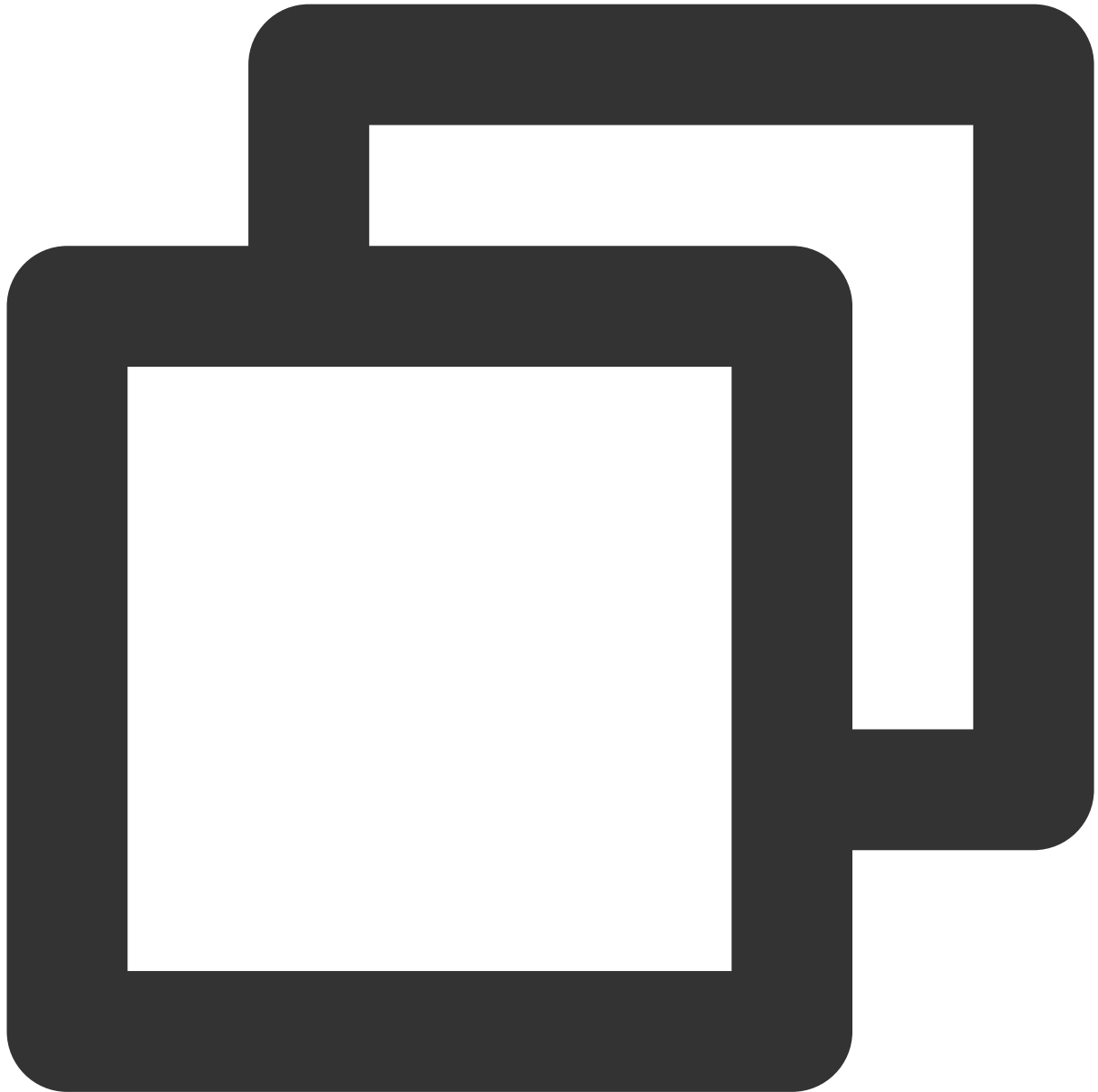
1. Ocular region: Eye width, eye height, removal of eye bags .
2. Eyebrows: Angle, Distance, Height, Length, Thickness, Eyebrow Peak.
3. Mouth: The Smiling Lips.
4. Facial: jaw reduction, wrinkle removal, and nasolabial fold removal. It is recommended to use "face shape" to achieve the comprehensive effect of large eyes and slim face.



```
void boolean enableEnhancedMode()
```

exportCurrentTexture

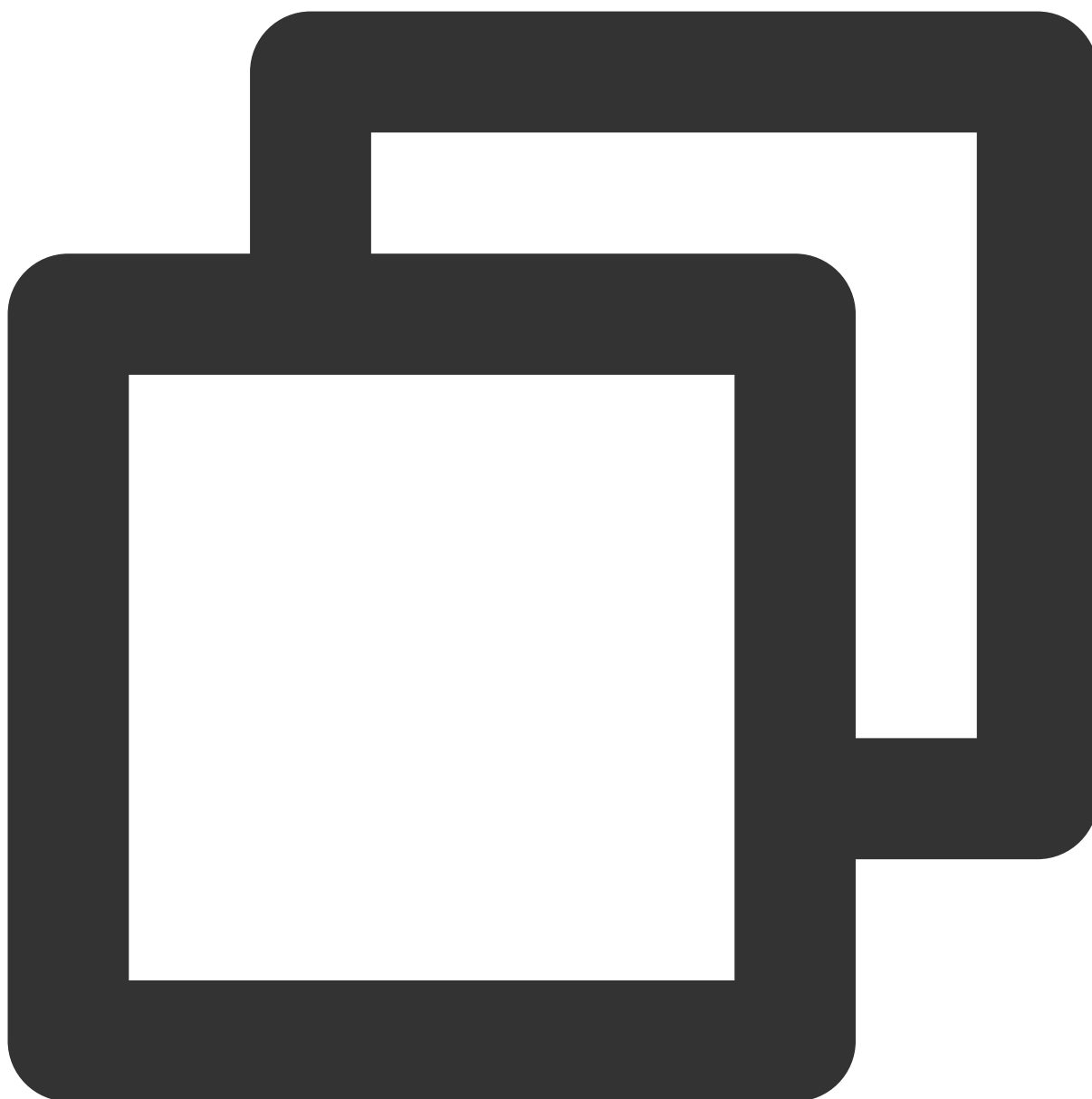
Get the picture on the current texture



```
void exportCurrentTexture(ExportTextureCallback callback)
```

setFeatureEnableDisable

Enable or disable a feature.



```
void setFeatureEnableDisable(String featureName, boolean enable)
```

Parameter

Parameter	Meaning
String featureName	feature Name Values : <code>XmagicConstant.FeatureName.ANIMOJI_52_EXPRESSION</code> facial expressions feature.

	<code>XmagicConstant.FeatureName.BODY_3D_POINT</code> 3D body data feature. <code>XmagicConstant.FeatureName.HAND_DETECT</code> gesture detection. <code>XmagicConstant.FeatureName.WHITEN_ONLY_SKIN_AREA</code> Brightening only applies to skin. <code>XmagicConstant.FeatureName.SEGMENTATION_SKIN</code> segmentation skin. <code>XmagicConstant.FeatureName.SMART_BEAUTY</code> smart beauty(reducing the intensity of beauty and makeup effects for males and babies).
boolean enable	"true" indicates enabling a capability, while "false" indicates disabling a capability. Note: If it is in downgrade mode, enabling skin segmentation is not allowed.

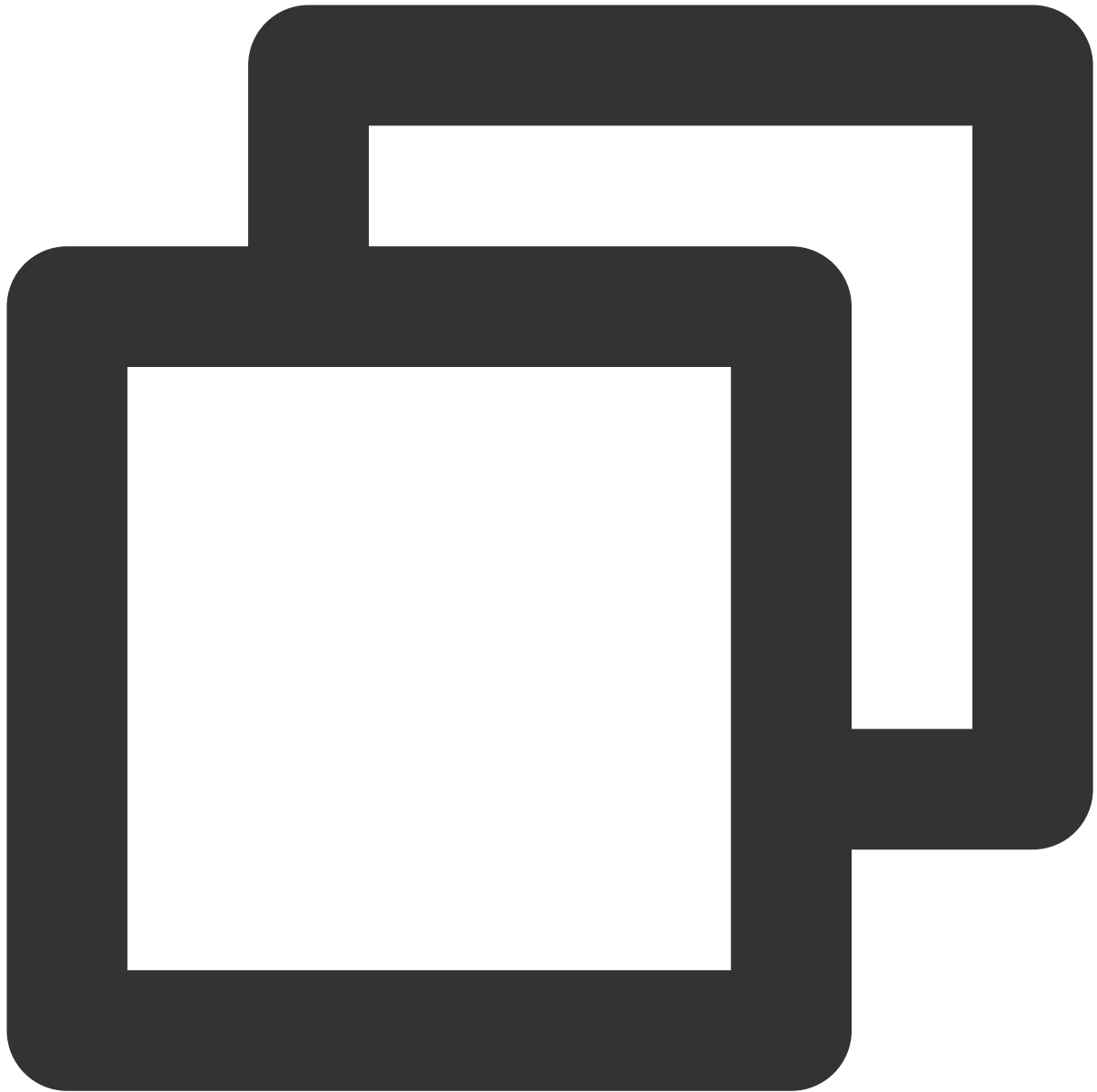
Static properties and methods

API	Description
VERSION	The SDK version number can be obtained through <code>XmagicApi.VERSION</code> (Added in V3.5.0)
setLibPathAndLoad	Setting libPath.
addAiModeFilesFromAssets	Transfer the content located within the directories <code>Light3DPlugin</code> , <code>LightCore</code> , <code>LightHandPlugin</code> , <code>LightBodyPlugin</code> , <code>LightSegmentPlugin</code> under application assets to your designated directory.
addAiModeFiles	Copy the AI model files downloaded by the client to the corresponding folders.
getDeviceLevel	get Device level

setLibPathAndLoad

Setting the path for SO and triggering a load. If SO is built into the assets, there is no need to invoke this method. If SO is dynamically downloaded, it must be invoked before authentication and `new XmagicApi`.

Passing in null indicates loading the so from the default path. Please ensure that the so is built into the APK package. Pass in non-null: such as `data/data/package name/files/xmagic_libs`, the so will be loaded from this directory.



```
static boolean setLibPathAndLoad(String path)
```

Parameter

Parameter	Meaning
String path	so: denotes the path to the library.

addAiModeFilesFromAssets

Transfer the content located within the directories Light3DPlugin, LightCore, LightHandPlugin, LightBodyPlugin, LightSegmentPlugin under application assets to your designated directory.

context applicationcontext.

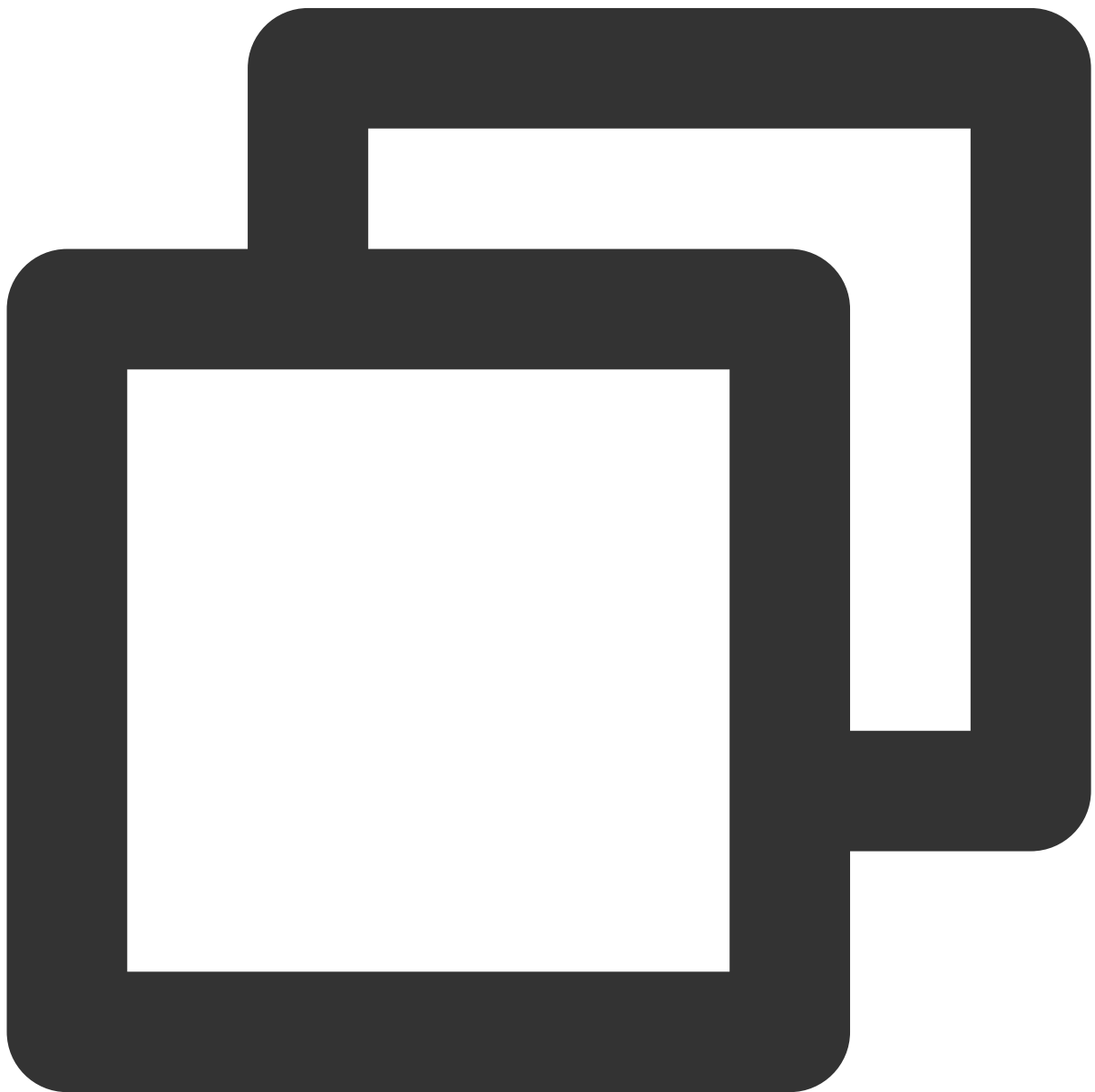
resDir serves as the root directory for storing beautification resources. This directory corresponds to the path passed when creating an xmagicApi object.

Returned values:

0: Copy successful

-1: denotes that the context is null

-2: denotes an IO error



```
static int addAiModeFilesFromAssets(Context context, String resDir)
```

Parameter

Parameter	Meaning
Context context	applicationcontext.
String resDir	Designated as the root directory for storing beautification resources, this directory corresponds to the path input when creating the xmagicApi object.

addAiModeFiles

Copy the AI model files downloaded by the client to the corresponding folders.

inputResDir is the directory of the successfully downloaded model files.

resDir serves as the root directory for storing beautification resources. This directory corresponds to the path passed when creating an xmagicApi object.

Returned values:

0: denotes success

-1:inputResDir is not exists

-2: Signifies an IO error



```
static int addAiModeFiles(String inputResDir, String resDir)
```

Parameter

Parameter	Meaning
String inputResDir	The folder containing the successfully downloaded model files.
String resDir	Designated as the root directory for storing beautification resources, this directory corresponds to the path input when creating the xmagicApi object.

getDeviceLevel (Added in V3.7.0)

get Device level

context application context。

return param :

DeviceLevel : Device levels, with corresponding values

DEVICE_LEVEL_VERY_LOW(1)

DEVICE_LEVEL_LOW(2)

DEVICE_LEVEL_MIDDLE(3)

DEVICE_LEVEL_MIDDLE_HIGH(4)

DEVICE_LEVEL_HIGH(5)



```
static DeviceLevel getDeviceLevel(Context context)
```


Flutter

Last updated : 2024-02-29 17:38:28

`TencentEffectApi` is the core API class of the Tencent Effect Flutter SDK. It offers capabilities including setting the effect strength and applying animated effects.

Public Member APIs

API	Description
<code>setResourcePath</code>	Set the local storage path of beauty resources (V0.3.5.0 version)
<code>initXmagic</code>	Initializes data. You need to call this API before using the Tencent Effect SDK (V0.3.1.1 and earlier) .
<code>setLicense</code>	Configures the license.
<code>setXmagicLogLevel</code>	Sets the log level of the SDK. We recommend you set it to <code>Log.DEBUG</code> for debugging and <code>Log.WARN</code> for official release. If you set it to <code>Log.DEBUG</code> in a production environment, the output of a large amount of log data may affect your application's performance.
<code>onResume</code>	Resumes rendering. Call this API when the page is visible.
<code>onPause</code>	Pauses rendering. Call this API when the page is invisible.
<code>enableEnhancedMode</code>	enable enhanced mode
<code>setDowngradePerformance</code>	Invoke this method to enable the high-performance pattern. Upon the activation of the high-performance pattern, the system CPU/GPU resources occupied by beauty filters are minimized, thereby reducing heat generation and latency issues in the mobile device. It is particularly suitable for prolonged use on low-end devices.
<code>setAudioMute</code>	set mute (Because some stickers have sound)
<code>setFeatureEnableDisable</code>	enable or disable the feature
<code>updateProperty</code>	Updates an effect property. This API can be called in any thread.
<code>setEffect</code>	Updates an effect property (V0.3.5.0 Version)
<code>setOnCreateXmagicApiErrorListener</code>	Configures the callback for creating an effect object. The callback will be triggered if an error occurs.

setTipsListener	Configures the callback for animated effect tips. The tips can be displayed on the UI.
setYTDataListener	Configure the callback of facial keypoints and other data, callback will only be available with the License authorization required to acquire facial keypoints (such as Atomic Capability X102).
setAIDataListener	Configures the callback of face, gesture, and body detection results.
isBeautyAuthorized	Checks whether the current license supports a particular type of effects. This API can only check the authorization of <code>BEAUTY</code> and <code>BODY_BEAUTY</code> effects. The result returned determines the value of <code>XmagicProperty.isAuth</code> .
isSupportBeauty	Checks whether the current device supports effects (OpenGL 3.0).
getDeviceAbilities	Gets a list of Tencent Effect capabilities supported by the current device.
isDeviceSupport	Checks whether a list of animated effect resources are supported. The result is indicated by <code>XmagicProperty.isSupport</code> . For unsupported resources, you can either disable tapping on the UI or delete them from the resource list.
isDeviceSupportMotion	Check if the current device supports this material.
getPropertyRequiredAbilities	Gets the Tencent Effect capabilities used by different animated effect resources.

API Description

setResourcePath (V0.3.5.0)

To set the local path for storing beauty resources



```
/// Set the local path for storing beauty resources. This method must be called bef  
/// Added in v0.3.5.0.  
void setResourcePath(String xmagicResDir);
```

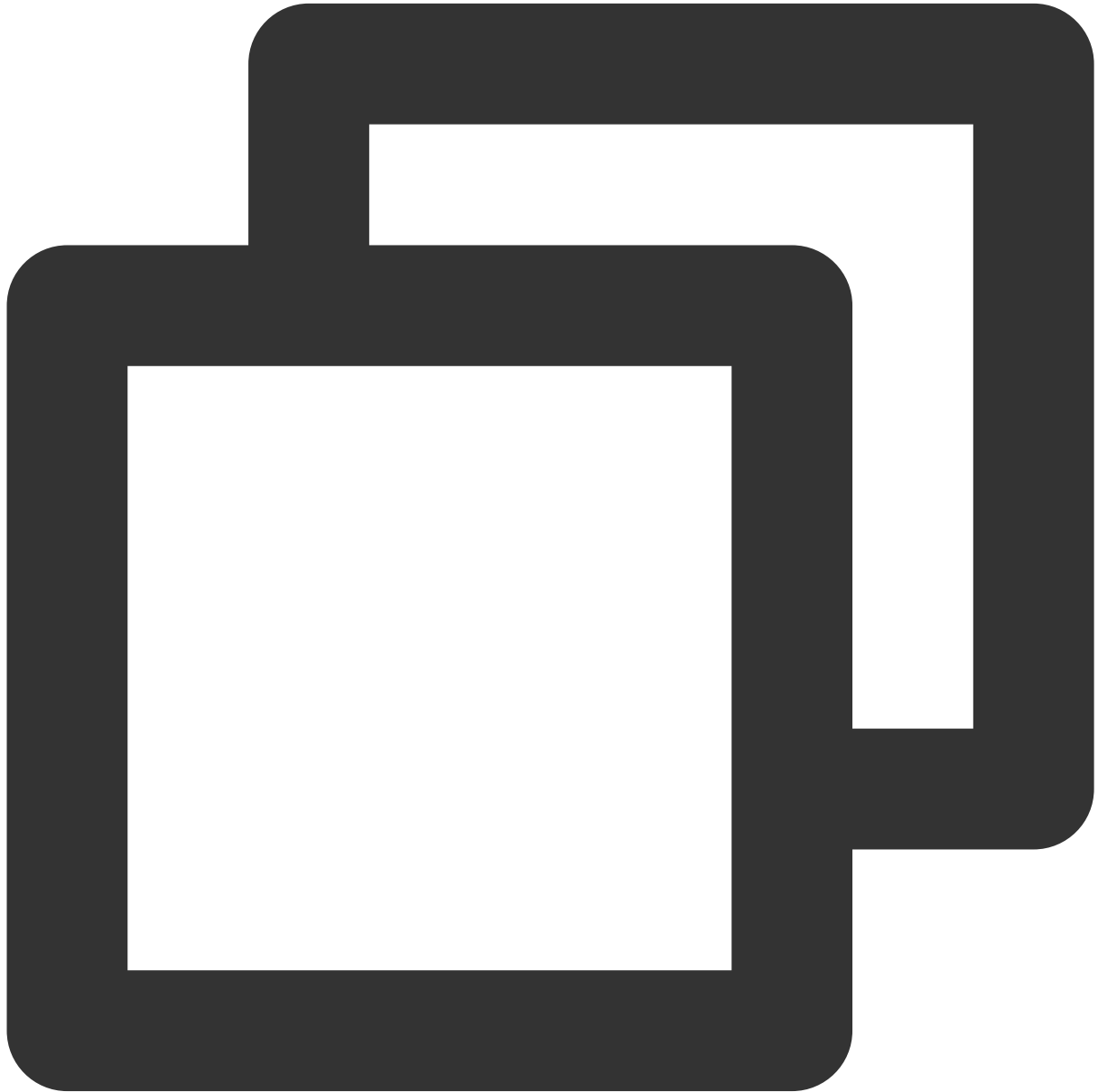
Parameters

Parameter	Description
String xmagicResDir	The resource directory.

initXmagic

Initialize beauty data. In versions prior to `v0.3.1.1`, this method must be called before using beauty effects. Starting from `v0.3.5.0`, this method only needs to be called once per version, and the `setResourcePath` method must be called before this method to set the resource path. In `v0.3.5.0`, the previous `xmagicResDir` parameter has been removed. Please refer to the latest demo for more information.

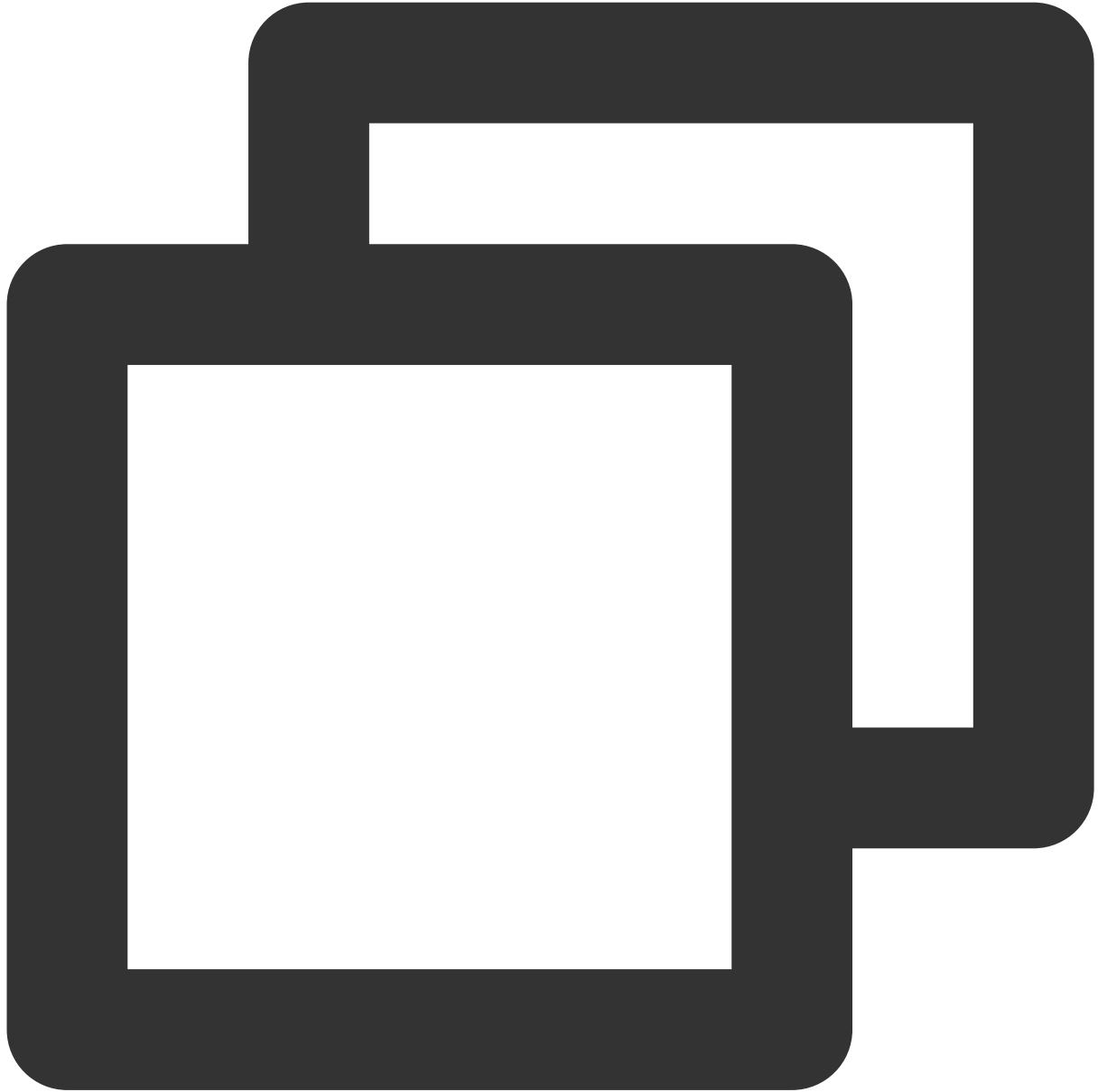
V0.3.5.0 :



```
void initXmagic(InitXmagicCallBack callBack);  
  
typedef InitXmagicCallBack = void Function(bool reslut);
```

V0.3.1.1 and earlier :

This API is used to initialize the Tencent Effect SDK.



```
void initXmagic(String xmagicResDir, InitXmagicCallBack callBack);  
  
typedef InitXmagicCallBack = void Function(bool reslut);
```

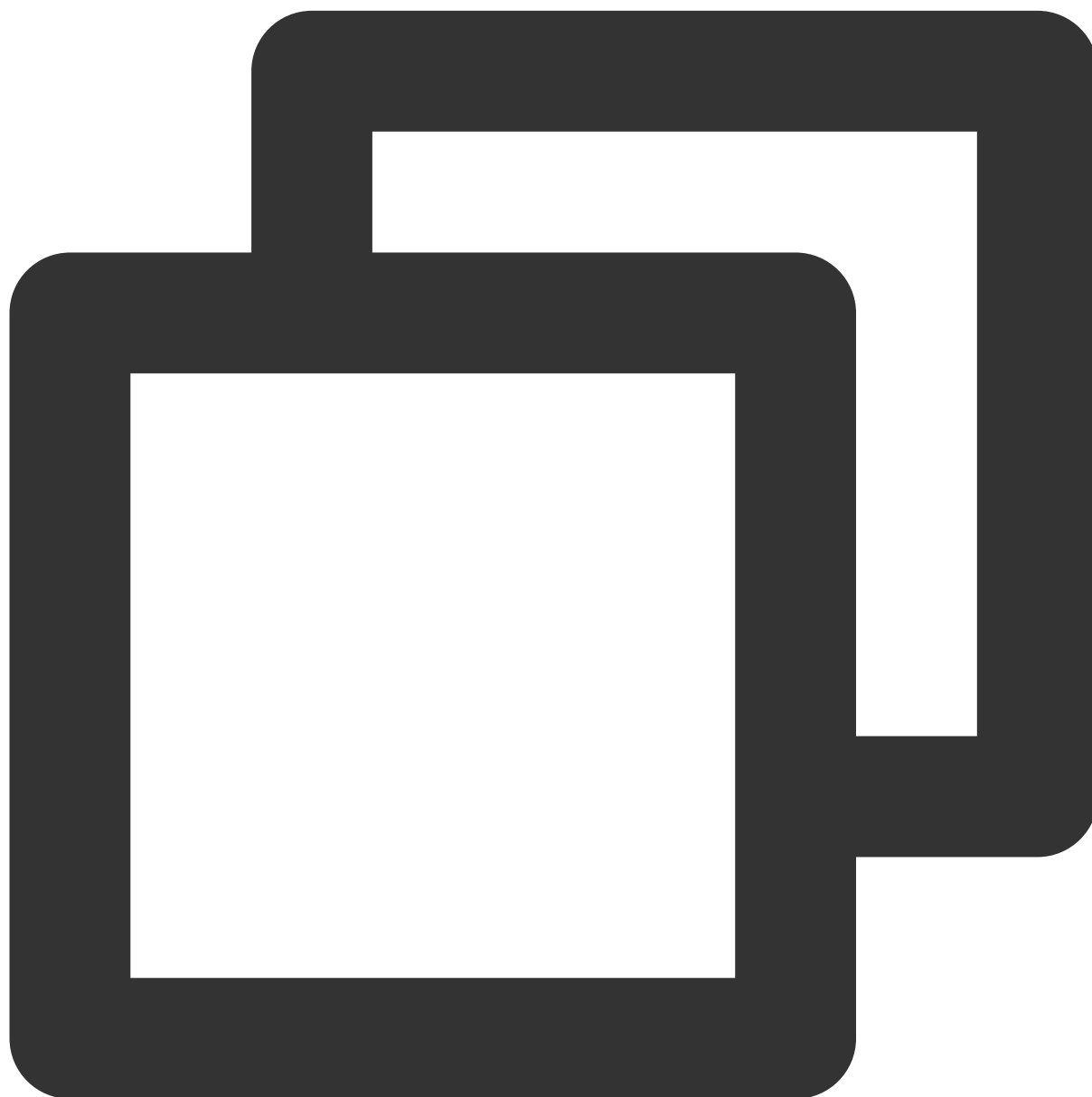
Parameters

Parameter	Description
-----------	-------------

String xmagicResDir	The resource directory.
InitXmagicCallBack callBack	The initialization callback.

setLicense

This API is used to set the license.



```
///Set the Tencent Effect license  
void setLicense(String licenseKey, String licenseUrl, LicenseCheckListener checkLis  
//The callback of the authorization result
```

```
typedef LicenseCheckListener = void Function(int errorCode, String msg);
```

Parameters

Parameter	Description
String licenseKey	The license key.
String licenseUrl	The license URL.
LicenseCheckListener checkListener	The callback of the authorization result.

setXmagicLogLevel

This API is used to set the log level of the SDK.



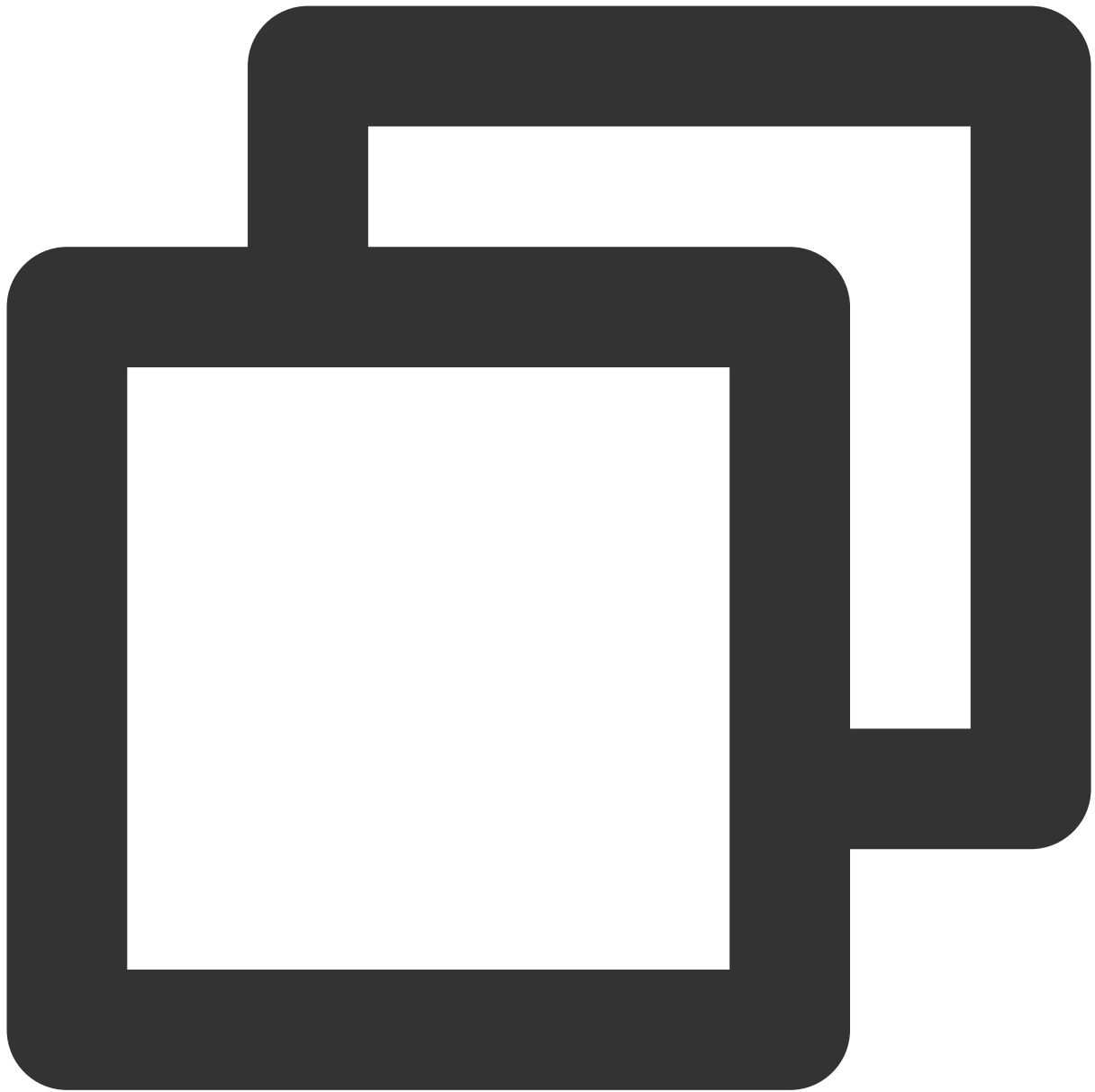
```
void setXmagicLogLevel(int logLevel);
```

Parameters

Parameter	Description
int logLevel	You can set the log level using a type defined for <code>LogLevel</code> .

onResume

This API is used to resume effect rendering.



```
void onResume();
```

onPause

This API is used to pause effect rendering.



```
void onPause();
```

enableEnhancedMode

enable enhanced mode



```
void enableEnhancedMode();
```

setDowngradePerformance (V0.3.1.1)

Invoke this method to enable the high-performance pattern



```
void setDowngradePerformance();
```

setAudioMute (V0.3.1.1)

To set the mute status, the parameter "true" represents mute, while "false" represents unmute.



```
/// Is the background music muted?  
void setAudioMute(bool isMute);
```

setFeatureEnableDisable (V0.3.1.1)

enable or disable one feature



```
/// enable or disable one feature  
void setFeatureEnableDisable(String featureName, bool enable);
```

Parameter

Parameter	Meaning
String featureName	feature Name Values : "ai.3dmmV2.enable" facial expressions feature.

	<p><code>"ai.body3dpoint.enable"</code> 3D body data feature.</p> <p><code>"ai.hand.enable"</code> gesture detection.</p> <p><code>"beauty.onlyWhitenSkin"</code> Brightening only applies to skin.</p> <p><code>"ai.segmentation.skin.enable"</code> segmentation skin.</p> <p><code>"auto_beauty_switch"</code> smart beauty(reducing the intensity of beauty and makeup effects for males and babies).</p>
boolean enable	<p>"true" indicates enabling a capability, while "false" indicates disabling a capability.</p> <p>Note: If it is in downgrade mode, enabling skin segmentation is not allowed.</p>

updateProperty

This API is used to set an effect value, an animated effect, or a filter. You can call it in any thread.



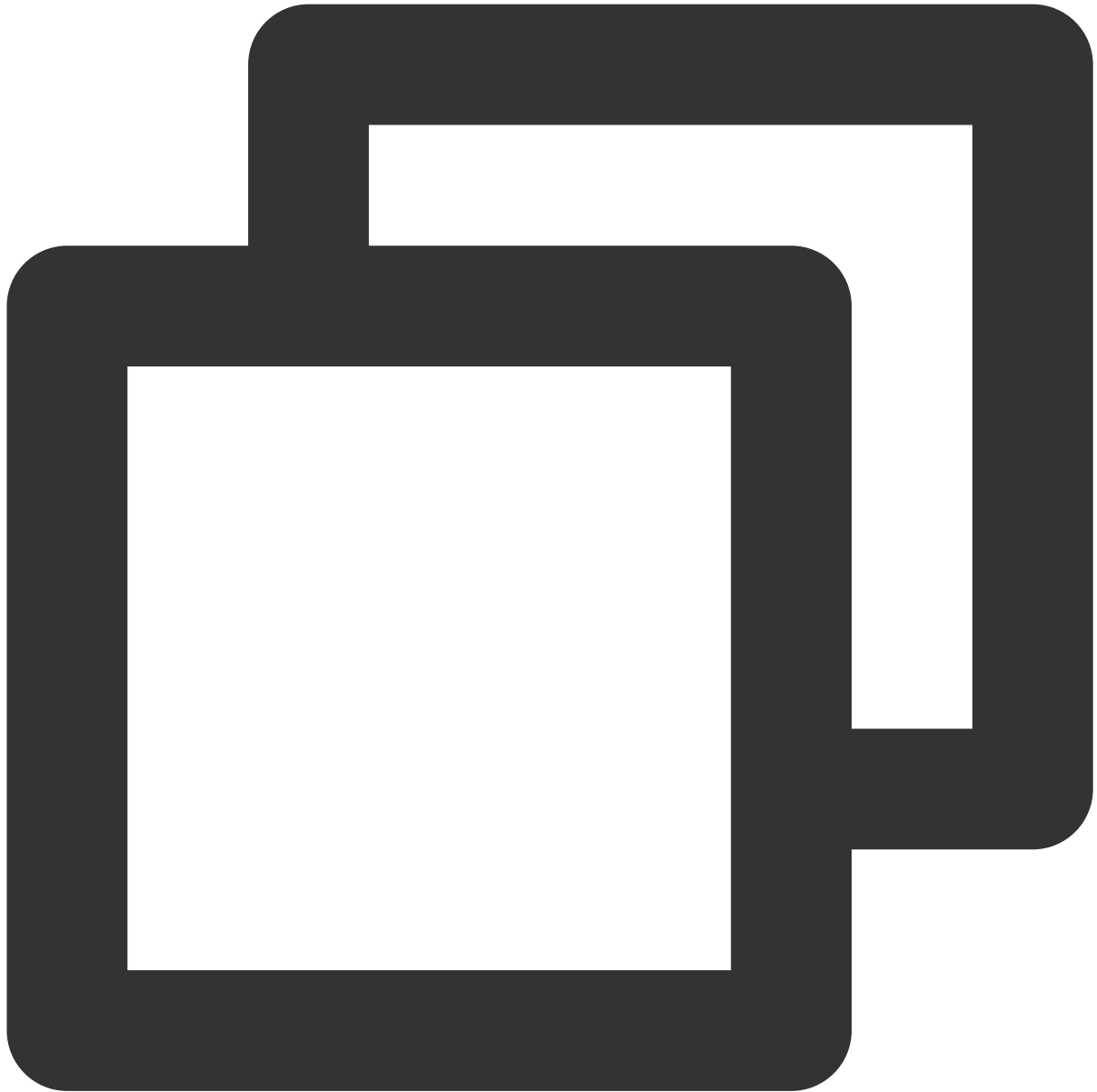
```
void updateProperty(XmagicProperty xmagicProperty);
```

Parameters

Parameter	Description
XmagicProperty xmagicProperty	The object of the effect property.

setEffect (V0.3.5.0)

You can set beautification, filters, makeup, stickers, and segmentation effects. This can be done from any thread. Please refer to the specific parameters for more details [Beautification Parameter Table](#).



```
///update beautification parameters  
void setEffect(String effectName,int effectValue,String? resourcePath,Map<String,St
```

setOnCreateXmagicApiErrorListener

This API is used to configure the callback for errors for the creation of an effect object.



```
void setOnCreateXmagicApiErrorListener(OnCreateXmagicApiErrorListener? errorListe
/// The callback for errors for the creation of an effect object
typedef OnCreateXmagicApiErrorListener = void Function(String errorMsg, int code);
```

Parameters

Parameter	Description
OnCreateXmagicApiErrorListener? errorListener	The callback for errors for the creation of an effect object.

Error codes:

Error Code	Description
-1	Unknown error.
-100	Failed to initialize the 3D engine.
-200	GAN materials are not supported.
-300	The device does not support this material component.
-400	The JSON template is empty.
-500	The SDK version is too old.
-600	Keying is not supported.
-700	OpenGL is not supported.
-800	The script is not supported.
5000	The resolution of the video to be keyed exceeds 2160 x 3840.
5001	Insufficient memory for keying.
5002	Failed to parse the video to be keyed.
5003	The video to be keyed is longer than 200 seconds.
5004	Unsupported video format for keying.

setTipsListener

This API is used to configure the callback for animated effect tips. The tips can be displayed on the UI, asking users to nod, show their palms, or make finger hearts.



```
void setTipsListener(XmagicTipsListener? xmagicTipsListener);

abstract class XmagicTipsListener {
    /// Show the tip
    /// @param tips: The content of the tip (string).
    /// @param tipsIcon: The icon for the tip.
    /// @param type: The display type. If it is set to `0`, both the tip string and i
    /// @param duration: How long (milliseconds) to show the tip.
    void tipsNeedShow(String tips, String tipsIcon, int type, int duration);

    /// *
```

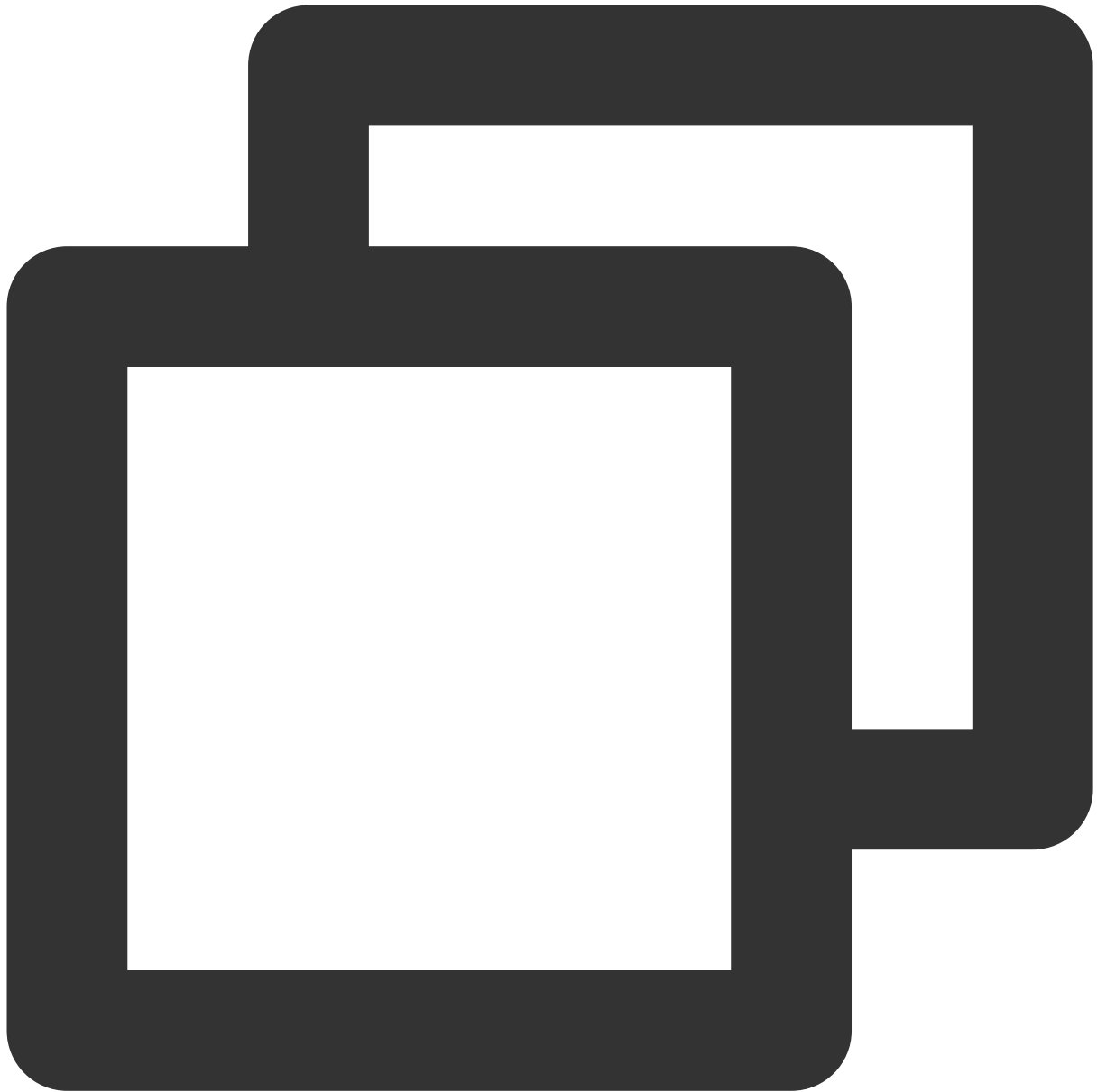
```
/// Hide the tip
/// @param tips: The content of the tip (string).
/// @param tipsIcon: The icon for the tip.
/// @param type: The display type. If it is set to `0`, both the tip string and i
void tipsNeedHide(String tips, String tipsIcon, int type);
}
```

Parameters

Parameter	Description
XmagicTipsListener xmagicTipsListener	The callback implementation class.

setYTDataListener

This API is used to configure the callback of facial keypoints and other data.



```
/// Configure the callback of facial keypoints and other data (only available in  
void setYTDataListener(XmagicYTDataListener? xmagicYTDataListener);  
Configure the callback of facial keypoints and other data
```

```
abstract class XmagicYTDataListener {  
    // YouTu AI data  
    void onYTDataUpdate(String data);  
}
```

`onYTDataUpdate` returns a JSON string structure that contains the information of up to 5 faces:



```
{
  "face_info": [{
    "trace_id": 5,
    "face_256_point": [
      180.0,
      112.2,
      ...
    ],
    "face_256_visible": [
      0.85,
      ...
    ]
  }]
```

```
],  
  "out_of_screen":true,  
  "left_eye_high_vis_ratio":1.0,  
  "right_eye_high_vis_ratio":1.0,  
  "left_eyebrow_high_vis_ratio":1.0,  
  "right_eyebrow_high_vis_ratio":1.0,  
  "mouth_high_vis_ratio":1.0  
},  
...  
]  
}
```

Fields

Field	Type	Range	Description
trace_id	int	[1,INF)	The face ID. If the faces obtained from a continuous video stream have the same face ID, they belong to the same person.
face_256_point	float	[0,screenWidth] or [0,screenHeight]	512 values in total for 256 facial keypoints. (0,0) is the top-left corner of the screen.
face_256_visible	float	[0,1]	The visibility of the 256 facial keypoints.
out_of_screen	bool	true/false	Whether only part of the face is captured.
left_eye_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the left eye.
right_eye_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the right eye.
left_eyebrow_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the left eyebrow.
right_eyebrow_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the right eyebrow.
mouth_high_vis_ratio	float	[0,1]	The percentage of keypoints with high visibility for the mouth.

Parameters

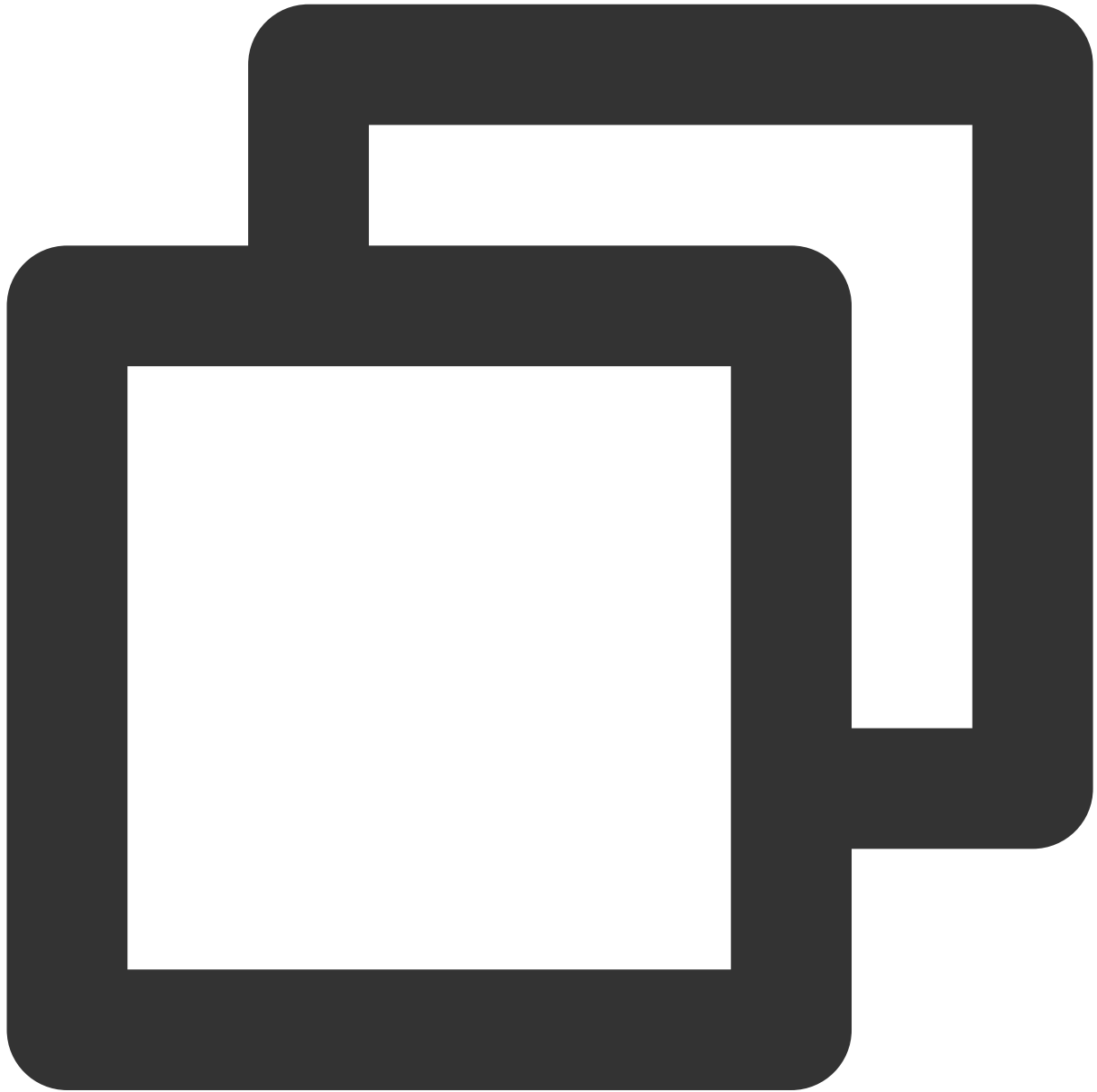
Parameter	Description

XmagicYTDataListener

The callback implementation class.

setAIDataListener

This API is used to configure the callback of face, gesture, and body detection results.



```
void setAIDataListener(XmagicAIDataListener? aiDataListener);
```

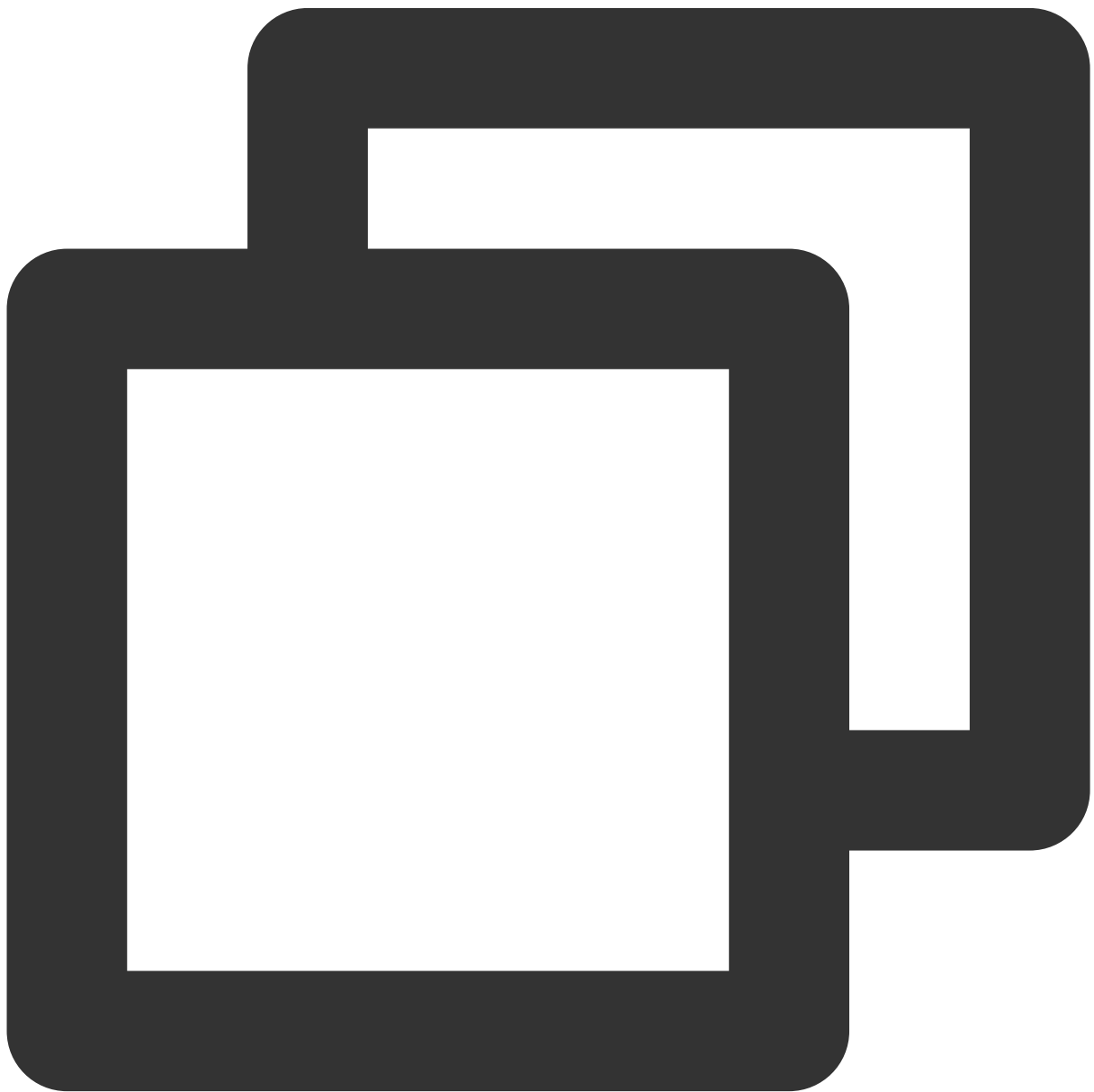
```
abstract class XmagicAIDataListener {  
    void onFaceDataUpdated(String faceDataList);  
}
```

```
void onHandDataUpdated(String handDataList);

void onBodyDataUpdated(String bodyDataList);
}
```

isBeautyAuthorized

This API is used to check whether the current license supports a particular type of effects. It can only check the authorization of `BEAUTY` and `BODY_BEAUTY` effects. The result returned determines the value of `XmagicProperty.isAuth`. If `isAuth` is `false`, you can disable the corresponding effects on the UI.



```
Future<List<XmagicProperty>> isBeautyAuthorized(  
    List<XmagicProperty> properties);
```

Parameters

Parameter	Description
List<XmagicProperty> properties	The type of effects to check.

isSupportBeauty

This API is used to check whether the current device supports effects (OpenGL 3.0).



```
Future<bool> isSupportBeauty();
```

Response

A Boolean value indicating whether effects are supported.

getDeviceAbilities

This API is used to get a list of Tencent Effect capabilities supported by the current device. You can use it together with `getPropertyRequiredAbilities`.



```
Future<Map<String, bool>> getDeviceAbilities();
```

Response

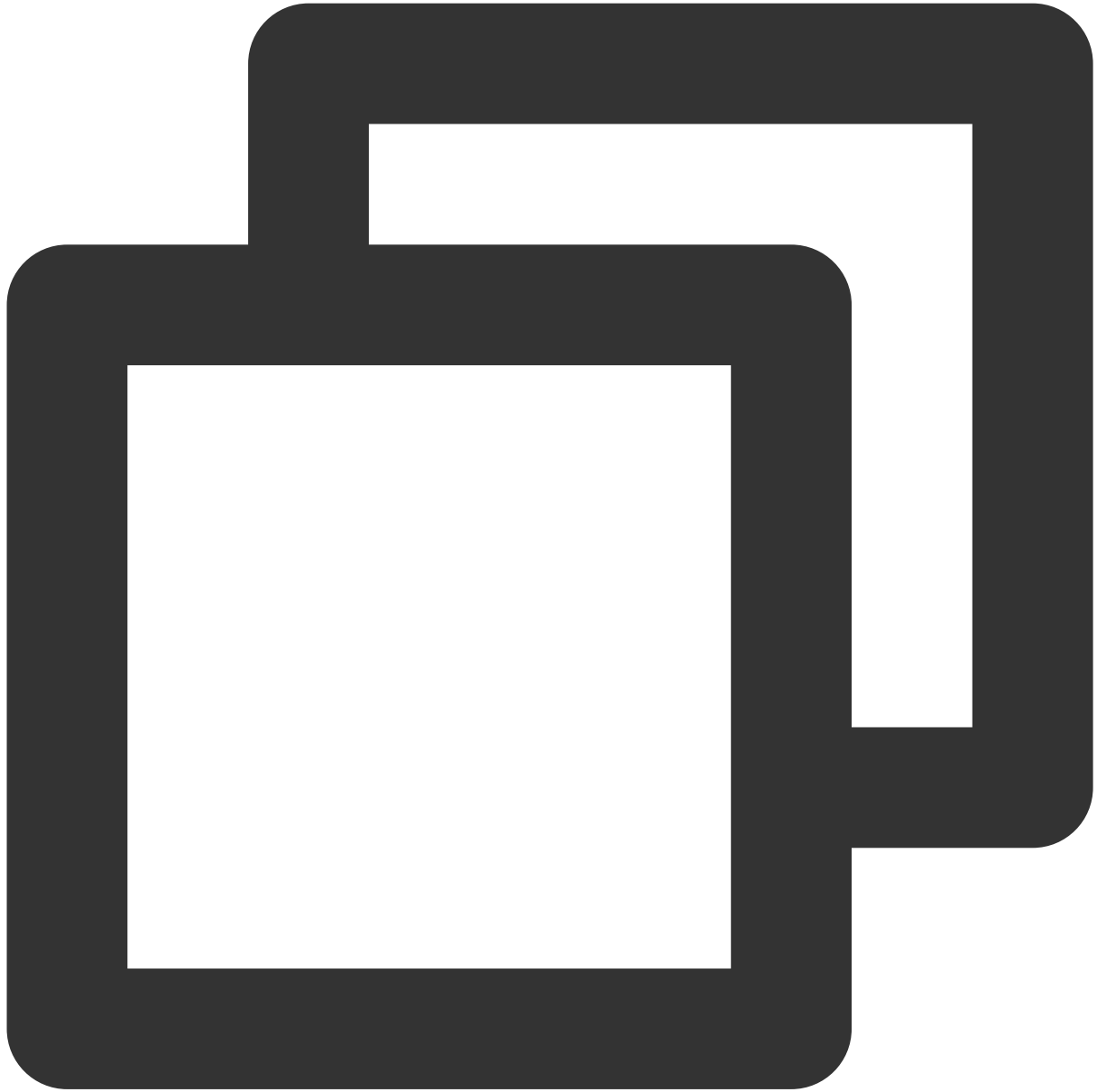
`Map<String, Boolean>` :

key: The name of a capability (the material name).

value: Whether the current device supports the capability.

isDeviceSupport

This API is used to check whether a list of animated effect resources are supported. The result is indicated by `XmagicProperty.isSupport` . For unsupported resources, you can either disable tapping on the UI or delete them from the resource list.



```
Future<List<XmagicProperty>> isDeviceSupport (List<XmagicProperty> assetsList);
```

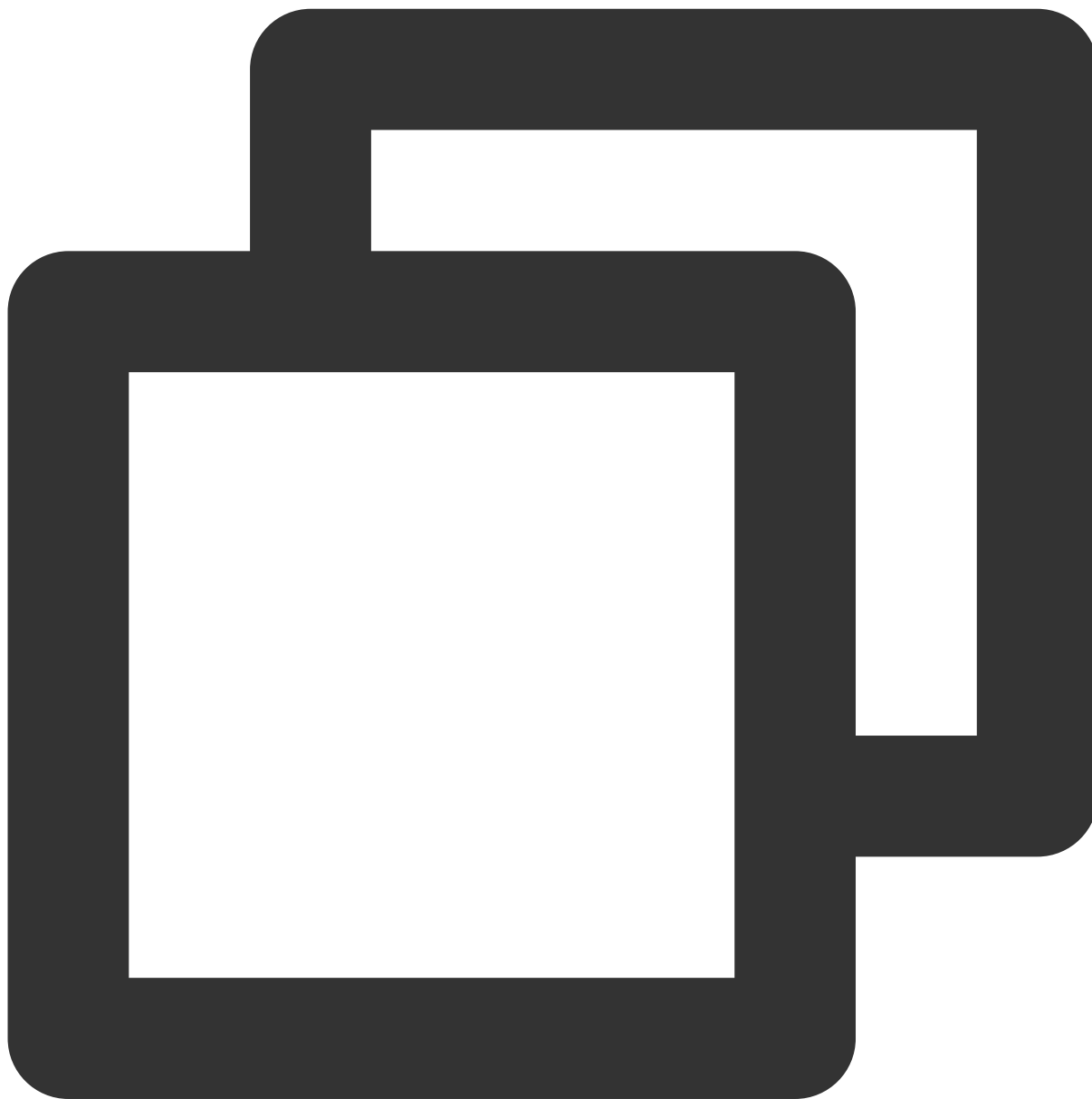
Parameters

Parameter	Description

List<XmagicProperty> assetsList	A list of animated effect resources to check.
---------------------------------	---

isDeviceSupportMotion (V0.3.5.0)

To check if the current device supports a particular material



```
Future<bool> isDeviceSupportMotion(String motionResPath);
```

Parameters

Parameter	Description
-----------	-------------

motionResPath	the sticker local file path
---------------	-----------------------------

getPropertyRequiredAbilities

This API is used to get the Tencent Effect capabilities used by different animated effect resources.

Use case:

This API is useful if you have purchased animated effects or made your own animated effect materials. It returns the capabilities each material uses. For example, material 1 uses capabilities A, B, and C, and material 2 relies on capabilities B, C, and D. You can store such information in the server. When a user downloads the two materials from the server, call `getDeviceAbilities` first to get the capabilities supported by their device. The result is then passed to the server. For example, if a user's device supports capabilities A, B, and C, but not D, the server will not provide material 2 to the user.



```
Future<Map<XmagicProperty, List<String>?>> getPropertyRequiredAbilities(  
    List<XmagicProperty> assetsList);
```

Parameters

Parameter	Description
List<XmagicProperty> assetsList	A list of the animated effects to check.

Response

Map<XmagicProperty, List<String>?>:

key: The entity class of the animated effect.

value: A list of capabilities used by the effect.

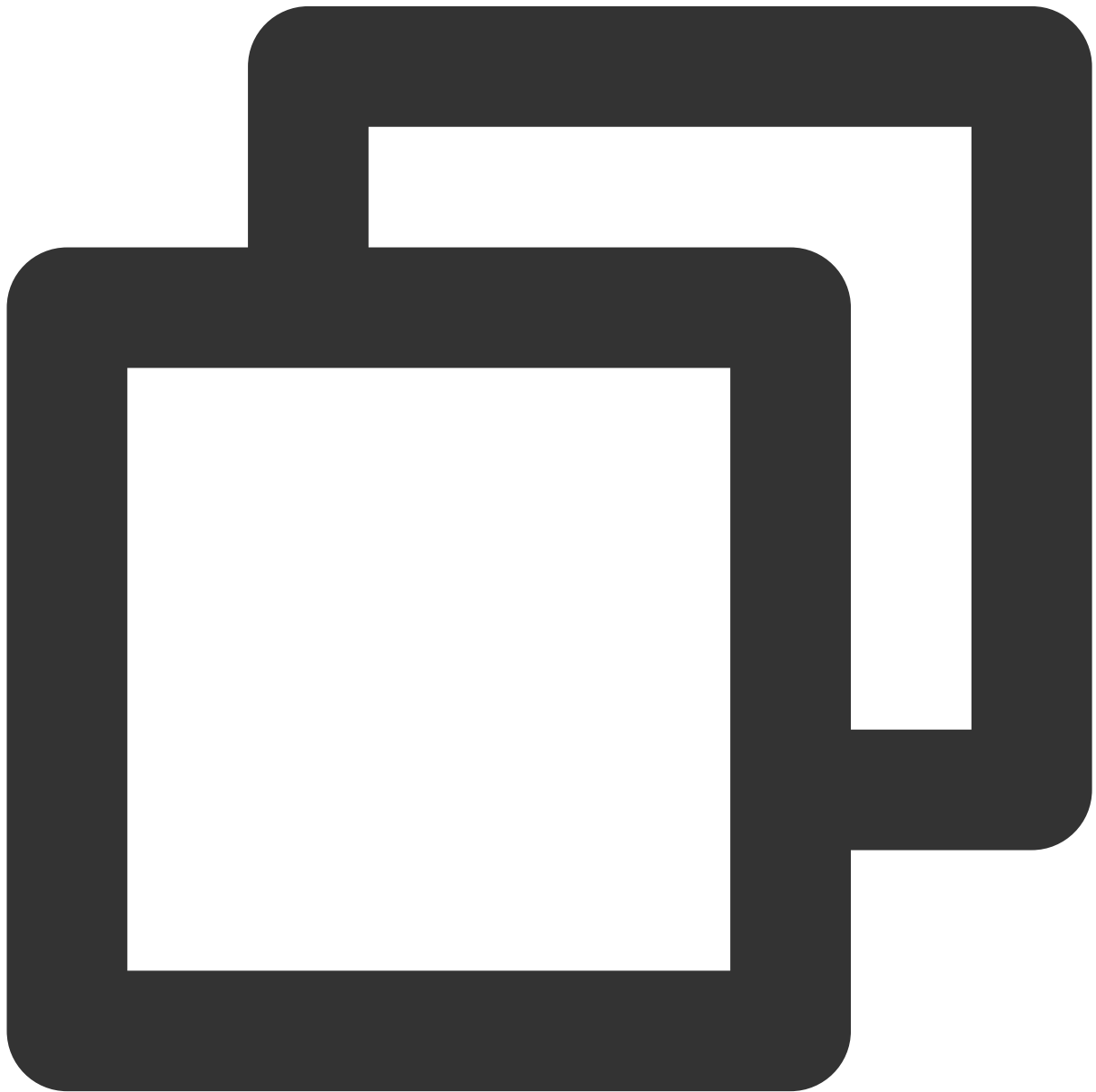
Web

Last updated : 2023-05-06 15:45:32

This document describes the core parameters and methods of the Beauty AR Web SDK.

Note:

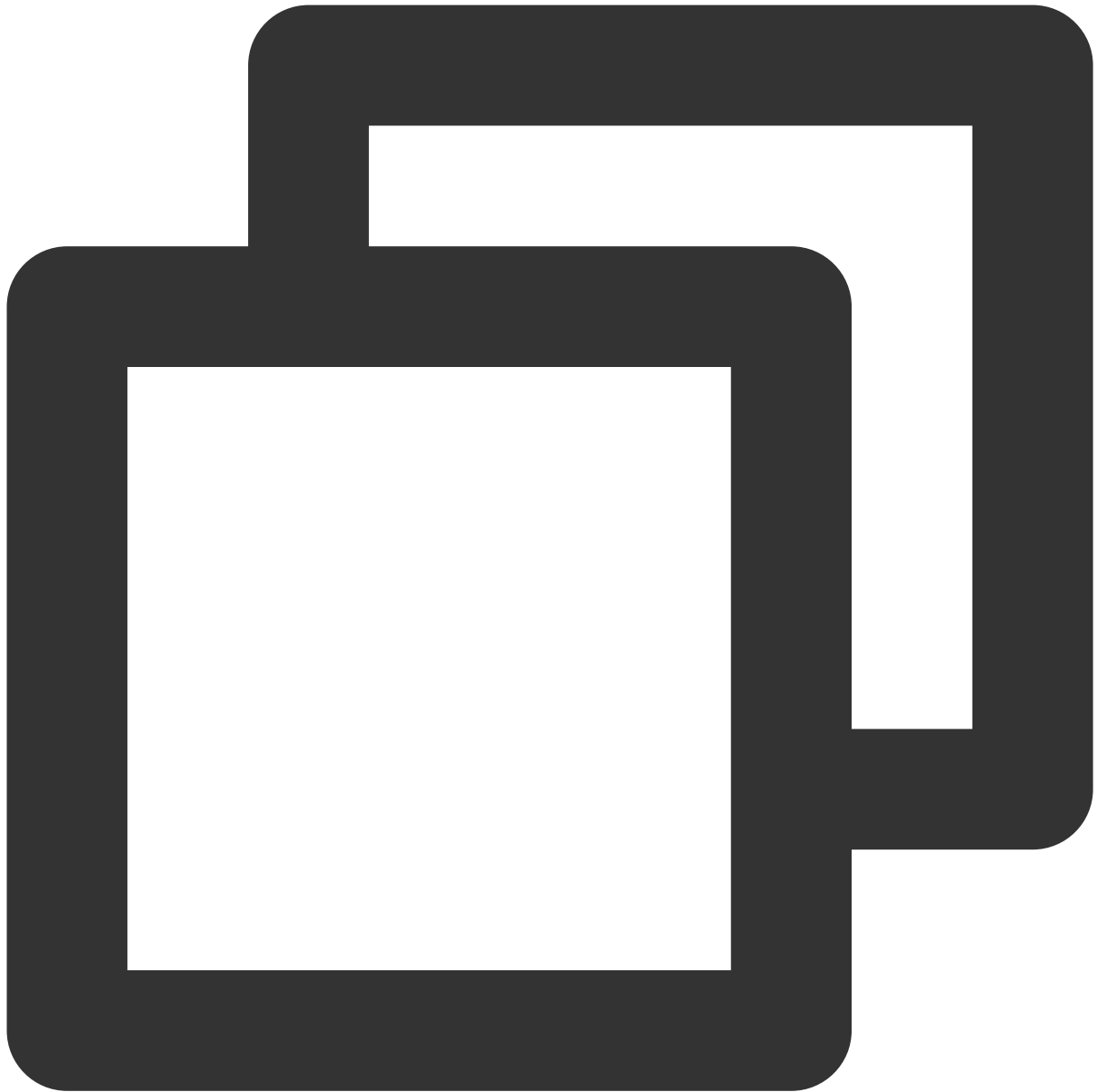
The Beauty AR Web SDK relies on hardware acceleration to achieve smooth rendering (this does not need to be checked in mini programs). The SDK allows you to check whether a browser supports hardware acceleration. You can block the browser if it does not support hardware acceleration.



```
import {ArSdk, isWebGLSupported} from 'tencentcloud-webar'

if(isWebGLSupported()) {
  const sdk = new ArSdk({
    ...
  })
} else {
  // The browser blocking logic
}
```

Initialization Parameters



```
import { ArSdk } from 'tencentcloud-webar'  
// Initialize the SDK  
const sdk = new ArSdk({  
  ...  
})
```

Config of the SDK supports the following initialization parameters:

Parameters	Description	Type
module	The module configuration	

auth	The authentication parameter	<pre>type bea seq }</pre>

```
type
  lic
  app
  aut
  s
  t
  }>
}
```

	w	b	r	f
type				

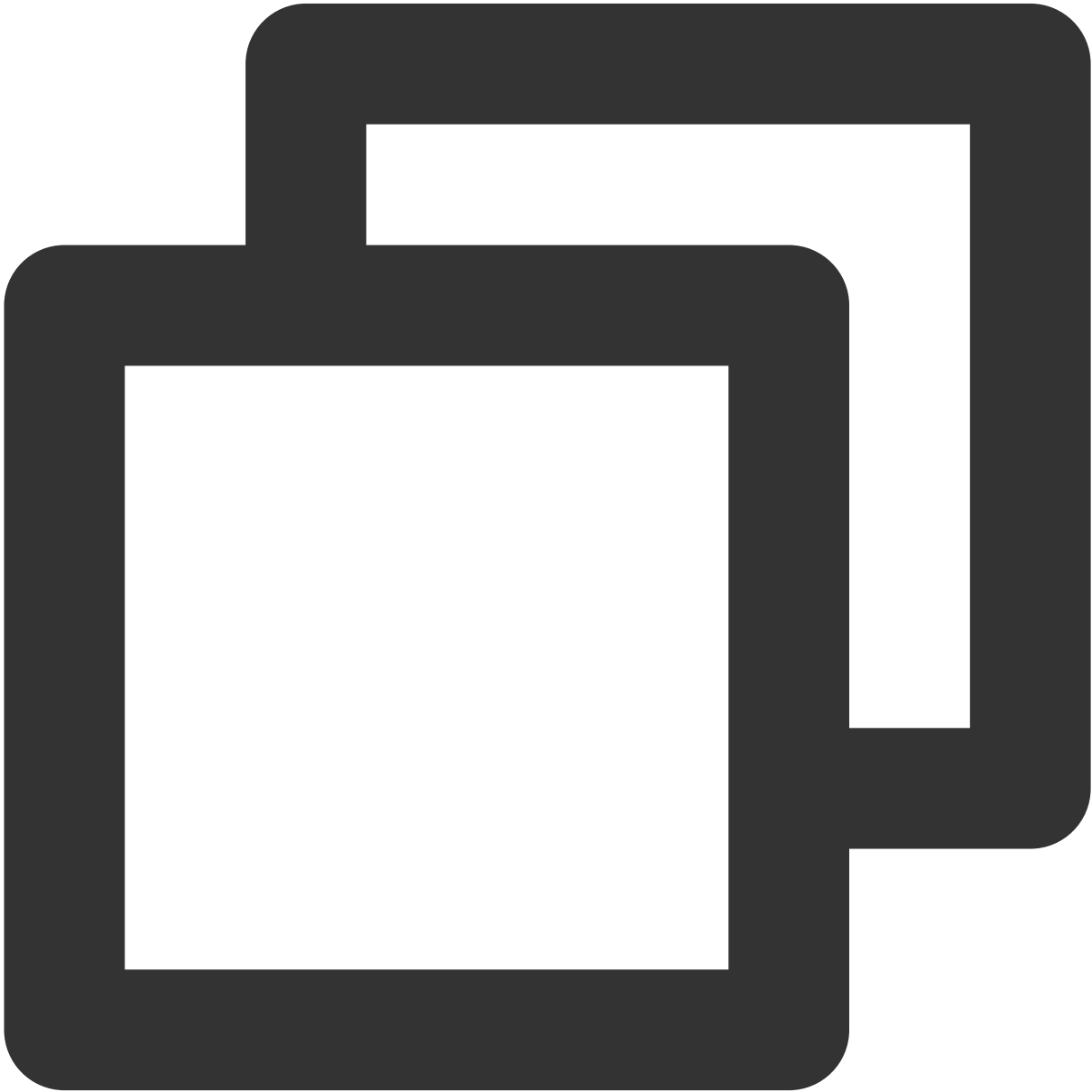
		}
beautify	The beauty filter parameter	
		type w c l s

		<div>€ C }</div>
background	The background parameter	
		<div>type t s</div>

		}
loading	The configuration of the built-in loading icon	
		type e s l s

		}
language	The language (supported since v1.0.6). Chinese (zh) and English (en) are supported.	String: zh

Callbacks



```
let effectList = [];
```



```
let filterList = [];  
// Using the callbacks of the SDK  
sdk.on('created', () => {  
  // Pull and display the filter and effect list in the `created` callback  
  sdk.getEffectList({  
    Type: 'Preset',  
    Label: 'Makeup',  
  }).then(res => {  
    effectList = res  
  });  
  sdk.getCommonFilter().then(res => {  
    filterList = res  
  })  
})  
  
sdk.on('cameraReady', async () => {  
  // By getting the output stream in the `cameraReady` callback, you can display  
  // You can choose this method if you want to display a video image as soon as p  
  // You don't need to update the stream after the effects start to work.  
  const arStream = await ar.getOutput();  
  // Play the stream locally  
  // localVideo.srcObject = arStream  
  
})  
  
sdk.on('ready', () => {  
  // Get the output stream in the `ready` callback. The initialization parameters  
  // You can get the output stream in `ready` if you want your video to show effe  
  // Between the two methods, choose the one that fits your needs.  
  const arStream = await ar.getOutput();  
  // Play the stream locally  
  // localVideo.srcObject = arStream  
  
  // Call `setBeautify`, `setEffect`, or `setFilter` in the `ready` callback  
  sdk.setBeautify({  
    whiten: 0.3  
  });  
  sdk.setEffect({  
    id: effectList[0].EffectId,  
    intensity: 0.7  
  });  
  sdk.setEffect({  
    id: effectList[0].EffectId,  
    intensity: 0.7,  
    filterIntensity: 0.5 // In v0.1.18 and later, you can use this parameter to  
  });  
  sdk.setFilter(filterList[0].EffectId, 0.5)  
})
```

Events	Description	Callback Parameter
created	The SDK authentication was completed and the instance was created successfully.	-
cameraReady	The SDK generated a video output (the video is not yet processed).	-
ready	Detection has been initialized. Effects are now applied to the output video. You can change the effect settings.	-
error	This callback is triggered when the SDK encounters an error.	The <code>error</code> object

APIs

API	Parameters
async getOutput(fps)	fps (optional): The output frame rate.
setBeautify(options)	



	 <pre>type BeautifyOptions = { whiten?: number, // The brightening effect. Value dermabrasion?: number // The smooth skin effect. lift?: number // The slim face effect. Value range shave?: number // The face width. Value range: 0- eye?: number // The big eyes effect. Value range: chin?: number // The chin effect. Value range: 0- }</pre>
setEffect(effects, callback)	effects: Effect ID Effect object Effect ID / An effect array 



```
effect:{  
  id: string,
```

```
intensity: number, // The effect strength. Value  
filterIntensity: number // The filter strength  
}
```

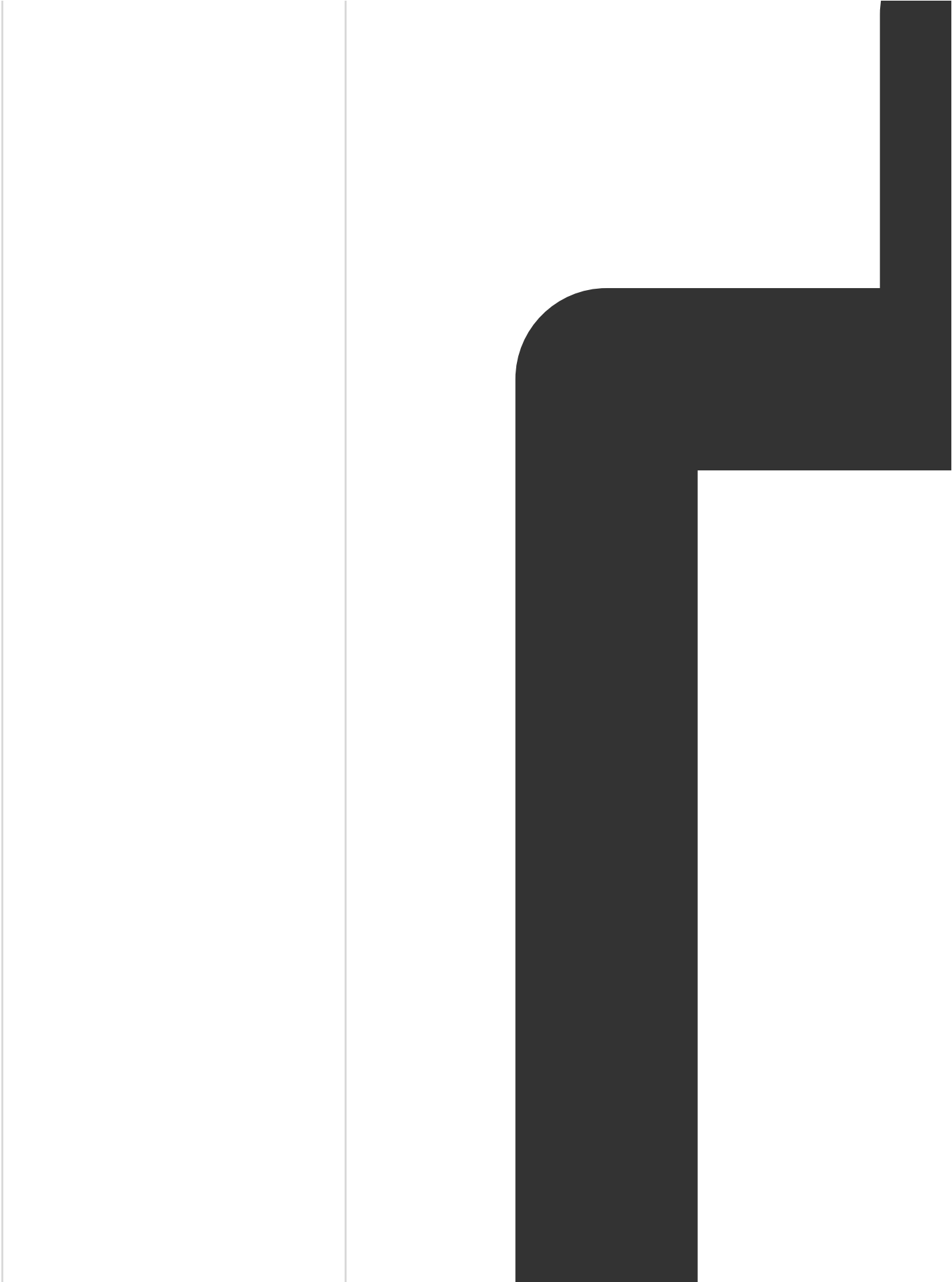
callback: The callback for successful configuration

setAvatar(params)



```
{  
  mode: 'AR' | 'VR',  
  effectId?: string, // Pass through `effectId` to  
  url?: string, // Pass through `url` to use the  
  backgroundUrl?: string, // Background image URI  
}
```

setBackground(options)





```
{
  type: 'image|blur|transparent',
  src: string // This parameter is required only
}
```

setFilter(id, intensity, callback)

id: The filter ID
intensity: The filter strength. Value range: 0-1.
callback: The callback for successful configuration

getEffectList(params)





```
{
  PageNumber: number,
  PageSize: number,
  Name: '',
  Label: Array,
  Type: 'Custom|Preset'
}
```

getAvatarList(type)



```
type = 'AR' | 'VR'
```

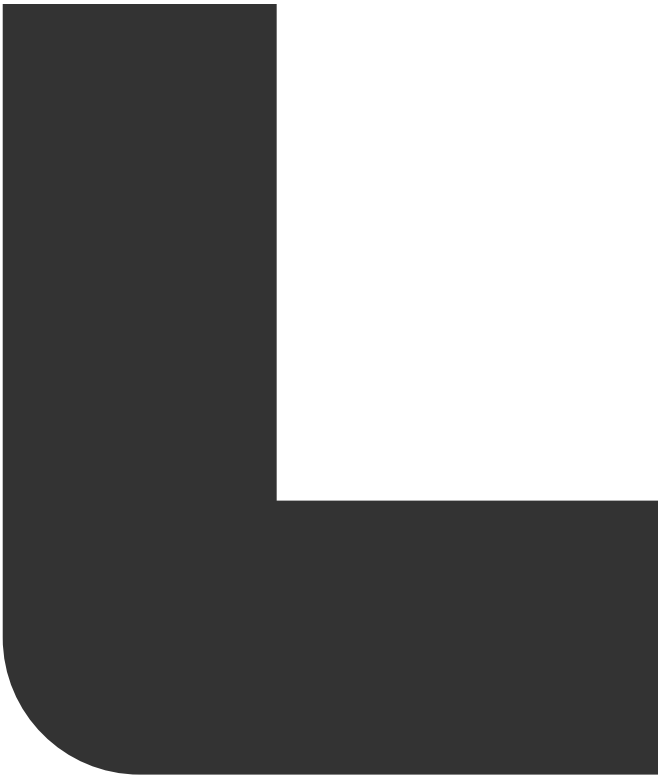
getEffect(effectId)

effectId: The effect ID

getCommonFilter()	-
async updateInputStream(src:MediaStream) (supported since v0.1.19)	src: New input stream (<code>MediaStream</code>)
disable()	-
enable()	-
async startRecord()	-

async stopRecord()





```
{
  useOriginAudio: boolean, // Whether to record t
  musicPath: string, // The background music URL,
}
```

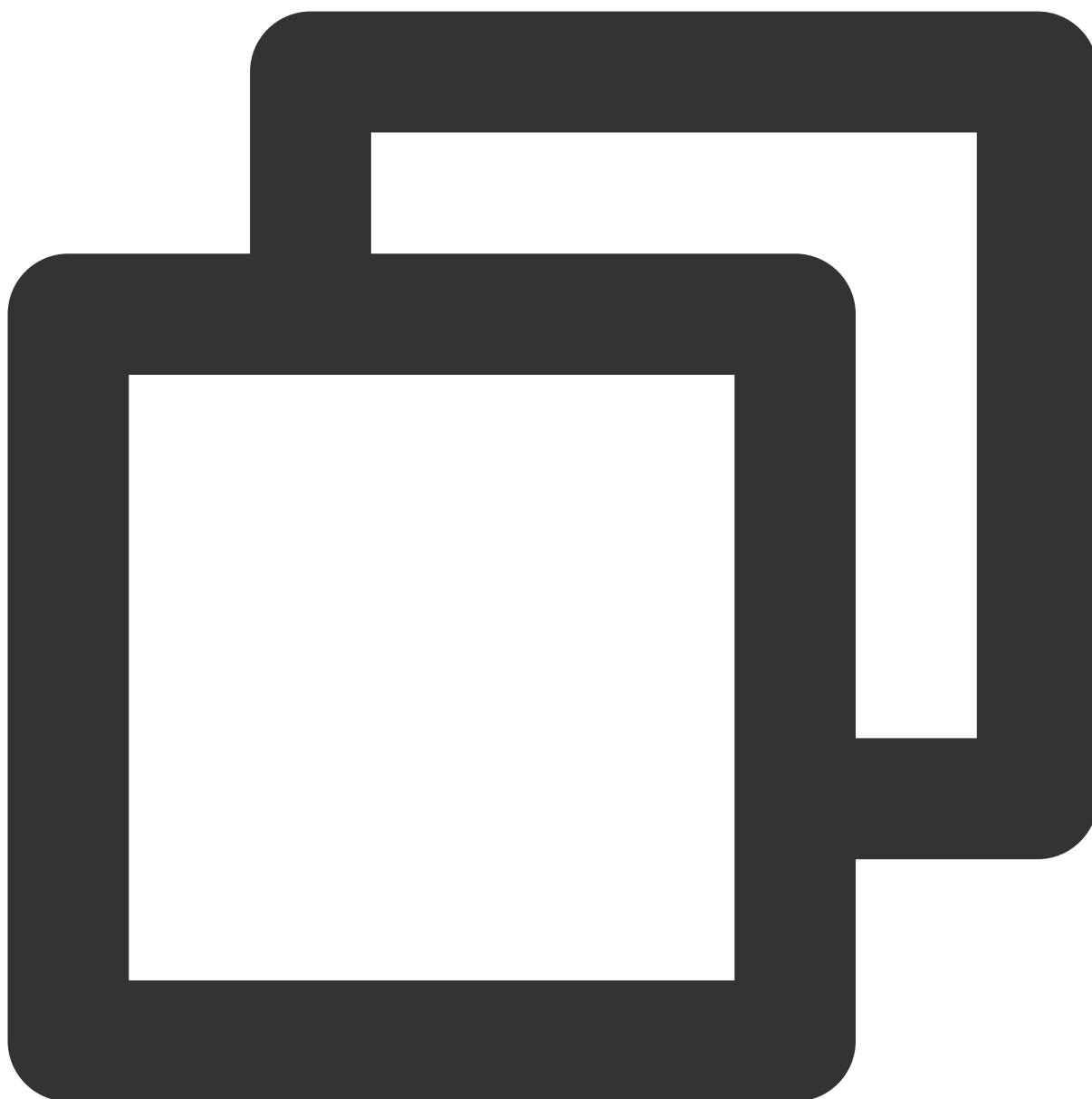
async takePhoto()

-

destroy()	-

Error Handling

The error object returned by the error callback includes the error code and error message, which facilitate troubleshooting.



```
sdk.on('error', (error) => {  
  // Handle an error in the error callback  
  const {code, message} = error  
  ...  
})
```

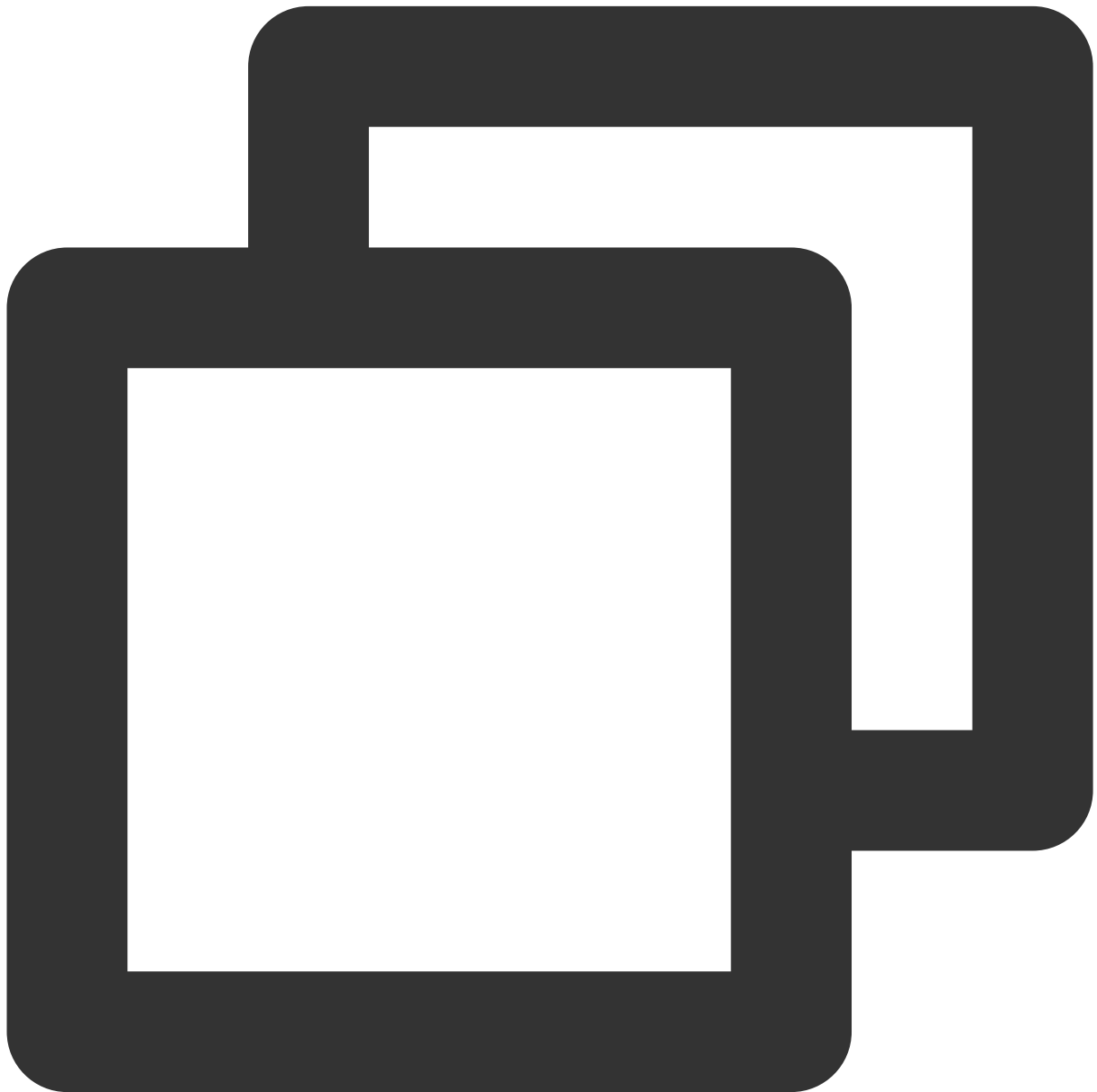
Error Code	Description	Remarks
10000001	The current browser environment is incompatible.	Recommend that the user use Chrome, Firefox, Safari, or the Weixin browser.

10000002	The render context is missing.	-
10000003	The rendering is slow.	Consider reducing the video resolution or disabling the feature.
10000005	An error occurred while parsing the input source.	-
10000006	Lag may occur due to insufficient browser support.	Recommend that the user use Chrome, Firefox, Safari, or the Weixin browser.
10001101	An error occurred while configuring the effect.	-
10001102	An error occurred while configuring the filter.	-
10001103	The effect strength parameter is incorrect.	-
10001201	Failed to turn on the camera.	-
10001202	The camera stopped.	-
10001203	Failed to get the camera permission.	The user needs to enable the camera permission by going to Settings > Privacy > Camera .
20002001	The authentication parameter is missing.	-
20001001	Authentication failed	Make sure you have created a license and the signature is correct.
20001002	The API request failed.	The error callback will return the data returned by the API. For details, see API Error Codes .
20001003	Failed to authenticate the effect configuration API.	The API is inaccessible. A Standard license cannot use the features of an Advanced license.
30000001	Failed to call <code>startRecord</code> in the mini program.	-
30000002	Failed to call <code>stopRecord</code> in the mini program.	-
40000000	An uncaught exception occurred.	-
40000001	As the current SDK version is too early, certain effects cannot be correctly displayed. Upgrade	-

	the SDK version.	
50000002	The effect is lost due to the resolution change.	The effect needs to be reconfigured.

Handling the missing render context error

On some computers, if the SDK is in the background for a long time, the `contextlost` error may occur. In such cases, you can call `ArSdk.prototype.resetCore(input: MediaStream)` to resume rendering.



```
sdk.on('error', async (error) => {  
  if (error.code === 10000002) {
```

```
        const newInput = await navigator.mediaDevices.getUserMedia({...})  
        await sdk.resetCore(newInput)  
    }  
})
```