Programming Guide:
Samsung Digital Health
- S Health Service

1.2.0
# Table of Contents

1. **OVERVIEW .......................................................................................................................... 4
   1.1. ARCHITECTURE .................................................................................................................. 4
   1.2. CLASS RELATIONSHIP ........................................................................................................ 5

2. **DEVELOPMENT ENVIRONMENT ......................................................................................... 7
   2.1. PREREQUISITES .................................................................................................................. 7
       - ANDROID VERSION ........................................................................................................ 7
       - AVAILABLE DEVICES ..................................................................................................... 7
       - S HEALTH VERSION ......................................................................................................... 7
   2.2. DOWNLOADING SAMSUNG DIGITAL HEALTH SDK ............................................................. 7

3. **DEVELOPMENT PROCESS .................................................................................................... 8
   3.1. CHECKING S HEALTH’S TRACKERS ................................................................................... 8
   3.2. CREATING AN APPLICATION ............................................................................................ 8
       - DATA SYNCHRONIZATION WITH S HEALTH .................................................................... 8
   3.3. TESTING YOUR APPLICATION ....................................................................................... 8
       - 3.3.1. S HEALTH’S DEVELOPER MODE .............................................................................. 9
       - 3.3.2. CHECKLIST FOR S HEALTH’S PARTNER APP .......................................................... 10
   3.4. PUBLISHING YOUR APPLICATION .................................................................................. 10
   3.5. REQUESTING FOR PARTNER APPS .................................................................................. 10

4. **HELLO TRACKER PROVIDER ................................................................................................ 11
   4.1. IMPORTING LIBRARIES .................................................................................................... 11
   4.2. CONNECTION TO S HEALTH ............................................................................................ 11
   4.3. INITIALIZATION ................................................................................................................ 11
   4.4. FEATURE AVAILABILITY .................................................................................................. 12
   4.5. POSTING YOUR APP’S TRACKER ..................................................................................... 12
       - MANIFEST FOR TRACKER DEFINITION ......................................................................... 12
       - REGISTRATION OF PLUGIN SERVICE .............................................................................. 13
       - HANDLING EVENT FROM S HEALTH ............................................................................... 13
       - CREATING A TRACKERTILEMANAGER’S INSTANCE ......................................................... 14
       - CREATING TRACKER TILE’S INTENT ............................................................................... 15
       - POSTING A TRACKER TILE .............................................................................................. 16
       - UPDATING THE POSTED TRACKER TILE ......................................................................... 17
   4.6. LAUNCHING S HEALTH’S TRACKER .................................................................................. 19
       - CREATING A TRACKERMANAGER’S INSTANCE ................................................................. 19
       - GETTING S HEALTH’S TRACKER INFO ............................................................................ 20
       - LAUNCHING THE AVAILABLE TRACKER ......................................................................... 20

5. **TRACKER AND TRACKER TILE ............................................................................................. 22
   5.1. TRACKER ............................................................................................................................ 22
   5.2. TRACKER TILE .................................................................................................................. 22
   5.3. TRACKER MANAGER ......................................................................................................... 23
       - LIMITATIONS .................................................................................................................. 24

6. **DEFINING APP’S TRACKER ................................................................................................ 25
   6.1. TRACKER DEFINITION ....................................................................................................... 25
   6.2. TRACKER TILE TEMPLATES .............................................................................................. 27
       - TRACKER TILE_TYPE_1 ...................................................................................................... 28
       - TRACKER TILE_TYPE_2 ...................................................................................................... 28
1. Overview

*S Health* is an application that monitors the user’s activities and helps the user has a healthier life. *S Health*’s collected data can be categorized and expressed in various ways. It is important to present proper information to the user in the required time for advanced experiences. *S Health 4.x* supports Android devices with KitKat 4.4 including non-Samsung devices.

Samsung Digital Health SDK enables your app to work with *S Health 4.x*. It provides the following packages.

- Health Data package
- S Health Service package

The SDK can be downloaded from the developer site. See development environment for more information.

This document contains descriptions for the SDK’s S Health Service package. See the Health Data package’s documents if you want to synchronize health data with *S Health*.

The S Health Service package of Samsung Digital Health SDK provides the tracker feature to:

- provide the app’s tracker to the *S Health*
- launch the *S Health’s* available tracker

The following glossary in Table 1 is for the S Health Service package.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>S Health</em></td>
<td>An application that helps monitor the user’s activities and helps the user makes a healthier life. It contains the ‘main screen’ area that shows tracker tiles. It’s written as italic in this document.</td>
</tr>
<tr>
<td>tracker</td>
<td>The tracker is the basic building block of <em>S Health</em>. See 5.1 for more information.</td>
</tr>
<tr>
<td>tracker tile</td>
<td>The tracker tile is a unit of viewer for the specific tracker on <em>S Health</em>. It represents the latest data collected by its tracker and can be a gateway. See 6.2 for more information.</td>
</tr>
<tr>
<td>tracker items</td>
<td>It’s the item list that shows all trackers in <em>S Health</em>.</td>
</tr>
</tbody>
</table>

Table 1: Glossary

1.1. Architecture

The S Health Service package is composed of the following features roughly.

- Tracker manager
- Tracker tile manager

The tracker manager provides information the *S Health*’s tracker and helps to launch the *S Health*’s available tracker.

The tracker tile manager provides the plugin service to install your app’s tracker on *S Health*. Your tracker is shown in the tracker item list page of *S Health* with existing trackers if the application is registered as the partner application of *S Health*. If the user subscribes to your app’s tracker, its tile is posted to the *S Health*’s main screen.
Figure 1 shows the architecture of the S Health Service package and its main features are described below.

![Architecture for S Health Service of Samsung Digital Health]

**1.2. Class Relationship**

The S Health Service library provides the following packages:

- `com.samsung.android.sdk.shealth`
- `com.samsung.android.sdk.shealth.tracker`

All APIs of the `com.samsung.android.sdk.shealth.tracker` package can be used after the following API succeeds.

- `Shealth.initialize(Context)` of the `com.samsung.android.sdk.shealth`

`com.samsung.android.sdk.shealth.tracker` enables your app to use *S Health*'s tracker feature.

Figure 2 shows the relationship between classes and interfaces of S Health Service.

![Class relationship of S Health Service]

Your app can define one tracker and S Health Service’s Plugin Service adds the tracker into the *S Health*'s tracker items. `TrackerEventListener` receives *S Health*'s tracker events. `TrackerManager` can post your tracker as `TrackerTile` with templates.

`TrackerManager` gets *S Health*'s tracker information and helps your app to launch the tracker if it is available. Note, it works on *S Health* 4.8 or above.
Interfaces and classes of the S Health Service package are described in the following table. For detailed information of each class and interface, refer to the API Reference under the [SamsungDigitalHealthSDK]/SHealthService/Docs/API_Reference folder.

<table>
<thead>
<tr>
<th>Interface / Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shealth</td>
<td>This class is a representative class of the S Health Service library. It helps on the usage of APIs in the com.samsung.android.sdk.shealth.tracker package.</td>
</tr>
<tr>
<td>TrackerEventListener</td>
<td>This interface defines event handlers related to your app’s tracker in S Health.</td>
</tr>
<tr>
<td>TrackerInfo</td>
<td>This class provides tracker information.</td>
</tr>
<tr>
<td></td>
<td>It works on S Health 4.8 or above.</td>
</tr>
<tr>
<td>TrackerManager</td>
<td>This class provides S Health's tracker information and help to launch the S Health's available tracker.</td>
</tr>
<tr>
<td></td>
<td>It works on S Health 4.8 or above.</td>
</tr>
<tr>
<td>TrackerTile</td>
<td>This class defines the app tracker's tile.</td>
</tr>
<tr>
<td>TrackerTileManager</td>
<td>This class is used to post or remove a tracker tile on the S Health.</td>
</tr>
</tbody>
</table>

Table 2: Interfaces and classes of the S Health Service library
2. Development Environment

To develop applications with Samsung Digital Health SDK, check prerequisites first.

2.1. Prerequisites

Check following prerequisites before downloading Samsung Digital Health SDK.

Android Version

Android 4.4 KitKat (API Level 19) or above

Available Devices

Android smartphones including non-Samsung devices that are available for S Health 4.x

S Health Version

A Samsung Digital Health application runs with S Health 4.0 or above.

2.2. Downloading Samsung Digital Health SDK

Samsung Digital Health SDK can be downloaded on the Samsung developer site and you can find the following content related to the S Health Service package for the application development.

<table>
<thead>
<tr>
<th>Folder in SDK</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Docs</td>
<td></td>
</tr>
<tr>
<td>API Reference</td>
<td>Describes S Health Service APIs of Samsung Digital Health</td>
</tr>
<tr>
<td>Programming Guide</td>
<td>Contains development information to create plugin applications with S Health Service</td>
</tr>
<tr>
<td>Tracker Design Guidelines</td>
<td>Guidelines for designing the tracker and tracker tile that are shown on S Health</td>
</tr>
<tr>
<td>Libs</td>
<td></td>
</tr>
<tr>
<td>samsung-digital-health-shealth-wa.b.c.jar</td>
<td>S Health Service library of Samsung Digital Health</td>
</tr>
<tr>
<td>sdk-v1.0.0.jar</td>
<td>Basic library for the Samsung SDK</td>
</tr>
<tr>
<td>Samples</td>
<td></td>
</tr>
<tr>
<td>PluginTracker</td>
<td>Sample application created by using S Health Service library. This application can post the sample tracker tile and launch the S Health’s tracker.</td>
</tr>
</tbody>
</table>

Table 3: S Health Service content of Samsung Digital Health
3. Development Process

If you checked prerequisites and downloaded the SDK, it’s ready to create your application. Make it with the following steps.

3.1. Checking S Health’s Trackers

The S Health Service package is related deeply to the S Health’s tracker feature. S Health provides useful trackers found in S Health > Manage items. Run S Health and check:

- the tracker operation on S Health
- S Health’s tracker kinds before defining a new tracker for your app

Note

Existing S Health’s trackers need to be considered first before creating an application. A duplicated tracker definition for your app may be rejected as S Health’s partner apps.

3.2. Creating an Application

Create your app with the SDK. Hello Tracker Provider shows how to use the S Health Service package.

Data Synchronization with S Health

Health data synchronization with S Health is essential to use the SDK. The SDK supports useful data types.

- Predefined data types

If your desired data type for your app, you can read S Health’s data or write your app’s data to S Health through the Health Data package. An app that defines its own tracker without synchronizing data with S Health can be rejected as S Health’s partner apps. See the following documents for the Health Data package.

- Health Data API Reference
- Programming Guide - Health Data

3.3. Testing Your Application

Though you’ve created an application with Samsung Digital Health SDK, the application can only work with S Health after the application is approved as S Health’s partner apps. To register your application as S Health’s partner app, see 3.5. You can test your application with S Health’s developer mode.
3.3.1. S Health’s Developer Mode

*S Health* 4.x provides the developer mode to help your app’s test before it is registered as *S Health’s* partner application. Refer to 3.3.2 to register your application as the partner application.

The *S Health’s* developer mode is not active by default. You can use the developer mode as the following steps.

1) Select the action overflow of *S Health* on the top-right side.

2) Find **Settings > About S Health** in the action list.

3) Tap the version region quickly for 10 times.
   The exact region, illustrated in the red and blue box of Figure 3 needs to be tapped.

4) If it succeeds, "*(Developer Mode)*" is shown in front of the version and it means the developer mode is activated. Now you can test your application on *S Health*.

Figure 3 shows how to turn on or off the developer mode in *S Health*. If you tap the version region quickly for 10 times in the developer mode as the right figure, the developer mode is deactivated.

Figure 3: On or off of S Health’s developer mode

It’s ready to test your application if the developer mode of *S Health* is activated.
3.3.2. Checklist for S Health’s Partner App

Samsung Digital Health provides a checklist to test an application for the S Health’s partner application that contains basic test items. Download it here and check your application if it satisfies checklist items. It helps you save time on the registration process for S Health’s partner app.

3.4. Publishing Your Application

If your application is ready, make a package and publish it on the app market such as Google Play.

3.5. Requesting for Partner Apps

The application will work properly with S Health after the partner application registration is approved. Because health data is closely connected to the privacy issue, Samsung checks violations in your application and registers your application as S Health’s partner application.

You can request for the partner application on the developer site.

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only the approved partner application runs with S Health 4.x. Otherwise the application can be run tested on the developer mode.</td>
</tr>
</tbody>
</table>
4. Hello Tracker Provider

This chapter describes how to create a simple tracker provider with the SDK. Create a new Android application project and follow the basic steps below.

- Importing Libraries
- Connection to S Health
- Initialization
- Feature Availability
- Posting Your App’s Tracker

Detailed descriptions are below.

4.1. Importing Libraries

Not only the S Health Service package’s library, but also the Health Data package’s one are required to use the tracker feature. Add the following libraries to the "libs" folder in your created application project.

- samsung-digital-health-healthdata-a.b.c.jar
- samsung-digital-health-shealth-va.b.c.jar
- sdk-v1.0.0.jar

4.2. Connection to S Health

Samsung Digital Health SDK works with S Health 4.0 or above. S Health installation and the installed S Health’s version need to be checked before using the S Health Service package’s APIs.

See the related FAQ or Health Data’s Programming Guide’s 4.2 for more information.

4.3. Initialization

Initializing the S Health Service with the following API is needed to use the S Health Service package’s other APIs.

- initialize(Context) of the com.samsung.android.sdk.shealth.Shealth class.

```java
import com.samsung.android.sdk.shealth.Shealth;
import android.app.Application;
import android.content.Context;
import android.util.Log;

public class PluginTracker extends Application {
    private static final String APP_TAG = "PluginTracker";

    @Override
    public void onCreate() {
        super.onCreate();
    }
}
```
**4.4. Feature Availability**

A certain S Health Service’s feature works on the specific S Health’s version or above because the tracker feature is deeply integrated to S Health. E.g., FEATURE_TRACKER_LAUNCH works on S Health 4.8 or above.

Check the feature availability of your desired feature in S Health Service. If any feature is not available, S Health should be upgraded to the latest version.

```java
// Check the feature availability for all wanted features.
if (mShealth.isFeatureEnabled(Shealth.FEATURE_TRACKER, Shealth.FEATURE_TRACKER_LAUNCH)) {
   // All wanted features are available.
   // You can use all APIs of the "com.samsung.android.sdk.shealth" package.
} else {
   // If any feature is not available, S Health should be upgraded.
   Log.d(APP_TAG, "S Health should be upgraded");

   // Let the user upgrade S Health to the latest version.
   Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("market://details?id=com.sec.android.app.shealth"));
   this.startActivity(intent);
}
```

**4.5. Posting Your App’s Tracker**

If you cannot find a wanted tracker on S Health, you can define your app’s own tracker.

**Manifest for Tracker Definition**

A new tracker can be defined in the manifest file as shown below. An ID, displayed name, icon and controller are required for the tracker definition. For more information, refer to Tracker Definition.

```xml
<resources>
  <string name="testtracker_manifest">
    { "tracker" : {
      "id" : "tracker.test",
      "display-name" : "tracker_display_name",
      "icon" : "tracker_icon",
      "controller" : "com.samsung.android.app.plugintracker.MyTracker"
    }
  }
</resources>
```
Registration of Plugin Service

The tracker of your application is installed as plugin for S Health by registering it for the plugin service with your defined tracker ID and its resource name in AndroidManifest.xml of your application project.

```
<service
    android:name="com.samsung.android.sdk.shealth.PluginService"
    android:exported="true">
    <meta-data
        android:name="tracker.test" android:value="@string/testtracker_manifest"/>
</service>
```

- **android:name** - the tracker ID of the tracker manifest file.
- **android:value** - the resource name of the tracker manifest file.

Handling Event from S Health

S Health sends its events to your application through TrackerEventListener and your app can handle them by implementing TrackerEventListener.

TrackerEventListener can be implemented basically as shown below.

```java
public class MyTracker implements TrackerEventListener {

    private TrackerTileManager mTrackerTileManager;

    public MyTracker() {
        // A default constructor has to be created.
    }

    @Override
    public void onCreate(Context context, String trackerId) {
        Log.d(APP_TAG, "onCreate( + trackerId + ")");

        if (mTrackerTileManager == null) {
            try {
                mTrackerTileManager = new TrackerTileManager(context);
            } catch (IllegalArgumentException e) {
                Log.d(APP_TAG, "MyTracker onCreate() - IllegalArgumentException");
            }
        }
    }

    @Override
    public void onSubscribed(Context context, String trackerId) {
        Log.d(APP_TAG, "onSubscribed( + trackerId + ");
        // User changed the subscription state of your tracker to subscribe.

        // Post your tracker tile here to show it on the S Health's main screen
        // right after user subscribed your tracker.

        // If you miss posting your tracker tile here and post it onTileRequested(),
        // your tracker tile will not be shown on the main screen
        // until onTileRequested() is called.
```
// TRACKER_TILE_TYPE_1 is used generally if there is no data yet.
}

@Override
public void onUnsubscribed(Context context, String trackerId) {
    Log.d(APP_TAG, "onUnsubscribed(" + trackerId + ")");
    // User changed the subscription state of your tracker to unsubscribe.
}

@Override
public void onPause(Context context, String trackerId) {
    Log.d(APP_TAG, "onPause(" + trackerId + ")");
    // S Health went to the background.
    // Stop posting tracker tiles.
}

@Override
public void onTileRequested(Context context, String trackerId, String tileId) {
    Log.d(APP_TAG, "onTileRequested(" + trackerId + ", " + tileId + ")");
    // S Health has been resumed on the foreground.
    // If tileId is null, it means there is no posted tracker tile successfully on the main screen.
    // Set the tracker ID if it’s null.

    // Post your tracker tile here.
    // TRACKER_TILE_TYPE_1 is used generally if there is no data yet.

    // Or you can update the posted tracker tile with updated data here.
}

@Override
public void onTileRemoved(Context context, String trackerId, String tileId) {
    Log.d(APP_TAG, "onTileRemoved(" + trackerId + ", " + tileId + ")");
    // User removed your posted tracker tile from the main screen.
}

When you implement TrackerEventListener, the implemented class’s default constructor is mandatory. E.g. if implemented class name is 'MyTracker', create its default constructor as shown below. Or your application won’t receive S Health’s events well.

public class MyTracker implements TrackerEventListener {
    public MyTracker() {
        // An empty constructor should be created.
    }
}

Creating a TrackerTileManager’s Instance

TrackerTileManager enables your application to post or remove your tracker’s tile.

public class MyTracker implements TrackerEventListener {
    private TrackerTileManager mTrackerTileManager;

    public MyTracker(Context context) {
        if (mTrackerTileManager == null) {
            try {

```
Creating Tracker Tile’s Intent

If you want to add service intent for a tile’s button to be performed in background, the service intent can be defined as the following example.

```java
final public class MyTrackerService extends IntentService {

    private static final String APP_TAG = "PluginTracker";
    private static final String SHARED_PREFERENCE_NAME = "tile_content";
    private static final String SHARED_PREFERENCE_CONTENT_VALUE_KEY = "content_value";
    private static final String VALIDATION_KEY = "validation_key";

    @Override
    protected void onHandleIntent(Intent intent) {
        if (intent == null) {
            return;
        }

        String trackerId = intent.getStringExtra(TrackerTileManager.EXTRA_TRACKER_ID);
        if (trackerId == null) {
            return;
        }

        String tileId = intent.getStringExtra(TrackerTileManager.EXTRA_TILE_ID);
        if (tileId == null) {
            return;
        }

        String validationValue = intent.getStringExtra(VALIDATION_KEY);
        SharedPreferences sp = getSharedPreferences(SHARED_PREFERENCE_NAME, Context.MODE_PRIVATE);
        String validationSavedValue = sp.getString(VALIDATION_KEY, "");
        if (validationValue.isEmpty() || !validationValue.equals(validationSavedValue)) {
            Log.d(APP_TAG, "invalid validation value");
            return;
        }

        int tileContent = sp.getInt(SHARED_PREFERENCE_CONTENT_VALUE_KEY, 0) + 1;
        Log.d(APP_TAG, "content value : " + String.valueOf(tileContent));
        sp.edit().putInt(SHARED_PREFERENCE_CONTENT_VALUE_KEY, tileContent).apply();

        MyTracker tracker = new MyTracker(this);
        tracker$updateTile(this, trackerId, tileId);
    }
```
Posting a Tracker Tile

If the user subscribes to your tracker on the S Health’s tracker item list, S Health sends an event to onSubscribed() of your application. Post your tracker tile in this event handler to show your tracker tile on the S Health’s main screen right after the user subscribes to your tracker. Keep in mind that a tracker tile will not be posted if the S Health’s main screen is in pause state.

```java
import android.app.Activity;
import android.content.Intent;
import android.graphics.drawable.Icon;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ImageButton;

public class MyTracker implements TrackerEventListener {

    private static final String MY_TILE_ID = "hello_tile";

    @Override
    public void onSubscribed(Context context, String trackerId) {
        Log.d(APP_TAG, "onSubscribed(" + trackerId + ")");
        postDefaultTile(context, trackerId, MY_TILE_ID);
    }
}
```

In case that there is no data for your tracker, use TRACKER_TILE_TYPE_1 with an icon, title and button. See 6.2 for more information. The following code shows how to post a TRACKER_TILE_TYPE_1 tracker tile.

```java
public class MyTracker implements TrackerEventListener {

    private int mTemplate = TrackerTile.TRACKER_TILE_TYPE_1;

    private void postDefaultTile(Context context, String trackerId, String tileId) {
        TrackerTile myTrackerTile;
        Intent launchIntent;
        Intent serviceIntent;

        // Your tracker tile is not posted successfully on the main screen of S Health.
        // Set the tile ID. Or the tile will not be shown though you call TrackerTileManager.post().
        if (tileId == null) {
            tileId = MY_TILE_ID;
        }

        try {
            // Create Intent to do an action
            // when the tracker tile is clicked
            Intent launchIntent = new Intent(context, MainActivity.class);

            // Create Intent to do an action
            // when the button on this tile is clicked
            Intent serviceIntent = new Intent(context, MyTrackerService.class);

            // Create TrackerTile with type 1 and set each values and intents
            myTrackerTile = new TrackerTile(context, trackerId, tileId, mTemplate);

            // Set Title
            myTrackerTile
                .setTitle(R.string.tracker_test_display_name) // Set icon resource
                .setIcon(R.drawable.tracker_icon) // Set content color
                .setContentColor(context.getResources().getColor(R.color.tracker_content_color)) // Set content intent
                .setContentIntent(TrackerTile.INTENT_TYPE_ACTIVITY, launchIntent) // Set button intent
                .setIcon(Icon.createWithResource(context, R.drawable.tracker_icon)) // Set icon resource
                .setTitle(R.string.tracker_test_display_name) // Set Title
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```
Make sure that S Health is on the developer mode before installing your application. S Health will suggest your new tracker on its main screen and you can add tracker easily by tapping the suggestion tile as Figure 4.

![Figure 4: Posted tracker tile on S Health](image)

**Figure 4: Posted tracker tile on S Health**

### Updating the Posted Tracker Tile

When the main screen is resumed on the foreground, implemented TrackerEventListener.onTileRequested() of your application receives an event from S Health. You can post a tracker tile or update the posted tile here.

```java
public class MyTracker implements TrackerEventListener {

    @Override
    public void onTileRequested(Context context, String trackerId, String tileId) {
        // The main screen requested the tracker tile.
        Log.d(APP_TAG, "onTileRequested(" + trackerId + ", " + tileId + ")");
        // Update your tracker tile.
    }
}
```
If the application has health related data to show it on the tracker tile, use the TRACKER_TILE_TYPE_2 or TRACKER_TILE_TYPE_3 tracker tile type.

If there is no posted tracker tile on the main screen, the tile ID is received as null in the event handler above. To post a tracker tile on the main screen, the tile ID must be specified.

The following code shows how to update the posted tracker tile with TRACKER_TILE_TYPE_3.

```java
public class MyTracker implements TrackerEventListener {

    public void updateTile(Context context, String trackerId, String tileId) {

        TrackerTile myTrackerTile = null;
        Intent launchIntent;

        // Set the measured value of the data
        int tileContentValue;
        // Set the measured time of the data
        int tileDate;

        // Your tracker tile is not posted successfully on the main screen of S Health.
        // Set the tile ID. Or the tile will not be shown though you call TrackerTileManager.post().
        if (tileId == null) {
            tileId = MY_TILE_ID;
        }

        try {
            // Create Intent to do an action
            // when the tracker tile is clicked
            launchIntent = new Intent(context, MainActivity.class);

            // Create Intent to do an action
            // when the button on this tile is clicked
            Intent serviceIntent = new Intent(context, MyTrackerService.class);

            // Set template
            mTemplate = TrackerTile.TRACKER_TILE_TYPE_3;

            // Create TrackerTile and set each values and intents
            myTrackerTile = new TrackerTile(context, trackerId, tileId, mTemplate);

            // Set Title
            myTrackerTile
                .setTitle(R.string.tracker_display_name)
                // Set Icon resource
                .setIcon(R.drawable.tracker_icon_30x30)
                // Set content value
                .setContentValue(String.valueOf(tileContentValue))
                // Set content unit
                .setContentUnit("LBS")
                // Set Date text
                .setDate(tileDate)
                // Set content color
                .setContentColor(Color.parseColor("#7CB342"))
                // Set content intent
        }
    }
```
If you use the same tile ID with the posted tracker tile and update the tile, the updated tracker tile will be shown in the same position of the main screen as Figure 5 where the tracker tile has been posted. Otherwise, the updated tracker tile will be shown as the last tile on the main screen.

![Updated tracker tile on S Health](image)

**Figure 5: Updated tracker tile on S Health**

### 4.6. Launching S Health’s Tracker

You can launch the *S Health*’s available tracker with the *S Health* Service package.

### Creating a TrackerManager’s Instance

Create a TrackerManager’s instance first. Launching the *S Health*’s tracker is supported in *S Health* 4.8 or above.

```java
public class MainActivity extends Activity {
    .setContentIntent(TrackerTile.INTENT_TYPE_ACTIVITY, launchIntent)
    // Set button intent
    .setButtonIntent("UPDATE", TrackerTile.INTENT_TYPE_SERVICE, serviceIntent);

    if (mTrackerTileManager != null) {
        mTrackerTileManager.post(myTrackerTile);
    }
}
```
private static final String APP_TAG = "PluginTracker";

private TrackerManager mTrackerManager;

private TrackerInfo getSpecificTrackerInfo(String trackerId) {
    // Construct a TrackerManager's instance
    try {
        mTrackerManager = new TrackerManager(this);
    } catch (IllegalArgumentException e) {
        Log.d(APP_TAG, "TrackerManager Constructor - " + e.toString());
    }
    // ...
```java
try {
    startActivity();
} catch (IllegalArgumentException e) {
    Log.d(APP_TAG, "startActivity() - " + e.toString());
} catch (IllegalStateException e) {
    Log.d(APP_TAG, "startActivity() - " + e.toString());
}
```
5. Tracker and Tracker Tile

The tracker and tracker tile are the basic features of *S Health*. Your app can show the measured data on the S Health’s tracker or launch it through the SDK. Understanding the tracker and tracker tile’s operation on *S Health* is required first.

5.1. Tracker

The tracker is the basic building block of *S Health*. User’s health data can be monitored, measured or updated through trackers such as counting steps, measuring heart rates, tracking exercises, collecting health data on background, or allowing the user to enter health data manually.

*S Health* has various trackers for its supported data types already as Figure 6. Check the supported trackers on the tracker tab of *S Health > Manage items*. The user can subscribe to the tracker by switching it on the tracker tab.

![Figure 6: S Health’s trackers](image)

5.2. Tracker Tile

The tracker is expressed to the user as the tracker tile. It is shown on the *S Health*’s main screen and represents the latest data collected by its tracker. It acts as a gateway when the user taps it. It’s jumped to:

- the tracker’s detailed page in case that the *S Health tracker*’s tile
- the app’s specific page if it’s the app tracker’s tile.
The SDK enables an application to synchronize its health data with *S Health* through the SDK’s Health Data package. If the application writes health data and the *S Health*’s related tracker exists, the tracker’s tile is posted to the *S Health*’s main screen without user’s subscription.

For example, there is an application measures the blood pressure and writes the measured data to *S Health* with the Health Data package. If it succeeds to write data to *S Health*, *S Health* posts its blood pressure tracker’s tile with the measured value and time on *S Health* automatically even the tracker has not been subscribed like Figure 7.

![Figure 7: Operation of S Health’s tracker](image)

It means that the app’s doesn’t need to define its tracker for the blood pressure on *S Health* seperately. Otherwise, the app’s defined tracker tile and *S Health*’s one will be posted at the same time with duplicated information. It would be not a good user experience definitely.

**5.3. Tracker Manager**

If *S Health* has a proper tracker for your application’s feature, the TrackerManager helps your app to launch the *S Health*’s available tracker.

Figure 8 shows the flow how to launch the *S Health*’s tracker.

1) The app gets the interesting tracker’s info and checks its availability.

2) If it is, the app launches the tracker. The *S Health*’s tracker is foregrounded.

3) The user selects the back button and your app is foregrounded.
Limitations

There are limitations on launching the *S Health*’s tracker.

- It works on *S Health* 4.8 or above.
- Some trackers of *S Health* may not be available by the local law or device specifications.
6. Defining App’s Tracker

The S Health Service package helps to create a tracker provider application that contains your own tracker. Your defined tracker is located on the S Health > Manage items menu after installing your application and will be shown on the S Health’s main screen upon the user’s subscription.

You need to check the existing S Health tracker before the new tracker definition. Define your application’s tracker and design the tracker tile with the S Health Service package only if there is no proper tracker to express your application’s health data on S Health.

The left screenshot of Figure 9 shows an added tracker of your installed application to the tracker list. If the user turns on the “Sample” tracker on the page and is back to the S Health’s main screen, its tracker tile will be shown as the right screenshot of Figure 9.

The S Health Service package provides three tracker tile templates to express user’s health data with an icon, a title, a button with intent, a data value and a unit, and tile intent. You can choose the proper tile type and make a natural service connection to your application with the tracker tile’s intent and the button intent.

![Figure 9: Sample tracker tile on S Health](image)

6.1. Tracker Definition

A tracker for your application can be defined as covering somewhat of a wide scope like 'My Exercise' that includes running, walking and swimming. Or it can be 'Walking Steps' in a narrower scope.

The tracker is defined with the Json formatted metadata structure in the manifest. It has the following fields shown in Table 4. Please be careful not to define one more tracker.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Tracker ID.</td>
</tr>
<tr>
<td></td>
<td>- Starts with the ‘tracker.’ prefix</td>
</tr>
<tr>
<td></td>
<td>- Needs to be unique in your application</td>
</tr>
<tr>
<td></td>
<td>- No space or additional dot</td>
</tr>
<tr>
<td></td>
<td>- (O) &quot;tracker.myexercise&quot;</td>
</tr>
<tr>
<td></td>
<td>- (X) &quot;mytracker.walk&quot; - caused by the different prefix</td>
</tr>
<tr>
<td></td>
<td>- (X) &quot;tracker.my walking steps&quot; - caused by spaces</td>
</tr>
<tr>
<td></td>
<td>- (X) &quot;tracker.myexercise.walk&quot; - caused by an additional dot</td>
</tr>
<tr>
<td>display-name</td>
<td>String resource ID of the displayed name for the tracker.</td>
</tr>
<tr>
<td></td>
<td>- The value of the string resource ID in 'strings.xml' will be displayed in tracker items of S Health.</td>
</tr>
<tr>
<td></td>
<td>- If your application covers more than one locale, add translated display names for supported locales to string resources.</td>
</tr>
<tr>
<td></td>
<td>- Do not make it too long as considering the width of the device display.</td>
</tr>
<tr>
<td>icon</td>
<td>Drawable resource name for the icon.</td>
</tr>
<tr>
<td></td>
<td>- Size: 36 * 36 dp</td>
</tr>
<tr>
<td>controller</td>
<td>Class name that implements TrackerEventListener</td>
</tr>
</tbody>
</table>

Table 4: Tracker metadata

The following example shows the tracker definition.

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="testtracker_manifest">
    {
      W"tracker" : {
        W"id" : W"tracker.test",
        W"display-name" : W"tracker_display_name",
        W"icon" : W"tracker_icon",
        W"controller" : W"com.samsung.android.app.plugintracker.MyTracker"
      }
    }
  </string>
</resources>
```

The defined tracker is shown in tracker items of S Health as Figure 10 after installing your tracker provider application. All information in the manifest above is used for your tracker in S Health > Manage items. The “tracker_icon” icon resource is shown inside the blue box region in Figure 10. The “tracker_display_name” resource is shown as the tracker’s title after the icon in the same page.
Figure 10: Icon in S Health > Manage items

6.2. Tracker Tile Templates

Your tracker’s tile can be expressed with the defined time templates and it contains following properties:

- Brand icon
- Health data and its unit
- Measured time of data
- Title
- Button with its text and intent

Intent to jump to detailed health information or start a specific action such as measurements can be added on the tracker tile also. Tracker tiles for your tracker are displayed on the main screen of S Health after the user subscribes to the tracker.

A tracker tile visualizes a tracker’s health data within a rectangular box on the main screen of S Health. It displays the latest data collected by its tracker such as calories burned, running distance, or number of activities. A tracker can be expressed as a tracker tile on the S Health’s main screen with various type templates depending on interested information related to health data.

- TRACKER_TILE_TYPE_1
- TRACKER_TILE_TYPE_2
- TRACKER_TILE_TYPE_3

If you need detailed design guidelines, see "Tracker Design Guidelines". You can also find it under the [SamsungDigitalHealthSDK]/SHealthService/Docs/ folder.
**TRACKER_TILE_TYPE_1**

TRACKER_TILE_TYPE_1 is useful when you need to show the tracker tile without content value. It contains the icon, title, and button as Figure 11.

![Figure 11: Tracker tile without content value (type 1)](image)

**TRACKER_TILE_TYPE_2**

If there is data value to show and no need of a button, use TRACKER_TILE_TYPE_2. It contains the icon, content value, unit, title, and date.

![Figure 12: Tracker tile without button (type 2)](image)
**TRACKER_TILE_TYPE_3**

TRACKER_TILE_TYPE_3 contains a button for the specific action additionally as comparing with TRACKER_TILE_TYPE_2.

![Tracker tile with button (type 3)](image)

Figure 13: Tracker tile with button (type 3)

Table 5 shows all properties of the tracker tile.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Icon</td>
<td>Resource of the representative icon</td>
</tr>
<tr>
<td>Content value*</td>
<td>Numeric value of measured data to be shown</td>
</tr>
<tr>
<td>Content unit*</td>
<td>Unit of the content value. It’s optional.</td>
</tr>
<tr>
<td>Title*</td>
<td>Title of the tracker tile</td>
</tr>
<tr>
<td>Date</td>
<td>Time point information for the tracker tile</td>
</tr>
<tr>
<td>Button</td>
<td>The followings can be set for the button.</td>
</tr>
<tr>
<td></td>
<td>- Button text</td>
</tr>
<tr>
<td></td>
<td>- Button intent</td>
</tr>
</tbody>
</table>

The designated color is applied to the button background.*

Table 5: Tracker tile properties

To define the tracker tile containing interesting information, select the proper tile type template and create a tile instance with the following TrackerTile’s constructor.

- TrackerTile(Context context, String trackerId, String tileId, int tileType)

The tracker tile can have intent with setContentIntent(int intentType, Intent intent) that performs an action such as jump to your application when the user selects the tile on the main screen of S health.
The tile’s background color is fixed in the platform but some properties of the tracker tile can be the brand color separately. Properties marked with * in Table 5 indicates ones that can be changed to its brand color. The brand color can be a single color with `setContentColor(int contentColor)`.

### 6.3 Tracker Tile Manager

The created tracker tile instance is managed with `TrackerTileManager`. It enables you to:

- checks the posted your tracker tile on `S Health`
- post or update your tracker tile
- remove your posted tracker tile

The tracker tile manager enables your application to post defined tracker tiles on the main screen or remove them from the main screen.

Figure 14 shows an internal flow how your tracker is posted to `S Health`’s main screen as a tracker tile.

1) Installed application provides a tracker as a plugin on `S Health`.
2) The application’s tracker is added to tracker items of `S Health`.
3) The user checks trackers on the tracker item page of `S Health` by selecting the '+’ button on the main screen.
4) The user subscribes to your tracker.
5) The subscribed tracker’s tile is posted to the `S Health`’s main screen.

**Figure 14:** Tracker tile of your application on `S Health`

### 6.4 Tracker Tile Event Handling

The tracker tile event can be received from the main screen of `S Health` in following cases.
- `onSubscribed()` is called if:
  - the suggestion tile is tapped on the S Health’s main screen.
  - the user subscribes your tracker on S Health.
- `onUnsubscribed()` is called if:
  - the user unsubscribes your tracker on S Health.
- `onPaused()` is called if:
  - the main screen of S Health goes to the background.
- `onTileRequested()` is called if:
  - the main screen of S Health is resumed.
- `onTileRemoved()` is called if:
  - the user removes the posted tracker tile from the main screen of S Health.

And you can add intent to your tracker tile itself or its button to link to the specific activity or service.
7. Sample Application - PluginTracker

The S Health Service package provides a sample application 'PluginTracker' to show how to post your tracker tile and work with S Health. Check the sample application by importing its project with Android Studio (or Eclipse).

Prerequisites

Prerequisites to run PluginTracker are shown below.

- Prepare an Android device that supports Android 4.4 KitKat (API level 19) or above. (Samsung and non-Samsung devices are available both.)
- Install S Health 4.x on the device.
- Turn on the developer mode as 3.3.1.
- Import PluginTracker with Android Studio (or Eclipse)
- Run PluginTracker on the device.

Overview

If S Health and PluginTracker are installed properly, you can:

- find the Sample tracker on the S Health’s tracker list after installing the sample app.
- Launch some S Health’s available tracker.

If you subscribe to the Sample tracker and go back to the S Health’s main screen, its tile will be shown as the middle figure. Its tile is posted with TRACKER_TILE_TYPE_1 at the first time. The sample app’s posted tile will be changed
to the different tile template, TRACKER TILE TYPE_3, that includes a content value, unit, and updated time whenever the tile’s button is tapped.

The posted tracker tile’s ID can be checked on PluginTracker as the Figure 15’s third screenshot and you can remove the posted tracker tile.

**Connection to Health Data Store**

The Samsung Digital Health application needs to use the Health Data package basically and connect to the health data store to handle various exceptions whether the app can work with S Health on the device.

But PluginTracker doesn’t include the Health Data package usage because it focuses to show the tracker usage only.

See the SimpleHealth sample application in the SDK and the Health Data package’s Programming Guide for the health data store connection.

**Source Description**

Table 6 describes source and resource files of PluginTracker.

<table>
<thead>
<tr>
<th>Source / Resource</th>
<th>Description</th>
</tr>
</thead>
</table>
| AndroidManifest.xml | - Registering the tracker tile button’s service intent (MyTrackerService) with the <service> element.  
- Declaring S Health’s PluginService with the <service> element. |
| res/values/ | |
| testtracker.xml | Defining tracker with its metadata. |
| src/com/samsung/android/app/plugintracker/ | PluginTracker’s main activity.  
- TrackerManager  
  • Showing a list for some S Health trackers.  
  • Launching the selected tracker.  
- TrackerTileManager  
  • Showing the posted tile’s ID on the S Health’s main screen.  
  • Containing a button to remove a posted tile. |
| MainActivity.java | Checking the feature availability.  
Implementing the app’s tracker.  
- Implementing TrackerEventListener to receive the following tracker events from S Health:  
  • If the user subscribes or unsubscribes the PluginTracker’s tracker,  
    => onSubscribed(), onUnsubscribed()  
  • If S Health goes to the background,  
    => onPause()  
  • If S Health requests the tracker tile of PluginTracker,  
    => onTileRequested()  
  • If the posted tile is removed from the S Health’s main screen,  
    => onTileRemoved()  
- Posting or updating the tracker tile of PluginTracker depending on the received tracker event. |
<table>
<thead>
<tr>
<th>PluginTrackerProvider.java</th>
<th>Initializing S Health Service.</th>
</tr>
</thead>
<tbody>
<tr>
<td>service/</td>
<td></td>
</tr>
<tr>
<td>MyTrackerService</td>
<td>Defining the tracker tile’s button intent to be performed in the background.</td>
</tr>
</tbody>
</table>

*Table 6: PluginTracker’s source description*

See detailed implementation in the imported `PluginsTracker`’s project through Android Studio (or Eclipse).
Copyright

Copyright © 2016 Samsung Electronics Co. Ltd. All Rights Reserved.
Though every care has been taken to ensure the accuracy of this document, Samsung Electronics Co., Ltd. cannot accept responsibility for any errors or omissions or for any loss occurred to any person, whether legal or natural, from acting, or refraining from action, as a result of the information contained herein. Information in this document is subject to change at any time without obligation to notify any person of such changes.
Samsung Electronics Co. Ltd. may have patents or patent pending applications, trademarks copyrights or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give the recipient or reader any license to these patents, trademarks copyrights or other intellectual property rights.
No part of this document may be communicated, distributed, reproduced or transmitted in any form or by any means, electronic or mechanical or otherwise, for any purpose, without the prior written permission of Samsung Electronics Co. Ltd.
The document is subject to revision without further notice.
All brand names and product names mentioned in this document are trademarks or registered trademarks of their respective owners.

For more information, please visit http://developer.samsung.com/