Creating a Healthcare Device Application

Published 2014-10-27 | (Compatible with SDK 3.5,4.5,5.0,5.1 and 2012,2013,2014 models)

> This tutorial demonstrates the use of External Interworking API healthcaredevice with a tutorial healthcare application. A healthcare application connects to a physical healthcare device, such as a weight scale or a blood pressure monitor, and gets the health data from the device through the TV platform.

Contents

Development Environment

Use Samsung Smart TV SDK to create the application. You can also use the emulator provided with the SDK to debug and test the application before uploading it in your TV. Later, you can run the application on a TV; see Testing Your Application on a TV. Note that applications may perform better on the TV than on the emulator.

You can find general instructions for creating applications in Implementing Your Application Code.

Prerequisites

To create a healthcare device application that runs on a TV screen, you need:

Samsung TV connected to the Internet
SDK or a text editor for creating HTML, JavaScript and CSS files (using Samsung Smart TV SDK is recommended)

You do not need a physical health device to use this tutorial, because the tutorial application emulates a healthcare device. At the moment, the Healthcare class supports two types of health devices: weight scales and blood pressure monitors.

This application does not contain any connection setting between the healthcare device and a TV. The connection is supported by the setting menu of the TV. Therefore, the developer does not need to implement the connection setting.

Standards

The HealthcareDevice class was created with reference to the following health device communication standards:

IEEE 11073-20601
IEEE 11073-10407
IEEE 11073-10415

For blood pressure monitor, the following standard was used:

IEEE 11073-10407

For weight scale, the following standard was used:

IEEE 11073-10415

Possible Extensions

HealthcareDevice class can be extended to support other health devices like pedometers. At the moment, only OCI_HC_ATTR_TIME_STAMP_ABS is supported as the time stamp. Other time types will be supported later.

Some health devices can send vendor specific extended data to the application. It is out of scope of specifications. For such cases, currently OCI_HC_UNKNOWN_TYPE is sent as their data type and OCI_HC_UNKNOWN_UNIT as their data unit. Support to bypass the vendor specific data type and unit to the application will be introduced soon. It will help the vendors to make their own applications for their health devices and use their specific data in it.

Source Files

> Note
>
> The files needed for the sample application are **here**.

The tutorial Healthcare Device Application directory is named Tutorial_HealthcareDevice. The source files for the application are located in this folder. They are explained in the table:

Directory	Description
image	image files for application:
js	Contains the JavaScript file test.js which does the following: initializes codes registers events key handling displays the results parses the healthcare device data of callback function from emulated healthcare devices

A Healthcare Device Application

This tutorial task consists of the following parts:

Running the Healthcare Device Application

This section demonstrates how to operate the application on the emulator.

Starting the Healthcare Device Application

To start the healthcare device application,

1. On the down menu, select Open App.
2. On the popup menu, select Tutorial_HealthcareDevice (see Figure 1).

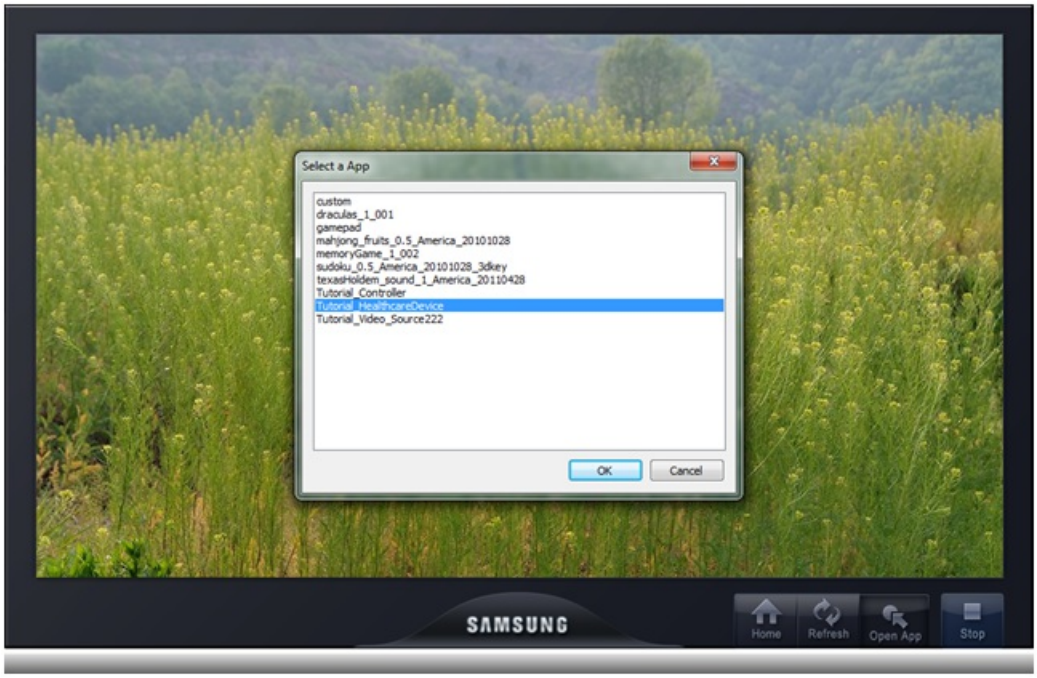

Figure 1: Select Tutorial_HealthcareDevice

If the screen shown in Figure 2 is displayed, the application has started successfully.

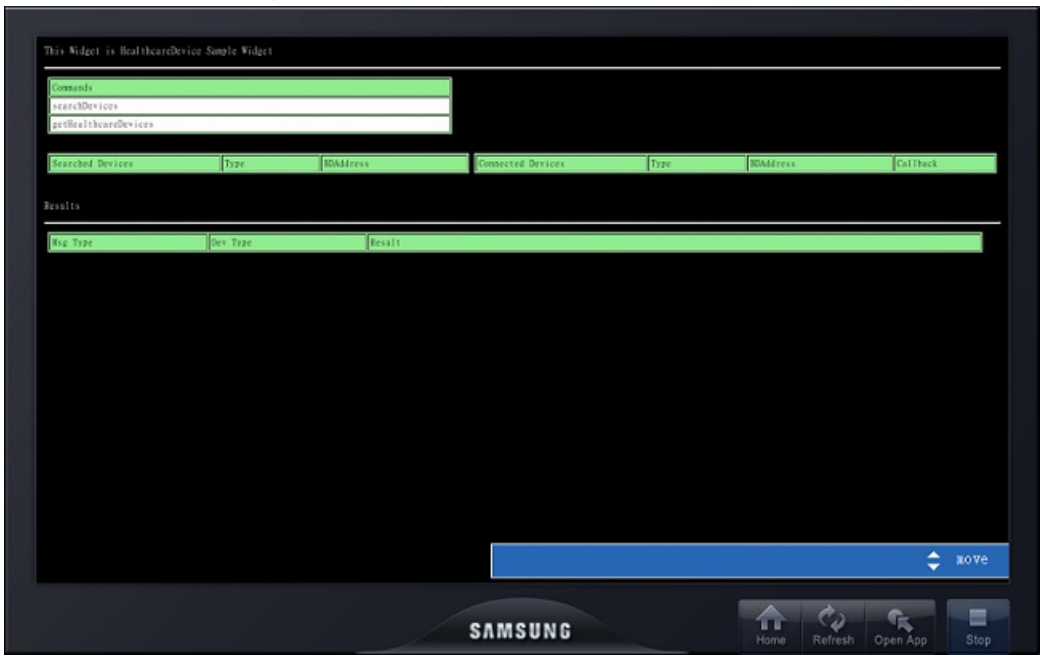

Figure 2: Healthcare application tutorial opening screen

Get Searched Healthcare Devices

To get a list of the healthcare devices that can be searched,

Because the healthcare devices normally use bluetooth as a communication method, search process is needed. In the TV SDK, you can get the devices easily, because the SDK emulates several bluetooth HDP devices.

1. Press the **down** key of the remote control to select the **searchDevices** menu (see Figure 3).

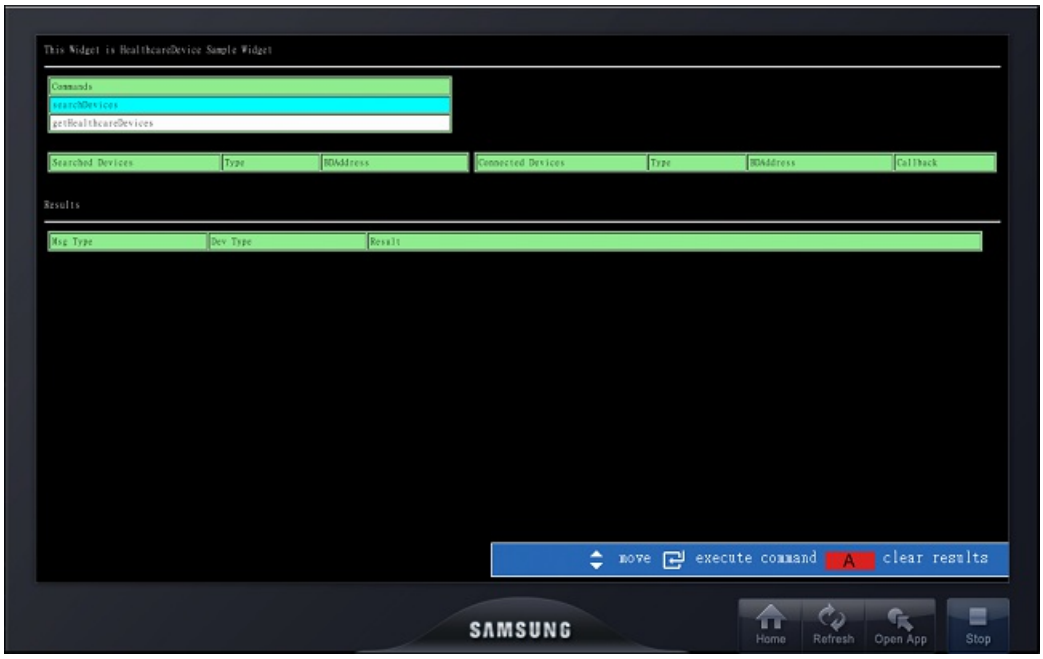

Figure 3: searchDevices menu selected

2. Press the **Enter** key of the remote control to get the list of searched devices from TV (see figure 4).

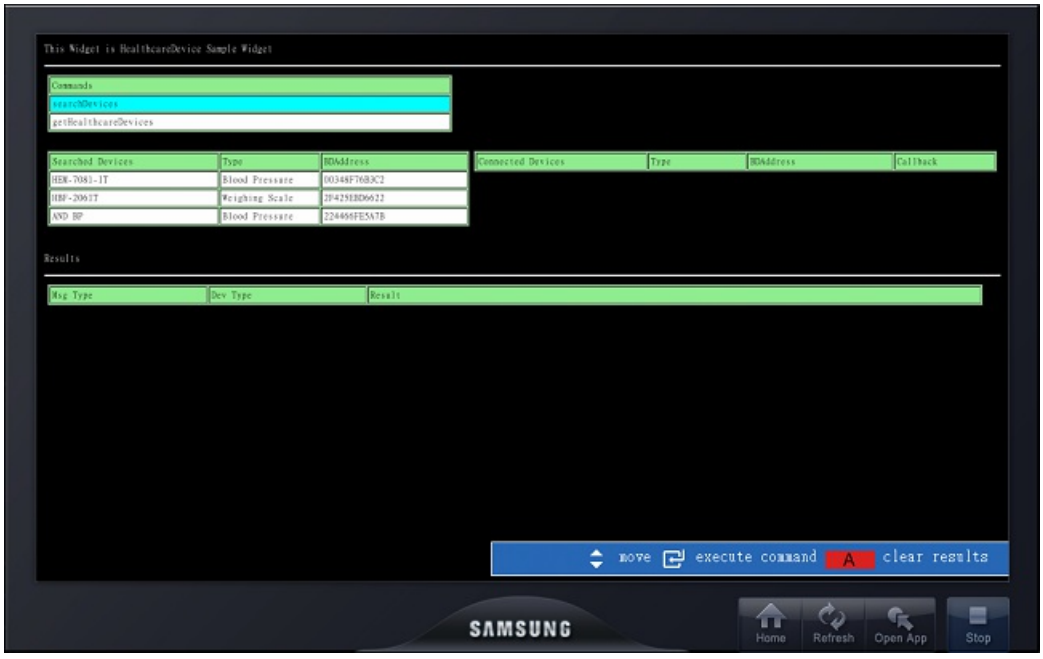

Figure 4: List of searched devices

Connect Healthcare Device

Before getting a data from healthcare devices, you have to connect it,

1. Press the **down** key of the remote control to select the **searched healthcare device** which is wanted to connect (see Figure 5).

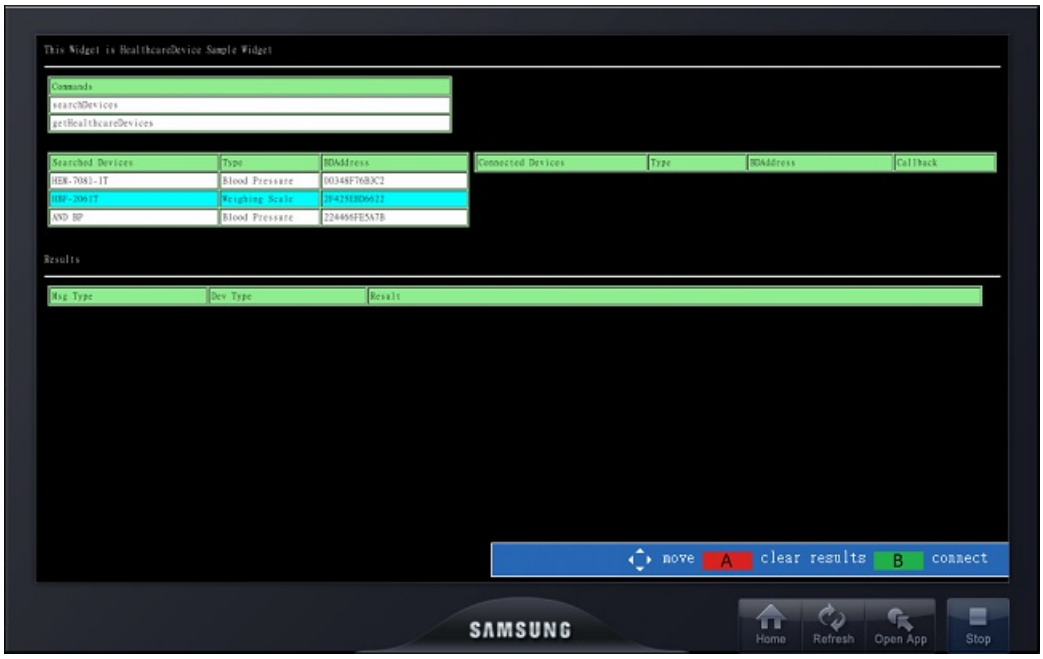

Figure 5: searched healthcare device selected

2. Press the **Green** key of the remote control several times to connect searched device (see figure 6).

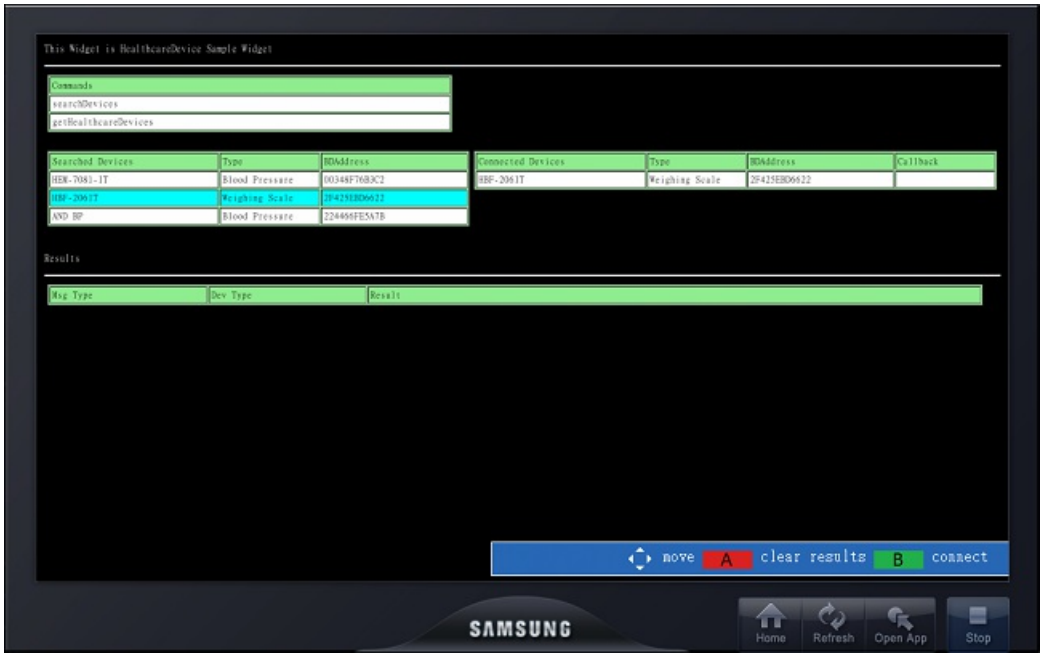

Figure 6: List of connected devices

Get Data from a Device

Because the SDK emulates healthcare devices, you can see healthcare device information and get device data, even though there are no physical health devices connected to the computer. Currently, the SDK emulates two types of healthcare devices: blood pressure monitors and weight scales. There are two manufacturers of these devices: A&D and OMRON. The emulated devices can be chosen freely. Register a callback function to get the data from a healthcare device as follows:

1. Press the **right** key of the remote control to navigate connected devices list.
2. And navigate the list of connected devices by pressing **up** and **down** keys of the remote control.
3. Press **Enter** on a device to register a callback. The data is then displayed automatically.

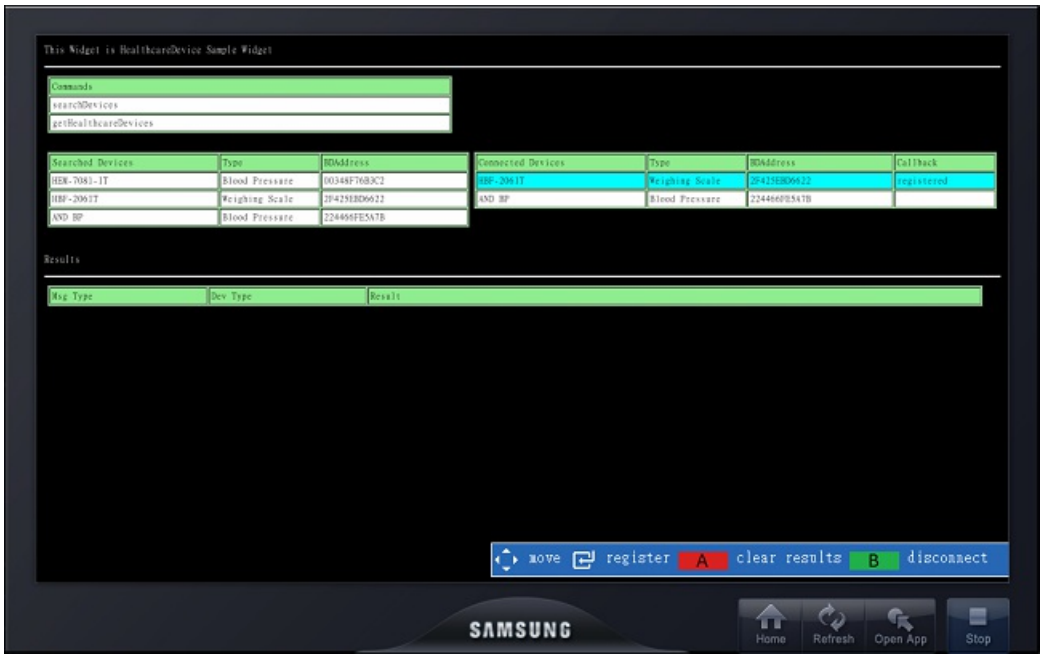

Figure 7: Callback registered

> Note
>
> In this tutorial, the health data, such as blood pressure, is generated automatically by the SDK. In a real application, you need to measure your health status with a healthcare device. For example, you can measure your blood pressure with the blood pressure monitor. After measuring, you have to send the data from the healthcare device to the TV.

In Figure 8, the application shows the measurement result on the display. In the result, Dev Type is the healthcare device type and Result is the measurement data.

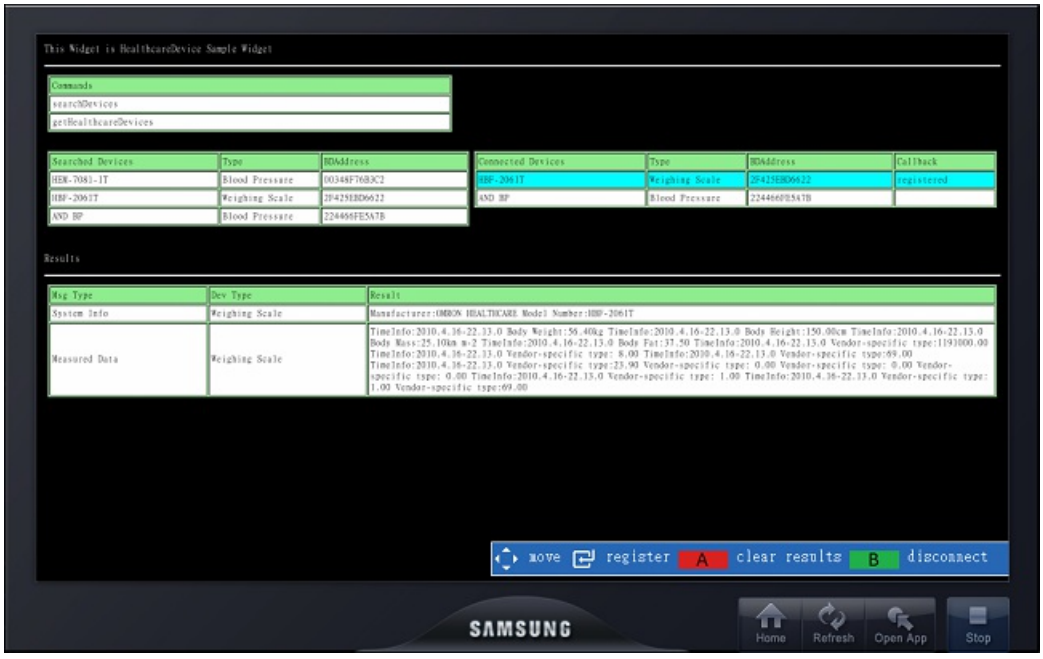

Figure 8: Health data displayed

Example Programs

searchDevices / getHealthcareDevices Code

Function	Description
searchDevices	request to searched healthcare device instances. This function is only for used to search wireless(usually bluetooth) devices.
stopSearchDevices	request to stop searching. It will not work on SDK.

Function	Description
getHealthcareDevices	Gets a list of healthcare devices that are connected to the TV. When this function is called first time, it will initialize all internal jobs.

```
MainFunc.js
Main.keyDown = function () {
    var keyCode = event.keyCode;

    .....
    case gTVKey.KEY_ENTER:
        ...
        switch (this.currentPosition) {
            case 0:
                Main.deleteDevicesOnTable("TableSearched");
                healthcaredevice.searchDevices();
                break;
            case 1:
                healthcaredevice.getHealthcareDevices(Main.onHealthcareDevicesObtained);
                break;
        }
        Main.deleteResultsOnTable();
        break;
    ...

Main.onHealthcareDevicesObtained = function (healthcaredevices) {
    ...
    // check connected devices
    if (healthcaredevices.length > 0) {
        alert("found " + healthcaredevices.length + " healthcaredevices");
        ...
    } else {
        alert("no healthcaredevices found");
    }
}
```

Registering a Callback Function for a Device Connection Event

To receive a device connection event, register your callback function by window.webapis.healthcaredevice.registerManagerCallback() function. Then, your callback function is called at connecting or disconnecting to microphone. The callback function can get window.webapis.healthcaredevice.ManagerEvent class object including event type, healthcaredevice's name, healthcaredevice's type as an input parameter.

Function	Description
registerManagerCallback	Register callback function to get the event about CONNECT/DISCONNECT/SEARCHED

```
Main.onLoad = function () {
    gWidgetAPI = new Common.API.Widget();    //    Create Common module
    gTVKey = new Common.API.TVKeyValue();
    gWidgetAPI.sendReadyEvent();        //    Send ready message to Application Manager

    healthcaredevice.registerManagerCallback(Main.onManagerCallback);
    ...
}

Main.onManagerCallback = function (ManagerEvent) {
    switch (ManagerEvent.eventType) {
        case healthcaredevice.MGR_EVENT_DEV_SEARCHED:
            Main.addDeviceOnTable("TableSearched", ManagerEvent.name, ManagerEvent.deviceType, ManagerEvent.U
ID);
            break;
        case healthcaredevice.MGR_EVENT_DEV_CONNECT:
            healthcaredevice.getHealthcareDevices(Main.onHealthcareDevicesObtained);
            break;
        case healthcaredevice.MGR_EVENT_DEV_DISCONNECT:
            healthcaredevice.getHealthcareDevices(Main.onHealthcareDevicesObtained);
            break;
        case healthcaredevice.MGR_EVENT_DEV_SEARCH_FINISHED:
            alert("Search HealthcareDevice Finished");
            break;
    }
}
```

Connect / Disconnect searched device

Before request to receiving data from HealthcareDevice, connection procedure needed.

Function	Description
connectDevice	request to connect healthcare device.
disconnectDevice	request to disconnect healthcare device.

```
Main.keyDown = function () {
    var table;
    ...
    case this.statusSearched:
        table = document.getElementById("TableSearched");
        healthcaredevice.connectDevice(table.rows[this.currentPosition+1].cells[2].innerHTML);
        break;
    case this.statusConnected:
        table = document.getElementById("TableConnected");
        healthcaredevice.disconnectDevice(healthcareDeviceArray[this.currentPosition].getUniqueID());
        break;
    ...
}
```

Registering a callback function to get a data from healthcare device

Function	Description
registerDeviceCallback	request to retrieve a data from each healthcare device which is already connected on TV

```
healthcareDeviceArray[this.currentPosition].registerDeviceCallback(Main.onDeviceCallback);
```

Analyze the Data from Device And Convert the Result into String Code

Function	Description
parseHealthcareDeviceInfo	Parses the data from the healthcare device.
toStringDeviceType	Converts the device type info into a string.
toStringUnit	Converts the unit info into a string.
toStringMeasuredInfo	Converts the measured info into a string.
toStringSystemInfo	Converts system info into a string.

test.js

```
Main.parseHealthcareDeviceInfo = function (healthcareInfo) {
    if (healthcareInfo!=null) {
        // Main.deleteResultsOnTable();

        var table = document.getElementById("Table2"),
            length = table.rows.length,
            row = table.insertRow(length),
            td1 = row.insertCell(0),
            td2 = row.insertCell(1),
            td3 = row.insertCell(2),
            iCount = 0;

        row.bgColor = "#FFFFFF";
        switch (healthcareInfo.infoType) {
            case webapis.healthcaredevice.DEV_INFO_MEASURE_DATA:
                td1.innerHTML = "Measured Data";
                break;
            case webapis.healthcaredevice.DEV_INFO_SYSTEM_INFO:
                td1.innerHTML = "System Info";
                break;
        }
        td2.innerHTML = "";
        for(iCount=0;iCount<healthcareInfo.deviceType.length;iCount++) {
            td2.innerHTML += Main.toStringDeviceType(healthcareInfo.deviceType[iCount]);
        }
        td3.innerHTML = "";
        switch (healthcareInfo.infoType) {
            case webapis.healthcaredevice.DEV_INFO_MEASURE_DATA:
                td3.innerHTML = Main.toStringMeasuredInfo(healthcareInfo.data);
                break;
            case webapis.healthcaredevice.DEV_INFO_SYSTEM_INFO:
                td3.innerHTML += Main.toStringSystemInfo(healthcareInfo.data);
                break;
        }
    }

}
```