

eCommerce Mobile SDK-Android

Reference Guide

Version 1.0

# Document Configuration Management

## Document Identification

|  |  |
| --- | --- |
| **Name of Project** | eCommerce Mobile SDK |
| **Name of Report** | Consumer Mobile SDK Reference Guide |
| **File Name** | eCommerce Mobile Consumer SDK\_Android\_V1.0 |

## Document Change Control

|  |  |  |
| --- | --- | --- |
| Version | Date Released | Change Notice |
| 1.0 | 10-Oct-2019 | First Release – Mobile SDK for eCommerce app |
|  |  |  |

Table of Contents

[1 Introduction 4](#_Toc21594380)

[2 Initialize SDK 5](#_Toc21594382)

[3 Set Proposition ID 5](#_Toc21594383)

[4 Configuration 6](#_Toc21594384)

[5 Hybris Authentication 8](#_Toc21594387)

[6 Products 11](#_Toc21594390)

[7 Shopping Cart 16](#_Toc21594395)

[8 Voucher 20](#_Toc21594400)

[9 Delivery Mode 25](#_Toc21594404)

[10 Address 27](#_Toc21594407)

[11 Retailers 39](#_Toc21594418)

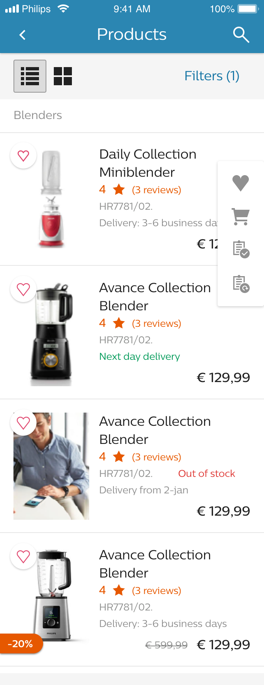
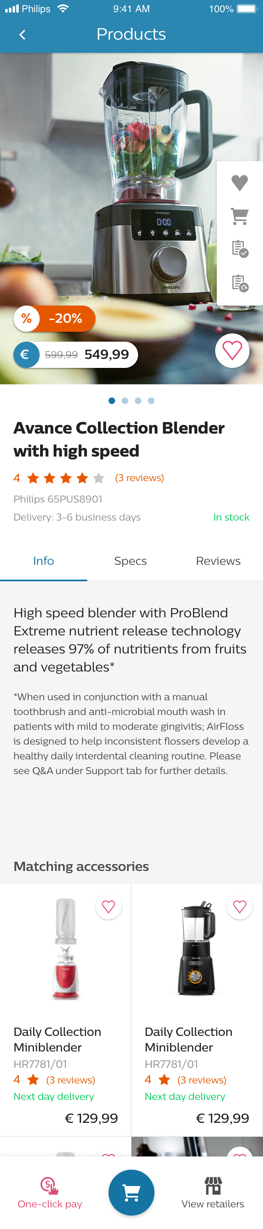
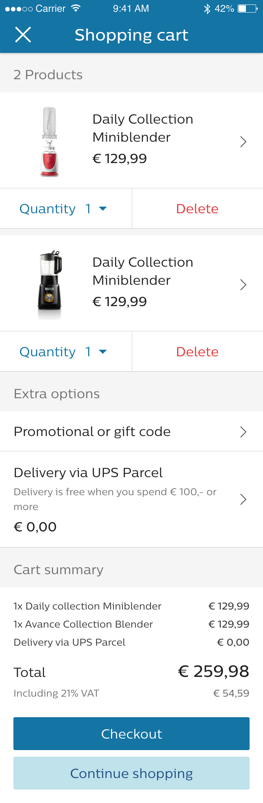
[12 Payment 41](#_Toc21594421)

[13 Order 45](#_Toc21594425)

[14 Fetch User Profile 50](#_Toc21594431)

# **Introduction**

# 



The eCommerce software development kit (SDK) defines the In-App purchase experience and how it is delivered to the users. Embedding this into your existing android application enables you to design an integrated solution.

The SDK provides the necessary tools to build the application and gives you control over how the application looks and operates. Your own workflow and images can be used to design all aspects of user experience such as how product list appears, how to display shopping cart, displaying address, what details to be shown and what information to collect or not collect during the intake process. This ability to uniquely tailor In-App purchase capabilities offers your application limitless possibilities for using the SDK and gives good user experience.

# **Initialize SDK**

eCommerce mobile SDK must be initialized with AppInfra object and proposition id for Hybris flow. Proposition id is a unique identifier for a business for specific country, which needs to be setup in back office. We recommend keeping a single ECSService instance to call all the APIs. Call the following API to initialize the SDK.

## ECSServiceProvider

**appInfra** = **new** AppInfra.Builder().build(getApplicationContext());

**public** ECSServices(@Nullable String propositionID, @NonNull appInfra)

{  
   
}

## This method has the following parameters:

* propositionID : It is a unique store identification string, which is specific to business and country
* appInfra : appInfra object created by proposition

## Sample

**appInfra** = **new** AppInfra.Builder().build(getApplicationContext());

ECSServices ecsServices = **new** ECSServices( **"your propositionID"**, appInfra);

# **Set Proposition ID**

This API is used to set the proposition id.

## ECSServiceProvider

**void** setPropositionID(@NonNull String propositionID);

## This method has the following parameter:

* propositionID : It is a unique store identification string, which is specific to business and country

## Sample

ECSConfiguration.***INSTANCE***.setPropositionID(**"your propositionID"**);

# **Configuration**

# **4.1 Configure ECS**

This API is used to configure the ECS SDK. This has to be called after ECS SDK is initialized or whenever proposition id or country is changed. This API initializes base URL and locale for the SDK and fetches configuration from Hybris.

## ECSServiceProvider

**void** configureECS(ECSCallback<Boolean,Exception> ecsCallback);

## This method has the following parameter:

* ecsCallback : an instance of ECSCallBack which contains TRUE if successfully configured,

otherwise FALSE and an Exception explaining any failures.

## Sample

ecsServices.configureECS(**new** ECSCallback<Boolean, Exception>() {  
 @Override  
 **public void** onResponse(Boolean result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5999 | ECSSomethingWentWrong | We have encountered technical glitch. Please try after some time |  |
| 5055 | ECSHybrisNotAvailable | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if Philips shop is not setup in Hybris office to use in mobile |

# **4.2 Configure ECS to Fetch the Configuration**

This API is used to configure the ECS. This has to be called after ECS SDK is initialized or whenever proposition id or country is changed. This API initializes base URL and locale for the SDK and fetches configuration from Hybris.

## ECSServiceProvider

**void** configureECSToGetConfiguration(ECSCallback<ECSConfig, Exception> ecsCallback);

## This method has the following parameter:

* ecsCallback : an instance of ECSCallBack which contains ECSConfig indicating success and

Exception explaining any failures.

## Sample

ecsServices.configureECSToGetConfiguration(**new** ECSCallback<ECSConfig, Exception>() {  
 @Override  
 **public void** onResponse(ECSConfig result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5999 | ECSSomethingWentWrong | We have encountered technical glitch. Please try after some time |  |
| 5055 | ECSHybrisNotAvailable | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if Philips shop is not setup in Hybris office to use in mobile |

# **Hybris Authentication**

# **5.1 Hybris OAuth Authentication**

This API is used to make the OAuth authentication with Hybris. User needs to send the object of class ECSOAuthProvider which contains all the authentication related data as a parameter to the method.

## ECSServiceProvider

**void** hybrisOAthAuthentication(@NonNull ECSOAuthProvider ecsoAuthProvider, @NonNull ECSCallback<ECSOAuthData,Exception> ecsListener);

## This method has the following parameters:

* ecsoAuthProvider : object of the class ECSOAuthProvider which provides information needed

for OAuth

* ecsCallback : an instance of ECSCallBack which contains ECSOAuthData indicating

success and Exception explaining any failures.

## Sample

ECSOAuthProvider oAuthInput = **new** ECSOAuthProvider() {  
 @Override  
 publicString getOAuthID() {  
 return "your Jainrain ID**"**;  
 }  
};

ecsServices. hybrisOAthAuthentication(oAuthInput, **new** ECSCallback<ECSOAuthData, Exception>() {  
 @Override  
 **public void** onResponse(ECSOAuthData result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5058 | ECSOAuthDetailError | Please provide complete OAuth details |  |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5000 | ECSinvalid\_grant | We have encountered technical glitch (Invalid Janrain Token). Please re-login | This error occurs if the janrain token provided for OAuth is invalid or expired |
| 5001 | ECSinvalid\_client | We have encountered technical glitch (Bad client credentials). Please re-login | This error occurs if the client details provided for OAuth is invalid or expired |
| 5002 | ECSUnsupported\_grant\_type | We have encountered technical glitch (Unsupported grant type). Please re-login |  |

# **5.2 Refresh Hybris OAuth**

This API is used to refresh the OAuth authentication with Hybris. User needs to send the object of class ECSOAuthProvider which contains all the authentication related data as a parameter to the API.

## ECSServiceProvider

**void** hybrisRefreshOAuth(ECSOAuthProvider oAuthInput, ECSCallback<ECSOAuthData, Exception> ecsListener);

## This method has the following parameters:

* ecsoAuthInput : object of the class ECSOAuthProvider which provides information needed

for OAuth

* ecsCallback : an instance of ECSCallBack which contains ECSOAuthData indicating

success and Exception explaining any failures.

## Sample

ECSOAuthProvider oAuthInput = **new** ECSOAuthProvider() {  
 @Override  
 publicString getOAuthID() {  
 return "your refresh token**"**;  
 }  
};

ecsServices. hybrisRefreshOAuth(oAuthInput, **new** ECSCallback<ECSOAuthData,Exception>() {  
 @Override  
 **public void** onResponse(ECSOAuthData result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5058 | ECSOAuthDetailError | Please provide complete OAuth details |  |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5000 | ECSinvalid\_grant | We have encountered technical glitch (Invalid Janrain Token). Please re-login | This error occurs if the janrain token provided for OAuth is invalid or expired |
| 5001 | ECSinvalid\_client | We have encountered technical glitch (Bad client credentials). Please re-login | This error occurs if the client details provided for OAuth is invalid or expired |

# **Products**

# **6.1 Fetch Product List**

This API is used to fetch the list of products from Hybris which includes summary data fetched from PRX. API can be called even if the user is not logged in.

## ECSServiceProvider

**void** fetchProducts(**int** currentPage, **int** pageSize,ECSCallback<ECSProducts,Exception> eCSCallback);

## This method has the following parameters:

* currentPage : page number for which the products have to be fetched
* pageSize : total number of products per page
* eCSCallback : an instance of ECSCallBack which contains list of ECSProducts indicating

success and Exception explaining any failures.

## Sample

ecsServices.fetchProducts(0, 20, **new** ECSCallback<ECSProducts, Exception>() {  
 @Override  
 **public void** onResponse(ECSProducts result) {  
  
 }  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server. |  |

# **6.2 Fetch Product for CTN**

This API is used to fetch the product for specific CTN from Hybris which includes summary data fetched from PRX. This API can be called even if the user is not logged in.

ECSServiceProvider

**void** fetchProduct(**String** ctn,ECSCallback<ECSProduct,Exception> eCSCallback);

## This method has the following parameters:

* ctn : CTN of the product
* eCSCallback : an instance of ECSCallBack which contains ECSProduct indicating

success and Exception explaining any failures.

## Sample

ecsServices.fetchProduct(mCTNValue, **new** ECSCallback <ECSProduct, Exception>() {  
 @Override  
 **public void** onResponse(ECSProducts result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server. |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **6.3 Fetch Product List for Set of CTNs**

This API is used to fetch list of products specific to list of CTNs from Hybris which includes summary data fetched from PRX. API can be called even if the user is not logged in.

## ECSServiceProvider

**void** fetchProductSummaries(**List<String>** ctns,ECSCallback<List<ECSProduct>,Exception> eCSCallback);

## This method has the following parameters:

* ctns : array of CTNs
* eCSCallback : an instance of ECSCallBack which contains list of ECSProduct indicating

success and Exception in case of failure

## Sample

ecsServices.fetchProductSummaries(productList, **new** ECSCallback<List<ECSProduct>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSProduct> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server. |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **6.4 Fetch Product Details**

This API is used to fetch product details containing assets and disclaimer for the specific product from PRX.

## ECSServiceProvider

**void** fetchProductDetails(ECSProduct product, ECSCallback<ECSProduct,Exception> ecsCallback);

## This method has the following parameters:

* product : an instance of ECSProduct for which detail needs to be fetched
* ecsCallback : an instance of ECSCallBack which includes ECSProduct containing details

indicating success and Exception in case of failure

## Sample

ecsServices.fetchProductDetails(**product**, **new** ECSCallback<ECSProduct, Exception>() {  
 @Override  
 **public void** onResponse(ECSProduct result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **Shopping Cart**

# **7.1 Create Shopping Cart**

This API is used to create the shopping cart for the user. User needs to be logged in to call this API.

## ECSServiceProvider

**void** createShoppingCart(ECSCallback<ECSShoppingCart,Exception> ecsCallback);

## This method has the following parameters:

* ecsCallback : an instance of ECSCallback which includes ECSShoppingCart indicating success

and Exception in case of failure

## Sample

ecsServices.createShoppingCart(**new** ECSCallback<ECSShoppingCart, Exception>() {  
 @Override  
 **public void** onResponse(ECSShoppingCart result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |

# **7.2 Add Product to Shopping Cart**

This API is used add product to the shopping cart. User needs to be logged in to call this API.

## ECSServiceProvider

**void** addProductToShoppingCart(ECSProduct product, ECSCallback<ECSShoppingCart, Exception> ecsCallback);

## This method has the following parameters:

* product : ECSProduct instance which needs to be added to the shopping cart
* ecsCallback : an instance of ECSCallback which includes ECSShoppingCart indicating

success and Exception in case of failure

## Sample

ecsServices.addProductToShoppingCart(product, **new** ECSCallback<ECSShoppingCart, Exception>() {  
 @Override  
 **public void** onResponse(ECSShoppingCart result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5056 | ECSCtnNotProvided | Please provide the CTN | You need to provide valid CTN to add to cart |
| 5004 | ECSCartError | No cart created yet |  |
| 5008 | ECSCartEntryError | The Product is not added in Cart |  |

# **7.3 Fetch Shopping Cart**

This API is used to get the shopping cart details for the user. User needs to be logged in to call this API.

## ECSServiceProvider

**void** fetchShoppingCart(ECSCallback<ECSShoppingCart,Exception> ecsCallback);

## This method has the following parameter:

* ecsCallback : an instance of ECSCallback which includes ECSShoppingCart indicating

success and Exception in case of failure

## Sample

ecsServices.fetchShoppingCart(**new** ECSCallback<ECSShoppingCart, Exception>() {  
 @Override  
 **public void** onResponse(ECSShoppingCart result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5004 | ECSCartError | No cart created yet |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server |  |

# **7.4 Update Shopping Cart**

This API is used to update the quantity of the product which is present in the shopping cart. User needs to be logged in to call this API. API can be used to **delete** the product by passing quantity as 0.

## ECSServiceProvider

**void** updateShoppingCart(**int** quantity, ECSEntries entriesEntity, ECSCallback<ECSShoppingCart, Exception> ecsCallback) ;

## This method has the following parameters:

* quantity : new quantity of the product to be updated
* entriesEntity : entry of the product to be updated/deleted in shopping cart
* ecsCallback : an instance of ECSCallback which includes ECSShoppingCart indicating

success and Exception in case of failure

## Sample

ecsServices.updateShoppingCart(count, entriesEntity, **new** ECSCallback<ECSShoppingCart, Exception>() {  
 @Override  
 **public void** onResponse(ECSShoppingCart result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5007 | ECSCommerceCartModificationError | Quantity must not be less than zero |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server |  |

# **Voucher**

# **8.1 Apply Voucher**

This API is used to apply promotional voucher for the shopping cart. User has to be logged in before calling this API.

## ECSServiceProvider

**void** applyVoucher(String voucherCode, ECSCallback<List<ECSVoucher>, Exception> ecsCallback);

## This method has the following parameters:

* voucherCode : promotional voucher code which has to be applied to cart
* ecsCallback : an instance of ECSCallback which includes list of applied vouchers

indicating success and Exception in case of failure

## Sample

ecsServices.applyVoucher(voucherId, **new** ECSCallback<List<ECSVoucher>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSVoucher> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5010 | ECSUnsupportedVoucherError | Sorry, this voucher can’t be redeemed |  |
| 5011 | ECSVoucherOperationError | Sorry, this voucher can’t be redeemed |  |

# **8.2 Fetch Applied Voucher**

This API is used to fetch the list of applied vouchers.

## ECSServiceProvider

**void** fetchAppliedVouchers(ECSCallback<List<ECSVoucher>, Exception> ecsCallback);

This method has the following parameter:

* ecsCallback : an instance of ECSCallback which includes list of applied vouchers indicating

success and Exception in case of failure

## Sample

ecsServices.fetchAppliedVouchers(**new** ECSCallback<List<ECSVoucher>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSVoucher> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5004 | ECSCartError | No cart created yet |  |

# **8.3 Remove Voucher**

This api is used to remove the applied voucher.

## ECSServiceProvider

**void** removeVoucher(String voucherCode, ECSCallback<List<ECSVoucher>, Exception> ecsCallback);

## This method has the following parameter:

* voucherCode : promotional voucher code which has to be removed from the cart
* ecsCallback : an instance of ECSCallback which includes list of applied vouchers

indicating success and Exception in case of failure

## Sample

**appInfra** = **new** AppInfra.Builder().build(getApplicationContext());

ECSServices ecsServices = **new** ECSServices( **"your propositionID"**, appInfra);

ecsServices.removeVoucher(voucherCode, **new** ECSCallback<List<ECSVoucher>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSVoucher> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
 });

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5004 | ECSCartError | No cart created yet |  |
| 5010 | ECSUnsupportedVoucherError | Sorry, this voucher can’t be redeemed |  |
| 5011 | ECSVoucherOperationError | Sorry, this voucher can’t be redeemed |  |

# **Delivery Mode**

# **9.1 Set Delivery Mode**

This API is used to set delivery mode for cart.

## ECSServiceProvider

**void** setDeliveryMode(ECSDeliveryMode deliveryModes, ECSCallback<Boolean, Exception> ecsCallback);

## This method has the following parameter:

* deliveryMode : ECSDeliveryMode object to be set as delivery mode
* ecsCallback : an instance of ECSCallback which contains TRUE if successfully set the

delivery mode, otherwise FALSE and Exception explaining any failures

## Sample

ecsServices. setDeliveryMode(deliveryMode, **new** ECSCallback<Boolean, Exception>(){  
 @Override  
 **public void** onResponse(Boolean result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |

# **9.2 Fetch Delivery Modes**

This API is used to fetch the list of delivery modes.

## ECSServiceProvider

**void** fetchDeliveryModes(ECSCallback<List<ECSDeliveryMode>, Exception> ecsCallback);

## This method has the following parameter:

* ecsCallback : an instance of ECSCallback which includes list of delivery modes supported for

the location indicating success and Exception in case of failure

## Sample

ecsServices.fetchDeliveryModes(**new** ECSCallback<List<ECSDeliveryMode>, Exception>() {  
 @Override  
 **public void** onResponse(ECSDeliveryMode result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |

# **Address**

# **10.1 Fetch Regions**

This api is used to fetch the list of regions for specific country.

## ECSServiceProvider

**void** fetchRegions(ECSCallback<List<ECSRegion>, Exception> ecsCallback);

## This method has the following parameter:

* ecsCallback : an instance of ECSCallback which includes list of ECSRegion objects indicating

success and Exception in case of failure

## Sample

ecsServices.fetchRegions(**new** ECSCallback<List<ECSRegion>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSRegion> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5059 | ECSCountryCodeNotGiven | Please provide valid country code |  |

# **10.2 Create Address**

This API is used to create new address for the user.

## ECSServiceProvider

**void** createAddress(ECSAddress address, ECSCallback<ECSAddress, Exception> ecsCallback);

## This method has the following parameter:

* address : ECSAddress instance contains new address details which has to be

created

* ecsCallback : an instance of ECSCallback which includes ECSAddress which has

been created indicating success and Exception in case of failure

## Sample

ecsServices.createAddress(address, **new** ECSCallback< ECSAddress, Exception>() {  
 @Override  
 **public void** onResponse(ECSAddress result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5059 | ECSCountryCodeNotGiven | Please provide valid country code |  |
| 5014 | ECSRegionisocode | Region selected is invalid |  |
| 5016 | ECSPostalCode | ZIP code selected is invalid |  |
| 5017 | ECSfirstName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5018 | ECSlastName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5019 | ECSphone1 | Phone number selected is invalid |  |
| 5015 | ECScountryisocode | Country selected is invalid |  |
| 5012 | ECSValidationError | Data validation failed |  |

# **10.3 Create and Fetch Address**

This api is used to create new address for the user and fetch the list of all saved addresses.

## ECSServiceProvider

**void** createAndFetchAddress(ECSAddress address, ECSCallback<List<ECSAddress>, Exception> ecsCallback);

## This method has the following parameter:

* address : an instance of ECSAddress which has to be added
* ecsCallback : which include list of ECSCallback instance representing saved address of

user indicating success and Exception in case of failure

## Sample

ecsServices.createAndFetchAddress(address, **new** ECSCallback<List<ECSAddress>, Exception>() {  
 @Override  
 **public void** onResponse(<List<ECSAddress> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5059 | ECSCountryCodeNotGiven | Please provide valid country code |  |
| 5014 | ECSRegionisocode | Region selected is invalid |  |
| 5016 | ECSPostalCode | ZIP code selected is invalid |  |
| 5017 | ECSfirstName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5018 | ECSlastName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5019 | ECSphone1 | Phone number selected is invalid |  |
| 5015 | ECScountryisocode | Country selected is invalid |  |
| 5012 | ECSValidationError | Data validation failed |  |

# **10.4 Fetch Saved Address**

This API is used to fetch the list of all saved addresses for the user.

## ECSServiceProvider

**void** fetchSavedAddresses(ECSCallback<List<ECSAddress>, Exception> ecsCallback);

## This method has the following parameter:

* ecsCallback : which includes list of ECSCallback instance representing saved address of user

indicating success and Exception in case of failure

## Sample

ecsServices.fetchSavedAddresses(**new** ECSCallback<List<ECSAddress>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSAddress> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server |  |

# **10.5 Set Delivery Address**

This api is used to set the delivery address of the user.

## ECSServiceProvider

**void** setDeliveryAddress(**boolean** isDefaultAddress, ECSAddress address, ECSCallback<Boolean, Exception> ecsCallback);

## This method has the following parameters:

* address : an instance of ECSAddress which has to be set as delivery address
* isDefaultAddress : Boolean which tells whether the address has to be made default address
* ecsCallback : an instance of ECSCallback which contains TRUE if successfully set the

delivery mode, otherwise FALSE and Exception in case of failures

## Sample

ecsServices. setDeliveryAddress(addresses, **new** ECSCallback<Boolean, Exception>() {  
 @Override  
 **public void** onResponse(Boolean result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **10.6 Set and Fetch Delivery Address**

This API is used to set the delivery address of the user and fetch the address.

## ECSServiceProvider

**void** setAndFetchDeliveryAddress(**boolean** isDefaultAddress, ECSAddress address, ECSCallback<List<ECSAddress>, Exception> ecsCallback);

## This method has the following parameters:

* address : an instance of ECSAddress which has to be set as delivery address
* isDefaultAddress : Boolean which tells whether the address has to be made default address
* ecsCallback : which includes list of ECSCallback instance representing saved address

of user indicating success and Error in case of failure

## Sample

ecsServices.setAndFetchDeliveryAddress(address,**new** ECSCallback<List<ECSAddress>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSAddress> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **10.7 Update Address**

This api is used to update the address of the user. This is used whenever the user edits the address.

## ECSServiceProvider

**void** updateAddress(ECSAddress address, ECSCallback<Boolean, Exception> ecsCallback);

## This method has the following parameter:

* address : an instance of ECSAddress which has to be set as delivery address
* ecsCallback : an instance of ECSCallback which contains TRUE if the address is

updated successfully, otherwise FALSE and Exception in case of any

failures

## Sample

ecsServices.updateAddress(**false**, addresses, **new** ECSCallback<Boolean, Exception>() {

@Override  
 **public void** onResponse(Boolean result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5059 | ECSCountryCodeNotGiven | Please provide valid country code |  |
| 5014 | ECSRegionisocode | Region selected is invalid |  |
| 5016 | ECSPostalCode | ZIP code selected is invalid |  |
| 5017 | ECSfirstName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5018 | ECSlastName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5019 | ECSphone1 | Phone number selected is invalid |  |
| 5015 | ECScountryisocode | Country selected is invalid |  |
| 5012 | ECSValidationError | Data validation failed |  |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **10.8 Update and Fetch Address**

This API is used to update the address of the user and fetch the list of addresses.

## ECSServiceProvider

**void** updateAndFetchAddress(ECSAddress address, ECSCallback<List<ECSAddress>, Exception> ecsCallback);

## This method has the following parameter:

* address : an instance of ECSAddress which has to be set as delivery address
* ecsCallback : which includes list of ECSCallback instance representing saved address

of user indicating success and Exception in case of failure

## Sample

ecsServices.updateAndFetchAddress(true, address**,new** ECSCallback<List<ECSAddress>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSAddress> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5059 | ECSCountryCodeNotGiven | Please provide valid country code |  |
| 5014 | ECSRegionisocode | Region selected is invalid |  |
| 5016 | ECSPostalCode | ZIP code selected is invalid |  |
| 5017 | ECSfirstName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5018 | ECSlastName | Please notice that the following characters are not allowed: \") ( \* ^ % $ # @ ! + , . ? / : ; { } [ ] ` ~ = |  |
| 5019 | ECSphone1 | Phone number selected is invalid |  |
| 5015 | ECScountryisocode | Country selected is invalid |  |
| 5012 | ECSValidationError | Data validation failed |  |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **10.9 Delete Address**

This api is used to delete the address of the user, based on the provided address object.

## ECSServiceProvider

**void** deleteAddress(ECSAddress address, ECSCallback<Boolean, Exception> ecsCallback);

## This method has the following parameter:

* address : an instance of ECSAddress which has to be deleted
* ecsCallback : an instance of ECSCallback which contains TRUE if the address is

deleted successfully, otherwise FALSE and an Error explaining any

failures

## Sample

ecsServices.deleteAddress(address,**new** ECSCallback<Boolean, Exception>() {  
 @Override  
 **public void** onResponse(Boolean result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **10.10 Delete and Fetch Address**

This API is used to delete the address of the user and fetch the list of addresses.

## ECSServiceProvider

**void** deleteAndFetchAddress(@NonNull ECSAddress address,@NonNull ECSCallback<List<ECSAddress>, Exception> ecsCallback);

## This method has the following parameter:

* address : an instance of ECSAddress which has to be deleted
* ecsCallback : include list of ECSCallback instance

representing saved address of user indicating success and Error in case of

failure

## Sample

ecsServices.deleteAndFetchAddress(address, **new** ECSCallback<List<ECSAddress>, Exception>() {

@Override  
 **public void** onResponse(List<ECSAddress> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **Retailers**

# **11.1 Fetch Retailers for CTN**

This api is used to fetch the list of available retailers for the selected product.

## ECSServiceProvider

**void** fetchRetailers(String ctn, ECSCallback<ECSRetailerList,Exception>ecsCallback);

## This method has the following parameters:

* Ctn : CTN of the product for which the retailer list has to be fetched
* ecsCallback : an instance of ECSCallback which includes instance of ECSRetailerList

which has list of retailers and Exception in case of any failure

## Sample

ecsServices. fetchRetailers(ctn, **new** ECSCallback<ECSRetailerList, Exception>()  
 @Override  
 **public void** onResponse(ECSRetailerList result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **11.2 Fetch Retailers for Product**

This API is used to fetch the list of available retailers for the selected product.

## ECSServiceProvider

**void** fetchRetailers(ECSProduct product, ECSCallback<ECSRetailerList,Exception> ecsCallback);

## This method has the following parameters:

* product : ECSProduct instance for which the retailers has be fetched
* ecsCallback : an instance of ECSCallback which includes instance of ECSRetailerList

which has list of retailers and Exception in case of any failure

## Sample

ecsServices.fetchRetailers(product, **new** ECSCallback<ECSRetailerList, Exception>(){  
 @Override  
 **public void** onResponse(ECSRetailerList result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **Payment**

# **12.1 Set Payment Details**

This API is used to set payment details for the purchase.

## ECSServiceProvider

**void** setPaymentDetails(String paymentDetailsId, ECSCallback<Boolean, Exception> ecsCallback);

## This method has the following parameters:

* paymentDetailsId : payment id for payment method has to be set for the order
* ecsCallback : an instance of ECSCallback which contains TRUE if the payment details

is set successfully, otherwise FALSE and Exception in case of any failures

## Sample

ecsServices.setPaymentDetails(paymentId, **new** ECSCallback<Boolean, Exception>(){  
 @Override  
 **public void** onResponse(Boolean result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5025 | ECSInvalidPaymentInfoError | Please provide valid payment details |  |

# **12.2 Fetch Payment Details**

This API is used to fetch list of existing payment methods.

## ECSServiceProvider

**void** fetchPaymentsDetails(ECSCallback<List<ECSPayment>, Exception> ecsCallback);

## This method has the following parameters:

* ecsCallback : an instance of ECSCallback which includes list of ECSPayment instances

which represent saved payments indicating success and Exception in case of failure

## Sample

ecsServices.fetchPaymentsDetails(**new** ECSCallback<List<ECSPayment>, Exception>() {  
 @Override  
 **public void** onResponse(List<ECSPayment> result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |

# **12.3 Make Payment**

This API is used to make the payment after entering all the payment details.

## ECSServiceProvider

**void** makePayment(ECSOrderDetail orderDetail, ECSAddress billingAddress, ECSCallback<ECSPaymentProvider, Exception> ecsCallback);

## This method has the following parameters:

* orderDetail : ECSOrderDetail instance which is the placed order for which the

payment has to be made

* billingAddress : ECSAddress instance which contains billing address for the order
* ecsCallback : an instance of ECSCallback which includes ECSPaymentProvider

instance indicating success and Exception in case of failure

## Sample

ecsServices.makePayment(orderDetail, billingAddress, **new** ECSCallback<ECSPaymentProvider, Exception>() {  
 @Override  
 **public void** onResponse(ECSPaymentProvider result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5060 | ECSorderIdNil | Please provide valid order |  |
| 5026 | ECSPaymentAuthorizationError | Payment authorization was not successful | This error occurs if there is any issue related to payment |

# **Order**

# **13.1 Submit Order**

This API is used to place the order. If the saved payment is selected, then you need to send the CVV of the saved payment.

## ECSServiceProvider

**void** submitOrder(String cvv, ECSCallback<ECSOrderDetail, Exception> ecsCallback);

* cvv : CVV of the saved card. This is optional should only be sent if the saved

payment is selected else this should be nil

* ecsCallback : an instance of ECSCallback which includes instance of ECSOrderDetail

which has complete details of placed order indicating success and

Exception in case of failure

## Sample

ecsServices.submitOrder(pSecurityCode, **new** ECSCallback<ECSOrderDetail, Exception>() {  
 @Override  
 **public void** onResponse(ECSOrderDetail result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5022 | ECSSessionCart | Please select your delivery address, delivery mode and payment method before submitting an order |  |

# **13.2 Fetch Order History**

This API is used to fetch the order history of the submitted orders, based on the pageNumber and pagSize.

## ECSServiceProvider

**void** fetchOrderHistory(**int** pageNumber, **int** pageSize, ECSCallback<ECSOrderHistory, Exception> ecsCallback);

## This method has the following parameters:

* pageNumber : page number for which the order history has to be fetched
* pageSize : total number of order histories per page
* ecsCallback : an instance of ECSCallback which contains list of ECSOrderHistory

indicating success and Exception explaining any failures.

## Sample

ecsServices.fetchOrderHistory(pageNo, 0, **new** ECSCallback<ECSOrderHistory, Exception>() {  
 @Override  
 **public void** onResponse(ECSOrderHistory result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |

# **13.3 Fetch Order Detail**

This API is used to fetch the details of order, based on the orderId.

## ECSServiceProvider

**void** fetchOrderDetail(String orderId, ECSCallback<ECSOrderDetail,Exception> ecsCallback);

This method has the following parameters:

* orderID : order id of the order for which the details has be fetched
* ecsCallback : an instance of ECSCallback which includes an instance of

ECSOrderDetail which has complete detail of the order indicating

success and Error in case of failure

## Sample

ecsServices.fetchOrderDetail(orderNumber, **new** ECSCallback<ECSOrderDetail, Exception>() {  
 @Override  
 **public void** onResponse(ECSOrderDetail result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5060 | ECSorderIdNil | Please provide valid order |  |

# **13.4 Fetch Order Detail**

This API is used to fetch the details of given orders, based on orderDetail object.

## ECSServiceProvider

**void** fetchOrderDetail(ECSOrderDetail orderDetail,ECSCallback<ECSOrderDetail,Exception> ecsCallback);

This method has the following parameters:

* orderDetail : ECSOrderDetail instance for which the complete details has to fetched
* ecsCallback : an instance of ECSCallback which includes an instance of

ECSOrderDetail which has complete detail of the order indicating

success and Error in case of failure

## Sample

ecsServices.fetchOrderDetail(orderDetail, **new** ECSCallback<ECSOrderDetail, Exception>() {  
 @Override  
 **public void** onResponse(ECSOrderDetail result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5060 | ECSorderIdNil | Please provide valid order |  |

# **13.5 Fetch Order Detail**

This API is used to fetch the details of given orders, using Order objects.

## ECSServiceProvider

**void** fetchOrderDetail(ECSOrders orders, ECSCallback<ECSOrders, Exception> ecsCallback);

This method has the following parameters:

* Orders : ECSOrders instance for which the complete details has to fetched
* ecsCallback : an instance of ECSCallback which includes an instance of

ECSOrderDetail which has complete detail of the order indicating

success and Error in case of failure

## Sample

ecsServices.fetchOrderDetail(orders, **new** ECSCallback<ECSOrders, Exception>() {  
 @Override  
 **public void** onResponse(ECSOrders result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |
| 5060 | ECSorderIdNil | Please provide valid order |  |

# **Fetch User Profile**

This API is used to fetch the user details.

## ECSServiceProvider

**void** fetchUserProfile(ECSCallback<ECSUserProfile,Exception> ecsCallback);

This method has the following parameters:

* ecsCallback : an instance of ECSCallback which includes ECSUserProfile instance

indicating success and Exception in case of failure

## Sample

ecsServices.fetchUserProfile(**new** ECSCallback<ECSUserProfile, Exception>() {  
 @Override  
 **public void** onResponse(ECSUserProfile result) {

}  
  
 @Override  
 **public void** onFailure(Exception error, ECSError ecsError) {  
  
 }  
});

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale | This error might occur if the service discovery environment is down as we get locale from service discovery |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available | This error occurs if the Philips shop is not configured for specific country |
| 5050 | ECSBaseURLNotFound | Base URL not found | This error occurs if the base URL is not found in service discovery for specific country |

**Common Errors**

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5002 | ECSUnsupported\_grant\_type | We have encountered technical glitch (Unsupported grant type). Please re-login |  |
| 50057 | ECSOAuthNotCalled | User authorization is required for this action | This error occurs if the OAuth is not called for the APIs which needs login |
| 5999 | ECSSomethingWentWrong | We have encountered technical glitch. Please try after some time | This is generic error which occurs if any uncaught exception comes from server. |
| 5009 | ECSInvalidTokenError | We have encountered technical glitch (Invalid access token) |  |

Exception

Exception (contains localized error message) and ECSError which contains error code and error type.

----------------------------------------------------------------End of Document--------------------------------------------------------------