PrivacyStreams Walkthrough

https://privacystreams.github.io/

Example

Suppose we want to detect the sleep state based on microphone loudness. Our task is to get loudness every 10 minutes.

4 steps to access data

1. Know what data you want to access and why;
2. Find a provider here*;
3. Add required permissions according to the provider;
4. Get data using `uqi.getData(Provider, Purpose)`.

*https://privacystreams.github.io/items.html

Example: accessing data

Step 1: Know what data you want to access and why;

- What data: record audio periodically;
- Why: monitoring sleep.

Example: accessing data

Step 2: Find a provider here*;

`MStreamProvider`	`Audio.recordPeriodic(long durationPerRecord, long interval)` Provide a live stream of Audio items. The audios are recorded from microphone periodically every certain time interval, and each Audio item is a certain duration of time long. For example, `recordPeriodic(1000, 4000)` will record audio from 0s-1s, 5s-6s, 10s-11s, ... This provider requires `Manifest.permission.RECORD_AUDIO` permission. - `durationPerRecord`: the time duration of each audio record, in milliseconds. - `interval`: the time interval between each two records, in milliseconds.

*https://privacystreams.github.io/items.html

Example: accessing data

Step 3: Add permissions according to the provider;

`MStreamProvider`	`Audio.recordPeriodic(long durationPerRecord, long interval)` Provide a live stream of Audio items. The audios are recorded from microphone periodically every certain time interval, and each Audio item is a certain duration of time long. For example, `recordPeriodic(1000, 4000)` will record audio from 0s-1s, 5s-6s, 10s-11s, ... This provider requires `Manifest.permission.RECORD_AUDIO` permission. - `durationPerRecord`: the time duration of each audio record, in milliseconds. - `interval`: the time interval between each two records, in milliseconds.

```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
```

Example: accessing data

Step 4: Get data using:

```
uqi.getData(Provider, Purpose)
```

```
UQI uqi = new UQI(context);
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .debug();
```

Provider

Purpose

4 steps to process data

1. Understand the concept of Stream, Item, Field;
2. Know the format of the stream you accessed;
3. Find proper Transformations here* ;
4. Find operators at here** or customize operators.

*https://privacystreams.github.io/pipeline.html
**https://privacystreams.github.io/operators.html

Example: processing data

Step 1. Understand Stream, Item, and Field

MStream
(multi-item stream)

SStream
(single-item stream)

Item

```
// An example of an Audio Item.
{
    Long "time_created": 1489528276655,
    Long "timestamp": 1489528266640,
    AudioData "audio_data": <AudioData@12416728>
}
```

Field

Example: processing data

Step 2. Know the format of the stream you accessed:

Option 1. Look up in documentation*

Reference	Name	Type	Description
`Audio.TIME_CREATED`	`"time_created"`	`Long`	The timestamp of when this item is created. It is a general field for all items.
`Audio.TIMESTAMP`	`"timestamp"`	`Long`	The timestamp of when the audio/record was generated.
`Audio.AUDIO_DATA`	`"audio_data"`	`AudioData`	The abstraction of audio data. The value is an `AudioData` instance.

Option 2. Use `debug()` method (item will be printed in logcat).

```
UQI uqi = new UQI(context);
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .debug();
```

*https://privacystreams.github.io/items.html

Example: processing data

Step 3. Find proper Transformations here*

For example, the `setField` transformation:

`MStream->MStream`

`setField(String fieldToSet, Function<Item,TValue> functionToComputeValue)`

Set a field to a new value for each item in the stream. The value is computed with a function that take the item as input. Eg. `setField("x", Comparators.gt("y", 10))` will set a new boolean field "x" to each item, which indicates whether the "y" field is greater than 10.

- `fieldToSet`: the name of the field to set, it can be a new name or an existing name.
- `functionToComputeValue`: the function to compute the new field value
- `<TValue>`: the type of the new field value

How to use:

```
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .setField("loudness", xxx)
        .debug();
```

*https://privacystreams.github.io/pipeline.html

Example: processing data

Step 4. Find proper operators here**

For example, `calcLoudness` operator:

`Function<Item,Double>`	`AudioOperators.calcLoudness(String audioDataField)` Calculate the average (RMS) loudness of the audio specified by an AudioData field. The loudness is an double number indicating the sound pressure level in dB. - `audioDataField`: the name of the AudioData field.

To use it with `setField` transformation:

```
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .setField("loudness", AudioOperators.calcLoudness("audio_data"))
        .debug();
```

New field type: Double

Existing AudioData-type field

**https://privacystreams.github.io/operators.html

4 steps to output data

1. Know what data you want to output and how (immediately or asynchronously);
2. Know the format of the stream after processing;
3. Find a proper Action here* ;
4. Find operators at here** or customize operators.

*https://privacystreams.github.io/pipeline.html
**https://privacystreams.github.io/operators.html

Example: outputting data

Step 1. Know what data you want to output and how

- What data: loudness
- How: asynchronously (non-blocking)

Example: outputting data

Step 2. Know the stream format after processing:

Option 1. Use `debug()` method (item will be printed in logcat).

```
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .setField("loudness", AudioOperators.calcLoudness("audio_data"))
        .debug();
```

Option 2. Infer based on code.

- setField(“loudness”, calcLoudness(…)) sets the output of `calcLoudness` to “loudness” field;
- The output type of `calcLoudness` is Double;
- ==> The type of “loudness” field is Double.

Example: outputting data

Step 3. Find a proper Action here*

For example, the `forEach` action:

`MStream->void`	Non-blocking	`forEach(String fieldToSelect, Callback<TValue> callback)` Callback with a certain field of each item. - `fieldToSelect`: the name of the field to callback with - `callback`: the callback to invoke for each item field - `<TValue>`: the type of the field

How to use:

```
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .setField("loudness", AudioOperators.calcLoudness("audio_data"))
        .forEach("loudness", xxx);
```

*https://privacystreams.github.io/pipeline.html

Example: outputting data

Step 4. Find proper operators or define your own.

For example, using a customized callback:

```
uqi.getData(Audio.recordPeriodic(10*1000, 10*60*1000), Purpose.HEALTH("monitoring sleep."))
        .setField("loudness", AudioOperators.calcLoudness("audio_data"))
        .forEach("loudness", new Callback<Double>() {
            @Override
            protected void onInput(Double input) {
                System.out.println("Current loudness is " + input + " dB.");
            }
        });
```