

Consumer Mobile SDK

Reference Guide

Version 0.1

# Document Configuration Management

## Document Identification

|  |  |
| --- | --- |
| **Name of Project** |  |
| **Name of Report** | Consumer Mobile SDK Reference Guide |
| **Version** | 0.1 (draft) |
| **File Name** |  |
| **Sensitivity Classification** | Confidential |

## Document Change Control

|  |  |  |
| --- | --- | --- |
| Version | Date Released | Change Notice |
| 0.1 |  | Initial (draft) for review and feedback |
|  |  |  |

Table of Contents

[Document Configuration Management 2](#_Toc20413504)

[Document Identification 2](#_Toc20413505)

[Document Change Control 2](#_Toc20413506)

[1 Introduction 7](#_Toc20413507)

[7](#_Toc20413508)

[2 Initialize SDK 8](#_Toc20413509)

[This method has the following parameters: 8](#_Toc20413510)

[Sample 8](#_Toc20413511)

[3 Set Proposition ID 8](#_Toc20413512)

[Sample 8](#_Toc20413513)

[4 Configuration 8](#_Toc20413514)

[Configure ECS 8](#_Toc20413515)

[This method has the following parameter: 9](#_Toc20413516)

[Sample 9](#_Toc20413517)

[Errors 9](#_Toc20413518)

[This method has the following parameter: 9](#_Toc20413519)

[Sample 9](#_Toc20413520)

[Errors 10](#_Toc20413521)

[5 Hybris Authentication 10](#_Toc20413522)

[Hybris OAuth Authentication 10](#_Toc20413523)

[This method has the following parameters: 10](#_Toc20413524)

[Sample 10](#_Toc20413525)

[Errors 11](#_Toc20413526)

[This method has the following parameters: 11](#_Toc20413527)

[Sample 11](#_Toc20413528)

[Errors 12](#_Toc20413529)

[6 Product 12](#_Toc20413530)

[Fetch Product List 12](#_Toc20413531)

[This method has the following parameters: 12](#_Toc20413532)

[Sample 12](#_Toc20413533)

[Errors 13](#_Toc20413534)

[This method has the following parameters: 14](#_Toc20413535)

[Sample 14](#_Toc20413536)

[Errors 14](#_Toc20413537)

[This method has the following parameters: 15](#_Toc20413538)

[Errors 15](#_Toc20413539)

[This method has the following parameters: 16](#_Toc20413540)

[Sample 16](#_Toc20413541)

[Errors 16](#_Toc20413542)

[7 Shopping Cart 17](#_Toc20413543)

[Create Shopping Cart 17](#_Toc20413544)

[This method has the following parameters: 17](#_Toc20413545)

[ completionHandler : ECSShoppingCartCompletion which includes ECSShoppingCart 17](#_Toc20413546)

[Sample 17](#_Toc20413547)

[Errors 17](#_Toc20413548)

[This method has the following parameters: 18](#_Toc20413549)

[Sample 18](#_Toc20413550)

[Errors 18](#_Toc20413551)

[This method has the following parameter: 18](#_Toc20413552)

[Sample 19](#_Toc20413553)

[Errors 19](#_Toc20413554)

[Update Shopping Cart 19](#_Toc20413555)

[This method has the following parameters: 19](#_Toc20413556)

[Sample 19](#_Toc20413557)

[Errors 20](#_Toc20413558)

[8 Voucher 21](#_Toc20413559)

[Apply Voucher 21](#_Toc20413560)

[This method has the following parameters: 21](#_Toc20413561)

[Sample 21](#_Toc20413562)

[Errors 21](#_Toc20413563)

[This method has the following parameter: 22](#_Toc20413564)

[Sample 22](#_Toc20413565)

[Errors 22](#_Toc20413566)

[This method has the following parameter: 22](#_Toc20413567)

[Sample 22](#_Toc20413568)

[Errors 23](#_Toc20413569)

[9 Delivery Mode 24](#_Toc20413570)

[This method has the following parameter: 24](#_Toc20413571)

[Sample 24](#_Toc20413572)

[Errors 24](#_Toc20413573)

[This method has the following parameter: 25](#_Toc20413574)

[Sample 25](#_Toc20413575)

[Errors 25](#_Toc20413576)

[10 Address 26](#_Toc20413577)

[This method has the following parameter: 26](#_Toc20413578)

[Sample 26](#_Toc20413579)

[Errors 26](#_Toc20413580)

[This method has the following parameter: 27](#_Toc20413581)

[Sample 27](#_Toc20413582)

[Errors 27](#_Toc20413583)

[This method has the following parameter: 28](#_Toc20413584)

[Sample 28](#_Toc20413585)

[Errors 28](#_Toc20413586)

[This method has the following parameter: 29](#_Toc20413587)

[Sample 29](#_Toc20413588)

[Errors 29](#_Toc20413589)

[This method has the following parameters: 29](#_Toc20413590)

[Sample 30](#_Toc20413591)

[Errors 30](#_Toc20413592)

[This method has the following parameter: 30](#_Toc20413593)

[Sample 31](#_Toc20413594)

[Errors 31](#_Toc20413595)

[This method has the following parameter: 31](#_Toc20413596)

[Sample 31](#_Toc20413597)

[Errors 32](#_Toc20413598)

[This method has the following parameter: 32](#_Toc20413599)

[Sample 33](#_Toc20413600)

[Errors 33](#_Toc20413601)

[This method has the following parameter: 34](#_Toc20413602)

[Sample 34](#_Toc20413603)

[Errors 34](#_Toc20413604)

[This method has the following parameter: 34](#_Toc20413605)

[Sample 35](#_Toc20413606)

[Errors 35](#_Toc20413607)

[11 Retailers 36](#_Toc20413608)

[This method has the following parameters: 36](#_Toc20413609)

[Sample 36](#_Toc20413610)

[Errors 36](#_Toc20413611)

[This method has the following parameters: 36](#_Toc20413612)

[Sample 37](#_Toc20413613)

[Errors 37](#_Toc20413614)

[12 Payment 38](#_Toc20413615)

[This method has the following parameters: 38](#_Toc20413616)

[Sample 38](#_Toc20413617)

[Errors 38](#_Toc20413618)

[This method has the following parameters: 38](#_Toc20413619)

[Sample 39](#_Toc20413620)

[Errors 39](#_Toc20413621)

[This method has the following parameters: 39](#_Toc20413622)

[Sample 39](#_Toc20413623)

[Errors 39](#_Toc20413624)

[13 Order 40](#_Toc20413625)

[This method has the following parameters: 40](#_Toc20413626)

[Sample 40](#_Toc20413627)

[Errors 40](#_Toc20413628)

[This method has the following parameters: 41](#_Toc20413629)

[Sample 41](#_Toc20413630)

[Errors 41](#_Toc20413631)

[Sample 42](#_Toc20413632)

[Errors 42](#_Toc20413633)

[Sample 42](#_Toc20413634)

[Errors 42](#_Toc20413635)

[Sample 43](#_Toc20413636)

[Errors 43](#_Toc20413637)

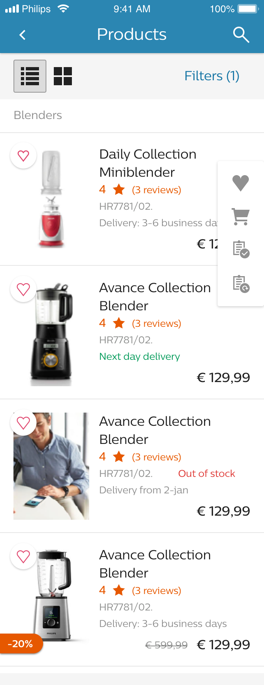
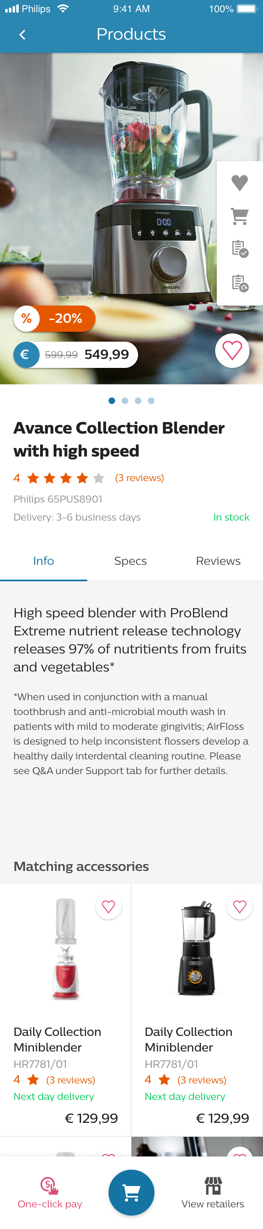
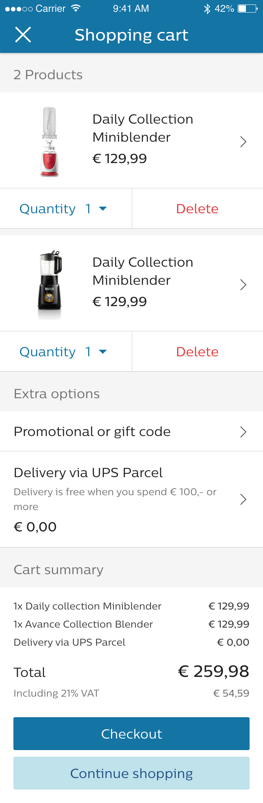
[14 Fetch User Profile 44](#_Toc20413638)

[Sample 44](#_Toc20413639)

[Errors 44](#_Toc20413640)

# **Introduction**

# 



The e-commerce 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 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 to keep a single ECSService instance to call all the APIs. Call the following API to initialize the SDK.

????????????????????????????????????????????????????????????

**public** **init**(propositionId: String?, appInfra: AIAppInfra)

## 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

**let** escService = ECSServices(propositionId: "Your proposition id", appInfra: appInfra)

# **Set Proposition ID**

This variable is a unique store identification string, specific to business and country.

**public** **var** propositionId: String?

## Sample

escService.propositionId = "Your proposition id"

# **Configuration**

# **Configure ECS**

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.

**public** **func** configureECS(completionHandler: **@escaping** ECSConfigCompletion)

## This method has the following parameter:

* completionHandler : ECSConfigCompletion which contains TRUE if successfully configured,

otherwise FALSE and an Error explaining any failures.

## Sample

escService.configureECS { (isHybrisConfigured, error) **in**

}

## 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 |  |

**Configure ECS to Fetch 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.

**public** **func** configureECSWithConfiguration(completionHandler: **@escaping** ECSGetConfigCompletion)

## This method has the following parameter:

* completionHandler : ECSGetConfigCompletion which contains ECSConfig indicating success

and Error explaining any failures.

## Sample

escService.configureECSWithConfiguration {(config, error) **in**

}

## 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 |  |

# **Hybris Authentication**

# **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.

**public** **func** hybrisOAuthAuthenticationWith(hybrisOAuthData: ECSOAuthProvider, completionHandler: **@escaping** ECSHybrisOAuthCompletion)

## This method has the following parameters:

* hybrisOAuthData : ECSOAuthProvider instance which provides information

needed for OAuth

* completionHandler : ECSHybrisOAuthCompletion which contains ECSOAuthData

indicating success and Error explaining any failures

## Sample

**let** oauthProvider = ECSOAuthProvider(token: “Janrain access token”)

ecsService?.hybrisOAuthAuthenticationWith(hybrisOAuthData: oauthProvider, completionHandler: { (hybrisOAuthData, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5058 | ECSOAuthDetailError | Please provide complete OAuth details |  |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5009 | ECSInvalidTokenError | We have encountered technical glitch (Invalid access token) |  |
| 5000 | ECSinvalid\_grant | We have encountered technical glitch (Invalid Janrain Token). Please re-login |  |
| 5001 | ECSinvalid\_client | We have encountered technical glitch (Bad client credentials). Please re-login |  |

**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.

**public** **func** hybrisRefreshOAuthWith(hybrisOAuthData: ECSOAuthProvider,

completionHandler: **@escaping** ECSHybrisOAuthCompletion)

## This method has the following parameters:

* hybrisOAuthData : ECSOAuthProvider instance which provides information

needed for OAuth

* completionHandler : ECSHybrisOAuthCompletion which contains ECSOAuthData indicating

success and Error explaining any failures

## Sample

**let** oauthProvider = ECSOAuthProvider(token: “Hybris Refresh Token”)

ecsService?.hybrisRefreshOAuthWith (hybrisOAuthData: oauthProvider, completionHandler: { (hybrisOAuthData, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5058 | ECSOAuthDetailError | Please provide complete OAuth details |  |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5009 | ECSInvalidTokenError | We have encountered technical glitch (Invalid access token) |  |
| 5000 | ECSinvalid\_grant | We have encountered technical glitch (Invalid Janrain Token). Please re-login |  |
| 5001 | ECSinvalid\_client | We have encountered technical glitch (Bad client credentials). Please re-login |  |

# **Product**

# **Fetch Product List**

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

**public** **func** fetchProducts(pageSize: Int? = **nil**, currentPage: Int = 0,

completionHandler: **@escaping** ECSProductsFetchCompletion)

## This method has the following parameters:

* currentPage : page number for which the products has to be fetched
* pageSize : total number of products per page
* completionHandler : ECSProductsFetchCompletion contains list of ECSProducts indicating

success and Error explaining any failures

## Sample

ecsService?.fetchProducts(currentPage: 1, completionHandler: { (hybrisProducts, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server. |  |

**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.

**public** **func** fetchProductFor(ctn: String,

completionHandler: **@escaping** ECSProductFetchCompletion)

## This method has the following parameters:

* ctn : CTN of the product
* completionHandler : ECSProductFetchCompletion which contains ECSProduct indicating

success and Exception explaining any failures

## Sample

ecsService?.fetchProductFor(ctn: "ctn1",

completionHandler: {(product, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server. |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

**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. This API can be called even if the user is not logged in.

**public** **func** fetchProductSummariesFor(ctns: [String],

completionHandler: **@escaping** ECSProductSummaryFetchCompletion)

## This method has the following parameters:

* ctns : array of CTNs
* completionHandler : ECSProductSummaryFetchCompletion contains list of ECSProduct

indicating success and invalid CTN list and Error in case of failure

**var** ctnList = ["ctn1", "ctn2", "ctn3"]

ecsService?.fetchProductSummariesFor(ctns: ctnList, completionHandler: { (productList, invalidCTNs, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server. |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

**Fetch Product Details**

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

**public** **func** fetchProductDetailsFor(product: ECSProduct,

completionHandler: **@escaping** ECSProductDetailsFetchCompletion)

## This method has the following parameters:

* product : an instance of ECSProduct for which detail needs to be fetched
* completionHandler : ECSProductDetailsFetchCompletion which includes ECSProduct

containing details indicating success and Exception in case of failure

## Sample

ecsService?.fetchProductDetailsFor(product: “Instance of ECSProduct”, completionHandler: { (product, error) **in**

})

## 

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **Shopping Cart**

# **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.

**public** **func** createShoppingCart(completionHandler: **@escaping** ECSShoppingCartCompletion)

## This method has the following parameters:

## completionHandler : ECSShoppingCartCompletion which includes ECSShoppingCart

indicating success and Exception in case of failure

## Sample

ecsService.createShoppingCart { (cart, error) **in**

}

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |

**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.

**public** **func** addProductToShoppingCart(product: ECSProduct,

completionHandler: **@escaping** ECSShoppingCartCompletion)

## This method has the following parameters:

* product : ECSProduct instance which needs to be added to the shopping cart
* completionHandler : ECSShoppingCartCompletion includes ECSShoppingCart indicating

success and Error in case of failure

## Sample

ecsService?.addProductToShoppingCart(product: “Instance of ECSProduct”, completionHandler: { [weak **self**](shoppingCart, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |
| 5004 | ECSCartError | No cart created yet |  |
| 5008 | ECSCartEntryError | The Product is not added in Cart |  |

**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.

**public** **func** fetchShoppingCart(completionHandler: **@escaping** ECSShoppingCartCompletion)

## This method has the following parameter:

* completionHandler : ECSShoppingCartCompletion includes ECSShoppingCart indicating

success and Error in case of failure

## Sample

ecsService?.fetchShoppingCart(completionHandler: {[weak **self**] (shoppingCart, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5004 | ECSCartError | No cart created yet |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server |  |

# **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. This API can be used to **delete** the product by passing quantity as 0

**public** **func** updateShoppingCart(quantity: Int, cartEntry: ECSEntry,

completionHandler: **@escaping** ECSShoppingCartCompletion)

## This method has the following parameters:

* quantity : new quantity of the product to be updated
* cartEntry : entry of the product to be updated/deleted in shopping cart
* completionHandler : ECSShoppingCartCompletion includes ECSShoppingCart indicating

success and Error in case of failure

## Sample

ecsService?.updateShoppingCart(quantity: "new quantity",

cartEntry: "product entry", completionHandler: { (shoppingCart, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5007 | ECSCommerceCartModificationError | Quantity must not be less than zero |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server |  |

# **Voucher**

# **Apply Voucher**

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

**public** **func** applyVoucher(voucherID: String,

completionHandler: **@escaping** ECSVoucherListCompletion)

## This method has the following parameters:

* voucherID : promotional voucher code which has to be applied to cart
* completionHandler : ECSVoucherListCompletion includes list of applied vouchers indicating

success and Error in case of failure

## Sample

ecsService?.applyVoucher(voucherID: “voucherCode”, completionHandler: {[weak **self**] (voucherList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5010 | ECSUnsupportedVoucherError | Sorry, this voucher can’t be redeemed |  |
| 5011 | ECSVoucherOperationError | Sorry, this voucher can’t be redeemed |  |

**Fetch Applied Voucher**

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

**public** **func** fetchAppliedVouchers(completionHandler: **@escaping** ECSVoucherListCompletion)

This method has the following parameter:

* completionHandler : ECSVoucherListCompletion includes list of applied vouchers indicating

success and Error in case of failure

## Sample

ecsService?.fetchAppliedVouchers(completionHandler: {[weak **self**] (voucherList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5004 | ECSCartError | No cart created yet |  |

**Remove Voucher**

This API is used to remove the applied voucher.

**public** **func** removeVoucher(voucherID: String,

completionHandler: **@escaping** ECSVoucherListCompletion)

## This method has the following parameter:

* voucherID : promotional voucher code which has to be removed from the cart
* completionHandler : ECSVoucherListCompletion includes list of applied vouchers indicating

success and Error in case of failure

## Sample

ecsService?.removeVoucher(voucherID: voucherCode,

completionHandler: {[weak **self**] (voucherList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 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**

**Set Delivery Mode**

This API is used to set delivery mode for cart.

**public** **func** setDeliveryMode(deliveryMode: ECSDeliveryMode,

completionHandler: **@escaping** ECSCompletion)

## This method has the following parameter:

* deliveryMode : ECSDeliveryMode object to be set as delivery mode
* completionHandler : ECSCompletion which contains TRUE if successfully set the delivery

mode, otherwise FALSE and an Error explaining any failures

## Sample

ecsService?.setDeliveryMode(deliveryMode: “selected delivery mode”,

completionHandler: {[weak **self**] (success, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |

**Fetch Delivery Modes**

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

**public** **func** fetchDeliveryModes(completionHandler: **@escaping** ECSDeliveryModesCompletion)

## This method has the following parameter:

* completionHandler : ECSDeliveryModesCompletion includes list of delivery modes

supported for the location indicating success and Error in case of failure

## Sample

ecsService?.fetchDeliveryModes(completionHandler: { [weak **self**] (deliveryModes, error) **in**

})

## 

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |

# **Address**

**Fetch Regions**

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

**public** **func** fetchRegionsFor(countryISO: String,

completionHandler: **@escaping** ECSRegionsCompletion)

## This method has the following parameter:

* countryISO : ISO code of the country for which the regions has to be fetched
* completionHandler : ECSRegionsCompletion includes list of ECSRegion objects indicating

success and Error in case of failure

## Sample

ecsService?.fetchRegionsFor(countryISO: countryCode,

completionHandler: {[weak **self**] (regionList, error) **in**

})

## 

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5059 | ECSCountryCodeNotGiven | Please provide valid country code |  |

**Create Address**

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

**public** **func** createAddress(address: ECSAddress,

completionHandler:**@escaping** ECSAddressCompletion)

## This method has the following parameter:

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

created

* completionHandler : ECSAddressCompletion includes an instance of ECSAddress which has

been created indicating success and Error in case of failure

## Sample

ecsService?.createAddress(address: “instance of ECSAddress”,

completionHandler: {[weak **self**] (address, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 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 |  |

**Create and Fetch Address**

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

**public** **func** createAddressWith(address: ECSAddress, completionHandler:**@escaping** ECSAddressListCompletion)

## This method has the following parameter:

* address : an instance of ECSAddress which has to be added
* completionHandler : ECSAddressListCompletion include list of ECSCallback instance

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

failure

## Sample

ecsService?.createAddressWith(address: “instance of ECSAddress”,

completionHandler: {(addressList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 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 |  |

**Fetch Saved Address**

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

**public** **func** fetchSavedAddresses(completionHandler: **@escaping** ECSAddressListCompletion)

## This method has the following parameter:

* completionHandler : ECSAddressListCompletion include list of ECSCallback instance

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

failure

## Sample

ecsService?.fetchSavedAddresses(completionHandler: {[weak **self**] (address, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5003 | ECSNoSuchElementError | An error occurred while fetching the data from the server |  |

**Set Delivery Address**

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

**public** **func** setDeliveryAddress(deliveryAddress: ECSAddress,

isDefaultAddress: Bool = **true**,

completionHandler: **@escaping** ECSCompletion)

## This method has the following parameters:

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

mode, otherwise FALSE and an Error explaining any failures

## Sample

ecsService?.setDeliveryAddress(deliveryAddress: “instance of ECSAddress”,

completionHandler: {[weak **self**] (success, error) **in**

})

## 

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

**Set delivery address and fetch saved address**

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

**public** **func** setDeliveryAddress(address: ECSAddress,

isDefaultAddress: Bool = **true**,

completionHandler: **@escaping** ECSAddressListCompletion)

## This method has the following parameter:

* address : an instance of ECSAddress which has to be set as delivery address
* isDefaultAddress : Boolean which tells whether the address has be to made default address
* completionHandler : ECSAddressListCompletion include list of ECSCallback instance

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

## Sample

ecsService?.setDeliveryAddress(address: “instance of ECSAddress”,

completionHandler: {[weak **self**](addressList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5013 | ECSUnsupportedDeliveryModeError | Selected delivery mode is not supported |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

**Update Address**

This API is used to update the address of the user.

**public** **func** updateAddress(address: ECSAddress,

completionHandler:**@escaping** ECSCompletion)

## This method has the following parameter:

* address : an instance of ECSAddress which has to be set as delivery address
* completionHandler : ECSCompletion which contains TRUE if the address is updated

successfully, otherwise FALSE and an Error explaining any failures

## Sample

ecsService?.updateAddress(address: “instance of ECSAddress”,

completionHandler: {(success, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 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 |  |

**Update and Fetch Address**

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

**public** **func** updateAddressWith(address: ECSAddress,

completionHandler:**@escaping** ECSAddressListCompletion)

## This method has the following parameter:

* address : an instance of ECSAddress which has to be set as delivery address
* completionHandler : ECSAddressListCompletion include list of ECSCallback instance

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

failure

## Sample

ecsService?.updateAddressWith(address: newAddress,

completionHandler: {[weak **self**](addressList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 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 |  |

**Delete Address**

This API is used to delete the address of the user.

**public** **func** deleteAddress(savedAddress: ECSAddress,

completionHandler: **@escaping** ECSCompletion)

## This method has the following parameter:

* savedAddress : an instance of ECSAddress which has to be deleted
* completionHandler : ECSCompletion which contains TRUE if the address is deleted

successfully, otherwise FALSE and an Error explaining any failures

## Sample

ecsService?.deleteAddress(savedAddress: “instance of ECSAddress”,

completionHandler: {[weak **self**](success, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

**Delete and Fetch Address**

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

**public** **func** deleteAddress(address: ECSAddress,

completionHandler: **@escaping** ECSAddressListCompletion)

## This method has the following parameter:

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

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

failure

## Sample

ecsService?.deleteAddress(address: “instance of ECSAddress”,

completionHandler: {[weak **self**] (addressList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5021 | ECSaddressId | Given address doesn't exist or belong to another user |  |

# **Retailers**

**Fetch Retailers for CTN**

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

**public** **func** fetchRetailerDetailsFor(productCtn: String,

completionHandler: **@escaping** ECSRetailerListCompletion)

## This method has the following parameters:

* productCtn : CTN of the product for which the retailer list has to be fetched
* completionHandler : ECSRetailerListCompletion includes instance of ECSRetailerList

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

## Sample

ecsService?.fetchRetailerDetailsFor(productCtn: "product CTN",

completionHandler: { (retailers, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

**Fetch Retailers for Product**

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

**public** **func** fetchRetailersFor(product: ECSProduct,

completionHandler: **@escaping** ECSRetailerListCompletion)

## This method has the following parameters:

* product : ECSProduct instance for which the retailers has be fetched
* completionHandler : ECSRetailerListCompletion includes instance of ECSRetailerList

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

## Sample

ecsService?.fetchRetailersFor(product: “instance of ECSAddress”,

completionHandler: { (retailers, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5056 | ECSCtnNotProvided | Please provide the CTN |  |

# **Payment**

**Fetch Payment Details**

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

**public** **func** fetchPaymentDetails(completionHandler: **@escaping** ECSFetchPaymentCompletion)

## This method has the following parameters:

* completionHandler : ECSFetchPaymentCompletion includes list of ECSPayment instances

which represent saved payments indicating success and Error in case of

failure

## Sample

ecsService?.fetchPaymentDetails(completionHandler: {[weak **self**] (paymentList, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |

**Set Payment Details**

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

**public** **func** setPaymentDetail(paymentDetail: ECSPayment,

completionHandler: **@escaping** ECSCompletion)

## This method has the following parameters:

* paymentDetail : ECSPayment instance which has to be set as payment method for the

order

* completionHandler : ECSCompletion which contains TRUE if the payment details is set

successfully, otherwise FALSE and an Error explaining any failures

## Sample

ecsService?.setPaymentDetail(paymentDetail: paymentInfo,

completionHandler: { [weak **self**](result, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5025 | ECSInvalidPaymentInfoError | Please provide valid payment details |  |

**Make Payment**

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

**public** **func** makePaymentFor(order: ECSOrderDetail,

billingAddress: ECSAddress,

completionHandler: **@escaping** ECSMakePaymentCompletion)

## This method has the following parameters:

* order : 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
* completionHandler : ECSMakePaymentCompletion includes ECSPaymentProvider instance

indicating success and Error in case of failure

## Sample

ecsService?.makePaymentFor(order: inOrder,

billingAddress: billingAddress,

completionHandler: {[weak **self**] (paymentProvider, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5060 | ECSorderIdNil | Please provide valid order |  |
| 5026 | ECSPaymentAuthorizationError | Payment authorization was not successful |  |

# **Order**

**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.

**public** **func** submitOrder(cvvCode: String? = **nil**,

completionHandler: **@escaping** ECSubmitOrderCompletion)

## This method has the following parameters:

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

payment is selected else this should be nil

* completionHandler : ECSubmitOrderCompletion includes instance of ECSOrderDetail

which has complete details of placed order indicating success and Error

in case of failure

## Sample

ecsService?.submitOrder(cvvCode:**”CVV code”**,

completionHandler: {[weak **self**] (orderDetail, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5022 | ECSSessionCart | Please select your delivery address, delivery mode and payment method before submitting an order |  |

**Fetch order history**

This API is used to fetch the order history of the submitted orders.

**public** **func** fetchOrderHistory(pageSize: Int = 20,

currentPage: Int = 0,

completionHandler: **@escaping** ECSFetchOrderHistoryCompletion)

## This method has the following parameters:

* currentPage : page number for which the order history has to be fetched
* pageSize : total number of order histories per page
* completionHandler : ECSFetchOrderHistoryCompletion contains list of

ECSOrderHistory indicating success and Error explaining any

failures.

## Sample

ecsService?.fetchOrderHistory(currentPage: 1,

completionHandler: {[weak **self**](orderHistory, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |

**Fetch Order Detail**

This API is used to fetch the details of given orders.

**public** **func** fetchOrderDetailsFor(orderID: String,

completionHandler: **@escaping** ECSFetchOrderCompletion)

This method has the following parameters:

* orderID : order id of the order for which the details has be fetched
* completionHandler : ECSFetchOrderCompletion includes an instance of ECSOrderDetail

which has complete detail of the order indicating success and Error in

case of failure

## Sample

ecsService?.fetchOrderDetailsFor(orderID: "order id",

completionHandler: {[weak **self**] (orderDetail, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5060 | ECSorderIdNil | Please provide valid order |  |

**Fetch Order Detail**

This API is used to fetch the details of given orders.

**public** **func** fetchOrderDetailsFor(orderDetail: ECSOrderDetail,

completionHandler: **@escaping** ECSFetchOrderCompletion)

This method has the following parameters:

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

which has complete detail of the order indicating success and Error in

case of failure

## Sample

ecsService?.fetchOrderDetailsFor(orderDetail: “instance of ECSOrderDetail”,

completionHandler: {[weak **self**](orderDetail, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5060 | ECSorderIdNil | Please provide valid order |  |

**Fetch Order Detail**

This API is used to fetch the details of given orders.

**public** **func** fetchOrderDetailsFor(order: ECSOrder,

completionHandler: **@escaping** ECSFetchOrderDetailsCompletion)

This method has the following parameters:

* order : ECSOrder instance for which the details has to be fetched
* completionHandler : ECSFetchOrderCompletion includes an instance of ECSOrderDetail

which has complete detail of the order indicating success and Error in

case of failure

## Sample

ecsService?.fetchOrderDetailsFor(order: orderModel,

completionHandler: {[weak **self**] (order, error) **in**

})

## 

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |
| 5060 | ECSorderIdNil | Please provide valid order |  |

# **Fetch User Profile**

This API is used to fetch the user details.

**public** **func** fetchUserProfile(completionHandler: **@escaping** ECSUserDetailsCompletion)

This method has the following parameters:

* completionHandler : ECSUserDetailsCompletion includes ECSUserProfile instance

indicating success and Error in case of failure

## Sample

ecsService?.fetchUserProfile(completionHandler: { (userProfile, error) **in**

})

## Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5052 | ECSLocaleNotFound | Please set the locale |  |
| 5054 | ECSSiteIdNotFound | Philips shop is not available for the selected country, only retailer mode is available |  |
| 5050 | ECSBaseURLNotFound | Base URL not found |  |

Common Errors

|  |  |  |  |
| --- | --- | --- | --- |
| Error code | Error type | Error message | Description |
| 5000 | ECSinvalid\_grant | We have encountered technical glitch (Invalid Janrain Token). Please re-login |  |
| 5001 | ECSinvalid\_client | We have encountered technical glitch (Bad client credentials). Please re-login |  |
| 5002 | ECSUnsupported\_grant\_type | We have encountered technical glitch (Unsupported grant type). Please re-login |  |
| 50057 | ECSOAuthNotCalled | User authorization is required for this action |  |
| 5999 | ECSSomethingWentWrong | "We have encountered technical glitch. Please try after some time" |  |