Table of Contents

References

DeviceListener

DeviceListener is a callback listener that receives callback data from the device.

```
public class DeviceListener implements IDeviceListener
{
    /**
     * Called when a {@link Punch} detected.
     * Punch data types are card, magnetic stripe, barcode and integrated
circuit.
     * @param data {@link Punch}
     */
    @Override
    public void onPunchDetected(Punch data) {
    }

    /**
     * Called when a {@link Input} detected.
     * @param data {@link Input}
     */
    @Override
    public void onInputDetected(Input data) {
    }

    /**
     * Called when a {@link Event} detected.
     * @param data {@link Event}
     */
    @Override
    public void onEventDetected(Event data) {
    }

    /**
     * Called when a {@link Fingerprint} detected.
     * @param data {@link Fingerprint}
     */
    @Override
    public void onFingerprintDetected(Fingerprint data) {
    }

    /**
     * Called when a {@link Fingerprint} scan completed.
     * @param data {@link Fingerprint}
     */
    @Override
    public void onFingerprintScanCompleted(Fingerprint data) {
```

```
    }

    /**
     * Called when a {@link Fingerprint} identified.
     * @param data {@link Fingerprint}
     */
    @Override
    public void onFingerprintIdentified(Fingerprint data) {
    }

    /**
     * Called when a {@link Card} scan completed.
     * @param data {@link Card}
     */
    @Override
    public void onCardScanCompleted(Punch data) {
    }

    /**
     * Called when a {@link Fingerprint} scan progressed.
     * @param scanTimeout finger scan timeout.
     */
    @Override
    public void onFingerprintScanProgress(int scanTimeout) {
    }

    /**
     * Called when a {@link Card} scan progressed.
     * @param scanTimeout card scan timeout.
     */
    @Override
    public void onCardScanProgress(int scanTimeout) {
    }
}
```

Versions

```
public class Version {
    /**
     * SDK version.
     */
    public String sdkVersion;
    /**
     * Firmware version.
     */
    public String firmwareVersion;
```

```
}
```

Device

```
public class Device {
    /**
     * Device ID.
     */
    public int deviceId;
    /**
     * Device model name.
     */
    public String modelName;
}
```

FirmwareOption

```
public class FirmwareOption extends SvpPayload {
    /**
     * Firmware file name.
     */
    public String fileName;
    /**
     * FTP server host name(IP).
     */
    public String host;
    /**
     * FTP user name.
     */
    public String username;
    /**
     * FTP user password.
     */
    public String password;
    /**
     * FTP server port
     */
    public int port;
}
```

ApplicationOption

```
public class ApplicationOption extends SvpPayload
{
    /**
     * Application auto start option.
     */
    public boolean useAutoStart;
    /**
     * Your application class name.
     */
    public String packageName;
    /**
     * Your application activity name.
     */
    public String className;
}
```

CardOption

```
public class CardOption extends SvpPayload
{
    /**
     * Card scanning timeout in seconds. The default is 10 seconds.
     */
    public int scanTimeout;
    /**
     * Order of how the byte of the card is stored.
     * When it is set as 0, will function as MSB. When it is set as 1, will
function as LSB.
     */
    public int byteOrder;
}
```

FingerprintOption

```
public class FingerprintOption extends SvpPayload
{
    /**
     * Fingerprint authentication security level. This is used across the
system.
     */
```

```
    public int securityLevel;
    /**
     * Fingerprint matching speed.
     */
    public int fastMode;
    /**
     * Sensitivity of the fingerprint sensor.
     */
    public int sensitivity;
    /**
     * Decides the sensor mode. 0 means the sensor is always on.
     * 1 means the sensor is activated when the finger is near the sensor.
     */
    public int sensorMode;
    /**
     * Fingerprint template type.
     */
    public int templateFormat;
    /**
     * Fingerprint scanning timeout in seconds. The default is 10 seconds.
     */
    public int scanTimeout;
    /**
     * Configuration for the LFD sensitivity.
     */
    public int lfdLevel;
    /**
     * Decides whether to utilize fingerprint quality information. If the
option is disabled,
     */
    public boolean useAdvancedEnrollment;
    /**
     * Decides whether to use bitmap image.
     */
    public boolean useBitmapImage;
}
```

WiegandOption

```
public class WiegandOption extends SvpPayload
{
    /**
     * Output pulse width having a range of 20 ~ 100 us.
     */
    public int outPulseWidth;
    /**
     * Output pulse frequency having a range of 200 ~ 20000 us.
```

```
     */
    public int outPulseInterval;
}
```

Channel

```
public enum Channel
{
    /**
     * Channel value of relay port 0
     */
    RELAY_PORT_0,
    /**
     * Channel value of relay port 1
     */
    RELAY_PORT_1
}
```

Relay

```
public enum Relay {
    /**
     * Off
     */
    OFF,
    /**
     * On
     */
    ON
}
```

LedColor

```
public enum LedColor
{
    /**
     * LED Off
     */
    LED_COLOR_OFF,
    /**
```

```
     * Red LED
     */
    LED_COLOR_RED,
    /**
     * Yellow LED
     */
    LED_COLOR_YELLOW,
    /**
     *  Green LED
     */
    LED_COLOR_GREEN,
    /**
     * Blue-Green LED
     */
    LED_COLOR_CYAN,
    /**
     * Blue LED
     */
    LED_COLOR_BLUE,
    /**
     * Magenta LED
     */
    LED_COLOR_MAGENTA,
    /**
     * White LED
     */
    LED_COLOR_WHITE
}
```

Finger

```
public class Finger extends SvpPayload implements Cloneable {
    /**
     * Finger ID.
     */
    public int      id;
    /**
     * The index of finger.
     */
    public int      index;
    /**
     * Used only in {@link com.supremainc.sdk.SvpManager#verifyFingerprint}.
     * Flag for update decision.
     * false : don't need to update template in DB.
     * true : need to update even template of the user of matched template.
     */
    public boolean  isUpdated;
```

```
    /**
     * The data of fingerprint template.
     */
    public FingerprintTemplate[] templates;
}
```

FingerList

```
public class FingerList extends SvpPayload implements Cloneable {
    /**
     * Finger array.
     */
    public ArrayList<Finger> fingers;
}
```

FingerprintTemplate

```
public class FingerprintTemplate extends SvpPayload implements Cloneable {
    /**
     * The data of fingerprint template.
     */
    public byte[] template;
}
```

RS232Data

```
public class RS232Data extends SvpPayload implements Cloneable {
    /**
     * The data of byte array.
     */
    private byte[] rs232Data;
}
```