Key & Value Tutorial

Overview

The **KeyManager** class provides a set of methods to access the request parameters and control behavior of hardware modules, including setting, obtaining, and listening to the values of keys and executing actions.

All keys supported by **KeyManager** can be defined and debugged through key-value pairs

1. AirLink Key

KeyALinkBandMode

Description: Transmission frequency band read/write

Request parameters

```
enum class AirLinkBandModeEnum (val value: Int) {

    /**
     * Location not obtained
     */
    BAND_MODE_UNKNOWN(0),

    /**
     * 2.4 GB
     */
    BAND_MODE_24G(1),

    /**
     * 5.2 GB
     */
    BAND_MODE_52G(2),

    /**
     * Auto switch between 2.4 GB and 5.2 GB
     */
    BAND_MODE_24_52G(3),

    /**
     * 5.8 GB
     */
    BAND_MODE_58G(4),

    /**
     * Auto switch between 2.4 GB and 5.8 GB
     */
    BAND_MODE_24_58G(5),

    /**
     * Auto switch between 5.2 GB and 5.8 GB
     */
```

```
    BAND_MODE_52_58G(6),

    /**
     * Auto switch between 2.4 GB, 5.2 GB, and 5.8 GB
     */
    BAND_MODE_24_52_58G(7);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyALinkTransmissionMode

Description: Transmission resolution mode

Request parameters

```
enum class VideoTransMissionModeEnum(val value: Int) {

    /**
     * Low latency
     */
    LOW_LATENCY(1),

    /**
     * High quality
     */
    HIGH_QUALITY(2),

    /**
     * 2.7K, Ultra HD
     */
    SUPER(3),

    /**
     * Unknown
     */
    UNKNOWN(255);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyALinkFccCeMode

Description: Radiated power mode

Request parameters

```
enum class FccCeModeEnum(val value: Int) {

    /**
     * CE (adaptive)
     */
    CE(0),

    /**
     * FCC (all-band)
     */
    FCC(1);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyALinkStartMatching

Description: Start of pairing

Request parameters: None

Type: **Action**

KeyALinkMatchingStatus

Description: Pairing progress reporting event

Request parameters: None

Response

```
enum class AirLinkMatchStatusEnum (val value: Int) {

    /**
     * Unknown
     */
    STATUS_UNKNOWN (0),

    /**
     * Pairing
     */
    STATUS_PAIRING (1),

    /**
     * Pairing completed
     */
    STATUS_SUC (2),

    /**
     * Pairing timed out or failed
     */
    STATUS_FAILED (3);
}
```

Type: **Listen**

KeyALinkMatchCostTime

Description: Reporting of pairing duration

Request parameters: None

Response

```
data class AirlinkMatchCostTimeBean(
    /**
     * 1: Remote controller, 2: Airplane
     */
    var deviceType: AirLinkDeviceTypeEnum = AirLinkDeviceTypeEnum.UNKNOWN,
    /**
     * Completion duration (in ms)
     */
    var costTime: Int = 0,
)

enum class AirLinkDeviceTypeEnum(val value: Int) {

    UNKNOWN(0),

    /**
     * Remote controller
     */
    REMOTE(1),

    /**
     * Aircraft
     */
    DRONE(2);
}
```

Type: **Listen**

KeyALinkSignalStrength

Description: Reporting of pairing signal strength

Request parameters: None

Response

```
data class AirLinkSignalStrengthBean(
    /**
     * Transmission interference strength
     */
    var interferenceSignalStrength: Int = 0,
    /**
     * Actual transmission signal strength
     */
    var realSignalStrength: Int = 0,
)
```

Type: **Listen**

KeyAirlinkResetMatchFlag

Description: Pairing reset

Request parameters: None

Response: Success or failure

Type: **Action**

KeyALinkConnectConfirm

Description: Connection confirmation

Request parameters: None

Response: None

Type: **Listen**

KeyAirlinkControlHighSpeed

Description: High-speed mode switchover

Request parameters

```
enum class HighSpeedEnum(val value: Int) {


    /**
     * Normal
     */
    NORMAL(0),

    /**
     * High-speed upload (mission file)
     */
    HIGH_UPLOAD(1),

    /**
```

```
     * High-speed download (photo/video download)
     */
    HIGH_DOWNLOAD(2),

    /**
     * Exclusive
     */
    EXCLUSIVE (3),

    /**
     * Exit
     */
    EXIT (4);
}
```

Response: Success or failure

Type: **Action**

KeyAirlinkGetHighSpeed

Description: High-speed mode acquisition

Request parameters: None

Response

```
enum class HighSpeedEnum(val value: Int) {


    /**
     * Normal
     */
    NORMAL(0),

    /**
     * High-speed upload (mission file)
     */
    HIGH_UPLOAD(1),

    /**
     * High-speed download (photo/video download)
     */
    HIGH_DOWNLOAD(2),

    /**
     * Exclusive
     */
    EXCLUSIVE (3),

    /**
     * Exit
     */
    EXIT (4);
}
```

Type: **Action**

KeyAirlinkSetDebugBandmode

Description: Frequency band configuration

Request parameters: **Int**

Response: Success or failure

Type: **Action**

KeyAirlinkSetDebugDynamicAdjust

Description: Dynamic adjustment of frequency bands

Request parameters

```
enum class DynamicAdjustEnum(val value: Int) {

    /**
     * Fixed frequency band
     */
    FIX(0),

    /**
     * Auto adaptation among compliant frequency bands
     */
    COMPLIANCE(1),

    /**
     * Auto adaptation among all frequency bands
     */
    ALL(2);
}
```

Response: Success or failure

Type: **Action**

KeyAirlinkEnterSilenceMode

Description: Entering transmission silent mode

Request parameters: None

Response: Success or failure

Type: **Action**

KeyAirlinkAppSplitScreenInfo

Description: Transmission display

Request parameters

```
data class GimbalTransmissionBean(
    /**
     * Gimbal label
     */
    var label: String? = null,
    /**
     * Camera ID (wide/IR/night, etc.)
     */
    var cameraId: Int = 0,

)
```

Response: Success or failure

Type: **Action**

2. Camera Key

KeyAELock

Description: AE Lock status

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAFMeterMode

Description: Auto focus mode

Request parameters

```
enum class AFLensFocusModeEnum(var value: Int) {
    /**
     * Average
     */
    AVERAGE(0),

    /**
     * Center
     */
    CENTER(1),

    /**
     * Point
     */
    SPOT(2),
    UNKNOWN(0xFF);
```

```
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAFState

Description: Reporting of AF status

Request parameters: None

Response:

```
enum class AFStateEnum(val value: Int) {
    /**
     * Start focusing
     */
    FOCUS_START(0),

    /**
     * Focusing
     */
    FOCUSING(1),

    /**
     * Focusing completed
     */
    FOCUS_COMPLETE(2);
}
```

Type: **Listen**

KeyAeAfStatusChange

Description: Reporting of AE/AF status change

Request parameters: None

Response:

```
data class CameraAFAEStatusBean(
    /**
     * AE status
     */
    var aeStatus: CameraAfAeEnum = CameraAfAeEnum.UNKNOWN,
    /**
     * AF status
     */
    var afStatus: CameraAfAeEnum = CameraAfAeEnum.UNKNOWN,
)
```

Type: **Listen**

KeyAeLockNtfyfy

Description: Reporting of AE Lock status

Request parameters: None

Response: **Boolean**

Type: **Listen**

KeyApertureMode

Description: Aperture mode

Request parameters

```
enum class ApertureModeEnum(var value: Int) {
    /**
     * Auto
     */
    AUTO(0),

    /**
     * Manual
     */
    MANUAL(1),

    /**
     * Unknown
     */
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyApertureSize

Description: Aperture size

Request parameters: **Double**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraAFAssistFocusEnable

Description: AF-assisted focus enabling

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraAntiflicker

Description: Anti-flicker mode

Request parameters

```
enum class AntiflickerEnum(val value: Int) {
    /**
     * Auto
     */
    AUTO(0),

    /**
     * 50 Hz
     */
    FIFTY_HZ(1),

    /**
     * 60 Hz
     */
    SIXTY_HZ(2),

    /**
     * Disable
     */
    CLOSE(3),

    /**
     * Unknown
     */
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraCapabilityVersion

Description: Camera capability set version

Request parameters: None

Response:

```
data class CameraCapabilityBean(
    /**
     * Camera capability set
     */
    val version: String,
    /**
     * File download path
     */
    val fileDownloadPath: String
)
```

Type: **Get**

KeyCameraDebugEvent

Description: Camera custom debugging

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraDehazeEnable

Description: Defog on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraDehazeStrength

Description: Defog level

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraDeviceInfo

Description: Camera information

Request parameters: None

Response:

```
data class DeviceInfoBean(
    /**
```

```
     * ID randomly generated by the system for the client to identify camera
connection. This ID remains the same before system reboot.
     */
    var identifyCode: Int = 0,
    /**
     * Device type
     */
    var deviceType: String? = null,
    /**
     * Device model
     */
    var deviceModel: String? = null,
    /**
     * Protocol version
     */
    var protocolVersion: Int = 0,
    /**
     * Manufacturer
     */
    var manufacturer: String? = null,
    /**
     * Hardware version
     */
    var firmwareVersion: String? = null,
    /**
     * SN
     */
    var serialNumber: String? = null,
    /**
     * Hardware ID
     */
    var hardwareId: String? = null,
    /**
     * Camera model
     */
    var lensModel: String? = null,
    /**
     * Camera software version
     */
    var lensSoftVersion: String? = null,
)
```

Type: **Get**

KeyCameraEisNtfy

Description: Reporting of electronic anti-shaking parameter

Request parameters: None

Response:

```
data class CameraEisReportBean(
    /**
     * Zoom rate
     */
```

```
    var zoomValue: Int = 0,
    /**
     * left angle
     */
    var leftAngle: Double = 0.0,
    /**
     * right angle
     */
    var rightAngle: Double = 0.0,
    /**
     * top angle
     */
    var topAngle: Double = 0.0,
    /**
     * bottom angle
     */
    var bottomAngle: Double = 0.0,
    /**
     * left value
     */
    var left: Double = 0.0,
    /**
     * right value
     */
    var right: Double = 0.0,
    /**
     * top value
     */
    var top: Double = 0.0,
    /**
     * bottom value
     */
    var bottom: Double = 0.0,
)
```

Type: **Listen**

KeyCameraElectronicAntiShaking

Description: Electronic anti-shaking on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraEnable

Description: Camera on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraEncryptProgressReportNtfy

Description: Reporting of decryption progress

Request parameters: None

Response:

```
data class CameraEncryptProgressReportBean(
    /**
     * Total
     */
    var totalCount: Int = 0,
    /**
     * Completed
     */
    var completeCount: Int = 0,
)
```

Type: **Listen**

KeyCameraFfc

Description: FFC shutter

Request parameters: None

Response: Success or failure

Type: **Action**

KeyCameraFocusSpotArea

Description: Focus point coordinate group

Request parameters

```
data class MeteringPointBean(
    /**
     * Metering point x-coordinate. Range: 0-100
     */
    var x: Int = 0,
    /**
     * Metering point y-coordinate. Range: 0-100
     */
    var y: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraGear

Description: Shooting mode

Request parameters

```
enum class CameraGearEnum(var value: Int) {
    /**
     * Auto
     */
    AUTO(0),

    /**
     * Manual
     */
    MANUAL(1),

    /**
     * Shutter Priority
     */
    SHUTTER_PRIORITY(2),

    /**
     * Aperture Priority
     */
    APERTURE_PRIORITY(3),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraImageStyleBrightness

Description: Image brightness

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraImageStyleContrast

Description: Image contrast

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraImageStyleHue

Description: Image hue

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraImageStyleSaturation

Description: Image saturation

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraImageStyleSharpness

Description: Image sharpness

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraImageStyleType

Description: Image type

Request parameters

```
enum class ImageStyleEnum(var value: Int) {
    /**
     * Standard
     */
    STANDARD(0),

    /**
     * Custom
     */
    CUSTOM(1),

    /**
     * Landscape
     */
    LANDSCAPE(2),

    /**
     * Soft
     */
    SOFT(3),
```

```
    /**
     * Fair
     */
    FAIR(4),

    /**
     * Contrast
     */
    CONTRAST(5),

    /**
     * Japan
     */
    JAPAN(6),

    /**
     * Moisture
     */
    MOISTURE(7),

    /**
     * Fresh
     */
    FRESH(8),

    /**
     * Gold Vibes
     */
    GOLDVIBES(9),

    /**
     * Gold Vibes 03
     */
    GOLDVIBES03(10),

    /**
     * Gold Vibes 04
     */
    GOLDVIBES04(11),

    /**
     * Gold Vibes 05
     */
    GOLDVIBES05(12),

    /**
     * BW
     */
    BW(13),

    /**
     * Nostalgic
     */
    NOSTALGIC(14),

    /**
     * Portrait
     */
```

```
    NATUREJ(15),

    /**
     * HK Movie 01
     */
    HKMOVIE01(16),

    /**
     * Retro
     */
    ANTIAUITY01(17),

    /**
     * Bright
     */
    BRIGHT(18),

    /**
     * Classic LUT
     */
    CLASSICLUT(19),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrEnhanceEnable

Description: IR image enhancement on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrEnhanceStrength

Description: IR image enhancement level

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrImageModeContrast

Description: IR image contrast

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrImageModeLum

Description: IR image brightness

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrImageModeType

Description: IR image mode

Request parameters

```
enum class ThermalImageModeEnum(var value: Int) {
    /**
     *  Manual (contrast/brightness)
     */
    MANUAL(0),

    /**
     * Auto 0
     */
    AUTO1(1),

    /**
     * Auto 1
     */
    AUTO2(2),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAlarmColdthred

Description: IR temperature warning property: low temperature threshold

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAlarmEnable

Description: IR temperature warning property on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAlarmHotthred

Description: IR temperature warning property: high temperature threshold

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAttrLimittemp

Description: IR temperature property: measurement limit

Request parameters

```
data class ParameterRectBean(
    /**
     *X-coordinate [0,100]
     */
    var x: Int = 0,
    /**
     *Y-coordinate [0,100]
     */
    var y: Int = 0,
    /**
     *Width [0,100]
     */
    var w: Int = 0,
    /**
     *Height [0,100]
     */
    var h: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAttrRegion

Description: IR temperature property: area temperature measurement

Request parameters

```
data class ParameterRectBean(
```

```
    /**
     *X-coordinate [0,100]
     */
    var x: Int = 0,
    /**
     *Y-coordinate [0,100]
     */
    var y: Int = 0,
    /**
     *Width [0,100]
     */
    var w: Int = 0,
    /**
     *Height [0,100]
     */
    var h: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAttrTouch

Description: IR temperature property: tap point coordinates

Request parameters

```
data class ParameterPointBean(
    /**
     *  X-coordinate [0,100]
     */
    var x: Int = 0,
    /**
     *  Y-coordinate [0,100]
     */
    var y: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraIrTempAttrType

Description: IR temperature property: temperature measurement

Request parameters

```
enum class TemperatureModeEnum(var value: Int) {
    /**
     *  None
     */
    NONE(0),
```

```
    /**
     * Center
     */
    CENTER(1),

    /**
     * Hot
     */
    HOT(2),

    /**
     * Cold
     */
    COLD(3),

    /**
     * Tap
     */
    TOUCH(4),

    /**
     * Area
     */
    REGION(5),

    /**
     * Track
     */
    TRACK(6),

    /**
     * User 1
     */
    USER1(7),

    /**
     * User 2
     */
    USER2(8),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraLinkageZoom

Description: Synced zoom on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraMFAssistFocusEnable

Description: MF-assisted focus enabling

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraMFObjectDistance

Description: MF object distance

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraMode

Description: Camera mode

Request parameters

```
data class CameraWorkModeInfoBean(
    /**
     *  Refer to CAMERA_WORK_MODE_TYPE.
     */
    var workMode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     *  Refer to CAMERA_TAKE_PHOTO_MODE_TYPE.
     */
    var photoMode: TakePhotoModeEnum = TakePhotoModeEnum.UNKNOWN,
    /**
     *  Refer to CAMERA_RECORD_MODE_TYPE.
     */
    var videoMode: RecordModeEnum = RecordModeEnum.UNKNOWN,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraModeSwitch

Description: Reporting of camera mode change

Request parameters: None

Response:

```
data class CameraModeSwitchBean(
    /**
     * Current mode
     */
    var currentMode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     * Recording sub-mode
     */
    var recordMode: Int = 0,
    /**
     * Photo sub-mode
     */
    var photoTakingMode: TakePhotoModeEnum = TakePhotoModeEnum.UNKNOWN,
)
```

Type: **Listen**

KeyCameraPivEnable

Description: PIV recording on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraPivInterval

Description: PIV interval

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraReboot

Description: Camera reboot

Request parameters: None

Response: Success or failure

Type: **Action**

KeyCameraRecordDuration

Description: Recording duration

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraRecordEnable

Description: Recording on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraRecordFps

Description: Recording frame rate

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraRecordNumber

Description: Recording count

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraReset

Description: Factory restore

Request parameters: None

Response: Success or failure

Type: **Action**

KeyCameraSaveMapTaskName

Description: Mission folder name

Request parameters: String

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraSaveMapUserDirName

Description: User-defined folder name

Request parameters: String

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraStatus

Description: Reporting of camera status messages

Request parameters: None

Response:

```
data class CameraStatusBean(
    /**
     * Device type
     */
    var deviceType: String? = null,
    /**
     *  System status
     */
    var systemStatus: SystemStatusEnum = SystemStatusEnum.UNKNOWN,
    /**
     *  Current mode
     */
    var currentMode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     * UI mode
     */
    var pattern: PatternModeEnum = PatternModeEnum.UNKNOWN,
    /**
     *  Display mode
     */
    var displayMode: DisplayModeEnum = DisplayModeEnum.UNKNOWN,
    /**
     *  Recording sub-mode
     */
    var recordMode: RecordModeEnum = RecordModeEnum.UNKNOWN,
    /**
     *  Photo sub-mode
     */
    var photoTakingMode: TakePhotoModeEnum = TakePhotoModeEnum.UNKNOWN,
    /**
     *  Storage type
     */
    var storageType: StorageTypeEnum = StorageTypeEnum.UNKNOWN,
    /**
     *  Horizontal FOV
     */
    var fovH: Float = 0f,
```

```
    /**
     *  Vertical FOV
     */
    var fovV: Float = 0f,
    /**
     * Pixel size
     */
    var pixelSize: Double = 0.0,
    /**
     * Focal length
     */
    var focalLength: Double = 0.0,
    /**
     * Min. photo interval, in ms
     */
    var photoIntervalMin: Int = 0,
)
```

Type: **Listen**

KeyCameraSubtitleKey

Description: Video subtitles on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraTransferInfoNtfy

Description: Reporting of transmission information

Request parameters: None

Response:

```
data class CameraTransferInfoBean(
    /**
     *  Camera ID
     */
    var cameraId: Int = 0,
    /**
     *  Transmission mode: 1 (low latency), 2 (high resolution), 3 (2.7 K)
     */
    var transferMode: Int = 0,
    /**
     *  Transmission encoding: 0 (H264), 1 (H265)
     */
    var transferPayloadType: Int = 0,
    /**
     *  Transmission frame rate (fps)
     */
    var fps: Int = 0,
```

```
    /**
     *  Transmission bitrate (kbps)
     */
    var bitrate: Int = 0,
    /**
     *  Sent keyframe count (per minute)
     */
    var sendIFrameNum: Int = 0,
    /**
     *  Requested keyframe count (per minute)
     */
    var requestIFrameNum: Int = 0,
    /**
     *   Transmission on/off
     */
    var transferEnable: Boolean = false,
)
```

Type: **Listen**

KeyCameraTransferPayloadType

Description: Video stream encoding (different from recording encoding)

Request parameters

```
enum class VideoCompressStandardEnum(var value: Int) {
    /**
     * video compress standard H264
     */
    H264(0),
    /**
     * video compress standard H265
     */
    H265(1),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraTurnOffArmLight

Description: Automatic deactivation of robotic arms during photo taking/recording

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraTypeEncryptEnable

Description: Camera data encryption on/off

Request parameters: None

Response: Boolean

Type: **Get**

KeyCameraTypeEncryptionKey

Description: Camera data encryption configuration

Request parameters

```
data class CameraEncryptSetBean(
    /**
     * Encryption type
     */
    var mode: CameraEncryptEnum = CameraEncryptEnum.CLOSE,
    /**
     * Password
     */
    var key: String = "",
)
```

Response: Success or failure

Type: **Action**

KeyCameraTypeXoomFixedFactor

Description: Fast zooming

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraUltraPixelEnable

Description: Photo/Recording night mode on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraVideoPictureStorageTypeGet

Description: Photo/Video storage type acquisition

Request parameters

```
data class CameraVideoPhotoStorageBean(
    /**
     * Type
     */
    var mode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     * Camera ID list
     */
    var ids: MutableList<Int> = mutableListOf(),
)
```

Response:

```
data class CameraVideoPhotoStorageListBean(
    /**
     * Type
     */
    var mode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     * Storage type list
     */
    var ids: MutableList<CameraVideoPhotoStorageInfoBean> = mutableListOf(),
)
```

Type: **Action**

KeyCameraVideoPictureStorageTypeSet

Description: Photo/Video storage type settings

Request parameters

```
data class CameraVideoPhotoStorageListBean(
    /**
     * Type
     */
    var mode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     * Storage type list
     */
    var ids: MutableList<CameraVideoPhotoStorageInfoBean> = mutableListOf(),
)
```

Response: Success or failure

Type: **Action**

KeyCameraVisualEnable

Description: Camera vision on/off

Request parameters: Boolean. **0**: On, **1**: Off

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraVisualTransfer

Description: Visual transmission

Request parameters: **Int**. **0**: Off, **1**: Front/Rear vision correction, **2**: Top/Bottom vision correction

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWatermarkSnEnable

Description: Stamp SN on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWatermarkGpsEnable

Description: Stamp GPS on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWhiteBalanceColorTemp

Description: White balance color temperature

Request parameters: **Int**. Range: 2000–10000

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWhiteBalanceType

Description: White balance mode

Request parameters

```
enum class WhiteBalanceEnum(var value: Int) {
    /**
     * Auto
```

```
     */
    AUTO(0),

    /**
     * Sunny
     */
    SUNNY(1),

    /**
     * Cloudy
     */
    CLOUDY(2),

    /**
     * Incandescent
     */
    INCANDESCENT(3),

    /**
     * Fluorescent
     */
    FLUORESCENT(4),

    /**
     * Custom. Range: 2000-10000, step: 100
     */
    CUSTOM(5),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWorkMode

Description: Camera mode

Request parameters

```
enum class CameraWorkModeEnum(var value: Int) {
    /**
     * Recording
     */
    RECORD(0),

    /**
     * Photo
     */
    PHOTO(1),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWorkModeTakePhoto

Description: Photo sub-mode

Request parameters

```
enum class TakePhotoModeEnum(var value: Int) {
    UNKNOWN(0xFF),

    /**
     * Single
     */
    SINGLE(0),

    /**
     * Burst
     */
    BUST(1),

    /**
     * Timer
     */
    INTERVAL(2),

    /**
     * AEB
     */
    AEB(3),

    /**
     * Hypersensitivity
     */
    HYPERSEN(4),

    /**
     * Panorama
     */
    PANORAMA(5),

    /**
     * Ultra HD matrix
     */
    MATRIX(6),

    /**
     * HDR
     */
    HDR(7),

    /**
     * Synced Zoom
     */
    LinkedZoom(8),

    /**
```

```
     * Night
     */
    NightShot(9);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCameraWorkModeVideo

Description: Recording sub-mode

Request parameters

```
enum class RecordModeEnum(var value: Int) {
    UNKNOWN(0xFF),

    /**
     * Standard
     */
    STANDARD(0),

    /**
     * Slow Motion
     */
    SLOW_MOTION(1),

    /**
     * Night Mode
     */
    ULTRA_PIXEL(2),

    /**
     * HDR
     */
    HDR(3),

    /**
     * Synced Zoom
     */
    LinkedZoom(4),

    /**
     * Super Night
     */
    SuperNightVideo(5);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyDefog

Description: Defog configuration

Request parameters

```
data class DefogBean(
    /**
     * On/Off
     */
    var enable: Boolean = false,
    /**
     * Defog level [1-10]
     */
    var strength: DefogEnum = DefogEnum.DEFOG_CLOSE
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyDelayShotStatus

Description: Reporting of timelapse status

Request parameters: None

Response:

```
data class DelayShotStatusBean(
    /**
     * Remaining time, in s
     */
    var remainTime: Int = 0,
    /**
     * Countdown to the photo, in s
     */
    var countDown: Int = 0,
    /**
     * Count of photos taken
     */
    var captured: Int = 0,
)
```

Type: **Listen**

KeyDisplayMode

Description: Display mode

Request parameters: None

Response:

```
enum class DisplayModeEnum(var value: Int) {
    /**
```

```
     * Optical
     */
    VISIBLE(0),

    /**
     * PiP
     */
    PICTURE(1),

    /**
     * IR
     */
    INFRARED(2),

    /**
     * Fusion
     */
    OVERLAP(3),
    UNKNOWN(0xFF);
}
```

Type: **Listen**

KeyDisplayModeSwitch

Description: Display switchover

Request parameters: None

Response:

```
enum class DisplayModeEnum(var value: Int) {
    /**
     * Optical
     */
    VISIBLE(0),

    /**
     * PiP
     */
    PICTURE(1),

    /**
     * IR
     */
    INFRARED(2),

    /**
     * Fusion
     */
    OVERLAP(3),
    UNKNOWN(0xFF);
}
```

Type: **Listen**

KeyFocusInfoMode

Description: Focus mode

Request parameters

```
enum class FocusModeEnum(var value: Int) {
    /**
     * focus mode of AF
     */
    AF(1),
    /**
     * focus mode of MF
     */
    MF(2),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyFocusNtfy

Description: Reporting of focus information

Request parameters: None

Response:

```
data class FocusInfoBean(
    /**
     * Focus mode
     */
    var mode: FocusModeEnum = FocusModeEnum.UNKNOWN,
    /**
     * AF
     */
    var meterMode: AFLensFocusModeEnum = AFLensFocusModeEnum.UNKNOWN,
    /**
     * Focus point x-coordinate (0 - 100)
     */
    var spotAreaX: Int = 0,
    /**
     * Focus point y-coordinate (0 - 100)
     */
    var spotAreaY: Int = 0,
    /**
     * Object distance (in mm), 3 segments: [ 10cm - 1m ], [ 1m - 10m ], [ 10m -
100m ]; 0: extremely close, -1: infinity
     */
    var objectDistance: Int = 0,
    /**
     * Assisted focus: 1-On, 0-Off
```

```
     */
    var aFAssistFocusEnable: Boolean = false,
```

Type: **Listen**

KeyFormatMmc

Description: MMC internal storage formatting

Request parameters: None

Response: Success or failure

Type: **Action**

KeyFormatSdCard

Description: SD card formatting

Request parameters: None

Response: Success or failure

Type: **Action**

KeyGetBaseParamMsg

Description: Basic camera parameter acquisition

Request parameters

```
data class CameraBaseParamBean(
    /**
     *  Storage type
     */
    var storageType: StorageTypeEnum = StorageTypeEnum.UNKNOWN,
    /**
     *  SD card status
     */
    var sdStatus: CardStatusBean? = null,
    /**
     *  Internal storage status
     */
    var mmcStatus: CardStatusBean? = null,
    /**
     *  Stamp
     */
    var waterMark: WatermarkBean? = null,
    /**
     *  HDR status
     */
    var hdrStatus: Boolean = false,
    /**
     *  Photo type
```

```
     */
    var picType: PhotoFormatEnum = PhotoFormatEnum.UNKNOWN,
    @ConvertIgnore
    /**
     * Photo resolution
     */
    var resolution: PhotoResolutionEnum = PhotoResolutionEnum.PR_UNKNOWN,
    /**
     * Burst count per second
     */
    var burstCount: Int = 0,
    /**
     * Timelapse photo interval, in s
     */
    var interval: Int = 0,
    /**
     * AEB photo count at a time
     */
    var aebCount: Int = 0,
    /**
     *  PIV status
     */
    var pivStatus: CameraRecordPivInfoBean? = null,
    /**
     *  File format
     */
    var fileFormat: VideoFormatEnum = VideoFormatEnum.UNKNOWN,
    /**
     * Subtitle on/off
     */
    var enableSubtitle: Boolean = false,
    /**
     * Recording resolution
     */
    var resolutionInfo: VideoResolutionBean? = null,
    /**
     * Video decoding
     */
    var encoding: VideoCompressStandardEnum = VideoCompressStandardEnum.UNKNOWN,
    /**
     * Image style
     */
    var imageStyleInfo: ImageStyleBean? = null,
    /**
     * AE Lock
     */
    var aeLock: Boolean = false,
    /**
     * Defog information
     */
    var dehazeInfo: DefogBean? = null,
    /**
     * Video standard
     */
    var videoStandard: VideoStandardEnum = VideoStandardEnum.UNKNOWN,
    /**
     * Anti-flicker
     */
```

```
    var antiFlicker: ResistanceBlinkEnum = ResistanceBlinkEnum.UNKNOWN,
    /**
     * Focus information
     */
    var focusInfo: FocusInfoBean? = null,
    /**
     * IR thermal color
     */
    var color: ThermalColorEnum = ThermalColorEnum.UNKNOWN,
    /**
     *  IR image mode
     */
    var irImageModeInfo: ThermalImageBean? = null,
    /**
     * Image enhancement
     */
    var irEnhanceInfo: ThermalEnhanceBean? = null,
    /**
     * Image denoiser
     */
    var irNr: Boolean = false,
    /**
     *  Image gain
     */
    var gain: ThermalGainEnum = ThermalGainEnum.UNKNOWN,
    /**
     * Isotherm
     */
    var irIsoThermInfo: ThermalIsothermBean? = null,
    /**
     * IR emissivity
     */
    var irTempEmit: Int = 0
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyHDR

Description: HDR on/off

Request parameters: **Bool**, **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyHDRNtfy

Description: Reporting of HDR information

Request parameters: None

Response: **Bool**, **false**: Not configured, **true**: Configured

Type: **Listen**

KeyISOMode

Description: ISO mode

Request parameters

```
enum class ISOModeEnum(var value: Int) {
    /**
     * Auto
     */
    AUTO(0),

    /**
     * Manual
     */
    MANUAL(1),
    /**
     * Unknown
     */
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyImageColor

Description: Image color parameter

Request parameters

```
enum class ImageColorEnum(var value: Int) {
    /**
     * None
     */
    NONE(0),

    /**
     * Log
     */
    LOG(1),

    /**
     * Vivid
     */
    VIVID(2),

    /**
     * BW
     */
    BLACK_WHITE(3),
```

```
    /**
     * Art
     */
    ART(4),

    /**
     * Film
     */
    FILM(5),

    /**
     * Beach
     */
    BEACH(6),

    /**
     * Dream
     */
    DREAM(7),

    /**
     * Classic
     */
    CLASSIC(8),

    /**
     * Nostalgic
     */
    NOSTALGIC(9),

    /**
     *
     */
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyImageExposure

Description: Image exposure parameter

Request parameters: **Double**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyImageIso

Description: Image ISO

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyImageStyle

Description: Image style

Request parameters

```
data class ImageStyleBean(
    /**
     *Image style
     */
    var style: ImageStyleEnum = ImageStyleEnum.UNKNOWN,
    /**
     *Brightness [-3, 3]
     */
    var brightness: Int = 0,
    /**
     *Contrast [-3, 3]
     */
    var contrast: Int = 0,
    /**
     *Saturation [-3, 3]
     */
    var saturation: Int = 0,
    /**
     *Hue [-3, 3]
     */
    var hue: Int = 0,
    /**
     *Sharpness [-3, 3]
     */
    var sharpness: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyInfraredCameraTempInfo

Description: Reporting of IR temperature information

Request parameters: None

Response:

```
data class InfraredCameraTempInfoBean(
    /**
     *  Average (°C), x10
     */
```

```
    var averageTemp: Int = 0,
    /**
     *  Center (˚C), x10
     */
    var centerTemp: Int = 0,
    /**
     *  Hot (˚C), x10
     */
    var hotTemp: Int = 0,
    /**
     *  Cold (˚C), x10
     */
    var coldTemp: Int = 0,
    /**
     *  Tap (˚C), x10
     */
    var touchTemp: Int = 0,
    /**
     * Hot point x-coordinate, image width ratio
     */
    var hotX: Int = 0,
    /**
     * Hot point y-coordinate, image height ratio
     */
    var hotY: Int = 0,
    /**
     * Cold point x-coordinate, image width ratio
     */
    var coldX: Int = 0,
    /**
     * Cold point y-coordinate, image height ratio
     */
    var coldY: Int = 0,
    /**
     * Actual x 100
     */
    var zoomValue: Int = 0,
)
```

Type: **Listen**

KeyIntervalShotStatus

Description: Reporting of timer countdown

Request parameters: None

Response: Int

Type: **Listen**

KeyLocationMeterInfo

Description: Reporting of metering point information

Request parameters: None

Response:

```
data class MeteringPointBean(
    /**
     * Metering point x-coordinate. Range: 0–100
     */
    var x: Int = 0,
    /**
     * Metering point y-coordinate. Range: 0–100
     */
    var y: Int = 0,
)
```

Type: **Listen**

KeyMMCStatus

Description: MMC internal storage status

Request parameters: None

Response:

```
data class CardStatusBean(
    /**
     *  Internal storage status
     */
    var storageStatus: CardStatusEnum = CardStatusEnum.UNKNOWN,
    /**
     *  Total internal storage, in KB
     */
    var totalSpace: Long = 0L,
    /**
     *  Remaining internal storage, in KB
     */
    var freeSpace: Long = 0L,
    /**
     * Remaining recording time, in s
     */
    var remainRecordTime: Long = 0L,
    /**
     * Remaining photo quota
     */
    var remainCaptureNum: Long = 0L,
)
```

Type: **Get**

KeyMmcStatusInfo

Description: Reporting of MMC status

Request parameters: None

Response:

```
data class CardStatusBean(
    /**
     *  Internal storage status
     */
    var storageStatus: CardStatusEnum = CardStatusEnum.UNKNOWN,
    /**
     *  Total internal storage, in KB
     */
    var totalSpace: Long = 0L,
    /**
     *  Remaining internal storage, in KB
     */
    var freeSpace: Long = 0L,
    /**
     * Remaining recording time, in s
     */
    var remainRecordTime: Long = 0L,
    /**
     * Remaining photo quota
     */
    var remainCaptureNum: Long = 0L,
)
```

Type: **Listen**

KeyMeteringPoint

Description: Metering point

Request parameters

```
data class MeteringPointBean(
    /**
     * Metering point x-coordinate. Range: 0-100
     */
    var x: Int = 0,
    /**
     * Metering point x-coordinate. Range: 0-100
     */
    var y: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyMissionRecordWaypoint

Description: Reporting of waypoint information for mission recording

Request parameters: None

Response:

```
data class MissionRecordWaypointBean(
    /**
     * Path
     */
    var filePath: String? = null,
    /**
     * Latitude
     */
    var latitude: Double = 0.0,
    /**
     * Longitude
     */
    var longitude: Double = 0.0,
    /**
     * Absolute altitude
     */
    var altitude: Double = 0.0,
    /**
     * Aircraft heading
     */
    var heading: Double = 0.0,
    /**
     * Gimbal pitch
     */
    var cameraPitch: Double = 0.0,
    /**
     * Gimbal yaw
     */
    var cameraYaw: Double = 0.0,
    /**
     * Zoom rate
     */
    var zoom: Int = 0,
    /**
     * Gimbal roll
     */
    var cameraRoll: Double = 0.0,
    /**
     * Relative altitude
     */
    var height: Double = 0.0,
)
```

Type: **Listen**

KeyMotionDelayShotStatus

Description: Reporting of motion timelapse status

Request parameters: None

Response:

```
data class MotionDelayShotBean(

    /**
     * Status
     */
    var status: MotionDelayShootEnum = MotionDelayShootEnum.UNKNOWN,
    /**
     * Time consumed, in s
     */
    var photoTime: Int = 0,
    /**
     * Total time, in s
     */
    var totalPhotoTime: Int = 0,
    /**
     * Photos taken
     */
    var photoNum: Int = 0,
    /**
     * Total photos needed
     */
    var totalPhotoNum: Int = 0,
)
```

Type: **Listen**

KeyPanoramaStatus

Description: Reporting of panorama status

Request parameters: None

Response:

```
data class PanoramaStatusBean(
    /**
     * Panorama status
     */
    var state: PanoramaShootStatusEnum = PanoramaShootStatusEnum.UNKNOWN,
    /**
     * Panorama type
     */
    var type: PanoramaEnum = PanoramaEnum.UNKNOWN,
    /**
     * Total steps
     */
    var totalStep: Int = 0,
    /**
     * Current steps
     */
    var currentStep: Int = 0,
    /**
     * Completion rate
     */
    var proportion: Float = 0f,
```

```
)
```

Type: **Listen**

KeyPatternMode

Description: Camera pattern

Request parameters

```
enum class PatternModeEnum(var value: Int) {
    /**
     * Manual flight
     */
    MANUAL(0),

    /**
     * Mission flight
     */
    MISSION_FLY(1),

    /**
     * Smart tracking, gesture identification, pinpoint
     */
    TRACK(2),

    /**
     * Timelapse
     */
    DELAY_SHOOT(3),

    /**
     * Orbit
     */
    VISION_ORBIT(4),

    /**
     * Panoramic shooting
     */
    PANORAMIC(5),

    /**
     * Mission recording
     */
    MISSION_RECORDER(6),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyPhotoExposure

Description: Reporting of camera exposure status

Request parameters: None

Response:

```
enum class ExposureEnum(var value: Int) {
    /**
     * Overexposure
     */
    OVEREXPOSURE(0),

    /**
     * Underexposure
     */
    UNDEREXPOSURE(1);

}
```

Type: **Listen**

KeyPhotoFileFormat

Description: Photo format

Request parameters

```
enum class PhotoFormatEnum(var value: Int) {
    /**
     * JPG format of image captured
     */
    JPG(0),
    /**
     * DNG format of image captured
     */
    DNG(1),
    /**
     * JPGDNG format of image captured
     */
    JPGDNG(2),
    /**
     * RJPEG format of image captured
     */
    RJPEG(3),
    /**
     * RJPEGTIFF format of image captured
     */
    RJPEGTIFF(4),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyPhotoFileInfo

Description: Reporting of photo information

Request parameters

```
data class PhotoFileInfoBean constructor(
    /**
     * Current mode
     */
    var currentMode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     * File path
     */
    var filePath: String? = null,
    /**
     * Thumbnail path
     */
    var thumbnailPath: String? = null,
    /**
     * Photo modification time
     */
    var fileModifyTime: Long = 0,
    /**
     * Photo size
     */
    var fileSize: Long = 0,
    /**
     * Photo format
     */
    var fileFormat: Int = 0,
    /**
     * Photo resolution
     */
    var photoResolution: PhotoResolutionBean? = null
)
```

Response: Request parameters will be returned for **Listen** requests.

Type: **Set**, **Listen**

KeyPhotoResolutionNtfy

Description: Reporting of photo resolution

Request parameters: None

Response:

```
enum class PhotoResolutionEnum(val width: Int, val height: Int, val tag:
String,val prId:String) {
    PR_4000_3000(4000, 3000, "12M","4000x3000"),
    PR_1920_1200(1920, 1200, "1200P","1920x1200"),
    PR_1920_1080(1920, 1080, "1080","1920x1080"),
    PR_4096_3072(4096, 3072, "4K","4096x3072"),
    PR_8192_6144(8192, 6144, "8K","8192x6144"),
    PR_640_512(640, 512, "640*512","640x512"),
    PR_3840_2160(3840, 2160, "4K","3840x2160"),
    PR_5472_3648(5472, 3648, "6K","5472x3648"),
    PR_5472_3076(5472, 3076, "6K","5472x3076"),
    PR_UNKNOWN(0, 0, "unknown","unknown");
}
```

Type: **Listen**

KeyProfessionalParamInfo

Description: Reporting of professional parameter information

Request parameters: None

Response:

```
data class ProfessionalParamInfoBean(
    /**
     * camera ID
     */
    var cameraID: Int = 0,
    /**
     * ISO mode
     */
    var isoMode: ISOModeEnum = ISOModeEnum.AUTO,
    /**
     * Aperture mode
     */
    var apertureMode: ApertureModeEnum = ApertureModeEnum.AUTO,
    /**
     * Shutter mode
     */
    var shutterMode: ShutterModeEnum = ShutterModeEnum.AUTO,
    /**
     * ISO value
     */
    var isoValue: Int = 0,
    /**
     *  Aperture value
     */
    var apertureValue: Double = 0.0,
    /**
     *  Shutter speed
     */
    var shutterSpeed: ShutterSpeedBean? = null,
    /**
     *  exposure value
```

```
     */
    var exposureValue: Double = 0.0,
    /**
     * White balance mode
     */
    var whiteMode: WhiteBalanceEnum = WhiteBalanceEnum.UNKNOWN,
    /**
     *  White balance color temperature value
     */
    var colorTemperature: Int = 0,
    /**
     *  Digital zoom rate. Actual value x 100
     */
    var zoomValue: Int = 0,
)
```

Type: **Listen**

KeyROI

Description: ROI configuration

Request parameters

```
data class ROIBean(
    /**
     *On/Off
     */
    var enable: Boolean = false,
    /**
     *ROI configuration. Up to 8 ROIs can be supported.
     */
    var roiRegionList: MutableList<RoiRegionBean>? = null,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRecordFileEncodeFormat

Description: Recording encoding

Request parameters

```
enum class VideoCompressStandardEnum(var value: Int) {
    /**
     * Video compression standard H264
     */
    H264(0),
    /**
     * video compression standard H265
     */
    H265(1),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRecordFileFormat

Description: Recording file type

Request parameters

```
enum class VideoFormatEnum(var value: Int) {
    /**
     * video format defines MOV
     */
    MOV(0),
    /**
     * video format defines MP4
     */
    MP4(1),
    /**
     * video format defines TIFF
     */
    TIFF(2),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRecordFileInfo

Description: Reporting of recording information

Request parameters: None

Response:

```
data class RecordFileInfoBean(
    /**
     * Current mode
     */
```

```
    var currentMode: RecordStatusEnum = RecordStatusEnum.UNKNOWN,
    /**
     * Status
     */
    var state: RecordVideoFileEnum = RecordVideoFileEnum.UNKNOWN,
    /**
     * File path
     */
    var filePath: String? = null,
    /**
     * Thumbnail path
     */
    var thumbnailPath: String? = null,
    /**
     * Recording modification time
     */
    var fileModifyTime: Long = 0L,
    /**
     * File size
     */
    var fileSize: Long = 0L,
    /**
     * File format
     */
    var fileFormat: Int = 0,
    /**
     * Duration
     */
    var videoDuration: Int = 0,
    /**
     * Resolution
     */
    var videoResolution: VideoResolutionBean? = null,
    /**
     * Compression standard. For details, see
CAMERA_VIDEO_COMPRESSION_STANDARD_TYPE.
     */
    var videoCompressionStandard: VideoCompressStandardEnum =
VideoCompressStandardEnum.UNKNOWN,
)
```

Type: **Listen**

KeyRecordPacket

Description: Recording packet size

Request parameters

```
data class RecordPacketBean(
    /**
     *Current storage file system format
     */
    var format: StorageTypeEnum = StorageTypeEnum.UNKNOWN,
    /**
     *Recording file packet size (MB). If the storage format is SD card FAT 32 or EMMC FAT 32, the size is fixed at 3500 MB; if it is SD card exFat, the size can be set as required (0: no packet; default: 3500 MB).
     */
    var recordPacket: Int = 0
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Set**, **Get**

KeyRecordParameters

Description: Recording parameters

Request parameters

```
data class RecordParametersBean(
    /**
     *  File format
     */
    var fileFormat: VideoFormatEnum = VideoFormatEnum.UNKNOWN,
    /**
     * Saves audio stream during recording
     */
    var enableAudio: Boolean = false,
    /**
     * Image resolution
     */
    var enableSubtitle: Boolean = false,
    /**
     * Enables automatic photo during recording
     */
    var enablePiv: Boolean = false,
    /**
     * Photo interval
     */
    var interval: Int = 0,
    /**
     * Slow motion recording resolution
     */
    var resolutionInfo: VideoResolutionBean? = null,
    /**
     * Maximum storage time of loop recording, in s
     */
    var maxRecordTime: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Set**, **Get**

KeyRecordPiv

Description: PIV recording status

Request parameters

```
data class CameraRecordPivInfoBean(
    /**
     * On/Off
     */
    var enable: Boolean = false,
    /**
     * Interval
     */
    var interval: Int = 0
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Set**, **Get**

KeyRecordResolution

Description: Recording resolution

Request parameters

```
data class VideoResolutionBean(
    /**
     * Resolution list
     */
    var videoResolutionEnum: VideoResolutionEnum =
VideoResolutionEnum.VR_UNKNOWN,
    /**
     * Frame rate
     */
    var fps: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Set**, **Get**

KeyRecordStatus

Description: Reporting of recording status

Request parameters: None

Response:

```
data class RecordStatusBean(
    /**
     *Current mode. Not set
     */
    var currentMode: CameraWorkModeEnum = CameraWorkModeEnum.UNKNOWN,
    /**
     *Recording status
     */
    var state: RecordStatusEnum = RecordStatusEnum.UNKNOWN,
    /**
     *Recording duration, in s
     */
    var currentRecordTime: Int = 0
)
```

Type: **Listen**

KeyResetCameraState

Description: Reporting of camera reset status

Request parameters: None

Response: **Boolean**. **true**: success, **false**: failure

Type: **Listen**

KeyROINtfy

Description: Reporting of ROI configuration

Request parameters: None

Response:

```
data class ROIBean(
    /**
     *On/Off
     */
    var enable: Boolean = false,
    /**
     *ROI configuration. Up to 8 ROIs can be configured.
     */
    var roiRegionList: MutableList<RoiRegionBean>? = null,
)
```

Type: **Listen**

KeySdCardStatus

Description: Reporting of SD card status

Request parameters: None

Response:

```
data class CardStatusBean(
    /**
     *  SD card status
     */
    var storageStatus: CardStatusEnum = CardStatusEnum.UNKNOWN,
    /**
     *  Total SD card storage, in KB
     */
    var totalSpace: Long = 0L,
    /**
     *  Remaining SD card storage, in KB
     */
    var freeSpace: Long = 0L,
    /**
     * Remaining recording time, in s
     */
    var remainRecordTime: Long = 0L,
    /**
     * Remaining photo count
     */
    var remainCaptureNum: Long = 0L,
)
```

Type: **Listen**

KeyShutterMode

Description: Shutter mode

Request parameters

```
enum class ShutterModeEnum(var value: Int) {
    /**
     * Auto
     */
    AUTO(0),

    /**
     * Manual
     */
    MANUAL(1),
    /**
     * Unknown
     */
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyShutterSpeed

Description: Shutter speed

Request parameters

```
data class ShutterSpeedBean(
    /**
     *Numerator
     */
    var numerator: Int = 0,
    /**
     *Denominator
     */
    var denominator: Int = 0
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeySmoothZoom

Description: Smooth zoom

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyStartRecord

Description: Start of recording

Request parameters: None

Response: Success or failure

Type: **Action**

KeyStartTakePhoto

Description: Start of photo taking

Request parameters: None

Response: Success or failure

Type: **Action**

KeyStopRecord

Description: Stop of recording

Request parameters: None

Response: Success or failure

Type: **Action**

KeyStopTakePhoto

Description: Stop of photo taking

Request parameters: None

Response: Success or failure

Type: **Action**

KeyStorageStatus

Description: SD card status

Request parameters: None

Response:

```
data class CardStatusBean(
    /**
     *  SD card status
     */
    var storageStatus: CardStatusEnum = CardStatusEnum.UNKNOWN,
    /**
     *  Total SD card storage, in KB
     */
    var totalSpace: Long = 0L,
    /**
     *  Remaining SD card storage, in KB
     */
    var freeSpace: Long = 0L,
    /**
     * Remaining recording time, in s
     */
    var remainRecordTime: Long = 0L,
    /**
     * Remaining photo count
     */
    var remainCaptureNum: Long = 0L,
)
```

Type: **Get**, **Listen**

KeyStorageStatusInfo

Description: Reporting of storage status

Request parameters: None

Response:

```
data class StorageStatusInfoBean(
    /**
     * Storage type
     */
    var storageType: StorageTypeEnum = StorageTypeEnum.UNKNOWN,
    /**
     * SD card status
     */
    var sDCardStatus: CardStatusEnum = CardStatusEnum.UNKNOWN,
    /**
     * Total SD card storage, in KB
     */
    var totalSpace: Long = 3,
    /**
     * Remaining SD card storage, in KB
     */
    var freeSpace: Long = 4,
    /**
     * MMC internal status
     */
    var mMCStatus: CardStatusEnum = CardStatusEnum.UNKNOWN,
    /**
     * Total MMC internal storage, in KB
     */
    var mMCTotalSpace: Long = 6,
    /**
     * Remaining MMC internal storage, in KB
     */
    var mMCFreeSpace: Long = 7,
    /**
     * Remaining recording time of the selected storage type, in s
     */
    var remainRecordTime: Long = 8,
    /**
     * Remaining photo count of the selected storage type
     */
    var remainCaptureNum: Long = 9,
)
```

Type: **Listen**

KeyStorageType

Description: Storage type

Request parameters

```
enum class StorageTypeEnum(var value: Int) {
    UNKNOWN(0xFF),
    /**
     * SD card
     */
    SD(0),

    /**
     * MMC
     */
    EMMC(1);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTakePhotoAebCount

Description: AEB photo count

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTakePhotoBustCount

Description: Burst count

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTakePhotoParameters

Description: Photo parameters

Request parameters

```
data class TakePhotoParametersBean(

    /**
     * Photo type
     */
    var picType: PhotoFormatEnum = PhotoFormatEnum.UNKNOWN,
    /**
     * Photo resolution
     */
    @ConvertIgnore
    var resolution: PhotoResolutionEnum = PhotoResolutionEnum.PR_UNKNOWN,
```

```
    /**
     * Timer countdown
     */
    var delaySeconds: Int = 0,
    /**
     * Gimbal roll step
     */
    var gimbalRollingStep: Int = 0,
    /**
     * Gimbal roll angle
     */
    var gimbalRollingDirection: GimbalRotateDirectionEnum =
GimbalRotateDirectionEnum.CLOCKWISE,
    /**
     * HDR on/off
     */
    var enableHDR: Boolean = false,
    /**
     * Burst count per second
     */
    var burstCount: Int = 0,
    /**
     * Timelapse photo interval, in s
     */
    var interval: Int = 0,
    /**
     * Timelapse duration
     */
    var duration: Int = 0,
    /**
     *  Timelapse video clip on/off
     */
    var composeVideo: Boolean = false,
    /**
     *  Timelapse video resolution
     */
    var resolutionInfo: VideoResolutionBean? = null,
    /**
     * AEB photo count at a time
     */
    var aebCount: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTakePhotoResolution

Description: Photo resolution

Request parameters

```
enum class PhotoResolutionEnum(val width: Int, val height: Int, val tag:
String,val prId:String) {
    PR_4000_3000(4000, 3000, "12M","4000x3000"),
    PR_1920_1200(1920, 1200, "1200P","1920x1200"),
    PR_1920_1080(1920, 1080, "1080","1920x1080"),
    PR_4096_3072(4096, 3072, "4K","4096x3072"),
    PR_8192_6144(8192, 6144, "8K","8192x6144"),
    PR_640_512(640, 512, "640*512","640x512"),
    PR_3840_2160(3840, 2160, "4K","3840x2160"),
    PR_5472_3648(5472, 3648, "6K","5472x3648"),
    PR_5472_3076(5472, 3076, "6K","5472x3076"),
    PR_UNKNOWN(0, 0, "unknown","unknown");
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTakePhotoStatus

Description: Reporting of photo status

Request parameters: None

Response:

```
data class TakePhotoStatusBean(
    /**
     *Current mode
     */
    var currentMode: TakePhotoModeEnum = TakePhotoModeEnum.UNKNOWN,
    /**
     *Photo status
     */
    var state: TakePhotoStatusEnum = TakePhotoStatusEnum.UNKNOWN,
)
```

Type: **Listen**

KeyTakePhotoTimeLapse

Description: Timelapse photo interval

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTempAlarm

Description: Reporting of temperature warning events

Request parameters: None

Response:

```
data class TempAlarmBean(
    /**
     * 0: high temperature; 1: low temperature
     */
    var status: TempEnum = TempEnum.HOT,
    /**
     * High temperature warning threshold (°C), x10
     */
    var hotTemp: Int = 0,
    /**
     * High temperature warning threshold x-coordinate, image width ratio
     */
    var hotX: Int = 0,
    /**
     * High temperature warning threshold y-coordinate, image height ratio
     */
    var hotY: Int = 0,
    /**
     * Low temperature warning threshold (°C), x10
     */
    var coldTemp: Int = 0,
    /**
     * Low temperature warning threshold x-coordinate, image width ratio
     */
    var coldX: Int = 0,
    /**
     * Low temperature warning threshold y-coordinate, image height ratio
     */
    var coldY: Int = 0,
)
```

Type: **Listen**

KeyThermalColor

Description: IR thermal color information

Request parameters

```
enum class ThermalColorEnum(var value: Int) {
    /**
     *  White hot
     */
    WHITE_HOT(0),

    /**
     * Black hot
     */
    BLACK_HOT(1),

    /**
```

```
     * Rainbow
     */
    RAINBOW(2),

    /**
     * Rainbow-HC
     */
    RAIN_HC(3),

    /**
     * Iron Bow
     */
    IRON_BOW(4),

    /**
     * Lava
     */
    LAVA(5),

    /**
     * Arctic
     */
    ARCTIC(6),

    /**
     * Glow
     */
    GLOW_BOW(7),

    /**
     * Graded
     */
    GRADED_FIRE(8),

    /**
     * Thermal
     */
    HOTTEST(9),

    //    DOUBLE_RAINBOW(11),//Double rainbow
//    COLOR_RING(12),//Color ring
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalDenoising

Description: IR image denoiser configuration

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalEnhance

Description: IR image enhancement configuration

Request parameters

```
data class ThermalEnhanceBean(
    /**
     * IR image enhancement on/off
     */
    var enable: Boolean = false,
    /**
     * Level: 1-8
     */
    var strength: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalGain

Description: IR image gain

Request parameters

```
enum class ThermalGainEnum(var value: Int) {
    /**
     * High gain
     */
    HIGH(0),

    /**
     * Low gain
     */
    LOW(1),

    /**
     * Auto
     */
    AUTO(2),
    UNKNOWN(0xFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalIsotherm

Description: IR isotherm

Request parameters

```
data class ThermalIsothermBean(
    /**
     * Mode
     */
    var mode: IsothermEnum = IsothermEnum.UNKNOWN,
    /**
     *Upper threshold in auto mode
     */
    var hotThred: Int = 0,
    /**
     *Lower threshold in auto mode
     */
    var coldThred: Int = 0
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalMode

Description: IR image mode

Request parameters

```
data class ThermalImageBean(
    /**
     * Image mode
     */
    var imageMode: ThermalImageModeEnum = ThermalImageModeEnum.UNKNOWN,
    /**
     * Contrast: 0-255
     */
    var contrast: Int = 0,
    /**
     * Brightness: 0-511
     */
    var lum: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalRadiance

Description: IR emissivity

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalTemperature

Description: IR temperature property

Request parameters

```
data class ThermalTempAttrBean(
    /**
     * Temperature mode
     */
    var tempMode: TemperatureModeEnum = TemperatureModeEnum.UNKNOWN,
    /**
     * Tap point x-coordinate, image width ratio
     */
    var touchX: Int = 0,
    /**
     * Tap point y-coordinate
     */
    var touchY: Int = 0,
    /**
     * Region temperature measurement, x-coordinate
     */
    var regionX: Int = 0,
    /**
     * Region temperature measurement, y-coordinate
     */
    var regionY: Int = 0,
    /**
     * Region temperature measurement, width
     */
    var regionW: Int = 0,
    /**
     * Region temperature measurement, height
     */
    var regionH: Int = 0,
    /**
     * Measurement limit, x-coordinate
     */
    var limitTempX: Int = 0,
    /**
     * Measurement limit, y-coordinate
     */
    var limitTempY: Int = 0,
    /**
     * Measurement area width limit
     */
    var limitTempW: Int = 0,
    /**
     * Measurement area height limit
     */
    var limitTempH: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThermalTemperatureAlarm

Description: IR temperature warning property

Request parameters

```
data class ThermalTempAlarmBean(
    /**
     * Warning on/off
     */
    var enable: Boolean = false,
    /**
     * High temperature warning (°C), x10
     */
    var hotThred: Int = 0,
    /**
     * Low temperature warning (°C), x10
     */
    var coldThred: Int = 0,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyVideoEncoderConfig

Description: Video encoding configuration

Request parameters

```
data class VideoEncoderConfigBean(
    /**
     *  Stream ID, 0-primary, 1-secondary
     */
    var streamId: Int = 0,
    /**
     * Video encoding
     */
    var encoding: VideoCompressStandardEnum = VideoCompressStandardEnum.UNKNOWN,
    /**
     * Video resolution
     */
    var resolution: VideoResolutionBean? = null,
    /**
     * QP value
     */
    var quality: Int = 0,
    /**
     * I-frame interval
```

```
     */
    var govLength: Int = 0,
    /**
     * Compression level
     */
    var profile: VideoCompressProfileEnum = VideoCompressProfileEnum.UNKNOWN,
    /**
     * Bitrate type
     */
    var bitrateType: VideoBitrateEnum = VideoBitrateEnum.UNKNOWN,
    /**
     * Bitrate
     */
    var bitrate: Int = 0,
    /**
     *  High frame rate on/off
     */
    var slow: Boolean = false,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyVideoResolutionNtfy

Description: Reporting of video resolution

Request parameters: None

Response:

```
data class VideoResolutionBean(
    /**
     * Resolution list
     */
    var videoResolutionEnum: VideoResolutionEnum =
VideoResolutionEnum.VR_UNKNOWN,
    /**
     * Frame rate
     */
    var fps: Int = 0,
)
```

Type: **Listen**

KeyVideoSourceConfig

Description: Video source configuration

Request parameters

```
data class VideoSourceConfigBean(
    /**
```

```
     *  Video rotation
     */
    var rotation: Int = 0,
    /**
     * Video standard
     */
    var videoStandard: VideoStandardEnum = VideoStandardEnum.UNKNOWN,
    /**
     * Anti-flicker
     */
    var antiFlicker: ResistanceBlinkEnum = ResistanceBlinkEnum.UNKNOWN,
    /**
     * 3D denoiser on/off
     */
    var enable3DNR: Boolean = false,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyWatermark

Description: Stamp

Request parameters

```
data class WatermarkBean(
    /**
     * Stamp on/off. 0: Off, 1: On
     */
    var enableTime: Int = 0,
    /**
     * Custom stamp
     */
    var enableCustom: Boolean = false,
    /**
     * Custom stamp content. Only letters, digits, and punctuations are
supported.
     */
    var custom: String? = null,
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyWhiteBalance

Description: White balance parameter

Request parameters

```
data class WhiteBalanceBean(
    /**
     * White balance mode
     */
    var mode: WhiteBalanceEnum = WhiteBalanceEnum.UNKNOWN,
    /**
     * Takes effect when the mode is set to Custom. Range: 2000–10000. Step: 100
     */
    var colorTemperature: Int = 0
)
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyWhiteBalanceNtfy

Description: Reporting of white balance parameter

Request parameters: None

Response:

```
data class WhiteBalanceBean(
    /**
     * White balance mode
     */
    var mode: WhiteBalanceEnum = WhiteBalanceEnum.UNKNOWN,
    /**
     * Takes effect when the mode is set to Custom. Range: 2000–10000. Step: 100
     */
    var colorTemperature: Int = 0
)
```

Type: **Listen**

KeyZoomFactor

Description: Digital/IR zoom

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

3. Common Key

KeyHeartBeatPhone

Description: Aircraft heartbeat

Request parameters: None

Response: None

Type: **Listen**

KeyHeartBeatApp

Description: App heartbeat

Request parameters: None

Response: None

Type: **Listen**

KeySetSystemDataTime

Description: System time configuration

Request parameters

```
data class SystemTimeInfoBean(
    /**
     * UTC timestamp
     */
    var utcTimeInterval: Long = 0,
    /**
     * Time zone
     */
    var timeZone: Float = 0f,
)
```

Response: Success or failure

Type: **Action**

KeyGetSystemInitData

Description: Initial system data acquisition

Request parameters

```
data class SystemInfoData(

//    var droneVersion: String? = null, // Aircraft package (system
initialization data has been removed)

    /**
     * Main version
     */
    var mainControlVersion: String? = null,

    /**
```

```
     * Aircraft SN
     */
    var droneSN: String? = null,

    /**
     * Aircraft type
     */
    var droneType: Int = 0,

    /**
     * Activation status
     */
    var activeState: Int = 0,

    /**
     * Flight mode. For details, see DRONE_FLIGHT_MODE.
     */
    var flightMode: DroneFlightModeEnum = DroneFlightModeEnum.UNKNOWN,

    /**
     * Flight control main mode. For details, see FLIGHT_CONTROL_MAIN_MODE.
     */
    var droneMainMode: FlightControlMainModeEnum =
FlightControlMainModeEnum.UNKNOWN,

    /**
     * Work mode. For details, see SMART_MODE.
     */
    var droneWorkMode: DroneWorkModeEnum = DroneWorkModeEnum.UNKNOWN,

    /**
     * Operating status. For details, see DRONE_WORK_STATUS.
     */
    var droneWorkStatus: DroneWorkStateEnum = DroneWorkStateEnum.UNKNOWN,

)
```

Response: Success or failure

Type: **Action**

KeyGetDroneDevicesInfo

Description: Aircraft information acquisition

Request parameters

```
data class DroneVersionItemBean(
    /**
     * After the capability set is built, this field is set by device
registration order. The IDs of the EVO Max Series are fixed, for example,
front/rear-view radar IDs.
     */
    var componentID: DroneComponentIdEnum = DroneComponentIdEnum.NONE,
    /**
     * Component SN
```

```
     */
    var componentSN: String? = null,
    /**
     * Hardware version
     */
    var hardwareVersion: String? = null,
    /**
     * Software version
     */
    var softwareVersion: String? = null,
    /**
     * Component model (integer)
     */
    var componentModel: Int = 0,  //
    /**
     * Component model (string, for example, XM802)
     */
    var strComponentModel: String? = null,
    /**
     * Reserved field, used to describe the current component. Currently
meaningless for the EVO Max Series.
     */
    var label: String? = null,
    /**
     * Reserved field, used to describe the current component type. Currently
meaningless for the EVO Max Series.
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,
    /**
     * MCU bootloader version
     */
    var bootloaderVersion: String? = null,
)
```

Response: Success or failure

Type: **Action**

KeyDroneSystemStatusHFNtfy

Description: Reporting of general aircraft parameter (5 Hz)

Request parameters: None

Response:

```
data class DroneSystemStateHFNtfyBean(
    /**
     * Latitude. When its value is invalid or when isGPSValid is set to NO,
positioning fails.
     */
    var droneLatitude: Double = 0.0,
    /**
     * Longitude. When its value is invalid or when isGPSValid is set to NO,
positioning fails.
     */
```

```
    var droneLongitude: Double = 0.0,
    /**
     * Relative altitude
     */
    var altitude: Float = 0f,
    /**
     * Absolute altitude
     */
    var altitudeMSL: Float = 0f,
    /**
     * X velocity
     */
    var velocityX: Float = 0f,
    /**
     * Y velocity
     */
    var velocityY: Float = 0f,
    /**
     * Z velocity
     */
    var velocityZ: Float = 0f,
    /**
     * Aircraft attitude
     */
    var droneAttitude: DroneAttitudeBean? = null,
    /**
     * Gimbal attitude
     */
    var gimbalAttitude: DroneAttitudeBean? = null,
    /**
     * Distance between aircraft and home point
     */
    var distance: Float = 0f,
    /**
     * Ultrasound altitude AGL
     */
    var ultrasoundAltitude: Float = 0f,
    /**
     * Laser range (cm)
     */
    var laserDistance: Int = 0,
    /**
     * Whether the laser range is valid. 0-no, 1-yes
     */
    var laserDistanceIsValid: Boolean = false,
    /**
     * Radar height
     */
    var radarHight: Double = 0.0,
    /**
     * Flight mode
     */
    var flightMode: DroneFlightModeEnum = DroneFlightModeEnum.UNKNOWN,
    /**
     * Flight control main mode. For details, see FLIGHT_CONTROL_MAIN_MODE.
     */
    var mainMode: FlightControlMainModeEnum = FlightControlMainModeEnum.UNKNOWN,
    /**
```

```
     * Smart flight mode. For details, see FlightControlDefine > DroneWorkMode.
     */
    var droneWorkMode: DroneWorkModeEnum = DroneWorkModeEnum.UNKNOWN,
    /**
     * Aircraft operating status
     */
    var droneWorkStatus: DroneWorkStateEnum = DroneWorkStateEnum.UNKNOWN,
    /**
     * Track status. 0: Exit tracking, 1: Checking, 2: Target tracking (moving
gimbal, still aircraft), 3: Planed tracking (moving gimbal and aircraft)
     */
    var trackWorkStatus: TrackWorkStateEnum = TrackWorkStateEnum.EXIT_FOLLOW,
    /**
     * GPS on/off (false: off, true: on)
     */
    var isGpsOpen: Boolean = false
)
```

Type: **Listen**

KeyDroneSystemStatusLFNtfy

Description: Reporting of general aircraft parameter (2 Hz)

Request parameters: None

Response:

```
data class DroneSystemStateLFNtfyBean(
    /**
     * Remaining flight time
     */
    var remainingFlightTime: Float = 0f,
    /**
     * Remaining battery level
     */
    var batteryRemainingPower: Int = 0,
    /**
     * Battery percentage
     */
    var batteryPercentage: Float = 0f,
    /**
     * Battery temperature
     */
    var batteryTemperature: Float = 0f,
    /**
     * Battery voltage
     */
    var batteryVoltage: Float = 0f,
    /**
     * Battery current
     */
    var batteryCurrent: Float = 0f,
    /**
     * Number of visible satellites
     */
```

```
    var satelliteCount: Int = 0,
    //Obsolete
    /**
     * GPS level. For details, see FlightControlDefine.
     */
    var gpsLevel: GPSLevelEnum = GPSLevelEnum.LEVEL_NONE,
    /**
     * Remote ID status. 1: On, 2: Off
     */
    var remoteIdStatus: Boolean = false,
    /**
     * SLAM confidence or visual positioning signal. High: > 0.7; Medium: 0.4-
0.7; Low: < 0.4
     */
    var slamConfidence: Double = 0.0,
    /**
     * GPS signal strength. Weak: < 40%; Medium: 40%-80%; Strong: > 80%
     */
    var gpsStrengthPercentage: Int = 0,
)
```

Type: **Listen**

KeyDroneWorkStatusInfoReport

Description: Reporting of flight status

Request parameters: None

Response:

```
data class FlightControlStatusInfo(
    /**
     * Aircraft type
     */
    var droneType: Int = 0,
//    var flightMode: DroneFlightModeEnum = DroneFlightModeEnum.UNKNOWN,//Flight
mode
//    var mainMode: FlightControlMainModeEnum =
FlightControlMainModeEnum.UNKNOWN,//Flight main mode. For details, see
FLIGHT_CONTROL_MAIN_MODE.
//    var droneWorkMode: DroneWorkModeEnum = DroneWorkModeEnum.UNKNOWN,//Smart
flight mode. For details, see FlightControlDefine > DroneWorkMode.
//    var droneWorkStatus: DroneWorkStateEnum =
DroneWorkStateEnum.UNKNOWN,//Aircraft work status
    /**
     * Aircraft mode. For details, see GEAR_LEVEL.
     */
    var droneGear: GearLevelEnum = GearLevelEnum.UNKNOWN,
    /**
     * Whether to support RTK
     */
    var supportRTK: Boolean = false,
    /**
     * Whether the home point is invalid
     */
```

```
    var isHomeInvalid: Boolean = false,
    /**
     * Home point latitude
     */
    var homeLatitude: Double = 0.0,
    /**
     * Home point longitude
     */
    var homeLongitude: Double = 0.0,
    /**
     * Enabling OA
     */
    var obstacleAvoidanceEnabled: Boolean = false,
    /**
     * Radar display
     */
    var radarChartDispalyed: Boolean = false,
//    var trackWorkStatus: Int = 0,//Track status
    /**
     * Downward auxiliary light status
     */
    var visionledStatus: Int = 0,
    /**
     * Landing protection on/off
     */
    var visionEnableSecureLanding: Boolean = false,
    /**
     * Precision landing on/off
     */
    var visionEnablePreciseLanding: Boolean = false,
    /**
     * Compass calibration status
     */
    var compassCalibrationStatus: CompassCalibrationStatusEnum =
CompassCalibrationStatusEnum.NONE,
    /**
     * IMU calibration status
     */
    var imuCalibrationStatus: IMUCalibrationStepEnum =
IMUCalibrationStepEnum.NONE,
    /**
     * Number of battery discharges
     */
    var numberOfDischarge: Int = 0,
    /**
     * Whether the battery is in place. 0: Yes 1: No
     */
    var batteryNotInPlaceFlag :Int= 0,
    /**
     * Surrounding brightness. 0: Sufficient, 1: Insufficient
     */
    var environmentInfo :EnvironmentEnum= EnvironmentEnum.NORMAL_BRIGHTNESS
)
```

Type: **Listen**

KeyDroneWarningMFNtfy

Description: Reporting of aircraft warnings

Request parameters: None

Response:

```
data class DroneWarningStateNtfyBean(
    /**
     * Low battery level
     */
    var lowPowerWarning: BatteryPowerWarning = BatteryPowerWarning.NONE,
    /**
     * Whether the battery is overheated
     */
    var isBatteryOverheated: Boolean = false,
    /**
     * Whether the battery is too cold
     */
    var isBatteryLowTemperature: Boolean = false,
    /**
     * Battery voltage imbalance
     */
    var isBatteryVoltageDiff: Boolean = false,
    /**
     * Power-on alerts. For details, see FLIGHT_CONTROL_TURN_ON_MOTOR_STATUS.
     */
    var turnOnMotorError: FlightControlTurnOnMotorStatusEnum =
FlightControlTurnOnMotorStatusEnum.NO_ERROR,
    /**
     * IMU overheating
     */
    var isIMUOverheated: Boolean = false,
    /**
     * Stick limit
     */
    var isStickLimited: Boolean = false,
    /**
     * Gales
     */
    var highWindWarning: Boolean = false,
    /**
     * Inaccurate return point
     */
    var isHomePointNotAccurate: Boolean = false,
    /**
     * Overload
     */
    var overloadWarning: Boolean = false,
    /**
     * IMU calibration
     */
    var imuCalibrationWarn: IMUCalibratoionWarningEnum =
IMUCalibratoionWarningEnum.NONE,
    /**
     * No-fly zone
     */
```

```
    var noFlyStatus: MainControllerNoflyZoneStatusEnum =
MainControllerNoflyZoneStatusEnum.FLY_STATUS_UNKNOW,
    /**
     * Compass interference
     */
    var compassInterferenceLevel: CompassInterferenceLevelEnum =
CompassInterferenceLevelEnum.LEVEL0,
    /**
     * Whether to reach the max. range
     */
    var isReachMaxRange: Boolean = false,
    /**
     * Whether to reach near to the max. range
     */
    var isNearMaxRange: Boolean = false,
    /**
     * Whether to reach max. height
     */
    var isReachMaxHeight: Boolean = false,
    /**
     * Whether the GPS signal is valid
     */
    var isGPSValid: Boolean = false,
    /**
     * Whether the compass is valid
     */
    var isCompassValid: Boolean = false,
    /**
     * Whether the vision is limited
     */
    var visionLimitedWhenDark: Boolean = false,
    /**
     * Insufficient space
     */
    var insufficientSpaceToPass: Boolean = false,
    /**
     * Move the aircraft to an open area.
     */
    var flyToOpenSpace: Boolean = false,
    /**
     * All vision errors, for example, whether the vision sensor works properly.
     */
    var visionErrorCode: Int = 0,
)
```

Type: **Listen**

KeyRCHardwareState

Description: Reporting of fixed frequency

Request parameters: None

Response:

```
data class RCHardwareStateNtfyBean(
```

```
    /**
     * Left stick horizontal value
      */
    var leftHorizontalValue: Int = 0,

    /**
     * Left stick vertical value
      */
    var leftVerticalValue: Int = 0,

    /**
     * Right stick horizontal value
      */
    var rightHorizontalValue: Int = 0,

    /**
     * Right stick vertical value
      */
    var rightVerticalValue: Int = 0,

    /**
     * Pitch value 1
      */
    var wheelValue1: Int = 0,

    /**
     * Pitch value 2
      */
    var wheelValue2: Int = 0,

    /**
     * Flight mode. For details, see RemoteControlDefine > RC_FLIGHT_MODE
(GPS/ATTI/IOC).
      */
    var flightMode: RcFlightModeEnum = RcFlightModeEnum.UNKNOWN,
    /**
     * Remote controller define > RC_BUTTON_TYPE
     */
    var buttonType: RCButtonTypeEnum = RCButtonTypeEnum.UNKNOWN,

    /**
     * Combined buttons
      */
    var combinedButtonType: Int = 0,
)
```

Type: **Listen**

KeyRCHardwareInfo

Description: Reporting of RC hardware buttons

Request parameters: None

Response:

```
data class HardwareButtonInfoBean(
    /**
     * RC pad button type state
     */
    var buttonType: RCButtonTypeEnum = RCButtonTypeEnum.UNKNOWN,
    /**
     * RC pad button clicked type state
     */
    var clickType: RcButtonStateEnum = RcButtonStateEnum.UNKNOWN,
    /**
     *Key value, indicating the value difference of dial wheels or zoom levels
[0-255]
     */
    var thumbWheelValue: Int = 0
)
```

Type: **Listen**

KeyRCState

Description: Reporting of RC status

Request parameters: None

Response:

```
data class RCStateNtfyBean(
    /**
     *RC signal strength [0-100]
     */
    var rcSignalQuality: Int = 0,
    /**
     *Image transmission signal strength [0-100]
     */
    var dspSignalQuality: Int = 0,
    /**
     *Remaining battery level [0-100]
     */
    var remainPowerPercent: Int = 0,
    /**
     *RC calibration status (If not used, enter 0)
     */
    var calibrationState: Int = 0,
    /**
     *RC battery temperature, in °C
     */
    var batteryTemperature: Int = 0,
    /**
     *RC input voltage, in mV
     */
    var chargingVoltage: Int = 0,
    /**
     *Nest mode. 0: No, 1: Yes
     */
    var isDroneNestMode: Int = 0,
```

```
    )
```

Type: **Listen**

KeyRCRockerCalibrationState

Description: Reporting of RC calibration

Request parameters: None

Response:

```
data class RockerCalibrationStateNtfyBean(
    /**
     * Right stick calibration status
     */
    var rightStickStatus: Int = 0,
    /**
     * Left stick calibration status
     */
    var leftStickStatus: Int = 0,
    /**
     * Right dial wheel calibration status
     */
    var rightThumbWheelStatus: Int = 0,
    /**
     * Left dial wheel calibration status
     */
    var leftThumbWheelStatus: Int = 0,
    /**
     * 1: centered, 0: sticks or dial wheels not centered
     */
    var midValibFlag: Int = 0,
) {

    /**
     * Left stick calibration status
     * @param directionEnum direction enum v
     alue
     * @return return gimbal left stick state
     */
    fun getLeftStickState(directionEnum: RcDirectionEnum): GimbalCalState {
        return when (directionEnum) {
            RcDirectionEnum.CENTER -> {
                GimbalCalState.findEnum(leftStickStatus and 0x3)
            }
            RcDirectionEnum.RIGHT -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 2)) shr 2)
            }
            RcDirectionEnum.RIGHT_TOP -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 4)) shr 4)
            }
            RcDirectionEnum.TOP -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 6)) shr 6)
            }
```

```
            RcDirectionEnum.LEFT_TOP -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 8)) shr 8)
            }
            RcDirectionEnum.LEFT -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 10)) shr
10)
            }
            RcDirectionEnum.LEFT_BOTTOM -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 12)) shr
12)
            }
            RcDirectionEnum.BOTTOM -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 14)) shr
14)
            }
            RcDirectionEnum.RIGHT_BOTTOM -> {
                GimbalCalState.findEnum((leftStickStatus and (0x3 shl 16)) shr
16)
            }
            else -> {
                GimbalCalState.findEnum(leftStickStatus and 0x3)
            }
        }
    }

    /**
     * Right stick calibration status
     * @param directionEnum direction enum value
     * @return return gimbal right stick state
     */
    fun getRightStickState(directionEnum: RcDirectionEnum): GimbalCalState {
        return when (directionEnum) {
            RcDirectionEnum.CENTER -> {
                GimbalCalState.findEnum(rightStickStatus and 0x3)
            }
            RcDirectionEnum.RIGHT -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 2)) shr
2)
            }
            RcDirectionEnum.RIGHT_TOP -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 4)) shr
4)
            }
            RcDirectionEnum.TOP -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 6)) shr
6)
            }
            RcDirectionEnum.LEFT_TOP -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 8)) shr
8)
            }
            RcDirectionEnum.LEFT -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 10)) shr
10)
            }
            RcDirectionEnum.LEFT_BOTTOM -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 12)) shr
12)
```

```
            }
            RcDirectionEnum.BOTTOM -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 14)) shr
14)
            }
            RcDirectionEnum.RIGHT_BOTTOM -> {
                GimbalCalState.findEnum((rightStickStatus and (0x3 shl 16)) shr
16)
            }
            else -> {
                GimbalCalState.findEnum(rightStickStatus and 0x3)
            }
        }
    }

    /**
     * Left dial wheel calibration status
     * @param directionEnum direction enum value
     * @return return gimbal left thumb wheel state
     */
    fun getLeftThumbWheelState(directionEnum: RcDirectionEnum): GimbalCalState {
        return when (directionEnum) {
            RcDirectionEnum.ROLL_LEFT -> {
                GimbalCalState.findEnum(leftThumbWheelStatus and 0x3)
            }
            RcDirectionEnum.ROLL_RIGHT -> {
                GimbalCalState.findEnum((leftThumbWheelStatus and (0x3 shl 4))
shr 4)
            }
            else -> {
                GimbalCalState.findEnum((leftThumbWheelStatus and (0x3 shl 2))
shr 2)
            }
        }
    }

    /**
     * Right dial wheel calibration status
     * @param directionEnum direction enum value
     * @return return gimbal right thumb wheel state
     */
    fun getRightThumbWheelState(directionEnum: RcDirectionEnum): GimbalCalState
{
        return when (directionEnum) {
            RcDirectionEnum.ROLL_LEFT -> {
                GimbalCalState.findEnum(rightThumbWheelStatus and 0x3)
            }
            RcDirectionEnum.ROLL_RIGHT -> {
                GimbalCalState.findEnum((rightThumbWheelStatus and (0x3 shl 4))
shr 4)
            }
            else -> {
                GimbalCalState.findEnum((rightThumbWheelStatus and (0x3 shl 2))
shr 2)
            }
        }
    }
}
```

```
enum class RcDirectionEnum(val value: Int) {
    /**
     * Center
     */
    CENTER(0),

    /**
     * Right
     */
    RIGHT(1),

    /**
     * Top right
     */
    RIGHT_TOP(2),

    /**
     * Top
     */
    TOP(3),

    /**
     * Top left
     */
    LEFT_TOP(4),

    /**
     * Left
     */
    LEFT(5),

    /**
     * Bottom left
     */
    LEFT_BOTTOM(6),

    /**
     * Bottom
     */
    BOTTOM(7),

    /**
     * Bottom right
     */
    RIGHT_BOTTOM(8),

    /**
     * Dial wheel center
     */
    ROLL_CENTER(9),

    /**
     * Dial wheel right
     */
    ROLL_RIGHT(10),

    /**
```

```
     * Dial wheel left
     */
    ROLL_LEFT(11),

    /**
     * Unknown
     */
    UNKNOWN(-1);
}
```

Type: **Listen**

KeyControlLed

Description: Control of an aircraft indicator

Request parameters

```
data class DroneLedStatusBean(
    /**
     *  Indicator type. For details, see DRONE_LED_TYPE.
     */
    var ledType: DroneLedTypeEnum = DroneLedTypeEnum.NONE,
    /**
     *  Indicator status. For details, see DRONE_LED_STATUS.
     */
    var ledStatus: DroneLedStatusEnum = DroneLedStatusEnum.CLOSE
)
```

Response: Success or failure

Type: **Action**

KeyQueryLedStatus

Description: Query of all aircraft indicators

Request parameters: None

Response:

```
data class DroneAllLedStatusBean(
    /**
     *Indicator status. For details, see DRONE_LED_STATUS.
     */
    var ledsStatus: List<DroneLedStatusBean>? = null,
    /**
     * Strobe on/off. 0: Off, 1: On
     */
    var nightLedStatus: Boolean = false
)
```

Type: **Action**

KeyDroneCalibrationCommand

Description: General calibration commands

Request parameters

```
data class CalibrationCommandBean(
    /**
     * Calibration type
     */
    var type: CalibrationTypeEnum = CalibrationTypeEnum.UNKNOWN,
    /**
     * Calibration instructions
     */
    var cmd: CmdEnum = CmdEnum.UNKNOWN,
)
```

Response: Success or failure

Type: **Action**

KeyDroneCalibrationEventNtfy

Description: Calibration event notifications

Request parameters: None

Response:

```
data class CalibrationEventBean(
    /**
     * Calibration type
     */
    var calibrationType: CalibrationTypeEnum = CalibrationTypeEnum.UNKNOWN,
    /**
     * Calibration event
     */
    var calibrationEvent: CalibrationEventEnum = CalibrationEventEnum.UNKNOWN,
)
```

Type: **Listen**

KeyDroneCalibrationScheduleNtfy

Description: Calibration progress notifications

Request parameters: None

Response:

```
data class CalibrationScheduleBean(
    /**
```

```
     * Used when calibrationType is set to IMU
     */
    var imcStep: ImuCalibrationStepEnum = ImuCalibrationStepEnum.STEP0,
    /**
     * Used when calibrationType is set to COMPASS
     */
    var compassStep: CompassCalibrationStepEnum =
CompassCalibrationStepEnum.STEP0,
    /**
     * Calibration progress
     */
    var calibrationPercent: Int = 0,
    /**
     * Calibration type. For details, see CALIBRATION_TYPE.
     */
    var calibrationType: CalibrationTypeEnum = CalibrationTypeEnum.UNKNOWN,
)
```

Type: **Listen**

KeyDroneControlNightNavigationLed

Description: Strobe control

Request parameters: Boolean

Response: Success or failure

Type: **Action**

KeyDroneVersionNtfy

Description: Aircraft message notifications

Request parameters: None

Response:

```
data class DroneVersionItemBean(
    /**
     * After the capability set is built, this field is set by device
registration order. The IDs of the EVO Max Series are fixed, for example,
front/rear-view radar IDs.
     */
    var componentID: DroneComponentIdEnum = DroneComponentIdEnum.NONE,
    /**
     * component serial number
     */
    var componentSN: String? = null,
    /**
     * hardware version
     */
    var hardwareVersion: String? = null,
    /**
     * software version
```

```
     */
    var softwareVersion: String? = null,
    /**
     * Component model (integer)
     */
    var componentModel: Int = 0,  //
    /**
     * Component model (string, for example, XM802)
     */
    var strComponentModel: String? = null,
    /**
     * Reserved field, used to describe the current component type. Currently
meaningless for the EVO Max Series.
     */
    var label: String? = null,
    /**
     * Reserved field, used to describe the current component. Currently
meaningless for the EVO Max Series.
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,
    /**
     * MCU bootloader version
     */
    var bootloaderVersion: String? = null,
)
```

Type: **Listen**

KeyDroneEventNtfy

Description: Aircraft event notifications

Request parameters: None

Response:

```
data class EventInfoBean(
    /**
     * Component type. For details, see VersionMessage.proto >
DRONE_COMPONENT_TYPE.
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,
    /**
     * Component ID. For details, see CommonHead.proto > DRONE_COMPONENT_ID.
     */
    var componentId: DroneComponentIdEnum = DroneComponentIdEnum.NONE,
    /**
     * Exception priority. For details, see ABNORMAL_PRIORITY.
     */
    var priority: EventPriorityEnum = EventPriorityEnum.UNKNOWN,
    /**
     * Event ID
     */
    var eventId: DroneEventEnum = DroneEventEnum.UNKNOWN,
)
```

Type: **Listen**

KeyDroneTempConnectNtfy

Description: Reporting of temporary device connections, which can be received when the aircraft is disconnected.

Request parameters: None

Response:

```
data class DeviceTempConnectBean(
    /**
     * Aircraft SN
     */
    var droneSn: String? = null,
    /**
     * Drone battery percentage
     */
    var droneBatteryPercentage: Float = 0f,
    /**
     * Whether the aircraft has been paired. 0: No, 1: Yes
     */
    var isMatched: Boolean = false,
    /**
     * Custom aircraft name
     */
    var droneName: String? = null
)
```

Type: **Listen**

KeyDroneUtcTimeSyncNtfy

Description: GPS UTC synchronization

Request parameters: None

Response:

```
data class DroneUTCTimeSyncBean(
    /**
     * UTC time
     */
    var gpsTime: Long = 0L
)
```

Type: **Listen**

KeyDroneSetCountryCode

Description: Country code

Request parameters: String

Response: Success or failure

Type: **Action**

KeyRCBandInfoTypeNtfy

Description: RC type notifications

Request parameters: None

Response:

```
data class RCBandInfoTypeBean(
    /**
     *String of up to 63 characters
     */
    var bandMode: String? = null,
    /**
     *String of up to 63 characters
     */
    var bandWidth: String? = null,
    /**
     *District value
     */
    var district: Int = 0,
    /**
     * disable set band mode value
     */
    var disableSetBandMode: Int = 0,
    /**
     * RC type integer
     */
    var RCType: Int = 0,
    /**
     * Modem mode
     */
    var modemMode: Int = 0,
    /**
     * FCC mode
     */
    var fccMode: Int = 0,
)
```

Type: **Listen**

KeyDroneCleanNoflyZone

Description: Clearance of no-fly zone files

Request parameters

```
enum class CleanNoFlyZoneEnum(var value: Int) {
    /**
```

```
     *  Unknown
     */
    UNKNOWN(0),

    /**
     *  Clear all files
     */
    ALL_FILE(1),

    /**
     *  Clear temporary no-fly zone files
     */
    TEMPORARY_NOFLY(2),

    /**
     *  Clear authorized zone files
     */
    AUTHORIZED_NOFLY(3),

    /**
     *  Clear e-defense files
     */
    ELECTRIC_FENCE_NOFLY(4);
}
```

Response: Success or failure

Type: **Action**

KeyDroneWarning

Description: Aircraft warning notifications

Request parameters: None

Response:

```
data class WarningAtom(

    /**
     * Component type
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,

    /**
     * Component ID
     */
    var componentId: DroneComponentIdEnum = DroneComponentIdEnum.NONE,

    /**
     * Warning ID
     */
    var warningId: WaringIdEnum = WaringIdEnum.UNKNOWN,
)
```

Type: **Listen**

KeyDroneRuntimeWarning

Description: Aircraft real-time warning notifications

Request parameters: None

Response:

```
data class WarningAtom(

    /**
     * Component type
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,

    /**
     * Component ID
     */
    var componentId: DroneComponentIdEnum = DroneComponentIdEnum.NONE,

    /**
     * Warning ID
     */
    var warningId: WaringIdEnum = WaringIdEnum.UNKNOWN,
)
```

Type: **Listen**

KeyFileUpload

- **Description**: File upload
- **Request parameters**: Data related to file upload
- **Response**: None
- **Type**: **Action**

FileUploadBean

```
data class FileUploadBean(
    var fileType: Int = 0, // File type. The value range varies by the two
devices involved.
    var fileName: String = "" // File name, typically with full path included.
The value varies by the two devices involved.
)
```

4. FlightMission Key

KeyEnter

Description: Entry into a waypoint mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyExit

Description: Exit from a waypoint mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyStart

Description: Start of a waypoint mission

Request parameters

```
data class MissionWaypointGUIDBean(
    /**
     * GUID data for mission waypoints
     */
    var guid: Int = 0
)
```

Response: Success or failure

Type: **Action**

KeyPause

Description: Pause of a waypoint mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyResume

Description: Resumption of a waypoint mission

Request parameters

```
data class MissionWaypointGUIDBean(
    /**
     * GUID data for mission waypoints
     */
    var guid: Int = 0
)
```

Response: Success or failure

Type: **Action**

KeyStop

Description: Stop of a waypoint mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyBreakRequest

Description: Query of waypoint mission interruption

Request parameters

```
data class MissionWaypointGUIDBean(
    /**
     * guid data for mission way points
     */
    var guid: Int = 0
)
```

Response:

```
data class MissionWaypointBreakRspBean(
    /**
     * Mission ID
     */
    var missionId: Int = 0,
    /**
     * Execution index
     */
    var executeIndex: Int = 0,
    /**
     * Mission type
     */
    var missionType: Int = 0,
    /**
     * Latitude (accuracy: 10e-7)
     */
    var lat: Double = 0.0,
    /**
     * Longitude (accuracy: 10e-7)
     */
    var lon: Double = 0.0,
    /**
     * Absolute altitude (accuracy: 10e-3)
     */
    var alt: Double = 0.0,
    /**
```

```
     * Query of history missions
     */
    var status: MissionQueryStatusEnum = MissionQueryStatusEnum.UNKNOWN,
    /**
     * Current mission status. For details, see MISSION_CURRENT_STATUS.
     */
    var executeStatus: MissionCurrentStateEnum = MissionCurrentStateEnum.UNKNOWN
)
```

Type: **Action**

KeyStatusReportNtfy

Description: Reporting of waypoint mission status

Request parameters: None

Response:

```
data class MissionWaypointStatusReportNtfyBean(

    /**
     * System timestamp, in ms
     */
    var timestamp: Long = 0L,
    /**
     * Mission ID
     */
    var missionId: Int = 0,
    /**
     * Waypoint No.
     */
    var wpSeq: Int = 0,
    /**
     * Remaining distance
     */
    var remainDistance: Int = 0,
    /**
     * For details, see MISSION_TYPE.
     */
    var missionType: MissionTypeEnum = MissionTypeEnum.UNKNOWN,
    /**
     * Number of photos taken
     */
    var photoNum: Int = 0,
    /**
     * Remaining time, in s
     */
    var remainTime: Int = 0,
    /**
     * Current mission status. For details, see MISSION_CURRENT_STATUS.
     */
    var status: MissionCurrentStateEnum = MissionCurrentStateEnum.UNKNOWN,
    /**
     * Mission GUID
     */
```

```
    var guid: Int = 0,
    /**
     * Action No.
     */
    var actionSeq: Int = 0,
    /**
     * Waypoint arrival. 0-arrived; 1-Not arrived
     */
    var arrived: Int = 0,
    /**
     * Mission speed: 10E-3 m/s
     */
    var speedSet: Int = 0,
    /**
     * Mission progress percentage
     */
    var percent: Int = 0

)
```

Type: **Listen**

KeyIPMEnter

Description: Entry into a POI mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyIPExit

Description: Exit from a POI mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyIPMStart

Description: Start of a POI mission

Request parameters

```
data class MissionInterestPointStartMsgBean(
    var latitude: Int = 0,      // Target latitude (accuracy: 10e-7) or NED x-coordinate
    var longitude: Int = 0,     // Target longitude (accuracy: 10e-7) or NED y-coordinate
    var altitude: Int = 0,      // Relative altitude of the target (mm) or NED z-coordinate
    var missionType: MissionInterestPointEnum = MissionInterestPointEnum.UNKNOWN,   // POI type. For details, see MISSION_INTEREST_POINT_TYPE.
    var speed: Int = 0,         // Flight speed (mm/s)
    var radius: Int = 0,         // Circling radius (mm)
    var reqId: Int = 0    //Request ID. You can use the POI ID as the request ID to ensure that the request ID is unique.
)
```

Response: Success or failure

Type: **Action**

KeyIPMStop

Description: Stop of a POI mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeyIPMStatusReport

Description: POI mission status

Request parameters: None

Response:

```
data class MissionInterestPointStatusReportNtfyBean(
    /**
     *  System timestamp, in ms
     */
    var timestamp: Long = 0,
    /**
     *  Mission ID
     */
    var missionId: Int = 0,
    /**
     *  Remaining distance
     */
    var remainDistance: Int = 0,
    /**
     *  For details, see MISSION_TYPE.
     */
```

```
    var missionType: MissionInterestPointEnum =
MissionInterestPointEnum.UNKNOWN,
    /**
     *  Progress percentage
     */
    var percent: Int = 0,
    /**
     *  Current mission status. For details, see MISSION_CURRENT_STATUS.
     */
    var status: MissionCurrentStateEnum = MissionCurrentStateEnum.UNKNOWN
)
```

Type: **Listen**

KeyIPMInfoReport

Description: Reporting of POI information

Request parameters: None

Response:

```
data class  MissionInterestPointInfoNtfyBean(
    var req_id: Int = 0, //Request ID. Same as the pinpoint request ID.

    var status: MissionResultEnum = MissionResultEnum.FAILED,//Pinpoint result.
0: failure, 1: success

    var latitude: Double = 0.0, //Target latitude (accuracy: 10e-7) or NED x-
coordinate

    var longitude: Double = 0.0, //Target longitude (accuracy: 10e-7) or NED y-
coordinate

    var altitude: Double  = 0.0,//Relative altitude of the target (mm) or NED z-
coordinate
)

enum class MissionResultEnum(val value: Int) {
        FAILED(0),            // Failure
        SUCCESE(1);           // Success
}
```

Type: **Listen**

KeyIPMCreatePoint

Description: POI pinpoint

Request parameters

```
data class MissionInterestPointCreatePointMsgBean(
    /**
     * Start point x-coordinate
```

```
     */
    var startX: Float = 0F,
    /**
     * Start point y-coordinate
     */
    var startY: Float = 0F,
    /**
     * Width
     */
    var width: Float = 0F,
    /**
     * Height
     */
    var height: Float = 0F
)
```

Response:

```
data class MissionInterestPointCreatePointRspBean(
    /**
     * Latitude (Accuracy: 10e-7)
     */
    var latitude: Double = .0,
    /**
     * Longitude (Accuracy: 10e-7)
     */
    var longitude: Double = .0,
    /**
     * Altitude (Accuracy: 10e-3)
     */
    var altitude: Float = .0F
)
```

Type: **Action**

KeyMissionOneClickStop

Description: Emergency stop

Request parameters: None

Response: None

Type: **Action**

KeySwarmEnter

Description: Entering a team mission

Request parameters: None

Response: None

Type: **Action**

KeySwarmExit

Description: Exiting a team mission

Request parameters: None

Response: None

Type: **Action**

KeySwarmStart

Description: Starting a team mission

Request parameters

```
data class MissionSwarmInfoBean(

    var missionId: Int = 0,

    var startTime: Int = 0
)
```

Response: Success or failure

Type: **Action**

KeySwarmPause

Description: Pausing a team mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeySwarmResume

Description: Resuming a team mission

Request parameters

```
data class MissionSwarmInfoBean(

    var missionId: Int = 0,

    var startTime: Int = 0
)
```

Response: Success or failure

Type: **Action**

KeySwarmStop

Description: Stopping a team mission

Request parameters: None

Response: Success or failure

Type: **Action**

KeySwarmDoAction

Description: Team mission execution

Request parameters: **Int**

Response: Success or failure

Type: **Action**

KeySwarmStatusNtfy

Description: Reporting of team mission status

Request parameters: None

Response:

```
data class MissionSwarmStatusNtfyBean(

    // Mission ID of the current mission
    var missionId: Int = 0,

    // Waypoint No. of the current mission
    var taskPointIndex: Int = 0,

    // Execution status. 0:Unknown, 1: Paused, 2: Executing, 3: Completed, 4:
Stopped
    var status: MissionCurrentStateEnum = MissionCurrentStateEnum.UNKNOWN,

    // Mission type. 0: Unknown, 1: Team building, 2: Team search
    var runTaskType: SwarmMissionTypeEnum = SwarmMissionTypeEnum.UNKNOWN,

    // Stage
    var taskStageIndex: Int = 4
)
```

Type: **Listen**

KeySwarmUploadNtfy

Description: Reporting of team mission upload status

Request parameters: None

Response:

```
data class MissionSwarmUploadFileNtfyBean(

    //ID of the received file
    var missionId: Int = 0,

    //Type of the received file
    var fileTyp : Int = 0,

    //Result: 0-unknown, 1-success, 2-failure
    var resultCode: Int = 0

)
```

Type: **Listen**

5. FlightControl Key

KeyCalibrateCompass

Description: Compass calibration

Request parameters: None

Response:

```
data class MissionSwarmUploadFileNtfyBean(

    //ID of the received file
    var missionId: Int = 0,

    //Type of the received file
    var fileTyp : Int = 0,

    //Result: 0-unknown, 1-success, 2-failure
    var resultCode: Int = 0

)
```

Type: **Listen**

KeyTakeOffAirCraft

Description: Takeoff

Request parameters: None

Response: Success or failure

Type: **Action**

KeySetLanding

Description: Auto landing on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Success or failure

Type: **Action**

KeyStartStopMotor

Description: Motor start/stop

Request parameters: Boolean. **false**: Stop, **true**: Start

Response: Success or failure

Type: **Action**

KeyStartStopAutoBack

Description: Auto return on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Success or failure

Type: **Action**

KeySetHomeLocation

Description: Home point settings

Request parameters: None

Response: Success or failure

Type: **Action**

KeyCustomHomeLocation

Description: Custom alternate point settings

Request parameters

```
data class HomeLocation(
    /**
     * latitude of home location
     */
```

```
    var latitude: Long = 0,
    /**
     * longitude of home location
     */
    var longitude: Long = 0,
    /**
     * height of home location
     */
    var height: Long = 0,
    /**
     * location enum type
     */
    var type: LocationTypeEnum=LocationTypeEnum.UNKNOWN)
```

Response: Success or failure

Type: **Action**

KeyCheckNFZUpload

Description: Whether to upload no-fly zone files

Request parameters

```
data class NoFlyQzoneBean(
    /**
     * Country code
     */
    var countryCode :String? = null,
    /**
     * No-fly zone file MD5
     */
    var fileMd5 : String? = null
)
```

Response: Boolean

Type: **Action**

KeyEnableNFZ

Description: No-fly zone mode on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Success or failure

Type: **Action**

KeySetAttiTakeOff

Description: Whether to allow takeoff in attitude mode

Request parameters: Boolean. **false**: No, **true**: Yes

Response: Success or failure

Type: **Action**

KeyCalibrateIMU

Description: IMU calibration

Request parameters: None

Response: Success or failure

Type: **Action**

KeyGetMissionGuid

Description: Mission GUID acquisition

Request parameters: None

Response: String

Type: **Action**

KeyCancelLowPowerBack

Description: Cancellation of return upon low power

Request parameters: None

Response: Success or failure

Type: **Action**

KeySetCompassTakeOff

Description: Whether to allow takeoff when the compass is abnormal

Request parameters: Boolean. **false**: No, **true**: Yes

Response: Success or failure

Type: **Action**

KeySetPortraitMode

Description: Fourth axis to portrait mode

Request parameters: Boolean. **false**: Off, **true**: On

Response: Success or failure

Type: **Action**

KeyGetCommonParams

Description: Flight control parameter list

Request parameters: None

Response:

```
data class DroneCommonParamSetBean(
    /**
     * RTK on/off
     */
    var supportRTK: Boolean = false,
    /**
     * Novice mode on/off
     */
    var beginMode: Boolean = false,
    /**
     * Aircraft mode
     */
    var gearLevel: GearLevelEnum = GearLevelEnum.UNKNOWN,
    /**
     * Lost action
     */
    var lostAction: DroneLostActionEnum = DroneLostActionEnum.UNKNOWN,
    /**
     * Maximum altitude
     */
    var maxHeight: Float = 0f,
    /**
     * Maximum distance
     */
    var maxRange: Float = 0f,
    /**
     * Return altitude
     */
    var backHeight: Float = 0f,
    /**
     * Low power warning threshold
     */
    var batLowWarningValue: Int = 0,
    /**
     * Critically low power warning threshold
     */
    var batSeriousWarningValue: Int = 0,
    /**
     * Whether to allow takeoff in attitude mode
     */
    var attiAllowFly: Boolean = false,
    /**
     * (Custom) Whether to allow takeoff when the compass is abnormal
     */
    var magErrorAllowFly: Boolean = false
)
```

Type: **Action**

KeyNestRetractPaddleControl

Description: Propeller controller

Request parameters: None

Response:

```
data class NestWaitTimeBean(
    /**
     *Wait time
     */
    var waitTime: Int = 0,
)
```

Type: **Action**

6. FlightProperty Key

KeyBeginMode

Description: Novice mode

Request parameters

```
enum class OperatorModeEnum(var value: Int = 0) {

    /**
     * Pro
     */
    PROFESSIONAL(0),

    /**
     * Novice
     */
    NEW_HAND(1),

    /**
     * Unknown
     */
    UNKNOWN(0XFF);

}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyMaxHeight

Description: Maximum flight altitude

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyMaxRadius

Description: Maximum flight radius

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyMaxHorizontalSpeed

Description: Maximum horizontal flight speed

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyMaxAscentSpeed

Description: Maximum ascent speed

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyDescentSpeed

Description: Maximum descent speed

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyMissionManagerBackHeight

Description: Return altitude

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBuzzingStatus

Description: Buzzing status (find aircraft)

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAttiMode

Description: ATTI mode on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyYawAngleSensitivity

Description: Yaw sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyPitchSensitivity

Description: Pitch sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRollSensitivity

Description: Roll sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyThrustSensitivity

Description: Throttle sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAttitudeSensitivity

Description: Attitude sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBrakeSensitivity

Description: Braking sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyYawTripSensitivity

Description: Yaw sensitivity

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBatteryLowWarning

Description: Low battery power warning

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBatSeriousLowWarning

Description: Critically low battery power warning

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyLowBatLowBack

Description: Low power return on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAircraftActivation

Description: Whether the aircraft is activated

Request parameters: Boolean. **false**: No, **true**: Yes

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRCLostAction

Description: Lost action

Request parameters

```
enum class DroneLostActionEnum(val value: Int) {

    /**
     * Unknown
     */
    UNKNOWN(0),

    /**
     * Return
     */
    BACK(1),

    /**
     * Hover
     */
    HOVER(2),

    /**
     * Land
```

```
     */
    LANDING(3);

}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyGearLever

Description: Aircraft mode

Request parameters

```
enum class  GearLevelEnum (var value: Int){

    /**
     * Unknown
     */
    UNKNOWN(0),

    /**
     * Smooth
     */
    SMOOTH(1),

    /**
     * Standard
     */
    NORMAL(2),

    /**
     * Ludicrous
     */
    SPORT(3);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyCoordinatedTurn

Description: Coordinated turn

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyLocationStatus

Description: Fusion positioning on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyFcsApasModeEn

Description: Obstacle avoidance on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeySilentModeStatus

Description: Silent mode

Request parameters: Boolean. **false**: Off, **true**: On

Type: **Get**, **Set**

KeyFCSEnSuperCap

Description: Supercapacity on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyFcsEnGpsMode

Description: GPS flying on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyFcsSwitchGpsMode

Description: GPS mode

Request parameters

```
enum class  DroneGpsEnum (var value: Int){
```

```
    //Supported GPS modes:
    //BDS
    //GPS
    //GLONASS
    //GALILEO
    /**
     * Unknown
     */
    UNKNOWN(0),
    BDS(1),
    GPS(2),
    BDS_GPS(3),
    BDS_GLO(5),
    GPS_GAL(10),
    BDS_GPS_GLO(7),
    BDS_GPS_GAL(11),
    GLO_GAL(14),
    BDS_GPS_GAL_GLO(15);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

7. Gimbal Key

KeyHeatBeat

Description: Reporting of gimbal information

Request parameters: None

Response:

```
data class DroneGimbalStateBean(
    /**
     *  Whether the gimbal attitude has been calibrated
     */
    var isAttitudeValid: Boolean = false,
    /**
     *  Gimbal
     */
    var workMode: GimbalWorkModeEnum = GimbalWorkModeEnum.UNKNOWN,
    /**
     *  Whether the gimbal is being calibrated
     */
    var isCalibrating: Boolean = false,
    /**
     *  Motor status
     */
    var motorStatus: GimbalMotorStatus = GimbalMotorStatus.NONE,
    /**
     *  Whether the gimbal roll is at maximum angle
     */
    var isRollReachMax: Boolean = false,
```

```
    /**
     *  Whether the gimbal is overheated
     */
    var isOverheat: Boolean = false,
    /**
     *  Whether the gimbal needs to caliabrate IMU
     */
    var isIMUNeedCalibrate: Boolean = false,
    /**
     *  4-axis rotation angle
     */
    var rotateAngle: Float = 0f,
    /**
     *  4-axis rotation angle speed
     */
    var rotateAngleRange: Float = 0f,
    /**
     *  Whether the gimbal is faulty
     */
    var isHardwareFail: Boolean = false,
    /**
     *  Whether the gimbal is inactivated for not receiving attitude data
     */
    var isSleepForNoAttitude: Boolean = false,
    /**
     *  Whether the gimbal is inactivated for reaching maximum roll angle.
     */
    var isSleepForRollReachMax: Boolean = false,
    /**
     *  Gimbal calibration status
     */
    var gimbalCalibrateStatus: Int = 0,
    /**
     *  Gimbal calibration percentage (0-100)
     */
    var gimbalCalibratePercent: Int = 0
)
```

Type: **Listen**

KeyRollAdjustAngle

Description: Gimbal roll adjustment

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyPitchAngleRange

Description: Gimbal pitch adjustment

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyYawAdjustAngle

Description: Gimbal yaw adjustment

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyStartIMUCalibration

Description: Gimbal IMU calibration

Request parameters: None

Response: Success or failure

Type: **Action**

KeyStartCalibration

Description: Gimbal calibration

Request parameters: None

Response: Success or failure

Type: **Action**

KeyRotateFouraxisAngle

Description: Rotating 4-axis gimbal

Request parameters

```
data class RotateFourAxisParamsBean(
    /**
     * Rotating speed
     */
    var rotateSpeed: Int = 0,
    /**
     * Rotation. +: clockwise, -:anti-clockwise
     */
    var rotateDirection: Int = 0,
    /**
     * Rotation. If set to 1, it will change to 0 after 0.05s.
     */
    var rotate: Int = 0
)
```

Response: Success or failure

Type: **Action**

KeyWordMode

Description: Gimbal work mode

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyPitchAngelRange

Description: Pitch range on/off

Request parameters: Boolean, **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyPitchSpeed

Description: Pitch speed

Request parameters: **Int**

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAngleControl

Description: Gimbal angle control

Request parameters: Float

Response: Success or failure

Type: **Action**

KeyOrientationControl

Description: Gimbal orientation control

Request parameters

```
enum class GimbalOrientationEnum(var value: Float) {
    /**
```

```
         * Down
         */
        DOWN(-(Math.PI / 2).toFloat()),

        /**
         * 45 degrees down
         */
        DOWN_45(-(Math.PI / 4).toFloat()),

        /**
         * Centering
         */
        FORWARD(0f);
    }
```

Response: Success or failure

Type: **Action**

KeyLaserRangingSwitch

Description: Laser rangefinder on/off

Request parameters: Boolean. **true**: On, **false**: Off

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

8. MissionManager Key

KeyRadarFusion

Description: Radar fusion on/off

Request parameters: Boolean. **true**: On, **false**: Off

Response: **Get**, **Set**

Type: Boolean

KeyPreciseLand

Description: Precision landing on/off

Request parameters: Boolean. **true**: On, **false**: Off

Response: **Get**, **Set**

Type: Boolean

KeyReturnObsEnable

Description: Obstacle avoidance on/off during return

Request parameters: Boolean. **true**: On, **false**: Off

Response: **Get**, **Set**

Type: Boolean

KeyDroneRunningMode

Description: Aircraft work mode

Request parameters: Integer. (1-Nest, 2-RC Manual)

Response: **Get**, **Set**

Type: Integer

KeyDoneTakeOffHeight

Description: Aircraft takeoff altitude (m)

Request parameters: Float

Response: **Get**, **Set**

Type: Float

KeyExitPrecisionLand

Description: Exit from precision landing

Request parameters: None

Response: None

Type: Action

KeyEnableLandingProtection

Description: Landing protection on/off

Request parameters: Boolean. **true**: On, **false**: Off

Response: **Get**, **Set**

Type: Boolean

KeyIgnoreLandRisk

Description: Landing cancellation

Request parameters: Integer. 0: Cancel, 2: Continue landing

Response: **Set**

Type: Integer

9. RemoteController Key

KeyRcDeviceInfo

Description: Remote controller information

Request parameters: None

Response: List `<DroneVersionItemBean>`

```
data class DroneVersionItemBean(
    /**
     * After the capability set is built, this field is set by device
registration order. The IDs of the EVO Max Series are fixed, for example,
front/rear-view radar IDs.
     */
    var componentID: DroneComponentIdEnum = DroneComponentIdEnum.NONE,
    /**
     * Component SN
     */
    var componentSN: String? = null,
    /**
     * Hardware version
     */
    var hardwareVersion: String? = null,
    /**
     * Software version
     */
    var softwareVersion: String? = null,
    /**
     * Component model (integer)
     */
    var componentModel: Int = 0,  //
    /**
     * Component model (string, for example, XM802)
     */
    var strComponentModel: String? = null,
    /**
     * Reserved field, used to describe the current component. Currently
meaningless for the EVO Max Series.
     */
    var label: String? = null,
    /**
     * Reserved field, used to describe the current component type. Currently
meaningless for the EVO Max Series.
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,
    /**
     * MCU bootloader version
     */
    var bootloaderVersion: String? = null,
)
```

Type: **Action**

KeyRcSystemTime

Description: Remote controller system time

Request parameters

```
data class SystemTimeInfoBean(
    /**
     * UTC timestamp
     */
    var utcTimeInterval: Long = 0,
    /**
     * Time zone
     */
    var timeZone: Float = 0f,
)
```

Response: Success or failure

Type: **Action**

KeyRCRockerControlMode

Description: Joystick mode

Request parameters

```
enum class RcOperateModeEnum(val value: Int) {
    /**
     * Mode 2
     */
    AMERICA_HAND(0),

    /**
     * Mode 3
     */
    CHINESE_HAND(1),

    /**
     * Mode 1
     */
    JAPANESE_HAND(2);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRCEnterCalibration

Description: Entry into remote controller calibration

Request parameters: None

Response: Success or failure

Type: **Action**

KeyRCExitCalibration

Description: Exit from remote controller calibration

Request parameters: None

Response: Success or failure

Type: **Action**

KeyEnableRemoteUpload

Description: Fixed-frequency reporting on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Success or failure

Type: **Action**

KeyRcQuickMatchChooseConnect

Description: Quick pairing

Request parameters: **Int**

Response: Success or failure

Type: **Action**

KeyRcQuickMatchClearBlacklist

Description: Quick blacklist pairing and clearing

Request parameters: None

Response: Success or failure

Type: **Action**

10. RtkProperty Key

KeyGetRTKAuthInfo

Description: Acquisition of RTK authorization information

Request parameters: None

Response:

```
data class RTKAuthInfo(
    /**
     * Account
     */
    var account : String? = null,
    /**
```

```
     * Password
     */
    var password : String? = null,
    /**
     * Device ID
     */
    var deviceId : String? = null,
    /**
     * Device type
     */
    var deviceType : String? = null
)
```

Type: **Action**

KeySetRTKAuthInfo

Description: RTK authorization configuration

Request parameters

```
data class RTKAuthInfo(
    /**
     * Account
     */
    var account : String? = null,
    /**
     * Password
     */
    var password : String? = null,
    /**
     * Device ID
     */
    var deviceId : String? = null,
    /**
     * Device type
     */
    var deviceType : String? = null
)
```

Response: Success or failure

Type: **Action**

KeyRtkReportInfo

Description: Reporting of RTK information

Request parameters: None

Response:

```
data class RtkReportBean(

```

```
    var solState:Int = 0,

    /**< Solution status. For details, see FlightControlDefine >
RTK_SOLUTION_TYPE.  */
    var posType: Int = 0,

    /**< Position type. For details, see FlightControlDefine >
RTK_POSITION_TYPE.  */
    var lat : Double = 0.0, //Latitude, degree/

    var lon : Double = 0.0,//Longitude, degree/

    var hgt: Double = 0.0, //Altitude, m/

    var latSigma: Float = 0.0f,//<Latitude standard deviation

    var lonSigma: Float = 0.0f,//Longitude standard deviation

    var hgtSigma: Float = 0.0f,//Altitude standard deviation

    var svCnt : Int = 0, //Number of satellites tracked/

    var solnSVs : Int = 0,//Number of satellites used in the solution/

    var gpsCnt : Int = 0,//Number of GPS satellites/

    var beidouCnt : Int = 0,//Number of Beidou satellites/

    var glonassCnt : Int = 0,//Number of Glonass satellites/

    var galileoCnt : Int = 0, //Number of Galileo satellites/

    var inPos : Int = 0,//RTK in position / 0: No, 1: Yes

    var rtkUsed : Int = 0, //RTK positioning on/off / 0: Off, 1: On

    var fixSta : Int = 0, // Fix status/ 0: Not fixed, 1: Fixed

    var locationType : Int = 0,//Positioning mode/ 0: GNSS, 1: RTK

    var signalType : RTKSignalEnum = RTKSignalEnum.SELF_NETWORK, //RTK signal
type. For details, see FlightControlDefine > RTK_SIGNAL_TYPE.

    var coordinateSys : RTKCoordinateEnum = RTKCoordinateEnum.MOBILE_STATION,
//RTK coordinate system. For details, see FlightControlDefine > RTK_COORDINATE.

)
```

Type: **Listen**

KeyRTKLocationEnable

Description: RTK positioning on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRTKSignalType

Description: RTK signal type

Request parameters

```
enum class RTKSignalEnum(val value: Int) {

    /**
     * Unknown
     */
    UNKNOWN(0),

    /**
     * RTK mobile station
     */
    MOBILE_STATION(1),

    /**
     * Network RTK
     */
    NETWORK(2),

    /**
     * Custom network RTK
     */
    SELF_NETWORK(3);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRTKCoordinateSystem

Description: RTK coordinate system

Request parameters

```
enum class RTKCoordinateEnum(val value: Int) {

    /**
     * WGS84
     */
    WGS84(0),

    /**
     * RTK mobile station
     */
    MOBILE_STATION(1),
```

```
    /**
     * Unknown
     */
    UNKNOWN(0XFF);
}
```

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

11. SystemManager Key

KeySystemManagerDebug

Description: System service debug command

Request parameters

```
data class SystemDebugBean(
    /**
     *Command. For details, see SYSTEM_DEBUG_CMD.
     */
    var command: SystemDebugCmdEnum = SystemDebugCmdEnum.UNKNOWN,
    /**
     *Custom function description request parameter
     */
    var parameter: String? = null,
    /**
     *Debug platform. For details, see SYSTEM_DEBUG_PLATFORM.
     */
    var platform: SystemDebugPlatformEnum = SystemDebugPlatformEnum.UNKNOWN,
    /**
     *Debug result storage path
     */
    var fullPath: String? = null,
    /**
     *Whether to enable response. 0: No, 1: Yes
     */
    var response: Int = 0,
)
```

Response: Success or failure

Type: **Action**

KeySystemStatusData

Description: System status

Request parameters: None

Response:

```
data class SystemStatusDataAtomBean(
    /**
     *Platform type. For details, see SYSTEM_DEBUG_PLATFORM.
     */
    var platform: SystemDebugPlatformEnum = SystemDebugPlatformEnum.UNKNOWN,
    /**
     *Chip type. For details, see SYSTEM_CORE_TYPE.
     */
    var coreType: SystemCoreTypeEnum = SystemCoreTypeEnum.UNKNOWN,
    /**
     *Chip index ID. For details, see SYSTEM_CORE_INDEX.
     */
    var coreIndex: SystemCoreIndexEnum = SystemCoreIndexEnum.CORE_0,
    /**
     *Data: load %, temperature ℃, frequency Hz, frequency limit Hz. If the
data does not exist or there is an exception, the data value is 0.
     */
    var dataList: List<Float> = mutableListOf(),
)
```

Type: **Listen**

KeySystemWorkStatusData

Description: Reporting of system operating status

Request parameters: None

Response:

```
data class SystemWorkStatusBean(
    /**
     *Debugging platform. For details, see SYSTEM_DEBUG_PLATFORM.
     */
    var platform: SystemDebugPlatformEnum = SystemDebugPlatformEnum.UNKNOWN,
    /**
     *System operating mode
     */
    var systemWorkMode: SystemWorkStatusEnum = SystemWorkStatusEnum.UNKNOWN,
)
```

Type: **Listen**

KeySystemManagerDroneCustomName

Description: Custom aircraft name

Request parameters: String

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeySystemManagerStatusReportControl

Description: System debugging information reporting on/off

Request parameters: Boolean

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

12. UpgradeService Key

KeyUpgradeQuery

Description: Update query

Request parameters

```
data class UpgradeQueryBean(
    /**
     *Client type. For details, see CLIENT_TYPE_E.
     */
    var clientType: UpgradeClientTypeEnum =
UpgradeClientTypeEnum.CLIENT_TYPE_GND,
)
```

Response: Success or failure

Type: **Action**

KeyUpgradeEnter

Description: Entering update state

Request parameters

```
data class UpgradeQueryBean(
    /**
     *Client type. For details, see CLIENT_TYPE_E.
     */
    var clientType: UpgradeClientTypeEnum =
UpgradeClientTypeEnum.CLIENT_TYPE_GND,
)
```

Response: Success or failure

Type: **Action**

KeyUpgradeTransfer

Description: Update package transmission progress

Request parameters: **Int**

Response: Success or failure

Type: **Action**

KeyUpgradeCheck

Description: Update package verification

Request parameters

```
data class UpgradeCheckBean(
    /**
     * MD5 verification
     */
    var MD5: String = "",
    /**
     * Package name, full path included
     */
    var filename: String = "",
)
```

Response: Success or failure

Type: **Action**

KeyUpgradeExecute

Description: Start of update

Request parameters

```
data class UpgradeExecuteReqBean(
    /**
     * Force update
     */
    var force: Boolean = false,
    /**
     * Package name, full path included
     */
    var filename: String = "",
)
```

Response: Success or failure

Type: **Action**

KeyUpgradeProgress

Description: Reporting of update progress

Request parameters: None

Response: **Int**

Type: **Listen**

KeyUpgradeResult

Description: Update result

Request parameters: None

Response:

```
data class UpgradeResultBean(
    /**
     *Error. For details, see ERROR_TYPE_E.
     */
    var errCode: UpgradeErrorTypeEnum = UpgradeErrorTypeEnum.UPGRADE_ERR_NONE,
    /**
     *Error descrption
     */
    var errDesc: String = "",
    /**
     *Resule: 1-success, 2-error
     */
    var result: ResponseResultEnum = ResponseResultEnum.UNKNOWN,
)
```

Type: **Listen**

KeyUpgradeComponentCheck

Description: Component consistency check

Request parameters: None

Response: List`<DroneVersionItemBean>`

```
data class DroneVersionItemBean(
    /**
     * After the capability set is built, this field is set by device
registration order. The IDs of the EVO Max Series are fixed, for example,
front/rear-view radar IDs.
     */
    var componentID: DroneComponentIdEnum = DroneComponentIdEnum.NONE,
    /**
     * Component SN
     */
    var componentSN: String? = null,
    /**
     * Hardware version
     */
    var hardwareVersion: String? = null,
    /**
     * Software version
     */
    var softwareVersion: String? = null,
    /**
     * Component model (integer)
     */
    var componentModel: Int = 0,  //
```

```
    /**
     * Component model (string, for example, XM802)
     */
    var strComponentModel: String? = null,
    /**
     * Reserved field, used to describe the current component. Currently
meaningless for the EVO Max Series.
     */
    var label: String? = null,
    /**
     * Reserved field, used to describe the current component. Currently
meaningless for the EVO Max Series.
     */
    var componentType: DroneComponentEnum = DroneComponentEnum.UNKNOWN,
    /**
     * MCU bootloader version
     */
    var bootloaderVersion: String? = null,
)
```

Type: **Listen**

KeyUpgradeComponentRecover

Description: Recovery of components

Request parameters: None

Response: Success or failure

Type: **Action**

KeyUpgradeStateQuery

Description: Update status query

Request parameters

```
enum class UpgradeStateEnum(var value: Int) {
    /**
     * Idle
     */
    IDLE(0),

    /**
     * Transmitting update package
     */
    TRANSFER(1),

    /**
     * Updating
     */
    EXECUTING(2),

    /**
```

```
     * Update completed
     */
    FINISH(3);

}
```

Response: Success or failure

Type: **Action**

13. Vision Key

KeyReportEmergency

Description: Radar chart warning

Request parameters: None

Response: List `<VisionRadarInfoBean>`

```
data class VisionRadarInfoBean(
    /**
     *Timestamp
     */
    var timeStamp: Long = 0,
    /**
     *Sensor position
     */
    var position: VisionSensorPositionEnum = VisionSensorPositionEnum.FRONT,
    /**
     *Distances between sensor and obstacles
     */
    var distances: List<Float>? = null

)
```

Type: **Listen**

KeyHorizontalObstacleAvoidance

Description: Horizontal obstacle avoidance on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyHorizontalBrakeDistance

Description: Horizontal braking distance

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyHorizontalWarningDistance

Description: Horizontal warning distance

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTopObstacleAvoidance

Description: Upward obstacle avoidance on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTopBrakeDistance

Description: Upward braking distance

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyTopWarningDistance

Description: Upward warning distance

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBottomObstacleAvoidance

Description: Downward obstacle avoidance on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBottomBrakeDistance

Description: Downward braking distance

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyBottomWarningDistance

Description: Downward warning distance

Request parameters: Float

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyRadarDetection

Description: Radar detection on/off

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

KeyAutonomyMifWorkStatus

Description: MIF visual positioning status

Request parameters: Boolean. **false**: Off, **true**: On

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

14. LteModuleKey

LTE_MODULE_ENABLE

Description: LTE module on

Request parameters: Integer

Response: None

Type: **Get**, **Set**

LTE_APN_DOMAIN_NAME

Description: LTE APN domain name

Request parameters: String

Response: None

Type: **Set**, **Get**

LTE_APN_USERNAME

Description: LTE APN username

Request parameters: String

Response: None

Type: **Set**, **Get**

LTE_APN_PASSWORD

Description: LTE APN password

Request parameters: String

Response: None

Type: **Set**, **Get**

setAPNAccount

Description: APN account

Request parameters

- **domainName**: APN domain name
- **userName**: APN username
- **password**: APN password
- **callback**: Completion callback

Response: None

Type: None

15. RtmpKey

KeyRtmpModuleEnable

Description: RTMP module on/off

Request parameters: Integer. **0**: Off, **1**: On

Response: **Get**, **Set**

Type: Integer

KeyRtmpServerAddress

Description: RTMP server address

Request parameters: String

Response: **Get**, **Set**

Type: String

16. NtripAccountKey

NTRIP_MODULE_ENABLE

Description: NTRIP module on/off

Request parameters: Integer. **0**: Off, **1**: On

Response: **Get**, **Set**

Type: Integer

NTRIP_DOMAIN_NAME

Description: NTRIP server domain name

Request parameters: String

Response: **Get**, **Set**

Type: String

NTRIP_USERNAME

Description: NTRIP server username

Request parameters: String

Response: **Set**

Type: String

NTRIP_PASSWORD

Description: NTRIP server password

Request parameters: String

Response: **Set**

Type: String

NTRIP_MOUNT_POINT

Description: NTRIP server mount point

Request parameters: String

Response: **Set**

Type: String

NTRIP_PORT

Description: NTRIP server port

Request parameters: Integer

Response: **Set**

Type: Integer

autherMobileServiceRtk

Description: 4G/5G mobile network authorization login for RTK services.

Request parameters

- **mHost**: String. RTK server host address
- **mPort**: Integer. RTK server port number
- **userName**: String. RTK server username
- **password**: String. RTK server password
- **mountPoint**: String. RTK server mount point
- **callback**: **RTKAuthoCallback** object. Completion callback

Response: None

Type: None

17. MqttPropertyKey

MQTT_DOMAIN_NAME

Description: MQTT domain name

Request parameters: String

Response: Request parameters will be returned for **Get** requests.

Type: **Get**, **Set**

MQTT_USERNAME

Description: MQTT username

Request parameters: String

Response: None

Type: **Set**

MQTT_PASSWORD

Description: MQTT password

Request parameters: String

Response: None

Type: **Set**

loginMqtt

Description: MQTT login

Request parameters

- **domainName**: String. MQTT server domain name
- **userName**: String. MQTT server username
- **password**: String. MQTT server password
- **callback**: **CommonCallbacks.CompletionCallback** object. Completion callback

Response: None

Type: None

```
/**
 * 4G/5G mobile network authorization login for RTK services.
 *
 * @param mHost RTK server host address
 * @param mPort RTK server port number
 * @param userName RTK server username
 * @param password RTK server password
 * @param mountPoint RTK server mount point
 * @param callback RTKAuthoCallback object. Completion callback
 *
 * @return None
 */
abstract fun autherMobileServiceRtk(
    mHost: String,
    mPort: Int,
    userName: String,
    password: String,
    mountPoint: String,
    callback: RTKAuthoCallback
)
```