



Zauí Application Developers Interface Manual

Currently only available in English

Recent API Changes

Date of Change	Description of changes
June 16, 2015	<p>Updates to the zapiGetActivityDetailsById and zapiCheckActivityInventoryByDate callbacks. New parameters added, and results for the activity times standardized.</p> <p>New rental data structure added on the result set for checking inventory.</p>
June 16, 2015	<p>Added new section in this document outlining common XML nodes that are resused throughout the API.</p> <p>Updated the call back restrictions for the batch availability check</p>
June 16 2015	<p>Deprecated the following routines:</p> <p>zapiIntrospect zapiAddTransactionToBooking zapiGetFeaturedActivity zapiCreateReservation zapiCancelAndReverseReservation zapiGetActivitiesByCompanyId zapiCheckAvailabilityForActivitiesAndProductsByDate</p>
July 13 2015	<p>Addition of the following routines:</p> <p>zapiAgentLogin</p>
July 14 2015	Updated the documentation adding in all new package based API calls
July 14, 2015	Added currency conversions to the cart totals, when configured by the user in the backoffice.
July 16, 2015	<p>Batch availability response structure updated to use the common activity times availability and inventory node.</p> <p>Restructured the batch availability result structure.</p>
September 1 2015	Addition of new routines supporting interaction with waitlists
September 3 2015	Addition of a common passengers XML node
September 15 2015	Addition of new access routines for Guests and Guest data
September 17 2015	Addition of the API to determine the best dates to book a package (Determine Minimum Dates to Book)

	Addition of new zapiGuest routines set and access login authenticate and expiry dates on the account
September 17 2015	Addition of an agents authentication routine
September 22 2015	Addition of the new package fields to indicate if the packages provides the ability to have open dates on its components
September 23 2015	Addition of the following new routines: zapiGuests_GetBookingDetails
November 6 2015	Addition of a new node that describes the day of the week, by date ranges, that an activity is available. A helper node was added, called DAYS_OF_WEEK_AVAILABLE, and this XML node was added to the zapiGetActivityDetailsById callback.
November 12 2015	Addition of description of the “exponential back off” algorithm that we encourage users of the API to implement to effectively handle API throttling when encountered in client systems
November 18 2015	Addition of new API calls to list, create and remove activity cancellations
November 24 2015	Addition of new field indicating if a activity time is cancelled for online bookings only. This change affects the Activity Times Node, and is returned as part of the overall Activity Node.
November 26 2015	Addition of new API calls: zapiAgents_GetAllCompanies zapiPackage_SetActivityComponentDate
February 1 2016	Addition of updating a guests gender on the API zapiGuests_UpdateProfile
February 22 2016	Changes to zapiGetPackageDetailsByPackageId request XML <ul style="list-style-type: none">• Addition of optional seed package date when called Changes to zapiGuests_GetBookingDetails response XML <ul style="list-style-type: none">• Addition of onStandby, onWaitList, waitListId fields – See details in the
February 23, 2016	Addition of a new API call which will return a complete XML description of all pricing components and details for a single package, regardless of sale or travel date. zapiPackage_RetrievePricingRates Updates to the zapiGetPackageDetailsByPackageId

	<ul style="list-style-type: none"> • New field packageType – see appendix E for system package types and a description of those <p>New Appendix for package type descriptions New Appendix for product type descriptions New XML node that describes a single price code New XML node that describes the configuration for multi-day itinerary based packages</p>
March 4 2016	Introduction of the new API call <code>zapiPackage_DetermineMinimumDatesToBook</code> . This routine helps determine, based on a package, and seed date, if all components of the package are bookable.
March 14 2016	A small change has been added to the <code>zapiAddProductToCart</code> , for percentage based product, you can now indicate a linked activity or package, also added to that, for which to base the product price. See the addition notes for this call below.
March 29 2016	API change notification set for April 25 2016, where the customer XML Node on the cart node will change. New change is documented in this document.
May 5 2016	Update the <code>zapiPackage_DetermineMinimumDatesToBook</code> . See details below for calling interface and return result.
May 6 2016	Updated the documentation and return data for any search result, specifically the data returned within a single booking, when calling any of the search routines on the API Added in the section defining the standardized booking data returned as part of the search results. Addition of transaction details as part of the returned data set for to search results (via booking number, last name, mobile phone) On all search results, added the agent details responsible for the booking, their name and company.
May 9 2016	Addition of rental activity details as part of the returned search details, section 2.4.18
June 2 2016	Addition of the agent Id, that optional can be passed through for payment transactions that provision for the company ID and agent ID, such as vouchers. See <code>zapiProcessCartWithPayment</code> and <code>zapiProcessSingleTransaction</code>
June 8 2016	Addition of the categoryName, imageUrl and description to the <code>zapiGetActivitiesByCategoryId</code>
July 18 2016	Addition of isFreeSell, ticketExpiry, ticketPrinterText to activities node

July 21 2016	Addition of new fields operatedByCompanyId and operatedByCompanyName which describe the operating company for any given activity. Part of the zapiGetActivityDetailsById routine
July 21 2016	Remove Supplier Confirmation Number add as part of API calls where the system returns details of any single activity component, part of a booking, or an existing cart order. The field is called supplierConfirmationNumber, which describes the booking reservation number returned from a remote Zauí system, and is part of the Zauí to Zauí project.
July 29 2016	Addition of the passengers configured as part of the overall system settings now as part of the zapiGetSystemInformation routine
August 1 2016	Addition of airlines that are configured within the system as part of the zapiGetSystemInformation routine
August 3 2016	Fixed misspelling of XML nodes in the zapiProcessSingleTransaction on the request structure
August 11 2016	Added as part of the returned payload on applying a promo code, the cart/order object. API calls affected: zapiPromoCodeApply , zapiGiftCertificateApply
August 14 2016	Addition of a new API call, allowing for cancellation of a booking within the system. zapiBookings_CancelBooking
August 21 2016	Addition of supplierConfirmationNumberSystemManaged and supplierConfirmationNumberDisplayed fields indicating how remote supplier confirmation numbers are handled for each activity. API calls affected: zapiGetActivityDetailsById , zapiAddActivityToCart
August 26 2016	Addition of verboseMode flag to the following API calls, default value in all cases is true : zapiGetPackagesByCategoryId zapiGetPackageDetailsByPackageId zapiGetActivitiesByCategoryId zapiGetActivityDetailsById The purpose of the verboseMode is to provide a mechanism of obtaining a light-weight method of obtaining a smaller sub-set of data from the system, thereby reducing time required to generate large data sets, the time required for the API call to generate the data, and also the returned data payloads.

March 20 2017	zapiProcessSingleTransaction – for credit card payments, the addition of the XML item <currencyId>. If currencies are configured in your system, then the Cart Obj will return, for each currency, a converted total including tax, for each currency. This currency ID, can then be passed along to the payment processing, to indicate the credit card amount is to be transacted with a specific (non-based currency) denomination, and the converted amount. Currencies can also be configured or a specific payment gateway.
August 14 2017	zapiPackage_DetermineMinimumDatesToBook – new fields were added to the return data on this API call, for the stand by, namely, the number of guests currently on standby, and if standby is available for a particular leg of journey. <numberOnStandby></numberOnStandby> <isStandbyAvailable></isStandbyAvailable>
January 16, 2018	Booking search results – addition of new fields, guestId and guestHashKey
January 20, 2018	Guest Standby logic has been added to many API calls, standardization of the standby details, and extra logic added to zapiPackage_SetActivity-ComponentDate to allow adding a guest to standby. The following changes were made as part of guest standby:
	<ul style="list-style-type: none"> • zapiPackage_SetActivityComponentDate (API request) <ul style="list-style-type: none"> ◦ added <bookingId> (optional) ◦ added <addToStandBy> (optional) ◦ added new logic to allow adding to standby when setting the component travel date • zapiPackage_SetActivityComponentDate (API response) <ul style="list-style-type: none"> ◦ added <standBy> and <numberOnStandBy> • zapiPackage_DetermineMinimumDatesToBook (API request) <ul style="list-style-type: none"> ◦ deprecated the <staggerDates>

TABLE OF CONTENTS

1	Preface.....	11
1.1	ZAPI Commercial Terms	11
1.2	Intended Audience.....	11
1.3	Legal Statement.....	11
1.4	Notice of non-liability	11
2	Getting Started	11
2.1	Authentication Credentials.....	11
2.2	Creating an API Token	12
2.3	Submitting to the API.....	12
2.3.1	ZAPI Server URI	12
2.3.2	ZAPI Ping test.....	13
2.3.3	API Throttle	14
2.3.4	Implementing Exponential Back-off.....	14
2.4	Supporting XML Nodes	15
2.4.1	Single Booking Details Node.....	15
2.4.2	Currencies Node.....	17
2.4.3	The Cart Node.....	17
2.4.4	Activity Times Node	18
2.4.5	Business Hours Node	18
2.4.6	Pickup Locations Node	19
2.4.7	Drop Off Locations Node	19
2.4.8	Rental Date/Times Nodes	19
2.4.9	Allowed Guest Types	19
2.4.10	Wait List Node	20
2.4.11	Passengers List Node	20
2.4.12	Guest or Agent Details Node	20
2.4.13	User Custom Fields Node	21
2.4.14	Transaction Details Node.....	21
2.4.15	Days of Week Availability Node.....	23
2.4.16	Price Code Node(s)	24
2.4.17	Package Multi Day Itinerary Node	24
2.4.18	Booking node – search results	25

2.4.19	Mobile data Node.....	26
2.5	General / Misc API Calls.....	27
2.5.1	zapiPing.....	27
2.5.2	zapiGetSystemInformation	28
2.5.3	zapiGenerateBarCodeImage	29
2.6	Interacting with a Cart Session API Calls	30
2.6.1	zapiClearCartSession	30
2.6.2	zapiRemoveCartSession	31
2.6.3	zapiGetAllCartSessions	31
2.6.4	zapiGetCartContents	32
2.6.5	zapiUpdateNotesToCart	34
2.6.6	zapiUpdateCustomerDetailsToCart	35
2.6.7	zapiAddProductToCart	36
2.6.8	zapiRemoveProductCartItem	37
2.6.9	zapiAddActivityToCart.....	38
2.6.10	zapiRemoveActivityCartItem	39
2.6.11	zapiLoadingBookingIntoCart	40
2.6.12	zapiProcessSingleTransaction.....	41
2.6.13	zapiProcessCartWithPayment.....	41
2.6.14	zapiCreateUniqueCartSession.....	43
2.6.15	zapiPromotionCodeApply.....	43
2.6.16	zapiGiftCertificateApply.....	44
2.7	Package Based API Calls	45
2.7.1	zapiGetPackageCategories.....	45
2.7.2	zapiGetPackagesByCategoryId.....	46
2.7.3	zapiGetPackageDetailsByPackageId	47
2.7.4	zapiAddPackageToCart.....	48
2.7.5	zapiPackage_DetermineMinimumDatesToBook.....	49
2.7.6	zapiPackage_SetActivityComponentDate	50
2.7.7	zapiPackage_RetrivePricingDetails	51
2.8	Activity Based API Calls.....	52
2.8.1	zapiGetActivityAndProductCatalog	52
2.8.2	zapiGetActivityCategories	52

2.8.3	zapiGetActivitiesByCategoryId	53
2.8.4	zapi GetAllActivitiesByName	54
2.8.5	zapi GetAllActivitiesByDate	55
2.8.6	zapiGetActivityDetailsByActivityId.....	56
2.8.7	zapiCheckActivityInventoryByDate	58
2.8.8	zapiPriceQuote (Rentals Only)	60
2.8.9	zapiActivity_GetCancellations	60
2.8.10	zapiActivity_CreateCancellation	61
2.8.11	zapiActivity_RemoveCancellation	63
2.9	Merchandise / Product Based API Calls	63
2.9.1	zapiGetMerchandiseCategories	63
2.9.2	zapiGetMerchandiseByCategoryId	65
2.9.3	zapi GetAllMerchandiseByName	66
2.9.4	zapiGetProductDetailsByProductId.....	67
2.9.5	zapiCheckProductInventory.....	68
2.9.6	zapiSetProductQuantity	69
2.10	Wait List API Calls	70
2.10.1	zapiAddWaitListGuest	70
2.10.2	zapiRemoveWaitListGuest.....	70
2.10.3	zapiConvertWaitListGuestToCart	71
2.10.4	zapiGetAllWaitListGuestsByActivityIdAndDate	72
2.11	Agent Specific Calls	72
2.11.1	zapiAgents_Authenticate	72
2.11.2	zapiAgentLogin.....	73
2.11.3	zapiAgents_GetProfile	74
2.11.4	zapiAgents_UpdateProfile	74
2.11.5	zapiAgents_UpdateUsername	75
2.11.6	zapiAgents_UpdatePassword.....	75
2.11.7	zapiAgents_GetAllCompanies.....	76
2.12	Guest Specific Calls	77
2.12.1	zapiGuests_Authenticate	77
2.12.2	zapiGuests_GetProfile	77
2.12.3	zapiGuests_Lookup.....	78

2.12.4	zapiGuests_UpdateProfile.....	78
2.12.5	zapiGuests_UpdateUsername	80
2.12.6	zapiGuests_UpdatePassword	81
2.12.7	zapiGuests_UpdateAccountExpiryDate	81
2.12.8	zapiGuests_GetBookingDetails	82
2.13	Searching API Calls.....	82
2.13.1	zapiSearchBookingsByLastname	82
2.13.2	zapiSearchBookingsByBookingNumber	83
2.13.3	zapiSearchBookingsByMobileNumber	84
2.14	Manifest API Calls.....	85
2.14.1	zapiGetManifestEntireDayByDate	85
2.14.2	zapiGetManifestIndividualByActivityIdAndDate	86
2.15	General Booking/Reservation Based API Calls	87
2.15.1	zapiEmailBookingItinerary	87
2.15.2	zapiCheckIn	88
2.15.3	zapiSetWaiverSignedValue	89
2.15.4	zapiUpdateReservationActivityComponent	89
2.15.5	zapiBatchAvailability.....	90
2.15.6	zapiBookings_CancelBooking.....	91
2.16	Zauí Mobile Specific API Calls.....	92
2.16.1	zapiMobileLogin.....	92
2.16.2	zapiLogout	93
3	APPENDIX A – PACKAGE TYPES.....	95
4	APPENDIX B - ACTIVITY TYPE ID'S	96
5	APPENDIX C - PRODUCT TYPE ID'S	97
6	APPENDIX D - ZAPI ERROR CODES	98
7	APPENDIX E - ZAPI PAYMENT METHOD TYPES	98
8	APPENDIX F - ZAPI METHOD ERROR CODES.....	99

Table of Contents

1 PREFACE

1.1 ZAPI COMMERCIAL TERMS

The ZAPI is geared toward providing partners, organizations and end customers of Zauí Software with a straight forward way of integrating Zauí into your existing applications, services and working environment.

1.2 INTENDED AUDIENCE

The typical user of ZAPI is a power user of Zauí Software who wishes to have a more in-depth integration of the Zauí system and its data with existing or new system or third party system in the Zauí client network.

1.3 LEGAL STATEMENT

This guide, in addition, to the software described within, is under the copyright owned by Zauí Software Limited and subject to license. This guide and any included software, whole or in part, cannot be published, downloaded, or reproduced, transmitted, transferred or combined with any other material or be used for any other purpose without written permission from Zauí Software Limited.

1.4 NOTICE OF NON-LIABILITY

Every effort has been made to ensure the accuracy of information published in this guide. However, Zauí Software cannot accept any responsibility for any errors, inaccuracies, or omission that may or may not be published in this guide.

Zauí Software is providing the information in this guide to you “AS-IS” with all faults. Zauí software makes no warranties of any kind (whether express, implied or statutory) with respect to the information contained herein. Zauí Software assumes no liability for damages (whether direct or indirect), caused by errors or omissions, or resulting from the use of this document or the information contained in this document or resulting from the application or use of the product or service described herein. Zauí Software Limited reserves the right to make changes to any information herein within prior notice or consent.

2 GETTING STARTED

2.1 AUTHENTICATION CREDENTIALS

A word on some important items on the topic of authentication with your Zauí system:

- Use of the ZAPI is always through an existing user (contact) in your Zauí system.
- There is no special ZAPI user.
- You are required to add a new contact to your system when you wish to generate a new token.

- Each application/service/user must have a single unique token
- Security is provided through your existing SSL certificate and is the only means of communicating with your Zauí system

Before starting, anyone organization wishing to use this API, must have been provided authentication credentials.

2.2 CREATING AN API TOKEN

Once you have created the contacts within your system you can now create and manage their ZAPI tokens. Tokens are managed from the **Contacts | Manage ZAPI Partners** from within in your Zauí system.

To create a new token:

- click Create ZAPI Partner from the left panel menu
- select the contact/application/service from the list
- click Save

Once you have added your new service to the system, you're not able to bring up the "details" for this particular partner. Some of the details include the following:

- Affiliate user name – this is the contact's name as you had created it
- Company/Organization – this is the parent organization that your contact is associated with
- ZAPI Token – the token that must be used for this partner to communicate with your Zauí system.
- Account Id: the account Id generated by your Zauí system. If your contact has a parent organization then this will have a value greater than zero.
- User Id: this is the unique Id generated by your Zauí system when you created the new contact in your system.
- Last 20 ZAPI requests and responses from this client – this give a detailed break down of the last 20 communications between the systems.

2.3 SUBMITTING TO THE API

2.3.1 ZAPI SERVER URI

All ZAPI requests to your Zauí system must be sent on the following URI:

<https://<mydomain>.zaui.net/zapi/>

- * All requests are POST
- **All requests are in XML, properly formatted for each function call
- ** All responses are in XML

2.3.2 ZAPI PING TEST

The ping ZAPI test is a great way to test if all components of your ZAPI client are working correctly.

PHP Request:

```
<?php
//Change the following
//
$host      = "hostUrl";
$token     = "zapiApiToken";
$accountId = "accountId";
$userId    = "userId";

$xmlRequest = ("<request>
    <zapiToken>$token</zapiToken>
    <zapiAccountId>$accountId</zapiAccountId>
    <zapiUserId>$userId</zapiUserId>
    <zapiMethod>
        <methodName>zapiPing</methodName>
    </zapiMethod>
    </request>");
$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, $host);
curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, ($xmlRequest));

$response = curl_exec ($ch);
curl_close ($ch);

//parse out the XML here, and do stuff
//
echo ("$response");
exit;
```

Response:

```
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>ZAPI Ok</message>
    <methodResponse>
        <methodName>zapiPing</methodName>
        <methodErrorMessage>zapiPing</methodErrorMessage>
    </methodResponse>
</response>
```

2.3.3 API THROTTLE

To ensure continuous quality of service, API usage can be subjected to throttling. The throttle will be applied once an API client reaches a certain threshold.

Zauí Software reserves the right to throttle any and all API clients to ensure quality of service for all Zauí Software customers.

Those clients who do encounter a throttle threshold will get met with a HTTP 503 response code and the error XML node populated.

We encourage all API developers to anticipate this error and take appropriate measures like:

- using a cached value from a previous call, or
- passing on a message to the end user that is subjected to this behavior (if any)
- implement exponential back-off in your logic (see below)

2.3.4 IMPLEMENTING EXPONENTIAL BACK-OFF

Exponential back-off is the process of a client periodically retrying a failed request over an increasing amount of time. ***It is a standard error handling strategy for network applications.*** Besides being “required”, using exponential back-off increases the efficiency of bandwidth usage, reduces the number of requests required to get successful response, and maximizes the throughput of requests in concurrent environments.

The flow of implementing a simple exponential back-off is as follows:

1. Make a request to the API
2. Receive the response, check for error that ***has a retryable error code (such as 503)***
3. Wait ***1s + random_number_milliseconds*** seconds
4. Retry request
5. Receive the response, check for error that ***has a retryable error code (such as 503)***
6. Wait ***2s + random_number_milliseconds*** seconds
7. Retry request
8. Receive the response, check for error that ***has a retryable error code (such as 503)***
9. Wait ***4s + random_number_milliseconds*** seconds
10. Retry request
11. Receive the response, check for error that ***has a retryable error code (such as 503)***
12. Wait ***8s + random_number_milliseconds*** seconds
13. Retry request
14. Receive the response, check for error that ***has a retryable error code (such as 503)***

15. Wait `16s + random_number_milliseconds` seconds
16. Retry request
17. If you still get an error, stop and log the error

Note: `random_number_milliseconds` MUST be redefined after each “Wait”

In the above flow, `random_number_milliseconds` is a random number of milliseconds less than or equal to 1000. This is necessary to avoid certain lock errors in some concurrent implementations.

Note: the wait is always $(2^n) + \text{random_number_milliseconds}$, where n is a monotonically increasing integer initially defined as 0. N is incremented by 1 for each iteration (each request)

The algorithm is set to terminate when $n == 5$. This ceiling is in place to prevent clients from retrying infinitely, and results in a total delay of 32 seconds before a deemed “unrecoverable error”.

2.4 SUPPORTING XML NODES

Through many of our XML calls, we make use of common objects which eventually are represented in our XML responses that are returned.

We will outline some of those common nodes here, and where they are part of the response we will reference that.

2.4.1 SINGLE BOOKING DETAILS NODE

When retrieving details for a single activity node, using various request calls in the system, Zauí will return a described single booking activity node in the following way:

```
<bookingDetails>
  <bookingNumber></bookingNumber>
  <activities>
    <activity>
      <cartItemId></cartItemId> (optional)
      <packageId></packageId> (optional)
      <packageName></packageName> (optional)
      <activityId></activityId>
      <activityType></activityType>
      <activityName></activityName>
      <activityDate></activityDate>
      <activityTime></activityTime>
      <pickupLocationId></pickupLocationId>
      <pickupLocationName></pickupLocationName>
      <pickupLocationAddress></pickupLocationAddress>
      <dropOffLocationId></dropOffLocationId>
```

```
<dropOffLocationName></dropOffLocationName>
<dropOffAddress></dropOffAddress>
<notes></notes>
<bookingHistoryNote></bookingHistoryNote>
<seniors></seniors>
<adults></adults>
<students></students>
<children></children>
<infants></infants>
<passengersAsString></passengersAsString>
<totalCost></totalCost>
<pricingOptions/></pricingOptions>
<onStandby> </onStandby>
<onWaitList> </onWaitList>
<waitListId></waitListId>
</activity>
</activities>
<products/>
<product>
  <productId></productId>
  <cartItemId></cartItemId> (optional)
  <packageId></packageId> (optional)
  <packageName></packageName> (optional)
  <productName></productName>
  <quantity></quantity>
  <price></price>
  <total></total>
<product>
</products>
<transactions>
<transaction>
  <transactionId></transactionId>
  <transactionName></transactionName>
  <wholesaleCompanyId></wholesaleCompanyId>
  <wholesaleCompanyName></wholesaleCompanyName>
  <wholesalePrimaryAddr>
    <primAddr></primAddr>
    <primCity></primCity>
    <primProvince></primProvince>
    <primPostalCode></primPostalCode>
    <primCountry></primCountry>
  </wholesalePrimaryAddr>
  <wholesaleSecondaryAddr>
    <secAddr></secAddr>
    <secCity></secCity>
    <secProvince></secProvince>
    <secPostalCode></secPostalCode>
    <secCountry></secCountry>
  </wholesaleSecondaryAddr>
  <wholesaleCertNumber></wholesaleCertNumber>
  <wholesaleCompanyId></wholesaleCompanyId>
  <wholesaleCommissionName></wholesaleCommissionName>
  <wholesaleCommissionRate></wholesaleCommissionRate>
```

```

<wholesaleCommissionAmount></wholesaleCommissionAmount>
<transactionMethod></transactionMethod>
<transactionAmount></transactionAmount>
<transactionDate></transactionDate>
</transaction>
</transactions>
<outstandingBalance></outstandingBalance>
<bookingBalance>
  <hasOutstandingBalance></hasOutstandingBalance>
  <balance></balance>
  <balanceFormated></balanceFormated>
</bookingBalance>
<subTotal></subTotal>
<tax></tax>
<total></total>
</bookingDetails>

```

2.4.2 CURRENCIES NODE

Since users of the back office system in Zauí can configure currencies, based on the sale dates in the system, the API will return.

```

<currencies>
  <currency>
    <currencyName></currencyName>
    <currencySymbol></currencySymbol>
    <isoCode></isoCode>
    <paymentGatewayId></paymentGatewayId>
    <rateId></rateId>
    <rate></rate>
    <totalConvertedWithRate></totalConvertedWithRate>
  </currency>
</currencies>

```

2.4.3 THE CART NODE

The following XML represents the cart node structure, and will depend on the contents of the cart.

```

<cart>
  <cartId></cartId>
  <customers>
    <customer>
    </customer>
  </customers>
  <activities/>
  <products/>
  <transactions/>
  <remainingBalance>
    <balance></balance>
  </remainingBalance>
  <cartTotals>

```

```

<bookingNumber></bookingNumber>
<modifyBookingNumber></modifyBookingNumber>
<activityTotal>$0.00</activityTotal>
<activityTax>$0.00</activityTax>
<productTotal>$0.00</productTotal>
<productTax>$0.00</productTax>
<packageTotal>$0.00</packageTotal>
<activitySurplusFees>$0.00</activitySurplusFees>
<surchargeTotal>$0.00</surchargeTotal>
<perPersonTax>$0.00</perPersonTax>
<total>$0.00</total>
<tax>$0.00</tax>
<totalIncludingTax>$0.00</totalIncludingTax>
</cartTotals>
{CURRENCIES NODE}
</cart>
```

2.4.4 ACTIVITY TIMES NODE

The following XML represents a single activity time node, but depending on the type of activity which the request is for, you may have many of these returned. The <activityTimes> node is singular and the <activityTime> node can be many.

```

<activityTimes>
  <activityTime>
    <activityStartTime>
    <activityTimeCancelled>
    <activityTimeOnlineOnly>
    <activityTimeAvailable>
    <activityTimeAvailableMessage>
    <inventoryCheckCode>
    <inventoryCheckMessage>
    <activityTimeSpotsRemaining>
  </activityTime>
</activityTimes>
```

2.4.5 BUSINESS HOURS NODE

Basic result data for business hours when sent back in response structures:

```

<businessHours>
  <hours>
    <startDate></startDate>
    <openingTime></openingTime>
    <closingTime></closingTime>
    <endDate></endDate>
  </hours>
</businessHours>
```

2.4.6 PICKUP LOCATIONS NODE

Basic result data for pickup locations when sent back in response structures:

```
<pickupLocations>
    <locationName></locationName>
    <locationId></locationId>
</pickupLocations>
```

2.4.7 DROP OFF LOCATIONS NODE

Basic result data for dropoff locations when sent back in response structures:

```
<dropOffLocations>
    <locationName></locationName>
    <locationId></locationId>
</dropOffLocations>
```

2.4.8 RENTAL DATE/TIMES NODES

Basic result data for rentals when checking on inventory for the rental:

```
<rentalStartDateTimes>
    <rentalStartDate>
        <date>
        <rentable>
        <equipmentId>
    </rentalStartDate>
</rentalStartDateTimes>
```

2.4.9 ALLOWED GUEST TYPES

Basic result data describing the allowed guest types for any given activity or package in the system:

```
<allowedPassengers>
    <passengerType>
        <systemTypeId/>
        <systemTypeName/>
        <name/>
        <defaultValue/>
        <basePrice/>
    </passengerType>
</allowedPassengers>
```

2.4.10 WAIT LIST NODE

Basic node describing wait lists guests in the system:

```
<waitListGuests>
  <guest>
    <waitForListId></waitForListId>
    <firstName></firstName>
    <lastName></lastName>
    <mobilePhone></mobilePhone>
    <email></email>
    <comments></comments>
    <dateCreated></dateCreated>
    {PASSENGERS_NODE}
  </guest>
</waitListGuests>
```

2.4.11 PASSENGERS LIST NODE

Basic node describing passenger types in the system:

```
<passengers>
  <seniors></seniors>
  <adults></adults>
  <students></students>
  <children></children>
  <infants></infants>
  <passengersAsString></passengersAsString>
</passengers>
```

2.4.12 GUEST OR AGENT DETAILS NODE

Basic node describing a single guest in the system:

Please note the root node here will be either **<guestDetails> or <agentDetails>**

```
<rootNodeName>
  <rootNodeNameID></rootNodeNameID>
  <rootNodeName HashKey></rootNodeName HashKey>

  <accountId></accountId>
  <expiryDate></expiryDate>
  <firstName></firstName>
  <lastName></lastName>
  <username></username>
  <mobilePhone></mobilePhone>
  <email></email>
  <gender></gender>
  <addressline1\>
  <addressline2\>
```

```

<city\>
<state\>
<country\>
<zipCode\>
<dateCreated></dateCreated>
{USER_CUSTOM_FIELDS_NODE}
<relatedBookings>
<totalNumberOfBookings></totalNumberOfBookings>
<totalPurchases></totalPurchases>
<allBookings/>
</relatedBookings>
</guestDetails>

```

2.4.13 USER CUSTOM FIELDS NODE

These are fields that can be defined using the back office of Zauí, providing an easy mechanism to create and define your own custom user fields within the system. The format of these fields in XML is the following:

When GETTING user profiles, the user custom field data will be send back in the following format:

```

<userCustomFields>
  <customField>
    <customFieldId></customFieldId>
    <customFieldRequired> </customFieldRequired>
    <customFieldLabel></customFieldLabel>
    <customFieldValue></customFieldValue>
  </customField>
</ userCustomFields >

```

When UPDATING user profiles, the user custom data is expected in the following format:

```

<userCustomFields>
  <customField>
    <customFieldId></customFieldId>
    <customFieldValue> </customFieldValue>
  </customField>
  <customField>
    <customFieldId></customFieldId>
    <customFieldValue> </customFieldValue>
  </customField>
  <customField>
    <customFieldId></customFieldId>
    <customFieldValue> </customFieldValue>
  </customField>
</userCustomFields>

```

2.4.14 TRANSACTION DETAILS NODE

Refer to the trailing appendix in this document for those transaction which are available on the API.

Depending on the type of transaction being submitted, your request will be required to provide particular details for that transaction.

Basic transaction node:

```
<transactionDetails>
  <transactionId></transactionId>
  <transactionName></transactionName>
  <transactionMethod></transactionMethod>
  <transactionAmount></transactionAmount>
  <transactionDate></transactionDate>
</transactionDetails>
```

Basic credit card transaction example node:

```
<transactionDetails>
  <transactionType></transactionType>
  <transactionAmount></transactionAmount>
  <creditCardDetails>
    <nameOnCard></nameOnCard>
    <number></number>
    <expMonth></expMonth>
    <expYear></expYear>
    <csv></csv>
    <paymentGatewayId></paymentGatewayId>
    <currencyId></currencyId>
  </creditCardDetails>
</transactionDetails>
```

NOTE: For all invoice/voucher and discount transaction, you must provide a activity and/or product cart item id, indicating which item you wish to have the invoice/voucher or discount transaction applied to.

NOTE:

paymentGatewayId – optionally your call can include a different payment gateway ID, then was is used as the default in your system. Payment gateways are passed through on the zapiGetSystemInformation call

currencyId – passing this will indicate that you wish to transaction the payment for a specific currency. To enable this you must:

- Have currencies enabled in your system
- Must have a valid currency for the sale date

Valid currencies for your order are always passed back as part of the Cart Object.

When using this feature, in conjunction with the paymentGatewayId field, please note, if the currencyId you have passed already has configured a payment gateway, this will be used instead of the paymentGatewayId you have passed. If the currencyId doesn't have a payment gateway configured, then the paymentGatewayId passed will be used. If neither of these are configured, the default gateway will be used.

Basic voucher/invoice transaction example node:

```
<transactionDetails>
```

```

<transactionType></transactionType>
<transactionAmount></transactionAmount>
<details>
    <companyAgentId></companyAgentId> (optional)
    <companyId></companyId>
    <voucherNumber></voucherNumber>
</details>
<activities>
    <activity>
        <cartItemId></cartItemId>
    </activity>
</activities>
<products>
    <product>
        <cartItemId></cartItemId>
    </product>
<products>
</transactionDetails>

```

Basic discount transaction example node:

```

<transactionDetails>
    <transactionType></transactionType>
    <transactionAmount></transactionAmount>
    <details>
        <discountId></discountId>
        <activities>
            <cartItemId></cartItemId>
        </activities>
        <products>
            <cartItemId></cartItemId>
        <products>
    </details>
</transactionDetails>

```

2.4.15 DAYS OF WEEK AVAILABILITY NODE

The days of week XML node provide a description of configuration, by date ranges, of the days of the week which the activity is available.

```

<daysOfWeekAvailable>
    <dateRange>
        <startDate/>
        <daysOfWeek>
            <monday/>
            <tuesday/>
            <wednesday/>
            <thrusday/>
            <friday/>
            <saturday/>

```

```

<sunday/>
</daysOfWeek>
<endDate/>
</dateRange>
</daysOfWeekAvailable>
```

2.4.16 PRICE CODE NODE(S)

The following XML node describes a single price code as configured within the system.

```

<priceCodes>
  <priceCode>
    <priceCodeId/>
    <priceCodeTypeAsString/>
    <priceCodeName/>
    <priceCodeBands>
      <band>
        <minValue/>
        <maxValue/>
        <bandPricing>
          <passengerType>
            <systemTypeId/>
            <systemTypeName/>
            <name/>
            <defaultValue/>
            <basePrice/>
          </passengerType>
        </bandPricing>
      </band>
    </priceCodeBands>
  </priceCode>
</priceCodes>
```

2.4.17 PACKAGE MULTI DAY ITINERARY NODE

For some packages, namely where the packageType field is indicating that the requested package is a multi-day itinerary package, Zauí will send back to you the following XML structure which describes the configuration for each day.

This XML node will generally be present where the packageType field is available, and its set to indicating multi-day itinerary.

```

<multiDayItineraries>
  <multiDayItinerary>
    <startDate/>
    <endDate/>
    {PRICE CODE NODE}
    <days>
      <day>
        <activities>
```

```
<activity>
  <activityId/>
  <activityName/>
</activity>
</activities>
<products>
  <product>
    <productId/>
    <productName/>
  </product>
</products>
</day>
</days>
</multiDayItinerary>
</multiDayItineraries>
```

2.4.18 BOOKING NODE – SEARCH RESULTS

We have standardized the result set for each booking which is returned as the result from a search API call. An example would be, searching for a specific booking ID.

The XML node for a single booking returned from a search is the following markup:

```
<booking>
  <bookingNumber></bookingNumber>
  <guestId>
  <guestHashKey>
  <lastName></lastName>
  <firstName></firstName>
  <email></email>
  <mobilePhone></mobilePhone>
  <outstandingBalance></outstandingBalance>
  <bookingBalance>
    <hasOutstandingBalance></hasOutstandingBalance>
    <balance></balance>
    <balanceFormated></balanceFormated>
  </bookingBalance>
  <subTotal></subTotal>
  <tax></tax>
  <total></total>
  <reservationist></reservationist>
  <activities>
    <activity>
      <activityName></activityName>
      <activityId></activityId>
      <activityType></activityType>
      <packageId></packageId>
      <packageName></packageName>
      <activityDate></activityDate>
      <activityDateTime></activityDateTime>
      <pickupLocationId></pickupLocationId>
```

```
<pickupTime></pickupTime>
<pickupName></pickupName>
<pickupLocationAddress></pickupLocationAddress>
<dropOffLocationId></dropOffLocationId>
<dropoffTime></dropoffTime>
<dropoffName></dropoffName>
<dropOffLocationAddress></dropOffLocationAddress>
<commisionAgentId>
<commissionAgentName>
<commissionAgentCompany>
<rentalEndDate>
<rentalStartTime>
<rentalEndTime>
<rentalEndDate>
<notes></notes>
<passengers>
  <seniors></seniors>
  <adults></adults>
  <students></students>
  <children></children>
  <infants></infants>
  <passengersAsString></passengersAsString>
</passengers>
<passengersAsString></passengersAsString>
<totalCost></totalCost>
<pricingOptions/>
<checkedInStatus></checkedInStatus>
<onStandby></onStandby>
<onWaitList></onWaitList>
  <waitListId></waitListId>
</activity>
</activities>
<products/>
</booking>
```

2.4.19 MOBILE DATA NODE

As part of some of our API calls, you can pass along additional location service data, to be stored into the system for subsequent reporting.

```
<mobileData>
  <mobileDeviceData>
    <model></model>
    <name></name>
    <systemName></systemName>
    <systemVersion></systemVersion>
  </mobileDeviceData>
  <iosLocationServiceData>
    <altitude></altitude>
    <coordinate></coordinate>
    <course></course>
    <horizontalAccuracy></horizontalAccuracy>
    <verticalAccuracy></verticalAccuracy>
    <speed></speed>
    <timeStamp></timeStamp>
  </iosLocationServiceData>
</mobileData>
```

2.5 GENERAL / MISC API CALLS

2.5.1 ZAPIPING

Request Structure:

```
<?xml version="1.0"?>
<request>
  <zapiToken></zapiToken>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiMethod>
    <methodName>zapiPing</methodName>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0"?>
<response>
  <zapiVersion>2.0</zapiVersion>
  <error>0</error>
  <message>ZAPI Ok</message>
  <methodResponse>
    <methodName>zapiPing</methodName>
    <methodErrorMessage>Pong</methodErrorMessage>
  </methodResponse>
</response>
```

2.5.2 ZAPIGETSYSTEMINFORMATION

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>

<zapiMethod>
  <methodName>zapiGetSystemInformation</methodName>
</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
  <methodName>
  <methodErrorCode>
  <methodErrorMessage>
    <systemName></systemName>
    <companyName></companyName>
    <companyTelephoneNumber></companyTelephoneNumber>
    <systemLogo></systemLogo>
    <currentVersion></currentVersion>
```

{BUSINESS HOURS NODE}

```
<systemDateTime>
<systemTimeZoneAsString>
<currencySettings>
  <currencySignificantDigits>
  <currencySymbol>
</currencySettings>
<currencies>
  <currency>
    <currencyName> </currencyName>
    <currencySymbol></currencySymbol>
    <isoCode> </isoCode>
    <paymentGatewayId></paymentGatewayId>
    <rate></rate>
  </currency>
</currencies>
{PASSENGERS_NODE}
<airlines>
  <airline>
```

```
<airlineId>
<airlineName>
<airlineAbbreviation>
</airline>
</airlines>
</message>
</methodResponse>
</response>
```

2.5.3 ZAPIGENERATEBARCODEIMAGE

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>

<zapiMethod>
<methodName>zapiGenerateBarcodeImage</methodName>

<bookingNumber></bookingNumber>
<type></type>
<activityId></activityId>
<activityDate></activityDate>

<productId></productId>
<productDate></productDate>

</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
<methodName>zapiGenerateBarcodeImage</methodName>
```

```
<activity>
  <activityName></activityName>
  <activityId></activityId>
  <activityDate></activityDate>
  <categoryName></categoryName>
  <categoryId>1</categoryId>
  <barCodeImage></barCodeImage>
</activity>

<product>
  <productName></productName>
  <productId></productId>
  <productDate></productDate>
  <categoryName></categoryName>
  <categoryId>1</categoryId>
  <barCodeImage></barCodeImage>
</product>
</methodResponse>
</response>
```

2.6 INTERACTING WITH A CART SESSION API CALLS

2.6.1 ZAPICLEARCARTSESSION

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiClearCartSession</methodName>
    <cartId></cartId>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion></zapiVersion>
    <error></error>
    <message></message>
    <methodResponse>
        <methodName>zapiClearCartSession</methodName>
        <methodErrorCode></methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </response>
```

2.6.2 ZAPIREMOVECARTSESSION

Will remove the cart Id from your list of allowed sessions, and clear any order contents that you might have attached to this session Id.

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId>
    <zapiAccountId>
    <zapiToken>
    <zapiMethod>
        <methodName>zapiRemoveCartSession</methodName>
        <cartId></cartId>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion></zapiVersion>
    <error></error>
    <message></message>
    <methodResponse>
        <methodName/>
        <methodErrorCode/>
        <methodErrorMessage/>
    </response>
```

2.6.3 ZAPIGETALLCARTSESSIONS

Will return a list of all the valid cart sessions active.

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId>
    <zapiAccountId>
    <zapiToken>
    <zapiMethod>
        <methodName>zapiGetAllCartSessions</methodName>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion></zapiVersion>
    <error></error>
    <message></message>
    <methodResponse>
        <methodName/>
        <methodErrorCode/>
        <methodErrorMessage/>
    <sessions/>
    <session>
        <cartId>
        <timeStamp>
    </session>
</sessions>
</response>
```

2.6.4 ZAPIGETCARTCONTENTS

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiGetCartContents</methodName>
        <cartId></cartId>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion></zapiVersion>
    <error></error>
    <message></message>
    <methodResponse>
        <methodName>zapiGetCartContents</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
        <cart>
            <customer>
                <clientFirstName></clientFirstName>
                <clientLastName></clientLastName>
                <homePhone></homePhone>
                <homePrefix></homePrefix>
                <homeLdd></homeLdd>
                <mobilePhone></mobilePhone>
                <mobilePrefix></mobilePrefix>
                <mobileLdd></mobileLdd>
                <primAddress></primAddress>
                <primPostal></primPostal>
                <primProvince></primProvince>
                <secAddress></secAddress>
                <city></city>
                <email></email>
                <height></height>
                <weight></weight>
                <gender></gender>
                <birthDate></birthDate>
                <clientId></clientId>
                <about></about>
                <countryId></countryId>
                <loyaltyProgram></loyaltyProgram>
                <newsLetter></newsLetter>
            <customFields></customFields>
                </customer>
                <activities>
                    <activity>
                        <activityId></activityId>
                        <activityName></activityName>
                        <activityDate></activityDate>
                        <activityTime></activityTime>
                        <seniors></seniors>
                        <adults></adults>
                        <students></students>
                        <children></children>
                        <infants></infants>
                        <totalCost></totalCost>
                    </activity>
                </activities>
            </cart>
        </methodResponse>
    </response>
```

```
<pricingOptions>
  <pricingOption>
    <optionId></optionId>
    <optionName></optionName>
    <quantity></quantity>
  </pricingOption>
</pricingOptions>

</activity>
</activities>
<products>
  <product>
    <productId></productId>
    <productName></productName>
    <quantity></quantity>
    <price></price>
    <total></total>
  </product>
</products>
<cartTotals>
  <surchargeTotal></surchargeTotal>
  <packageTotal></packageTotal>
  <activityTotal></activityTotal>
  <activityTax></activityTax>
  <productTotal></productTotal>
  <productTax></productTax>
  <perPersonTax></perPersonTax>
  <total></total>
  <tax></tax>
  <giftCertificateAmountToBeApplied></giftCertificateAmountToBeApplied>
  <totalIncludingTax></totalIncludingTax>
</cartTotals>
</cart>
</methodResponse>
</response>
```

2.6.5 ZAPIUPDATENOTESTOCART

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiUpdateNotesToCart</methodName>
        <cartId></cartId>
        <notes></notes>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiUpdateNotesToCart</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage>Updated notes</methodErrorMessage>
    </methodResponse>
</response>
```

2.6.6 ZAPIUPDATECUSTOMERDETAILSTOCART

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiUpdateCustomerDetailsToCart</methodName>
        <cartId></cartId>
        <firstName></firstName>
        <lastName></lastName>
        <mobileNumber></mobileNumber>
        <email></email>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiUpdateCustomerDetailsToCart</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </methodResponse>
</response>
```

2.6.7 ZAPIADDPRODUCTTOCART

This API call will simply add the requested quantity to any already existing quantities that are in the cart for the same product.

Optional parameters for linking a product to an individual package or activity are also provided.

For percentage based products, the linkage to a package or activity will be required. This will also automatically determine the product price, when it's a percentage.

Request Structure:

```
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiAddProductToCart</methodName>
        <cartId></cartId>
        <productId></productId>
        <linkedPackageId></linkedPackageId>
        <linkedActivityId></linkedActivityId>
        <quantity>1</quantity>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiAddProductToCart</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
        <cart></cart>
    </methodResponse>
</response>
```

NOTES:

As part of successfully updating or adding an activity to the cart, the response will contain the updated cart contents. See the details in zapiGetCartContents for the XML details for this particular callback.

2.6.8 ZAPIREMOVEPRODUCTCARTITEM

This API callback will remove the product entirely from the cart, regardless of the quantity for that product. If the same product is part of a package that is also within the cart, these will remain unaffected.

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiAddRemoveProductCartItem</methodName>
        <cartId></cartId>
        <productId></productId>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiversion>2.0</zapiversion>
    <error>0</error>
    <message>OK</message>
    <methodresponse>
        <methodname>zapiRemoveProductCartItem</methodname>
        <methoderrorcode>0</methoderrorcode>
        <methoderrormessage></methoderrormessage>
    </methodresponse>
</response>
```

2.6.9 ZAPIADDACTIVITYTOCART

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId/>
    <zapiAccountId/>
    <zapiToken/>
    <zapiMethod>
        <methodName>zapiAddActivityToCart</methodName>
        <cartId/>
        <activityId/>
        <activityDate/>
        <activityEndDate/>
        <rentalStartTime/>
        <rentalEndTime/>
        <pricingOptions>
            <option>
                <optionId/>
                <quantity/>
            </option>
        </pricingOptions>
        {PASSENGERS_NODE}
        <activityTime/>
        <pickupLocationId/>
        <dropOffLocationId/></zapiMethod>
        <supplierConfirmationNumber>
    </request>
```

NOTES:

Based on the activity type, the “activityTime”, “pickupLocationId” and “dropOffLocationId” may be required fields for adding an activity to the cart. See the appendix on activity types. In general the following rules will apply:

- *Activity time required for activity type Id's 502, 503*
- *Pickup and drop off location Id's require for activity Id's 504*
- *For ALL rental activity types, the activityEndDate, rentalStartTime and rentalEndTime are required fields*

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.0</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiAddActivityToCart</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <cart></cart>
  </methodResponse>
</response>
```

NOTES:

As part of successfully updating or adding an activity to the cart, the response will contain the updated cart contents. See the details in `zapiGetCartContents` for the XML details for this particular callback.

2.6.10 ZAPI REMOVE ACTIVITY CART ITEM

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiRemoveActivityCartItem</methodName>
    <cartId></cartId>
    <activityId></activityId>
    <activityDate>YYYY-MM-DD</activityDate>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiRemoveActivityCartItem</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </methodResponse>
</response>
```

2.6.11 ZAPILOADINGBOOKINGINTOCART

Request Structure:

```
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiLoadBookingIntoCart</methodName>
        <bookingNumber></bookingNumber>
    </zapiMethod>
</request>
```

Response Structure:

```
<response>
    <zapiVersion>2.1</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiLoadBookingIntoCart</methodName>
        <cartId> </cartId>
        {CART_NODE}
        <currencies/>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </methodResponse>
</response>
```

2.6.12 ZAPIPROCESSSINGLETRANSACTION

*Note that this is for creating a new booking from the cart contents.
If you want to add a transaction to an existing booking loaded into your cart,
use zapiProcessSingleTransaction

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiProcessSingleTransaction</methodName>
    <cartId></cartId>
    <freeCartIdOnComplete>false</freeCartIdOnComplete>
    {TRANSACTION_DETAILS_NODE}
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiProcessSingleTransaction</methodName>
    <methodErrorCode></methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <transactionResult>
      <transactionErrorCode>0</transactionErrorCode>
      <bookingId></bookingId>
      <transactionId></transactionId>
      <transactionAmount></transactionAmount>
      <transactionMethod></transactionMethod>
      <bookingMessage></bookingMessage>
    </transactionResult>
    {CART_NODE}
  </methodResponse>
</response>
```

2.6.13 ZAPIPROCESSCARTWITHPAYMENT

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiProcessCartWithPayment</methodName>
    <cartId> </cartId>
    <paymentMethod>
      <paymentMethodType></paymentMethodType>
    </paymentMethod>
    <paymentGatewayId></paymentGatewayId>
    <creditCardDetails>
      <nameOnCard></nameOnCard>
      <number></number>
      <expMonth></expMonth>
      <expYear></expYear>
      <csv></csv>
      <cardType></cardType>
      <rawSwipeData></rawSwipeData>
      <track1Data></track1Data>
      <track2Data></track2Data>
    </creditCardDetails>
    {MOBILE_DATA_NODE}
    <voucherDetails>
      <companyName></companyName>
      <companyAgentId></companyAgentId> (optional)
      <companyId></companyId>
      <voucherNumber></voucherNumber>
    </voucherDetails>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiProcessCartWithPayment</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage> </methodErrorMessage>
    <bookingInformation>
      <bookingId></bookingId>
      <bookingMessage> </bookingMessage>
    </bookingInformation>
  </methodResponse>
</response>
```

2.6.14 ZAPICREATEUNIQUECARTSESSION

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiCreateUniqueCartSession</methodName>
</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiCreateUniqueCartSession</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>

    <cartId></cartId>
    <timeStamp></timeStamp>
  </methodResponse>
</response>
```

2.6.15 ZAPIPROMOTIONCODEAPPLY

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiPromotionCodeApply</methodName>

  <promoCodeName></promoCodeName>
  <cartId></cartId>
</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiPromotionCodeApply</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>

    <cartId></cartId>
    {CART NODE}
  </methodResponse>
</response>
```

2.6.16 ZAPIGIFTCERTIFICATEAPPLY

Request Structure:

```
<?xml version=\"1.0\" encoding=\"utf-8\"?>
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiGiftCertificateApply</methodName>
    <giftCertificate></giftCertificate>
    <cartId></cartId>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiPromotionCodeApply</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>

    {CART NODE}
  </methodResponse>
</response>
```

2.7 PACKAGE BASED API CALLS

2.7.1 ZAPIGETPACKAGECATEGORIES

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiToken></zapiToken>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiMethod>
        <methodName>zapiGetPackageCategories</methodName>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.1</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiGetPackageCategories</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage/>
        <categories>
            <category>
                <categoryName></categoryName>
                <categoryId></categoryId>
                <numberOfPackages></numberOfPackages>
            </category>
        </categories>
    </methodResponse>
</response>
```

2.7.2 ZAPIGETPACKAGESBYCATEGORYID

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
  <zapiToken></zapiToken>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiMethod>
    <methodName>zapiGetPackagesByCategoryId</methodName>
    <categoryId></categoryId>
  </zapiMethod>
</request>
```

Optional Parameters:

```
  <verboseMode>
```

Values: true/false

Description: Using this flag will curtail the amount of detail provided in the response.

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGetPackagesByCategoryId</methodName>
    <methodErrorCode></methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <categoryId></categoryId>
    <categoryName></categoryName>
    <packages>
      <package>
        <packageId></packageId>
        <categoryId></categoryId>
        <packageName></packageName>
        <description></description>
        <packageStartDate></packageStartDate>
        <packageEndDate></packageEndDate>
        {ALLOWED GUEST TYPES NODE}
        {ACTIVITIES NODES}
        {PRODUCTS NODES}
      </package>
    </packages>
  </methodResponse>
</response>
```

2.7.3 ZAPIGETPACKAGEDETAILSBYPACKAGEID

Request Structure:

```
<request>
<zapiToken></zapiToken>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiMethod>
    <methodName>zapiGetPackageDetailsByPackageld</methodName>
    <packageld></packageld>
    <packageDate></packageDate>
</zapiMethod>
</request>
```

Note: Package Date is an optional field, when provided must be in the format of YYYY-MM-DD, and is used to determine pricing for the package.

Optional Parameters:

<verboseMode>

Values: true/false

Description: Using this flag will curtail the amount of detail provided in the response.

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.1</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiGetPackageDetailsByPackageld</methodName>
        <package>
            <packageld></packageld>
            <packageType></packageType>
            {PACKAGE MULTI-DAY ITINERARY NODE}
            <allowsOpenDatedActivities></allowsOpenDatedActivities>
            <enforceOpenDatesInSequencialOrder></enforceOpenDatesInSequencialOrder>
            <categoryId></categoryId>
            <packageName></packageName>
            <description>
                <packageStartDate></packageStartDate>
                <packageEndDate></packageEndDate>
            {ALLOWED GUEST TYPES NODE}
            {ACTIVITIES NODE}
            {PRODUCTS NODE}
        </package>
    </methodResponse>
</response>
```

2.7.4 ZAPIADDPACKAGETOCART

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiToken></zapiToken>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiMethod>
        <methodName>zapiAddPackageToCart</methodName>
        <cartId></cartId>
        <packageId></packageId>
        <{PASSENGERS_NODE}>
            <activities>
                <activity>
                    <activityId></activityId>
                    <activityDate></activityDate>
                    <activityTime></activityTime>
                    <pickupLocationId></pickupLocationId>
                    <pickupAddress/>
                    <dropOffLocationId></dropOffLocationId>
                    <dropoffAddress/>
                    <pricingOptions/>
                </activity>
                <activity>
                    <activityId></activityId>
                    <activityDate></activityDate>
                    <activityTime></activityTime>
                    <pricingOptions/>
                </activity>
            </activities>
            <products/>
        </zapiMethod>
    </request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.1</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiAddPackageToCart</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </methodResponse>
```

```
{CART NODE}
</response>
```

2.7.5 ZAPIPACKAGE_DETERMINEMINIMUMDATESTOBOOK

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiPackage_DetermineMinimumDatesToBook</methodName>
    <packageId/>
    <startDate/>
    <passengers>
      <seniors>
      <adults>
      <students>
      <children>
      <infants>
    </passengers>
    <activityStartId>
      <activityEndId>
    </zapiMethod>
  </request>
```

Note: Passengers, activityStartId and activityEndId are optional.

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiPackage_DetermineMinimumDatesToBook</methodName>
    <packageId></packageId>
    <startDate></startDate>
    <passengers>
      <seniors/>
      <adults/>
      <students/>
      <children/>
      <infants/>
    </passengers>
    <activityStartId></activityStartId>
    <activityEndId></activityEndId>
    <isPackageBookable></isPackageBookable>
    <isBookableMinimumStartDate></isBookableMinimumStartDate>
```

```

<activities>
  <activity>
    <activityName></activityName>
    <activityId></activityId>
    <activityType></activityType>
    <activityTypeAsString></activityTypeAsString>
    <isBookable></isBookable>
    <activityDate></activityDate>
    <activityTimes>
      <activityTime>
        <activityTime></activityTime>
        <isBookable></isBookable>
        <isBookableMessage></isBookableMessage>
        <spotsRemaining></spotsRemaining>
        <numberOnStandby></numberOnStandby>
        <isStandbyAvailable></isStandbyAvailable>
      </activityTime>
    </activityTimes>
    <isAvailable></isAvailable>
    <isBookableDate></isBookableDate>
  </activity>
</activities>
<methodErrorCode>0</methodErrorCode>
<methodErrorMessage></methodErrorMessage>
</methodResponse>
</response>

```

2.7.6 ZAPIPACKAGE_SETACTIVITYCOMPONENTDATE

Request Structure:

```

<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiPackage_SetActivityComponentDate</methodName>
    <cartId> (OPTIONAL)
    <bookingId> (OPTIONAL)
    <addToStandBy> (OPTIONAL – default false)
    <packageId></packageId>
    <activityId></activityId>
    <activityDate></activityDate>
    <activityTime></activityTime> (OPTIONAL)
    <pickupId></pickupId> (OPTIONAL)
    <dropoffId></dropoffId> (OPTIONAL)
  </zapiMethod>
</request>

```

Notes:

- Either cart Id or booking Id are required
- Supplied the addToStandBy with check first at the activity is configured for standby, and that the activity is sold out, and spots are available for standby (auto promotion could apply)

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiPackage_SetActivityComponentDate</methodName>
    <packageId></packageId>
    <activityId></activityId>
    <activityDate></activityDate>
    {CART_NODE}
  </methodResponse>
</response>
```

2.7.7 ZAPIPACKAGE_RETRIVEPRICINGDETAILS

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiPackage_RetrievePricingDetails</methodName>
    <packageId></packageId>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>

  {PACKAGE MULTI DAY ITINERARY NODE}
</response>
```

2.8 ACTIVITY BASED API CALLS

2.8.1 ZAPIGETACTIVITYANDPRODUCTCATALOG

Request Structure:

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiGetActivityAndProductCatalog</methodName>
</zapiMethod>
</request>
```

Response Structure:

```
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
<methodName>zapiGetActivityAndProductCatalog</methodName>
<activities>
  <activity>
    <categoryId></categoryId>
    <categoryName></categoryName>
    <activityId></activityId>
    <activityName></activityName>
  </activity>
</activities>
<products>
  <product>
    <productId></productId>
    <productname></productname>
  </product>
</products>
</response>
```

2.8.2 ZAPIGETACTIVITYCATEGORIES

Request Structure:

```
<?xml version="1.0"?>
<request>
```

```

<zapiToken></zapiToken>
<zapiUserId></zapiUserId>
<zapiMethod>
    <methodName>zapiGetActivityCategories</methodName>
</zapiMethod>
</request>

```

Response Structure:

```

<?xml version="1.0"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>ZAPI Ok</message>
    <methodResponse>
        <methodName>zapiGetActivityCategories</methodName>
        <categories>
            <category>
                <categoryId></categoryId>
                <categoryName></categoryName>
                <numberOfActivities></numberOfActivities>
                <activities>
                    <activity>
                        <activityId></activityId>
                        <activityType></activityType>
                        <activityName></activityName>
                        <activityTimes>
                            <activityTime>
                                <activityStartTime></activityStartTime>
                                <activityTimeCancelled></activityTimeCancelled>
                            </activityTime>
                        </activityTimes>
                    </activity>
                </activities>
            </category>
        </categories>
    </methodResponse>
</response>

```

2.8.3 ZAPIGETACTIVITIESBYCATEGORYID**Request Structure:**

```

<?xml version="1.0"?>
<request>
    <zapiToken></zapiToken>
    <zapiUserId></zapiUserId>
    <zapiMethod>
        <methodName>zapiGetActivitiesByCategoryId</methodName>
        <categoryId></categoryId>
    </zapiMethod>
</request>

```

Optional Parameters:

<verboseMode>

Values: *true/false*

Description: Using this flag will curtail the amount of detail provided in the response.

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.0</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
    <methodName>zapiGetActivitiesByCategoryId</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <categoryId>
        <categoryName>
            <imageUrl>
                <description>
                    <activities>
                        <activity>
                            <activityId></activityId>
                            <activityName></activityName>
                            <hasBookings>
                                <childrenActivities>
                                    </activity>
                            </activities>
                        </methodResponse>
                    </response>
                
```

2.8.4 ZAPIGETALLACTIVITIESBYNAME

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiGetAllActivitiesByName</methodName>
        <activityNameToSearch></activityNameToSearch>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.0</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
    <methodName>zapiGetAllMerchandiseByName</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <activities>
        <activity>
            <activityId></activityId>
            <activityName></activityName>
            <activityTypeId></activityTypeId>
            <image></image>
            <duration></duration>
            <categoryId></categoryId>
            <categoryName></categoryName>
        </activity>
    </activities>
</methodResponse>
</response>
```

2.8.5 ZAPIGETALLACTIVITIESBYDATE

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
    <methodName>zapiGetAllActivitiesByDate</methodName>
    <activityDate>YYYY-MM-DD</activityDate>
</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiGetAllActivitiesByDate</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
        <activityCategories>
            <activityCategory>
                <categoryName></categoryName>
                <categoryId></categoryId>
                <activities>
                    <activity>
                        <activityName></activityName>
                        <activityId></activityId>
                    </activity>
                </activityCategory>
            </activityCategories>
        </methodResponse>
    </response>
```

2.8.6 ZAPIGETACTIVITYDETAILSBYACTIVITYID

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiGetActivityDetailsByActivityId</methodName>
        <activityId></activityId>
        <activityDate>YYYY-MM-DD</activityDate> (Optional)
    </zapiMethod>
</request>
```

Optional Parameters:

<verboseMode>

Values: true/false

Description: Using this flag will curtail the amount of detail provided in the response.

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.0</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
    <methodName>zapiGetActivityDetailsByActivityId</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <activity>
        <categoryName>Fall Sightseeing</categoryName>
        <categoryId></categoryId>
        <activityName></activityName>
        <activityId></activityId>
        <activityTypeId></activityTypeId>
        <activityTypeAsString></activityTypeAsString>
        <activityThreshold></activityThreshold>
        <activityMaxThreshold></activityMaxThreshold>
        <operatedByCompanyId></operatedByCompanyId>
        <operatedByCompanyName></operatedByCompanyName>
        <supplierConfirmationNumberSystemManaged>
        <supplierConfirmationNumberDisplayed>
        <description></description>
        <imageUrl></imageUrl>
        <isFreeSell></isFreeSell>
        <ticketExpiry></ticketExpiry>
        <ticketPrinterText></ticketPrinterText>
        {ACTIVITY TIMES NODE}
        {DAYS OF WEEK NODE}
        {PICKUP LOCATIONS NODE}
        {DROP OFF LOCATIONS NODE}
        {BUSINESS HOURS NODE}
    <pricingOptions>
        <option>
            <name></name>
            <optionId></optionId>
            <price></price>
        </option>
    </pricingOptions>
    <duration></duration>
    {ALLOWED GUEST TYPES NODE}
    </activity>
<sharedInventoryDetails>
    <isParent>true/false</isParent>
    <childrenActivities>
        <childActivity>
            <activity/> //This is an entire XML node for a single activity
        </childActivity>
    </childrenActivities>
</sharedInventoryDetails>
```

```
</methodResponse>
</response>
```

NOTE:

systemType – indicate the system mapping for the passenger type to Senior, Adult, Student, Child and Infant

*The API will return one of the following **activityTypeld** (see the table Activity Type Id's appendix):*

500, 501, 502, 503 or 504

For each of these activity type id's optional additional information will be returned in the XML response. These are outlined here:

- Each type will return the **activityTimes** XML element. ONLY 502 and 503 will have variable times. 500, 501, 504 will be single times*
- 504 will return additional **pickupLocations** and **dropOffLocations** XML elements*

*The API call to have an activity added to the cart **zapiAddActivityToCart** will then require these optional values to be passed as part of the request, depending on the **activityTypeld**.*

2.8.7 ZAPICHECKACTIVITYINVENTORYBYDATE

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiCheckActivityInventoryByDate</methodName>
        <activityId></activityId>
        <activityDate>YYYY-MM-DD</activityDate>
        <activityTime>HH:MM:SS</activityTime>
        <pickupLocationId> (required or shuttles)
        <dropoffLocationId> (required or shuttles)s
        <requestedPassengers>
            <seniors></seniors>
            <adults></adults>
            <students></students>
            <children></children>
            <infants></infants>
        </requestedPassengers>
```

```
<pricingOptions>
<option>
    <optionId></optionId>
    <quantity></quantity>
</option>
</pricingOptions>

</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion></zapiVersion>
    <error></error>
    <message></message>
    <methodResponse>
        <methodName>zapiCheckActivityInventoryByDate</methodName>
        <activityId></activityId>
        <activityDate></activityDate>
        <activityName></activityName>
        <activityType></activityType>
        <totalRequestedPassengers></totalRequestedPassengers>
        <activityTimes>
            <activityTime>
                <activityTimeStart></activityTimeStart>
                <activityTimeAvailable></activityTimeAvailable>
                <activityTimeAvailableMessage/>
                <inventoryCheckCode></inventoryCheckCode>
                <inventoryCheckMessage></inventoryCheckMessage>
                <activityTimeSpotsRemaining></activityTimeSpotsRemaining>
            </activityTime>
        </activityTimes>
    </methodResponse>
</response>
```

Notes:

Activity date, activity time and requested passengers are optional parameters.

Activity date – if left blank, the current date will be used

Activity Time – if left blank, the call will return details for each activity time configured for the activity/activity date. If you provide an activity time, then details for only the requested time slot will be returned

Requested Passengers – if left blank, this will be treated as 0. If you provide them, then inventories will be compared and filtered based on the requested passengers.

2.8.8 ZAPIPRICEQUOTE (RENTALS ONLY)

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiMethod>
    <methodName>zapiPriceQuote</methodName>
    <activityId></activityId>
    <activityDate></activityDate>
    <activityTime></activityTime>
    <activityEndDate></activityEndDate>
    <rentalStartTime></rentalStartTime>
    <rentalEndTime></rentalEndTime>
    <{PASSENGERS_NODE}>
    <pricingOptions>
      <option>
        <optionId/>
        <quantity/>
      </option>
    </pricingOptions>
    <pickupLocationId></pickupLocationId>
    <dropOffLocationId></dropOffLocationId>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiPriceQuote</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.8.9 ZAPIACTIVITY_GETCANCELLATIONS

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
```

```

<zapiMethod>
  <methodName>zapiActivity_GetCancellations</methodName>
  <activityId/>
  <cancellationStartDate/>
  <cancellationEndDate/>
</zapiMethod>
</request>

```

Response Structure:

```

<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiActivity_GetCancellations</methodName>
    <cancellations>
      <cancellation>
        <cancellationId></cancellationId>
        <activityId></activityId>
        <activityName></activityName>
        <cancellationStartDate></cancellationStartDate>
        <cancellationEndDate></cancellationEndDate>
        <canceltionStartTime></canceltionStartTime>
        <cancellationEndTime></cancellationEndTime>
        <daysOfWeek>
          <sunday></sunday>
          <monday></monday>
          <tuesday></tuesday>
          <wednesday></wednesday>
          <thursday></thursday>
          <friday></friday>
          <saturday></saturday>
        </daysOfWeek>
      </cancellation>
    </cancellations>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage><![CDATA[ ]]></methodErrorMessage>
  </methodResponse>
</response>

```

2.8.10 ZAPIACTIVITY_CREATECANCELLATION

Request Structure:

```

<request>
  <zapiUserId/>
  <zapiAccountId/>

```

```
<zapiToken/>
<zapiMethod>
  <methodName>zapiActivity_CreateCancellation</methodName>
  <activityId/>
  <cancellationStartDate/>
  <cancellationEndDate/>
  <cancellationStartTime/>
  <cancellationEndTime/>
  <daysOfWeek>
    <sunday></sunday>
    <monday></monday>
    <tuesday></tuesday>
    <wednesday></wednesday>
    <thursday></thursday>
    <friday></friday>
    <saturday></saturday>
  </daysOfWeek>
  <flags>
    <canceledOnlineOnly/>
    <cancelWithBookings/>
    <skipDaysWithBookings/>
  </flags>
</zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiActivity_CreateCancellation</methodName>
    <cancellations>
      <cancellation>
        <cancellationId></cancellationId>
        <activityId></activityId>
        <activityName></activityName>
        <cancellationStartDate></cancellationStartDate>
        <cancellationEndDate></cancellationEndDate>
        <cancellationStartTime></cancellationStartTime>
        <cancellationEndTime></cancellationEndTime>
        <daysOfWeek>
          <sunday></sunday>
          <monday></monday>
          <tuesday></tuesday>
          <wednesday></wednesday>
          <thursday></thursday>
          <friday></friday>
          <saturday></saturday>
        </daysOfWeek>
      </cancellation>
    </cancellations>
  </methodResponse>
</response>
```

```
</daysOfWeek>
</cancellation>
</cancellations>
<methodErrorCode>0</methodErrorCode>
<methodErrorMessage></methodErrorMessage>
</methodResponse>
</response>
```

2.8.11 ZAPIACTIVITY_REMOVECANCELLATION

Request Structure:

```
<request>
<zapiUserId/>
<zapiAccountId/>
<zapiToken/>
<zapiMethod>
  <methodName>zapiActivity_RemoveCancellation</methodName>
  <cancellationId/>
</zapiMethod>
</request>
```

Response Structure:

```
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
  <methodName>zapiActivity_RemoveCancellation</methodName>
  <methodErrorCode>0</methodErrorCode>
  <methodErrorMessage></methodErrorMessage>
</methodResponse>
</response>
```

2.9 MERCHANDISE / PRODUCT BASED API CALLS

2.9.1 ZAPIGETMERCANDISECATEGORIES

Request Structure:

```
<?xml version="1.0"?>
<request>
```

```

<zapiToken></zapiToken>
<zapiUserId></zapiUserId>
<zapiMethod>
    <methodName>zapiGetMerchandiseCategories</methodName>
</zapiMethod>
</request>

```

Response Structure:

```

<?xml version="1.0"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>ZAPI Ok</message>
    <methodResponse>
        <methodName>zapiGetMerchandiseCategories</methodName>
        <categories>
            <category>
                <categoryId></categoryId>
                <categoryName></categoryName>
                <categoryDescription></categoryDescription>
            </category>
        </categories>
    </methodResponse>
</response>

```

Element	M/O /C	Format	Length	Details
Response	M	a-z A-Z		Top level element
ZapiVersion	M	0-9	5	
Error	M	0-9	2	
Message	M	a-z A-Z 0-9	Variable	Error Message details
Method response	M			
MethodName	M	a-z A-Z 0-9	Variable	See documentation for method options
Items	M	Element	Variable	
Item		Element	Variable	
CategoryId	M	0-9		Unique ID for the category
CategoryName	M	a-z A-Z 0-9		Category name
CategoryDescription	M	a-z A-Z 0-9	Variable	Category Description

2.9.2 ZAPIGETMERCANDISEBYCATEGORYID

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiGetMerchandiseByCategoryId</methodName>
    <categoryId></categoryId>
  </zapiMethod>
</request>
```

Element	M/O/C	Format	Length	Details
Request	M	a-z A-Z	7	Top level element
ZapiToken	M	a-z A-Z 0-9	40	The generated ZAPI token
ZapiUserId	M	0-9	2	Supplied user ID.
ZapiMethod	M			All requests must have a method
MethodName	M	a-z A-Z 0-9	Variable	See documentation for method options
CategoryId	M	0-9		

Response Structure:

```
<?xml version="1.0"?>
<response>
  <zapiVersion>2.0</zapiVersion>
  <error>0</error>
  <message>ZAPI Ok</message>
  <methodResponse>
    <methodName>zapiGetMerchandiseByCategoryId</methodName>
    <products>
      <product>
        <productId></productId>
        <productName></productName>
        <productDescription></productDescription>
        <pricePreTax></pricePreTax>
        <priceWithTax></priceWithTax>
      </product>
    </products>
  </methodResponse>
</response>
```

Element	M/O Format /C	Format	Length	Details
Response	M	a-z A-Z		Top level element
ZapiVersion	M	0-9	5	
Error	M	0-9	2	
Message	M	a-z A-Z 0-9	Variable	Error Message details
Method response	M			
MethodName	M	a-z A-Z 0-9	Variable	See documentation for method options
Items	M	Element	Variable	
Item		Element	Variable	
ProductId	M	0-9		Unique ID for the product
ProductName	M	a-z A-Z 0-9		Product name
ProductDescription	M	a-z A-Z 0-9	Variable	Product Description
PricePreTax	M	0-9.0-9 0-9	Float	Price of product pre-tax (based on qty of 1)
PriceWithTax	M	0-9.0-9 0-9	Float	Price of product with taxes (based on qty of 1)

2.9.3 ZAPIGETALLMERCANDISEBYNAME

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiGetAllMerchandiseByName</methodName>
        <productNameToSearch></productNameToSearch>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.0</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
    <methodName>zapiGetAllMerchandiseByName</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <products>
        <product>
            <productId></productId>
            <image></image>
            <listPrice></listPrice>
            <categoryId></categoryId>
            <categoryName></categoryName>
        </product><product>
            <productId></productId>
            <image></image>
            <listPrice></listPrice>
            <categoryId></categoryId>
            <categoryName></categoryName>
        </product>
    </products>
</methodResponse>
</response>
```

2.9.4 ZAPIGETPRODUCTDETAILSBYPRODUCTID

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiGetProductDetailsByProductId</methodName>
        <productId></productId>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiGetProductDetailsByProductId</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
        <product>
            <categoryName></categoryName>
            <categoryId></categoryId>
            <productName></productName>
            <productId></productId>
            <description></description>
            <imageUrl></imageUrl>
            <inventory></inventory>
            <listPrice></listPrice>
        </product>
    </methodResponse>
</response>
```

2.9.5 ZAPICHECKPRODUCTINVENTORY

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiCheckProductInventory</methodName>
        <productId></productId>
        <quantity></quantity>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.0</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiCheckProductInventory</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <productId></productId>
    <productName></productName>
    <totalRequested></totalRequested>
    <inventoryRemaining></inventoryRemaining>
    <inventoryCheckCode></inventoryCheckCode>
    <inventoryCheckMessage></inventoryCheckMessage>
  </methodResponse>
</response>
```

2.9.6 ZAPISETPRODUCTQUANTITY

Request Structure:

```
<?xml version=\"1.0\" encoding=\"utf-8\"?>
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiSetProductQuantity</methodName>
    <cartId></cartId>
    <productId></productId>
    <quantity></quantity>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiSetProductQuantity</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.10 WAIT LIST API CALLS

2.10.1 ZAPIADDWAITLISTGUEST

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiAddWaitListGuest</methodName>
    <activityId></activityId>
    <activityDate></activityDate>
    <activityTime/>
    <firstName> </firstName>
    <lastName> </lastName>
    <mobilePhone></mobilePhone>
    <email> </email>
    <comments> </comments>
    <b>{PASSENGERS_NODE}</b>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiAddWaitListGuest</methodName>
    <waitListId></waitListId>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.10.2 ZAPIREMOVEWAITLISTGUEST

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
```

```
<methodName>zapiRemoveWaitListGuest</methodName>
<waitListId></waitListId>
<activityId></activityId>
<activityDate></activityDate>
<activityTime/>
</zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiRemoveWaitListGuest</methodName>
    <waitListId></waitListId>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.10.3 ZAPICONVERTWAITLISTGUESTTOCART

Request Structure:

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiConvertWaitListGuestToCart</methodName>
    <waitListId></waitListId>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiConvertWaitListGuestToCart</methodName>
    <waitListId></waitListId>
    <cartId></cartId>
    {CART_NODE}
```

```

<methodErrorCode>0</methodErrorCode>
  <methodErrorMessage></methodErrorMessage>
</methodResponse>
</response>

```

2.10.4 ZAPIGETALLWAITLISTGUESTSBYACTIVITYIDANDDATE

Request Structure:

```

<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiGetAllWaitListGuestsByActivityIdAndDate</methodName>
    <activityId></activityId>
    <activityDate></activityDate>
    <activityTime/>
  </zapiMethod>
</request>

```

Response Structure:

```

<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGetAllWaitListGuestsByActivityIdAndDate</methodName>
    <activityId></activityId>
    <activityDate></activityDate>
    <{WAITLIST_NODE}>
      <totalGuestsOnWaitList></totalGuestsOnWaitList>
      <methodErrorCode>0</methodErrorCode>
      <methodErrorMessage><![CDATA[ ]]></methodErrorMessage>
    </methodResponse>
</response>

```

2.11 AGENT SPECIFIC CALLS

2.11.1 ZAPIAGENTS_AUTHENTICATE

Please note that this routine simply authenticates that the agent is part of your system. The system will return a success or failure, along with both the agent ID and the agent hash key. These can then be used later to add items to the cart session of packages, activities and products.

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiAgents_Authenticate</methodName>
    <username></username>
    <password></password>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiAgents_Authenticate</methodName>
    {AGENT_DETAILS_NODE}
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.11.2 ZAPIAGENTLOGIN

Please note – this routine is intended to be used, when you wish to have the system create and issue a new API Token, API User ID and API Account ID. This routine authenticates first that the agent is an active agent in your system (based on the expiry date of their account).

This call will return the API token, API User ID and API Account ID, which can then be used on subsequent calls.

Response Structure:

```
<request>
  <zapiUsername></zapiUsername>
  <zapiPassword></zapiPassword>
  <zapiMethod>
    <methodName>zapiAgentLogin</methodName>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion/>
  <error/>
  <message/>
```

```
<methodResponse>
    <methodName>zapiMobileLogin</methodName>
    <methodErrorCode> </methodErrorCode>
    <methodErrorMessage> </methodErrorMessage >
    <zapiUsername> </zapiUsername>
    <zapiAccountId> </zapiAccountId>
    <zapiUserId> </zapiUserId>
    <firstName></firstName>
    <lastName></lastName>
    <zapiApiToken/>
    <cartId></cartId>
</methodResponse>
</response>
```

2.11.3 ZAPIAGENTS_GETPROFILE

Request Structure:

```
<request>
    <zapiToken></zapiToken>
    <zapiAccountId></zapiAccountId>
    <zapiUserId></zapiUserId>
    <zapiMethod>
        <methodName>zapiAgents_GetProfile</methodName>
        <agentID></agentID>
        <agentHashKey></agentHashKey>
    </zapiMethod>
</request>
```

Response Structure:

```
<response>
    <zapiVersion>2.1</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiAgents_GetProfile</methodName>
        {AGENTS_DETAIL_NODE}
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </methodResponse>
</response>
```

2.11.4 ZAPIAGENTS_UPDATEPROFILE

Please note that when updating a user account, the user ID will remain the same, but the hash key can change based on the data that has been requested to be updated.

2.11.5 ZAPIAGENTS_UPDATEUSERNAME

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiAgents_UpdateUsername</methodName>
    <agentID></agentID>
    <agentHashKey></agentHashKey>
    <username> </username>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiAgents_UpdateUsername</methodName>
    {AGENTS_DETAIL_NODE}
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.11.6 ZAPIAGENTS_UPDATEPASSWORD

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiAgents_UpdatePassword</methodName>
    <agentID></agentID>
    <agentHashKey></agentHashKey>
    <username> </username>
    <password> </ password>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiAgents_UpdatePassword </methodName>
    {AGENTS_DETAILS_NODE}
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.11.7 ZAPIAGENTS_GETALLCOMPANIES**Request Structure:**

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiAgents_GetAllCompanies</methodName>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiAgents_GetAllCompanies</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage/>
    <companies>
      <company>
        < companyName> </companyName>
        < companyId></companyId>
        < companyType> </companyType>
      </company>
    </companies>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage><![CDATA[ ]]></methodErrorMessage>
  </methodResponse>
</response>
```

2.12 GUEST SPECIFIC CALLS

2.12.1 ZAPIGUESTS_AUTHENTICATE

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiGuests_Authenticate</methodName>
    <username></username>
    <password></password>
    <bookingId></bookingId>
  </zapiMethod>
</request>
```

Response Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiGuests_Authenticate</methodName>
    {GUEST_DETAILS_NODE}
  </zapiMethod>
</request>
```

2.12.2 ZAPIGUESTS_GETPROFILE

Zauí treats a guest as unique based on 4 pieces of information: first name, last name, mobile number and email address.

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiGuests_GetProfile</methodName>
    <guestID></guestID>
    <guestHashKey></guestHashKey>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGuests_GetGuestID</methodName>
    <{GUEST_DETAILS_NODE}>
      <methodErrorCode>0</methodErrorCode>
      <methodErrorMessage></methodErrorMessage>
    </methodResponse>
  </response>
```

2.12.3 ZAPIGUESTS_LOOKUP

Zauí treats a guest as unique based on 4 pieces of information: first name, last name, mobile number and email address.

Request Structure:

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiGuests_Lookup</methodName>
    <firstName> </firstName>
    <lastName> </lastName>
    <mobilePhone></mobilePhone>
    <email> </email>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGuests_Lookup </methodName>
    <{GUEST_DETAILS_NODE}>
      <methodErrorCode>0</methodErrorCode>
      <methodErrorMessage></methodErrorMessage>
    </methodResponse>
  </response>
```

2.12.4 ZAPIGUESTS_UPDATEPROFILE

Please note that when updating a user account, the ID will remain the same, but the hash key can change based on the data that has been requested to be updated.

Request Structure:

*Note – all fields are optional with the following exceptions:

- guestID
- guestHashKey
- firstName
- lastName
- mobilePhone
- email

Gender – valid settings are ‘F’ or ‘M’ or blank.

Country – since this is a string value, we will attempt to match this against the user defined list of countries. If it’s not found a new entry will be created, and this new ID used.

```
<request>
  <zapiUserId/>
  <zapiAccountId/>
  <zapiToken/>
  <zapiMethod>
    <methodName>zapiGuests_UpdateProfile</methodName>
    <guestID/>
    <guestHashKey/>
    <firstName/>
    <lastName/>
    <mobilePhone/>
    <email/>
    <birthDate/>
    <gender/>
    <addressLine1/>
    <addressLine2/>
    <country/>
    <city/>
    <state/>
    <zipCode/>
    <userCustomFields>
      <userCustomField>
        <customFieldId/>
        <customFieldValue/>
      </userCustomField>
    </userCustomFields>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
    <zapiVersion>2.1</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiGuests_UpdateProfile</methodName>
        <guestDetails>
            <guestID/>
            <guestHashKey/>
            <accountId></accountId>
            <expiryDate></expiryDate>
            <username></username>
            <firstName></firstName>
            <lastName></lastName>
            <mobilePhone></mobilePhone>
            <email></email>
            <addressLine1></addressLine1>
            <addressLine2></addressLine2>
            <city></city>
            <state></state>
            <zipCode></zipCode>
            <birthDate></birthDate>
            <dateCreated></dateCreated>
        {CUSTOM FIELDS NODE}
        {RELATED BOOKINGS NODE}

        </guestDetails>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
    </methodResponse>
</response>
```

2.12.5 ZAPIGUESTS_UPDATEUSERNAME

Request Structure:

```
<request>
    <zapiToken></zapiToken>
    <zapiAccountId></zapiAccountId>
    <zapiUserId></zapiUserId>
    <zapiMethod>
        <methodName>zapiGuests_UpdateUsername</methodName>
        <guestID></guestID>
        <guestHashKey></guestHashKey>
        <username></username>
    </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGuests_UpdateUsername</methodName>
    {GUEST_DETAILS_NODE}
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.12.6 ZAPIGUESTS_UPDATEPASSWORD**Request Structure:**

```
<request>
  <zapiToken></zapiToken>
  <zapiAccountId></zapiAccountId>
  <zapiUserId></zapiUserId>
  <zapiMethod>
    <methodName>zapiGuests_UpdatePassword</methodName>
    <guestID></guestID>
    <guestHashKey></guestHashKey>
    <username></username>
    <password></password>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGuests_UpdatePassword</methodName>
    {GUESTDETAILS_NODE}
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.12.7 ZAPIGUESTS_UPDATEACCOUNTEXPIRYDATE**Request Structure:**

Response Structure:**2.12.8 ZAPIGUESTS_GETBOOKINGDETAILS****Request Structure:**

```
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiGuests_GetBookingDetails</methodName>
    <guestID></guestID>
    <guestHashKey></guestHashKey>
    <bookingNumber></bookingNumber>
  </zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGuests_GetBookingDetails</methodName>
    <{SINGLE BOOKING DETAILS NODE}>
    <methodErrorCode></methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.13 SEARCHING API CALLS**2.13.1 ZAPISearchBookingsByLastName****Request Structure:**

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiSearchBookingsByLastname</methodName>
        <lastName></lastName>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiSearchBookingsByLastname</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage></methodErrorMessage>
        <allBookings>
            {BOOKING SEARCH RESULT NODE}
        </allBookings>
    </methodResponse>
</response>
```

2.13.2 ZAPISearchBookingsByBookingNumber

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
    <zapiUserId></zapiUserId>
    <zapiAccountId></zapiAccountId>
    <zapiToken></zapiToken>
    <zapiMethod>
        <methodName>zapiSearchBookingsByBookingNumber</methodName>
        <bookingNumber></bookingNumber>
    </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.0</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
  <methodName>zapiSearchBookingsByBookingNumber</methodName>
  <methodErrorCode>0</methodErrorCode>
  <methodErrorMessage></methodErrorMessage>
{BOOKING SEARCH RESULT NODE}
</methodResponse>
</response>
```

2.13.3 ZAPISearchBookingsByMobileNumber

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>

    <methodName>zapiSearchBookingsByMobileNumber</methodName>
    <mobileNumber></mobileNumber>

  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
  <methodName>zapiSearchBookingsByMobileNumber</methodName>
  <methodErrorCode>0</methodErrorCode>
  <methodErrorMessage></methodErrorMessage>
  <allBookings>
{BOOKING SEARCH RESULT NODE}

  </allBookings>
</methodResponse>
</response>
```

2.14 MANIFEST API CALLS

2.14.1 ZAPIGETMANIFESTENTIREDAYBYDATE

Request Structure:

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiGetManifestEntireDayByDate</methodName>
  <cartId></cartId>
  <date>YYYY-MM-DD</date>
</zapiMethod>
</request>
```

Response Structure:

```
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
  <methodName>zapiGetManifestEntireDayByDate</methodName>
  <manifestDate>YYYY-MM-DD</manifestDate>
  <totalGuests></totalGuests>
  <totalGuestsCheckedIn></totalGuestsGcheckedIn>
  <activities>
    <activity>
      <activityId></activityId>
      <activityName></activityName>
      <availability></availability>
      <capacity></capacity>
      <allBookings>
        <booking>
          <bookingNumber></bookingNumber>
          <bookingAttributId></ bookingAttributId >
          <supplierConfirmationNumber></suppierConfirmationNumber>
          <hasWaiverSigned></ hasWaiverSigned >
          <activityTime></activityTime>
          <rentalStartTime></rentalStartTime>
          <rentalEndTime></rentalEndTime>
```

```
<notes></notes>
<clientId></clientId>
<guestFirstName></guestFirstName>
<guestLastName></guestLastName>
<mobile></mobile>
<email></email>
<outstandingBalance></outstandingBalance>
<checkedInStatus></checkedInStatus>
<pickupLocation></pickupLocation>
<pickupTime></pickupTime>
<dropoffLocation></dropoffLocation>
<passengersAsString></passengersAsString>
</booking>
</allBookings>
</activity>
</activities>
</methodResponse>
</response>
```

2.14.2 ZAPIGETMANIFESTINDIVIDUALBYACTIVITYIDANDDATE

Request Structure:

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
<methodName>zapiGetManifestIndividualByActivityIdAndDate</methodName>
<cartId></cartId>
<activityId></activityId>
<date> </date>
<activityTime></activityTime>

</zapiMethod>
</request>
```

NOTE: Activity time here is optional. If provided, then the API callback will return data matching the specific activity time slot you have requested.

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiGetManifestIndividualByActivityIdAndDate</methodName>
    <activity>
      <activityId></activityId>
      <activityName></activityName>
      <availability></availability>
      <capacity></capacity>
      <allBookings>
        <booking>
          <bookingNumber></bookingNumber>
          <attributeId></attributeId>
          <activityTime></activityTime>
          <rentalStartTime></rentalStartTime>
          <rentalEndTime></rentalEndTime>

          <guestFirstName></guestFirstName>
          <guestLastName></guestLastName>

          {PASSENGERSLIST_NODE}

        </booking>
      </allBookings>
      {WAITLIST_NODE}

    </activity>
  </methodResponse>
</response>
```

2.15 GENERAL BOOKING/RESERVATION BASED API CALLS

2.15.1 ZAPIEMAILBOOKINGITINERARY

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
  <zapiUserId></zapiUserId>
  <zapiAccountId></zapiAccountId>
  <zapiToken></zapiToken>
  <zapiMethod>
    <methodName>zapiEmailBookingItinerary</methodName>
    <bookingNumber></bookingNumber>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <zapiVersion>2.0</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiEmailBookingItinerary</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
  </methodResponse>
</response>
```

2.15.2 ZAPICHECKIN**Request Structure:**

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiCheckIn</methodName>
  <cartId></cartId>
  <bookingNumber></bookingNumber>
  <checkInType></checkInType>
  <activityId></activityId>
  <activityDate>YYYY-MM-DD</activityDate>
  <productId></productId>
  <productDate>YYYY-MM-DD</productDate>
  <toggleStatus>true/false</toggleStatus>
</zapiMethod>
</request>
```

Note: Check in Type is of either A or P – activity or product.

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiCheckIn</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage>Checked In</methodErrorMessage>
```

```
<bookingBalance>
  <hasOutstandingBalance/>
  <balance/>
  <balanceFormated/>
</bookingBalance>
</methodResponse>
</response>
```

2.15.3 ZAPISETWAIVERSIGNEDVALUE

Request Structure:

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiSetWaiverSignedValue</methodName>
  <bookingId></bookingId>
  <clientId></clientId>

  <activityId></activityId>
  <activityDate></activityDate>
  <activityTime></activityTime>
  <waiverValue>false</waiverValue>
</zapiMethod>
</request>
```

Response Structure:

```
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
  <methodName>zapiSetWaiverSignedValue</methodName>
  <methodErrorCode>0</methodErrorCode>
  <methodErrorMessage></methodErrorMessage>
</methodResponse>
</response>
```

2.15.4 ZAPIUPDATERESERVATIONACTIVITYCOMPONENT

Request Structure:

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
```

```
<zapiMethod>
  <methodName>zapiUpdateReservationActivityComponent</methodName>
  <bookingId></bookingId>
  <clientId></clientId>

  <bookingNumber/>
  <sourceAttributeId/>
  <destinationActivityId/>
  <destinationActivityDate/>
  <destinationActivityTime/>

</zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiUpdateReservationActivityComponent</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage></methodErrorMessage>
    <bookingBalance>
      <hasOutstandingBalance/>
      <balance/>
      <balanceFormated/>
    </bookingBalance>

  </methodResponse>
</response>
```

2.15.5 ZAPI BATCH AVAILABILITY

This callback will allow checking bulk or batch availability for a single activity, across many days. This callback does have the following restrictions:

- Dates must be greater than or equal to the current system date
- The end date must be greater than or equal to the start date
- If the start date is less than the current system date, it will be reset to the current date
- Proper validation of the date formats in YYYY-MM-DD formats
- The difference in start to end date can not exceed 31 days
- The end date can not exceed 31 days from the current date
- The activity must be available on the API

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
  <methodName>zapiBatchAvailability</methodName>
  <activityId></activityId>
  <activityStartDate></activityStartDate>
  <activityEndDate></activityEndDate>
</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion>2.1</zapiVersion>
<error>0</error>
<message>OK</message>
<methodResponse>
<methodName>zapiBatchAvailability</methodName>
<activities>
  <activity>
    <activityName>
    <activityId>
      <batchAvailability>
        <dates>
          <date>
            <activityDate>
            <availability>
              {ACTIVITY TIME NODE}
            </availability>
          </date>
        </dates>
      </batchAvailability>
    </activity>
  </activities>
```

2.15.6 ZAPIBOOKINGS_CANCELBOOKING

Note, that this API call will cancel the booking, even if there is an outstanding balance on the booking. If you attempting to balance the booking transactions, then you must submit the appropriate transactions on the booking to reconcile it.

Request Structure:

```
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
```

```
<methodName>zapiBookings_CancelBooking</methodName>
<bookingNumber></ bookingNumber >

</zapiMethod>
</request>
```

Response Structure:

```
<response>
  <zapiVersion>2.1</zapiVersion>
  <error>0</error>
  <message>OK</message>
  <methodResponse>
    <methodName>zapiBookings_CancelBooking</methodName>
    <methodErrorCode>0</methodErrorCode>
    <methodErrorMessage> </methodErrorMessage>
    < bookingNumber \>
    <bookingBalance>
      <hasOutstandingBalance/>
      <balance/>
      <balanceFormated/>
    </bookingBalance>
  </methodResponse>
</response>
```

2.16 ZAUI MOBILE SPECIFIC API CALLS

2.16.1 ZAPIMOBILELOGIN

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
  <zapiUsername/>
  <zapiPassword/>
  <zapiMethod>
    <methodName>zapiMobileLogin</methodName>
    <{MOBILE_DATA_NODE}>
  </zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
<zapiVersion></zapiVersion>
<error></error>
<message></message>
<methodResponse>
    <methodName>zapiMobileLogin</methodName>
    <zapiToken></zapiToken>
    <zapiAccountId></zapiAccountId>
    <zapiUserId></zapiUserId>
    <zapiAccountType></zapiAccountType>
    <cartId></cartId>
</methodResponse>
</response>
```

2.16.2 ZAPILOGOUT

Request Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<request>
<zapiUserId></zapiUserId>
<zapiAccountId></zapiAccountId>
<zapiToken></zapiToken>
<zapiMethod>
    <methodName>zapiLogout</methodName>
    <cartId></cartId>
</zapiMethod>
</request>
```

Response Structure:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <zapiVersion>2.0</zapiVersion>
    <error>0</error>
    <message>OK</message>
    <methodResponse>
        <methodName>zapiLogout</methodName>
        <methodErrorCode>0</methodErrorCode>
        <methodErrorMessage>Logout Successful</methodErrorMessage>
    </methodResponse>
</response>
```


3 APPENDIX A – PACKAGE TYPES

Where you will find indication of a package type, is typically on any specific call where a detailed description of the package will be returned as part of the XML. Examples of this would be:

- zapiGetPackageDetailsByPackageId
 - Data field: packageType

There will be other calls using the same field. When the packageType is a type 1, meaning a multi day itinerary styled package, an additional XML node for the packageMultiDayItinerary will be added.

Package Type ID	Notes/Comments
0	This type of package describes any packages
1	All packages that are configured as a multi-day itinerary based package. The addition packageMultiDayItinerary XML node will be provided.

4 APPENDIX B - ACTIVITY TYPE ID'S

Activity Type Id	Message	API Notes/Comments
500	Regular	Activity Date – single Activity Times – single time
501	Custom	Activity Date – single Activity Times – single time
502	Third-party	Activity Date – single Activity Times – 1-n (variable)
503	Interval	Activity Date – single Activity Times – 1-n (variable)
504	Shuttle	Activity Date – single Activity Times – 1-n (variable) ; Pickup Locations and Drop Off Locations 1-n (variable)
505	Rental	Activity Date – multi Start Time – single End Time - single

5 APPENDIX C - PRODUCT TYPE ID'S

Product Type ID	Notes/Comments
1	This type of product describes products with “flat” rate pricing
2	This type of product describes products with “percentage” based pricing, where the price of the product is generated based on a percentage of the overall total of a reservation.

6 APPENDIX D - ZAPI ERROR CODES

Error Code	Message	Details
0	ZAPI OK	No error detected
1	ZAPI_VALIDATION_ERROR	Check the Token, user ID and account ID
2	ZAPI_RESPONSE_LIMIT_EXCEEDED	The threshold on number of API requests per minute has been exceeded.
3	ZAPI_INVALID_XML	XML Document is not valid.

7 APPENDIX E - ZAPI PAYMENT METHOD TYPES

Payment Type Id	Payment Type
2000	Credit Card Sale
2002	Cash Sale
2003	Override pay later
2004	Invoice/Voucher transaction – addition transaction details will be required
2005	Travellers Cheque
2006	Cheque
2016	User Defined
2018	Debit
2020	User Defined
2023	Deposit as commission
2027	User Defined
2029	User Defined
2031	User Defined

2033	User Defined
------	--------------

8 APPENDIX F - ZAPI METHOD ERROR CODES

Method Error Code	Message	Details
20	ZAPI_FILTER_ERROR	Invalid data type being passed as parameter for ZAPI method call.
40	ZAPI_MERCHANDISE_ERROR_SOLDOUT	Indicates that the requested merchandise qty level is not available.
41	ZAPI_MERCHANDISE_ERROR_NOINVENTORY	Not enough inventory for request
42	ZAPI_MERCHANDISE_ERROR_NOTPUBLISHED	Not available for online/remote purchase
43	ZAPI_MERCHANDISE_ERROR_NODATA	No data for request to return.
50	ZAPI_ACTIVITY_ERROR_SOLDOUT	Activity is not available for the requested amount
51	ZAPI_ACTIVITY_ERROR_DATE	Wrong date format or invalid date
52	ZAPI_ACTIVITY_ERROR_NOTPUBLISHED	Not available for online/remote purchase
53	ZAPI_ACTIVITY_ERROR_NODATA	No data for request to return.
70	ZAPI_RESERVATIONS_ERROR_INVALID_USER	User doesn't have permission for requested method action
71	ZAPI_RESERVATIONS_ERROR_INACTIVE	Reservation has already been canceled and reversed.
99	ZAPI_GENERAL_ERROR	General error code reserved for aggregation of non-specific errors. Also the number of the best hockey player in the world.
100	ZAPI_TRANSACTION_TYPE_NOTVALID	Transaction type not valid.