

DESIGN DOCUMENTATION

“Release 1.1.32”

Ready Wireless – Client API

Prepared by: **IT Department**

*Date: 3/11*/2014

Contents

[Document Information and Approvals 8](#_Toc382296856)

[Version Control 8](#_Toc382296857)

[Approvals 12](#_Toc382296858)

[Overview 14](#_Toc382296859)

[Purpose 14](#_Toc382296860)

[Connectivity 14](#_Toc382296861)

[Web Service Technology 14](#_Toc382296862)

[Base Request Layout 14](#_Toc382296863)

[Base Response Layout 14](#_Toc382296864)

[Security 15](#_Toc382296865)

[Assumptions 15](#_Toc382296866)

[Restrictions 15](#_Toc382296867)

[Return Codes 15](#_Toc382296868)

[API Reference 18](#_Toc382296869)

[Order API Group 18](#_Toc382296870)

[Build New Order 18](#_Toc382296871)

[Add Order Device 18](#_Toc382296872)

[Add Order Pin Product 19](#_Toc382296873)

[Add Order AirTime Product 20](#_Toc382296874)

[Add Order Accessory Product 21](#_Toc382296875)

[Add Order Extended Product 21](#_Toc382296876)

[Add Order Fee Product 22](#_Toc382296877)

[Add Order Credit Product 22](#_Toc382296878)

[Add Order Promo Product 23](#_Toc382296879)

[Add Order Location 23](#_Toc382296880)

[Remove Order Line Item 24](#_Toc382296881)

[Set Order Device 24](#_Toc382296882)

[Set Order Payment Account 25](#_Toc382296883)

[Place Order 25](#_Toc382296884)

[CancelOrder 26](#_Toc382296885)

[Release Order 26](#_Toc382296886)

[Get Order Quote 26](#_Toc382296887)

[Get Order Adhoc Quote 27](#_Toc382296888)

[Get Orders Lookup 28](#_Toc382296889)

[Build Change Order 30](#_Toc382296890)

[Change Order Airtime Product 31](#_Toc382296891)

[Change Order Accessory Product 32](#_Toc382296892)

[Change Order Account Billing Mode 33](#_Toc382296893)

[Build Cancel Order 35](#_Toc382296894)

[Cancel Order Airtime Product 35](#_Toc382296895)

[Cancel Order Device 36](#_Toc382296896)

[Cancel Order Extended 36](#_Toc382296897)

[Apply Pin Product 37](#_Toc382296898)

[Get Pin Products 37](#_Toc382296899)

[Get Partners Lookup 37](#_Toc382296900)

[Move Order To Existing Customer 39](#_Toc382296901)

[Set Provisioning Zip 39](#_Toc382296902)

[Add Order Note 40](#_Toc382296903)

[Change Order Note 40](#_Toc382296904)

[Get Order Note 40](#_Toc382296905)

[Consumer API Group 41](#_Toc382296906)

[Add Consumer Account 41](#_Toc382296907)

[Get Consumer Accounts Lookup 42](#_Toc382296908)

[Change Consumer Account 43](#_Toc382296909)

[Close Consumer Account 44](#_Toc382296910)

[Set Consumer Payment Account 45](#_Toc382296911)

[Authenticate Consumer Account 45](#_Toc382296912)

[Reset Consumer Password 45](#_Toc382296913)

[Get Consumer Usage 46](#_Toc382296914)

[Get Consumer Statement 47](#_Toc382296915)

[Get Consumer Statements 48](#_Toc382296916)

[Get Consumer Active Products 48](#_Toc382296917)

[Add Consumer Note 50](#_Toc382296918)

[Change Consumer Note 50](#_Toc382296919)

[Get Consumer Note 51](#_Toc382296920)

[Get Consumer Payments 51](#_Toc382296921)

[Back Office API Group 53](#_Toc382296922)

[Get Defined Reports 53](#_Toc382296923)

[Get Report Detail 53](#_Toc382296924)

[Get Report Detail Paged 54](#_Toc382296925)

[Wholesale API Group 55](#_Toc382296926)

[Build Wholesale Order 55](#_Toc382296927)

[Add Wholesale Order Item 55](#_Toc382296928)

[Remove Wholesale Order Item 56](#_Toc382296929)

[Place Wholesale Order 56](#_Toc382296930)

[Cancel Wholesale Order 56](#_Toc382296931)

[Get Wholesale Order 56](#_Toc382296932)

[Add Dealer 57](#_Toc382296933)

[Change Dealer 58](#_Toc382296934)

[Close Dealer 58](#_Toc382296935)

[Get Dealers Lookup 58](#_Toc382296936)

[Product API Group 60](#_Toc382296937)

[Get Products 60](#_Toc382296938)

[Get Product Features 62](#_Toc382296939)

[Get Product Dependencies 64](#_Toc382296940)

[Service API Group 66](#_Toc382296941)

[Add Service Subscription 66](#_Toc382296942)

[Close Service Subscription 66](#_Toc382296943)

[Change Service Subscription 66](#_Toc382296944)

[Get Service Subscription 67](#_Toc382296945)

[Get Services Lookup 67](#_Toc382296946)

[Provision API Group 69](#_Toc382296947)

[Bulk Load Inventory 69](#_Toc382296948)

[Bulk Unload Inventory **(future)** 70](#_Toc382296949)

[Bulk Get Inventory **(future)** 70](#_Toc382296950)

[Bulk Activate Inventory **(future)** 70](#_Toc382296951)

[Bulk Deactivate Inventory **(future)** 70](#_Toc382296952)

[Get Device 70](#_Toc382296953)

[Activate Device 71](#_Toc382296954)

[Deactivate Device **(future)** 72](#_Toc382296955)

[Load Device 72](#_Toc382296956)

[Unload Device **(future)** 73](#_Toc382296957)

[Wireless Coverage Check 73](#_Toc382296958)

[Get Pending Auto Replenish 74](#_Toc382296959)

[Enable Pending Auto Replenish 74](#_Toc382296960)

[Cancel Pending Auto Replenish 75](#_Toc382296961)

[Payment API Group 76](#_Toc382296962)

[Add Payment Profile 76](#_Toc382296963)

[Get Payment Profile 77](#_Toc382296964)

[Close Payment Profile 78](#_Toc382296965)

[Process Payment 78](#_Toc382296966)

[Authorize Payment 79](#_Toc382296967)

[Delete Payment 79](#_Toc382296968)

[Refund Payment 80](#_Toc382296969)

[Set Payment Profile Default 80](#_Toc382296970)

[Lifeline API Group 81](#_Toc382296971)

[Build Lifeline Order (Future) 81](#_Toc382296972)

[Add Lifeline Order Item (Future) 81](#_Toc382296973)

[Remove Lifeline Order Item (Future) 82](#_Toc382296974)

[Place Lifeline Order (Future) 82](#_Toc382296975)

[Cancel Lifeline Order (Future) 82](#_Toc382296976)

[Get Lifeline Order (Future) 82](#_Toc382296977)

[Add Agent (Future) 83](#_Toc382296978)

[Change Agent (Future) 84](#_Toc382296979)

[Close Agent (Future) 84](#_Toc382296980)

[Get Agents Lookup (Future) 84](#_Toc382296981)

[Retail API Group 86](#_Toc382296982)

[Build Retail Order (Future) 86](#_Toc382296983)

[Add Retail Order Item (Future) 86](#_Toc382296984)

[Remove Retail Order Item (Future) 87](#_Toc382296985)

[Place Retail Order (Future) 87](#_Toc382296986)

[Cancel Retail Order (Future) 87](#_Toc382296987)

[Get Retail Order (Future) 87](#_Toc382296988)

[Add Retail Location (Future) 88](#_Toc382296989)

[Change Retail Location (Future) 89](#_Toc382296990)

[Close Retail Location(Future) 89](#_Toc382296991)

[Get Retail Locations Lookup (Future) 90](#_Toc382296992)

[Internal API Group 91](#_Toc382296993)

[Internal Add User (Future) 91](#_Toc382296994)

[Internal Change User (Future) 92](#_Toc382296995)

[Internal Delete User (Future) 93](#_Toc382296996)

[Internal Get user (Future) 93](#_Toc382296997)

[Internal Add Organization (Future) 94](#_Toc382296998)

[Internal Change Organization (Future) 96](#_Toc382296999)

[Internal Delete Organization (Future) 97](#_Toc382297000)

[Internal Get Organization (Future) 97](#_Toc382297001)

[Internal Get Organizations (Future) 99](#_Toc382297002)

[Internal Add User Organization (Future) 99](#_Toc382297003)

[Internal Delete User Organization (Future) 99](#_Toc382297004)

[Internal Get Users In Organization (Future) 100](#_Toc382297005)

[Internal Add Payment Customer 101](#_Toc382297006)

[Internal Get Payment Customer 103](#_Toc382297007)

[Internal Change Payment Customer 104](#_Toc382297008)

[Internal Close Payment Customer 105](#_Toc382297009)

[Internal Add Payment Profile 105](#_Toc382297010)

[Internal Get Payment Profile 107](#_Toc382297011)

[Internal Change Payment Profile 108](#_Toc382297012)

[Internal Close Payment Profile 110](#_Toc382297013)

[Internal Process Payment 111](#_Toc382297014)

[Internal Authorize Payment 112](#_Toc382297015)

[Internal Delete Payment 113](#_Toc382297016)

[Internal Refund Payment 114](#_Toc382297017)

[Internal Authorize User 114](#_Toc382297018)

[Internal Get Application Permission 116](#_Toc382297019)

[Internal Get Application Permissions 117](#_Toc382297020)

[Internal Get Menus 117](#_Toc382297021)

[Internal Get User Accounts 118](#_Toc382297022)

[Content API Group 120](#_Toc382297023)

[Get Templates 120](#_Toc382297024)

[Example Usage 120](#_Toc382297025)

[Get Blocks 121](#_Toc382297026)

[Example Usage 121](#_Toc382297027)

[Get Tags 122](#_Toc382297028)

[Example Usage 122](#_Toc382297029)

[Get Attributes 123](#_Toc382297030)

[Example Usage 123](#_Toc382297031)

[Carrier API Group 124](#_Toc382297032)

[Change Carrier Ownership 124](#_Toc382297033)

[Flow Examples 125](#_Toc382297034)

[Scenario 1 125](#_Toc382297035)

[Scenario 2 126](#_Toc382297036)

[Scenario 3 127](#_Toc382297037)

[Scenario 4 131](#_Toc382297038)

[Scenario 5 133](#_Toc382297039)

# 

# Document Information and Approvals

### Version Control

|  |  |
| --- | --- |
| Version History | |
| **Version #** | **Date** | | **Revised By** | **Reason for change** |
| 1.0.0 | 11/15/2012 | | Mike Schmidt | Initial draft |
| 1.0.1 | ¼/2013 | | Mike Schmidt | Flow changes, addition of new functions. |
| 1.0.2 | 1/5/2013 | | Kasey Drexler | Added User API Group and Functions. |
| 1.0.3 | 3/12/2013 | | Scott Bonen | Added Payment functions |
| 1.1.0 | 6/10/2013 | | Mike Schmidt | Major changes for integrated eCommerce |
| 1.1.1 | 6/19/2013 | | Mike Schmidt | Revisions from design discussion |
| 1.1.2 | 6/20/2013 | | Mike Schmidt | Banner image and ENUM changes |
| 1.1.3 | 6/24/2013 | | Mike Schmidt | MemberID, Opt-In/Opt-Out, Fee Lookups |
| 1.1.4 | 6/27/2013 | | Mike Schmidt | Final draft for eCommerce project |
| 1.1.5 | 7/1/2013 | | Mike Schmidt | BuildNewOrder billing\_mode change |
| 1.1.6 | 7/1/2013 | | Mike Schmidt | GetPartnersLookup modification |
| 1.1.7 | 7/11/2013 | | Mike Schmidt | Added GroupTag to dependencies, added new order methods. |
| 1.1.8 | 7/16/2013 | | Mike Schmidt | Modifications to GetPartnersLookup, GetOrdersLookup, GetOrdersQuote, and GetProducts. |
| 1.1.9 | 7/16/2013 | | Mike Schmidt | Modified GetOrderQuote for improved tax item handling. |
| 1.1.10 | 7/19/2013 | | Mike Schmidt | Changed dependency group type from PLAN to AIRTIME. Added dependency group type FEE. Added dependency type REQUIRE\_SHARED. |
| 1.1.11 | 8/14/2013 | | Mike Schmidt | * Changed GetOrdersLookup response to include Billing\_Mode and Customer ID. * Modified BulkLoadInventory, GetDevice, LoadDevice, and ActivateDevice. * Modified AddOrderDevice, AddOrderAirtimeProduct, AddOrderAccessoryProduct, AddOrderExtendedProduct, AddOrderPinProduct, AddOrderCreditProduct, AddOrderFeeProduct, AddOrderPromoProduct, AddOrderLocation to accept OrderIDs from BuildChangeOrder. * Added LineItemState to line item array response of GetOrdersLookup. * Added GetConsumerActiveProducts |
| 1.1.12 | 8/19/2013 | | Mike Schmidt | * Modified BuildNewOrder to accept an optional PartnerID field. * Changed AddPaymentProfile and GetPaymentProfile to include credit card type. |
| 1.1.13 | 9/4/2013 | | Mike Schmidt | * Modified BuildChangeOrder to accept a billing\_mode toggle. |
| 1.1.14 | 9/13/2013 | | Mike Schmidt | * Added InternalAuthorizeUser, InternalGetApplicationPermission, InternalGetApplicationPermissions, InternalGetMenus. |
| 1.1.15 | 9/18/2013 | | Mike Schmidt | * Added First/Last name to AddPaymentProfile CC info. |
| 1.1.16 | 9/20/2013 | | Mike Schmidt | * Added First/Last name to GetPaymentProfile * Changed GetOrdersLookup SummaryFlag to be mandatory. * Added function InternalGetUserAccounts and updated InternalAuthorizeUser to return a nested account array. |
| 1.1.17 | 9/24/2013 | | Mike Schmidt | * Change Line Array to Billing Account Array in GetConsumerActiveProducts. * Add PermissionID to permission array of InternalAuthorizeUser and InternalGetApplicationPermissions |
| 1.1.18 | 10/21/2013 | | Mike Schmidt | * Modified AddOrderDevice, AddOrderAirtimeProduct, AddOrderAccessoryProduct, AddOrderExtendedProduct, AddOrderPinProduct, AddOrderCreditProduct, AddOrderFeeProduct, AddOrderPromoProduct, AddOrderLocation to no longer accept OrderIDs from BuildChangeOrder. * Modified ChangeOrderAirtimeProduct to support adding an airtime product to an existing order device. Also added Prorate input fields. * Created CancelOrderDevice, CancelOrderExtended, ChangeOrderAccountBillingMode, and ChangeOrderAccessoryProduct * Modified GetOrdersLookup to include optional state filter and change result state to ENUM. * Modified BuildChangeOrder, BuildCancelOrder to only accept order IDs from GetConsumerActiveProducts. * Modified GetOrderQuote to only function against orders of type NEW. * Included primary device oolean to GetConsumerActiveProducts. * Modified GetPartnersLookup to return a corresponding Google Analytics ID (from account\_properties). |
| 1.1.19 | 11/15/2013 | | Mike Schmidt | * Created the SetPaymentProfileDefault method and enhanced AddPaymentProfile to require the Default Profile flag. |
| 1.1.20 | 11/25/2013 | | Mike Schmidt | * Add default profile flag to GetPaymentProfile |
| 1.1.21 | 12/02/2013 | | Scott Bonen | * Added new Consumer Method ResetConsumerPassword(). This provides the ability to reset a consumer’s password without having to know the current password. Note: Should only be called used by Customer Service type Applications. |
| 1.1.22 | 12/05/2013 | | Scott Bonen | * GetPartnersLookup now returns an array (Max of 5) of defined Social Media Icons and associated URLs. |
| 1.1.23 | 12/18/2013 | | Scott Bonen | * Modified getDefinedReports to support additional input/output parameters * Modified getReportDetail to support additional input/output parameters * Created new getReportDetailPaged method for supporting paging of HTML reports |
| 1.1.24 | 12/19/2013 | | Scott Bonen | * Added pageable flag to getDefinedReports response |
| 1.1.25 | 12/31/2013 | | Mike Schmidt | * Correct date/time format. |
| 1.1.26 | 1/24/2014 | | Todd Brady | * Added GetConsumerPayments method to consumer API section |
| 1.1.27 | 1/27/2014 | | Scott Bonen | * Modified GetOrdersLookup method to return Order total and associated Payment Transaction ID * Modified GetConsumerAccountsLookup method to return the State of the consumer |
| 1.1.28 | 1/27/2014 | | Todd Brady | * Modified GetConsumerPayments to have in parameters of DateFrom and DateTo * Removed out parameters of StatementID and RefundFlag for GetConsumerPayments |
| 1.1.29 | 2/11/2014 | | Scott Bonen | * Added Carrier API * Added statement that returned results are in an unordered state |
| 1.1.30 | 2/18/2014 | | Todd Brady | * Updated the content API methods fields. * Added usage sections for each method in content API |
| 1.1.31 | 3/5/2014 | | Todd Brady | * Added Order API Get Order Adhoc Quote method and updated the output of Get Order Quote |
| 1.1.32 | 3/11/2014 | | Todd Brady | * Added fee array to the output of the Consumer API GetConsumerActiveProducts call return |

### Approvals

This document is intended for external distribution and represents the official Ready Wireless API design and technical reference.

Following approval of this document, requirement changes will be governed by the project’s change management process, including impact analysis, appropriate reviews and approvals.

|  |  |
| --- | --- |
| Document Approvals | |
| **Approver Name** | **Title/Role** | | **Signature/Electronic Approval** | **Date** |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 11/18/2012 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 1/4/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 6/10/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 6/20/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 7/1/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 7/15/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 7/16/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 7/17/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 7/19/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED \* | 8/14/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.12\* | 8/19/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.13\* | 9/4/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.14\* | 9/13/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.15\* | 9/18/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.16\* | 9/20/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.17\* | 9/24/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.18\* | 10/21/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.19\* | 11/15/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.20\* | 11/25/2013 |
| Mike Schmidt | VP, Technology | | \* APPROVED 1.1.25\* | 12/31/2013 |
|  |  | |  |  |

# Overview

## Purpose

This document defines the integration services provided by Ready Wireless. These services can be utilized by Ready Wireless clients for order processing, phone provisioning and activation, as well as many other services provided by Ready Wireless.

## Connectivity

Connectivity to the Web Service will be provided over the public Internet with the HTTPS protocol on port 443. (Note: Straight HTTP is not supported)

URI format is defined as follows (scheme://domain:port/path)

* Scheme -> https://
* Domain -> api-rw.readywireless.com
* Port -> 443 (Not required as Ready utilizes default HTTPS port 443)
* Path -> /servicename/version/clientname
* Servicename -> Name of web service being called. IE OrderWS, CustomerWS, etc
* Version -> Version of Web Service. Format is ‘V’ followed by number. IE v1, v2, v3, etc.
* Clientname -> Defined client name as provided by Ready Wireless. IE Ready Mobile, Ready Wireless, etc
* Example URI’s:
* <https://api-rw.readywireless.com/order/v1/clientX>
* <https://api-rw.readywireless.com/customer/v3/clientY>

Ready Wireless utilizes firewalls for all Internet connectivity so appropriate firewall rules need to be applied in order for external clients to utilize these services. The following information is required to build the proper rules:

* Source IP Address or range. Example: 66.196.128.23, 66.196.128.0/24
* Point of contact information. Name, email, Business Phone
* Port: Will always be 443

## Web Service Technology

Ready Wireless utilizes industry standard SOAP technology via Web Service Description Language (WSDL) 2.0.

## Base Request Layout

All function calls contain a SOAP header and body section. The header section is a constant with the body section varying by Web Service and method. The header section contains the following fields.

* Client GUID:String:64🡪 The assigned secure key for the client.
* (Optional) Message ID:String:64🡪 Client may assign any value, returned in response

## Base Response Layout

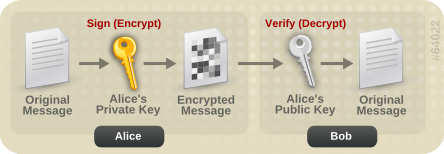
All function calls return the base response message block; which contains the following fields.

* Client GUID:String:64 🡪 The assigned secure key for the client.
* (Optional) Message ID:String:64🡪Message ID passed in by request.
* Return Code:Hex String:6 🡪 **0x0000** for Success, otherwise a category error return code.
* Return Sub-Code:Hex String:6 🡪 Code representing a specific error condition.
* Result Description:String:256 🡪Error text related to the error code. May contain appended text from external vendor partners.

## Security

Client identification and security is provided through two methods. Signed messages using x.509 certificates and Ready Wireless assigned 128-bit GUID.

Message signing will require the exchanged of CA signed or self-signed Public keys between the client and Ready Wireless. SOAP extension WS-Security is the model being used by Ready Wireless. Figure 1 depicts the high level flow.

 Figure 1:

Additionally each service request requires the assigned Ready Wireless GUID be provided. The GUID will be generated and provided by Ready Wireless. The GUID will not expire, but can be changed if either the client or Ready Wireless believes the GUID has been compromised. A future enhancement will be provided allowing the client the option of requesting a new GUID anytime through the provided service.

Example GUID: 2875d12f-24e8-49fe-807b-dd5b5be0e752

## Assumptions

* WSDL can be obtained by appending on ?WSDL to the client URI. Example: <https://api-rw.readywireless.com/order/v1/clientX?WSDL>
* Test site (URI) will be provided by Ready Wireless allowing clients to test their integration. Example: <https://apitest-rw.readywireless.com/order/v1/clientX>
* Test site functionality will be identical to production with the exception of data availability.

## Restrictions

* Not all methods defined in this document will be available to all clients.
* Method availability is based on client contract terms.
* API requests that return array lists are in an unordered state. This implies that returned results (data) will not always be in the same order.

## Return Codes

SUCCESS

The standard return code for SUCCESS in all categories is **0x0000**.

ERROR

Error codes are categorized by function with each group containing a number of sub- codes.

The error categories and corresponding return codes include:

* SYSTEM 🡪 **0x0001**
* DATABASE 🡪 **0x0002**
* APPLICATION 🡪 **0x0003**
* CARRIER 🡪 **0x0004**
* VENDOR 🡪 **0x0005**
* SECURITY 🡪 **0x0006**
* API 🡪 **0x0007**

SYSTEM Base Sub-codes

* General error 🡪 **0x0001**
* Application server host unavailable 🡪 **0x0002**
* Database server host unavailable 🡪 **0x0003**
* Network unavailable 🡪 **0x0004**
* Problem resolving hostname 🡪 **0x0005**
* Application server platform unavailable 🡪 **0x0006**
* (TBD)

DATABASE Base Sub-codes

* General error 🡪 **0x0001**
* Internal error 🡪 **0x0002**
* SQL error 🡪 **0x0003**
* Storage issue 🡪 **0x0004**
* Problems with configuration 🡪 **0x0005**
* (TBD)

APPLICATION Base Sub-codes

* General error 🡪 **0x0001**
* (TBD)

CARRIER Base Sub-codes

* General error 🡪 **0x0001**
* Unavailable 🡪 **0x0002**
* (TBD)

VENDOR Base Sub-codes

* General error 🡪 **0x0001**
* (TBD)

SECURITY Base Sub-codes

* Missing required security GUID 🡪 **0x0001**
* Problem with digital certificate signing 🡪 **0x0002**
* (TBD)

API Base Sub-codes

* General function error 🡪 **0x0001**
* Invalid input parameter 🡪 **0x0002**
* Permission denied 🡪 **0x0003**
* (TBD)

FUNCTION SPECIFIC

Function specific Return Sub Codes are documented with each individual function definition and fall under the API error code category. For example, a BuildNewOrder error return code would have a return code of **0x0007** and a sub-code of **0x0100**. It is also possible to receive a base sub-code within each of the return code categories. For example, a “General function error” would have a return code of **0x0007** with a sub-code of **0x0001**.

# API Reference

The following represent the available API functions for client implementation.

## Order API Group

The Order section contains the functions necessary to manage consumer orders. This API group is intended to serve as the backend for eCommerce, Customer Service Portals, and IVR systems. Orders are “built up” using base components to meet the client’s requirements.

Example API pseudo code to create a new prepaid anonymous customer with one device, one product and provisioned by Ready Wireless. Assumes client is managing payment gateway.

* NewOrderID = BuildNewOrder()
* LineItem {ID, Rate} = AddOrderDevice(NewOrderID, ModelID, ProvisionFlag)
* LineItem {ID, Rate} = AddOrderAirtimeProduct(NewOrderID, LineItemID, ProductID, Onetime-Flag)
* Totals {LineItems} = GetQuote(NewOrderID)
* Result = PlaceOrder(NewOrderID)

### Build New Order

This function creates the “shell” for a new consumer order within the Ready Wireless MNVE backend systems. All consumer orders must begin with a call to **BuildNewOrder** to setup the order data structures.

**Note**: The order structures created with **BuildNewOrder** will persist in a “BUILD” state for 48 hours before being swept up and marked “AUTO CANCEL”.

Inputs

* (Optional) Customer ID:Int:8 🡪If present the order will be tied to the customer.
  + If provided and customer does not exist an error response will be returned.
  + If no customer ID is present, the order will be tied to an anonymous customer account.
* Billing Mode: ENUM 🡪 One of ‘PRE’,’POST’,’POST\_SHARED’.
* (Optional) Partner ID:Int:8 🡪 The partner ID. If Customer ID is also provided, Partner ID will be ignored.

Outputs

* <standard response message block>
* Customer ID:Int:8 🡪 ID of the customer account associated with the order, whether that is an auto-generated (anonymous) account or specified by the caller.
* NewOrderID:Int:8 🡪 This will be the reference ID used when adding devices and products to the order.

Return Sub Codes

* 0x0100

### Add Order Device

Add a device to a new order.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* ProductID:Int:8 🡪Product ID of the device to add to the order.
  + **Note**: Product ID may also be a Product Group ID.
* Add Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Provisioning Flag:**0x01** 🡪 Set if using Ready Wireless fulfillment system.
    - **Note**: Does not require Ready Wireless fulfill the orders, only that the Ready Wireless provisioning system is used.
    - **Note**: Client must be configured within the MVNE platform for this function to successfully execute.
  + (Optional) Activate-Only Flag:**0x02** 🡪 Set if not using Ready Wireless’s fulfillment system and the provided ESN requires automatic activation. Use for scenarios where end-consumer purchases a deactivated phone from a branch location and uses a web site to activate and purchase products.
    - (Required) DeviceID:String:128 🡪The ESN/SIM of the device to activate.
      * **Note**: The ESN/SIM must exist within the client’s device inventory. Requires being loaded previously with either **BulkLoadInventory** or **LoadDevice**.
      * **Note**: The status of the device activation can be determined via **GetDevice.**
  + (Optional) Reference-Only Flag:**0x03** 🡪 Set if adding a device for future reference within the order or if this is a Retail, Lifeline, or Wholesale device that has been distributed/sold to an end consumer. Use this when the consumer has an activated/existing device and is purchasing additional products to load or if this is the first time the consumer is purchasing additional products. This flag will link the device with the consumer’s account if the link does not already exist. The LineItemID returned is used as the reference for add order product function calls.
    - (Required) DeviceID:String:128 🡪The ESN/SIM of the device to load products on.
    - **Note**: The ESN/SIM must exist within the client’s device inventory. Requires being loaded previously with either **BulkLoadInventory** or **LoadDevice**.
  + (Optional) Client Provision Flag:**0x04** 🡪 Set this if the client is fulfilling the device. Since the actual device is not known at the point of sale (eCommerce) the order will sit in a “HOLD” state until the client picks the device, provisions manually with **ActivateDevice**, and then calls **SetOrderDevice** to relate the provisioned ESN/SIM to the order, and then finally **ReleaseOrder** to have the products applied.
    - **Note**: The entire order will be placed on hold regardless of other devices added to the order that do not have the Client Provision Flag set.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0200

### Add Order Pin Product

Add a PIN purchase to the order for future use. Once order is placed a call to **GetPinProducts** is required to obtain the generated PINs.

**Note**: Does not require an association with a device.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* ProductID:Int:8 🡪Product ID of the pin product to generate.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0300

### Add Order AirTime Product

Use to add an airtime product to a device. This call requires an association with a device. User **AddOrderDevice** and pass in the returned LineItemID to associate the product with the device.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪The line item ID of a device on the order that these products should be loaded on.
* ProductID:Int:8 🡪Product ID of the product to apply.
* (Optional) Prepaid Flag:Int:2 🡪 Must be one of the following:
  + (Optional) One-Time Flag:**0x01** 🡪 Set if this product is being purchased for loading only once.
  + (Optional) Auto-Replenish Flag:**0x02** 🡪 Use this flag if the product is being setup for auto-replenish.
    - **Note**: Client must be configured for auto replenish feature. Either Ready Wireless handles the customer account, automatic charges, and product refreshes…or…Ready Wireless handles the auto-replenish trigger events and client manages processing of the events. See **GetPendingAutoReplenish**, **EnablePendingAutoReplenish**, and **CancelPendingAutoReplenish** for details on client manages auto replenish.
    - One of the following replenish options (or both) must be set:
      * (Optional) ReplenishByDayOfMonth:String:16 🡪 {First, Last, or specific day}
      * (Optional) ReplenishByDepletionPercentage:Decimal:2.2 🡪 percentage remaining
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Output

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0400

### Add Order Accessory Product

Add accessory product to the order. This call will take an optional device LineItemID if the caller wishes the accessory to be associated with a device.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* (Optional) LineItemID:Int:4 🡪The line item ID of a device on the order to associate the product with.
* ProductID:Int:8 🡪Product ID of the product to add.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0500

### Add Order Extended Product

Used for adding extended products to an order like warranties, support services, long distance services, etc.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* (Optional) LineItemID:Int:4 🡪The line item ID of a device on the order to associate the product with.
* ProductID:Int:8 🡪Product ID of the product to add.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0600

### Add Order Fee Product

Used for adding fee products to an order like add-a-line, data-add-a-line, activation fee, deposit, and shipping. Fee products must be added to an order to accurately reflect the order total presented to the user.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* (Optional) LineItemID:Int:4 🡪The line item ID of a device on the order to associate the product with.
* ProductID:Int:8 🡪Product ID of the product to add.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0620

### Add Order Credit Product

Used for adding credit products to an order for returns/refunds.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* (Optional) LineItemID:Int:4 🡪The line item ID of a device on the order to associate the product with.
* ProductID:Int:8 🡪Product ID of the product to add.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0640

### Add Order Promo Product

Used for adding promotion/discount products to an order. Multiple promo products may be added to an order based upon the client requirements.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* (Optional) LineItemID:Int:4 🡪The line item ID of a device on the order to associate the product with.
* ProductID:Int:8 🡪Product ID of the product to add.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
* Line Item Details
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Credit, Promo, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0660

### Add Order Location

Add an address to the order. If the order is associated with a Customer Account this function is optional since the Customer Account already has associated location objects. Even if a customer account is present this function may still be called to override the customer’s default location object. Only one instance of a location type is allowed per order. Each call to **AddOrderLocation** will overwrite any previous call’s data.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “BUILD” state.
* Location Flag:Int:2 🡪 Must be set to one of the following:
  + (Optional) Shipping Location Flag:**0x02** 🡪 Use when adding a shipping location to the order.
    - First Name:String:64
    - Last Name:String:64
    - Address Line 1:String:128
    - (Optional) Address Line 2:String:128
    - City:String:64
    - State:String:2
    - Zip:Int:5
    - (Optional) Contact Number:String:16
    - (Optional) Zip4:Int:4

Outputs

* <standard response message block>

Return Sub Codes

* 0x0700

### Remove Order Line Item

Use this function to remove a line item and all associated children from an order.

**Note**: This function does not re-number the items on the order.

Inputs

* OrderID:Int:8 🡪ID of the order being constructed.
  + **Note**: This function will accept order IDs from any build order function. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪The line item ID to remove from the order. All associated children will be removed also.

Outputs

* <standard response message block>

Return Sub Codes

* 0x0800

### Set Order Device

This call provides a mechanism for specific “client provisioned” devices to be associated with device line items on an order. For scenarios where the client picks, provisions, and ships their own devices the process to place an order via eCommerce is split across multiple organizations. By adding devices to an order using **AddOrderDevice** with the “client provision” flag set, placeholders are put into the order so a proper quote can be created and the client can later tie actual device IDs to the line item.

Inputs

* NewOrderID:Int:8 🡪ID of the new order being constructed.
  + **Note**: Only IDs from **BuildNewOrder** are accepted. Order must be in the “HOLD” state.
* LineItemID:Int:4 🡪The line item ID of a device on the order that the ESN/SIM will be associated with.
* One of the following is required. **Note**: The ESN/SIM must exist within the client’s device inventory. Requires being loaded previously with either BulkLoadInventory or LoadDevice.
  + (Optional) DeviceID:String:128 🡪The ESN/SIM of the device to associate with the line item.
  + (Optional) Subscriber ID:Int 🡪 The subscriber ID for this inventory device.

Outputs

* <standard response message block>

Return Sub Codes

* 0x0900

### Set Order Payment Account

Use to associate a payment gateway account with an order. The customer account has a default payment profile, but this method can be used to associate one-time payment profiles to orders or alternate permanent profiles.

Inputs

* Order ID:Int:8 🡪 The OrderID of the order being placed.
* Payment Profile ID:Int:8 🡪 Returns from GetPaymentProfiles.

Outputs

* <standard response message block>

Return Sub Codes

* 0x0950

### Place Order

Submits an order for processing. This function assumes the client has accounted for all financial requirements before placing the order. Recommended flow is to pre-authorize payment first, place the order, then apply final payment. Only functions on order IDs in the “BUILD” state.

**Note**: **PlaceOrder** will check the integrity of the order and return errors if line items are missing. For example, at least one independent product must be present. And, both a billing location and shipping location must be present if the order is not associated with a customer account. The client can make the necessary changes to the order based on the error codes returned.

Inputs

* OrderID:Int:8 🡪ID of a NEW order, a CHANGE order, or a CANCEL order.
* (Optional) CustomerID:Int:8 🡪 Last chance to associate the order with a Customer Account.
  + Note: Will override any CustomerID value supplied with the BuildNewOrder call.
* (Optional) Tax Class:String:32 🡪 Defaults to RESIDENTIAL. Can be one of RESIDENTIAL, COMMERCIAL, EXCEMPT.

Outputs

* <standard response message block>

Return Sub Codes

* 0x0A00

### CancelOrder

Multi-purpose cancel function. Will cancel orders in the “BUILD”, “PLACED”, AND “PROCESSING” states. Orders in the “SHIPPED” or “CLOSED” state will return an error.

Inputs

* OrderID:Int:8 🡪ID of a NEW order, a CHANGE order, or a CANCEL order.

Outputs

* <standard response message block>

Return Sub Codes

* 0x0B00

### Release Order

Moves an order from “HOLD” to “PLACED” for pickup by the automated order processing subsystems. This function is used in the scenario where an external client/vendor is fulfilling the devices on an order and must set the specific activated devices on the order. Once the devices are provisioning and associated with the consumer order ReleaseOrder() is called to continue with the order workflow.

Inputs

* Order ID:Int:8 🡪 ID of a NEW order, a CHANGE order, or a CANCEL order.

Outputs

* <standard response message block>

Return Sub Codes

* 0x0C00

### Get Order Quote

While the **BuildNewOrder**, **BuildChangeOrder**, and **BuildCancelOrder** can be used to mirror a shopping cart and also leveraged as persistent cart session storage, ultimately an order total must be calculated based on the billing location of the consumer. This function returned an invoice quote based upon all line items present in the order at the time of the call.

For example, an order with no line items will return a $0 total. An order with device and product line items will return a total broken by NRC and MRC values, but no taxes or fees. Only an order that has viable products plus a valid billing location will return a complete quote. It is the responsibility of the client to make the quote call at the appropriate times.

**NOTE**: This function will add MRC subtotal, NRC subtotal, total, and detailed tax line items to an order. If called multiple times, the previous quote-generated line items will be removed and re-added (with different line item IDs). If syncing the output from this function to a local cart, these items will need to be removed from the cart in order to stay in sync.

Inputs

* Order ID:Int:8 🡪 ID of a NEW order, a CHANGE order, or a CANCEL order.
  + **NOTE**: Currently only works against orders of type NEW.
* Zip:Int:5 🡪 Required to properly calculate taxes for the order
* (Optional) Zip4:Int:4
* (Optional) Tax Class:ENUM 🡪 Defaults to RESIDENTIAL.
  + Can be one of RESIDENTIAL, COMMERCIAL, EXEMPT

Outputs

* <standard response message block>
* Quote Line Item Array
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemProductID:Int:8 🡪 Product ID of the line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 The line item type. One of:
    - (Device, Airtime, Accessory, Extended)
    - (Discount, Savings, Promo)
    - (Tax, Sub Total, Total)
  + LineItemSubType:ENUM 🡪The line item subtype. One of:
    - FLIP, QWERTY, SMART, SIM, USB, MIFI, SMS, DATA, VOICE, VOICE\_SMS\_DATA, SMS\_AND\_DATA, SMS\_AND\_VOICE, ADD\_A\_LINE, ACTIVATION, SHIPPING, DEVICE\_SHIPPING, DEPOSIT, DATA\_ADD\_A\_LINE, BATTERY, BUNDLE, CASE, CHARGER, HEADSET, MEMORY, BYOD
  + LineItemRate:Decimal:6.4 🡪 Price paid for by consumer
  + (Optional) LineItemTax: Decimal: 6.4 🡪 Tax amount if applicable/available
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)
  + (Optional) LineItemParentID: Int: 8 🡪 The parent id (if exists) for this line item.

Return Sub Codes

* 0x0D00

Example Flow

* NewOrderID = BuildNewOrder()
* DeviceLineItem = AddOrderDevice(NewOrderID, ModelID, ProvisionFlag)
* AirtimeLineItem = AddOrderAirtimeProduct(NewOrderID, DeviceLineItem, ProductID, Onetime-Flag)
* QuoteResultSet = GetQuote(NewOrderID)
* Result = PlaceOrder(NewOrderID)

### Get Order Adhoc Quote

Similar to Get Order Quote except an order is not necessary. Instead of passing in an Order ID, an array of line items can be passed in and the quote response will be returned.

Inputs

* Array of Order Quote Line Items
  + Line Item Number:Int:4 🡪 ID of the order line item.
  + Line Item Product ID:Int:8 🡪 Product ID of the line item.
  + Line Item RateDecimal:6.4 🡪 Price paid for by consumer.
  + Parent Line Item Number:Int:8 🡪 The parent id (if exists) for this line item.
* BillingtMode:ENUM 🡪 Can be one of PRE, POST, PRE\_SHARED, POST\_SHARED
* Zip:Int:5 🡪 Required to properly calculate taxes for the order.
* (Optional) Zip4:Int:4
* (Optional) Tax Class:ENUM 🡪 Defaults to RESIDENTIAL.
  + Can be one of RESIDENTIAL, COMMERCIAL, EXEMPT

Outputs

* <standard response message block>
* Quote Line Item Array
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemProductID:Int:8 🡪 Product ID of the line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 The line item type. One of:
    - (Device, Airtime, Accessory, Extended)
    - (Discount, Savings, Promo)
    - (Tax, Sub Total, Total)
  + LineItemSubType:ENUM 🡪The line item subtype. One of:
    - FLIP, QWERTY, SMART, SIM, USB, MIFI, SMS, DATA, VOICE, VOICE\_SMS\_DATA, SMS\_AND\_DATA, SMS\_AND\_VOICE, ADD\_A\_LINE, ACTIVATION, SHIPPING, DEVICE\_SHIPPING, DEPOSIT, DATA\_ADD\_A\_LINE, BATTERY, BUNDLE, CASE, CHARGER, HEADSET, MEMORY, BYOD
  + LineItemRate:Decimal:6.4 🡪 Price paid for by consumer
  + (Optional) LineItemTax: Decimal: 6.4 🡪 Tax amount if applicable/available
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)
  + (Optional) LineItemParentID: Int: 8 🡪 The parent id (if exists) for this line item.

Return Sub Codes

* 0x0D00

Example Flow

* NewOrderID = BuildNewOrder()
* DeviceLineItem = AddOrderDevice(NewOrderID, ModelID, ProvisionFlag)
* AirtimeLineItem = AddOrderAirtimeProduct(NewOrderID, DeviceLineItem, ProductID, Onetime-Flag)
* QuoteResultSet = GetQuote(NewOrderID)
* Result = PlaceOrder(NewOrderID)

### Get Orders Lookup

Returns information regarding orders associated with an order ID, customer, ESN, or MDN. Will return all line item details for all orders associated with the search criteria. Could potentially be a large result set depending on circumstances.

**Note**: Requires at least one of the optional search parameters to be present.

**Note**: This call may return multiple clients and associated orders with a limit of 25 customers returned.

Inputs

* (Optional) Order ID:Int:8 🡪 Only returns info pertaining to order ID.
* (Optional) Customer ID:Int:8 🡪A customer ID from the GetCustomerAccount function.
* (Optional) Name/Address 🡪The customer’s name and address information.
  + First Name:String:64
  + Last Name:String:64
  + Street:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + (Optional) Zip4:Int:4
* (Optional) Device ID:String:128 🡪 ESN/SIM for lookup.
* (Optional) MDN:String:16🡪 An MDN to lookup orders associated with.
* Summary Flag:Boolean:1 🡪Do not provide order line item details. Only order summary information: order number, order date, order total, order state, order type, etc.
* (Optional) State:String:64 🡪 Order state (BUILDING, PLACED, HOLDING, PROCESSING, SHIPPED, COMPLETED, CLOSED, CANCELED). Use to filter orders by state. Can be combined using “+” (I.E. PLACED+PROCESSING).

Output

* <standard response message block>
* Order Array
  + Order Summary Info
    - Order ID:Int:8 🡪 Order ID specific to this order
    - Type:String:16 🡪 Order type (NEW, CANCEL, CHANGE)
    - State:ENUM 🡪 Order state (BUILDING, PLACED, HOLDING, PROCESSING, SHIPPED, COMPLETED, CLOSED, CANCELED)
    - BuildDate:Date:32 🡪 Date order build started. Format: YYYY-MM-DD HH24:MM:SS
    - PlacedDate:Date:32 🡪 Date order was placed. Format: YYYY-MM-DD HH24:MM:SS
    - CurrentStateDate:Date:32 🡪 Current state started. Format: YYYY-MM-DD HH24:MM:SS
    - Billing Mode: ENUM 🡪 One of ‘PRE’,’POST’,’POST\_SHARED’.
    - Customer ID:Int:8 🡪 The account ID of the order.
    - Order Total:Decimal:6.4 🡪 Total amount of order
    - Payment Transaction ID:Int:8 🡪 Payment Transaction ID for Order
  + (Optional) Shipping Location
    - First Name:String:64
    - Last Name:String:64
    - Address Line 1:String:128
    - (Optional) Address Line 2:String:128
    - City:String:64
    - State:String:2
    - Zip:Int:5
    - (Optional) Contact Number:String:16
    - (Optional) Zip4:Int:4
  + (Optional) Billing Location
    - First Name:String:64
    - Last Name:String:64
    - Address Line 1:String:128
    - (Optional) Address Line 2:String:128
    - City:String:64
    - State:String:2
    - Zip:Int:5
    - (Optional) Contact Number:String:16
    - (Optional) Zip4:Int:4
  + Order Line Item Array
    - LineItemID:Int:4 🡪 ID of the order line item.
    - LineItemProductID:Int:8 🡪 Product ID of the line item.
    - LineItemShortName:String:32 🡪 Short name of the product.
    - LineItemDescription:String:128 🡪 Detail text description of the orders product.
    - LineItemType:ENUM 🡪 One of:
      * (Device, Airtime, Accessory, Extended)
      * (Discount, Savings, Promo)
      * (Tax, Sub Total, Total)
    - LineItemRate:Decimal:6.4 🡪 Price paid for by consumer
    - LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)
    - LineItemState:ENUM 🡪 (‘ACTIVE’,’CHANGED’,’CANCELED’,’ERROR’)
    - (Optional) LineItemTaxes:Decimal:6.4 🡪 Taxes and fees paid by consumer for this product.
    - (Optional) LineItemParentID: Int: 8 🡪 The parent id (if exists) for this line item.
    - Order Line Item Details
      * Device ID:String:128 🡪 ESN/SIM of assigned device, null if not a device line item.

Return Sub Codes

* Missing required search option 🡪 0x0E 01
* Missing required minimum search field criteria 🡪 0x0E 02

Example Flow

* OrderResultSet = GetOrdersLookup(“John”, “Doe”, “123 Park St”, “Middletown”, “KS”, “39344”);

### Build Change Order

This function creates the “shell” for a consumer change order within the Ready Wireless MNVE backend systems. All changes to a consumer order must begin with a call to **BuildChangeOrder**.

**Note**: The order structures created with **BuildChangeOrder** will persist in a “BUILD” state for 48 hours before being swept up and marked “AUTO CANCEL”.

Inputs

* Order ID:Int:8 🡪An order ID returned by GetConsumerActiveProducts. Order IDs returned by GetOrdersLookup will return an error if used. GetConsumerActiveProducts returns order IDs correctly synchronized within the platform.

Outputs

* <standard response message block>
* ChangeOrderID:Int:8 🡪 This will be the reference ID used when calling change feature functions on the order and for calling **PlaceOrder**, **GetOrderQuote**, or **CancelOrder**.
* Original Order Line Item Array 🡪 The line item details of the original order. The LineItemID of these details will be needed as input to change feature function calls in order to replace previously purchased items.
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x0F00

### Change Order Airtime Product

Use to replace or add an airtime product from a previous order. This call requires a reference to a previous order airtime product line item.

**Note**: This function may be called multiple times to apply airtime plan changes to multiple devices or plans on an order.

Inputs

* ChangeOrderID:Int:8 🡪ID of the change order being constructed.
  + **Note**: Only IDs from **BuildChangeOrder** are accepted. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪 The line item ID provided can be one of the following line item types (DEVICE or AIRTIME). When adding a new airtime plan to an existing device provide a previous order line item ID for a device. When changing an existing airtime plan provide the previous order line item ID of the plan to change.
  + DEVICE: The line item ID of the device on the original order to associate the new airtime plan.
  + AIRTIME: The line item ID of the airtime plan on the original order to replace.
* ProductID:Int:8 🡪Product ID of the product replacing the original.
* Change Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Auto-Replenish Flag:**0x01** 🡪 Use this flag if a auto-replenish product is being changed.
    - **Note**: Client must be configured for auto replenish feature. Either Ready Wireless handles the customer account, automatic charges, and product refreshes…or…Ready Wireless handles the auto-replenish trigger events and client manages processing of the events. See **GetPendingAutoReplenish**, **EnablePendingAutoReplenish**, and **CancelPendingAutoReplenish** for details on client manages auto replenish.
    - One of the following replenish options (or both) must be set:
      * (Optional) ReplenishByDayOfMonth:String:16 🡪 {First, Last, or specific day}
      * (Optional) ReplenishByDepletionPercentage:Decimal:2.2 🡪 percentage remaining
  + (Optional) MRC Flag:**0x02** 🡪 Use this option for changing post-paid plans.
    - <future implementation>
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.
* (Optional) Prorate Flag:ENUM 🡪 (NOW, BILLING\_CYCLE, CUSTOM)
  + NOW: New plans go into effect immediately. (Default)
  + BILLING\_CYCLE: New plans go into effect at the next billing cycle.
  + CUSTOM: New plans will go into effect at the date provided.
* (Optional) Prorate Date:Datetime 🡪 The date new plans go into effect.

Output

* <standard response message block>
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x1000

### Change Order Accessory Product

Use to add an accessory product to a previous order. This call requires a reference to a previous order device line item.

**Note**: This function may be called multiple times to apply accessory items to single or multiple devices on an order.

Inputs

* ChangeOrderID:Int:8 🡪ID of the change order being constructed.
  + **Note**: Only IDs from **BuildChangeOrder** are accepted. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪 The line item ID of the device on the original order to associate the new accessory.
* ProductID:Int:8 🡪Product ID of the product replacing the original.
* (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Output

* <standard response message block>
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x1020

### Change Order Account Billing Mode

This method allows the caller to modify a consumer’s account; changing the billing mode from POST to POST\_SHARED or vice versa. The caller is required to have all of the necessary order and order line ID information required to rebuild the account. The caller is also responsible for ensuring all of the necessary airtime plans and fees are applied.

The caller will need to get the device dependencies for each active device on the account and ensure the necessary plans are included. An error will be returned if dependency rules are not followed.

This function automatically toggles the account from the current state to the opposing state. For example, if the account is shared (POST\_SHARED) then the input values must match the rules required for a non-shared (POST) accounts (I.E. each line has a voice plan, etc.).

For reference:

* Shared accounts…
  + Only allow airtime plans to be defined under the primary line. Attempts to add airtime plans to additional lines will result in an error.
  + Additional lines may only have fees defined.
  + Device dependencies must be checked to determine the minimum required airtime plan(s)
    - Note: Typically GetConsumerProducts will return a smart phone as the primary device if account maintenance has removed the original primary device from the account.
* Non-Shared accounts…
  + Each line is required to provide airtime plan(s) based upon the device dependencies.

Inputs

* Order ID:Int:8 🡪 The order ID of the current primary device. Returned by GetConsumerActiveProducts.
* (Optional) Prorate Flag:ENUM 🡪 (NOW, BILLING\_CYCLE, CUSTOM)
  + NOW: New plans go into effect immediately. (Default)
  + BILLING\_CYCLE: New plans go into effect at the next billing cycle.
  + CUSTOM: New plans will go into effect at the date provided.
* (Optional) Prorate Date:Datetime 🡪 The date new plans go into effect.
* Primary Line
  + Order Line Item ID:Int:4 🡪 The order line item ID of the current primary device. Returned by GetConsumerActiveProducts.
  + Airtime Product Array
    - Airtime Product (Repeat for each product)
      * Product ID:Int:8 🡪 the ID of the airtime products to associate with the primary device.
      * (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.
* (Optional) Additional Line 🡪 May repeat multiple times as needed.
  + (Optional) Order Line Item ID:Int:4 🡪 The order line item ID of the device. Returned by GetConsumerActiveProducts.
  + (Optional) Airtime Product Array
    - Airtime Product (Repeat for each product)
      * Product ID:Int:8 🡪 the ID of the airtime products to associate with the primary device.
      * (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.
  + (Optional) Fee Array
    - Fee Product (Repeat for each fee)
      * Product ID:Int:8 🡪 the ID of the fee product to associate with this device
      * (Optional) Rate:Decimal:6.4 🡪 Override the default rate configured for the product.

Outputs

* <standard response message block>
  + LineItemID:Int:4 🡪 ID of the account mode change line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Change\_Shared, Change\_NonShared)

Return Sub Codes

* 0x1040 ->

### Build Cancel Order

This function creates the “shell” for a consumer cancel order within the Ready Wireless MNVE backend systems. All cancels on a consumer order must begin with a call to **BuildCancelOrder**.

**Note**: The order structures created with BuildCancelOrder will persist in a “BUILD” state for 48 hours before being swept up and marked “AUTO CANCEL”.

Inputs

* Order ID:Int:8 🡪An order ID returned by GetConsumerActiveProducts. Order IDs returned by GetOrdersLookup will return an error if used. GetConsumerActiveProducts returns order IDs correctly synchronized within the platform.

Outputs

* <standard response message block>
* CancelOrderID:Int:8 🡪 This will be the reference ID used when calling cancel feature functions on the order and for calling **PlaceOrder**, **GetOrderQuote**, or **CancelOrder**.
* Original Order Line Item Array 🡪 The line item details of the original order. The LineItemID of these details will be needed as input to cancel feature function calls in order to replace previously purchased items.
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Fee, Taxes, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x1100

### Cancel Order Airtime Product

Use to cancel an airtime product from a previous order. This call requires a reference to a previous order airtime product line item.

Inputs

* CancelOrderID:Int:8 🡪ID of the cancel order being constructed.
  + **Note**: Only IDs from **BuildCancelOrder** are accepted. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪The line item ID of the original order to cancel. This function will determine the device to associate based on the original order.
* Cancel Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Auto-Replenish Flag:**0x01** 🡪 Use this flag if an auto-replenish product is being canceled.
  + (Optional) MRC Flag:**0x02** 🡪 Use this option for canceling post-paid plans.

Output

* <standard response message block>
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Tax, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x1200

### Cancel Order Device

Use to suspend/restore/deactivate a device from a previous order. This call requires a reference to a previous order device line item.

Inputs

* CancelOrderID:Int:8 🡪ID of the cancel order being constructed.
  + **Note**: Only IDs from **BuildCancelOrder** are accepted. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪The line item ID of the original order to cancel. Only line items that reference DEVICE product types are allowed.
* Cancel Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Suspend Flag:**0x01** 🡪 Don’t deactivate/release the device and MDN. Just suspend it.
  + (Optional) Restore Flag:**0x02** 🡪 Use this option to restore a previously suspended device.
  + (Optional) Deactivate Flage:**0x03** 🡪 Release the MDN and remove the device from the account.

Output

* <standard response message block>
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Tax, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x1220

### Cancel Order Extended

Use to cancel an extended service plan a previous order. This call requires a reference to a previous order device line item.

Inputs

* CancelOrderID:Int:8 🡪ID of the cancel order being constructed.
  + **Note**: Only IDs from **BuildCancelOrder** are accepted. Order must be in the “BUILD” state.
* LineItemID:Int:4 🡪The line item ID of the original order to cancel. Only line items that reference EXTENDED product types are allowed.

Output

* <standard response message block>
  + LineItemID:Int:4 🡪 ID of the order line item.
  + LineItemShortName:String:32 🡪 Short name of the product.
  + LineItemDescription:String:128 🡪 Detail text description of the orders product.
  + LineItemType:ENUM 🡪 (Device, Airtime, Accessory, Extended, Tax, Sub Total, Total)
  + LineItemRate:Decimal:6.4 🡪 Rate applied to product on order
  + LineItemRateType: ENUM 🡪 (‘NRC’ – Non Recurring Charge, ‘MRC’ – Monthly Recurring, ‘NA’)

Return Sub Codes

* 0x1240

### Apply Pin Product

Apply a previously purchased PIN product to a device.

**Note**: Requires one of the optional device inputs to be present.

Inputs

* PIN:String:64 🡪 The PIN reference to lookup the product to load.
* (Optional) Device ID:String:128 🡪 The ESN/SIM of the device to load the product.
* (Optional) MDN:String:16 🡪 The MDN of the device to load the product.
* (Optional) Load Date:Date:32 🡪 Date to apply PIN in the future. Format: YYYY-MM-DD HH24:MM:SS
  + **Note**: This is a future enhancement.

Outputs

* <standard response message header>

Return Sub Codes

* 0x1300

### Get Pin Products

Call to get the PINs produced by a new order containing pin products.

Inputs

* Order ID:Int:8 🡪 The order ID that contains the pin products purchased. Returns an error if no pin products are located within the order.

Outputs

* <standard response message block>
* PIN Array
  + PIN:String:64
  + Associated Product ID:Int:8
  + Associated Custom Key:String:32

Return Sub Codes

* 0x1400

### Get Partners Lookup

Use to get a list of all partners supported within this account/site (active only).

**Note**: Can be used to retrieve specific partner account by ID or Name. This function will also return the top-level site account if no Partner ID is provided. For example: READYMOBILE is the top-level account over multiple partner accounts. The top-level account can be determined using the Type enumerator and will be of type ‘ECOMMERCE’.

Inputs

* (Optional) Partner ID:Int:8 🡪 Can be used to filter a specific partner rather than return the entire list.
* (Optional) Account Identifier: String:128 🡪 Partner unique account code.
* (Optional) Name:String:128 🡪 Use to filter by partner name. I.E. (Farm Bureau)
* (Optional) Promo Code:String:64 🡪 Search all promo codes for this site to find the matching partner.

Outputs

* <standard response message block>
* Partner Array
  + Partner ID:Int:8 🡪 The partner ID.
  + Account Identifier:String:128 🡪 The partner code.
  + Name:String:128 🡪 Dealer specific name.
  + Type: ENUM 🡪 One of (‘ECOMMERCE’,’PARTNER’).
  + Partner Carrier Codes Array 🡪 Carriers support by this partner.
    - Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
  + Address Line 1:String:128
  + Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + Zip4:Int:4
  + Contact Number:String:16
  + Contact Email Address:String:128
  + (Optional) Standard Logo Image URL:String:512 🡪 Relative or physical path to image on website
  + (Optional) Alternate Logo Image URL:String:512 🡪 Relative or physical path to image on website
  + (Optional) Banner Array 🡪 Up to five images for rotating banners
    - Banner Image URL:String:512 🡪 Relative or physical path to banner image on website
    - Banner Image Link URL: String: 512 🡪 Full path to linked site.
  + (Optional) Social Media Array 🡪 Up to five social media definitions
    - Social Media Image URL:String:512 🡪 Relative or physical path to Social Media image on website
    - Social Media Link URL: String: 512 🡪 Full path to linked site.
  + (Optional) Promotions Array
    - Product ID:Int:8 🡪 Product ID
    - Code:String:64 🡪 The user supplied promotions code
    - Status:ENUM 🡪 One of ACTIVE,DISABLED,PENDING,EXPIRED
    - Start Date:Date:32 🡪 Promo campaign start. Format: YYYY-MM-DD HH24:MM:SS
    - End Date:Date:32 🡪 Promo campaign stop. Format: YYYY-MM-DD HH24:MM:SS
    - Type:ENUM 🡪 One of following:
      * PRODUCT PERCENT = Reduce a product by a percentage. Check the product dependency array for the required product needed to discount.
      * PRODUCT AMOUNT = Discount a product by a specific amount. Check the product dependency array for the required product needed to discount.
      * PRODUCT FREE = A free product. Check the product dependency array for requirements and free product info.
      * INVOICE PERCENT = Discount an entire invoice by a percentage. Could have minimum invoice amount restriction and included product dependencies.
      * INVOICE AMOUNT = Discount an entire invoice by a set amount. Could have minimum invoice amount restriction and included product dependencies.
    - Value:Decimal:6.4 🡪 Either an amount or a percent depending upon type.
    - (Optional) Invoice Minimum:Decimal:6.4 🡪 Minimum invoice total amount accepted.
    - (Optional) Billing Mode:String 🡪 Billing mode of promo product. One of (‘PRE’,’POST’,’POST SHARED’).
  + Google Analytics Account ID:String:64 🡪 The account ID to switch to for subsequent Google Analytics calls.

Return Sub Codes

* 0x1450

### Move Order To Existing Customer

Call to move an order to an existing customer account. This is used primarily when the user fails to login into their account until after an order has been created. Calling this function will move the order from the anonymous customer account created when calling BuildNewOrder to the user’s proper account. The anonymous customer account will be deleted.

Inputs

* Order ID:Int:8 🡪 The Order ID to be moved.
* Customer ID:Int:8 🡪 The Customer ID the order will be assigned towards.

Outputs

* <standard response message block>

Return Sub Codes

* 0x1475

### Set Provisioning Zip

Use to set the provisioning zip location on the order. This is at the order level. All devices being activated from this order will be provisioned to this zip code. Overrides the order shipping address zip.

Inputs

* Order ID:Int:8 🡪 The Order ID to be moved.
* Zip:Int:5 🡪 The zip code to provision against.

Outputs

* <standard response message block>

Return Sub Codes

* 0x1480

### Add Order Note

Use to add a free-form note to an order. Note: If defined, note will be passed to configured billing system.

Inputs

* Order ID:Int:8 🡪 The Order ID to associate note.
* Subject:String:256 🡪 Subject of the Note.
* Note:Text 🡪 The Note.
* User ID:Int:8 🡪 The ID of the User adding the Note.

Outputs

* <standard response message block>

Return Sub Codes

* 0x16A0 🡪 Success
* 0x16A1 🡪 Internal API Error, escalate issue to support
* 0x16A2 🡪 Invalid Parameter
* 0x16A3 🡪 Order ID not found
* 0x16A4 🡪 User ID not found

### Change Order Note

Use to change an existing note associated with an order. Note: If defined, note will be passed to configured billing system.

Inputs

* Order ID:Int:8 🡪 The Order ID the note is attached too.
* Note ID:Int:8 🡪 The ID of the note to change.
* Note:Text 🡪The Note.
* User ID:Int:8 🡪The ID of the User changing the Note.

Outputs

* <standard response message block>

Return Sub Codes

* 0x16C0 🡪 Success
* 0x16C1 🡪 Internal API Error, escalate to support
* 0x16C2 🡪 Invalid Parameter
* 0x16C3 🡪 Order ID not found
* 0x16C4 🡪 User ID not found
* 0x16C5 🡪 Note ID not found
* 0x16C6 🡪 Note is not of type Order

### Get Order Note

Use to get the details of a note.

Inputs

* Order ID:Int:8 🡪 The Order ID to get notes for.
* (Optional) Note ID:Int:8 🡪 If provided than only this notes information is returned.

Outputs

* <standard response message block>
* Note Array
  + ID:Int:8 🡪 ID of the Note
  + Subject:String:256
  + Type:String:32 🡪 Always “Order”
  + Note History Array
    - Created Date:Date
    - Username:String:32
    - Note:Text

Return Sub Codes

* 0x16E0 🡪 Success
* 0x16E1 🡪 Internal API Error, escalate to support
* 0x16E2 🡪 Invalid Parameter
* 0x16E3 🡪 Order ID not found
* 0x16E4 🡪 Note ID not found

## Consumer API Group

The Consumer API group is for manipulating customer account information. It provides the mechanisms necessary to manage consumer accounts for eCommerce sites, Customer Service portals, and IVR systems.

### Add Consumer Account

Function call to define and setup a customer account. By creating a consumer account, orders placed towards that account will associate products purchased, usage, and security features.

Inputs

* Username:String:128
* Password:String:128
* (Optional) Partner ID: Int: 8 🡪 If set, the consumer account will be created under this Partner. If the Customer ID field is also provided the previous anonymous account will be moved under the Partner.
* (Optional) Customer ID:Int:8 🡪 Must be an existing customer account returned from Build New Order. If set, a previous anonymous account will be updated with the information provided and turned into a named account.
* (Optional) Legacy Account Pin:String:4
* (Optional) Member ID: String: 64 🡪 Include any specific Partner or Client identification number.
* Contact Info
  + First Name:String:64
  + Last Name:String:64
  + (Optional) Phone Number:String:16
  + (Optional) Email Address:String:128
* Default Billing Location
  + Address Line 1:String:128
  + (Optional) Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + (Optional) Contact Number:String:16
  + (Optional) Zip4:Int:4
* Default Shipping Location
  + Address Line 1:String:128
  + (Optional) Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + (Optional) Contact Number:String:16
  + (Optional) Zip4:Int:4

Outputs

* <standard response message block>
* Customer ID:Int:8

Return Sub Codes

* 0x1501 -> Required field Username missing
* 0x1502 -> Required field Password missing
* 0x1503 -> Internal System Error (Retry call or escalate to ReadyWireless)
* 0x1504 -> Required field Email missing
* 0x1505 -> Account already exists
* 0x1506 -> Invalid Password
* 0x1507 -> Creation of Account Failed
* 0x1508 -> Invalid Billing Location
* 0x1509 -> Invalid Shipping Location

### Get Consumer Accounts Lookup

Returns customer account details.

Inputs

* (Optional) Partner ID: Int: 8 🡪 If set, the consumer account search will start at the specified Partner ID.
* One of the following can be used for account filtering.
  + (Optional) Customer ID:Int:8
  + (Optional) If set, either First Name/Last Name or Address/City/State/Zip or All must be provided
    - (Optional) First Name:String:64
    - (Optional) Last Name:String:64
    - (Optional) Address Line:String:128
    - (Optional) City:String:64
    - (Optional) State:String:2
    - (Optional) Zip:Int:5
  + (Optional) Device ID:String:128 🡪 ESN/SIM
  + (Optional) MDN:String:16
  + (Optional) Username:String:128
  + (Optional) Member ID: String: 64 🡪 Partner or Client identification number.

Outputs

* <standard response message block>
* Consumer Account Array
  + Customer ID:Int:8
  + Customer State:ENUM 🡪 One of ACTIVE, INACTIVE, DELETED, or UNKNOWN (Note: Unknown is set when the state of the consumer cannot be determined. If received this should be escalated to support for further investigation)
  + Username:String:128
  + Contact Info
    - First Name:String:64
    - Last Name:String:64
    - Phone Number:String:16
    - Email Address:String:128
  + Default Billing Location
    - Address Line 1:String:128
    - (Optional) Address Line 2:String:128
    - City:String:64
    - State:String:2
    - Zip:Int:5
    - (Optional) Contact Number:String:16
    - (Optional) Zip4:Int:4
  + Default Shipping Location
    - Address Line 1:String:128
    - (Optional) Address Line 2:String:128
    - City:String:64
    - State:String:2
    - Zip:Int:5
    - (Optional) Contact Number:String:16
    - (Optional) Zip4:Int:4
  + Order ID Array
    - OrderID:Int:8
* (Optional) Partner ID:Int:8 🡪 If this consumer is part of a partner program the Partner ID is returned.
* (Optional) Member ID: String: 64 🡪 Partner or Client identification number.

Return Sub Codes

* 0x1601 -> Invalid Lookup Request, No lookup parameters provided
* 0x1602 -> Not Found, No accounts found for given criteria
* 0x1603 -> Internal System Error (Retry call or escalate to ReadyWireless)

### Change Consumer Account

Use to change customer account information.

Inputs

* Customer ID:Int:8
* (Optional) Username:String:128
* (Optional) Old Password:String:128 🡪 If changing password (New Password populated) this must be set correctly with the current password on the account.
* (Optional) New Password:String:128
* (Optional) Legacy Account Pin:String:4
* (Optional) Member ID: String: 64 🡪 Include any specific Partner or Client identification number.
* (Optional) Contact Info
  + First Name:String:64
  + Last Name:String:64
  + Phone Number:String:16
  + Email Address:String:128
* (Optional) Default Billing Location
  + Address Line 1:String:128
  + (Optional) Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + (Optional) Contact Number:String:16
  + (Optional) Zip4:Int:4
* (Optional) Default Shipping Location
  + Address Line 1:String:128
  + (Optional) Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + (Optional) Contact Number:String:16
  + (Optional) Zip4:Int:4

Outputs

* <standard response message block>

Return Sub Codes

* 0x1701 -> Invalid Consumer
* 0x1702 -> Account Not Found
* 0x1703 -> Internal System Error (Retry call or escalate to ReadyWireless)
* 0x1704 -> Password Mismatch, Passed OldPassword doesn’t match current
* 0x1705 -> Password Failure, unable to process new Password
* 0x1706 -> Deleted accounts cannot be modified

### Close Consumer Account

Use this function to close a customer’s account. This does not delete the account (for audit purposes), but the account cannot be re-enabled once closed.

Inputs

* Customer ID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x1801 -> Account Not Found
* 0x1802 -> Accounts marked as deleted cannot be closed
* 0x1803 -> Internal System Error (Retry call or escalate to ReadyWireless)

### Set Consumer Payment Account

Use to associate a default payment gateway account with a customer. Used to establish a payment profile for future eCommerce purchases, auto-replenish IVR top-ups, etc.

Inputs

* Customer ID:Int:8
* Payment Profile ID:Int:8 🡪 Returns from GetPaymentProfiles.

Outputs

* <standard response message block>

Return Sub Codes

* 0x1950

### Authenticate Consumer Account

Use this function to authenticate a username and password.

Inputs

* Customer ID:Int:8
* Username:String:128
* Password:String:128

Outputs

* <standard response message block>
* (Optional) Partner ID:Int:8 🡪 If this consumer is part of a partner program the Partner ID is returned.

Return Sub Codes

* 0x1901 -> Invalid Parameters, Username and Password required
* 0x1902 -> Account Not Found
* 0x1903 -> Internal System Error (Retry call or escalate to ReadyWireless)
* 0x1904 -> Account is Inactive

### Reset Consumer Password

Use this function to reset a consumer’s password. The function will generate a random password and if requested will send a password reset email to the consumers primary email address.

Inputs

* Customer ID:Int:8
* Email Flag:Boolean: 🡪 Generated password should be emailed to consumer. Defaults to False

Outputs

* <standard response message block>
* Password:String:128 🡪 Random generated password.

Return Sub Codes

* 0x1910

### Get Consumer Usage

Function to look up and retrieve summary or detail usage information

**Note**: Detail usage is very bandwidth intensive so each call is limited to one month of data per call. It is advised that detail usage is presented in a paged manner for the consumer to visually understand. If data range is not provided the current usage for the month is returned (may not be a full month).

Inputs

* Customer ID:Int:8 🡪 The Customer ID to be retrieved.
* Device Lookup, must be one or more of the following.
  + (Optional) Subscriber ID:Int:8 🡪 Direct LineID associated with this CustomerID.
  + (Optional) MDN:String:32 🡪 The specific telephone number.
  + (Optional) Identifier:String:64 🡪 Device ESN (CDMA) or IMEI (GSM)
* Date From:Datetime 🡪 Date/Time usage will be greater than or equal too.
* Date To:Datetime 🡪 Date/Time usage will be less than.
* Mode: ENUM 🡪 One of (‘DETAIL’,’SUM’,’BOTH’);
  + DETAIL = Individual call details
  + SUM = Summary of usage and overage by billing cycle month. Shared accounts have combined usage/overage associated with the primary line.
  + BOTH = Return both data sets
* Type: ENUM 🡪 One of (‘INTERSTATE’,’DATA’,’SMS’,’ALL’)

Outputs

* <standard response message block>
* Line Array
  + MDN:String:32 🡪 The specific telephone number.
  + Account Mode:ENUM 🡪 One of (‘PRE’,’PRE SHARED’,’POST’,’POST SHARED’)
  + Usage Detail Array
    - Calldate:Datetime 🡪 Start Date/Time usage occurred.
    - Ratetype:ENUM 🡪 One of (‘INTERSTATE’,’DATA’,’SMS’,’TOLLFREE’,’WIRELESS\_611’,’WIRELESS\_411’)
    - TN:String:32 🡪 The specific telephone number.
    - City:String:128
    - State:String:2
    - Other TN:String:32 🡪 The calling/called party telephone number
    - Other City:String:128 🡪 The calling/called party city
    - Other State:String:2
    - Inbound Flag:Boolean 🡪 If True customer was called, otherwise call made to Other
    - Usage:Decimal:10.4 🡪 One of:
      * Usage in minutes if Rate Type = ‘INTERSTATE’,’TOLLFREE’,’WIRELESS\_411’,’WIRELESS\_611’
      * Usage in text messages if Rate Type = ‘SMS’
      * Usage in kilobytes if Rate Type = ‘DATA’
    - Carrier Code:String:64 🡪 Carrier usage was incurred upon
  + Usage Summary Array
    - Cycle Date:Date 🡪 Billing cycle date usage occurred.
    - Ratetype:ENUM 🡪 One of (‘INTERSTATE’,’DATA’,’SMS’,’TOLLFREE’,’WIRELESS\_611’,’WIRELESS\_411’)
    - Usage:Decimal:10.4 🡪 One of:
      * Usage in minutes if Rate Type = ‘INTERSTATE’,’TOLLFREE’,’WIRELESS\_411’,’WIRELESS\_611’
      * Usage in text messages if Rate Type = ‘SMS’
      * Usage in kilobytes if Rate Type = ‘DATA’
    - Carrier Code:String:64 🡪 Carrier usage was incurred upon
  + Plan Allowance Summary Array
    - Cycle Date:Date 🡪 Billing cycle date allowance in force.
    - Ratetype:ENUM 🡪 One of (‘INTERSTATE’,’DATA’,’SMS’,’TOLLFREE’,’WIRELESS\_611’,’WIRELESS\_411’)
    - Allowance:Decimal:10.4 🡪 One of:
      * Plan allowance in minutes if Rate Type = ‘INTERSTATE’,’TOLLFREE’,’WIRELESS\_411’,’WIRELESS\_611’
      * Plan allowance in text messages if Rate Type = ‘SMS’
      * Plan allowance in kilobytes if Rate Type = ‘DATA’
    - Carrier Code:String:64 🡪 Carrier usage was incurred upon
  + Overage Summary Array
    - Cycle Date:Date 🡪 Billing cycle date usage occurred.
    - Ratetype:ENUM 🡪 One of (‘INTERSTATE’,’DATA’,’SMS’)
    - Usage:Decimal:10.4 🡪 One of:
      * Usage in minutes if Rate Type = ‘INTERSTATE’
      * Usage in text messages if Rate Type = ‘SMS’
      * Usage in kilobytes if Rate Type = ‘DATA’
    - Allowance:Int:8 🡪 One of:
      * Minutes if Rate Type = ‘INTERSTATE’
      * Text messages if Rate Type = ‘SMS’
      * Kilobytes if Rate Type = ‘DATA’
    - Overage:Decimal:10.4 🡪 One of:
      * Overage in minutes if Rate Type = ‘INTERSTATE’
      * Overage in text messages if Rate Type = ‘SMS’
      * Overage in kilobytes if Rate Type = ‘DATA’
    - Carrier Code:String:64 🡪 Carrier usage was incurred upon

Return Sub Codes

* 0x1915

### Get Consumer Statement

Function to lookup and retrieve customer billing statements

Inputs

* Customer ID:Int:8 🡪 The Customer ID to be retrieved.
* Statement Lookup, must be one of the following
  + (Optional) Statement ID:String:32 🡪 ID returned from Get Consumer Statements
  + (Optional) Billing Cycle Date:Date 🡪 Date of billing cycle to return.

Outputs

* PDF Document:Binary 🡪 Binary blob data in PDF format to be streamed into a viewer.

Return Sub Codes

* 0x1925

### Get Consumer Statements

Function to lookup and retrieve customer billing statements

Inputs

* Customer ID:Int:8 🡪 The Customer ID to be retrieved.
* (Optional) Billing Cycle Date:Date 🡪 Date of billing cycle to return.

Outputs

* Statement Array
  + Statement ID:String:32
  + Due Date:Datetime
  + Created Date:Datetime
  + Invoice Amount:Decimal:10.4
  + Credit Memo Amount:Decimal:10.4
  + Payment Amount:Decimal:10.4
  + Amount Due:Decimal:10.4
  + Previous Balance:Decimal:10.4
  + Billing Cycle Date:Date

Return Sub Codes

* 0x1930

### Get Consumer Active Products

Function to lookup and retrieve the customer’s active devices, associated accessories, and MRC billable products.

Inputs

* Customer ID:Int:8 🡪 The ID of the consumer account to be retrieved.

Outputs

* <standard response message block>
* Billing Account Array (Can be 1 Post/Post\_Shared and up to 10 for PRE lines)
  + Billing Mode: ENUM 🡪 One of ‘PRE’,’POST’,’POST\_SHARED’.
  + Device Array (For PRE only 1, for Post/PostShared up to 5)
    - MDN:String:32 🡪 The specific telephone number.
    - Device ID:String:128 🡪 The device ESN/SIM.
    - Carrier ID:String:128 🡪 The carrier ID
    - Product ID:Int:8 🡪 The reference ID for the product
    - Short Name:String:32 🡪 Short reference name
    - Description:String:128 🡪 The product description in detail
    - Inventory Subtype:ENUM 🡪 Product subtype code, (‘SMART’,’QWERTY’,’FLIP’,’SIM’,’USB’,’MIFI’)
    - Billing Type:ENUM 🡪 One of NRC (non-recurring charge) or MRC (monthly recurring charge)
    - Price Paid:Decimal:5.4 🡪 The price paid by the consumer
    - Order ID:Int:8 🡪 The Order ID this device is associated with.
    - Order Line Item ID:Int:8 🡪 The line item on the order this device is associated with.
    - Primary Line:Boolean:1 🡪 True if this is the primary device on the account.
    - Plan Array (Can be 0 to Many depending on line type of Pre, Post, or PostShared)
      * Product ID:Int:8 🡪 The reference ID for the product
      * Short Name:String:32 🡪 Short reference name
      * Description:String:128 🡪 The product description in detail
      * Inventory Subtype:ENUM 🡪 Subtype code, (‘VOICE’,’DATA’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’,’VOICE\_SMS\_DATA’)
      * Billing Type:ENUM 🡪 One of NRC or MRC
      * Price Paid:Decimal:5.4 🡪 The price paid by the consumer
      * Order ID:Int:8 🡪 The Order ID this device is associated with.
      * Order Line Item ID:Int:8 🡪 The line item on the order this device is associated with.
      * (Optional) Allowance Type:ENUM 🡪 One of ‘VOICE’,’DATA’,’SMS’.
      * (Optional) Allowance:Int 🡪 Allowance amount (VOICE=minutes, DATA=kilobytes, SMS=messages)
    - Fee Array (Can be 0 to Many depending on life type)
      * Product ID:Int:8 🡪 The reference ID for the product
      * Short Name:String:32 🡪 Short reference name
      * Description:String:128 🡪 The product description in detail
      * Inventory Subtype:ENUM 🡪 Subtype code, (‘ADD\_A\_LINE,’DATA\_ADD\_A\_LINE’,’SHIPPING’,’ACTIVATION’)
      * Billing Type:ENUM 🡪 One of NRC or MRC
      * Price Paid:Decimal:5.4 🡪 The price paid by the consumer
      * Order ID:Int:8 🡪 The Order ID this device is associated with.
      * Order Line Item ID:Int:8 🡪 The line item on the order this device is associated with.
    - Accessory Array (Can be 0-to-Many for all line types)
      * Product ID:Int:8 🡪 The reference ID for the product
      * Short Name:String:32 🡪 Short reference name
      * Description:String:128 🡪 The product description in detail
      * Inventory Subtype:ENUM 🡪 Product subtype code, (‘BATTERY’,‘BUNDLE’,’CASE’,’CHARGER’,HEADSET’,’MEMORY’)
      * Billing Type:ENUM 🡪 One of NRC or MRC
      * Price Paid:Decimal:5.4 🡪 The price paid by the consumer
      * Order ID:Int:8 🡪 The Order ID this device is associated with.
      * Order Line Item ID:Int:8 🡪 The line item on the order this device is associated with.
    - Extended Service Plan Array (Can only be 1 per device for Post, PostShared only)
      * Product ID:Int:8 🡪 The reference ID for the product
      * Short Name:String:32 🡪 Short reference name
      * Description:String:128 🡪 The product description in detail
      * Inventory Subtype:ENUM 🡪 Product subtype code
      * Billing Type:ENUM 🡪 One of NRC or MRC
      * Price Paid:Decimal:5.4 🡪 The price paid by the consumer
      * Order ID:Int:8 🡪 The Order ID this device is associated with.
      * Order Line Item ID:Int:8 🡪 The line item on the order this device is associated with.

Return Sub Codes

* 0x1935

### Add Consumer Note

Use to add a free-form note to a consumer account. Note: If defined, note will be passed to configured billing system.

Inputs

* Customer ID:Int:8 🡪 The Consumer ID the applies too.
* Subject:String:256 🡪 Subject of the Note.
* Note:Text 🡪 The Note.
* User ID:Int:8 🡪 The ID of the User adding the Note.

Outputs

* <standard response message block>

Return Sub Codes

* 0x1640 🡪 Success
* 0x1641 🡪 Internal API Error, escalate issue to support
* 0x1642 🡪 Invalid Parameter
* 0x1643 🡪 Passed Consumer ID not found
* 0x1644 🡪Passed User ID not found

### Change Consumer Note

Use to change an existing note associated with a consumer. Note: If defined, note will be passed to configured billing system

Inputs

* Customer ID:Int:8 🡪 The Consumer ID the note is attached too.
* Note ID:Int:8 🡪 The ID of the note to change.
* Note:Text 🡪The Note.
* User ID:Int:8 🡪The ID of the User changing the Note.

Outputs

* <standard response message block>

Return Sub Codes

* 0x1660 🡪 Success
* 0x1661 🡪 Internal API Error, escalate to support
* 0x1662 🡪 Invalid Parameter
* 0x1663 🡪 Consumer ID not found
* 0x1664 🡪 User ID not found
* 0x1665 🡪 Note ID not found
* 0x1666 🡪 Note is not of type Order

### Get Consumer Note

Use to get the details of a note.

Inputs

* Consumer ID:Int:8 🡪 The Consumer ID to get notes for.
* (Optional) Note ID:Int:8 🡪 If provided than only this notes information is returned.

Outputs

* <standard response message block>
* Note Array
  + ID:Int:8 🡪 ID of the Note
  + Subject:String:256
  + Type:String:32 🡪 Always “Account”
  + Note History Array
    - Created Date:Date
    - Username:String:32
    - Note:Text

Return Sub Codes

* 0x1680 🡪 Success
* 0x1681 🡪 Internal API Error, escalate to support
* 0x1682 🡪 Invalid Parameter
* 0x1683 🡪 Consumer ID not found
* 0x1684 🡪 Note ID not found

### Get Consumer Payments

Function to lookup and retrieve customer payments.

Inputs

* Customer ID:Int:8 🡪 The Consumer ID to get notes for.
* (Optional) Date From:DateTime 🡪If provided all payment records with a date after or equal are returned.
* (Optional) Date To:DateTime 🡪 If provided all payment records with a date before or equal are returned.

Outputs

* <standard response message block>
* Payment Array
  + ID:Int:8 🡪 ID of the Payment
  + Amount: Decimal:8.4 🡪 The amount of the transaction
  + Created Date:Date 🡪 The date the payment was created
  + Source:ENUM 🡪 One of AUTO\_DEBIT, INHOUSE, WEB
  + Method: ENUM 🡪 One of CC, CASH, CHECK, MONEY\_ORDER

Return Sub Codes

* 0x1720 🡪 Success
* 0x1721 🡪 Internal API Error, escalate to support
* 0x1722 🡪 Invalid Parameter
* 0x1723 🡪 Customer ID not found

## Back Office API Group

The Back Office API group is a collection of routines for handling scheduled tasks, reports, and data extracts.

### Get Defined Reports

This function call returns a list of all defined reports and parameter requirements for a given account.

Inputs

* Account Identifier has to be one of the following
  + (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID, Agent ID.
  + (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup by Account Identifier.
* (Optional) Report ID:Int:8 🡪If provide pull details for this report
* Report Type:ENUM 🡪 One of OPERATIONS, FINANCIAL, SALES, ALL

Outputs

* <standard response message block>
* Report List Array
  + Report ID:Int:8
  + Report Type:ENUM
  + Report Name:String:32
  + Report Description:String:128
  + Report Pageable:Boolean:True/False 🡪 If the report can be paged
  + Report Path:String:128
  + Report Parameters Array
    - Name:String:32
    - Mandatory Flag:Boolean
    - Type:ENUM: One of STRING, INTEGER, DATE, ARRAY

Return Sub Codes

* 0x0520

### Get Report Detail

Generic mechanism for remote invocation of defined reports. If report type is HTML and paging flag is set to true the method will return a paging cache key which is used by the getReportDetailPaged function for getting the HTML data.

**Note**: Function will return an error if parameters supplied and associated report does not require the parameters.

Inputs

* Account ID:Int:8
* Report ID:Int:8
* Report Output Type:ENUM 🡪 One of HTML, PDF, EXCEL
* Paging Flag:Boolean(True/False or 0/1) 🡪 Provides the ability to page HTML reports
* Parameters Array
  + Name:String:32 🡪 Parameter Name
  + Report Parameter Type:ENUM 🡪 One of STRING, INTEGER, DATE, ARRAY
  + String Value:String Array 🡪 String Values
  + Integer Value:Integer Array 🡪 Integer Values
  + Date Value:String Array 🡪 DateTime, Format must be YYYY-MM-DDTHH:MM:SS (T = Letter T)

Outputs

* <standard response message block>
* Report Output Type:ENUM 🡪 One of HTML, PDF, EXCEL
* Total Rows:int 🡪 Total Number of Rows in the Report. Note: This count includes header rows
* Paging Cache Key:String 🡪 Unique Key required by getReportDetailPaged() API Call
* HTML Report:String 🡪 Populated if report type is HTML
* PDF Report:Byte Array 🡪 Populated if report type is PDF
* EXCEL Report:Byte Array 🡪 Populated if report type id EXCEL

Return Sub Codes

* 0x0540

### Get Report Detail Paged

This function will return specific rows of an HTML report. This function only works with HTML generated reports.

Inputs

* Account ID:Int:8
* Paging Cache Key:String:16 🡪 Returned by getReportDetail() function if paging flag was true
* First Row:Int:8 🡪 First row to start data pull
* Max Rows:Int:8 🡪 Total number of rows to pull

Outputs

* <standard response message block>
* Report Output Type:ENUM 🡪 Will always be HTML
* Paging Cache Key:String
* HTML Report:String 🡪 Requested rows in HTML format

Return Sub Codes

* 0x1620

## Wholesale API Group

### Build Wholesale Order

This begins the process of creating a wholesale order. Basically this function generates empty data structures in preparation for subsequent **AddWholesaleOrderItem** function calls; which will require the WholesaleOrderID.

Inputs

* Dealer ID:Int:8🡪 The ID of the dealer this order will be placed for.

Outputs

* <standard response message block>
* WholesaleOrderID:Int:8

Return Sub Codes

* 0x1C00

### Add Wholesale Order Item

This function is used to add inventory to a wholesale order.

**Note**: A quantity check will be made by this function to ensure adequate inventory is available.

Inputs

* WholesaleOrderID:Int:8 🡪 The ID of the wholesale order being created.
* Inventory Identifier has to be one of:
  + (Optional) Product ID:Int:8 🡪 Product ID of the specific inventory type to add to the order.
  + (Optional) Custom Key:String:32 🡪 Client defined value input at inventory load. Can be used as a model mapping field. I.E. Model SKU
* Add Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Autopick:**0x01** 🡪 Let system pick the ESN/SIMs randomly from available inventory.
  + (Optional) Provided:**0x02** 🡪 Client provides the ESN/SIM values to be selected from inventory.
    - **Note**: ESN/SIM values must be present and available within inventory and match the ProductID/Custom Key provided.
    - Device ID Array
      * Device ID:String:128 🡪 ESN/SIM value
* Quantity:Int:5 🡪 The quantity to auto pick or provided by client. Device ID array quantity must match this value.

Outputs

* <standard response message block>
* LineItemID:Int:4
* Product ID:Int:8
* Custom Key:String:32

Return Sub Codes

* 0x1D00

### Remove Wholesale Order Item

Use to reverse line items already part of the order build.

Inputs

* WholesaleOrderID:Int:8 🡪 The ID of the wholesale order being created.
* LineItemID:Int:4 🡪 The line item to remove.

Outputs

* <standard response message block>

Return Sub Codes

* 0x1E00

### Place Wholesale Order

Records the dealer order and queues the job for provisioning/fulfillment.

**Note**: Provisioning workflow type is configured at the system level for either of automated activation or processing via Ready Wireless fulfillment/provisioning system.

Inputs

* WholesaleOrderID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x1F00

### Cancel Wholesale Order

For canceling a wholesale order. Can succeed on orders in “BUILD”, “PLACED”, “HOLD” states. Once processing begins the order cannot be stopped.

Inputs

* WholesaleOrderID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x2000

### Get Wholesale Order

Returns the wholesale order details and associated ESNs/SIMs.

**Note**: If order state is “PROCESSING” the device ID arrays will not be returned…only the order summary information. The processed quantity will change with each additional call. Once the state is “COMPLETE” the results are final and can be relied upon to remain constant. It is possible for the quantity processed to be less than the quantity ordered if there is errors occur during the activation step.

Inputs

* WholesaleOrderID:Int:8

Outputs

* <standard response message block>
* WholesaleOrder ID:Int:8 🡪 Order ID specific to this order
* Type:String:16 🡪 Order type (NEW, CANCEL, CHANGE)
* State:String:16 🡪 Order state (BUILD, PLACED, HOLD, PROCESSING, SHIPPED, COMPLETE, CLOSED, CANCELED, AUTO CANCELED)
* BuildDate:Date:32 🡪 Date order build started. Format: YYYY-MM-DD HH24:MM:SS
* PlacedDate:Date:32 🡪 Date order was placed. Format: YYYY-MM-DD HH24:MM:SS
* CurrentStateDate:Date:32 🡪 Date the current state started. Format: YYYY-MM-DD HH24:MM:SS
* Order Line Item Array
  + Inventory Identifier, one of:
    - Product ID:Int:8
    - Custom Key:String:32
  + Add Flag:Int:2 🡪 Must be one of the following: Autopick:**0x01**, Provided:**0x02**
  + (Optional) Device ID Array
    - Device ID:String:128 🡪 ESN/SIM value
  + Quantity Ordered:Int:5
  + Quantity Processed:Int:5 🡪 Number applied to order. May not match quantity ordered.

Return Sub Codes

* 0x2100

### Add Dealer

The mechanism for defining a dealer within the system. Once defined, order may be placed for the dealer.

Inputs

* Name:String:128 🡪 Dealer specific name.
* Account ID:String:128 🡪 The dealer code.
* Dealer Carrier Codes Array 🡪 Carriers support by this dealer.
  + Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
* Address Line 1:String:128
* Address Line 2:String:128
* City:String:64
* State:String:2
* Zip:Int:5
* (Optional) Zip4:Int:4
* Contact Number:String:16
* Contact Email Address:String:128

Outputs

* <standard response message block>
* Dealer ID:Int:8 🡪 Used for all subsequent dealer related function calls.

Return Sub Codes

* 0x2200

### Change Dealer

Function to change dealer specific information.

Inputs

* Dealer ID:Int:8 🡪 The dealer ID to change.
* (Optional) Name:String:128 🡪 Dealer specific name.
* (Optional) Account ID:String:128 🡪 The dealer code.
* (Optional) Dealer Carrier Codes Array 🡪 Carriers support by this dealer.
  + Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
* (Optional) Address Line 1:String:128
* (Optional) Address Line 2:String:128
* (Optional) City:String:64
* (Optional) State:String:2
* (Optional) Zip:Int:5
* (Optional) Zip4:Int:4
* (Optional) Contact Number:String:16
* (Optional) Contact Email Address:String:128

Outputs

* <standard response message block>

Return Sub Codes

* 0x2300

### Close Dealer

Use this function to close a dealer account.

Inputs

* Dealer ID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x2400

### Get Dealers Lookup

Use to get a listing of all dealers configured (active only).

**Note**: Can be used to retrieve specific dealer account by ID or Name.

Inputs

* (Optional) Dealer ID:Int:8 🡪 Can be used to filter a specific dealer rather than return the entire list.
* (Optional) Name:String:128 🡪 Use to filter by dealer name.

Outputs

* <standard response message block>
* Dealer Array
  + Dealer ID:Int:8 🡪 The dealer ID to change.
  + Account ID:String:128 🡪 The dealer code.
  + Name:String:128 🡪 Dealer specific name.
  + Dealer Carrier Codes Array 🡪 Carriers support by this dealer.
    - Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
  + Address Line 1:String:128
  + Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + Zip4:Int:4
  + Contact Number:String:16
  + Contact Email Address:String:128

Return Sub Codes

* 0x2500

## 

## Product API Group

### Get Products

Returns product information from the client configured product catalog.

This function can return product group or “package” information as well. Product groups roll-up multiple products into one Product ID that can be atomically applied in a single call.

This function also can provide a list of the defined Pick Groups within the client’s configuration. Pick Groups define a single generic product that can be fulfilled by any of the associated products. For example, a “Flip Phone” Pick Group may have many associated Product IDs that can all be used to fulfill a generic flip phone.

Inputs

* (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID, Agent ID. Otherwise defaults to the products defined at the Client level.
* (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup by Account Identifier.
* (Optional) Product ID:Int:8 🡪 If set, only returns information for the product ID.
* (Optional) Carrier:String:128 🡪 If set, filters by the carrier name supplied
  + (‘SPRINT’,’VERIZON’,’TMOBILE’)
  + A composite list can also be provided. I.E. ‘SPRINT+VERIZON’
* (Optional) Billing Mode:ENUM 🡪 If set, filters by the billing mode supplied
  + (‘PRE’,’POST’,’POST\_SHARED’).
* (Optional) Inventory Type:String:128 🡪 If set, filters by the product type
  + (‘DEVICE’,’AIRTIME’,’ACCESSORY’,’EXTENDED’,’FEE’,’TAX’,’CREDIT’,’PROMO’)
  + A composite list can also be provided. I.E. ‘AIRTIME+EXTENDED’
* (Optional) Inventory Subtype:ENUM 🡪 If set, filters by the product subtype. The inventory type field must be set if this field is used.
  + ‘FLIP,’QWERTY’,’SMART’,’SIM’,’USB’,’MIFI’
  + ’DATA’,’VOICE’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’
  + ’ADD\_A\_LINE’,’DATA\_ADD\_A\_LINE’,’ACTIVATION’,’SHIPPING’,’DEPOSIT’
* (Optional) Mode Flag:Int 🡪 Filters by product mode. If set, must be one of:
  + (Optional) All Products: **0x01** (default) 🡪 Retrieves all products, product groups, pick groups defined.
  + (Optional) Base Products Only: **0x02** 🡪 Does not return product groups or pick groups.
  + (Optional) Product Groups Only: **0x03** 🡪 Only the product groups.
  + (Optional) Pick Groups Only: **0x04** 🡪 Only the pick groups.
  + (Optional) Base and Products Groups: **0x05** 🡪 Only return base products and product groups.
* (Optional) Details Flag:Int 🡪 Limits returned values. If set, must be one of:
  + (Optional) Product: **0x01** (default) 🡪 Returns only the base product info and optional associations.
  + (Optional) eCommerce: **0x02** 🡪 Includes display description, image URLs, feature lists, dependencies.
* (Optional) Zip:Int:5 🡪 If set, taxes and fees will be calculated.
* (Optional) Zip4:Int:4 🡪 If set, Zip must be set also.

Outputs

* <standard response message block>
* Product Array
  + Product ID:Int:8 🡪 The reference ID for the product
  + Short Name:String:32 🡪 Short reference name
  + Description:String:128 🡪 The product description in detail
  + Detailed Description:String:4096 🡪 Detailed product content (optionally HTML formatted)
  + User Guide URI Location:String:512 🡪 Relative URI for user guide location on web server (typically this is a link to a PDF formatted document)
  + Type:String:32 🡪 One of PRODUCT, PRODUCT GROUP, PICK GROUP
  + Inventory Type:ENUM 🡪 Product type code.
    - (‘DEVICE’,‘AIRTIME’,’ACCESSORY’,’EXTENDED’,’PROMO’)
  + Inventory Subtype:ENUM 🡪 Product subtype code.
    - (‘SMART’,’QWERTY’,’FLIP’,’SIM’,’USB’,’MIFI’)
    - (‘VOICE’,’DATA’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’)
    - (’ADD\_A\_LINE’,’DATA\_ADD\_A\_LINE’,’ACTIVATION’,’SHIPPING’,’DEPOSIT’)
  + Inventory Count:Int 🡪 Count of current products available.
  + Inventory Minimum Dispense Limit 🡪 The level inventory can deplete before no longer available.
  + Billing Mode:ENUM 🡪 One of PRE or POST.
  + Billing Type:ENUM 🡪 One of NRC (non-recurring charge) or MRC (monthly recurring charge)
  + Base Rate:Decimal:5.4 🡪 The default product price
  + (Optional) Overage Rate:Decimal:5.4 🡪 The “per unit” cost of overage
  + (Optional) Taxes:Decimal:5.4 🡪 Calculated if zip/zip4 provided.
  + Carrier:String:128 🡪 The carrier for the product (SPRINT,VERIZON,TMOBILE,ALL).
  + Color Code:String:32 🡪 The “color” assigned to this carrier to group carrier products.
  + Color Hex Code:String:16 🡪 The RGB hex code of the color assigned.
  + Custom Key:String:32 -> Client defined product key
  + Custom1:String:64 🡪 Client defined field
  + Custom2:String:64 🡪 Client defined field
  + Custom3:String:64 🡪 Client defined field
  + (Optional) Grade Type:ENUM 🡪 One of NEW,USED,MIXED.
  + (Optional) Allowance Type:ENUM 🡪 One of ‘VOICE’,’DATA’,’SMS’.
  + (Optional) Allowance:Int 🡪 Allowance amount (VOICE=minutes, DATA=kilobytes, SMS=messages)
  + (Optional) Associated Products Array
    - Product ID:Int:8 🡪 the ID of the associated products if this is a product or pick group.
  + (Optional) Dependent/Minimum Products Array
    - Product ID:Int:8 🡪 the ID of the dependent product that must be sold/associated simultaneously with this primary product.
    - Inventory Type:ENUM 🡪 Product type code.
      * (‘DEVICE’,‘AIRTIME,’ACCESSORY’,’EXTENDED’,’FEE’,’PROMO’)
    - Inventory Subtype:ENUM 🡪 Product subtype code.
      * (‘SMART’,’QWERTY’,’FLIP’,’SIM’,’USB’,’MIFI’)
      * (‘VOICE’,’DATA’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’)
      * (’ADD\_A\_LINE’,’DATA\_ADD\_A\_LINE’,’ACTIVATION’,’SHIPPING’,’DEPOSIT’)
    - Dependency Type:ENUM 🡪 One of 'REQUIRED’, 'REQUIRE\_ONE’, ’REQUIRE\_SHARED’ ,'RESTRICTED',’OPTIONAL’)
      * REQUIRED = This product must be sold/associated with the primary product.
      * REQUIRE\_ONE = For every product in the array marked “REQUIRE\_ONE” there must be one of these products sold/associated with the primary product on the order.
      * REQUIRE\_SHARED = This product is required for shared devices. This is not required for the primary device on an account.
      * RESTRICTED = Any product with this flag cannot be sold/associated with the primary product.
      * OPTIONAL = Default behavior, revert to normal selection processing.
    - Group Type:ENUM 🡪 One of (‘DEVICE’,’AIRTIME’,’ACCESSORY’,’EXTENDED’,’FEE’,’PROMO’)
    - Group Tag: String: 32 🡪 A code that groups like dependencies together when there are more than one configuration of the same Group Type. For example, four ACCESSORY dependencies for a device with “REQUIRE ONE” could be grouped two each with different group tags; requiring two accessories to be selected rather than just one.
  + (Optional) Feature Array
    - (Optional) Feature Key:String:64 🡪 Common feature key. I.E. ‘Data’,’Voice Commands’,’Screen’
    - (Optional) Feature Display Description:String:128 🡪 Descriptive name to display on site/portal.
    - (Optional) Feature Value:String:1024 🡪 Detailed feature text (optionally HTML formatted).
  + (Optional) Image URLs (pointers to images stored on web server)
    - Thumb Front URI:String:512 🡪 Relative path to image location.
    - Thumb Left URI:String:512 🡪 Relative path to image location.
    - Thumb Right URI:String:512 🡪 Relative path to image location.
    - Thumb Rear URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Front URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Left URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Right URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Rear URI:String:512 🡪 Relative path to image location.
    - Alternate Image Front URI:String:512 🡪 Relative path to image location.
    - Alternate Image Left URI:String:512 🡪 Relative path to image location.
    - Alternate Image Right URI:String:512 🡪 Relative path to image location.
    - Alternate Image Rear URI:String:512 🡪 Relative path to image location.

Return Sub Codes

* 0x2600

Example Flow

* resultSet = GetProducts(1033); /\* Returns array of one product matching ID 1033, no taxes \*/

### Get Product Features

Returns the product feature array, detailed product information, and dependencies.

Inputs

* (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID. Otherwise defaults to the products defined at the Client level.
* (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup by Account Identifier.
* (Optional) Product ID:Int:8 🡪 If set, only returns information for the product ID.
* (Optional) Feature Key:String:64 🡪 If set, limits the returned data to this key.
* (Optional) Inventory Type:ENUM 🡪 If set, filters by the product type
  + (‘DEVICE’,’AIRTIME’,’ACCESSORY’,’FEE’,’TAX’,’CREDIT’,’PROMO’)

Outputs

* <standard response message block>
* Product Array
  + Product ID:Int:8 🡪 The reference ID for the product
  + Detailed Description:String:4096 🡪 Detailed product content (optionally HTML formatted)
  + User Guide URI Location:String:512 🡪 Relative URI for user guide location on web server (typically this is a link to a PDF formatted document)
  + (Optional) Dependent/Minimum Products Array
    - Product ID:Int:8 🡪 the ID of the dependent product that must be sold/associated simultaneously with this primary product.
    - Inventory Type:ENUM 🡪 Product type code.
      * (‘DEVICE’,‘AIRTIME,’ACCESSORY’,’EXTENDED’,’FEE’,’PROMO’)
    - Inventory Subtype:ENUM 🡪 Product subtype code.
      * (‘SMART’,’QWERTY’,’FLIP’,’SIM’,’USB’,’MIFI’)
      * (‘VOICE’,’DATA’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’)
      * (’ADD\_A\_LINE’,’DATA\_ADD\_A\_LINE’,’ACTIVATION’,’SHIPPING’,’DEPOSIT’)
    - Dependency Type:ENUM 🡪 One of 'REQUIRED’, 'REQUIRE\_ONE’, ’REQUIRE\_SHARED’ ,'RESTRICTED',’OPTIONAL’)
      * REQUIRED = This product must be sold/associated with the primary product.
      * REQUIRE\_ONE = For every product in the array marked “REQUIRE ONE” there must be one of these products sold/associated with the primary product on the order.
      * REQUIRE\_SHARED = This product is required for shared devices. This is not required for the primary device on an account.
      * RESTRICTED = Any product with this flag cannot be sold/associated with the primary product.
      * OPTIONAL = Default behavior, revert to normal selection processing.
    - Group Type:ENUM 🡪 One of (‘DEVICE’,’AIRTIME’,’ACCESSORY’,’EXTENDED’,’FEE’,’PROMO’)
    - Group Tag: String: 32 🡪 A code that groups like dependencies together when there are more than one configuration of the same Group Type. For example, four ACCESSORY dependencies for a device with “REQUIRE ONE” could be grouped two each with different group tags; requiring two accessories to be selected rather than just one.
  + (Optional) Feature Array
    - (Optional) Feature Key:String:64 🡪 Common feature key. I.E. ‘Data’,’Voice Commands’,’Screen’
    - (Optional) Feature Display Description:String:128 🡪 Descriptive name to display on site/portal.
    - (Optional) Feature Value:String:1024 🡪 Detailed feature text (optionally HTML formatted).
  + (Optional) Image URLs (pointers to images stored on web server)
    - Thumb Front URI:String:512 🡪 Relative path to image location.
    - Thumb Left URI:String:512 🡪 Relative path to image location.
    - Thumb Right URI:String:512 🡪 Relative path to image location.
    - Thumb Rear URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Front URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Left URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Right URI:String:512 🡪 Relative path to image location.
    - Standard Size Image Rear URI:String:512 🡪 Relative path to image location.
    - Alternate Image Front URI:String:512 🡪 Relative path to image location.
    - Alternate Image Left URI:String:512 🡪 Relative path to image location.
    - Alternate Image Right URI:String:512 🡪 Relative path to image location.
    - Alternate Image Rear URI:String:512 🡪 Relative path to image location.

Return Sub Codes

* 0x2700

Example Flow

* resultSet = GetProductFeatures(1033); /\* Returns information for the 1033 product \*/

### Get Product Dependencies

Returns product dependency information from the configured product catalog.

Filters on this API call can be used to restrict the dependency list to just the inventory types being worked on.

Inputs

* (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID, Agent ID. Otherwise defaults to the products defined at the Client level.
* (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup by Account Identifier.
* (Optional) Product Array
  + Product ID:Int:8 🡪 Get dependencies for this product ID.
* (Optional) Dependency Group:ENUM 🡪 Return the dependency information for this dependency group
  + (‘DEVICE’,’PLAN’,’ACCESSORY’,’EXTENDED’,’PROMO’)

Outputs

* <standard response message block>
* Product Array
  + Product ID:Int:8 🡪 The reference ID for the product
  + Inventory Type:ENUM 🡪 Product type code.
    - (‘DEVICE’,‘AIRTIME’,’ACCESSORY’,’EXTENDED’,’PROMO’)
  + Inventory Subtype:ENUM 🡪 Product subtype code.
    - (‘SMART’,’QWERTY’,’FLIP’,’SIM’,’USB’,’MIFI’)
    - (‘VOICE’,’DATA’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’)
    - (’ADD\_A\_LINE’,’DATA\_ADD\_A\_LINE’,’ACTIVATION’,’SHIPPING’,’DEPOSIT’)
  + (Optional) Dependent/Minimum Products Array
    - Product ID:Int:8 🡪 the ID of the dependent product that must be sold/associated simultaneously with this primary product.
    - Inventory Type:ENUM 🡪 Product type code.
      * (‘DEVICE’,‘AIRTIME,’ACCESSORY’,’EXTENDED’,’FEE’,’PROMO’)
    - Inventory Subtype:ENUM 🡪 Product subtype code.
      * (‘SMART’,’QWERTY’,’FLIP’,’SIM’,’USB’,’MIFI’)
      * (‘VOICE’,’DATA’,’SMS’,’SMS\_AND\_DATA’,’SMS\_AND\_VOICE’)
      * (’ADD\_A\_LINE’,’DATA\_ADD\_A\_LINE’,’ACTIVATION’,’SHIPPING’,’DEPOSIT’)
    - Dependency Type:ENUM 🡪 One of 'REQUIRED’, 'REQUIRE\_ONE’, ’REQUIRE\_SHARED’ ,'RESTRICTED',’OPTIONAL’)
      * REQUIRED = This product must be sold/associated with the primary product.
      * REQUIRE\_ONE = For every product in the array marked “REQUIRE ONE” there must be one of these products sold/associated with the primary product on the order.
      * REQUIRE\_SHARED = This product is required for shared devices. This is not required for the primary device on an account.
      * RESTRICTED = Any product with this flag cannot be sold/associated with the primary product.
      * OPTIONAL = Default behavior, revert to normal selection processing.
    - Group Type:ENUM 🡪 One of (‘DEVICE’,’AIRTIME’,’ACCESSORY’,’EXTENDED’,’FEE’,’PROMO’)
    - Group Tag: String: 32 🡪 A code that groups like dependencies together when there are more than one configuration of the same Group Type. For example, four ACCESSORY dependencies for a device with “REQUIRE ONE” could be grouped two each with different group tags; requiring two accessories to be selected rather than just one.

Return Sub Codes

* 0x2750

Example Flow

* resultSet = GetProductDependencies(1033); /\* Returns array of one product matching ID 1033 \*/

## Service API Group

### Add Service Subscription

Use this API function to Opt-In or Opt-Out of a service or campaign feature.

Services like “Usage Alerts” must be configured within the system at either the Client or Partner account level.

Inputs

* Account ID:Int:8 🡪 The Consumer ID of the account to setup the subscription on.
* Subscription Mode:ENUM 🡪 One of (‘OPT\_IN’,’OPT\_OUT’).
* Must supply one of the following:
  + (Optional) Service ID: Int: 8 🡪 If set, is the Service ID to add this subscription towards. Overrides the Service Name if also provided.
  + (Optional) Service Name: String: 128 🡪 Text version of the Service ID. I.E. “Usage Alerts”.
* (Optional) MDN:String:16 🡪 If set, only setup the subscription for this MDN. Otherwise all MDN/Lines for the account will be setup.

Outputs

* <standard response message block>

Return Sub Codes

* 0x2150

### Close Service Subscription

Use this API function to remove the Opt-In or Opt-Out setup for this account and service.

Inputs

* Account ID:Int:8 🡪 The Consumer ID of the account to setup the subscription on.
* Must supply one of the following:
  + (Optional) Service ID: Int: 8 🡪 If set, is the Service ID to add this subscription towards. Overrides the Service Name if also provided.
  + (Optional) Service Name: String: 128 🡪 Text version of the Service ID. I.E. “Usage Alerts”.
* (Optional) MDN:String:16 🡪 If set, only setup the subscription for this MDN. Otherwise all MDN/Lines for the account will be setup.

Outputs

* <standard response message block>

Return Sub Codes

* 0x2160

### Change Service Subscription

Use this API function to modify the service subscription behaviors for this account and service.

Inputs

* Account ID:Int:8 🡪 The Consumer ID of the account to setup the subscription on.
* Subscription Mode:ENUM 🡪 One of (‘OPT\_IN’,’OPT\_OUT’).
* Must supply one of the following:
  + (Optional) Service ID: Int: 8 🡪 If set, is the Service ID to add this subscription towards. Overrides the Service Name if also provided.
  + (Optional) Service Name: String: 128 🡪 Text version of the Service ID. I.E. “Usage Alerts”.
* (Optional) MDN:String:16 🡪 If set, only setup the subscription for this MDN. Otherwise all MDN/Lines for the account will be setup.

Outputs

* <standard response message block>

Return Sub Codes

* 0x2170

### Get Service Subscription

Use this API function to modify the service subscription behaviors for this account and service.

Inputs

* Account ID:Int:8 🡪 The Consumer ID of the account to setup the subscription on.
* Must supply one of the following:
  + (Optional) Service ID: Int: 8 🡪 If set, is the Service ID to add this subscription towards. Overrides the Service Name if also provided.
  + (Optional) Service Name: String: 128 🡪 Text version of the Service ID. I.E. “Usage Alerts”.
* (Optional) MDN:String:16 🡪 If set, only lookup the subscription for this MDN. Otherwise all MDN/Lines for the account will be returned.

Outputs

* <standard response message block>
* Service Subscription Array
  + Service ID: Int: 8
  + Service Name: String: 128
  + Subscription Mode: ENUM 🡪 One of (‘OPT\_IN’, ‘OPT\_OUT’)
  + (Optional) MDN: String: 16 🡪 Only set if the subscription was set for a specific MDN, otherwise an account level response will be provided that covers all lines on the account.

Return Sub Codes

* 0x2180

### Get Services Lookup

Use this API function to lookup configured Services for this Client or Partner.

Inputs

* Account ID:Int:8 🡪 One of eCommerce ID, Partner ID, Dealer ID, Lifeline ID, Retail ID.
* Optional filters.
  + (Optional) Service ID: Int: 8 🡪 If set, return only this Service ID.
  + (Optional) Service Name: String: 128 🡪 If set, return on the Service ID that matches this Name.

Outputs

* <standard response message block>
* Services Array
  + Service ID: Int: 8
  + Service Name: String: 128
  + Service Description: String: 1024
  + Service Start: Datetime
  + Service Stop: Datetime
  + Service State: ENUM: 🡪 One of (‘ACTIVE’,’INACTIVE’)

Return Sub Codes

* 0x2190

## Provision API Group

### Bulk Load Inventory

This function loads inventory into a client’s “account”; which is configured manually within the system. Once loaded into the system the inventory can be used for subsequent wholesale or consumer based API calls.

This function will not create a Subscriber entry and sets the inventory state to LOADED.

**Note**: This call functions as a batch load so any failures during loading will be returned depending upon the load method selected.

Inputs

* (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID to force the load into the specific account. Otherwise defaults to the Client account.
* (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup account by Account Identifier.
* Load Method:Int:2 🡪 Flag to determine how errors are handled.
  + Fail All:**0x01** 🡪 Return a general error code and roll-back all changes.
  + Fail Individual:**0x02** 🡪 Return individual record failures and keep successful entries.
* Device Array
  + Inventory Identifier has to be one of:
    - (Optional) Product ID:Int:8 🡪 Product ID of the specific inventory type to add.
    - (Optional) Custom Key:String:32 🡪 Client defined value that maps to internal product ID.
  + Device Type:ENUM 🡪 One of CDMA, GSM.
  + Device ID:String:128 🡪 ESN/SIM identifier
  + Grade:String:16 🡪 Quality classification (NEW, REFURB, A, B, C, D, E)
  + (Optional) Model Serial Number:String:64 🡪 Serial number of device.
  + (Optional) MSL:Int:16 🡪 Device MSL code if CDMA.
  + (Optional) MSID:String:128 🡪 MSID code of the device if CDMA.
  + (Optional) Model Name:String:128 🡪 Model name for this device. I.E. Fascinate
  + (Optional) Model Brand:String:128 🡪 Model manufacturer. I.E. Samsung
  + (Optional) Model Number:String:128 🡪 Model number. I.E. 2, V2, etc.
  + (Optional) Serial Number:String:64 🡪 Serial number of device.
* Accessory Array
  + Inventory Identifier has to be one of:
    - (Optional) Product ID:Int:8 🡪 Product ID of the specific inventory type to add.
    - (Optional) Custom Key:String:32 🡪 Client defined value that maps to internal product ID.
  + Quantity:Int:8 🡪 The quantity of product available.
  + Grade:String:16 🡪 Quality classification (NEW, REFURB, A, B, C, D, E)

Outputs

* <standard response message block>
* Batch ID:Int:8 🡪 Used for subsequent bulk inventory function calls.
* Device ID Fail Array
  + Device ID:String:128 🡪 ESN/SIM identifier
  + Fail Reason:String:128

Return Sub Codes

* 0x2900

### Bulk Unload Inventory **(future)**

Inputs

Outputs

* <standard response message block>

Return Sub Codes

* 0x2A00

### Bulk Get Inventory **(future)**

Inputs

Outputs

* <standard response message block>

Return Sub Codes

* 0x2B00

### Bulk Activate Inventory **(future)**

Inputs

Outputs

* <standard response message block>

Return Sub Codes

* 0x2C00

### Bulk Deactivate Inventory **(future)**

Inputs

Outputs

* <standard response message block>

Return Sub Codes

* 0x2D00

### Get Device

Individual device lookup routine for obtaining state and detailed device information.

Inputs

* One of the following optional inputs must be specified
  + (Optional) Device ID:String:128 🡪 ESN/SIM identifier
  + (Optional) MDN:String:16 🡪 MDN
  + (Optional) Subscriber ID:Int 🡪 The “line” ID of the consumer.

Outputs

* <standard response message block>
* Device ID:String:128 🡪 The ESN/SIM for this device.
* Device Type:ENUM 🡪 One of CDMA, GSM.
* (Optional) MDN:String:32 🡪 If the device is active the MDN assigned.
* (Optional) MSL:String 16 🡪 If defined, the MSL code
* (Optional) MSID:String:128 🡪 If defined, the MSID code.
* Product ID:Int:8
* (Optional) Custom Key:String:32 🡪 If provided by client/vendor.
* Inventory State:ENUM 🡪 ASSIGNED, AVAILABLE, LOADED, UNAVAILABLE, ERROR
* (Optional) Subscriber State:ENUM 🡪 PLACED, CANCELED, HOLDING, ACTIVE, INACTIVE, ERROR, SUSPENDED.
* (Optional) State Error:String:1024 🡪 Error message returned from provider/system. Populated if subscriber or inventory state is ERROR.
* (Optional) Activation Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, if device is active.
* (Optional) Deactivation Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, if device is inactive.
* (Optional) Suspend Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, if device is suspended.
* (Optional) Restore Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, if device has been restored from suspended.
* (Optional) Load Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, when device was loaded.
* (Optional) Assigned Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, if device assigned to order.
* (Optional) Available Date:Date:32 🡪 Format: YYYY-MM-DD HH24:MM:SS, if device moves from unavailable to available.
* Grade:String:16 🡪 Quality classification (NEW, REFURB, A, B, C, D, E)
* (Optional) Model Name:String:128 🡪 Model name for this device. I.E. Fascinate
* (Optional) Model Brand:String:128 🡪 Model manufacturer. I.E. Samsung
* (Optional) Model Number:String:128 🡪 Model number. I.E. 2, V2, etc.
* (Optional) Serial Number:String:64 🡪 Serial number of device.
* Batch ID:Int:8

Return Sub Codes

* 0x2E00

### Activate Device

Use this function to activate individual devices when fulfilling/packaging orders. This function will work against device inventory/subscriber states LOADED/NA and AVAILABLE/HOLDING. Upon completion of this function the device inventory state will be one of ASSIGNED, UNAVAILABLE, or ERROR. And the subscriber state will be one of ACTIVE or ERROR.

Note: This is an asynchronous API call. Make periodic calls to GetDevice() to obtain the subscriber status of your activation. The response delay can be typically attributed to the response time of the carrier systems and can range from sub-second to 30+ minutes depending on backlog.

Inputs

* Device ID:String:128 🡪 ESN/SIM identifier

Outputs

* <standard response message block>
* Device ID:String:128 🡪 The ESN/SIM for this device.
* Subscriber ID:Int 🡪 The subscriber ID created when the device is activated. A subscriber entry will be created by this function if inventory loaded using BulkLoadInventory.
* Inventory State:ENUM 🡪 ASSIGNED, AVAILABLE, LOADED, UNAVAILABLE, ERROR
* Subscriber State:ENUM 🡪 PLACED, CANCELED, HOLDING, ACTIVE, INACTIVE, ERROR, SUSPENDED.

Return Sub Codes

* 0x2F00

### Deactivate Device **(future)**

Inputs

Outputs

* <standard response message block>

Return Sub Codes

* 0x3000

### Load Device

This function loads device inventory into a client’s “account” one device at a time. Once loaded into the system the inventory can be used for subsequent wholesale or consumer based API calls like ActivateDevice() and SetOrderDevice();

This API call will create a Subscriber entry with an inventory state of AVAILABLE and subscriber state HOLDING.

Inputs

* (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID to force the load into the specific account. Otherwise defaults to the Client account.
* (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup account by Account Identifier.
* Device
  + Inventory Identifier has to be one of:
    - (Optional) Product ID:Int:8 🡪 Product ID of the specific inventory type to add.
    - (Optional) Custom Key:String:32 🡪 Client defined value that maps to internal product ID.
  + Device Type:ENUM 🡪 One of CDMA, GSM.
  + Device ID:String:128 🡪 ESN/SIM identifier
  + Grade:String:16 🡪 Quality classification (NEW, REFURB, A, B, C, D, E)
  + (Optional) Model Serial Number:String:64 🡪 Serial number of device.
  + (Optional) MSL:Int:16 🡪 Device MSL code if CDMA.
  + (Optional) MSID:String:128 🡪 MSID code of the device if CDMA.
  + (Optional) Model Name:String:128 🡪 Model name for this device. I.E. Fascinate
  + (Optional) Model Brand:String:128 🡪 Model manufacturer. I.E. Samsung
  + (Optional) Model Number:String:128 🡪 Model number. I.E. 2, V2, etc.
  + (Optional) Serial Number:String:64 🡪 Serial number of device.

Outputs

* <standard response message block>
* Batch ID:Int:8 🡪 Used for subsequent inventory function calls.
* Load Result
  + Device ID:String:128 🡪 ESN/SIM identifier
  + Fail Reason:String:128

Return Sub Codes

* 0x3100

### Unload Device **(future)**

Inputs

Outputs

* <standard response message block>

Return Sub Codes

* 0x3200

### Wireless Coverage Check

Utility function that returns the percentage of coverage for a zip code area.

**Note**: Client configuration determines which carriers can be accessed.

**Note**: Function will not return carriers with less than 30% coverage.

Inputs

* (Optional) Account ID:Int:8 🡪 If set, can be one of Partner ID, Dealer ID, Lifeline ID, Retail ID to force the lookup to the products defined for each carrier. Otherwise defaults to the Client account carrier setup.
* (Optional) Account Identifier:String:128 🡪 Same as Account ID, but lookup account by Account Identifier.
* Carrier Code:String:32 🡪 The carriers to check if coverage available in the area.
  + SPRINT, VERIZON, TMOBILE.
  + ALL will check all supported carriers and return an array of results.
  + A composite list can also be provided. I.E. ‘SPRINT+VERIZON’
* Zip Code:Int:5 🡪 Area to search coverage database.
* (Optional) Zip4:Int:4 🡪 If set, Zip must be set also.

Outputs

* <standard response message block>
* Coverage Array
  + Coverage Carrier:String:32 🡪 Carrier for this coverage percentage
  + Coverage Percentage:Decimal:2.2🡪 Percentage of area covered by carrier.
  + Coverage Zip:Int:5 🡪 The zip code provided for the coverage check.
  + (Optional) Zip4:Int:4 🡪 The zip4 code if available.
  + Color Code:String:32 🡪 The “color” assigned to this carrier to group carrier products.
  + Color Hex Code:String:16 🡪 The RGB hex code of the color assigned.

Example Flow

* covPercentage = WirelessCoverageCheck(‘SPRINT’, 52233);

Return Sub Codes

* 0x3400

### Get Pending Auto Replenish

This function returns a list of pre-paid devices ready for product auto-replenishment based upon the criteria setup using **PlaceOrder**. Each call to **GetPendingAutoReplenish** will mark the queued auto replenish events as “CONSUMED”. At that point it is the client’s responsibility to either enable the auto-replenishment or cancel it.

**Note**: For clients implementation this hybrid auto-replenishment feature, the **GetPendingAutoReplenish** call should be made on a regular basis (every 30 minutes for example) to maintain processing for time sensitive replenishments.

**Note**: Client configuration will determine how much padding is applied for time based auto replenishments. For example, slower payment processes will likely want 2-3 days’ notice where more automated processes may only need a few hours.

Inputs

* (Optional) Limit:Int:5 🡪 If set will return up to **Limit** events to help client manage load.

Outputs

* <standard response message block>
* Event Array
  + Event ID:Int:8 🡪 The ID of the replenish event. Use for enabling/cancelling.
  + Auto Replenish Array: One of the following replenish options (or both) must be set:
    - (Optional) ReplenishByDayOfMonth:String:16 🡪 {First, Last, or specific day}
    - (Optional) ReplenishByDepletionPercentage:Decimal:2.2 🡪 {percentage remaining
  + Device ID:String:128 🡪 ESN/SIM identifier.
  + Line ID:Int:8 🡪 The specific line for the auto replenish.
  + Product ID:Int:8 🡪 ID of the product to be replenished.
  + Description:String:128 🡪 The product description
  + Rate:Decimal:5.4 🡪 The default product price
  + Taxes:Decimal:5.4 🡪 The tax and fees for the product based on the order billing location

Return Sub Codes

* 0x3500

Example Flow

Setup

* NewOrderID = BuildNewOrder();
* DeviceLineItemID = AddOrderDevice(NewOrderID, ProductID, ProvisionFlag);
* ProductLineItemID = AddOrderAirtimeProduct(NewOrderID, ProductID, DeviceLineItemID,

### Enable Pending Auto Replenish

Take the auto replenish event ID and loads the product tied to the event.

**Note**: If the product changes during the course of collecting payment, the event must be cancelled, then the auto-replenish configuration changed to the new product.

Inputs

* Event ID:Int:8 🡪 The ID of the event to enable.

Outputs

* Return <standard response message block>

Codes

* 0x3600

### Cancel Pending Auto Replenish

Takes the auto replenish event ID and marks it cancelled. No product is loaded.

Inputs

* Event ID:Int:8 🡪 The ID of the event to enable.

Outputs

* <standard response message block>

Return Sub Codes

* 0x3700

## Payment API Group

### Add Payment Profile

Function call to define a payment profile for a customer.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* Payment Profile Info, requires one of Bank ACH/eCheck or Credit Card.
  + Type:ENUM:(Individual or Business) 🡪 Type of customer
  + (Optional) Bank Account Information
    - AccountType:ENUM:(Checking, Savings)
    - AccountName:String:32
    - ECheckType:ENUM:(CCD, PPD, TEL, WEB)
    - BankName:String:64
    - RoutingNumber:String:9
    - AccountNumber:String:17
  + (Optional) Credit Card Information
    - CardNumber:String:16
    - ExpirationDate:String:7 🡪 Format: YYYY-MM
    - CardCode:String:4 🡪 CCV value
    - CardType:ENUM 🡪 (‘MC’ = MasterCard, ‘DS’ = Discover, ‘VC’ = Visa, ‘AM’ = American Express)
    - First Name:String:64 🡪 First name of card holder.
    - Last Name:String:64 🡪 Last name of card holder.
  + (Optional) Driver License
    - State:String:2
    - DriverLicenseNumber:String:20
    - DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
  + (Optional) Tax ID:String:9 🡪 Customers Tax ID or SSN
* Validation Mode: ENUM: 🡪 Used to validate card or verify funds. One of (‘CARD’,’FUNDS’).
  + ‘CARD’: Payment provider will validate card is not stolen, is real card, checks expiration date, etc.
  + ‘FUNDS’: Payment provide will perform actual $ amount charge against the card to test funds.
* Profile Type: ENUM: 🡪 One of (‘PERMANENT’,’ONE\_TIME’)
  + ‘PERMANENT’ means the profile is visible in the consumer’s account and editable.
  + ‘ONE\_TIME’ means the profile can only be tied to an order once. Subsequent use will fail.
* (Optional) Billing Location Info, Overrides the account billing location
  + (Optional) Billing Address
    - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25
* Default Profile: Boolean: 🡪 Make this profile the default payment profile. Must be set either true/false;

Outputs

* <standard response message block>
* PaymentProfileID:Int:8
* ResponseCode:String:8 🡪Response Code (Provider specific)
* ResponseReasonDesc:String:255 🡪Description of the validation result (Provider specific)

Return Sub Codes

* 0x5400 🡪Provider Gateway Failure
* 0x5401 🡪Internal System Failure

### Get Payment Profile

Get payment and/or shipping profile information for a customer.

Inputs

One of the following search filters must be provided.

* (Optional) Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* (Optional) PaymentProfileID:Int:8 🡪 The Payment Profile ID of the customer.

Outputs

* <standard response message block>
* Payment Profiles Array
  + Payment Profile ID:Int:8 🡪 The payment profile ID.
  + Payment Profile Type:ENUM 🡪 One of ‘PERMANENT’,’ONE TIME’.
  + Payment Profile Info, one of Bank ACH/eCheck, Credit Card, Manual.
    - Type:ENUM:(Individual or Business) 🡪 Type of customer
    - (Optional) Bank Account Information
      * AccountType:ENUM:(Checking, Savings)
      * AccountName:String:32
      * ECheckType:ENUM:(CCD, PPD, TEL, WEB)
      * BankName:String:64
      * RoutingNumber:String:9
      * AccountNumber:String:17
    - (Optional) Credit Card Information
      * CardNumber (last 4 digits):String:4
      * ExpirationDate:String:7 🡪 Format: YYYY-MM
      * CardType:ENUM 🡪 (‘MC’ = MasterCard, ‘DS’ = Discover, ‘VC’ = Visa, ‘AM’ = American Express)
      * First Name:String:64 🡪 First name of card holder.
      * Last Name:String:64 🡪 Last name of card holder.
    - (Optional) Driver License
      * State:String:2
      * DriverLicenseNumber:String:20
      * DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
    - (Optional) Tax ID:String:9 🡪 Customers Tax ID or SSN
  + Billing Location Info
    - Billing Address
      * FirstName:String:64
      * MiddleInitial:String:1
      * LastName:String:64
      * Company:String:50
      * Address1:String:128
      * Address2:String:128
      * City:String:64
      * State:String:64
      * Zip:String:5
      * Zip4:String:4
      * Country:String:128
      * PhoneNumber:String:25
      * FaxNumber:String:25
  + Default Profile: Boolean: 🡪 True if this profile is the default.

Return Sub Codes

* 0x5500 🡪Provider Gateway Failure
* 0x5501 🡪Internal System Failure

### Close Payment Profile

Closes a payment profile for the given customer. Once a profile has been closed it cannot be reopened.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* PaymentProfileID:Int:8 🡪 The Payment Profile ID of the customer.

Outputs

* <standard response message block>

Return Sub Codes

* 0x5700 🡪Provider Gateway Failure
* 0x5701 🡪Internal System Failure

### Process Payment

Process a payment against a customer and specific payment profile.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* PaymentProfileID:Int:8 🡪 The Payment profile ID of the customer.
* TransactionAmount:Decimal:8.4 🡪 Total amount of transaction. If PaymentTransactionID set this amount must match the amount set using AuthorizePayment.
* (Optional) Validation Mode: ENUM: 🡪 Used to validate card or verify funds. One of (‘CARD’,’FUNDS’). Defaults to the validation mode setup on the payment profile if not specified.
  + ‘CARD’: Payment provider will validate card is not stolen, is real card, checks expiration date, etc.
  + ‘FUNDS’: Payment provide will perform actual $ amount charge against the card to test funds.
* (Optional) PaymentTransactionID:String:32 🡪 Transaction ID from AuthorizePayment call
  + **Note**: If populated the method will submit the previous AuthorizePayment transaction for payment.
* (Optional) Order ID:Int:8 🡪 The Order ID of an order current in Payment Hold status.

Outputs

* <standard response message block>
* PaymentTransactionID:String:32 🡪 Transaction ID for this payment
* ResponseCode:String:8 🡪Response Code (Provider specific)
* ResponseReasonDesc:String:255 🡪Description of the validation result (Provider specific)

Return Sub Codes

* 0x5820 🡪Provider Gateway Failure
* 0x5821 🡪Internal System Failure

### Authorize Payment

Verifies a payment request is valid.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* PaymentProfileID:Int:8 🡪 The Payment profile ID of the customer.
* TransactionAmount:Decimal:8.4 🡪 Total amount of transaction.
* (Optional) Validation Mode: ENUM: 🡪 Used to validate card or verify funds. One of (‘CARD’,’FUNDS’). Defaults to the validation mode setup on the payment profile if not specified.
  + ‘CARD’: Payment provider will validate card is not stolen, is real card, checks expiration date, etc.
  + ‘FUNDS’: Payment provide will perform actual $ amount charge against the card to test funds.
* (Optional) Order ID:Int:8 🡪 The Order ID of an order current in Payment Hold status.

Outputs

* <standard response message block>
* PaymentTransactionID:String:32 🡪 Transaction ID for use with ProcessPayment
* ResponseCode:String:8 🡪Response Code (Provider specific)
* ResponseReasonDesc:String:255 🡪Description of the validation result (Provider specific)

Return Sub Codes

* 0x5900 🡪Provider Gateway Failure
* 0x5901 🡪Internal System Failure

### Delete Payment

Voids a prior payment authorization.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* PaymentProfileID:Int:8 🡪 The Payment profile ID of the customer.
* PaymentTransactionID:String:32 🡪 Transaction ID from a prior payment AuthorizePayment call
* (Optional) Reason:String:255 🡪 User provided Reason

Outputs

* <standard response message block>
* ResponseCode:String:8 🡪Response Code (Provider specific)
* ResponseReasonDesc:String:255 🡪Description of the validation result (Provider specific)

Return Sub Codes

* 0x5A00 🡪Provider Gateway Failure
* 0x5A01 🡪Internal System Failure

### Refund Payment

Applies a refund to the customer’s profile.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* PaymentProfileID:Int:8 🡪 The Payment profile ID of the customer.
* PaymentTransactionID:String:32
* TransactionAmount:Decimal:8.4 🡪 Total amount of refund
  + **Note**: Must be less than or equal to original transaction

Outputs

* <standard response message block>
* PaymentTransactionID:String:32 🡪 Transaction ID for this refund
* ResponseCode:String:8 🡪Response Code (Provider specific)
* ResponseReasonDesc:String:255 🡪Description of the validation result (Provider specific)

Return Sub Codes

* 0x5B00 🡪Provider Gateway Failure
* 0x5B01 🡪Internal System Failure

### Set Payment Profile Default

Set the requested payment profile as the default profile for the account.

Inputs

* Account ID:Int:8 🡪 One of Customer ID, Partner ID, Dealer ID, Agent ID, Wholesale ID, etc.
* PaymentProfileID:Int:8 🡪 The Payment profile ID of the customer.

Outputs

* <standard response message block>

Return Sub Codes

* 0x5B50

## Lifeline API Group

### Build Lifeline Order (Future)

This begins the process of creating a lifeline order. Basically this function generates empty data structures in preparation for subsequent **AddLifelineOrderItem** function calls; which will require the LifelineOrderID.

Inputs

* Agent ID:Int:8🡪 The ID of the dealer this order will be placed for.

Outputs

* <standard response message block>
* LifelineOrderID:Int:8

Return Sub Codes

* 0x3800

### Add Lifeline Order Item (Future)

This function is used to add inventory to a lifeline order.

**Note**: A quantity check will be made by this function to ensure adequate inventory is available.

Inputs

* LifelineOrderID:Int:8 🡪 The ID of the wholesale order being created.
* Inventory Identifier has to be one of:
  + (Optional) Product ID:Int:8 🡪 Product ID of the specific inventory type to add to the order.
  + (Optional) Custom Key:String:32 🡪 Client defined value input at inventory load. Can be used as a model mapping field. I.E. Model SKU
* Add Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Autopick:**0x01** 🡪 Let system pick the ESN/SIMs randomly from available inventory.
  + (Optional) Provided:**0x02** 🡪 Client provides the ESN/SIM values to be selected from inventory.
    - **Note**: ESN/SIM values must be present and available within inventory and match the ProductID/Custom Key provided.
    - Device ID Array
      * Device ID:String:128 🡪 ESN/SIM value
* Quantity:Int:5 🡪 The quantity to auto pick or provided by client. Device ID array quantity must match this value.

Outputs

* <standard response message block>
* LineItemID:Int:4
* Product ID:Int:8
* Custom Key:String:32

Return Sub Codes

* 0x3900

### Remove Lifeline Order Item (Future)

Use to reverse line items already part of the order build.

Inputs

* LifelineOrderID:Int:8 🡪 The ID of the lifeline order being created.
* LineItemID:Int:4 🡪 The line item to remove.

Outputs

* <standard response message block>

Return Sub Codes

* 0x3A00

### Place Lifeline Order (Future)

Records the lifeline order and queues the job for provisioning/fulfillment.

**Note**: Provisioning workflow type is configured at the system level for either of automated activation or processing via Ready Wireless fulfillment/provisioning system.

Inputs

* LifelineOrderID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x3B00

### Cancel Lifeline Order (Future)

For canceling a lifeline order. Can succeed on orders in “BUILD”, “PLACED”, “HOLD” states. Once processing begins the order cannot be stopped.

Inputs

* LifelineOrderID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x3C00

### Get Lifeline Order (Future)

Returns the lifeline order details and associated ESNs/SIMs.

**Note**: If order state is “PROCESSING” the device ID arrays will not be returned…only the order summary information. The processed quantity will change with each additional call. Once the state is “COMPLETE” the results are final and can be relied upon to remain constant. It is possible for the quantity processed to be less than the quantity ordered if there is errors occur during the activation step.

Inputs

* LifelineOrderID:Int:8

Outputs

* <standard response message block>
* LifelineOrder ID:Int:8 🡪 Order ID specific to this order
* Type:String:16 🡪 Order type (NEW, CANCEL, CHANGE)
* State:String:16 🡪 Order state (BUILD, PLACED, HOLD, PROCESSING, SHIPPED, COMPLETE, CLOSED, CANCELED, AUTO CANCELED)
* BuildDate:Date:32 🡪 Date order build started. Format: YYYY-MM-DD HH24:MM:SS
* PlacedDate:Date:32 🡪 Date order was placed. Format: YYYY-MM-DD HH24:MM:SS
* CurrentStateDate:Date:32 🡪 Date the current state started. Format: YYYY-MM-DD HH24:MM:SS
* Order Line Item Array
  + Inventory Identifier, one of:
    - Product ID:Int:8
    - Custom Key:String:32
  + Add Flag:Int:2 🡪 Must be one of the following: Autopick:**0x01**, Provided:**0x02**
  + (Optional) Device ID Array
    - Device ID:String:128 🡪 ESN/SIM value
  + Quantity Ordered:Int:5
  + Quantity Processed:Int:5 🡪 Number applied to order. May not match quantity ordered.

Return Sub Codes

* 0x3D00

### Add Agent (Future)

The mechanism for defining an agent within the system. Once defined, lifeline orders may be placed for the agent.

Inputs

* Name:String:128 🡪 Agent specific name.
* Agent Carrier Codes Array 🡪 Carriers support by this dealer.
  + Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
* Address Line 1:String:128
* Address Line 2:String:128
* City:String:64
* State:String:2
* Zip:Int:5
* (Optional) Zip4:Int:4
* Contact Number:String:16
* Contact Email Address:String:128

Outputs

* <standard response message block>
* Agent ID:Int:8 🡪 Used for all subsequent dealer related function calls.

Return Sub Codes

* 0x3F00

### Change Agent (Future)

Function to change agent specific information.

Inputs

* Agent ID:Int:8 🡪 The dealer ID to change.
* (Optional) Name:String:128 🡪 Agent specific name.
* (Optional) Dealer Carrier Codes Array 🡪 Carriers support by this agent.
  + Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
* (Optional) Address Line 1:String:128
* (Optional) Address Line 2:String:128
* (Optional) City:String:64
* (Optional) State:String:2
* (Optional) Zip:Int:5
* (Optional) Zip4:Int:4
* (Optional) Contact Number:String:16
* (Optional) Contact Email Address:String:128

Outputs

* <standard response message block>

Return Sub Codes

* 0x4000

### Close Agent (Future)

Use this function to close a agent account.

Inputs

* Agent ID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x4100

### Get Agents Lookup (Future)

Use to get a listing of all agents configured (active only).

**Note**: Can be used to retrieve specific agent account by ID or Name.

Inputs

* (Optional) Agent ID:Int:8 🡪 Can be used to filter a specific agent rather than return the entire list.
* (Optional) Name:String:128 🡪 Use to filter by agent name.

Outputs

* <standard response message block>
* Agent Array
  + Agent ID:Int:8 🡪 The agent ID to change.
  + Name:String:128 🡪 Agent specific name.
  + Agent Carrier Codes Array 🡪 Carriers support by this agent.
    - Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
  + Address Line 1:String:128
  + Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + Zip4:Int:4
  + Contact Number:String:16
  + Contact Email Address:String:128

Return Sub Codes

* 0x4200

## Retail API Group

### Build Retail Order (Future)

This begins the process of creating a retail order. This function generates empty data structures in preparation for subsequent **AddRetailOrderItem** function calls; which will require the RetailOrderID.

Inputs

* RetailID:Int:8🡪 The ID of the retail location this order will be placed for.

Outputs

* <standard response message block>
* RetailOrderID:Int:8

Return Sub Codes

* 0x4300

### Add Retail Order Item (Future)

This function is used to add inventory to a retail order.

**Note**: A quantity check will be made by this function to ensure adequate inventory is available.

Inputs

* RetailOrderID:Int:8 🡪 The ID of the retail order being created.
* Inventory Identifier has to be one of:
  + (Optional) Product ID:Int:8 🡪 Product ID of the specific inventory type to add to the order.
  + (Optional) Custom Key:String:32 🡪 Client defined value input at inventory load. Can be used as a model mapping field. I.E. Model SKU
* Add Flag:Int:2 🡪 Must be one of the following:
  + (Optional) Autopick:**0x01** 🡪 Let system pick the ESN/SIMs randomly from available inventory.
  + (Optional) Provided:**0x02** 🡪 Client provides the ESN/SIM values to be selected from inventory.
    - **Note**: ESN/SIM values must be present and available within inventory and match the ProductID/Custom Key provided.
    - Device ID Array
      * Device ID:String:128 🡪 ESN/SIM value
* Quantity:Int:5 🡪 The quantity to auto pick or provided by client. Device ID array quantity must match this value.

Outputs

* <standard response message block>
* LineItemID:Int:4
* Product ID:Int:8
* Custom Key:String:32

Return Sub Codes

* 0x4400

### Remove Retail Order Item (Future)

Use to reverse line items already part of the order build.

Inputs

* RetailOrderID:Int:8 🡪 The ID of the retail order being created.
* LineItemID:Int:4 🡪 The line item to remove.

Outputs

* <standard response message block>

Return Sub Codes

* 0x4500

### Place Retail Order (Future)

Records the retail order and queues the job for provisioning/fulfillment.

**Note**: Provisioning workflow type is configured at the system level for either of automated activation or processing via Ready Wireless fulfillment/provisioning system.

Inputs

* RetailOrderID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x4600

### Cancel Retail Order (Future)

For canceling a retail order. Can succeed on orders in “BUILD”, “PLACED”, “HOLD” states. Once processing begins the order cannot be stopped.

Inputs

* RetailOrderID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x4700

### Get Retail Order (Future)

Returns the retail order details and associated ESNs/SIMs.

**Note**: If order state is “PROCESSING” the device ID arrays will not be returned…only the order summary information. The processed quantity will change with each additional call. Once the state is “COMPLETE” the results are final and can be relied upon to remain constant. It is possible for the quantity processed to be less than the quantity ordered if there is errors occur during the activation step.

Inputs

* RetailOrderID:Int:8

Outputs

* <standard response message block>
* RetailOrderID:Int:8 🡪 Order ID specific to this order
* Type:String:16 🡪 Order type (NEW, CANCEL, CHANGE)
* State:String:16 🡪 Order state (BUILD, PLACED, HOLD, PROCESSING, SHIPPED, COMPLETE, CLOSED, CANCELED, AUTO CANCELED)
* BuildDate:Date:32 🡪 Date order build started. Format: YYYY-MM-DD HH24:MM:SS
* PlacedDate:Date:32 🡪 Date order was placed. Format: YYYY-MM-DD HH24:MM:SS
* CurrentStateDate:Date:32 🡪 Date the current state started. Format: YYYY-MM-DD HH24:MM:SS
* Order Line Item Array
  + Inventory Identifier, one of:
    - Product ID:Int:8
    - Custom Key:String:32
  + Add Flag:Int:2 🡪 Must be one of the following: Autopick:**0x01**, Provided:**0x02**
  + (Optional) Device ID Array
    - Device ID:String:128 🡪 ESN/SIM value
  + Quantity Ordered:Int:5
  + Quantity Processed:Int:5 🡪 Number applied to order. May not match quantity ordered.

Return Sub Codes

* 0x4800

### Add Retail Location (Future)

The mechanism for defining a retail location within the system. Once defined, retail orders may be placed for this location.

Inputs

* Name:String:128 🡪 Retail location specific name.
* Retail Carrier Codes Array 🡪 Carriers support by this location.
  + Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
* Address Line 1:String:128
* Address Line 2:String:128
* City:String:64
* State:String:2
* Zip:Int:5
* (Optional) Zip4:Int:4
* Contact Number:String:16
* Contact Email Address:String:128

Outputs

* <standard response message block>
* RetailID:Int:8 🡪 Used for all subsequent retail location related function calls.

Return Sub Codes

* 0x4900

### Change Retail Location (Future)

Function to change retail location specific information.

Inputs

* RetailID:Int:8 🡪 The retail location ID to change.
* (Optional) Name:String:128 🡪 Retail location specific name.
* (Optional) Retail Carrier Codes Array 🡪 Carriers support by this retail location.
  + Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
* (Optional) Address Line 1:String:128
* (Optional) Address Line 2:String:128
* (Optional) City:String:64
* (Optional) State:String:2
* (Optional) Zip:Int:5
* (Optional) Zip4:Int:4
* (Optional) Contact Number:String:16
* (Optional) Contact Email Address:String:128

Outputs

* <standard response message block>

Return Sub Codes

* 0x4A00

### Close Retail Location(Future)

Use this function to close a retail location account.

Inputs

* RetailID:Int:8

Outputs

* <standard response message block>

Return Sub Codes

* 0x4B00

### Get Retail Locations Lookup (Future)

Use to get a listing of all retail locations configured (active only).

**Note**: Can be used to retrieve specific retail location account by ID or Name.

Inputs

* (Optional) RetailID:Int:8 🡪 Can be used to filter a specific retail location rather than return the entire list.
* (Optional) Name:String:128 🡪 Use to filter by retail location name.

Outputs

* <standard response message block>
* Retail Location Array
  + RetailID:Int:8 🡪 The retail location ID to change.
  + Name:String:128 🡪 Retail location specific name.
  + Retail Carrier Codes Array 🡪 Carriers support by this retail location.
    - Carrier Code:String:32 🡪 SPRINT, VERIZON, etc.
  + Address Line 1:String:128
  + Address Line 2:String:128
  + City:String:64
  + State:String:2
  + Zip:Int:5
  + Zip4:Int:4
  + Contact Number:String:16
  + Contact Email Address:String:128

Return Sub Codes

* 0x4C00

## Internal API Group

### Internal Add User (Future)

Function call to define and setup a user account.

Created\_date:Datetime and modified\_date:Datetime are set to the current time for both User and Logistics tables.

Inputs

* Shortname:String:32 → will store the screenName field from liferay
* Description:String:128
* Guid:String:128→ used to store user’s UUID from liferay
* Client\_id:BigInt:20
* User\_id:BigInt:20→ used to store user’s userID from liferay
* Contact\_id:BigInt:20
* Type: ENUM('admin','user','agent','api','portal')
* Logistics
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20
  + State\_id:BigInt:20

Outputs

* <standard response message block>
* Id:BigInt:20

Return Sub Codes

* 0x4C00

### Internal Change User (Future)

Use to change user account information.

Modified\_date:Datetime will be updated in the User table if user information is changed.

Modified\_date:Datetime will be updated in the Logistics table if logistics information is changed.

DOES NOT ALLOW CHANGES TO DELETED USER.

Inputs

* ClientID: Int:8
* Id:BigInt:20
* (Optional) User\_id:BigInt:20
* (Optional) Guid:String:128
* (Optional) Shortname:String:32
* (Optional) Description:String:128
* (Optional) Contact\_id:BigInt:20
* (Optional) Type: ENUM('admin','user','agent','api','portal')
* (Optional) Logistics
  + (Optional) First\_name:String:64
  + (Optional) Middle\_name:String:64
  + (Optional) Last\_name:String:64
  + (Optional) Organization\_name:String:128
  + (Optional) Email\_address1:String:128
  + (Optional) Email\_address2:String:128
  + (Optional) Home\_phone:String:32
  + (Optional) Work\_phone:String:32
  + (Optional) Mobile\_phone:String:32
  + (Optional) Fax:String:32
  + (Optional) Address\_line1:String:128
  + (Optional) Address\_line2:String:128
  + (Optional) City:String:64
  + (Optional) State:String:2
  + (Optional) Zip5:String:5
  + (Optional) Zip4:String:4
  + (Optional) Country:String:64
  + (Optional) Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + (Optional) Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + (Optional) Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + (Optional) Client\_id:BigInt:20

Outputs

* <standard response message block>

Return Sub Codes

* 0x5000

### Internal Delete User (Future)

For disabling a user’s account. This does not delete the account, but the account cannot be re-enabled once closed. The user’s state is updated and entry is created in the state\_history table.

Inputs

* ClientID: Int:8
* Id:BigInt:20

Outputs

* <standard response message block>

Return Sub Codes

* 0x4D00

### Internal Get user (Future)

Returns user details. Will return deleted user.

**Note**: May use one optional search criteria.

Inputs

* ClientID: Int:8
* Id:BigInt:20
* (Optional) User\_id:BigInt:20
* (Optional) Logistics
  + First Name:String:64
  + Last Name:String:64
* (Optional) Shortname:String:32

Outputs

* <standard response message block>
* Id:BigInt:20
* User\_id:BigInt:20
* Shortname:String:32
* Description:String:128
* Guid:String:128
* Client\_id:BigInt:20
* Contact\_id:BigInt:20
* Type: ENUM('admin','user','agent','api','portal')
* State:String:unknown
* State\_description:String:unknown
* Created\_date:Datetime
* Modified\_date:Datetime
* Logistics
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20
  + State:String:unknown
  + State\_description:String:unknown
  + Created\_date:Datetime
  + Modified\_date:Datetime

Return Sub Codes

* 0x4F00

### Internal Add Organization (Future)

Function call to define and setup an account (organization).

Created\_date:Datetime and Modified\_date:Datetime are set to the current time.

State\_id:BigInt:20 along with an entry into the state history table are created at this time.

Inputs

* Legal\_name:String:128
* Type:ENUM('Client','Agent','Dealer','Retail','Consumer')
* Client\_id:BigInt:20
* Organization\_id:BigInt:20 -> used to store organization ID from liferay
* Parent\_account\_id:BigInt:20
* Default\_billing (logistics)
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20
* Default\_shipping (logistics)
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20

Outputs

* <standard response message block>
* Account\_id:BigInt:20

Return Sub Codes

* 0x4C00

### Internal Change Organization (Future)

Function call to change an account. (organization) DOES NOT ALLOW CHANGES TO DELTED ACCOUNTS

Inputs

* Account\_id:BigInt:128
* Organization\_id:BigInt:20
* (optional) Legal\_name:String:128
* (optional) Type:ENUM('Client','Agent','Dealer','Retail','Consumer')
* (optional) Client\_id:BigInt:20
* (optional) Parent\_account\_id:BigInt:20
* (optional) Default\_billing (logistics)
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20
* (optional) Default\_shipping (logistics)
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20

Outputs

* <standard response message block>
* Account\_id:Int:8

Return Sub Codes

* 0x4C00

### Internal Delete Organization (Future)

Function call to delete an account. Accounts are not actually deleted. But the state id is updated and an entry is made into the state history table

Inputs

* ClientID: Int:8
* Account\_id:BigInt:20
* Organization\_id:BigInt:20

Outputs

* <standard response message block>

Return Sub Codes

* 0x4C00

### Internal Get Organization (Future)

Function call to get an account. Will return account information even if state is deleted.

Inputs

* Client\_id:Int:8
* Account\_id:BigInt:20

Outputs

* <standard response message block>
* Account\_id:BigInt:20
* Organization\_id:BigInt:20
* State:String:unknown
* State\_description:String:unknown
* Legal\_name:String:128
* Type:ENUM('Client','Agent','Dealer','Retail','Consumer')
* Client\_id:BigInt:20
* Parent\_account\_id:BigInt:20
* Default\_billing
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20
* Default\_shipping
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20

Return Sub Codes

* 0x4C00

### Internal Get Organizations (Future)

Function call to get an account. DOES NOT RETURN DELTED ACCOUNTS.

Inputs

* Client\_id:Int:8

Outputs

* <standard response message block>
* Account\_id:BigInt:20
* Legal\_name:String:128
* Type:ENUM('Client','Agent','Dealer','Retail','Consumer')
* Client\_id:BigInt:20
* Parent\_account\_id:BigInt:20

Return Sub Codes

* 0x4C00

### Internal Add User Organization (Future)

Function call to define and setup a user account. Inputs

* ClientID: Int:8
* User\_id:BigInt:20
* Account\_id:BigInt:20

Outputs

* <standard response message block>

Return Sub Codes

* 0x4C00

### Internal Delete User Organization (Future)

Function call to delete a user from an account (organization).

Inputs

* ClientID: Int:8
* User\_id:BigInt:20
* Account\_id:BigInt:20

Outputs

* <standard response message block>

Return Sub Codes

* 0x4C00

### Internal Get Users In Organization (Future)

Function call to get all users from an account (organization). DOES NOT RETURN DELETED USERS

Inputs

* ClientID: Int:8
* Account\_id:BigInt:20

Outputs

* <standard response message block>
* Id:BigInt:20
* Shortname:String:32
* Description:String:128
* Guid:String:128
* Client\_id:BigInt:20
* Contact\_id:BigInt:20
* Type: ENUM('admin','user','agent','api','portal')
* Created\_date:Datetime
* Modified\_date:Datetime
* Logistics
  + First\_name:String:64
  + Middle\_name:String:64
  + Last\_name:String:64
  + Organization\_name:String:128
  + Email\_address1:String:128
  + Email\_address2:String:128
  + Home\_phone:String:32
  + Work\_phone:String:32
  + Mobile\_phone:String:32
  + Fax:String:32
  + Address\_line1:String:128
  + Address\_line2:String:128
  + City:String:64
  + State:String:2
  + Zip5:String:5
  + Zip4:String:4
  + Country:String:64
  + Primary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Secondary\_contact\_type: ENUM('MobilePhone','HomePhone','WorkPhone','Email1','Email2')
  + Type: ENUM('Contact','Billing Default','Shipping Default','Organization','Alias','DBA','Billing Order','Shipping Order')
  + Client\_id:BigInt:20
  + State:String:unknown
  + State\_description:String:unknown
  + Created\_date:Datetime
  + Modified\_date:Datetime

Return Sub Codes

* 0x4C00

### Internal Add Payment Customer

Function call to define and setup a customer profile.

Inputs

* Customer Identification Info. At least one of the following is required.
  + (Optional)Customer ID:String:20 🡪 Caller Assigned Customer ID
  + (Optional)Description:String:255 🡪 Description of the customer
  + (Optional)Email:String:255 🡪 Email address associated with the customer
* Payment Profile – Optional
  + **NOTE**: If payment profile information is passed then either Bank Account or Credit Card need to be defined.
  + Type:ENUM:(Individual or Business) 🡪 Type of customer
  + (Optional) Billing Address
    - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25
  + (Optional) Bank Account Information
    - AccountType:ENUM:(Checking, Savings)
    - AccountName:String:32
    - ECheckType:ENUM:(CCD, PPD, TEL, WEB)
    - BankName:String:64
    - RoutingNumber:String:9
    - AccountNumber:String:17
  + (Optional) Credit Card Information
    - CardNumber:String:16
    - ExpirationDate:String:7 🡪 Format: YYYY-MM
    - CardCode:String:4 🡪 CCV value
  + (Optional) Driver License
    - State:String:2
    - DriverLicenseNumber:String:20
    - DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
  + (Optional) Tax ID:String:9 🡪 Customers Tax ID or SSN
* (Optional) Shipping Profile
  + (Optional)FirstName:String:64
  + (Optional)MiddleInitial:String:1
  + (Optional)LastName:String:64
  + (Optional)Company:String:50
  + (Optional)Address1:String:128
  + (Optional)Address2:String:128
  + (Optional)City:String:64
  + (Optional)State:String:64
  + (Optional)Zip:String:5
  + (Optional)Zip4:String:4
  + (Optional)Country:String:128
  + (Optional)PhoneNumber:String:25
  + (Optional)FaxNumber:String:25
* Validate:Enum 🡪 How validation of this customer should be handled
  + liveMode 🡪 Generates a transaction to the processor
  + testMode 🡪 Field validation only
  + none 🡪 No validation is performed
    - **Note**: Use this value if no PaymentProfile is passed

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* CustomerProfileID:String:32 🡪 Profile ID of the created customer account. This ID is required by majority of the other payment methods.
* (Optional) Customer Payment Profile ID array
  + CustomerPaymentProfileID:String:32 🡪 If Payment Profile was passed the assigned Payment Profile ID
* (Optional) Customer Shipping Profile ID array
  + CustomerShippingProfileID:String:32 🡪 If Shipping Profile was passed the assigned Shipping Profile ID
* (Optional) Payment validation array 🡪 If passed Validate was liveMode/testMode and payment profile was passed this structure provides the result of the validation
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5000 🡪 Success
* 0x5001 🡪 Payment Gateway failure
* 0x5003 🡪 Internal failure

### Internal Get Payment Customer

Returns customer payment details.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment Profile ID of the customer.

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* Customer Identification Info
  + (Optional)Customer ID:String:20 🡪 Caller Assigned Customer ID
  + (Optional)Description:String:255 🡪 Description of the customer
  + (Optional)Email:String:255 🡪 Email address associated with the customer
* (Optional)Customer Payment Profile ID array
  + CustomerPaymentProfileID:String:32
  + Type:ENUM:(Individual or Business) 🡪 Type of customer
  + (Optional) Billing Address
    - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25
  + (Optional) Bank Account Information
    - AccountType:ENUM:(Checking, Savings)
    - AccountName:String:32
    - ECheckType:ENUM:(CCD, PPD, TEL, WEB)
    - BankName:String:64
    - RoutingNumber:String:9
    - AccountNumber:String:17
  + (Optional) Credit Card Information
    - CardNumber:String:16
    - ExpirationDate:String:7 🡪 Format: YYYY-MM
    - CardCode:String:4 🡪 CCV value
  + (Optional) Driver License
    - State:String:2
    - DriverLicenseNumber:String:20
    - DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
  + (Optional) Tax ID:String:9 🡪 Customers Tax ID or SSN
* (Optional)Customer Shipping Profile ID array
  + CustomerShippingProfileID:String:32
  + Address Record
    - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25

Return Sub Codes

* 0x5100 🡪Success
* 0x5101 🡪Provider Gateway Failure
* 0x5103 🡪Internal System Failure

### Internal Change Payment Customer

Use to change customer profile information.

**Note**: If some fields in this request are not submitted or are submitted with a blank value, the values in the original profile are removed. As a best practice to prevent this from happening, call getCustomer before calling changeCustomer. That will return all current information. Then, simply change the field that needs updating and use that to call changeCustomer.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* Customer Identification Info
  + (Optional)Customer ID:String:20 🡪 Caller Assigned Customer ID
  + (Optional)Description:String:255 🡪 Description of the customer
  + (Optional)Email:String:255 🡪 Email address associated with the customer

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description

Return Sub Codes

* 0x5200 🡪Success
* 0x5201 🡪Provider Gateway Failure
* 0x5203 🡪Internal System Failure

### Internal Close Payment Customer

Close a customer’s account. This does not delete the account, but the account cannot be re-enabled once closed.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description

Return Sub Codes

* 0x5300 🡪Success
* 0x5301 🡪Provider Gateway Failure
* 0x5303 🡪Internal System Failure

### Internal Add Payment Profile

Function call to define a payment and/or shipping profile for a customer.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* (Optional) Payment Profile
  + **NOTE**: If payment profile information is passed then either Bank Account or Credit Card need to be defined.
  + Type:ENUM:(Individual or Business) 🡪 Type of customer
  + (Optional) Billing Address
    - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25
  + (Optional) Bank Account Information
    - AccountType:ENUM:(Checking, Savings)
    - AccountName:String:32
    - ECheckType:ENUM:(CCD, PPD, TEL, WEB)
    - BankName:String:64
    - RoutingNumber:String:9
    - AccountNumber:String:17
  + (Optional) Credit Card Information
    - CardNumber:String:16
    - ExpirationDate:String:7 🡪 Format: YYYY-MM
    - CardCode:String:4 🡪 CCV value
  + (Optional) Driver License
    - State:String:2
    - DriverLicenseNumber:String:20
    - DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
  + Tax ID:String:9 🡪 Customers Tax ID or SSN
* (Optional) Shipping Profile
  + - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25
* Validate:Enum 🡪 How validation of the passed PaymentProfile should be handled
  + liveMode 🡪 Generates a transaction to the processor
  + testMode 🡪 Field validation only
  + none 🡪 No validation is performed
    - **Note**: Use this value if no PaymentProfile is passed

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* (Optional)CustomerPaymentProfileID:String:32
* (Optional)CustomerShippingProfileID:String:32
* (Optional) Payment validation array 🡪 If passed Validate was liveMode/testMode and payment profile was passed this structure provides the result of the validation
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5400 🡪Success
* 0x5401 🡪Provider Gateway Failure
* 0x5403 🡪Internal System Failure

### Internal Get Payment Profile

Get payment and/or shipping profile information for a customer.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* (Optional) CustomerPaymentProfileID:String:32 🡪 The associated Payment Profile Record
* (Optional) CustomerShippingPorfileID:String:32 🡪 The associated Shipping Profile Record
  + **Note**: If neither CustomerPaymentProfileID or CustomerShippingProfileID are passed than all Payment and Shipping profiles will be returned.

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* Zero to Many Customer Payment Profiles
  + CustomerPaymentProfileID:String:32
  + CustomerPaymentProfile
    - Type:ENUM:(Individual or Business) 🡪 Type of customer
    - (Optional) Billing Address
      * (Optional)FirstName:String:64
      * (Optional)MiddleInitial:String:1
      * (Optional)LastName:String:64
      * (Optional)Company:String:50
      * (Optional)Address1:String:128
      * (Optional)Address2:String:128
      * (Optional)City:String:64
      * (Optional)State:String:64
      * (Optional)Zip:String:5
      * (Optional)Zip4:String:4
      * (Optional)Country:String:128
      * (Optional)PhoneNumber:String:25
      * (Optional)FaxNumber:String:25
    - (Optional) Bank Account Information
      * AccountType:ENUM:(Checking, Savings)
      * AccountName:String:32
      * ECheckType:ENUM:(CCD, PPD, TEL, WEB)
      * BankName:String:64
      * RoutingNumber:String:9
      * AccountNumber:String:17
    - (Optional) Credit Card Information
      * CardNumber:String:16
      * ExpirationDate:String:7 🡪 Format: YYYY-MM
      * CardCode:String:4 🡪 CCV value
    - (Optional) Driver License
      * State:String:2
      * DriverLicenseNumber:String:20
      * DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
    - (Optional) Tax ID:String:9 🡪 Customers Tax ID or SSN
* Zero to Many Customer Shipping Profiles
  + CustomerShippingProfileID:String:32
  + CustomerShippingProfile
    - (Optional)FirstName:String:64
    - (Optional)MiddleInitial:String:1
    - (Optional)LastName:String:64
    - (Optional)Company:String:50
    - (Optional)Address1:String:128
    - (Optional)Address2:String:128
    - (Optional)City:String:64
    - (Optional)State:String:64
    - (Optional)Zip:String:5
    - (Optional)Zip4:String:4
    - (Optional)Country:String:128
    - (Optional)PhoneNumber:String:25
    - (Optional)FaxNumber:String:25

Return Sub Codes

* 0x5500 🡪Success
* 0x5501 🡪Provider Gateway Failure
* 0x5503 🡪Internal System Failure

### Internal Change Payment Profile

Modify a payment and/or shipping profile record for a given customer.

**Note**: If some fields in this request are not submitted or are submitted with a blank value, the values in the original profile are removed. As a best practice to prevent this from happening, call getPaymentProfile before calling changePaymentProfile. That will return all current information. Then, simply change the field that needs updating and use that to call changePaymentProfile.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* (Optional) Payment Profile
  + CustomerPaymentProfileID:String:32
  + Type:ENUM:(Individual or Business) 🡪 Type of customer
  + Billing Address
    - FirstName:String:64
    - MiddleInitial:String:1
    - LastName:String:64
    - Company:String:50
    - Address1:String:128
    - Address2:String:128
    - City:String:64
    - State:String:64
    - Zip:String:5
    - Zip4:String:4
    - Country:String:128
    - PhoneNumber:String:25
    - FaxNumber:String:25
  + Bank Account Information
    - AccountType:ENUM:(Checking, Savings)
    - AccountName:String:32
    - ECheckType:ENUM:(CCD, PPD, TEL, WEB)
    - BankName:String:64
    - RoutingNumber:String:9
    - AccountNumber:String:17
  + Credit Card Information
    - CardNumber:String:16
    - ExpirationDate:String:7 🡪 Format: YYYY-MM
    - CardCode:String:4 🡪 CCV value
  + Driver License
    - State:String:2
    - DriverLicenseNumber:String:20
    - DateOfBirth:String:10 🡪 Format: YYYY-MM-DD
  + Tax ID:String:9 🡪 Customers Tax ID or SSN
* (Optional) Shipping Profile
  + CustomerShippingProfileID:String:32
  + Address Info
    - FirstName:String:64
    - MiddleInitial:String:1
    - LastName:String:64
    - (Optional)Company:String:50
    - Address1:String:128
    - Address2:String:128
    - City:String:64
    - State:String:64
    - Zip:String:5
    - Zip4:String:4
    - Country:String:128
    - PhoneNumber:String:25
    - FaxNumber:String:25
* Validate:Enum 🡪 How validation of the passed PaymentProfile should be handled
  + liveMode 🡪 Generates a transaction to the processor
  + testMode 🡪 Field validation only
  + none 🡪 No validation is performed
    - **Note**: Use this value if no PaymentProfile is passed

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* (Optional) Payment validation array 🡪 If passed Validate was liveMode/testMode and payment profile was passed this structure provides the result of the validation
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5600 🡪Success
* 0x5601 🡪Provider Gateway Failure
* 0x5603 🡪Internal System Failure

### Internal Close Payment Profile

Closes a payment and/or shipping profile for the given customer. Once a profile has been closed it cannot be reopened.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* (Optional) CustomerPaymentProfileID:String:32 🡪 The associated Payment Profile Record
* (Optional) CustomerShippingPorfileID:String:32 🡪 The associated Shipping Profile Record

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description

Return Sub Codes

* 0x5700 🡪Success
* 0x5701 🡪Provider Gateway Failure
* 0x5703 🡪Internal System Failure

### Internal Process Payment

Process a payment against a customer and specific payment profile.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* CustomerPaymentProfileID:String:32 🡪 The associated Payment Profile Record
* (Optional) CustomerShippingProfileID:String:32 🡪 The shipping profile to use
  + **Note**: If not passed, shipping information will not be used for payment processing
* Transaction Information
  + TransactionAmount:Decimal:8.4 🡪 Total amount of transaction
  + (Optional) PaymentTransactionID:String:32 🡪 Transaction ID from AuthorizePayment call
    - **Note**: If populated the method will submit a transaction for payment.
  + (Optional) CCVCode:String:4 🡪 Card Code Verification, 3 or 4 digits
  + (Optional) TaxExemptFlag:Boolean:True/False 🡪 Tax exempt transaction
  + (Optional) RecurringBillingFlag:Boolean:True/False 🡪 Is this a recurring transaction
  + (Optional) Line Items
    - Quantity:Decimal:4.2
    - UnitPrice:Decimal:4.2
    - (Optional)ItemID:String:31
    - (Optional)Name:String:31
    - (Optional)Description:String:255
    - (Optional)Taxable:Boolean:True/False
  + (Optional) Tax
    - Amount:Decimal:8.4
    - (Optional)Name:String:31
    - (Optional)Description:String:255
  + (Optional) Shipping
    - Amount:Decimal:8.4
    - (Optional)Name:String:31
    - (Optional)Description:String:255
  + (Optional) Duty
    - Amount:Decimal:8.4
    - (Optional)Name:String:31
    - (Optional)Description:String:255
  + (Optional) Order
    - (Optional)InvoiceNumber:String:20
    - (Optional)Description:String:255
    - (Optional)PONumber:String:255
  + (Optional) Prior Approval Code 🡪 Currently not implemented

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* Payment Result
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5800 🡪Success
* 0x5801 🡪Provider Gateway Failure
* 0x5803 🡪Internal System Failure

### Internal Authorize Payment

Verifies a payment request is valid.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* CustomerPaymentProfileID:String:32 🡪 The associated Payment Profile Record
* (Optional) CustomerShippingProfileID:String:32 🡪 The shipping profile to use
  + **Note**: If not passed, shipping information will not be used for payment processing
* Transaction Information
  + TransactionAmount:Decimal:8.4 🡪 Total amount of transaction
  + (Optional) PaymentTransactionID:String:32 🡪 Transaction ID from AuthorizePayment call
    - **Note**: If populated the method will submit a transaction for payment.
  + (Optional) CCVCode:String:4 🡪 Card Code Verification, 3 or 4 digits
  + (Optional) TaxExemptFlag:Boolean:True/False 🡪 Tax exempt transaction
  + (Optional) RecurringBillingFlag:Boolean:True/False 🡪 Is this a recurring transaction
  + (Optional) Line Items
    - Quantity:Decimal:4.2
    - UnitPrice:Decimal:4.2
    - (Optional)ItemID:String:31
    - (Optional)Name:String:31
    - (Optional)Description:String:255
    - (Optional)Taxable:Boolean:True/False
  + (Optional) Tax
    - Amount:Decimal:8.4
    - (Optional)Name:String:31
    - (Optional)Description:String:255
  + (Optional) Shipping
    - Amount:Decimal:8.4
    - (Optional)Name:String:31
    - (Optional)Description:String:255
  + (Optional) Duty
    - Amount:Decimal:8.4
    - (Optional)Name:String:31
    - (Optional)Description:String:255
  + (Optional) Order
    - (Optional)InvoiceNumber:String:20
    - (Optional)Description:String:255
    - (Optional)PONumber:String:255
  + (Optional) Prior Approval Code 🡪 Currently not implemented

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* Payment Result
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5900 🡪Success
* 0x5901 🡪Provider Gateway Failure
* 0x5903 🡪Internal System Failure

### Internal Delete Payment

Voids a prior payment authorization.

Inputs

* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* CustomerPaymentProfileID:String:32 🡪 The associated Payment Profile Record
* (Optional) CustomerShippingProfileID:String:32 🡪 The shipping profile to use
  + **Note**: If not passed, shipping information will not be used for processing
* PaymentTransactionID:String:32 🡪 Transaction ID from a prior AuthorizePayment call
* (Optional) Reason:String:255 🡪 User provided Reason

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* Payment Result
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5A00 🡪Success
* 0x5A01 🡪Provider Gateway Failure
* 0x5A03 🡪Internal System Failure

### Internal Refund Payment

Applies a refund to the customer’s profile.

Inputs

* PaymentTransactionID:String:32
* CustomerPaymentID:String:32 🡪 The Payment ID of the customer.
* CustomerPaymentProfileID:String:32 🡪 The associated Payment Profile Record
* (Optional) CustomerShippingProfileID:String:32 🡪 The shipping profile to use
  + **Note**: If not passed, shipping information will not be used for processing
* TransactionAmount:Decimal:8.4 🡪 Total amount of refund
  + **Note**: Must be less than or equal to original transaction

Outputs

* <standard response message block>
  + PaymentGatewayResult:String:16 🡪Provider result code
  + PaymentGatewayDescriptionResult:String:255 🡪 Provider result description
* Payment Result
  + ResponseCode:String:8 🡪Response Code (Provider specific)
  + ResponseReasonCode:String:8 🡪 Code that represents more details about the result (Provider specific)
  + ResponseReasonDesc:String:255 🡪Description of the validation result
  + AuthorizationCode:String:6 🡪 Authorization or approval code
  + TransactionID:String:32 -> Payment gateway assigned identification number
  + InvoiceNumber:String:20 🡪User assigned invoice number
  + VerificationResponse:String:8 🡪The cardholder authentication verification response code
  + ProviderRawReturn:String:512 🡪 Raw response from payment provider

Return Sub Codes

* 0x5B00 🡪Success
* 0x5B01 🡪Provider Gateway Failure
* 0x5B03 🡪Internal System Failure

### Internal Authorize User

Function to verify internal platform username and password credentials. This function returns

Inputs

* User Name:String:128
* Password:String:64
* (Optional) Detail:ENUM 🡪 One of ‘AUTH’,’MENU’,’ALL’.
  + ‘AUTH’: Authenticate only. Do not return menu or permission structures
  + ‘MENU’: Authenticate and also return the menu structure to display
  + ‘ALL’: Authenticate and return the menu and application permission structures.

Outputs

* <standard response message block>
* User ID:Int 🡪 The assigned user ID value
* User Name:String:128 🡪 Return the supplied username for debugging purposes
* User Description:String:256 🡪 Details for this specific user
* Default Landing Page ID:Int 🡪 The OSS portal page to display to the user after login. This reference is to a permission entry that defines the action and controller to redirect.
* Default Client ID:Int 🡪 The default selection for the client model dropdown. This references the account ID to set at login and use for subsequent API calls.
* Email Address:String:128 🡪 Email address for the user
* Full Name:String:128 🡪 The combined first name and last name for the user
* Portal Logout Time:Int 🡪 Time in seconds portal will automatically logout a user if no activity.
* (Optional) Menu Array 🡪 Can be up to 3 levels deep and is for the default account ID.
  + Display Tag:String:32 🡪 The menu value to display to the user.
  + Display Order:Int 🡪 The order the entries should be displayed.
  + Tooltip Description:String:1024 🡪 Detailed information for mouse over.
  + Type:ENUM 🡪 One of ‘GROUP’,’ACTION’,’SEPARATOR’.
    - ‘GROUP’: This is a parent menu item with a sub array of menu items.
    - ‘ACTION’: This is an actionable menu item.
    - ‘SEPARATOR’: This entry is for displaying separator bars between menu items. If at the top level (no parent id) then display separator vertically for the menu bar, otherwise display horizontally.
  + (Optional) Parent ID:Int 🡪 If set points to the group entry for this menu item.
  + (Optional) Permission ID:Int 🡪 Only set for GROUP and ACTION types. Points to the entry within the permissions array that corresponds with this menu item and should be checked if this menu item selected.
  + (Optional) Sub Menu Array
    - …Fields repeated the same as top level…
* (Optional) Permissions Array 🡪 Only ‘APPLICATION’ permissions are returned for the default account ID.
  + Permission ID:Int 🡪 Unique permission ID. Used to match up with specific menu actionable entries or as a reference for building a local permissions object.
  + Action:String:128 🡪 This defines the permission ID and the name of the MVC controller action to fire (allow access).
  + Controller:String:128 🡪 This defines the application this permission is associated with and the subsequent MVC controller the action resides within.
  + Description:String:256 🡪 A detailed description of what this permission is for.
* Account Array 🡪 Can be up to 5 levels deep and begins with the “Client” account.
  + Legal Name:String:128 🡪 The primary account name for display to the user.
  + Account Identifier:String:128 🡪 An alternate description for the account.
  + Account ID: Int 🡪 The unique identifier for the account.
  + Type: ENUM: 🡪 ‘CLIENT’
  + (Optional) Model Account Array
    - Legal Name:String:128 🡪 The primary account name for display to the user.
    - Account Identifier:String:128 🡪 An alternate description for the account.
    - Account ID: Int 🡪 The unique identifier for the account.
    - Type: ENUM: 🡪 ‘ECOMMERCE’, ‘RETAIL’, ‘WHOLESALE’, ‘ETC’
    - Parent Account ID: Int 🡪 The parent account ID for this account.
    - (Optional) Sub Account Level 1 Array
      * Legal Name:String:128 🡪 The primary account name for display to the user.
      * Account Identifier:String:128 🡪 An alternate description for the account.
      * Account ID: Int 🡪 The unique identifier for the account.
      * Type: ENUM: 🡪 ‘PARTNER’,’REGION’,’LOCATION’,’AGENT’,’SUBAGENT’,’DEALER’
      * Parent Account ID: Int 🡪 The parent account ID for this account.
      * (Optional) Sub Account Level 2 Array
        + Legal Name:String:128 🡪 The primary account name for display to the user.
        + Account Identifier:String:128 🡪 An alternate description for the account.
        + Account ID: Int 🡪 The unique identifier for the account.
        + Type: ENUM: 🡪 ‘LOCATION’,’SUBAGENT’,’DEALER’
        + Parent Account ID: Int 🡪 The parent account ID for this account.
        + (Optional) Sub Account Level 3 Array

Legal Name:String:128 🡪 The primary account name for display to the user.

Account Identifier:String:128 🡪 An alternate description for the account.

Account ID: Int 🡪 The unique identifier for the account.

Type: ENUM: 🡪 ‘DEALER’

Parent Account ID: Int 🡪 The parent account ID for this account.

Return Sub Codes

* 0x5B25

### Internal Get Application Permission

Verifies if the user should have access to a specific action in an application in the OSS portal.

Organization accounts are passed with this call to make sure that the user has access to the application action for that organization account.

Inputs

* User ID:Int 🡪 The assigned user ID value
* Action:String:128 🡪 This defines the permission ID and the name of the MVC controller action to fire (allow access).
* Controller:String:128 🡪 This defines the application this permission is associated with and the subsequent MVC controller the action resides within.
* Account ID:Int 🡪 The account ID of the organization currently selected by the user.

Outputs

* <standard response message block>
* Access Flag:BOOLEAN 🡪 One of ‘TRUE’ (1) or ‘FALSE’ (0).

Return Sub Codes

* 0x5B35

### Internal Get Application Permissions

Based upon the User ID returns an array of available permissions for the user. User has access to any permissions returned within the resulting Permission Array.

Inputs

* User ID:Int 🡪 The assigned user ID value
* Account ID:Int 🡪 The account ID of the organization currently selected by the user.

Outputs

* <standard response message block>
* Permissions Array 🡪 Only ‘APPLICATION’ permissions are returned for the default account ID.
  + Permission ID:Int 🡪 Unique permission ID. Used to match up with specific menu actionable entries or as a reference for building a local permissions object.
  + Action:String:128 🡪 This defines the permission ID and the name of the MVC controller action to fire (allow access).
  + Controller:String:128 🡪 This defines the application this permission is associated with and the subsequent MVC controller the action resides within.
  + Description:String:256 🡪 A detailed description of what this permission is for.

Return Sub Codes

* 0x5B45

### Internal Get Menus

Returns the menu structure for the OSS portal based on the user and organization account provided.

Sub menu array items are considered nested menus. The top level menu array is used to build the menu bar in the fashion designated by the calling application.

Inputs

* User ID:Int 🡪 The assigned user ID value
* Account ID:Int 🡪 The account ID of the organization currently selected by the user.

Outputs

* <standard response message block>
* Menu Array 🡪 Can be up to 3 levels deep and is for the default account ID.
  + Display Tag:String:32 🡪 The menu value to display to the user.
  + Display Order:Int 🡪 The order the entries should be displayed.
  + Tooltip Description:String:1024 🡪 Detailed information for mouse over.
  + Type:ENUM 🡪 One of ‘GROUP’,’ACTION’,’SEPARATOR’.
    - ‘GROUP’: This is a parent menu item with a sub array of menu items.
    - ‘ACTION’: This is an actionable menu item.
    - ‘SEPARATOR’: This entry is for displaying separator bars between menu items. If at the top level (no parent id) then display separator vertically for the menu bar, otherwise display horizontally.
  + (Optional) Parent ID:Int 🡪 If set points to the group entry for this menu item.
  + (Optional) Permission ID:Int 🡪 Only set for GROUP and ACTION types. Points to the entry within the permissions array that corresponds with this menu item and should be checked if this menu item selected.
  + (Optional) Sub Menu Array
    - …Fields repeated the same as top level…

Return Sub Codes

* 0x5B55

### Internal Get User Accounts

Returns the account array the OSS portal based on the user provided. This function will return sub accounts if the user has access to recurse accounts and configured at an account level with sub accounts present.

Inputs

* User ID:Int 🡪 The assigned user ID value

Outputs

* <standard response message block>
* Account Array 🡪 Can be up to 5 levels deep and begins with the “Client” account.
  + Legal Name:String:128 🡪 The primary account name for display to the user.
  + Account Identifier:String:128 🡪 An alternate description for the account.
  + Account ID: Int 🡪 The unique identifier for the account.
  + Type: ENUM: 🡪 ‘CLIENT’
  + (Optional) Model Account Array
    - Legal Name:String:128 🡪 The primary account name for display to the user.
    - Account Identifier:String:128 🡪 An alternate description for the account.
    - Account ID: Int 🡪 The unique identifier for the account.
    - Type: ENUM: 🡪 ‘ECOMMERCE’, ‘RETAIL’, ‘WHOLESALE’, ‘ETC’
    - Parent Account ID: Int 🡪 The parent account ID for this account.
    - (Optional) Sub Account Level 1 Array
      * Legal Name:String:128 🡪 The primary account name for display to the user.
      * Account Identifier:String:128 🡪 An alternate description for the account.
      * Account ID: Int 🡪 The unique identifier for the account.
      * Type: ENUM: 🡪 ‘PARTNER’,’REGION’,’LOCATION’,’AGENT’,’SUBAGENT’,’DEALER’
      * Parent Account ID: Int 🡪 The parent account ID for this account.
      * (Optional) Sub Account Level 2 Array
        + Legal Name:String:128 🡪 The primary account name for display to the user.
        + Account Identifier:String:128 🡪 An alternate description for the account.
        + Account ID: Int 🡪 The unique identifier for the account.
        + Type: ENUM: 🡪 ‘LOCATION’,’SUBAGENT’,’DEALER’
        + Parent Account ID: Int 🡪 The parent account ID for this account.
        + (Optional) Sub Account Level 3 Array

Legal Name:String:128 🡪 The primary account name for display to the user.

Account Identifier:String:128 🡪 An alternate description for the account.

Account ID: Int 🡪 The unique identifier for the account.

Type: ENUM: 🡪 ‘DEALER’

Parent Account ID: Int 🡪 The parent account ID for this account.

Return Sub Codes

* 0x5B65

## Content API Group

### Get Templates

Function call to get back a list of templates. If the Filter field is set, the method will also return back the desired sub items.

Inputs

* Account ID:Int:20 → The Account ID.
* These 3 fields are optional, but only one can be set:
  + (Optional) Template ID:Int:20 → The Template ID.
  + (Optional) Template Name:String:64 → The Name of the Template
  + (Optional) Type:Enum → One of HOME, SITE. Only return back templates of this type.
* (Optional) Filter:Enum → One of BLOCKS, BLOCKS\_TAGS, BLOCKS\_TAGS\_ATTRIBUTES. If this field is empty no child records will be returned.

Outputs

* <standard response message block>
* Template Array
  + ID:Int:20
  + Name:String:64
  + Description:String:256
  + Type:String:64
  + Path:String:512
  + Block Array (Optional)
    - ID:int:20
    - Name:String:64
    - Description:String:256
    - Type:String:64
    - Path:String:512
    - Tag Array (Optional)
      * ID:Int:20
      * Name:String:64
      * Description:String:256
      * Type:String:64
      * Value:String:4096
      * Attribute Array (Optional)
        + ID:Int:20
        + Name:String:64
        + Value:String:4096

Return Sub Codes

* 0x1740

## Example Usage

* Get all templates with the HOME type, and with child records of BLOCKS, TAGS, and ATTRIBUTES
  + AccountID: 25
  + Type: HOME
  + Filter: BLOCKS\_TAGS\_ATTRIBUTES
* Get all templates with ID 2 with no child records
  + AccountID: 25
  + TemplateID: 2

### Get Blocks

Function call to get back a list of blocks in the given Template ID. If the Filter field is set, the method will also return back the desired sub items.

Inputs

* Account ID:Int:20 → The Account ID.
* Template ID: Int:20 → The Template ID.
* These 3 fields are optional, but only one can be set:
  + (Optional) Block ID:Int:20 → The Block ID.
  + (Optional) Block Name:String:64 → The Name of the Block
  + (Optional) Type:ENUM →One of HEADER, TRAILER, CONTENT, FAQ, PHONE, MENU(?). Only return back blocks of this type.
* (Optional) Filter:ENUM → One of TAGS, TAGS\_ATTRIBUTES. If this field is empty no child records will be returned.

Outputs

* <standard response message block>
* Block Array (Optional)
  + ID:int:20
  + Name:String:64
  + Description:String:256
  + Type:String:64
  + Path:String:512
  + Tag Array (Optional)
    - ID:Int:20
    - Name:String:64
    - Description:String:256
    - Type:String:64
    - Value:String:4096
    - Attribute Array (Optional)
      * ID:Int:20
      * Name:String:64
      * Value:String:4096

Return Sub Codes

* 0x1760

## Example Usage

* Get all blocks from the template ID 1 with the HEADER type, and with child records of TAGS, and ATTRIBUTES
  + AccountID: 25
  + TemplateID: 1
  + Type: HEADER
  + Filter: TAGS\_ATTRIBUTES
* Get all blocks from the template ID 4 with the CONTENT type, with no child records
  + AccountID: 25
  + TemplateID: 4
  + Type: CONTENT

### Get Tags

Function call to get back a list of tags in the given Block ID. If the Attributes flag is set, the method will also return the tag attributes.

Inputs

* Account ID:Int:20 → The Account ID.
* Block ID: Int:20 → The Block ID.
* These 3 fields are optional, but only one can be set:
  + (Optional) Tag ID:int:20 → The Tag ID.
  + (Optional) Tag Name:String:64 → The Tag Name.
  + (Optional) Type:ENUM → One of TEXT, IMAGE, FILE. Only return back tags of this type.
* (Optional) Attributes Flag:Boolean → Returns attributes for each tag if set to True.

Outputs

* <standard response message block>
* Tag Array (Optional)
  + ID:Int:20
  + Name:String:64
  + Description:String:256
  + Type:String:64
  + Value:String:4096
  + Attribute Array (Optional)
    - ID:Int:20
    - Name:String:64
    - Value:String:4096

Return Sub Codes

* 0x1780

## Example Usage

* Get all tags from the block ID 3 with the IMAGE type, and all attributes for each tag
  + AccountID: 25
  + BlockID: 3
  + Type: IMAGE
  + AttributesFlag: 1
* Get tags with name of ‘header\_image’ from the block ID 3, with no attributes returned
  + AccountID: 25
  + BlockID: 3
  + TagName: header\_image

### Get Attributes

Function call to get back a list of attributes for the given Tag ID.

Inputs

* Account ID:Int:20 → The Account ID.
* Tag ID: Int:20 → The Tag ID.
* Both of the fields are optional, but only one can be set:
  + Attribute ID:int 8 → The Attribute ID.
  + Attribute Name:String → The Attribute Name.

Outputs

* <standard response message block>
  + Attribute Array (Optional)
    - ID:Int:20
    - Name:String:64
    - Value:String:4096

Return Sub Codes

* 0x17A0

## Example Usage

* Get all attributes from the tag ID 10
  + AccountID: 25
  + TagID: 10
* Get attribute with ID 2 from the tag ID 10
  + AccountID: 25
  + TagID: 10
  + AttributeID: 2

## Carrier API Group

The Carrier section contains the functions necessary to manage devices on a carrier network. This API group is intended to serve as the backend for eCommerce, Customer Service Portals, and IVR systems.

Example Usage:

* Bring your own Device (BYOD) – Workflow for a consumer that already has an available device for a specific carrier and needs to active
  + ProvisionAPI: BatchID = LoadDevice(AccountID, DeviceInfo)
    - Loads the device into Inventory. Reference: Load Device
  + CarrierAPI: changeCarrierOwnership(CarrierName, DeviceID, Ownership)
    - Changes the ownership of the device. For any failures the result description will contain the reason the carrier failed to change ownership. IE: Lost, Stolen, Blacklisted, etc
    - If ownership successfully changed the device can be used within the order process
  + OrderAPI: LineInfo = addOrderDevice(OrderID, ProductID, Flags)
    - For BYOD the flag type must be “2” with the flag string value containing the Device ID. Reference: Add Order Device

### Change Carrier Ownership

This function provides the ability to switch a device on the carrier network from “retail” to “wholesale” or vice versa.

Inputs

* Carrier Name:ENUM 🡪One of “SPRINT”, “VERIZON”
* Device ID:String:128 🡪 ESN/SIM identifier
* Ownership:ENUM 🡪 One of “RETAIL”, “WHOLESALE”
  + RETAIL -> Change the ownership of the device on the carrier network from Wholesale to Retail
  + WHOLESALE -> Change the ownership of the device on the carrier network from Retail to Wholesale. This ownership mode allows for the ability to activate the device on the ReadyWireless network.

Outputs

* <standard response message block>

Return Sub Codes

* 0x81C0 🡪 Success
* 0x81C1 🡪 Internal API Error, escalate issue to support
* 0x81C2 🡪 Invalid Parameter
* 0x81C3 🡪 Device not in Inventory. Load device first using LoadDevice()
* 0x81C4 🡪 Device already assigned to consumer
* 0x81C5 🡪 Ownership change failed, reference ResultDescription for reason

# Flow Examples

This section lays out various use-case scenarios and restrictions.

## Scenario 1

This use-case describes the flow and function calls necessary to order/activate devices and airtime products when the sale of devices and fulfillment process is being managed by an external client or third-party; and Ready Wireless is providing airtime plans/services/and activation. There are generally two methods available to implement this workflow; the in-process method or the post-sale method. Either approach has its advantage and disadvantage. The in-process method requires building the order during the sale process, but provides accurate taxes/fees for the order. The post-sale method creates the order after the eCommerce sales is complete, but cannot provide an order quote with taxes/fees.

The following will describe the In-Process method.

**Note**: This is a guide to implementing the function calls. Not all required input parameters are depicted. This scenario also assumes a PREPAID billing mode. POSTPAID flows are available, but the implementation differs slightly.

Setup

* External party and Ready Wireless define the products and associated custom keys then configure all required systems.
  + **Note**: All inventory items within Ready Wireless systems are keyed by Product ID. For situations where multiple devices or products can fulfill a generic product ID (for example “Flip Phone”) a Pick Group is defined that references each Product ID that can fulfill the type.

**In-Process Method**

During the eCommerce Order

* Site displays available Ready Wireless devices
  + Site calls GetProducts(Inventory Type=’DEVICE’) and displays the available devices.
    - In the scenario where the list of Ready Wireless supported devices is less than the total device list sold by the client/vendor one option is to compare the returned list with the selected device to see if a Ready Wireless airtime product can be purchased. This allows the client/vendor to display any list of devices and still only allow Ready Wireless airtime products to be displayed if the device is supported.
* Consumer selects device to purchase and it matches a Ready Wireless supported device.
  + Site calls BuildNewOrder to start the Ready Wireless order creation process.
  + Site saves resulting NewOrderID is session for future API calls.
  + AddOrderDevice(Add Flag=0x04) is called to add a device product to the order based upon the consumer selection. Add Flag=4 sets a device placeholder within the order that will need to be updated with SetOrderDevice later in the process.
* Consumer requests a Ready Wireless airtime plan
  + Site calls GetProducts(Inventory Type=’AIRTIME’) and displays available airtime products
  + User selects a plan along with auto-replenish options; the site calls AddOrderAirtimeProduct to add the selected product to the order.
  + Use RemoveOrderLineItem when the consumer changes their mind and selects a different product.
    - **Note**: Airtime, Accessory, Extended Service, Add-A-Line products must be associated with a device and removed first before the actual device can be removed from the order.
* Once the consumer is satisfied with their order, eCommerce sites typically display an order verification screen.
  + Site will call GetOrderQuote to provide consumer with final totals (including taxes and fees)
  + When the consumer accepts the order the site calls PlaceOrder to finish the order.

Fulfillment

* Since client/vendor is fulfilling the order (AddOrderDevice(Add Flag=0x04) it is put into a HOLD state. Client/vendor must provision/fulfill devices and then release the order using the following flow.
  + Client/vendor picks physical device from inventory and activates the device on Ready Wireless’ network.
    - External client/vendor system calls BulkLoadInventory to get the device details into the system.
    - ActivateDevice is called with subsequent calls to GetDevice; checking the SubscriberStatus field for ‘ACTIVE’.
      * Note: The ActivateDevice method is an asynchronous routine so calling GetDevice is needed to determine the status. Typical response times range from 1 second to 90 seconds depending upon the carrier.
  + Upon successful activation the client/vendor system calls SetOrderDevice to tie the selected device to the order line item.
  + Once all devices provisioned, client calls ReleaseOrder to continue automated order completion.

Auto Replenishment

* If the consumer requested auto replenish and the client/vendor is handling collecting of payment, Ready Wireless’ system triggers replenishment events and queues the events for client/vendor consumption.
* External system calls GetPendingAutoReplenish to create local queue of accounts to collect payment on.
* Once payment collected, external client/vendor system calls EnablePendingAutoReplenish; which will automatically apply the product load.
* If payment is not collected, system calls CancelPendingAutoReplenish; which takes the event out of the queue.

Consumer Changes Auto Replenish (changes auto replenish product)

* If a consumer changes their auto replenish product the external client/vendor system starts the change order by calling BuildChangeOrder.
* Then ChangeOrderAirTimeProduct is called to change the currently active service order to a new product.
* GetOrderQuote can be called to display the new total change.
* Then PlaceOrder is made to finalize the order and change the auto-replenish product.

## Scenario 2

Dealer order activation and subsequent consumer product loads via eCommerce. Client is fulfilling orders.

Setup

* Client and Ready Wireless define products and client custom keys then configure system.
  + **Note**: All inventory items within Ready Wireless systems are keyed by Product ID. For situations where multiple devices or products can fulfill a generic product ID (for example “Flip Phone”) a Pick Group is defined that references each Product ID that can fulfill the type.
* BulkLoadInventory 🡪 Client loads inventory using pre-defined product IDs/custom keys.

Dealer Order Placed

* BuildWholesaleOrder to start the order creation process.
* AddWholesaleOrderItem is used to add a specific quantity of a product to the order.
  + Depending upon the detail level of the client’s inventory processes either the “auto pick” or “provide” options are used. “Auto pick” will select devices from inventory that are available and client will need to locate the selected devices from inventory and ship to dealer location. With the “provide” option the client will supply an array of device IDs to use.
* PlaceWholesaleOrder is then called to assign the designated inventory to the dealer selected and start the activation process.
* Multiple calls to GetWholesaleOrder can then be made to check the order status. Once the order state is “Complete” a full list of all device IDs is returned.

Consumer PIN Loads

* If an eCommerce portal is available for virtual PIN purchase the following flow is used.
* BuildNewOrder to start the consumer order process.
* AddOrderPinProduct to purchase future PINs.
* GetOrderQuote to display the full total to the consumer before payment collected.
* PlaceOrder to complete the order process.
* Then call GetPinProducts to retrieve the newly purchased Pins. Consumer can then be presented with applying the Pins then or for future consumption on the site or via IVR.
  + Call ApplyPinProduct to put the PINs into use immediately.

Consumer Product Auto Replenish

* If a mechanism will exist for setting up auto-replenishment for dealer distributed devices follow the steps outlined in Scenario 1.

## Scenario 3

Typical eCommerce order flow when Client is delivering consistent eCommerce services for multiple partner companies within the same site (URL). In this scenario the site will “rebrand” once it determines the partner (when the consumer enters a promo code).

This flow is for “Always On” or postpaid accounts. Converting this to a prepaid flow would only require the following modification of the order flow.

* The BuildNewOrder call would set Billing Mode to ‘POST’ or ‘POST SHARED’
* Adding products to the order would require AIRTIME products rather than PLAN products.
* Support of anonymous purchases would require login via PIN code rather than username/password.
* Extended service plans may/may not be needed.

Consumer Flow for a typical “Get Started” order flow is as follows:

* User enters site for the first time this session (new site session started)
  + Web site session account would be defaulted to the client level Account ID hard-coded during site setup. All new sessions will begin this way.
  + Because we do not know the user’s associated partner, the site will display the products assigned at the default site account level by calling the GetPartnersLookup API call without a PartnerID set. This will return the top-level ‘ECOMMERCE’ type account information.
* User enters the “Get Started” flow.
  + User is presented with an option to login to their existing account if they have one.
    - If User does attempt login
      * API Call 🡪 GetConsumerAccountsLookup(username)
      * API Call 🡪 AuthenticateConsumerAccount(customer id, username, password)
      * If passed, save returned Partner ID, Customer ID in local session variables. Site should now display products associated with the Partner account ID.
  + If user not logged in, display Account Type Selection and Promo Code page
    - User selects their account/billing type (prepaid/postpaid) and optional promo code.
    - Save account type and promo code to local session variables.
    - API Call 🡪 GetPartnersLookup(promo code)
      * Save PartnerID to local session variable
  + API Call 🡪 BuildNewOrder(<pass in Customer ID if user logged in>, PRE/POST/POST SHARED, PartnerID if promo code entered)
    - Save returned Customer ID in local session variable if user not logged in.
    - Save returned Order ID in local session variable.
    - Start local session order/cart array. At minimum store information that ties the display item with the order LineID.
* User moves to the Wireless Coverage Check page
  + User is prompted to enter their zip code
  + API Call 🡪 WirelessCoverageCheck(Partner ID, ALL carriers,zip code)
  + Display results to user, user selects carrier. Carrier selection is limited by the account type. This must be hardcoded into the flow of the site.
* Flow moves to the device selection page
  + API Call 🡪 GetProducts(PartnerID,InventoryType=DEVICE,Details=0x02,Carrier=User Selected)
    - Save the returned products, features, associations,dependencies into local session variables
  + Display phone selection page to user, user selects phones
  + For each phone selected
    - API Call 🡪 AddOrderDevice(OrderID,ProductID of device, 0x01)
      * Save returned LineID in local order session array
* Next page in flow is plan selection
  + API Call 🡪 GetProducts(PartnerID,InventoryType=PLAN,Details=0x02,Carrier=User Selected)
    - Save the returned plans, features, and plan dependencies into local session variables
  + API Call 🡪 GetProductDependencies(PartnerID, SelectedDeviceProductIDArray, PLAN group)
    - Match up the device dependency limits with the returned plans in order to display which plans are available for each device and which plans must be selected to move to the next page in the flow. Review the plan dependency array from the plan GetProducts call as well to ensure all plan dependencies are accounted for. There may be cases where a particular voice plan requires a text or data plan.
  + Display the plan selected page, user selects plan(s)
  + For each plan selected
    - API Call 🡪 AddOrderAirtimeProduct(OrderID, LineID of device, ProductID, prepaid flag)
      * Save returned LineID in local order session array
* The next page may be for adding Accessories
  + API Call 🡪 GetProducts(PartnerID,InventoryType=ACCESSORY,Details=0x02,Carrier=UserSelected)
    - Save the returned products,features,dependencies into local session variables.
  + API Call 🡪 GetProductDependencies(PartnerID, SelectedPhonesProductIDArray, ACCESSORY)
    - Match up the dependency requirements of the phones to determine which accessory goes with which phone.
  + Display the available accessories
  + For each plan selected
    - API Call 🡪 AddOrderAccessoryProduct(OrderID,LineID of device,ProductID)
      * Save returned LineID in local order session array
* Extended service plan selected could be next for some sites
  + API Call 🡪 GetProducts(PartnerID,InventoryType=EXTENDED,Details=0x02,Carrier=UserSelected)
    - Save the returned products,features,dependencies into local session variables.
  + API Call 🡪 GetProductDependencies(PartnerID, SelectedPhones/PlansProductIDArray, EXTENDED)
    - Match up the dependency requirements of both the phones and plans to determine which extended service plans can be offered for each phone (or possibly plan).
  + Display the available extended service plans(s)
  + For each plan selected
    - API Call 🡪 AddOrderExtendedProduct(OrderID,LineID of device/plan,ProductID)
      * Save returned LineID in local order session array
* Now entering the order account/payment page
  + If user is not logged in, ask them again if they have an account and would like to login
    - If user attempts login
      * API Call 🡪 GetConsumerAccountsLookup(username)
      * API Call 🡪 AuthenticateConsumerAccount(customer id, username, password)
      * API Call 🡪 MoveOrderToExistingCustomer(OrderID,existing CustomerID)
      * Reset CustomerID in local session variable to the real Customer ID
  + Display the account and billing application page
    - If user is logged in, prefill the billing, shipping, and payment info
    - User then enters or updates billing, shipping, payment info and submits order
    - If existing customer
      * If site allows one-time shipping location and payment info you may need to override the default shipping, billing, and payment profiles set on the customer account. Use the following API calls as needed depending upon how the site flow is coded.
        + API Call 🡪 Add Order Location(OrderID, Shipping Location Flag)
        + API Call 🡪 Add Payment Profile(Payment Info, Billing Location Info)
      * Else the customer wants to update their profile or use the defaults.
        + Update billing, shipping, payment info on current Customer profile is user wishes

API Call 🡪 AddPaymentProfile to add new payment and billing info or

API Call 🡪 ChangePaymentProfile to update existing payment and billing info

API Call 🡪 ChangeConsumerAccount(Shipping Location Info)

* + - If this is not an existing customer we need to create a consumer account and move the anonymous order over.
      * API Call 🡪 AddConsumerAccount(…shipping, billing info from user…)
      * API Call 🡪 AddPaymentProfile(Payment Info, Billing Info(if needed))
      * API Call 🡪 SetConsumerPaymentAccount(new Customer ID, Payment Profile ID) to set the default payment profile for this customer account.
      * API Call 🡪 MoveOrderToExistingCustomer(OrderID,new CustomerID) moves the order from the anonymous customer account over to the newly created named account.
      * Reset CustomerID in local session variable
    - Now that we have a payment profile identified, set it on the order so we know how to charge the customer.
      * API Call 🡪 SetOrderPaymentAccount(Order ID, Payment Profile ID)
  + Confirm the Order Total with the customer
    - API Call 🡪 GetOrderQuote(Order ID) returns a detailed order with tax and totals
    - Display the details to the customer and wait for button click confirmation before moving to the finalize order step.
  + Finalize the order
    - API Call 🡪 AuthorizePayment(CustomerID, PaymentProfileID, amount…)
    - API Call 🡪 PlaceOrder(OrderID)
    - Display order confirmation details to user, email receipt

**Notes**

* Multiple orders can be maintained simultaneously to handle scenarios in the future where a customer may want to open a shared post-paid account while also purchasing a pre-paid product. Behind the scenes this situation would generate two billing accounts; one for the post-paid and one for the pre-paid service.
* During error scenarios there are numerous API calls available to cancel/close customer accounts, orders, payment profiles, etc. Please reference the API for those details.
* Product dependency checking is a complex process. The process below is a general outline for how dependency checking should be done using the dependency arrays returned by GetProducts and GetProductDependencies.
  + Each product can have a set of dependency rules. Generally speaking DEVICE products will have dependencies on PLANs or EXTENDED services. PLANs may have dependencies on other PLANs and so on.
  + For example, a smartphone device might require one data plan of a specific allowance and restrict others that do not have as high a data allowance…say any plan with 2GB or higher, but not 250MB plans. Smart phone product dependencies returned by GetProducts or GetProductDependencies would return something like the following table for that product’s PLAN dependency group.

|  |  |  |  |
| --- | --- | --- | --- |
| Product ID | Plan Type | Dependency Type | Comment |
| 1000 | VOICE 5000 | Require One | High-End Device |
| 1001 | VOICE 2000 | Require One | High-End Device |
| 1002 | VOICE 100 | Restricted | Low-End Device |
| 2000 | SMS 5000 | Require One | High-End Device |
| 2002 | SMS 1000 | Require One | High-End Device |
| 2003 | SMS 100 | Restricted | Low-End Device |
| 3000 | DATA 2000 | Require | High-End Device |
| 3001 | DATA 100 | Restricted | Low-End Device |

* + With the configuration given in the above table the web site would need to only display voice, sms, and data plans in the High-End Device category and require one of the voice and sms plans, and require the high-end data plan. Conversely, a qwerty phone would have dependency types most likely set to “Require One”.
  + Even with the detailed dependency structure defined above it may still be possible for a high-end plan to require another plan as a bundled service. For example the VOICE 5000 plan may also require the SMS 5000 AND DATA 2000 plan if selected. To determine if this is how the products are configured the site will need to evaluate and merge dependency structures from both the DEVICE product list and the PLAN product list. For example, the VOICE 5000 plan product could have a dependency configuration like the following table; which if the user selects the VOICE 5000 plan to satisfy the smart phone device requirements, the site will then need to override the device dependency and require the SMS 5000 and DATA 2000 plans because the VOICE 5000 plan has a higher “Required” dependency.

|  |  |  |
| --- | --- | --- |
| Product ID | Plan Type | Dependency Type |
| 2000 | SMS 5000 | Required |
| 3000 | DATA 2000 | Required |

* + The most local dependency defined wins. In this case the DEVICE product indicated a minimum set of requirements for available PLANs, but one of the plans (the VOICE 5000) had a direct dependency that overrides the DEVICE requirement because it is the closest to the product type selected.

## Scenario 4

This use-case describes the flow and function calls necessary to order/activate devices and airtime products when the sale of devices and fulfillment process is being managed by an external client or third-party; and Ready Wireless is providing airtime plans/services/and activation.

New Customer Order Use-Case

**Starting on the client/vendor website**

* Consumer is shown a display of available devices to purchase from client/vendor stock.
  1. Note: Sprint devices are in RETAIL mode and will need to be brought into WHOLESALE during provisioning.
* Consumer selects one or more devices to purchase and indicates they are ready to complete the order.
* Client/vendor checks to see if selected devices are Ready Wireless supported devices.
  1. Client/vendor site calls Ready Wireless API **GetProducts(“DEVICE”, summary flag)** for list of supported devices and compares to selected devices.
* If devices are supported, the Client/vendor site displays plan offerings to the Consumer.  If not supported proceed directly to normal checkout process.
  1. Note: Plan offering page is a static page that prompts the Consumer to select an account type (PRE, POST, POSTSHARED), input a promo code (if available), and check coverage.
  2. Client/vendor site calls Ready Wireless API **WirelessCoverageCheck(zip code)** to check coverage.
* Consumer will then either decline or accept plan offer.
  1. If Consumer declines the offered plan, proceed directly to the normal checkout process.
  2. If Consumer agrees to plan purchase the Client/vendor site creates a Ready Wireless order then continues to the normal checkout process.
     1. Requires calls to **BuildNewOrder(shared toggle)**, and **AddOrderDevice(add flag=0x04)** with the proper device product ID based on the account type (PRE, POST, POSTSHARED), **AddOrderPromoProduct()** for each device if the Consumer has input a promo code.
     2. Client/vendor will need to save the OrderID with their local Consumer account for future reference and status checks.
     3. Possible Issue:  The Client/vendor site will be required to know the account type (PRE/POST/POST SHARED) in order to use the existing API flow and select the correct device product ID.
* Consumer completes the device purchase process normally and provides Client/vendor payment for the devices.
  1. Note: Client/vendor will maintain their own Consumer account for device purchases.  Ready Wireless will maintain an account for plan/accessory purchases and MRC payments.
* If Consumer agreed to a plan purchase they are provided a “Transferring to Ready Mobile” page and the Client/vendor site launches the Ready Mobile purchase workflow.
  1. The Client/vendor site will need to pass the OrderID returned by BuildNewOrder and security token into the ReadyMobile site at a specific URL location with param &setorder=###.
     1. Issue: This has a security concern that will require an API change to generate a security token.
     2. Issue: The Ready Mobile site will need to be modified to accept a mid-order launch point and auto-populate the order up to the point of plan selection.

**Control transfers to the Ready Mobile website**

* Note: Once on the Ready Mobile site, if the Consumer abandons the order at any time we will do the following.
  1. ReadyWireless will not do anything.  The system will clean-up abandoned orders automatically after 48 hours.
  2. Future Enhancement: If email address available a notice email could be sent with a link to continue the order process.
  3. Client/vendor will need to periodically call the Ready Wireless API **GetOrdersLookup()** and check the status of the orders requiring fulfillment by them.  They will need to determine what the frequency is and if the order is not updated to “HOLD” within X hours they should abandon provisioning and fulfill the order as an inactivated device.
     1. If this occurs and the Consumer did want this account they will need to call Ready Wireless customer service to rectify.
* Given that Consumer is serious about plan purchase, the Consumer is shown the initial plan selection screen and selects plans available for each device on the order.
  1. Note: The site has prepopulated the cart and devices already purchased from the OrderID passed in by Client/vendor.
* Consumer is then shown display of available accessories per device and selects accessories for each device on the order.
  1. Note: Since the Ready Wireless API/Backend does not currently support “split fulfillment” we will not be offering accessories at this time.
* Consumer is shown display of available service plans per device and selects service plans.
  1. Note: Since the device is being sold by Client/vendor we will not be supporting extended service plans at this time.
* Consumer moves to the order confirmation screen and provides billing, shipping, and payment information.
  1. Note: Since there will not be any physical products shipped from Ready there will not be any shipping products required.
  2. Issue: This may require special coding within the ReadyMobile website to skip the shipping selection step if shipping products are not defined.
* Consumer completes the order.
  1. ReadyMobile site will call **PlaceOrder()** and **AuthorizePayment()**, this will place funds on hold and move the order from BUILDING to HOLD state.  The order will remain in the HOLD state until Client/vendor fulfills each device on the order.

**Control transfers to Client/vendor order back-office processes**

* Client/vendor receives notification of completed Ready Mobile order and begins the fulfillment process on their end.
  1. Client/vendor periodically calls Ready Wireless API **GetOrdersLookup()** and processes all orders in HOLD status.
* Client/vendor provides Ready Wireless with device details and requests Ready Wireless activate each device on the order.
  1. Client/vendor calls Ready Wireless API **LoadDevice()** with device details to save the necessary information into the provisioning system.
     1. Ready system will migrate Sprint devices to WHOLESALE during this step.
  2. Client/vendor calls Ready Wireless API **ActivateDevice()** to light-up the device on the appropriate carrier network.
  3. Client/vendor calls periodically the Ready Wireless API **GetDevice()** to check on the activation status.
  4. If device successfully activated, Client/vendor calls Ready Wireless API **SetOrderDevice(order device line item id)** to associate the new device with the Consumer order.
  5. If device not activated, Client/vendor repeats this section until activated devices are available to associate with the Consumer order.
* Ready provides back the resulting MDN to Client/vendor for inclusion with the shipped device.
  1. When Client/vendor calls the Ready Wireless API **GetDevice()** to determine the activation status they will also have the MDN assigned.
* Client/vendor tests each device and ships the packaged products; along with a Ready Wireless support/instruction guide.
* Client/vendor notifies Ready Wireless of the shipped package tracking information and expected delivery timeframe.
  1. Client/vendor calls Ready Wireless API **ReleaseOrder()**.
  2. Future Enhancement: Develop the workorder process that allows Ready to process “split fulfillment” so accessories can be sold on these orders.
  3. Ready Wireless order workflow completes the order and charges Consumer for purchased products based upon the expected delivery timeframe.

**Control passes back to the Consumer**

* Consumer receives shipped purchase and confirms receipt of device(s) and operational state of device(s).
* If expected device(s) are not received Consumer calls Client/vendor support.
* If device(s) are not operational Consumer calls Ready Wireless support.
* If shipment is correct and devices operational Consumer may access either the Client/vendor site or Ready Mobile site for My Account information/functions related to their purchase.

## Scenario 5

These high-level POST/POST-SHARED use-cases describe the flow and function calls necessary to change airtime plans, billing\_modes, add a line, or add an accessory/ESP to an existing order. Orders within the ReadyWireless OSS Platform remain active until all products associated with an order have been deleted/canceled or replaced. Therefore in order to change a previous purchase it is necessary to know the order ID and order line item(s) being modified.

The following sections represent the most common use-cases.

**Change Plan (same billing mode with a different plan)**

* GetOrdersLookup OR BETTER YET call GetConsumerActiveProducts to retrieve the current orders/line items.
* BuildChangeOrder passing in the original order ID.
* ChangeOrderAirtimeProduct passing in the original plan order line item ID and new plan product ID.
  + **Note:** The correct plan product ID must be provided that matches the current billing mode.
* If needed, AddOrderFeeProduct
* If needed, AddOrderPromoProduct
* PlaceOrder

**Change Plan (change billing mode and plan)**

**Note:** If changing to POST SHARED must have two or more lines on the account; either active or included within this change order.

* GetOrdersLookup OR BETTER YET call GetConsumerActiveProducts to retrieve the current orders/line items.
* BuildChangeOrder passing in the original order ID and the new billing mode.
* ChangeOrderAirtimeProduct passing in the original plan order line item ID and new plan product ID.
  + **Note:** The correct plan product ID must be provided that matches the new billing mode.
  + **Note:** If switching from POST SHARED to POST the ChangeOrderAirtimeFunction must be called multiple times for each device on the account to facilitate setting a different airtime plan for each device.
* If needed, AddOrderFeeProduct
* If needed, AddOrderPromoProduct
* PlaceOrder

**Change Plan (change billing mode only)**

**Note:** If changing to POST SHARED must have two or more active lines on the account.

**Note**: The current airtime plan (shared version) of the primary line will be used if switching to POST SHARED.

**Note**: Switching from POST SHARED to POST will automatically apply the non-shared version of all airtime plans to each device on the account.

* GetOrdersLookup OR BETTER YET call GetConsumerActiveProducts to retrieve the current orders/line items.
* BuildChangeOrder passing in the original order ID and the new billing mode.
* PlaceOrder

**Add a Line**

* GetOrdersLookup OR BETTER YET call GetConsumerActiveProducts to retrieve the current orders/line items.
* BuildChangeOrder passing in the original order ID
* AddOrderDevice
* If not POST SHARED, AddOrderAirtimeProduct
* If needed, AddOrderAccessoryProduct
* If needed, AddOrderExtendedProduct
* If needed, AddOrderFeeProduct
* If needed, AddOrderPromoProduct
* PlaceOrder

**Add Accessory/ESP**

* GetOrdersLookup OR BETTER YET call GetConsumerActiveProducts to retrieve the current orders/line items.
* BuildChangeOrder passing in the original order ID
* AddOrderAccessoryProduct passing in the original device order line item ID to associate this accessory with.
* AddOrderExtendedProduct passing in the original device order line item ID to associate this accessory with.
* If needed, AddOrderFeeProduct
* If needed, AddOrderPromoProduct
* PlaceOrder