4.12 Recording

In recording mode, motion data is stored in the sensor internal storage and can be exported for post processing. Bluetooth connection is not required once the recording is started. With the SDK, you can start/stop recording, set timed recording and export recording data.

Figure 6 shows the recommended workflow to start and stop recording with iOS SDK.

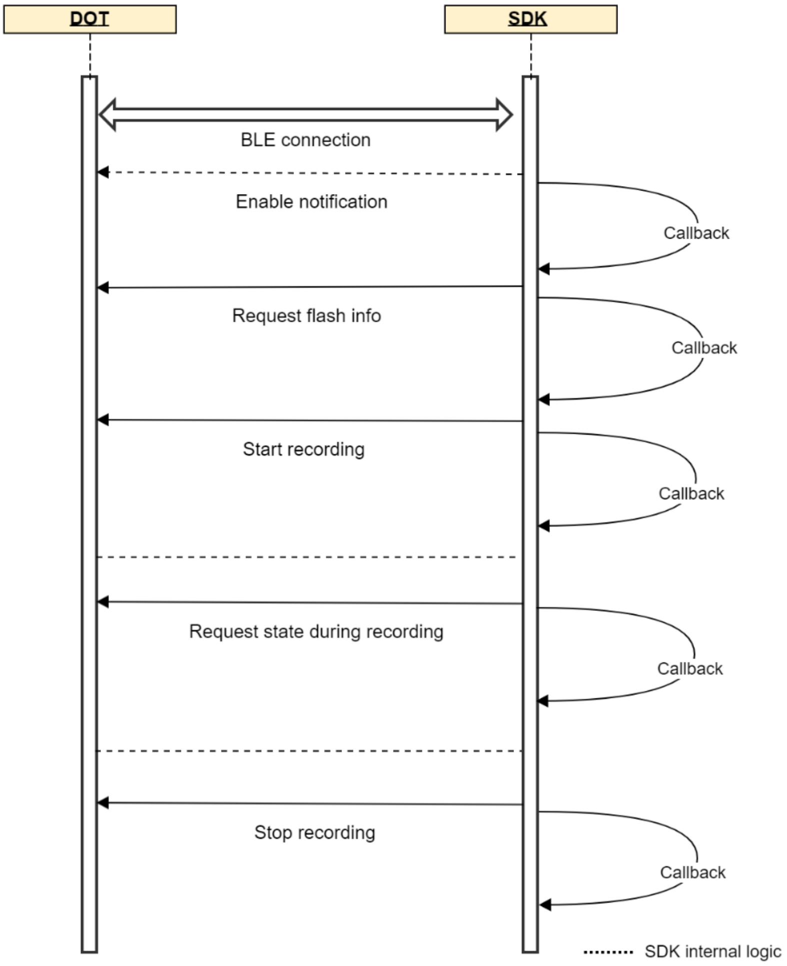

Figure 6: Workflow to start and stop recording

4.12.1 Get flash information

Flash information refers to recording flash size and its usage. The flash size that can be used for recording accounts for about 90% of the total size (16 MB for v1 sensor, 64MB for v2 sensor). So firstly, we need to get the available flash space and the remaining recording time before start recording.

DotDevice.totalSpace and *DotDevice.usedSpce* can only be initialized after calling *getFlashInfo*.

```
[DotDevice getFlashInfo];
```

This can be used to update the flash status:

```
[DotDevice setFlashInfoDoneBlock:^(XSFlashInfoStatus status) { }];
```

Table 16: Flash status

Flash status	Description
XSFlashInfoIsReady	Recording flash space is ready for recording.
XSFlashInfoIsFull	Recording flash space is full (⩾ 90% the whole size).
XSFlashInfoIsUninitialized	Recording flash space is uninitialized or invalid.

Note that uninitialized or invalid flash means that the current recording flash space is not compatible with the flash structure supported in this firmware. Erase the flash space to reset the structure.

4.12.2 Start recording

You can start recording or set a timed recording with the parameter of *startRecording*.

Set the param to 0xFFFF to start recording; the recording will continue unless a stop recording command is received or sensor stops automatically. Set the param to other values to start a timed recording. The unit is second. Note that the maximum recording time of the sensor is 88 minutes.

```
[DotDevice startRecording:0xFFFF];
```

4.12.3 Get recording status

Use *getRecordingStatus* to check the status of recording.

```
[DotDevice getRecordingStatus];
```

After call this method, the *DotDevice.recording.recordingStatus* will be updated with the recording status.

Table 17: Recording status

Recording status	Description
XSRecordingIsIdle	Idle status
XSRecordingIsRecording	Sensor is recording
XSRecordingIsRecordingStopped	Recording is stopped
XSRecordingIsErasing	Erasing recording data
XSRecordingIsFlashInfo	Sensor is getting flash information

In the meantime, *updateRecordingStatus* block will be updated automatically.

```
[DotDevice.recording setUpdateRecordingStatus:^(XSRecordingStatus
status) {

}];
```

This block will also be updated after calling *startRecording* and *stopRecording*.

4.12.4 Stop recording

Use *stopRecording* to stop a normal recording or a timed recording.

```
[DotDevice stopRecording];
```

Recording will also stop automatically in the following situations:

- power button is pressed over 1 second.
- time is up for timed recording.
- flash memory is over 90%.

4.12.5 Get recording time

If the *recordingStatus* is *XSRecordingIsRecording*, you can call *getRecordingTime* method to get *recordingDate*, *recordingTime* and *remainingTime*.

```
[DotDevice getRecordingTime];
```

```
DotDevice.recording.recordingDate
// start recording time 4 bytes unit is second.
DotDevice.recording.recordingTime
// recording Time that you startRecording set 2 bytes unit is second.
DotDevice.recording.remainingTime
// the remaining time after you start recording if you set
recordingTime is not 0XFFFF;
```

4.12.6 Erase flash

Erase all the recording data space, other flash space will not be affected.

```
[DotDevice eraseData];
```

setEraseDataDoneBlock will be updated if the erase process is done.

```
[DotDevice setEraseDataDoneBlock:^(int success) {}];
```

4.13 Recording data export

A stand-alone application – Movella DOT Data Exporter is provided to export the recording data to PC via USB cable. You can download Windows or MacOS version in developers page.

Figure 7 shows the recommended workflow to start and stop recording with SDK.

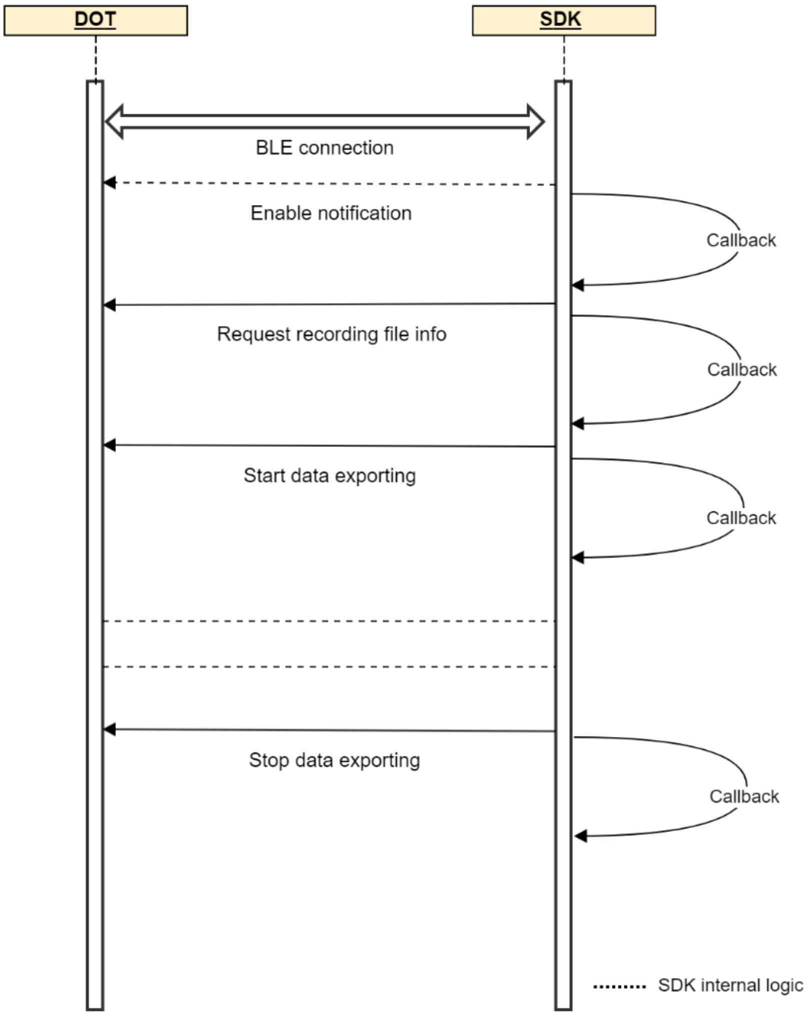

Figure 7: Workflow to export recording data

Follow the steps below to implement data export function in your application.

4.13.1 Set export data format

Refer to Table 18 for available data quantities when configuring the export data format. For physical meanings and other information of the data, please refer to chapter 4 in *Movella DOT User Manual*.

Note that free acceleration is not provided in this firmware. Refer to this BASE article to calculate free acceleration from orientation (quaternion) and acceleration.

Table 18: Export data quantities

Data	Description
XSRecordingDataTimestamp	TimeStamp
XSRecordingDataQuaternion	Quaternion
XSRecordingDataEulerAngles	Euler angles
XSRecordingDataDq	dq
XSRecordingDataDv	dv
XSRecordingDataAcceleration	Calibrated acceleration
XSRecordingDataAngularVelocity	Calibrated angular velocity
XSRecordingDataMagneticField	Calibrated magnetic field
XSRecordingDataStatus	Status
XSRecordingDataClipCountAcc	clipCountAcc
XSRecordingDataClipCountGyr	clipCountGyro

If *exportDataFormat* is not set, the default value is:

- XSRecordingDataTimestamp
- XSRecordingDataEulerAngles
- XSRecordingDataAcceleration
- XSRecordingDataAngularVelocity

```
UInt8 bytes[4] = { XSRecordingDataTimestamp, XSRecordingDataQuaternion
, XSRecordingDataDq , XSRecordingDataDv };
NSData *exportData = [NSData dataWithBytes:defaultBytes
length:sizeof(bytes)];
DotDevice.exportDataFormat = exportData;
```

4.13.2 Get export file information

You can use the *getExportFileInfo* method to get the start time UTC of the recording files. *DotDevice.recording.files* will be initialized only after calling *DotDevice getFlashInfo*. Since part of the file information is written in flash information.

```
 [DotDevice getExportFileInfo];
 [DotDevice setExportFileInfoDone:^(BOOL success) { }];
```

4.13.3 Set export file list

All the available recording files will be saved in *recording.files* array. Export file list is a *DotDevice.recording.files* index array. You must set export file list via recording.exportFileList before start export file data, otherwise no files will be exported. The max index cannot exceed *DotDevice.recording.files.count*.

```
NSArray *array = [[NSArray alloc] initWithObjects:@0, @1, @2, nil];
DotDevice.recording.exportFileList = array;
or
NSArray *array = [[NSArray alloc] initWithObjects:@2, nil];
DotDevice.recording.exportFileList = array;
```

4.13.4 Start export

Before starting to export thedata file, make sure you have set the export data format and get the export file information.

```
[DotDevice startExportFileData];
```

Call *ExportFileDone* block after the data export is finished:

```
[DotDevice.recording setExportFileDone:^(NSUInteger index, BOOL
result) { }];
```

If you want to use recording export file data, call parse file data block:

```
[DotDevice setDidParseExportFileDataBlock:^(DotPlotData * _Nonnull
plotData) { }];
```

4.13.5 Stop export

Stop export file data.

```
[DotDevice stopExportFileData];
```

4.13.6 Update export status

UpdateExportingStatus block will be updated after calling *startExportFileData* and *stopExportFileData*.

```
[DotDevice.recording setUpdateExportingStatus:^(XSExportStatus status)
{ }];
```

Table 19: Export status

Export status	Description
XSExportIsExportingData	The flash is exporting recording data
XSExportIsStopExportingData	Data exporting is stopped