App Note: Schedules

05/15/2017

Introduction

Each device may contain one or more schedules. Schedules, and their associated ScheduleActions, provide the ability to programmatically change the value of a device property in the future, for example, turn on the light at 7:00 PM every weekday.

AylaSchedule

When using Android library all the Schedules are accessed using AylaSchedule object. There are 2 different types of Schedules viz. Fixed Schedule and Dynamic Schedule. These depend on the OEM provider requirements.

Fixed Schedule
Most common cases are fixed Schedules where you cannot create or delete Actions for the AylaSchedule. They are provided to devices via the device template when the device is registered. These Actions are set during manufacture time and cannot be created or destroyed but can be updated. In case of Fixed Schedule first fetch the Schedule from the service and then update the Schedule and its corresponding ScheduleActions.

Dynamic Schedule
For Dynamic Schedule the ScheduleActions can also be created and deleted from the application. It allows all CRUD methods viz. create, fetch, update and delete Actions for that Schedule

These are all the methods that can be called on AylaDevice Object in the SDK relating to Schedules

1. **Fetch Schedules**
   This is used for fetching an Array of Existing Schedule for a given AylaDevice from the service.

```
fetchSchedules(final Response.Listener<AylaSchedule[]>
successListener,final ErrorListener errorListener)
```

params:

Response.Listener<AylaSchedule[]> successListener : A Listener to Receive successful fetch of AylaSchedules
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

2. Update Schedules

This is used for updating an AylaSchedule for a given AylaDevice from the service.

updateSchedule(final AylaSchedule schedule,final Response.Listener<AylaSchedule> successListener,final ErrorListener errorListener)

params:

AylaSchedule schedule: An Existing Schedule object with all the desired values set
Response.Listener<AylaSchedule> successListener : A Listener to Receive on successful Update of Schedule
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

3. Enable a Schedule

Enable an Existing Schedule

EnableSchedule(final AylaSchedule schedule,final Response.Listener<AylaSchedule> successListener,final ErrorListener errorListener)

params:

AylaSchedule schedule: An existing AylaSchedule
Response.Listener<AylaSchedule> successListener : A Listener to Receive on successful Enabling of Schedule
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

4. Disable a Schedule

Disable an Existing Schedule

DisableSchedule(final AylaSchedule schedule,final Response.Listener<AylaSchedule> successListener,final

ErrorListener errorListener)

params:
AylaSchedule schedule: An existing AylaSchedule
Response.Listener<AylaSchedule> successListener : A Listener to Receive on successful Disabling of Schedule
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

AylaScheduleAction

A Schedule has one or more ScheduleAction associated with it. A ScheduleAction provides an ability to change a value of Device property programmatically in the future. A ScheduleAction is tied to a Device Property. Just like Schedules for a Fixed Schedule ScheduleActions are precreated in the OEM template.

These are the methods that can be called on AylaSchedule

1. Create a ScheduleAction

This is only for Dynamic Schedules. It is invoked on AylaSchedule object

createAction(final AylaScheduleAction scheduleAction,final Response.Listener<AylaScheduleAction> successListener,final ErrorListener errorListener)

params:
AylaScheduleAction scheduleAction: A new Schedule Action object with all the desired values set
Response.Listener<AylaSchedule> successListener : A Listener to Receive on successful Creation of ScheduleAction
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

2. Fetch Actions

This is used for fetching an Array of Existing ScheduleActions for a given Schedule from the service.

fetchActions(final Response.Listener<AylaScheduleAction[]> successListener,final ErrorListener errorListener)

params:
Response.Listener<AylaScheduleAction[]> successListener : A

Listener to Receive successful fetch of AylaScheduleActions
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

3. **Update Actions**
   This is used for updating an array of existing AylaScheduleActions

updateActions(final AylaScheduleAction[] scheduleActions,final Response.Listener<AylaScheduleAction[]> successListener,final ErrorListener errorListener)

params:
AylaScheduleAction[]scheduleActions: An Array of existing ScheduleActions that need to be updated
Response.Listener<AylaScheduleAction[]> successListener : A Listener to Receive on successful Update of ScheduleActions
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

4. **Delete a Action**
   This is only for Dynamic Schedules. It is used for deleting an existing AylaScheduleAction

deleteAction(final AylaScheduleAction scheduleAction,final Response.Listener<AylaAPIRequest.EmptyResponse> successListener,final ErrorListener errorListener)

params:
AylaScheduleAction scheduleAction: An existing AylaScheduleAction
Response.Listener<AylaAPIRequest.EmptyResponse > successListener : A Listener to Receive deletion of ScheduleAction
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

5. **Delete All Actions**
   This is only for Dynamic Schedules. It is used for deleting all ScheduleActions for a given AylaSchedule

deleteAllActions(finalResponse.Listener<AylaAPIRequest.EmptyResp

onse> successListener,final ErrorListener errorListener)

params:
Response.Listener<AylaAPIRequest.EmptyResponse > successListener : A Listener to Receive on Deletion of All Actions
ErrorListener errorListener: An Error Listener should one occur.

This method returns an AylaAPIRequest

Common Setters and Getters for AylaSchedule

These are the common getter and setters for AylaSchedule

public String getDirection();
public String getName();
public String getDisplayName();
public boolean isActive();
public boolean isUtc();
public String getStartDate();
public String getEndDate();
public String getStartTimeEachDay();
public String getEndTimeEachDay();
public int[] getDaysOfWeek();
public int[] getDaysOfMonth();
public int[] getMonthsOfYear();
public int[] getDayOccurOfMonth();
public int getDuration();
public int getInterval();
public boolean hasFixedActions();

```
public void setDirection(String direction);
public void setName(String name);
public void setDisplayName(String displayName);
public void setActive(boolean active);
public void setUtc(boolean utc);
public void setStartDate(String startDate);
public void setEndDate(String endDate);
public void setEndTimeEachDay(String endTimeEachDay);
public void setDaysOfWeek(int[] daysOfWeek);
public void setDaysOfMonth(int[] daysOfMonth);
public void setMonthsOfYear(int[] monthsOfYear);
```

```
public void setDayOccurOfMonth(int[] dayOccurOfMonth);
public void setDuration(int duration);
public void setInterval(int interval);
public void setStartTimeEachDay(String startTimeEachDay);
```

Common Setters and Getters for AylaScheduleAction

```
public String getName();
public String getType();
public boolean isInRange();
public boolean isAtStart();
public boolean isAtEnd();
public boolean isActive();
public String getBaseType();
public String getValue();

public void setName(String name);
public void setType(String type);
public void setActive(boolean active);
public void setBaseType(String baseType);
public void setValue(String value);
public void
setScheduleActionFirePoint(AylaScheduleActionFirePoint
scheduleActionFirePoint);
```

Common Usage of Schedules

Normally OEM preconfigures the schedules so the end developer needs to fetch the Schedule, change when the schedule is fired. Check the java code in ScheduleFragment in Android Aura source code. We need to have a AylaDevice object to get all the schedules for that device. Fetching of Schedules is done by calling

```
device.fetchSchedules(
        new Response.Listener<AylaSchedule[]>() {
            @Override
            public void onResponse(AylaSchedule[] response) {
            //Go through the array of Schedules and check the
Schedule you want to change
         },new ErrorListener() {
             @Override
```

```
            public void onErrorResponse(AylaError error) {
            //Log this error or Show a Toast message

           }
```

Once the Schedule is fetched from the service use the Setters described above for AylaSchedule to Change when Schedule is fired viz. `setStartDate, setEndDate, setDuration etc.`

Then these changed values are saved in the service by calling method UpdateSchedule

```
device.updateSchedule(aylaSchedule,
        new Response.Listener<AylaSchedule>() {
            @Override
            public void onResponse(AylaSchedule response) {
           //Updated schedule is returned in the response
         },new ErrorListener() {
             @Override
           public void onErrorResponse(AylaError error) {
           //Log this error or Show a Toast message

          }
```

In similar way ScheduleActions need to be fetched by calling fetchActions on the AylaSchedule we fetched using `fetchSchedules.`

```
schedule.fetchActions(
        new Response.Listener<AylaScheduleAction[]>() {
        @Override
         public void onResponse(AylaScheduleAction[] response) {
            //Go through the array AylaScheduleAction and check
//the ScheduleAction that needs to be changed
         },new ErrorListener() {
             @Override
           public void onErrorResponse(AylaError error) {
           //Log this error or Show a Toast message

          }
```

Once we get the ScheduleAction from the fetch method above use the Setters for AylaScheduleAction described in the document above. Most common usage is

changing the Value using method setValue and setting the ScheduleActionFirePoint (viz. AtStart, AtEnd, InRange) using method setScheduleActionFirePoint.

Once the needed ScheduleActions are changed they are saved back to Service using updateActions method on AylaSchedule

```
schedule.updateActions(schduleActions[],
        new Response.Listener<AylaScheduleAction[]>() {
        @Override
        public void onResponse(AylaScheduleAction[]response)  {
           //Updated scheduleAction array is returned in the
//response
         },new ErrorListener() {
             @Override
           public void onErrorResponse(AylaError error) {
           //Log this error or Show a Toast message

          }
```