In-App Purchase Programming Guide

A method using AppControl for Tizen

Table of Contents

Abstract

Introduction

Selling items from within your application, you can get large revenue. You can implement In-Application Purchases in your application using the AppControl mechanism for Tizen Native and Web applications.

You can use the IAP in any of the following scenarios:

- An application that enables additional features
- Any application that allows the user to remove adverts
- An audio-book application that allows the user to purchase and download new books
- A game which offers new levels to play
- A RPG game that allows to buy virtual items
- And many other features…

What is the In-App Purchase AppControl mechanism?

If you want to sell items inside your applications such as in-game coins, levels and others, you may need to set servers up to communicate with billing server and user authentications server, and many other infra servers. In-App Purchase will simply handle all those things mentioned above and you don't need to worry about it.

In order to using In-App Purchase, you need to prepare your applications for communicate with it. For example, with in-app purchase of items, your applications just find the proper AppControl, starts the purchase method and receive the result of the purchase in the method of the listener method. IAP will show dialogs (in case of errors), communicate with Tizen Stores server and invoke the listener methods to return the result of the request.

This document explains the process of using the IAP from A to Z. Please read this document thoroughly before you use it.

Supported Item Types

IAP supports the item types described below.

Type	Description
Consumable	If you purchase an item of this type and use it, it is consumed. These items can be repurchased. ※ Example: Consumable items such as bullets in games.
Non-Consumable	Once purchased, you can use an item of this type permanently. These items cannot be repurchases. ※ Example: Non-Consumable items, including books, that do not need to repurchased.

Subscription (Non-renewing)	Once a certain period has passed after an item purchase, these items can be repurchased. These items are not automatically repurchased after a certain period. ※ Example: Items that can be repurchased after a certain period such as monthly magazines with expiration dates.

Terms and abbreviations

IAP	In-App Purchase
MCC	This code (3 digits) identifying mobile country code.
MNC	This code (chars/digits) identifying carrier in current country.
AppControl	It is a standard mechanism in Tizen for using specific operations exported by other applications.
Operation ID	Defines the behavior of the AppControl.
Application ID	Used to identify each application control suppliers.
Item Group ID	Collection of items identified by single ID (Item list).
Item	It is a single piece of content.

Applying IAP to your application.

It is not difficult to apply the IAP to your application, because it is very simple and lightweight. How to apply are as follow:

① Register your items to Tizen Store Seller Office. (http://seller.tizenstore.com)
② Program your application to work with IAP.
③ Test and upload your application.

1. Registering your items to Tizen Store Seller Office

Before you set up and program your application for IAP, you need to register item group and items to Tizen Store Seller Office. The item group is a collection of items and it will be connected to your application in Tizen Store Seller Office. For example, if there are 10 kinds of special items you need for your game, you need to create one item group for your game and 10 individual items. Registering is really simple.

Connect to Tizen Store Seller Office.

Run a web browser and connect to Tizen Store Seller Office. The address of Tizen Store Seller Office is http://seller.tizenstore.com . Log in to Tizen Store Seller Office. If you don't have any account for Tizen Store Seller Office, you can make your own. After you have logged in to the Tizen Store, if you want to sell applications under your personal name, you should register as a private seller. And , if you want to sell applications under your company name register as a corporate seller.

Register a New Item Group

Follow steps are explain how to register a new item group on Seller Office.

① Click 'Applications > Item' to navigate to the selected menu option.
② Click [Add Item Group].
③ Fill in the fields for the Item Group Title and Description, and click [OK] to register your item group.
④ When the item group is registered, you can click [Edit] to modify its information.
⑤ Select a checkbox and click [Delete] to delete an item group. When you delete an item group, all the items included in that group will also be deleted.
⑥ You can copy registered item groups from the item group list. When you copy an item group, all the items included in that group will also be copied.
⑦ An Item group can be modified, deleted, or copied when the application has the following statuses.

Add a new item (temporary status)

From the item list, select the name of a registered item group to navigate to its information page. After an item group is created, a list of temporary tabs will appear on the information screen when you first access it.

① Click [Add Item] in the temporary tab on the view page to enter basic information, such as Title, Price, and Description, in the Add Item popup window. Then click [OK] to add items. See the [Application Registration Guide] for details on price setting.
② You can batch register items by clicking [Item Bulk Upload] to upload an Excel file.

③ When the item has been added, you can click its title in the item list to view the added content.

1) Edit an Item (temporary status)

You can add, modify, or delete an item if there is an application listed in the item group, or if registered or revised applications in the item group are listed under Pre-Certification.

① You can click the title of the item on the temporary tab list and check or edit the item title, price, description, and image in the View popup window.
② If all the applications containing the item are listed under Pre-Certification, select the checkbox and click [Delete] to delete the item from all of them. However, if the statuses of items mapped for applications are for sale, it cannot be deleted.
③ All changes will be applied when all the mapped applications are validated. The [On Standby] button is displayed until validation is complete.
④ Item statuses for different types of application status are as follows:

2) Item sales (sales status)

① Items are registered and modified and you can view them in sales.
② If all the mapped applications that are being registered or revised in the item group are listed under Pre-Certification, click the [Modify Item(s)] button to create a temporary tab that will allow you to modify the applications.

2. Setting your application project up to use IAP

There is nothing additional to be done in the project to make your application work with IAP. It uses an AppControl mechanism, and as such requires no updates to the application project.

Working with IAP means simply getting the proper AppControl.

You can find the AppControl tutorials in Help section of Tizen IDE. Or directly you can refer to the link:

- Application Controls

3. Programming your application to work with IAP

Programming with IAP is very similar to programming with Native Tizen API and other AppControls supported in the system. Before you implement your code you should edit the “manifest” file first.

Section 3.1 explains more about “manifest” file.

Note that the IAP allows for two modes of operation (key name “_mode”):

- Normal mode or commercial mode: To be used with the finished application, released on the market. Proper data will be obtained and returned from the IAP server.
- Developer mode: This can be used while development of the application for testing purpose.

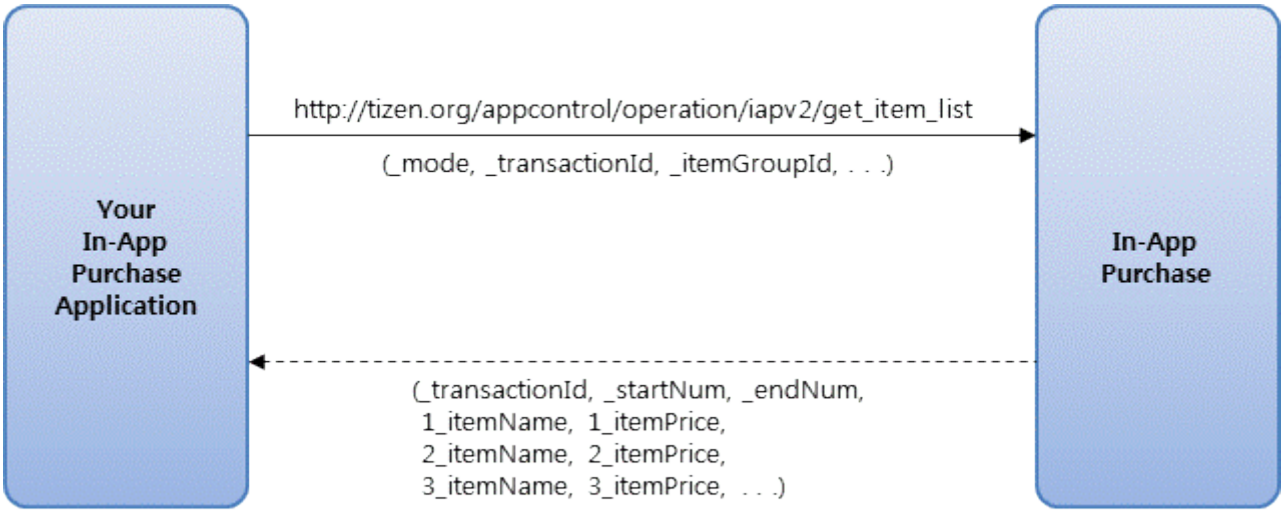
Payment always succeed.

Following table describe the Interfaces list

Application ID	Operation ID	Description
`org.tizen.inapppurchase.iapclient`	http://tizen.org/appcontrol/operation/iapv2/purchase	The operation makes a purchase of Item. It shows a purchase screen for an item. During this step a purchase screen will be displayed. And the user need to provide user's details (e-mail, password) to make a purchase. The output value indicates the result of purchase(success or not), and it is used to verify the purchase.
`org.tizen.inapppurchase.iapservice`	http://tizen.org/appcontrol/operation/iapv2/get_item_list	The operation returns a list of item available for purchase. The output data values are used to send a list of items available for purchase.
	http://tizen.org/appcontrol/operation/iapv2/get_purchased_item_list	The operation returns a list of already purchased items.
	http://tizen.org/appcontrol/operation/iapv2/get_country_list	The operation returns a list of counties' MCC and MNC codes to be used in developer mode during testing in-app purchases.

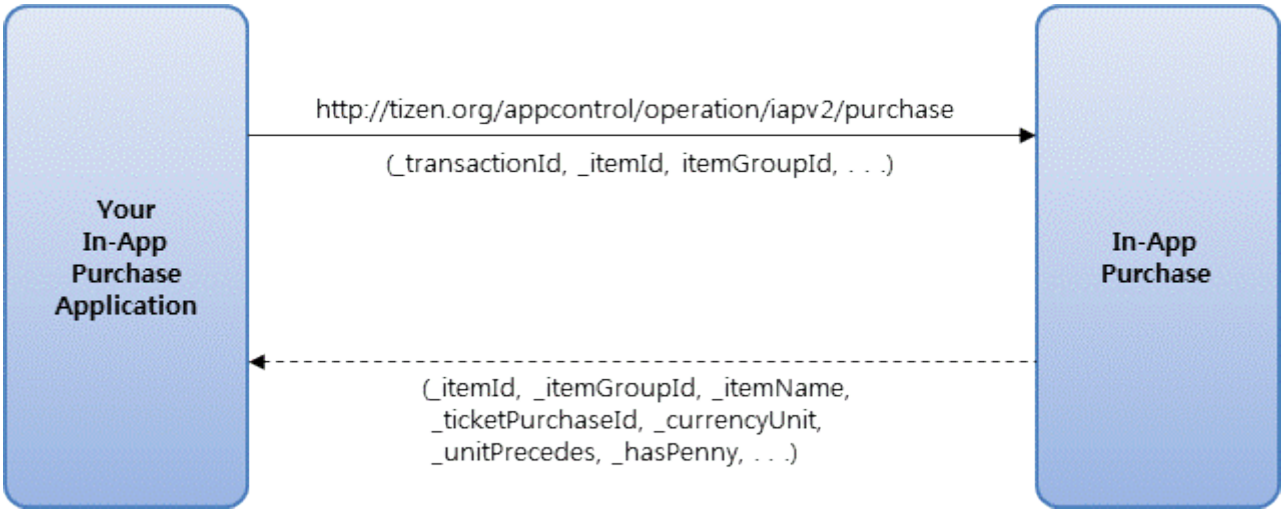
Purchase Item flow:

1. Get Item list: Use the `org.tizen.inapppurchase.iapservice` Application ID to retrieve a list items from Tizen Store IAP Server for a given groupId:



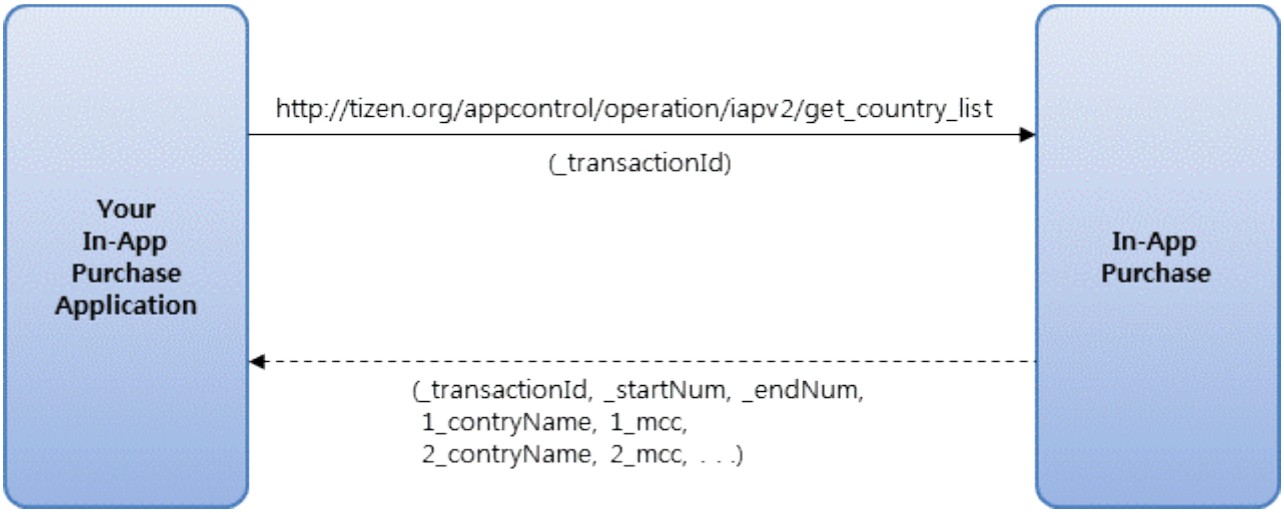
2. Display a list of items in Your In-App Purchase Application.

3. Purchase the item: Use the `org.tizen.inapppurchase.iapclient` Application ID to make a purchase of particular item.



Get a list of country

In developer mode, you are able to test In-App Purchase with different country servers. To get a list of available servers you can use `http://tizen.org/appcontrol/operation/iapv2/get_country_list` operation:



3.1. Add permissions to manifest file.

IAP uses AppControl interface to handle purchases. To add required permissions to your native application project, open tizen-manifest.xml file and add following permission:

- http://tizen.org/privilege/appmanager.launch

Below is the content of the manifest file of native application which contains required permissions.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<manifest xmlns="http://tizen.org/ns/packages" package="org.tizen.iapsample" version="1.0.0">
        . . .
        <privileges>
                <privilege>http://tizen.org/privilege/appmanager.launch</privilege>
                . . .
        </privileges>
        . . .
</manifest>
```

On the other hand, to add required permission to your web application project, open config.xml file and add following permission:

- http://www.tizen.org/privilege/application.launch

Below is the content of the config file of web application which contains required permissions.

```
<?xml version="1.0" encoding="UTF-8"?>
< widget xmlns="http://www.w3.org/ns/widgets" xmlns:tizen="http://tizen.org/ns/widgets" id=   . . .>
        . . .
        <tizen:privilege name = “http://tizen.org/privilege/application.launch”/>
        . . .
</widget>
```

3.2 IAP Service Control (get item list, get purchased item list, get country list)

The IAP Service instance allows you to get a list of items available for purchase and to get a list of already purchased item. It also allows you to get a list of countries available for testing in developer mode

Application ID

This application control can be accessed using aliased application ID of “`org.tizen.inapppurchase.iapservice`”.

Operation ID

This application supports the `http://tizen.org/appcotnrol/operation/iapv2/get_item_list` and `http://tizen.org/appcontrol/operation/iapv2/get_purchased_item_list` operations. It also supports `http://tizen.org/appcontrol/operation/iapv2/get_country_list` operation.

Get Item List operation

This operation returns a list of items available for purchase.

Input data

The following table show the (key, value) pairs required in the input extra data for the `http://tizen.org/appcontrol/operation/iapv2/get_item_list` operation:

Key	Value	Description

Key	Value	Description
_mode	0 | 1	Mode type. This information is optional. The default value is 0. 0 is normal (commercial) mode. This mode need to be used in application submitted to Tizen Store. 1 is developer mode (always success). This mode can be used while development of the application for testing purpose. Payment always succeed.
_transactionId	Transaction ID number, such as 1, 2, …	Transaction ID. This information is mandatory. This is used to track a transaction between requests.
_startNumber	Index of the first item	Index of first item on the list This information is mandatory Start downloading item from this number
_endNumber	Index of the last item	Index of last item on the list This information is mandatory. Stop downloading items up to this number.
_itemGroupId	ID of Group, such as 100000001455	Group ID This information is mandatory. Group ID is associated with a particular Group of items in the Tizen Store Seller Office site. You need to register your Group Id on the Tizen Store Seller Office website first.

Key	Value	Description
_languageCd	Language Code, such as eng, rus	Language Code This information is optional. According to ISO 639-2, thus the name three character code. Language Code is associated with display language of item details in the Tizen Store Seller Office site. Output parameters (itemName, itemDescription, reserved1, reserved2) are changed according to this Language code.
_itemTypeCd	Item Type, such as 00 | 01 | 02 | 10	Item type code This information is optional. String representation of values: 00: Non-consumable 01: Consumable 02: Subscription (Non-renewing) 10: All
_mcc	MCC Code, such as 250	MCC Code This information is optional. It can be used only in developer mode. You can receive a list of available MCC codes through get country list operation
_mnc	MNC Code, such as 01	MNC Code This information is optional. It can be used only in developer mode. MNC Code is Mobile Network Code.

Example code for get item list

```
app_control_h app_control;
int rt = app_control_create(&app_control);

if (rt == APP_CONTROL_ERROR_NONE){
        app_control_set_app_id(app_control, " org.tizen.inapppurchase.iapservice");
        app_control_set_operation(app_control, "http://tizen.org/appcontrol/operation/iapv2/get_item_list");
        app_control_add_extra_data(app_control, "_mode", "0");
        app_control_add_extra_data(app_control, "_transactionId", "123");
        app_control_add_extra_data(app_control, "_startNumber", "1");
        app_control_add_extra_data(app_control, "_endNumber", "10");
        app_control_add_extra_data(app_control, "_itemGroupId", "100000000012");
```

```
        app_control_add_extra_data(app_control, "_languageCd", "ENG");
        app_control_add_extra_data(app_control, "_itemTypeCd", "00");

        rt = app_control_send_launch_request(app_control, get_item_list_cb, NULL);
}

if (app_control != NULL){
        app_control_destroy(app_control);
}
```

Output Data

The results of the operation are returned in the app control callback.

The following table shows the output data for the `http://tizen.org/appcotnrol/operation/iapv2/get_item_list` operation:

Key	Value	Description
_method	OnItemInformationListReceived	Method to be called as a purchase request result. This value will be OnItemInformationListReceived
_result	<Result code>	Result code number. Result codes are described in the end of this section.
_resultDescription	<Result code/Function ID number>	“Result code /Function ID” when “_result” value is not “0” (success). Result codes are described in the end of this section
_transactionId	<TransactionID>	Transaction ID number This is the same as the transaction ID that is used to request .
_startNumber	<start number>	Start Number Index of the first item on the list
_endNumber	<end number>	End Number Index of the last item on the list
_totalCount	<total count>	Total items countNumber of items based on the startNumber and endNumber

Key	Value	Description
_itemTotalCount	<item total count>	Total counts of registered items in the group ID.

We also have _totalCount number of items in an output data, each key is a PREFIX (list item index value) plus a key (i.e. “12_itemId”) as below:

Key	Value	Description
PREFIX_itemId	<ItemID>	Item ID number This is the same as a Item ID that is used to request.
PREFIX_itemGroupId	<ItemGroup ID>	Item group ID This is a collection of items and it will be linked to your application in Tizen Store Seller site.
PREFIX_itemName	<Item Name>	Item Name This is a name provided during item registration on the Tizen Store Seller Office.
PREFIX_currencyUnit	Currency unit, such as: $,Won, Pound	Currency Unit Device user currency unit.
PREFIX_unitPrecedes	0 | 1	Unit Precedes String representation of values: 0: Tall (2.99 TL) 1: Front ($ 2.99)
PREFIX_hasPenny	0 | 1	Has Penny Informations if currency unit has penny: representation 0 : no 1 : yes
PREFIX_itemPrice	<Item price>	Item price This is a price of item in local currency.

Key	Value	Description
PREFIX_itemDownloadUrl	<URL>	Item download URL This is a URL provided during item registration on the Tizen Store Seller Office.
PREFIX_itemImageUrl	<URL>	Item image URL This is a URL provided during item registration on the Tizen Store Seller Office
PREFIX_itemDescription	<Item description>	Item description A description provided during item registration.
PREFIX_reserved1	<Reserved Field 1>	Reserved Field 1
PREFIX_reserved2	<Reserved Field 2>	Reserved Field 2
PREFIX_itemTypeCd	00 | 01 | 02	Item type code String representation of values: 00: Non-consumable 01: Consumable 02: Subscription (Non-renewing)
PREFIX_itemSubsBillDurationCd	00 | 01 | 02 | 03	Item subsbillduration code ‘_itemTypeCd’ Value is ‘03’, the only. If the product type is subscription, the available four units for the validity period of the product are YEAR, MONTH, WEEK, and DAY. The units should be in capital code. String representation of values: 00: Year 01: Month 02: Week 03: Day

Key	Value	Description
PREFIX_subscriptionDurationMultiplier	Subscription duration multiplier	If the _itemTypeCdis subscription (-02), this is the item duration. Combined withPREFIX_itemSubsBillDurationCd, it is means 1MONTH.
PREFIX_timeStamp	<Time stamp>	Time stamp Based on GMT +0, Server time. (yyyyMMddHHmmss)

Example code for retrieving a list of items

```
void get_item_list_cb(app_control_h request, app_control_h reply, app_control_result_e result, void *user_data){

	char* rt_method = NULL;
	char* rt_result = NULL;
	char* rt_resultDescription = NULL;
	char* rt_transactionId = NULL;
	char* rt_startNumber = NULL;
	char* rt_endNumber = NULL;
	char* rt_totalCount = NULL;
	char* rt_itemTotalCount = NULL;
	char* rt_itemId = NULL;
	char* rt_itemGroupId = NULL;
	char* rt_itemName = NULL;
	char* rt_currencyUnit = NULL;
	char* rt_unitPrecedes = NULL;
	char* rt_hasPenny = NULL;
	char* rt_itemPrice = NULL;
	char* rt_itemDownloadUrl = NULL;
	char* rt_itemImageUrl = NULL;
	char* rt_itemDescription = NULL;
	char* rt_reserved1 = NULL;
	char* rt_reserved2 = NULL;
	char* rt_itemTypeCd = NULL;
	char* rt_itemSubsBillDurationCd = NULL;
	char* rt_subscriptionDurationMultiplier = NULL;
	char* rt_timeStamp = NULL;

	if (result == APP_CONTROL_RESULT_SUCCEEDED){

		rt = app_control_get_extra_data(reply, "_method", &rt_method);
		rt = app_control_get_extra_data(reply, "_result", &rt_result);

		//succeed
		if (!strcmp("0", rt_result)){
			rt = app_control_get_extra_data(reply, "_resultDescription", &rt_resultDescription);
			rt = app_control_get_extra_data(reply, "_transactionId", &rt_transactionId);
			rt = app_control_get_extra_data(reply, "_startNumber", &rt_startNumber);
			rt = app_control_get_extra_data(reply, "_endNumber", &rt_endNumber);
			rt = app_control_get_extra_data(reply, "_totalCount", &rt_totalCount);
			rt = app_control_get_extra_data(reply, "_itemTotalCount", &rt_itemTotalCount);

			int start = atoi(rt_startNumber);
```

```
                        int end = atoi(rt_endNumber);

                        char keyId[100] = {0, };
                        for(; start <= end; start++){
                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemId");
                                rt = app_control_get_extra_data(reply, key_id, &rt_itemId);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemGroupId);");
                                rt = app_control_get_extra_data(reply, key_id, &rt_itemGroupId);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemName");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemName);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_currencyUnit");
                                rt = app_control_get_extra_data(reply, keyId, &rt_currencyUnit);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_unitPrecedes");
                                rt = app_control_get_extra_data(reply, keyId, &rt_unitPrecedes);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_hasPenny");
                                rt = app_control_get_extra_data(reply, keyId, &rt_hasPenny);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemPrice");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemPrice);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemDownloadUrl");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemDownloadUrl);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemImageUrl");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemImageUrl);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemDescription");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemDescription);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_reserved1");
                                rt = app_control_get_extra_data(reply, keyId, &rt_reserved1);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_reserved2");
                                rt = app_control_get_extra_data(reply, keyId, &rt_reserved2);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemTypeCd");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemTypeCd);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemSubsBillDurationCd");
                                rt = app_control_get_extra_data(reply, keyId, &rt_itemSubsBillDurationCd);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_subscriptionDurationMultiplier");
                                rt = app_control_get_extra_data(reply, keyId, &rt_subscriptionDurationMultiplier);

                                snprintf(key_id, sizeof(keyId), "%d%s", start, "_timeStamp");
                                rt = app_control_get_extra_data(reply, keyId, &rt_timeStamp);
                        }
                }
        }
}
```

GetPurchasedItemList operation

This operation returns a list of already purchased items.

Input data

The following table show the (key, value) pairs required in the input extra data for the `http://tizen.org/appcontrol/operation/iapv2/get_purchased_item_list` operation:

Key	Value	Description
_mode	0 | 1	Mode type. This information is optional. The default value is 0. 0 is normal (commercial) mode. This mode need to be used in application submitted to Tizen Store. 1 is developer mode (always success). This mode can be used while development of the application for testing purpose. Payment always succeed.
_transactionId	Transaction ID number, such as 1, 2, …	Transaction ID. This information is mandatory. This is used to track a transaction between requests.
_startNumber	Index of the first item	Index of first item on the list This information is mandatory Start downloading item from this number
_endNumber	Index of the last item	Index of last item on the list This information is mandatory. Stop downloading items up to this number.
_startDate	Start date, such as 20131031	The start date of the requested inbox list. This information is optional.
_endDate	End date, such as 20131031	The end date of the requested inbox list. This information is optional.

Key	Value	Description
_itemGroupId	ID of Group, such as 100000002501	Group ID This information is mandatory. Group ID is associated with the particular Group of items in the Tizen Store Seller Office site. You need to register your Group Id on the Tizen Store Seller Office website first.
_languageCd	Language Code, such as eng, rus	Language code This information is optional. According to ISO 639-2, thus the name three character code. Language Code is associated with display language of item details in the Tizen Store Seller Office site. Output parameters(itemName , itemDescription, reserved1, reserved2) are changed according to this Language code.
_mcc	MCC Code, such as 250	MCC Code This information is optional. It can be used only in developer mode. You can receive a list of available MCC codes through get country list operation
_mnc	MNC Code, such as 01	MNC Code This information is optional. It can be used only in developer mode. MNC Code is Mobile Network Code.

Example code for retrieving a list of purchased items

```
app_control_h app_control;
int rt = app_control_create(&app_control);

if (rt == APP_CONTROL_ERROR_NONE){
		app_control_set_app_id(app_control, " org.tizen.inapppurchase.iapservice");
		app_control_set_operation(app_control,
"http://tizen.org/appcontrol/operation/iapv2/get_purchased_item_list");
		app_control_add_extra_data(app_control, "_mode", "0");
		app_control_add_extra_data(app_control, "_transactionId", "123");
		app_control_add_extra_data(app_control, "_startNumber", "1");
		app_control_add_extra_data(app_control, "_endNumber", "10");
		app_control_add_extra_data(app_control, "_startDate", "20140101");
```

```
		app_control_add_extra_data(app_control, "_endDate", "20141231");
		app_control_add_extra_data(app_control, "_itemGroupId", "100000000012");
		app_control_add_extra_data(app_control, "_languageCd", "ENG");

		rt = app_control_send_launch_request(app_control, get_purchased_item_list_cb, NULL);
}

if (app_control != NULL){
		app_control_destroy(app_control);
}
```

Output Data

The results of the operation are returned in the app control callback.

The following table shows the output data for the `http://tizen.org/appcontrol/operation/iapv2/get_purchased_item_list` operation:

Key	Value	Description
_method	OnPurchasedItem InformationListReceived	Method to be called as a GetPurchasedItemList request result. This value will be OnPurchasedItem InformationListReceived
_result	<Result code>	Result code number. Result codes are described in the end of this section.
_resultDescription	<Result code/Function ID number>	“Result code /Function ID” when “_result” value is not “0” (success). Result codes are described in the end of this section
_transactionId	<TransactionID>	Transaction ID number This is the same as the transaction ID that is used to request.
_startNumber	<start number>	Start Number Index of the first item on the list
_endNumber	<end number>	End Number Index of the last item on the list
_totalCount	<total count>	Total items count Number of items based on the startNumber and endNumber

Key	Value	Description
_itemTotalCount	<item total count>	Total count of purchased items in the group ID.

We also have _totalCount number of items in the output data, each key is a PREFIX (list item index value) plus a key (i.e. “12_itemId”) as below:

Key	Value	Description
PREFIX_itemId	<ItemID>	Item ID number This is the same as a Item ID that is used to request.
PREFIX_itemGroupId	<ItemGroup ID>	Item group ID This is a collection of items and it will be linked to your application in Tizen Store Seller Office.
PREFIX_itemName	<Item Name>	Item Name This is a name provided during item registration on the Tizen Store Seller Office.
PREFIX_currencyUnit	Currency unit, such as: $,Won, Pound	Currency Unit Device user currency unit.
PREFIX_unitPrecedes	0 | 1	Unit Precedes String representation of values: 0 : Tall (2.99 TL) 1 : Front ($ 2.99)
PREFIX_hasPenny	0 | 1	Has Penny Informations if currency unit has penny: representation 0 : no 1 : yes
PREFIX_itemPrice	<Item price>	Item price This is a price of item in local currency.

Key	Value	Description
PREFIX_itemDownloadUrl	<URL>	Item download URL This is a URL provided during item registration on the Tizen Store Seller Office.
PREFIX_itemImageUrl	<URL>	Item image URL This is a URL provided during item registration on the Tizen Store Seller Office
PREFIX_itemDescription	<Item description>	Item description A description provided during item registration.
PREFIX_reserved1	<Reserved Field 1>	Reserved Field 1
PREFIX_reserved2	<Reserved Field 2>	Reserved Field 2
PREFIX_paymentId	<Payment ID>	ID of payment
PREFIX_purchaseDate	<Date>	Date of purchase
PREFIX_itemTypeCd	00 | 01 | 02	Item type code String representation of values: 00: Non-consumable 01: Consumable 02: Subscription (Non-renewing)
PREFIX_itemSubsBillDurationCd	00 | 01 | 02 | 03	Item subs bill duration code '_itemTypeCd' Value is '02', the only. If the product type is subscription, the available four units for the validity period of the product are YEAR, MONTH, WEEK, and DAY. The units should be in capital code. String representation of values: 00: Year 01: Month 02: Week 03: Day

Key	Value	Description
PREFIX_subscriptionDurationMultiplier	Subscription duration multiplier	If the _itemTypeCd is subscription (-02), this is the item duration. Combined with PREFIX_itemSubsBillDurationCd, it is means 1MONTH.
PREFIX_timeStamp	<Time stamp>	Time stamp Based on GMT +0, Server time. (yyyyMMddHHmmss)

Example code for retrieving a list of purchased items

```
void get_purchased_item_list_cb(app_control_h request, app_control_h reply, app_control_result_e result, void *user_data){

	char* rt_method = NULL;
	char* rt_result = NULL;
	char* rt_resultDescription = NULL;
	char* rt_transactionId = NULL;
	char* rt_startNumber = NULL;
	char* rt_endNumber = NULL;
	char* rt_totalCount = NULL;
	char* rt_itemTotalCount = NULL;
	char* rt_itemId = NULL;
	char* rt_itemGroupId = NULL;
	char* rt_itemName = NULL;
	char* rt_currencyUnit = NULL;
	char* rt_unitPrecedes = NULL;
	char* rt_hasPenny = NULL;
	char* rt_itemPrice = NULL;
	char* rt_itemDownloadUrl = NULL;
	char* rt_itemImageUrl = NULL;
	char* rt_itemDescription = NULL;
	char* rt_reserved1 = NULL;
	char* rt_reserved2 = NULL;
	char* rt_paymentId = NULL;
	char* rt_purchaseDate = NULL;
	char* rt_itemTypeCd = NULL;
	char* rt_itemSubsBillDurationCd = NULL;
	char* rt_subscriptionDurationMultiplier = NULL;
	char* rt_timeStamp = NULL;

	if (result == APP_CONTROL_RESULT_SUCCEEDED){

		rt = app_control_get_extra_data(reply, "_method", &rt_method);
		rt = app_control_get_extra_data(reply, "_result", &rt_result);

		//succeed
```

```
		if (!strcmp("0", rt_result)){
			rt = app_control_get_extra_data(reply, "_resultDescription", &rt_resultDescription);
			rt = app_control_get_extra_data(reply, "_transactionId", &rt_transactionId);
			rt = app_control_get_extra_data(reply, "_startNumber", &rt_startNumber);
			rt = app_control_get_extra_data(reply, "_endNumber", &rt_endNumber);
			rt = app_control_get_extra_data(reply, "_totalCount", &rt_totalCount);
			rt = app_control_get_extra_data(reply, "_itemTotalCount", &rt_itemTotalCount);

			int start = atoi(rt_startNumber);
			int end = atoi(rt_endNumber);

			char keyId[100] = {0, };
			for(; start <= end; start++){
				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemId");
				rt = app_control_get_extra_data(reply, key_id, &rt_itemId);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemGroupId);");
				rt = app_control_get_extra_data(reply, key_id, &rt_itemGroupId);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemName");
				rt = app_control_get_extra_data(reply, keyId, &rt_itemName);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_currencyUnit");
				rt = app_control_get_extra_data(reply, keyId, &rt_currencyUnit);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_unitPrecedes");
				rt = app_control_get_extra_data(reply, keyId, &rt_unitPrecedes);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_hasPenny");
				rt = app_control_get_extra_data(reply, keyId, &rt_hasPenny);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemPrice");
				rt = app_control_get_extra_data(reply, keyId, &rt_itemPrice);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemDownloadUrl");
				rt = app_control_get_extra_data(reply, keyId, &rt_itemDownloadUrl);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemImageUrl");
				rt = app_control_get_extra_data(reply, keyId, &rt_itemImageUrl);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemDescription");
				rt = app_control_get_extra_data(reply, keyId, &rt_itemDescription);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_reserved1");
				rt = app_control_get_extra_data(reply, keyId, &rt_reserved1);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_reserved2");
				rt = app_control_get_extra_data(reply, keyId, &rt_reserved2);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_paymentId");
				rt = app_control_get_extra_data(reply, keyId, &rt_paymentId);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_purchaseDate");
				rt = app_control_get_extra_data(reply, keyId, &rt_purchaseDate);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemTypeCd");
				rt = app_control_get_extra_data(reply, keyId, &rt_itemTypeCd);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_itemSubsBillDurationCd");
```

```
                                        rt = app_control_get_extra_data(reply, keyId, &rt_itemSubsBillDurationCd);

                                        snprintf(key_id, sizeof(keyId), "%d%s", start, "_subscriptionDurationMultiplier");
                                        rt = app_control_get_extra_data(reply, keyId, &rt_subscriptionDurationMultiplier);

                                        snprintf(key_id, sizeof(keyId), "%d%s", start, "_timeStamp");
                                        rt = app_control_get_extra_data(reply, keyId, &rt_timeStamp);
                                }
                        }
                }
}
```

GetCountryList operation

This operation returns a list of countries available for testing in-application purchase.

Input data

The following table show the (key, value) pairs required in the input extra data for the `http://tizen.org/appcontrol/operation/iapv2/get_country_list` operation:

Key	Value	Description
_transactionId	Transaction ID number, such as 1, 2, …	Transaction ID. This information is mandatory. This is used to track a transaction between requests.

Example code for get country list

```
app_control_h app_control;
int rt = app_control_create(&app_control);

if (rt == APP_CONTROL_ERROR_NONE){
        app_control_set_app_id(app_control, " org.tizen.inapppurchase.iapservice");
        app_control_set_operation(app_control,
"http://tizen.org/appcontrol/operation/iapv2/get_country_list");
        app_control_add_extra_data(app_control, "_mode", "0");
        app_control_add_extra_data(app_control, "_transactionId", "123");

        rt = app_control_send_launch_request(app_control, get_country_list_cb, NULL);
}

if (app_control != NULL){
        app_control_destroy(app_control);
}
```

Output Data

The results of the operation are returned in the app control callback.

The following table shows the output data for the `http://tizen.org/appcontrol/operation/iapv2/get_country_list` operation:

Key	Value	Description
_method	OnCountryListReceived	Method to be called as a purchase request result. This value will be OnCountryListReceived
_result	<Result code>	Result code number. Result codes are described in the end of this section.
_resultDescription	<Result code/Function ID number>	“Result code /Function ID” when “_result” value is not “0” (success). Result codes are described in the end of this section
_transactionId	<TransactionID>	Transaction ID number This is the same as the transaction ID that is used to request.
_startNumber	<start number>	Start Number Index of the first item on the list
_endNumber	<end number>	End Number Index of the last item on the list
_totalCount	<total count>	Total items count Number of items based on the startNumber and endNumber

We also have _totalCount number of countries in the output data, each key is a PREFIX (list item index value) plus a key (i.e. “1_countryName”) as below:

Key	Value	Description
PREFIX_countryName	<String>	Country Name Name of a country.
PREFIX_mcc	<String>	MCC Mobile country code.

Example code for retrieving a list of get country list

```
void get_country_list_cb(app_control_h request, app_control_h reply, app_control_result_e result, void *user_data){

	char* rt_method = NULL;
	char* rt_result = NULL;
	char* rt_resultDescription = NULL;
	char* rt_transactionId = NULL;
	char* rt_startNumber = NULL;
	char* rt_endNumber = NULL;
	char* rt_totalCount = NULL;
	char* rt_countryName = NULL;
	char* rt_mcc = NULL;


	if (result == APP_CONTROL_RESULT_SUCCEEDED){

		rt = app_control_get_extra_data(reply, "_method", &rt_method);
		rt = app_control_get_extra_data(reply, "_result", &rt_result);

		//succeed
		if (!strcmp("0", rt_result)){
			rt = app_control_get_extra_data(reply, "_resultDescription", &rt_resultDescription);
			rt = app_control_get_extra_data(reply, "_transactionId", &rt_transactionId);
			rt = app_control_get_extra_data(reply, "_startNumber", &rt_startNumber);
			rt = app_control_get_extra_data(reply, "_endNumber", &rt_endNumber);
			rt = app_control_get_extra_data(reply, "_totalCount", &rt_totalCount);

			int start = atoi(rt_startNumber);
			int end = atoi(rt_endNumber);

			char keyId[100] = {0, };
			for(; start <= end; start++){
				snprintf(key_id, sizeof(keyId), "%d%s", start, "_countryName");
				rt = app_control_get_extra_data(reply, key_id, &rt_countryName);

				snprintf(key_id, sizeof(keyId), "%d%s", start, "_mcc");
				rt = app_control_get_extra_data(reply, key_id, &rt_mcc);
			}
		}
	}
}
```

Result code values

Following is a list of possible values of `_result` key for the `org.tizen.inapppurchase.iapservice` Application ID:

Value	Description	Remarks
0	*Succeed*	The status code for success
200	*NetworkError*	The status code for network errors

1000	*ProcessError*	The status code for process errors
2001	*NoApplicationStore*	The status code if application store not present for the country
9201	*ItemGroupIdNotFound*	The status code if the item group ID is not found
9207	*ItemIdNotFound*	The status code if the item ID is not found
9502	*InvalidRequestParameter*	The status code if the wrong request parameter passed

3.3 IAP Client Control (Purchase of item)

The IAP Client instance allows you to initialize a purchase of item and to complete the purchase.

Application ID

This application control can be accessed using aliased application ID of "`org.tizen.inapppurchase.iapclient`".

Operation ID

This application supports the `http://tizen.org/appcontrol/operation/iapv2/purchase` operation only.

Purchase operation

This operation launches the purchase application and allows purchasing of In-App items. The input data passed in this operation are used to display a purchase form for particular item. The device user can choose between available payment methods, register his credit card, and confirm the purchase.

Input data

The following table show the (key, value) pairs required in the input extra data for the `http://tizen.org/appcontrol/operation/iapv2/purchase` operation.

Key	Value	Description
_mode	0 | 1	Mode type. This information is optional. The default value is 0. 0 is normal (commercial) mode. This mode need to be used in application submitted to Tizen Store. 1 is developer mode (always success). This mode can be used while development of the application for testing purpose. Payment will always succeed.
_transactionId	Transaction ID number, such as 1, 2, …	Transaction ID. This information is mandatory. This is used to track a transaction between

Key	Value	Description
		requests.
_itemId	ID of Item, such as 000000003501	Item ID This information is mandatory. Item ID is associated with the particular Item in the Tizen Store Seller Office. You can also retrieve a list of items available for purchase through org.tizen.inapppurchase.iapservice Application ID
_itemGroupId	ID of Group, such as 100000001455	Group ID This information is mandatory. Group ID is a associated with the particular Group of items in the Tizen Store Seller Office. You need to register your Group Id on the Tizen Store Seller Office first.
_languageCd	Language code such as eng, rus	Language code This information is optional. According to ISO 639-2, thus the name three character code. Language Code is associated with display language of item details in the Tizen Store Seller Office site. Or leave it to use default name as in seller site _itemName is higher priority than _languageCd.
_itemName	Item Name such as “Sword” or “칼”	Item Name This information is optional. You can set your item name with _item Name. Or leave it to use default name as in seller site’s “purchase page” of IAP.
_mcc	MCC Code, such as 250	MCC Code This information is optional. It can be used only in developer mode. You can receive a list of available MCC codes through get country list operation
_mnc	MNC Code, such as 01	MNC Code

Key	Value	Description
		This information is optional. It can be used only in developer mode. MNC Code is Mobile Network Code.

Example code for Purchase

```
app_control_h app_control;
int rt = app_control_create(&app_control);

if (rt == APP_CONTROL_ERROR_NONE){
        app_control_set_app_id(app_control, " org.tizen.inapppurchase.iapclient");
        app_control_set_operation(app_control, "http://tizen.org/appcontrol/operation/iapv2/purchase");
        app_control_add_extra_data(app_control, "_mode", "0");
        app_control_add_extra_data(app_control, "_itemId", "000000000001");
        app_control_add_extra_data(app_control, "_itemGroupId", "100000000012");
        app_control_add_extra_data(app_control, "_languageCd", "ENG");          //optional
        app_control_add_extra_data(app_control, "_itemName", "Item 1");         //optional

        rt = app_control_send_launch_request(app_control, get_purchase_cb, NULL);
}

if (app_control != NULL){
        app_control_destroy(app_control);
}
```

Output Data

The results of the operation are returned in the app control callback.

This data can be used to verify the payment with Tizen Store IAP Server.

The following table shows the output data for the `http://tizen.org/appcontrol/operation/iapv2/purchase` operation during process of purchase:

Key	Value	description
_method	OnPurchaseItemReceived	Method to be called as a purchase request result This value will be OnPurchaseItemReceived.
_result	<Result code>	Result code number. Result codes are described in the end of this section.
_resultDescription	<Result code/Function ID number> | <HTML tags>	Display _resultDescription value as a pop-up by using Web-Control when you receive the “_result” value as 5600 Result codes are described in the end of this section. ex>“(E1000/9100)”

Key	Value	description
_itemId	<ItemID>	Item ID number This is the same as an Item ID that is used to request.
_itemGroupId	<ItemGroup ID>	Item group ID Item Group ID is a collection of items and it will be linked to your application in Tizen seller site.
_itemName	<Item Name>	Item Name Item Name is a name provided during item registration on the Tizen Store Seller Office.
_ticketPurchaseId	<Purchase Ticked ID>	Purchase ticked ID This ID can be used to verify the purchase with Tizen Store IAP Server.
_currencyUnit	Currency unit, such as: $,Won, Pound	Currency Unit Device user currency unit.
_unitPrecedes	0 | 1	Unit Precedes String representation of values: 0 : Tall (2.99 TL) 1 : Front ($ 2.99)
_hasPenny	0 | 1	Has Penny Informations if currency unit has penny: representation 0 : no 1 : yes
_itemPrice	<Item price>	Item price This is a price of item in local currency.
_itemDownloadUrl	<URL>	Item download URL This is a URL provided during item registration on the Tizen Store Seller Office.
_itemImageUrl	<URL>	Item image URL This is a URL provided during item registration on the Tizen Store Seller Office
_itemDescription	<Item description>	Item description A description provided during item registration.

Key	Value	description
_reserved1	<Reserved Field 1>	Reserved Field 1
_reserved2	<Reserved Field 2>	Reserved Field 2
_paymentId	<Payment ID>	Payment ID ID of payment.
_ticketVerifyUrl	URL, such as: http://tizen.org/appcontrol/operation/iap/purchase	Server's URL This URL can be used with combination of other parameters to verify the purchase with Tizen Store IAP Server.
_ticketPurchaseId	<Purchase TickedID>	Purchase ticked ID This ID can be used to verify the purchase with Tizen Store IAP Server.
_ticketParam1	<Ticket parameter 1>	Parameter 1 This parameter is to be used with URL
_ticketParam2	<Ticket parameter 2>	Parameter 2 This parameter is to be used with URL
_ticketParam3	<Ticket parameter 3>	Parameter 3 This parameter is to be used with URL
_ticketParam4	<Ticket parameter 4>	Parameter 4 This parameter is to be used with URL
_ticketParam5	<Ticket parameter 5>	Parameter 5 This parameter is to be used with URL
_purchaseDate	Date	Purchase date Date of purchase.
_timeStamp	<Time stamp>	Time stamp Based on "GMT +0", Server time. (yyyyMMddHHmmss)

Example code for Purchase result

```
void get_purchase_cb(app_control_h request, app_control_h reply, app_control_result_e result,
        void *user_data){

        char* rt_method = NULL;
        char* rt_result = NULL;
        char* rt_resultDescription = NULL;
```

```
	char* rt_transactionId = NULL;
	char* rt_itemId = NULL;
	char* rt_itemGroupId = NULL;
	char* rt_itemName = NULL;
	char* rt_currencyUnit = NULL;
	char* rt_unitPrecedes = NULL;
	char* rt_hasPenny = NULL;
	char* rt_itemPrice = NULL;
	char* rt_itemDownloadUrl = NULL;
	char* rt_itemImageUrl = NULL;
	char* rt_itemDescription = NULL;
	char* rt_reserved1 = NULL;
	char* rt_reserved2 = NULL;
	char* rt_paymentId = NULL;
	char* rt_ticketVerifyUrl = NULL;
	char* rt_ticketPurchaseId = NULL;
	char* rt_ticketParam1 = NULL;
	char* rt_ticketParam2 = NULL;
	char* rt_ticketParam3 = NULL;
	char* rt_ticketParam4 = NULL;
	char* rt_ticketParam5 = NULL;
	char* rt_purchaseDate = NULL;
	char* rt_itemTypeCd = NULL;
	char* rt_itemSubsBillDurationCd = NULL;
	char* rt_subscriptionDurationMultiplier = NULL;
	char* rt_timeStamp = NULL;

	if (result == APP_CONTROL_RESULT_SUCCEEDED){

		rt = app_control_get_extra_data(reply, "_method", &rt_method);
		rt = app_control_get_extra_data(reply, "_result", &rt_result);

		//succeed
		if (!strcmp("0", rt_result)){
			rt = app_control_get_extra_data(reply, "_resultDescription", &rt_resultDescription);
			rt = app_control_get_extra_data(reply, "_transactionId", &rt_transactionId);
			rt = app_control_get_extra_data(reply, "_itemId", &rt_itemId);
			rt = app_control_get_extra_data(reply, "_itemGroupId", &rt_itemGroupId);
			rt = app_control_get_extra_data(reply, "_itemName", &rt_itemName);
			rt = app_control_get_extra_data(reply, "_currencyUnit", &rt_currencyUnit);
			rt = app_control_get_extra_data(reply, "_unitPrecedes", &rt_unitPrecedes);
			rt = app_control_get_extra_data(reply, "_itemPrice", &rt_itemPrice);
			rt = app_control_get_extra_data(reply, "_itemDownloadUrl", &rt_itemDownloadUrl);
			rt = app_control_get_extra_data(reply, "_itemImageUrl", &rt_itemImageUrl);
			rt = app_control_get_extra_data(reply, "_itemDescription", &rt_itemDescription);
			rt = app_control_get_extra_data(reply, "_reserved1", &rt_reserved1);
			rt = app_control_get_extra_data(reply, "_reserved2", &rt_reserved2);
			rt = app_control_get_extra_data(reply, "_paymentId", &rt_paymentId);
			rt = app_control_get_extra_data(reply, "_ticketVerifyUrl", &rt_ticketVerifyUrl);
			rt = app_control_get_extra_data(reply, "_ticketPurchaseId", &rt_ticketPurchaseId);
			rt = app_control_get_extra_data(reply, "_ticketParam1", &rt_ticketParam1);
```

```
                                rt = app_control_get_extra_data(reply, "_ticketParam2", &rt_ticketParam2);
                                rt = app_control_get_extra_data(reply, "_ticketParam3", &rt_ticketParam3);
                                rt = app_control_get_extra_data(reply, "_ticketParam4", &rt_ticketParam4);
                                rt = app_control_get_extra_data(reply, "_ticketParam5", &rt_ticketParam5);
                                rt = app_control_get_extra_data(reply, "_purchaseDate", &rt_purchaseDate);
                                rt = app_control_get_extra_data(reply, "_itemTypeCd", &rt_itemTypeCd);
                                rt = app_control_get_extra_data(reply, "_itemSubsBillDurationCd",
&rt_itemSubsBillDurationCd);
                                rt = app_control_get_extra_data(reply, "_subscriptionDurationMultiplier",
&rt_subscriptionDurationMultiplier);
                                rt = app_control_get_extra_data(reply, "_timeStamp", &rt_timeStamp);
                        }
                }
}
```

Result code values

Following is a list of possible values of `_result` key of the `org.tizen.inapppurchase.iapclient` application ID.

Value	Description	Remarks
0	*Succeed*	The status code for success
100	*Cancel*	The status code if the user cancels
200	*NetworkError*	The status code for network errors
1000	*ProcessError*	The status code for process errors
5600	*PGError*	The status code for the payment gateway error. Display the pop-up by using web-control.
9201	*ItemGroupIdNotFound*	The status code if the item group ID is not found
9207	*ItemIdNotFound*	The status code if the item ID is not found
9502	*InvalidRequestParameter*	The status code if the request parameter is invalid
9291	*RepurchaseError*	The status code for repurchase error. This error occurs only on consumable items.
9292	*Update is progressing*	The status code if the application is on updating.
9293	*Account validation is not Completed*	The status code if the Samsung account validation is not completed.

4. Verifying with Tizen Store IAP Server

After purchase operation, `http://tizen.org/appcontrol/operation/iapv2/purchase`, you can use output data's, values to verify a purchase on the Tizen Store IAP Server. It's simple process. You can send a query to output data's `_ticketVerifyUrl` address. The query can be made with combination of output data's `_ticketPurchaseId`, `_ticketParam1`, `_ticketParam2`, `_ticketParam3`, `_ticketParam4`, `_ticketParam5`.

Template as following,

```
Value of (_ticketVerifyUrl)?purchaseID=value of (_ticketPurchaseId)&param1=value of
(_ticketParam1)&param2=value of (_ticketParam2)&param3=value of
(_ticketParam3)&param4=value of (_ticketParam4)&param5=value of (_ticketParam5)
```

The below is an example of request:

```
http://iap.tizenstore.com/appsItemVerifyIAPReceipt.as?purchaseID=2bf8fe4fdef1dae29974e5
400c106bfced6a650793efa3ce68a79e026481193d&param1=abe87f635bf41aae0178b5384cbc09c108302
6e93e44bd0efd69e66a9cc2ace6&param2=d49e3385783366868999e17bae3410597c0b6bcf69a92cd74cab
17454cf9d4d6&param3=af8050beb9c0f63c773fb86f7218bb6cbd6cf78a5cdff281c2e22a229f9a1485&pa
ram4=1&param5=I20131115RU00001243
```

If a request is processed successfully, the JSON response is received as shown. The status value of "true" means a verifying the purchase is successful and 'false' when it has failed.

The JSON example of successful result:

```
{"paymentID":"TPMTID20131115RU00001243","paymentAmount":"105.02","itemName":"ttttttt","
itemID":"000000000072","status":"true","purchaseDate":"2013-11-15
10:31:23","itemDesc":"dkfldfldkfl","paymentMethod":"Tizen RBS Russia
CreditCard","mode":"1"}
```

The JSON example of failed result

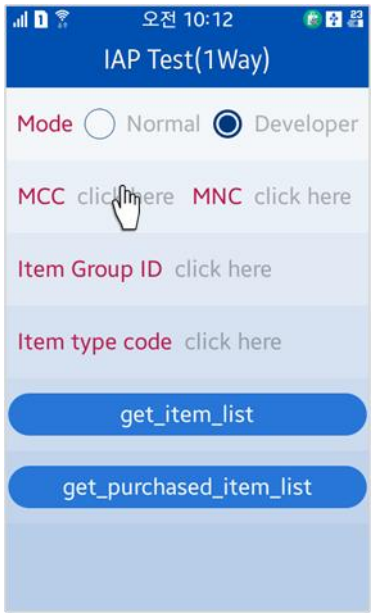
```
{"status":"false"}
```

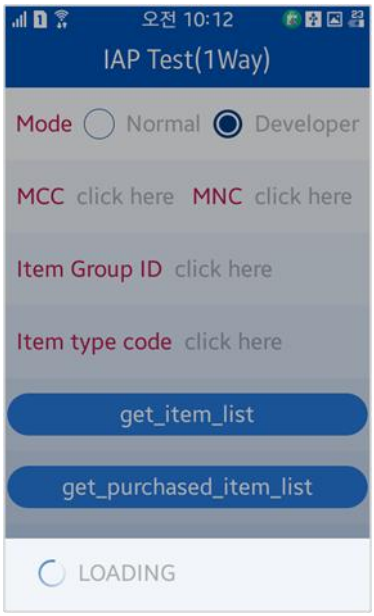
If your application uses a server-client model, server-to-server purchase verification explained above is recommended.

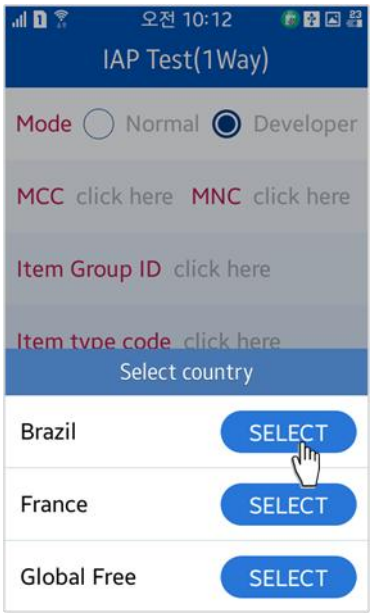
Attached sample native application

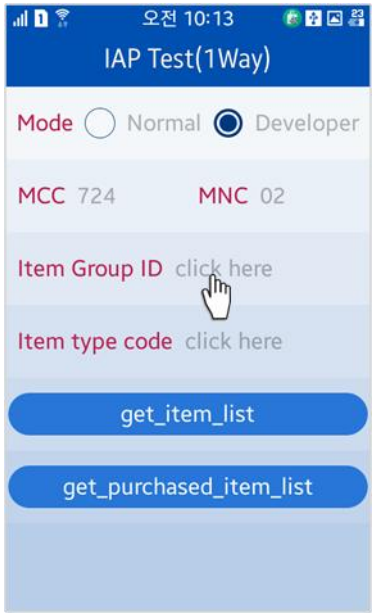
The attached sample native application allows a user to show a list of items for purchase, purchase an item, show purchased items, and make a purchase:

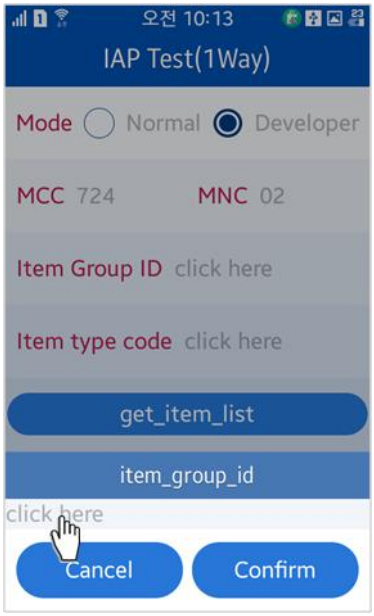
- In developer mode, the top of the screen, it is possible to scroll up and down direction.

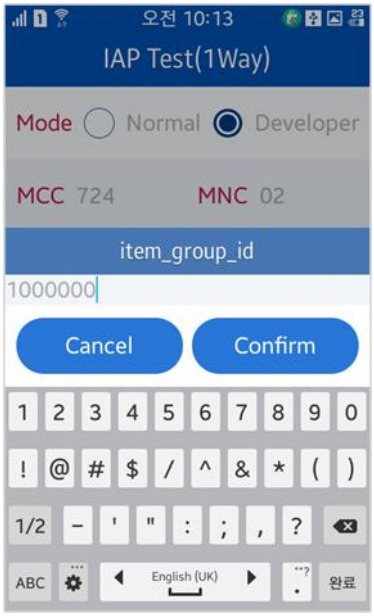


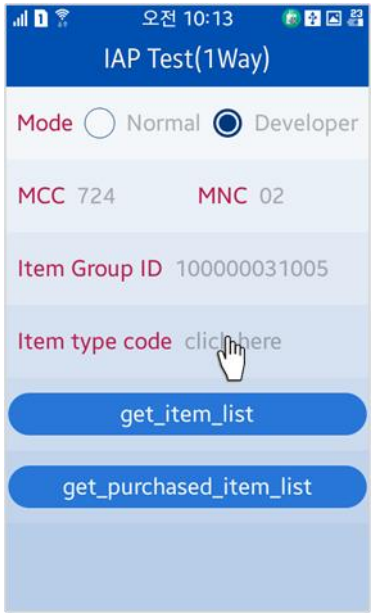


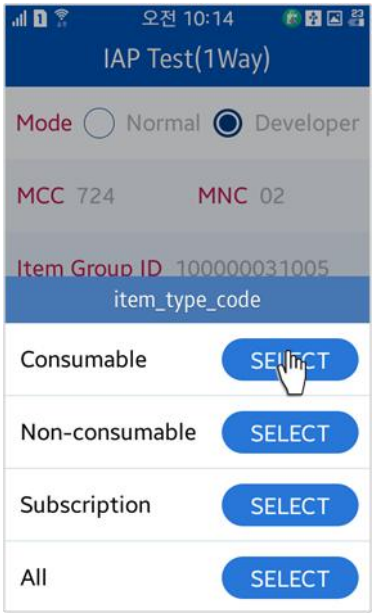


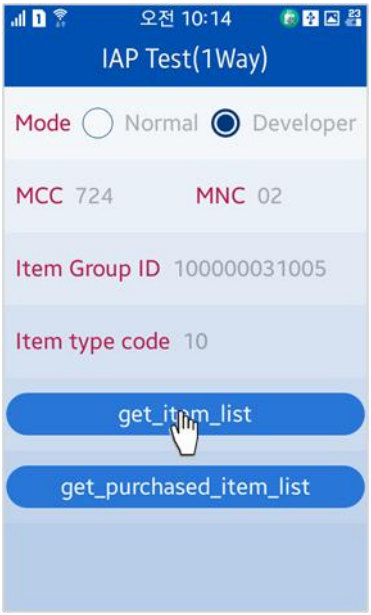


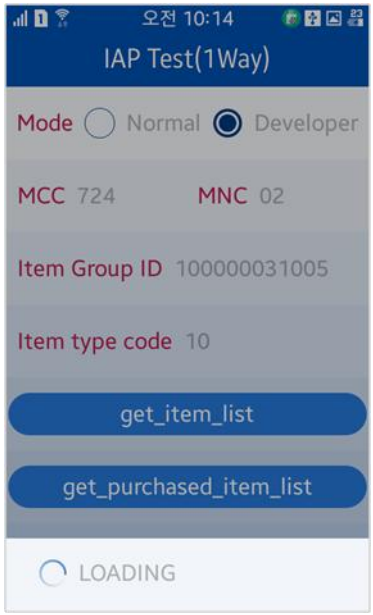


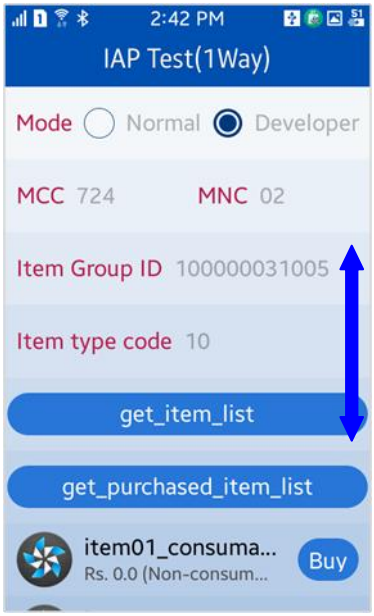


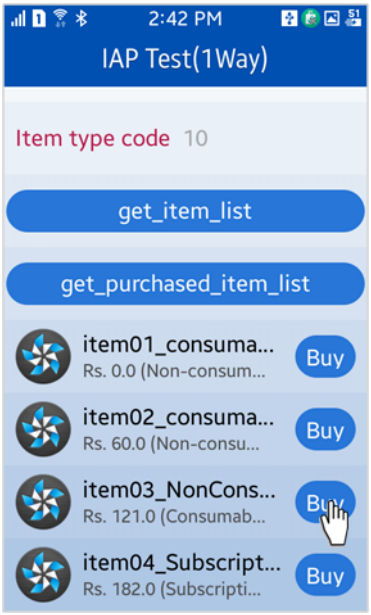

오전 10:13
IAP Test(1Way)
Mode Normal Developer
MCC 724 MNC 02
Item Group ID 100000031005
Item type code click here
get_item_list
get_purchased_item_list

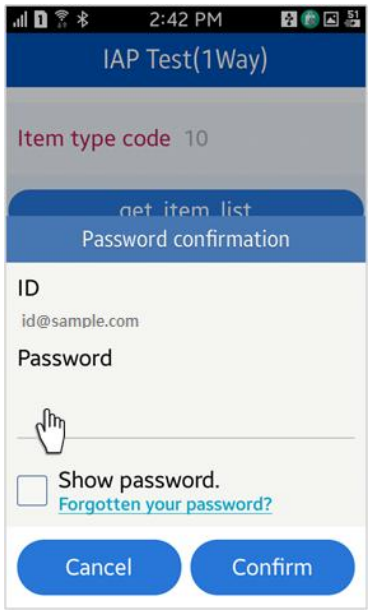

오전 10:14
IAP Test(1Way)
Mode Normal Developer
MCC 724 MNC 02
Item Group ID 100000031005
item_type_code
Consumable SELECT
Non-consumable SELECT
Subscription SELECT
All SELECT

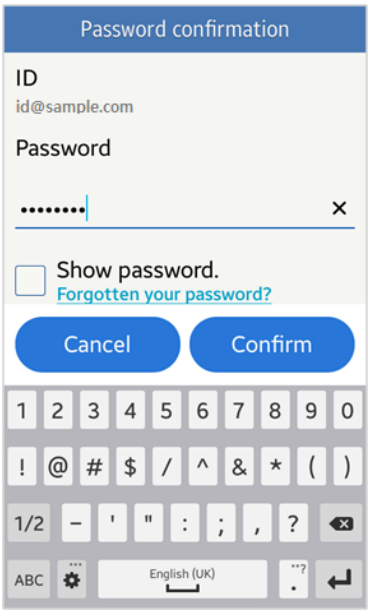

오전 10:14
IAP Test(1Way)
Mode Normal Developer
MCC 724 MNC 02
Item Group ID 100000031005
Item type code 10
get_item_list
get_purchased_item_list

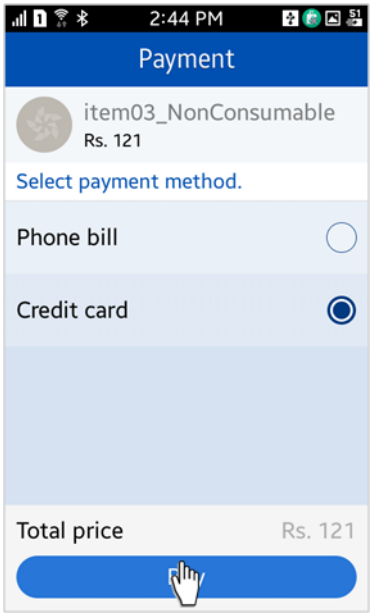

오전 10:14
IAP Test(1Way)
Mode Normal Developer
MCC 724 MNC 02
Item Group ID 100000031005
Item type code 10
get_item_list
get_purchased_item_list
LOADING


2:42 PM
IAP Test(1Way)
Mode Normal Developer
MCC 724 MNC 02
Item Group ID 100000031005
Item type code 10
get_item_list
get_purchased_item_list
item01_consuma...
Rs. 0.0 (Non-consum...
Buy

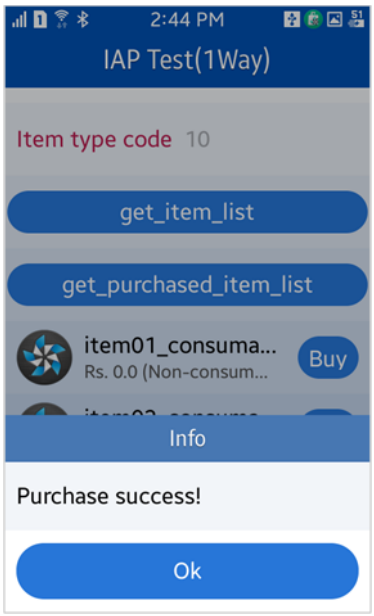

2:42 PM
IAP Test(1Way)
Item type code 10
get_item_list
get_purchased_item_list
item01_consuma...
Rs. 0.0 (Non-consum...
Buy
item02_consuma...
Rs. 60.0 (Non-consu...
Buy
item03_NonCons...
Rs. 121.0 (Consumab...
Buy
item04_Subscript...
Rs. 182.0 (Subscripti...
Buy

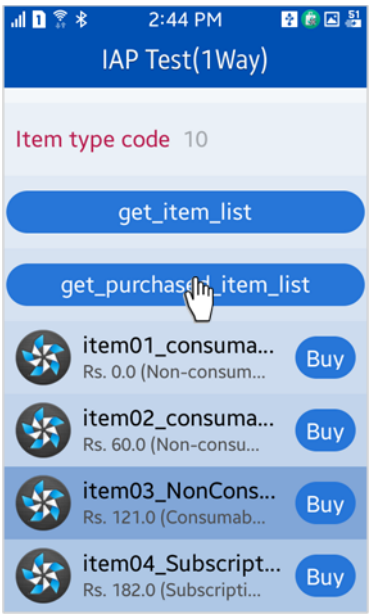

2:42 PM
IAP Test(1Way)
Item type code 10
get_item_list
Password confirmation
ID
id@sample.com
Password
Show password.
Forgotten your password?
Cancel
Confirm

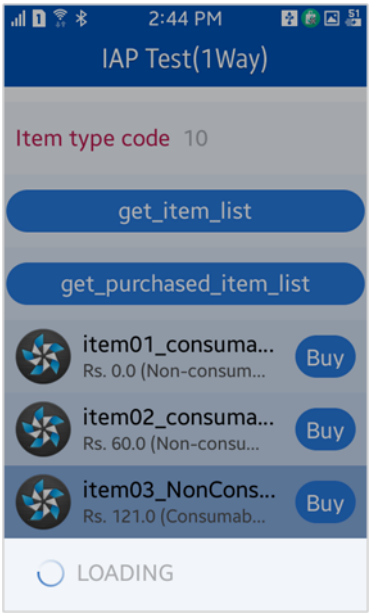

Password confirmation
ID
id@sample.com
Password
Show password.
Forgotten your password?
Cancel
Confirm

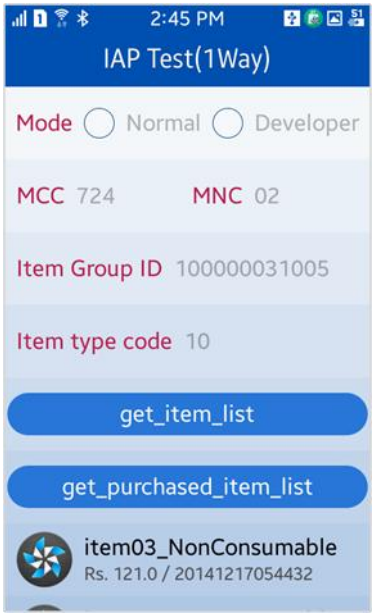

2:44 PM
Payment
item03_NonConsumable
Rs. 121
Select payment method.
Phone bill
Credit card
Total price
Rs. 121

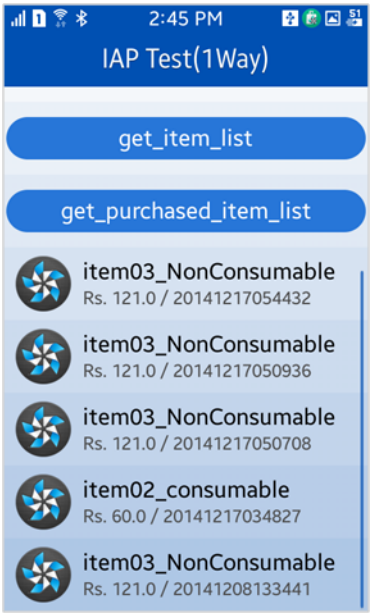

2:44 PM
Payment
item03_NonConsumable
Rs. 121
Select payment method.
Phone bill
Credit card
Total price
Rs. 121
LOADING


2:44 PM
IAP Test(1Way)
Item type code 10
get_item_list
get_purchased_item_list
item01_consuma...
Rs. 0.0 (Non-consum...
Buy
Info
Purchase success!
Ok


2:44 PM
IAP Test(1Way)
Item type code 10
get_item_list
get_purchased_item_list
item01_consuma...
Rs. 0.0 (Non-consum...
Buy
item02_consuma...
Rs. 60.0 (Non-consu...
Buy
item03_NonCons...
Rs. 121.0 (Consumab...
Buy
item04_Subscript...
Rs. 182.0 (Subscripti...
Buy





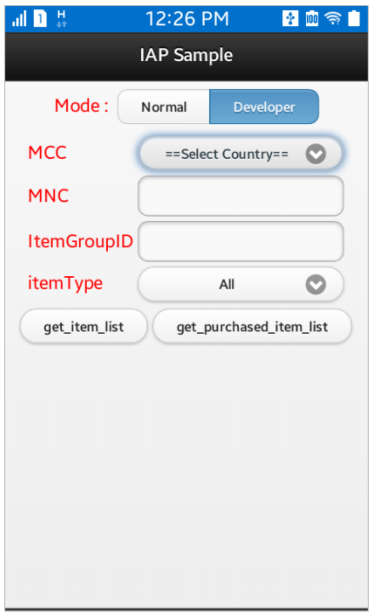


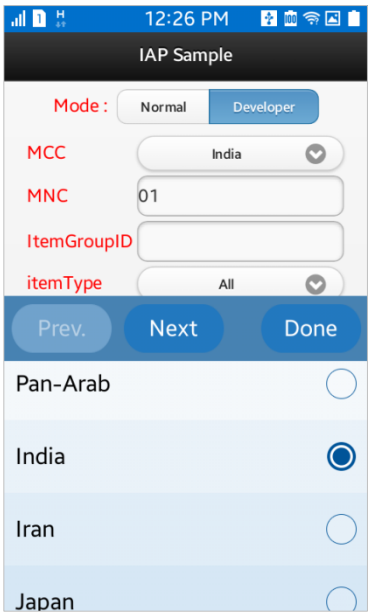
The sample flow of purchase item

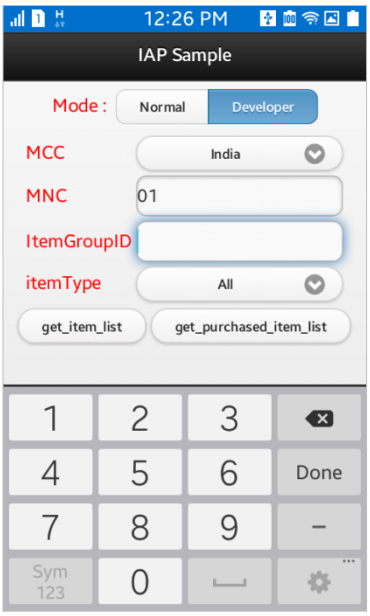
Above is an example of third case: when a user clicks on the “Simple purchase” button, a list of items will be shown, and a user will be able to purchase an item.

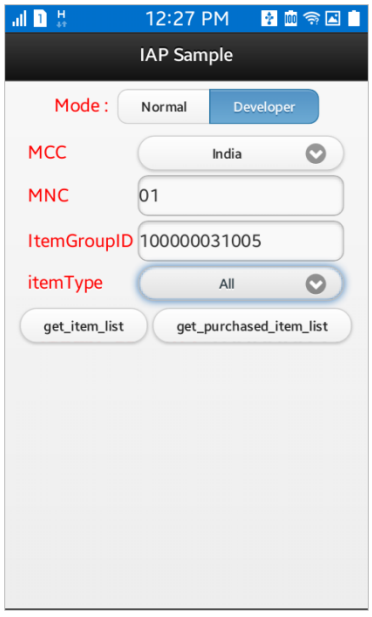
Attached sample web application

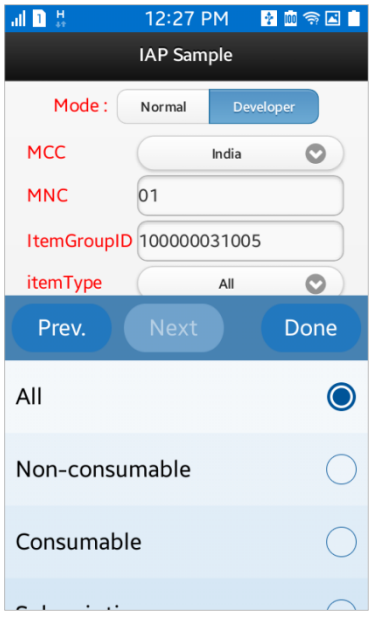
The attached sample web application allows a user to show a list of items for purchase, purchase an item, show purchased items, and make a purchase:

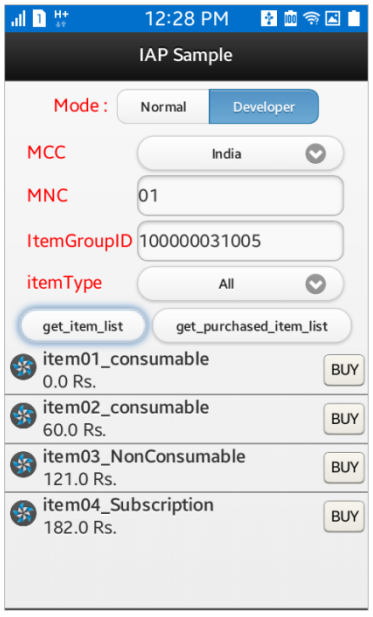


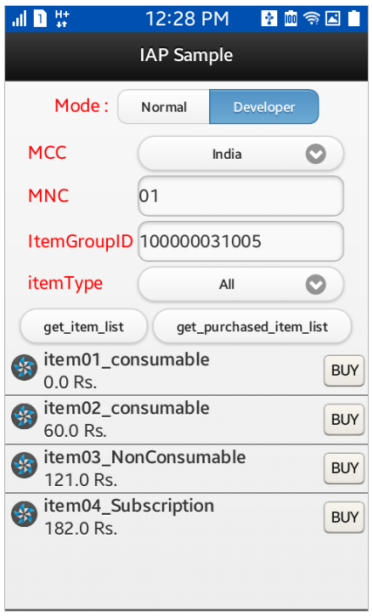


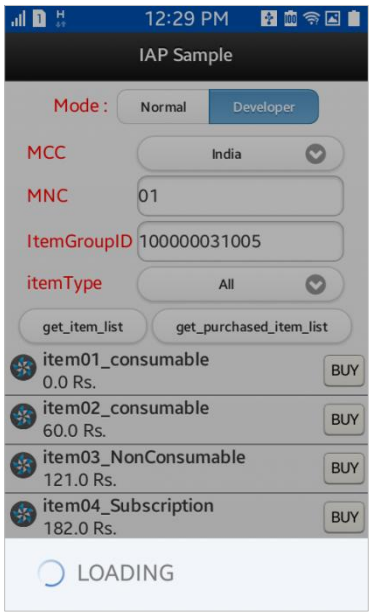


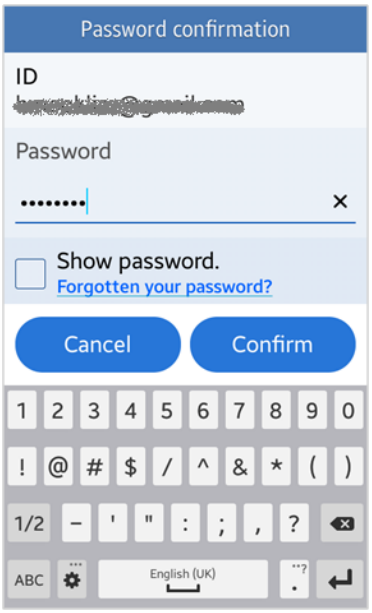


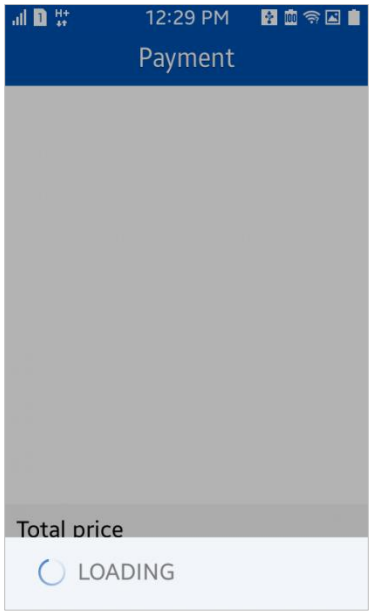


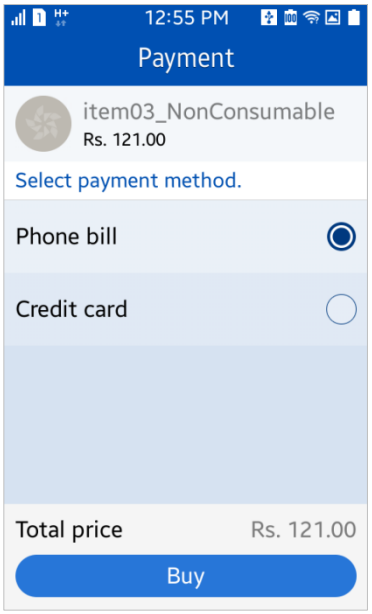


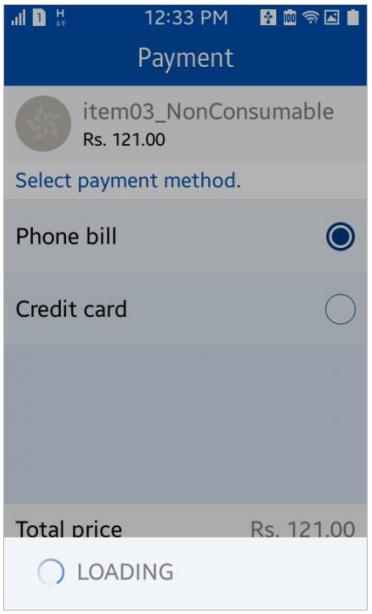

12:28 PM
IAP Sample
Mode :
Normal
Developer
MCC
India
MNC
01
ItemGroupID
100000031005
itemType
All
get_item_list
get_purchased_item_list
item01_consumable
0.0 Rs.
BUY
item02_consumable
60.0 Rs.
BUY
item03_NonConsumable
121.0 Rs.
BUY
item04_Subscription
182.0 Rs.
BUY


12:29 PM
IAP Sample
Mode :
Normal
Developer
MCC
India
MNC
01
ItemGroupID
100000031005
itemType
All
get_item_list
get_purchased_item_list
item01_consumable
0.0 Rs.
BUY
item02_consumable
60.0 Rs.
BUY
item03_NonConsumable
121.0 Rs.
BUY
item04_Subscription
182.0 Rs.
BUY
LOADING


Password confirmation
ID
Password
Show password.
Forgotten your password?
Cancel
Confirm


12:29 PM
Payment
Total price
LOADING


12:55 PM
Payment
item03_NonConsumable
Rs. 121.00
Select payment method.
Phone bill
Credit card
Total price
Rs. 121.00
Buy


12:33 PM
Payment
item03_NonConsumable
Rs. 121.00
Select payment method.
Phone bill
Credit card
Total price
Rs. 121.00
LOADING

The sample flow of purchase item

Above is an example of third case: when a user clicks on the “Simple purchase” button, a list of items will be shown, and a user will be able to purchase an item.

History

- 12 Jan 2015: The draft version of this guide document.
- 16 Feb 2015: complement the description of launch permission.