
rts-solutions.com

# Open Interface API Specification
# Version 1.20

November 2019

# Table of Contents


Version Change History ..... 3
Introduction ..... 5
Communication ..... 5
RTS Test Server Information ..... 6
Performance Schedule ..... 7
Selling - Fees and Adjustments ..... 10
Gift Card / Loyalty Card Information ..... 11
Gift Certificate Purchasing ..... 15
Loyalty Card Information ..... 17
Ticket Purchasing – Check Sold Out ..... 18
Ticket Purchasing – Check Redeem ..... 19
Ticket Purchasing – Verify Transaction ..... 20
Ticket Purchasing – Transaction Details ..... 21
Ticket Purchasing – Redeem ..... 24
Ticket Purchasing – Refund ..... 26
Ticket Purchasing – Reverse Transaction ..... 30
Ticket Purchasing (E2E) Overview ..... 30
Ticketing Purchasing (E2E) ..... 32
Ticket Purchasing (3rd Party Credit Card Processing) ..... 36
Ticket Purchasing – General Admission ..... 37
Seating Charts – Get Seat Layouts ..... 43
Seating Charts – Get Seat Plan For Performance ..... 46
Reserved Seating – Check Seat Picks ..... 51
Reserved Seating – Holding and Releasing Seats ..... 54
Ticket Purchasing – Reserved Seating ..... 56
Concession Prices and Sales ..... 60
Error Codes ..... 62
Reserved Seating Codes ..... 64
Film and Show Bit Field Values ..... 65

# Version Change History

| Date | Change Notes | Version |
|---|---|---|
| 12/28/08 | Added Error Listing | 0.2 |
| 06/16/09 | Added Retrieve Seating Chart Section | 1.0 |
| 06/17/09 | Added PHP sample code | 1.01 |
| 03/15/10 | Added the CustomerInfo and EmailAddress tags. Used to look-up transactions at the POS. | 1.02 |
| 04/13/10 | Added Gift Balance functions (requires server version 7.0.7564+).<br>Added gift number to purchase request, which allows adding money to an existing gift card (requires server version 7.0.7564+). | 1.03 |
| 11/22/10 | General document clean-up.<br>Corrected 'GiftInfo' to be 'GiftInformation'<br>Added information for concession prices XML and concession sales<br>Added 'SalesTaxCheck' information<br>Moved all references regarding encryption to a summary pages at back of documentation<br>Updated PHP sample code to use TLS for the connection protocol | 1.04 |
| 11/30/10 | Added test server information, credit card information, gift card information, etc<br>Updated PHP sample code to contain additional sample packets | 1.05 |
| 12/13/10 | Added sample of purchase ticket response<br>Updated sample response for GiftInformation, showing Registered Card information | 1.06 |
| 06/11/13 | Added multiple <Payment> sample<br>Added GroupID for Reserved Seating – Seat Chart<br>Added Registering a mag card<br>Added earning loyalty points for purchases<br>Added CardCredits to GiftInformation sample response | 1.07 |
| 01/07/2014 | Corrected an error in the Fees section regarding <TicketFee> | 1.08 |
| 01/15/2015 | Removed sample code (too many people using for production environments)<br>Switched from DIGEST to BASIC authentication<br>Added new error codes<br>Removed HTTP + Encryption section (no longer relevant)<br>Added CheckSoldOut request – shows Sold /Remaining / SoldOut levels for a performance | 1.09 |
| 03/20/2015 | Removed obsolete Seating Chart sample request | 1.10 |
| 06/03/2015 | Added HostedCheckout sample requests and responses<br>Added 3rd party credit card processing sample request<br>Added listing of bit field settings used for films and shows<br>Added new error numbers / messages<br>Updated existing error numbers / messages | 1.11 |

| Date | Change Notes | Version |
|---|---|---|
| 10/24/2016 | Added CheckRedeem request sample<br>Added GetSeatLayouts request sample<br>Added GetSeatPlanForPerf request sample<br>Added CheckSeatPicks request sample<br>Added VerifyTransaction request sample<br>Added ReverseTransaction request sample<br>Added Refund request sample<br>Updated BUY request to include <ThirdPartyID> node information<br>Simplified table of contents | 1.12 |
| 02/08/2017 | Added Bit Field values for various new amenities | 1.13 |
| 5/30/2017 | Fixed documentation error with the CHECKSEATPICKS response | 1.14 |
| 10/19/2017 | Fixed typo in 'CHECKSEATPICKS' sample request<br>Added GridRef to 'GETSEATPLANFORPERF' response | 1.15 |
| 02/02/2018 | Added note regarding GZIP compression now being required for schedule request | 1.16 |
| 07/24/2018 | Updated test credit card information<br>Added Bit Field values for various new amenities | 1.17 |
| 08/13/2018 | Added TransactionDetails request sample<br>Added Redeem request sample<br>Updated Refund request to include PartialRefund support | 1.18 |
| 09/26/2019 | Added <SOGen> node to schedule data to indicate SoldOutGeneral state<br>Updated verbiage on <SO> and <LI> node descriptions<br>Added gift card PIN node to BUY request<br>Added 3rd party gift card support<br>Added custom seat pricing data to GETSEATPLANFORPERF call<br>Added note regarding send confirmation emails on purchase via API<br>Updated existing error numbers / messages | 1.19 |
| 11/25/2019 | Updated seat type nodes to support new types<br>Added support for 3rd party gift cards into the GIFTINFORMATON call<br>Added a call for GIFTINFORMATIONWITHPIN to support PIN requiring card types | 1.20 |

# Introduction

RTS Ticketing has created an open interface to accommodate third party access to our sales engine. In order to allow access, we have created an HTTPS server that accepts XML packets.

Credit card payments are processed either through the POS locations' credit card processing account, or through your own processing account. So a separate shopping cart or credit card processing account is not needed, but you have the option of building your own.

Purchased tickets can be redeemed at the locations POS at for actual tickets. These tickets can be redeemed using any of three methods:

1. The credit card that was used to purchase the tickets
2. The receipt number assigned to the purchase
3. The bar code that is provided by the purchase response packet.

**Note: Some locations do not have bar code readers or the bar code may be illegible when printed on a low-quality printer. It is strongly suggested that the confirmation number is also printed with the bar code.**

# Communication

Communications with the locations POS server are done using a HTTPS Post of an XML packet. **Basic** authentication is performed for each Post.

The URL for posting data is:
https://<Theatre RTN Number>.formovietickets.com:2235/Data.ASP

**Note: The listening port is 2235 (not the default web server port 80)**

# RTS Test Server Information

It is a good idea to develop your application against a test server, and not against a live theatre server. Below is the information for the RTS Test Server:

The URL for posting data to is:
https://5.formovietickets.com:2235/Data.ASP

The credentials for this connection are:
Schedule Username: test
Schedule Password: test
Selling Username: test
Selling Password: test

Notes:
There are 2 test servers available: Non-E2E on port 2235, and E2E on port 2245.

The test credit card information FOR NON-E2E accounts is:
Number: 5499990123456781
Expiration: Any future month and year (mmyy format)
AvsStreet: 150 Mercury Village
AvsPostal: 81301
CVV: 123
NameOnCard: Any name

The test credit card information E2E accounts is:
Number: 4003000123456781
Expiration: Any future month and year (mmyy format)
AvsStreet: 150 Mercury Village
AvsPostal: 81301
CVV: 123
NameOnCard: Any name

The <ChargeAmount> for the test credit card cannot exceed $10.

To obtain a test gift card number, purchase a new gift card, and then use the returned number as your future test gift card number. See page 10 for details on purchasing new gift cards.

# Performance Schedule

**Performance Schedule - Sample Request Packet:**

**!! NOTE: GZIP Compression is required when requesting schedule data !!**

**!! NOTE: Sending in <ShowAvalTickets> and <ShowSales> values that do not match the settings on the locations configured schedule API account will result in a 401 – Unauthorized error !!**

```
<Request>
	<Version>1</Version>
	<Command>ShowTimeXml</Command>
	<ShowAvalTickets>1</ShowAvalTickets>
	<ShowSales>1</ShowSales>
	<ShowSaleLinks>1</ShowSaleLinks>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | None | Char | Yes | 1 |
| Command | None | Char | Yes | ShowTimeXml |
| ShowAvalTickets | None | Integer | No | Not 1 = Do not return available tickets (default)<br>1 = Return available tickets |
| ShowSales | None | Integer | No | Not 1 = Do not return sales (default)<br>1 = Return sales |
| ShowSaleLinks | None | Integer | No | Not 1 = Do not return sale links (default)<br>1 = Return sale links |

**Performance Schedule - Sample Response Packet:**

```
<Response>
      <ShowSchedule>
            <FileVersion>1.1</FileVersion>
            <RtsVersion>7.0.7238.0</RtsVersion>
            <LinkPrefix>
      https://www.readyticket.net/webticket/webticket2.aspWCI=BuyTicket&WCI=
            </LinkPrefix>
            <Tickets>
                  <Ticket>
                        <Code>1</Code>
                        <Name>Adult</Name>
                        <Price>7.5</Price>
                        <Tax>.5</Tax>
                  </Ticket>
                  <Ticket>
                        <Code>2</Code>
                        <Name>Child</Name>
                        <Price>7</Price>
                        <Tax>.43</Tax>
                  </Ticket>
            </Tickets>
            <Films>
                  <Film>
                        <Title>+21 STRANGE TITLE</Title>
                        <TitleShort>+21 STRANGE TITLE</TitleShort>
                        <Length>120</Length>
                        <Rating>NC17</Rating>
                        <WebSite>http://www.rts-solutions.com</WebSite>
                        <FilmCode>+24230</FilmCode>
                        <MtFilmCode></MtFilmCode>
                        <Shows>
                              <Show>
                                    <DT>200811261235</DT>
                                    <Aud>1</Aud>
                                    <ID>+24230|200811261235|1</ID>

      <SaleLink>%2B21+STRANGE+TITLE,112620081235,1,5,NC17</SaleLink>
                                    <RE>225</RE>
                                          <Sold>22</Sold>
                                    <SO>0</SO>
                                    <LI>0</LI>
                                    <TIs>
                                          <TI>
                                                <C>1|1</C>
                                          </TI>
                                          <TI>
                                                <C>1|2</C>
                                          </TI>
                                    </TIs>
                              </Show>
                        </Shows>
                  </Film>
            </Films>
      </ShowSchedule>
</Response>
```

| Node | Child Node | Type | Required | Description <ShowSchedule> |
|---|---|---|---|---|
| FileVersion | | String | Yes | 1.1 |
| RtsVersion | | String | Yes | Version of the POS |
| LinkPrefix | | String | Yes | Used to construct a direct ticketing link for a performance |
| Tickets | Ticket | Sub Node | Maybe | Lists available tickets |
| Films | Film | Sub Node | Yes | Lists films scheduled |

| Node | Child Node | Type | Required | Description <Ticket> |
|---|---|---|---|---|
| Code | | String | Yes | Ticket code used in purchase transactions |
| Name | | String | Yes | Ticket name as it should be displayed on the Internet |
| Price | | Decimal | Yes | Total ticket price |
| Tax | | Decimal | Maybe | Any tax included in the ticket price |

| Node | Child Node | Type | Required | Description <Film> |
|---|---|---|---|---|
| Title | | String | Yes | Film title name |
| Length | | Integer | Yes | Film length (minutes) |
| WebSite | | String | Maybe | URL associated with the film in the POS |
| FilmCode | | String | Maybe | RTS Film Code setup in the POS |
| MtFilmCode | | String | Maybe | Movietickets.com Film Code setup in the POS |
| Shows | Show | Sub Node | Yes | List of available shows for a film |

| Node | Child Node | Type | Required | Description <Show> |
|---|---|---|---|---|
| DT | | String | Yes | Showing time (yyyyMMddhhmm) |
| Aud | | String | Yes | Auditorium name |
| ID | | String | Yes | Performance ID, used in purchase transactions |
| Link | | String | Maybe | URL to purchase tickets page. The <LinkPrefix> is inserted before this tag when creating the complete URL |
| RE | | Integer | Maybe | Number of seats remaining in the auditorium |
| Sold | | Integer | Maybe | Number of seats sold in the auditorium |
| SO | | Integer | Maybe | 1 if show is sold out (internet) state |
| SOGen | | Integer | Maybe | 1 if show is sold out (general) state |
| LI | | Integer | Maybe | 1 if show is in low seats state |
| TIs | TI | Sub Node | Maybe | List of tickets available for this show |

# Selling - Fees and Adjustments

The Open Interface API currently allows for 2 different types of Fees, and an Adjustment.

The <TicketFee> item is a **single charge** added to the transaction.

The <TransactionFee> item is a **single charge** added to the transaction.

The <Adjust> item is an adjustment used to modify the charge amount. If you sell 3 tickets with -$1 in this item, it will **deduct** $1 from your transaction amount. This amount can be positive or negative, depending on how you wish to adjust the transaction.

All three of these items are placed inside the <Fees> tags inside the purchase packet, see page 13 for a sample packet. These items can be used in any purchase packet: tickets, items, gift cards, etc.

# Gift Card / Loyalty Card Information

**Gift Card / Loyalty Card - Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>GiftInformation</Command>
	<Data>
		<Packet>
			<GiftCards>
				<GiftCard>1234123412341234</GiftCard>
			</GiftCards>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | GiftInformation |
| Data | Packet | Sub Node | Yes | |
| Packet | GiftCards | Sub Node | Yes | |
| GiftCards | GiftCard | Sub Node | Yes | |
| GiftCard | | String | Yes | Gift card number |

**Gift Card / Loyalty Card - Sample Response Packet – Non-Registered Card:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<GiftCard>
			<GiftNumber>1234123412341234</GiftNumber>
			<DebitRemain>1061.16</DebitRemain>
			<Registered>0</Registered>
		</GiftCard>
	</Packet>
</Response>
```

**Gift Card / Loyalty Card (With PIN) - Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>GiftInformationWithPIN</Command>
	<Data>
		<Packet>
			<GiftCards>
				<GiftCard>
					<CardNumber>1234123412341234</CardNumber>
					<CardPIN>1234</CardPIN>
				</GiftCard>
			</GiftCards>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | GiftInformation |
| Data | Packet | Sub Node | Yes | |
| Packet | GiftCards | Sub Node | Yes | |
| GiftCards | GiftCard | Sub Node | Yes | |
| GiftCard | | Sub Node | Yes | One <GiftCard> node for each card in request |
| CardNumber | | String | Yes | Gift card number |
| CardPIN | | String | Yes | Gift card PIN |

**Gift Card / Loyalty Card (With PIN) - Sample Response Packet – Non-Registered Card:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<GiftCard>
			<GiftNumber>1234123412341234</GiftNumber>
			<DebitRemain>1061.16</DebitRemain>
			<Registered>0</Registered>
		</GiftCard>
	</Packet>
</Response>
```

**Gift Card / Loyalty Card - Sample Response Packet – Registered Card:**

Note: RTS mag cards are the only type that will return registration data!

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<GiftCard>
			<GiftNumber>2012700000745808</GiftNumber>
			<DebitRemain>1061.16</DebitRemain>
			<Registered>1</Registered>
			<RegisteredInfo>
				<FirstName>JOHN</FirstName>
				<LastName>DOE</LastName>
				<Address1>4 DOE ROAD</Address1>
				<Address2/>
				<City>DOEVILLE</City>
				<State>DO</State>
				<Postal>11111</Postal>
			</RegisteredInfo>
			<CardCredits>
				<TicketCredit>
					<Expiration>None</Expiration>
					<StartDate>None</StartDate>
					<Amount>1</Amount>
					<TitleRestriction/>
					<TicketRestriction/>
				</TicketCredit>
				<TicketCredit>
					<Expiration>20130405000000</Expiration>
					<StartDate>20130405000000</StartDate>
					<Amount>1</Amount>
					<TitleRestriction>WORLD WAR Z</TitleRestriction>
					<TicketRestriction/>
				</TicketCredit>
				<ItemCredit>
					<Expiration>None</Expiration>
					<StartDate>20130426000000</StartDate>
					<Amount>1</Amount>
					<Item>SM POPCORN</Item>
				</ItemCredit>
			</CardCredits>
		</GiftCard>
	</Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Code | | Integer | Yes | -1 |
| Packet | GiftCard | Sub Node | Yes | |
| GiftCard | RegisteredInfo | Sub Node | Yes | |
| GiftCard | CardCredits | Sub Node | Yes | |
| GiftNumber | | String | Yes | Gift card number |
| DebitRemain | | Decimal | Yes | Gift card remaining balance |
| Registered | | Integer | Yes | 0 = not registered, 1 = registered |

| Node | Child Node | Type | Required | Description <RegisteredInfo> |
|---|---|---|---|---|
| FirstName | | String | Yes | Customer first name |
| LastName | | String | Yes | Customer last name |
| Address1 | | String | Yes | Customer street address 1 |
| Address2 | | String | Yes | Customer street address 2 |
| City | | String | Yes | Customer city |
| State | | String | Yes | Customer state |
| Postal | | String | Yes | Customer postal code |

| Node | Child Node | Type | Required | Description <CardCredits> |
|---|---|---|---|---|
| TicketCredit | | Sub Node | No | |
| ItemCredit | | Sub Node | No | |

| Node | Child Node | Type | Required | Description <TicketCredit> |
|---|---|---|---|---|
| Expiration | | String | Yes | None = No expiration date, or date of expiration (yyyyMMddhhmmss) |
| StartDate | | String | Yes | None = No start date, or date ticket credit becomes valid (yyyyMMddhhmmss) |
| Amount | | Integer | Yes | Amount of tickets available |
| TitleRestriction | | String | Yes | Empty node = no restriction, or title of movie this credit is good for |
| TicketRestriction | | String | Yes | Empty node = no restriction, or name of ticket this credit is good for |

| Node | Child Node | Type | Required | Description <ItemCredit> |
|---|---|---|---|---|
| Expiration | | String | Yes | None = No expiration date, or date of expiration (yyyyMMddhhmmss) |
| StartDate | | String | Yes | None = No start date, or date item credit becomes valid (yyyyMMddhhmmss) |
| Amount | | Integer | Yes | Amount of item available |
| Item | | String | Yes | Name of item credit is good for |

# Gift Certificate Purchasing

**New Gift Card Purchase - Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseGifts>
				<PurchaseGift>
					<Amount>25</Amount>
				</PurchaseGift>
			</PurchaseGifts>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<Number>5499990123456781</Number>
					<Expiration>0513</Expiration>
					<AvsStreet>4 Main St</AvsStreet>
					<AvsPostal>30329</AvsPostal>
					<CID>123</CID>
					<NameOnCard>John Doe</NameOnCard>
					<ChargeAmount>7.5</ChargeAmount>
				</Payment>
			</Payments>
		</Packet>
	</Data>
</Request>
```

**Add Money to Existing Gift Card Purchase - Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseGifts>
				<PurchaseGift>
					<GiftCard>1234-1234-1234-1234</GiftCard>
					<Amount>25</Amount>
				</PurchaseGift>
			</PurchaseGifts>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<Number>5499990123456781</Number>
					<Expiration>0513</Expiration>
					<AvsStreet>4 Main St</AvsStreet>
					<AvsPostal>30329</AvsPostal>
					<CID>123</CID>
					<NameOnCard>John Doe</NameOnCard>
					<ChargeAmount>7.5</ChargeAmount>
				</Payment>
			</Payments>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <PurchaseGifts> |
|---|---|---|---|---|
| PurchaseGifts | PurchaseGift | Sub Node | No | Contains one or more gift card purchases |

| Node | Child Node | Type | Required | Description <PurchaseGift> |
|---|---|---|---|---|
| Amount | | Decimal | Yes | Amount to add to the gift card |
| GiftCard | | String | No | If specified, amount will be added to the gift card. If not specified, a new number will be generated and the amount added to the new number |

**Gift Card Purchase - Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<Code>0</Code>
			<ResponseText>OK</ResponseText>
			<TransactionID>1476206</TransactionID>
			<GiftPurchases>
				<GiftPurchase>
					<GiftNumber>1234-1234-1234-1234</GiftNumber>
					<Amount>1</Amount>
				</GiftPurchase>
			</GiftPurchases>
		</Response>
	</Packet>
</Response>
```

# Loyalty Card Information

**Registering a Loyalty Card – Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>GIFTINFORMATION</Command>
	<Data>
		<RegisterCards>
			<RegisterCard>
				<CardNumber>12345</CardNumber>
				<FirstName>Jon</FirstName>
				<LastName>Doe</LastName>
				<Address1>Address Line 1</Address1>
				<Address2>Address Line 2</Address2>
				<City>The City</City>
				<State>The State</State>
				<Postal>Postal Code</Postal>
				<RegisteredID>Reg ID</RegisteredID>
				<Email>Support@rts-solutions.com</Email>
				<DOB>19710323</DOB>
				<DoNotEmail>1</DoNotEmail>
			</RegisterCard>
		</RegisterCards>
	</Data>
</Request>
```

You can register multiple cards by passing in additional <RegisterCard> nodes.

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Command | | String | Yes | GIFTINFORMATION |
| Data | RegisterCards | Sub Node | Yes | |
| RegisterCards | RegisterCard | Sub Node | Yes | |
| CardNumber | | String | Yes | Card number to register |
| FirstName | | String | Yes | Customer first name |
| LastName | | String | Yes | Customer last name |
| Address1 | | String | Yes | Customer street address 1 |
| Address2 | | String | Yes | Customer street address 2 |
| City | | String | Yes | Customer city |
| State | | String | Yes | Customer state |
| Postal | | String | Yes | Customer postal code |
| RegisteredID | | String | Yes | ???? |
| Email | | String | Yes | Customer email address |
| DOB | | String | Yes | Customer date of birth |
| DoNotEmail | | Integer | Yes | 0 = email, 1 = do not email |

# Ticket Purchasing – Check Sold Out

**Checking Performance Sold / Remaining / SoldOut Levels – Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>CheckSoldOut</Command>
	<Data>
		<Packet>
			<PerformanceID>013732000045</PerformanceID>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | CheckSoldOut |
| PerformanceID | | String | Yes | The PerformanceID for the show to check |

**Checking Performance Sold / Remaining / SoldOut Levels – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<Sold>3</Sold>
	<IsReserved>0</IsReserved>
	<TotalSeats>500</TotalSeats>
	<SoldOut_General>2</SoldOut_General>
	<SoldOut_Internet>5</SoldOut_Internet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Sold | | Integer | | Number of tickets sold to this performance |
| IsReserved | | Integer | | 0 = General Admission, 1 = Reserved |
| TotalSeats | | Integer | | Total number of seats for the show |
| SoldOut_General | | Integer | | General sold out seats remaining level |
| SoldOut_Internet | | Integer | | Internet sold out seats remaining level |

An example of the levels meaning is: 500 seats total, Internet Sold Out happens with 5 seats remaining. Internet Sold Out happens when Sold hits 495.

If the number of tickets in a transaction is greater than the number seats remaining until Internet Sold Out happens, the transaction will fail.

# Ticket Purchasing – Check Redeem

This call allows you to see the status of a ticket redemption. If the tickets have not yet been redeemed, the transaction could be refunded or reversed. If the tickets have already been redeemed by the customer, the transaction cannot be altered.

**Checking Ticket Redemption Status – Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>CheckRedeem</Command>
	<Data>
		<Packet>
			<PickupNumber>558722177437</PickupNumber>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | CheckRedeem |
| PickupNumber | | String | Yes | The PickupNumber from the sale transaction |

**Checking Ticket Redemption Status – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<Redeemed>1</Redeemed>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Redeemed | | Boolean | Yes | 0 = Not Redeemed, 1 = Redeemed |

# Ticket Purchasing – Verify Transaction

This call allows you to see if a transaction was successfully received and processed. If, for example, you passed in a buy request (that uses the <ThirdPartyID> node) and never received a response from the point-of-sale, this call can be used to see if the transaction was received and processed.

Found transactions will return the <PickupNumber> associated with the purchase.

**Verify Transaction Status – Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>VERIFYTRANSACTION</Command>
	<Data>
		<Packet>
			<ThirdPartyID>63a96a1c66d44b568b3115aeeadc6b12</ThirdPartyID>
			<TransactionDateTimeUTC>20161024153500</TransactionDateTimeUTC>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | VerifyTransaction |
| ThirdPartyID | | String | Yes | Your ID for the sale transaction (must have been passed in with the original sale) |
| TransactionDateTimeUTC | | String | Yes | The sale transaction date/time in UTC (format: yyyyMMddhhmmss). |

**Verify Transaction Status – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<TransactionFound>1</TransactionFound>
	<PickupNumber>728722814391</PickupNumber>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| TransactionFound | | Boolean | Yes | 0 = Not Found, 1 = Found |
| PickupNumber | | String | No | If transaction found, this will be the pickup number for the customer redemption. |

# Ticket Purchasing – Transaction Details

This call let you retrieve the details of a previous transaction by the PickupNumber associated with the purchase.

**Transaction Details – Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>TRANSACTIONDETAILS</Command>
	<Data>
		<Packet>
			<PickupNumber>249728765499</PickupNumber>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | TransactionDetails |
| PickupNumber | | String | Yes | The PickupNumber from the sale transaction |

**Transaction Details – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<TransactionFound>1</TransactionFound>
	<Performances>
		<Performance>
			<Title>ALIEN: COVENANT</Title>
			<DateTime>052920182115</DateTime>
			<Auditorium>2</Auditorium>
			<Tickets>
				<Ticket>
					<Status>Redeemed - Online</Status>
					<PickupDateTime>052920181119</PickupDateTime>
					<Type>senior</Type>
					<IsReserved>1</IsReserved>
					<Section>Ruby</Section>
					<RowDesc>H</RowDesc>
					<ColDesc>15</ColDesc>
				</Ticket>
			</Tickets>
		</Performance>
	</Performances>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| TransactionFound | | Boolean | Yes | 0 = Not Found, 1 = Found |
| Performances | Performance | Sub-Node | No | If transaction found, this will contain the Performance(s) in the transaction. |

| Node | Child Node | Type | Required | Description <Performances> |
|---|---|---|---|---|
| Performance | | Sub-Node | No | If transaction found, there will be 1 or more Performance nodes, depending on the transaction contents |

| Node | Child Node | Type | Required | Description <Performance> |
|---|---|---|---|---|
| Title | | String | Yes | The title for this performance |
| DateTime | | String | Yes | Performance start time (MMddyyyyhhmm) |
| Auditorium | | String | Yes | The auditorium this performance is in |
| Tickets | Ticket | Sub-Node | Yes | The tickets for this performance |

| Node | Child Node | Type | Required | Description <Tickets> |
|---|---|---|---|---|
| Ticket | | Sub-Node | No | If transaction found, there will be 1 or more Performance nodes, depending on the transaction contents |

| Node | Child Node | Type | Required | Description <Ticket> |
|---|---|---|---|---|
| Status | | String | Yes | The current status of the ticket pickup:<br><br>Valid - Ticket is available for redemption<br>Expired - Ticket has expired<br>Refunded - Transaction was refunded via the API<br>Reversed - Transaction was reversed via the API<br>Redeemed - Online - Ticket was picked up via the API<br>Redeemed - Printed - Ticket was picked up in theatre |
| PickupDateTime | | String | Yes | Will be present if the ticket is not "Valid" or "Expired" (MMddyyyyhhmm) |
| PickupLocation | | String | Yes | Will be present if the ticket was picked up in theatre |
| PickupUser | | String | Yes | Will be present if the ticket was picked up in theatre |
| Type | | String | Yes | The type of ticket purchased |
| IsReserved | | Boolean | Yes | 0 = Not reserved, 1 = Reserved |
| Section | | String | Yes | Will be present if the ticket is reserved, indicates the section name the reserved seat is in |

| Node | Child Node | Type | Required | Description <Ticket> continued |
|---|---|---|---|---|
| RowDesc | | String | Yes | Will be present if the ticket is reserved, indicates the Row description |
| ColDesc | | String | Yes | Will be present if the ticket is reserved, indicates the Column (or Seat) description |

# Ticket Purchasing – Redeem Tickets

This call let you tell the POS system that you have redeemed tickets via a 3rd party app.

**Redeem – Sample Request Packet:**

```
<Request>
	<Version>1</Version>
	<Command>RDEEM</Command>
	<Data>
		<Packet>
			<PickupNumber>249728765499</PickupNumber>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | TransactionDetails |
| PickupNumber | | String | Yes | The PickupNumber from the sale transaction |

**Redeem – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<HasRedemptions>1</HasRedemptions>
	<HasExpirations>0</HasExpirations>
	<HasPickedUp>0</HasPickedUp>
	<ValidTickets>
		<Ticket>
			<Type>senior</Type>
			<IsReserved>1</IsReserved>
			<Section>Ruby</Section>
			<RowDesc>H</RowDesc>
			<ColDesc>15</ColDesc>
		</Ticket>
	</ValidTickets>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| HasRedemptions | | Boolean | Yes | 0 = False, 1 = True |
| HasExpirations | | Boolean | Yes | 0 = False, 1 = True |
| HasPickedUp | | Boolean | Yes | 0 = False, 1 = True |
| ValidTickets | Ticket | Sub-Node | No | If there are redemptions, this node will indicate the tickets being redeemed |
| ExpiredTickets | Ticket | Sub-Node | No | If there are expirations, this node will indicate the tickets that have expired |
| PickedUpTickets | Ticket | Sub-Node | No | If there are already redeemed tickets, this node will indicate the tickets already redeemed |

| Node | Child | Type | Required | Description <Ticket> |
|---|---|---|---|---|

| | Node | | | |
|---|---|---|---|---|
| Type | | String | Yes | The type of ticket purchased |
| IsReserved | | Boolean | Yes | 0 = Not reserved, 1 = Reserved |
| Section | | String | Yes | Will be present if the ticket is reserved, indicates the section name the reserved seat is in |
| RowDesc | | String | Yes | Will be present if the ticket is reserved, indicates the Row description |
| ColDesc | | String | Yes | Will be present if the ticket is reserved, indicates the Column (or Seat) description |

# Ticket Purchasing – Refunds

**Only use this call if you are processing the credit card transactions outside of the point-of-sale system!** Allows for refunding of ticket purchases BEFORE the show has started, as long as the tickets from the sale have not already been redeemed.

**Whole Transaction Refund – Sample Request Packet:**

```
<Request>
   <Version>1</Version>
   <Command>Refund</Command>
   <Data>
      <Packet>
         <Fees>
            <TicketFee>2</TicketFee>
            <TransactionFee>1</TransactionFee>
            <Adjust>3.50</Adjust>
         </Fees>
         <PickupNumber>220000528377</PickupNumber>
      </Packet>
   </Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | Refund |
| Fees | | Sub-Node | Yes | |
| TicketFee | | Decimal | Yes | The ticket fee from the sale transaction |
| TransactionFee | | Decimal | Yes | The transaction fee from the sale transaction |
| Adjust | | Decimal | Yes | The sale transaction total (positive amount). |
| PickupNumber | | String | Yes | The PickupNumber from the sale transaction |

**Partial Transaction Refund - Sample Request Packet:**

```
<Request>
   <Version>1</Version>
   <Command>Refund</Command>
   <Data>
      <Packet>
         <Fees>
            <TicketFee>-3.4</TicketFee>
            <TransactionFee>0</TransactionFee>
            <Adjust>0</Adjust>
         </Fees>
         <PickupNumber>312729628817</PickupNumber>
         <PartialRefunds>

            <!-- option 1: whole performance from a ticket day -->
            <PartialRefund>
               <PerformanceID>017988000012</PerformanceID>
            </PartialRefund>

            <!-- option 2: reserved partial ticket for performance-->
            <PartialRefund>
               <PerformanceID>017988000012</PerformanceID>
               <Tickets>
                  <Ticket>
                     <Amount>1</Amount>
                     <TypeCode>6</TypeCode>
                     <Seat>
                        <Section>21</Section>
                        <Row>7</Row>
                        <Col>10</Col>
                     </Seat>
                  </Ticket>
               </Tickets>
            </PartialRefund>

            <!-- option 3: general admission partial ticket for performance-->
            <PartialRefund>
               <PerformanceID>017988000012</PerformanceID>
               <Tickets>
                  <Ticket>
                     <Amount>1</Amount>
                     <TypeCode>6</TypeCode>
                  </Ticket>
               </Tickets>
            </PartialRefund>
         </PartialRefunds>
      </Packet>
   </Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | Refund |
| Fees | | Sub-Node | Yes | |
| TicketFee | | Decimal | Yes | The ticket fee from the sale transaction |
| TransactionFee | | Decimal | Yes | The transaction fee from the sale transaction |
| Adjust | | Decimal | Yes | The sale transaction total (positive amount) |
| PickupNumber | | String | Yes | The PickupNumber from the sale transaction, from any <TicketDay> in the purchase response |
| PartialRefunds | PartialRefund | Sub-Node | Yes | |

| Node | Child Node | Type | Required | Description <PartialRefunds> |
|---|---|---|---|---|
| PartialRefund | | Sub-Node | Yes | Include this node for each partial refund action you want to take |

| Node | Child Node | Type | Required | Description <PartialRefund> |
|---|---|---|---|---|
| PerformanceID | | String | Yes | The performance ID for the performance you want to refund against |
| Tickets | Ticket | Sub-Node | No | If not present, will refund all tickets for this performance |

| Node | Child Node | Type | Required | Description <Ticket> |
|---|---|---|---|---|
| Amount | | Integer | Yes | General Admission: The number of tickets to refund. Reserved Seating: Will be 1 and <Seat> is required |
| TypeCode | | String | Yes | The ID of the ticket type being refunded |
| Seat | | Sub-Node | No | If performance is reserved seating this is where you specify the seat to be refunded |

| Node | Child Node | Type | Required | Description <Seat> |
|---|---|---|---|---|
| Section | | String | Yes | The section ID for the seat being refunded |
| Row | | Integer | Yes | The row index for the seat being refunded |
| Col | | Integer | Yes | The col index for the seat being refunded |

**Refund – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<ResponseText>OK</ResponseText>
			<TransactionID>14722824</TransactionID>
		</Response>
	</Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Code | | Char | Yes | -1 |
| TransactionFound | | Boolean | Yes | 0 = Not Found, 1 = Found |
| PickupNumber | | String | No | If transaction found, this will be the pickup number for the customer redemption. |

# Ticket Purchasing – Reverse Transaction

**Only use this call if you are processing the credit card transactions outside of the point-of-sale system!** Allows for reversal of a sale transaction by ThirdPartyID, in the event that you never received a response with a PickupNumber. Will be allowed as long as the tickets from the sale have not already been redeemed, and the show is still scheduled. Should be used in conjunction with "VerifyTransaction" to establish the state of the sale.

**ReverseTransaction – Sample Request Packet:**

```
<Request>
   <Version>1</Version>
   <Command>ReverseTransaction</Command>
   <Data>
      <Packet>
         <Fees>
            <TicketFee>2</TicketFee>
            <TransactionFee>1</TransactionFee>
            <Adjust>3.50</Adjust>
         </Fees>
         <ThirdPartyID>220000528377</ThirdPartyID>
      </Packet>
   </Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| Command | | String | Yes | Refund |
| Fees | | Sub-Node | Yes | |
| TicketFee | | Decimal | Yes | The ticket fee from the sale transaction |
| TransactionFee | | Decimal | Yes | The transaction fee from the sale transaction |
| Adjust | | Decimal | Yes | The sale transaction total (positive amount). |
| ThirdPartyID | | String | Yes | Your ID for the sale transaction (must have been passed in with the original sale) |

**ReverseTransaction – Sample Response Packet:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<ResponseText>OK</ResponseText>
			<TransactionID>14722826</TransactionID>
		</Response>
	</Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | -1 |
| TransactionFound | | Boolean | Yes | 0 = Not Found, 1 = Found |
| PickupNumber | | String | No | If transaction found, this will be the pickup number for the customer redemption. |

# Ticket Purchasing (E2E) Overview

The introduction of End-to-End Encryption to the credit card processing system has changed the way credit cards are processed by the software. Instead of the software processing the transaction, it is processed directly at the processor via a HostedCheckout system. This means the customer needs to be redirected to a secure credit card entry page that is provided by the credit card processing company.

To summarize the process:
1. A request is sent into the software to create a payment at the processor. This request contains the charge amount to authorize and 2 URLs: a ProcessCompleteUrl, and a ReturnUrl. Be sure to URL Encode your URLs.

2. The software will return a create payment response which indicates if the create was successful, a TransactionId, a RedirectURL to send the customer to for credit card information entry, and the PostData to send into the processor page you redirect the customer to. If the create was unsuccessful there will be an ErrorMsg present to indicate the issue.

3. The customer will then need to be redirected to the RedirectURL in the create payment response, posting in the PostData in the create payment response. If the customer completes the process the processor will redirect back to the ProcessCompleteURL from step 1, which will contain the processor response in the PostData. If the customer cancels the process, the processor will redirect back to the ReturnURL.

4. If the payment was successful, a Buy request is then sent back to the software with the TransactionId, ProcessCompletePostData included in the Payment node. The software will then validate the payment and return either a failure or a success.

It is important to note that you could repeat steps 1 – 3 for multiple credit cards if needed, then simply include multiple Payment nodes in step 4. Be sure to authorize each payment for only the amount needed, not the total transaction amount.

# Ticketing Purchasing (E2E)

### Create a HostedCheckout Payment – Sample Request:

```
<Request>
	<Version>1</Version>
	<Command>CreatePayment</Command>
	<Data>
		<Packet>
			<ChargeAmount>0.62</ChargeAmount>

	<ProcessCompleteUrl>http%3A%2F%2Flocal.formovietickets.com%2FT.asp%3FWCI%3DBT%26Page%
3DConfirm%26Mode%3DCCPAYRESPONSE</ProcessCompleteUrl>

	<ReturnUrl>http%3A%2F%2Flocal.formovietickets.com%2FT.asp%3FWCI%3DBT%26Page%3DConfirm
%26Mode%3DPayments</ReturnUrl>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description |
|---|---|---|---|---|
| ChargeAmount | | Decimal | Yes | The amount this payment will authorize for. |
| ProcessCompleteUrl | | String | Yes | The URL the customer will be redirected back to after they complete the credit card entry process |
| ReturnUrl | | String | Yes | The URL the customer will be redirected back to if they cancel the credit card entry process |

### Create a HostedCheckout Payment – Sample Response:

```
<Response>
	<Version>1</Version>
	<Packet>
		<CreatePayment>
			<Worked>1</Worked>
			<TransactionId>3abac539-1731-421d-9291-989dffb65688</TransactionId>

	<RedirectUrl>https%3A%2F%2Fhc.mercurydev.net%2FCheckout.aspx</RedirectUrl>
			<PostData>587c7d7d-d990-4f29-8ff6-d121ddd98d6f</PostData>
		</CreatePayment>
	</Packet>
</Response>
```

### Ticket Purchase (E2E) – Sample Request – PerformanceID:

Note: See **Ticket Purchasing – General Admission** and **Ticket Purchasing – Reserved Seating** for additional information on the Buy packet nodes

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseTitles>
				<PurchaseTitle>
					<PerformanceID>000166000023</PerformanceID>
					<Tickets>
						<Ticket>
							<Amount>2</Amount>
							<TypeCode>5</TypeCode>
						</Ticket>
					</Tickets>
				</PurchaseTitle>
			</PurchaseTitles>
			<Fees>
				<TicketFee>0</TicketFee>
				<TransactionFee>0</TransactionFee>
				<Adjust>0</Adjust>
			</Fees>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<TransactionId>5f627c67-8f5e-4f41-b26b-
e8d2c8080167</TransactionId>
					<ProcessCompletePostData>PaymentID=cf679a14-dd17-4c0a-
b151-
be0fdcb27dfd&ReturnCode=0&ReturnMessage=Your+transaction+has+been+approved.</ProcessComplet
ePostData>
					<ChargeAmount>0.62</ChargeAmount>
				</Payment>
			</Payments>
			<CustomerInfo>
				<EmailAddress>testemail@yahoo.com</EmailAddress>
			</CustomerInfo>
		</Packet>
	</Data>
</Request>
```

**Ticket Purchase (E2E) – Sample Response:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<Code>0</Code>
			<ResponseText>OK</ResponseText>
			<TransactionID>1524289</TransactionID>
			<Pickups>
				<Pickup>
					<TicketDay>20101213</TicketDay>
					<PickupNumber/>
					<BarCodes>
						<BarCode>
							<Type>Transaction</Type>
							<CodeType>UPC</CodeType>
							<BarCodeData>525524289675</BarCodeData>
						</BarCode>
					</BarCodes>
				</Pickup>
			</Pickups>
		</Response>
	</Packet>
</Response>
```

# Ticket Purchasing (3rd Party Credit Card Processing)

This method of ticketing purchasing is to be used when the RTS software is not handling anything in the credit card processing chain.

### Ticket Purchase (3rd Party) – Sample Request – PerformanceID:

Note: The key differences between a 3rd party buy and a normal buy is the used of the <Adjust> node inside <Fees> to offset the amount charged by the 3rd party, and the <ChargeAmount> being zero.

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseTitles>
				<PurchaseTitle>
					<PerformanceID>000158000012</PerformanceID>
					<Tickets>
						<Ticket>
							<Amount>1</Amount>
							<TypeCode>5</TypeCode>
						</Ticket>
					</Tickets>
				</PurchaseTitle>
			</PurchaseTitles>
			<Fees>
				<TicketFee>1</TicketFee>
				<TransactionFee>0</TransactionFee>
				<Adjust>-1.31</Adjust>
			</Fees>
			<Payments>
				<Payment>
					<ChargeAmount>0</ChargeAmount>
				</Payment>
			</Payments>
			<ThirdPartyID>3677745fb2cb4e20bc6f289fd65753aa</ThirdPartyID>
			<CustomerInfo>
				<EmailAddress>testemail@yahoo.com</EmailAddress>
			</CustomerInfo>
		</Packet>
	</Data>
</Request>
```

# Ticket Purchasing – General Admission

The performance you wish to purchase can be specified in 3 different ways:

1. Title, ShowTime, and AuditoriumID
2. PerformanceID
3. MtFilmCode, MTPerformanceID, and ShowTime

**Ticket Purchase - Sample Request – Title, ShowTime, and AuditoriumID:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseTitles>
				<PurchaseTitle>
					<Title>The Title</Title>
					<ShowTime>200812051235</ShowTime>
					<Auditorium>1</Auditorium>
					<Tickets>
						<Ticket>
							<Amount>1</Amount>
							<TypeCode>1|1</TypeCode>
						</Ticket>
					</Tickets>
				</PurchaseTitle>
			</PurchaseTitles>
			<Fees>
				<TicketFee>1</TicketFee>
				<TransactionFee>1</TransactionFee>
				<Adjust>0</Adjust>
			</Fees>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<Number>5499990123456781</Number>
					<Expiration>0513</Expiration>
					<AvsStreet>4 Main St</AvsStreet>
					<AvsPostal>30329</AvsPostal>
					<CID>123</CID>
					<NameOnCard>John Doe</NameOnCard>
					<ChargeAmount>9.5</ChargeAmount>
				</Payment>
			</Payments>
			<Loyalty>
				<Cards>
					<Type>Earn</Type>
					<CardNumber>12345</CardNumber>
				</Cards>
			</Loyalty>
			<ThirdPartyID>3677745fb2cb4e20bc6f289fd65753aa</ThirdPartyID>
			<CustomerInfo>
				<EmailAddress>testemail@yahoo.com</EmailAddress>
			</CustomerInfo>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <PurchaseTitle> |
|---|---|---|---|---|
| Title | | String | Yes | Performance Title |
| ShowTime | | String | Yes | Performance start time (yyyyMMddhhmm) |
| Auditorium | | String | Yes | Performance auditorium number |
| Tickets | Ticket | Sub Node | Yes | One or more ticket types to purchase |

**Ticket Purchase - Sample Request – PerformanceID:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseTitles>
				<PurchaseTitle>

	<PerformanceID>+3934kd|392372374|23784237</PerformanceID>
					<Tickets>
						<Ticket>
							<Amount>1</Amount>
							<TypeCode>1|1</TypeCode>
						</Ticket>
					</Tickets>
				</PurchaseTitle>
			</PurchaseTitles>
			<Fees>
				<TicketFee>1</TicketFee>
				<TransactionFee>1</TransactionFee>
				<Adjust>0</Adjust>
			</Fees>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<Number>5499990123456781</Number>
					<Expiration>0513</Expiration>
					<AvsStreet>4 Main St</AvsStreet>
					<AvsPostal>30329</AvsPostal>
					<CID>123</CID>
					<NameOnCard>John Doe</NameOnCard>
					<ChargeAmount>9.5</ChargeAmount>
				</Payment>
			</Payments>
			<Loyalty>
				<Cards>
					<Type>Earn</Type>
					<CardNumber>12345</CardNumber>
				</Cards>
			</Loyalty>
			<ThirdPartyID>3677745fb2cb4e20bc6f289fd65753aa</ThirdPartyID>
			<CustomerInfo>
				<EmailAddress>testemail@yahoo.com</EmailAddress>
			</CustomerInfo>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <PurchaseTitle> |
|---|---|---|---|---|
| PerformanceID | | String | Yes | The PerformanceID |
| Tickets | Ticket | | Yes | One or more ticket types to purchase |

**Ticket Purchase - Sample Request – MtFilmCode, MtPerformanceID, and ShowTime:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseTitles>
				<PurchaseTitle>
					<PerformanceID>93j2kd902j42d</PerformanceID>
					<ShowTime>200812051235</ShowTime>
					<MtFilmCode>TH12345</MtFilmCode>
					<Tickets>
						<Ticket>
							<Amount>1</Amount>
							<TypeCode>1|1</TypeCode>
						</Ticket>
					</Tickets>
				</PurchaseTitle>
			</PurchaseTitles>
			<Fees>
				<TicketFee>1</TicketFee>
				<TransactionFee>1</TransactionFee>
				<Adjust>0</Adjust>
			</Fees>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<Number>5499990123456781</Number>
					<Expiration>0513</Expiration>
					<AvsStreet>4 Main St</AvsStreet>
					<AvsPostal>30329</AvsPostal>
					<CID>123</CID>
					<NameOnCard>John Doe</NameOnCard>
					<ChargeAmount>9.5</ChargeAmount>
				</Payment>
			</Payments>
			<Loyalty>
				<Cards>
					<Type>Earn</Type>
					<CardNumber>12345</CardNumber>
				</Cards>
			</Loyalty>
			<CustomerInfo>
				<EmailAddress>testemail@yahoo.com</EmailAddress>
			</CustomerInfo>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <PurchaseTitle> |
|---|---|---|---|---|
| MtFilmCode | | String | Yes | Movietickets.com film code |
| MtPerformanceID | | String | Yes | Movietickets.com PerformanceID |
| ShowTime | | String | Yes | Performance start time (yyyyMMddhhmm) |
| Tickets | Ticket | | Yes | One or more ticket types to purchase |

To sell multiple tickets in the same transaction, duplicate the <Ticket> node for each type and specify the amount of each ticket.

For example, to sell 3 tickets of 2 different types, your <Tickets> node would look like:

```
<Tickets>
	<Ticket>
		<Amount>2</Amount>
		<TypeCode>1|1</TypeCode>
	</Ticket>
	<Ticket>
		<Amount>1</Amount>
		<TypeCode>1|2</TypeCode>
	</Ticket>
</Tickets>
```

To accept multiple payment types in the same transaction, duplicate the <Payment> node for each type of payment, and specify the appropriate information.

For example, to take credit card and gift card, your <Payment> node would look like:

```
<Payments>
	<Payment>
		<Type>CreditCard</Type>
		<Number>5499990123456781</Number>
		<Expiration>0513</Expiration>
		<AvsStreet>4 Main St</AvsStreet>
		<AvsPostal>30329</AvsPostal>
		<CID>123</CID>
		<NameOnCard>John Doe</NameOnCard>
		<ChargeAmount>9.5</ChargeAmount>
	</Payment>
	<Payment>
	<Payment>
		<Type>Gift</Type>
		<ChargeAmount>1.00</ChargeAmount>
		<GiftNumber>1234123412341234</GiftNumber>
		<GiftCardPIN>12345</GiftCardPIN>
	</Payment>
</Payments>
```

| Node | Child Node | Type | Required | Description <Ticket> |
|---|---|---|---|---|
| Amount | | Integer | Yes | Number of this type of ticket to purchase |
| TypeCode | | String | Yes | Ticket type code (from schedule XML) |

| Node | Child Node | Type | Required | Description <Fees> |
|---|---|---|---|---|
| TicketFee | | Decimal | No | Amount to charge for ticket fees in the transaction. Recorded in the POS as the configured 'Ticket Fee' item. |
| TransactionFee | | Decimal | No | Amount to charge for the transaction fees in the transaction. Recorded in the POS as the configured 'Transaction Fee' item. |
| Adjust | | Decimal | No | Allows for positive or negative adjustment to the charge amount. Recorded in the POS as the configured 'Adjust' item. |

| Node | Child Node | Type | Required | Description <Payments> |
|---|---|---|---|---|
| Payments | Payment | Sub Node | Yes | Contains one or more payment types for a transaction |

| Node | Child Node | Type | Required | Description <Payment> |
|---|---|---|---|---|
| Type | | String | Yes | Specifies the payment type: **CreditCard** or **Gift** |
| Number | | String | Yes | Credit card number |
| GiftNumber | | String | Yes | Gift card number |
| GiftCardPIN | | String | Maybe | Certain gift card processors in RTS can be marked as requiring a PIN during purchase. If this is enabled, this node is required. |
| Expiration | | String | Yes | Credit card expiration date (MMyy) |
| AvsStreet | | String | No | Credit card billing street address |
| AvsPostal | | String | No | Credit card billing postal code |
| CID | | String | No | Credit card CID (CVV, Security Code) number |
| NameOnCard | | String | No | Credit card customer name |
| ChargeAmount | | Decimal | Yes | Amount to charge on the credit or gift card |

| Node | Child Node | Type | Required | Description <ThirdPartyID> |
|---|---|---|---|---|
| ThirdPartyID | | Sub Node | No | Allows you to specify a transaction ID to the purchase, this can be used with "VerifyTransaction" to check for completion of the sale if normal response is not received. |

| Node | Child Node | Type | Required | Description <Loyalty> |
|---|---|---|---|---|
| Loyalty | Cards | Sub Node | No | |
| Type | | String | No | Earn |
| CardNumber | | String | No | Card number to use for loyalty earn. Multiple cards can be passed in via additional <CardNumber> tags |

| Node | Child Node | Type | Required | Description <CustomerInfo> |
|---|---|---|---|---|
| EmailAddress | | String | No | Customer email address. Can be used to look-up transactions in the POS.<br><br>NOTE: There is an option on the API selling account to have RTS send a confirmation email on a successful purchase. This email is NOT customizable. |

**Ticket Purchase - Sample Response:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<Code>0</Code>
			<ResponseText>OK</ResponseText>
			<TransactionID>1524289</TransactionID>
			<Pickups>
				<Pickup>
					<TicketDay>20101213</TicketDay>
					<PickupNumber/>
					<BarCodes>
						<BarCode>
							<Type>Transaction</Type>
							<CodeType>UPC</CodeType>
							<BarCodeData>525524289675</BarCodeData>
						</BarCode>
					</BarCodes>
				</Pickup>
			</Pickups>
		</Response>
	</Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Code | | Integer | Yes | -1 |
| Packet | Response | Sub Node | Yes | |
| ResponseText | | String | Yes | OK |
| TransactionID | | String | Yes | Transaction receipt number |
| Pickups | Pickup | Sub Node | Yes | |
| TicketDay | | String | Yes | Date of performance (yyyyMMdd) |
| PickupNumber | | String | Yes | Not currently used |
| BarCodes | BarCode | Sub Node | Yes | |
| Type | | String | Yes | Transaction |
| CodeType | | String | Yes | The bar code type: UPC |
| BarCodeData | | String | Yes | The confirmation number to be encoded. BAR CODE IMAGE IS NOT PROVIDED BY THE POS. |

# Seating Charts

All RTS layouts are built around the grid size provided in the <GridRef> node. Using these values in conjunction with your own known grid values, you can determine the ratio needed to convert each seat xpos, ypos, width, and height, to your own values, for example:

Your grid width divided by our grid width = your X ratio
Your grid height divided by our grid height = your Y ratio
Our Xpos multiplied by X ratio = your Xpos
Our Ypos multiplied by Y ratio = your Ypos
Our Width multiplied by X ratio = your width
Our Height multiplied by Y ration = your height

**Get Seat Layouts – Sample Request:**

```
<Request>
      <Version>1</Version>
      <Command>GetSeatLayouts</Command>
</Request>
```

This call allows you to pull the reserved seating layouts for all of the auditoriums in the point-of-sale system. It also includes a list of seat types configured in the system, so that any custom seat types are available to you.

**Get Seat Layouts – Sample Response:**

```
<Response>
    <Code>-1</Code>
    <ResponseText>OK</ResponseText>
    <GridRef>
        <Width>10000</Width>
        <Height>10000</Height>
    </GridRef>
    <SeatTypes>
        <Type>
            <ID>0</ID>
            <CustomerName>Normal</CustomerName>
        </Type>
        ......
        <Type>
            <ID>17</ID>
            <CustomerName>CompanionLeft</CustomerName>
        </Type>
    </SeatTypes>
    <Layouts>
        <Layout>
            <LayoutName>Reserved Auditorium 13</LayoutName>
            <SectionName>Section 1</SectionName>
            <SectionID>8</SectionID>
            <Seats>
                <Seat>
                    <HideOnInternet>0</HideOnInternet>
                    <HideRowDesc>0</HideRowDesc>
                    <HideSeparator>0</HideSeparator>
                    <RowIndex>1</RowIndex>
                    <ColIndex>1</ColIndex>
                    <RowDesc>A</RowDesc>
```

```
                    <ColDesc>1</ColDesc>
                    <TypeID>0</TypeID>
                    <GroupID>1</GroupID>
                    <GroupDesc>1</GroupDesc>
                    <GroupPos>1</GroupPos>
                    <XPos>2539</XPos>
                    <YPos>2659</YPos>
                    <Width>456</Width>
                    <Height>740</Height>
                    <Angle>0</Angle>
                </Seat>
                .......
                <Seat>
                    <HideOnInternet>0</HideOnInternet>
                    <HideRowDesc>0</HideRowDesc>
                    <HideSeparator>0</HideSeparator>
                    <RowIndex>3</RowIndex>
                    <ColIndex>13</ColIndex>
                    <RowDesc>C</RowDesc>
                    <ColDesc>13</ColDesc>
                    <TypeID>0</TypeID>
                    <GroupID>3</GroupID>
                    <GroupDesc>13</GroupDesc>
                    <GroupPos>13</GroupPos>
                    <XPos>8193</XPos>
                    <YPos>5305</YPos>
                    <Width>442</Width>
                    <Height>740</Height>
                    <Angle>0</Angle>
                </Seat>
            </Seats>
        </Layout>
    </Layouts>
</Response>
```

| Node | Child Node | Type | Required | Description <GridRef> |
|---|---|---|---|---|
| Width | | Integer | Yes | The relative grid width |
| Height | | Integer | Yes | The relative grid height |

| Node | Child Node | Type | Required | Description <SeatTypes> |
|---|---|---|---|---|
| Type | | Sub-Node | Yes | |
| ID | | Integer | Yes | The ID of the SeatType |
| CustomerName | | String | Yes | The customer name for the SeatType |

| Node | Child Node | Type | Required | Description <Layouts> |
|---|---|---|---|---|
| Layout | | Sub-Node | Yes | There will be a <Layout> node for each system layout |
| LayoutName | | String | Yes | The name of the layout |
| SectionName | | String | Yes | The name of the section |
| SectionID | | Integer | Yes | The ID of the section |
| Seats | Seat | Sub-Node | Yes | |

| Node | Child Node | Type | Required | Description <Seat> |
|---|---|---|---|---|
| HideOnInternet | | Boolean | Yes | 0 = Do not hide, 1 = Needs to be hidden |
| HideRowDesc | | Boolean | Yes | 0 = Do not hide row desc., 1 = hide row desc. |
| HideSeparator | | Boolean | Yes | 0 = Do not hide row/col separator, 1 = hide separator |
| RowIndex | | Integer | Yes | The row index of the seat |
| ColIndex | | Integer | Yes | The column index of the seat |
| RowDesc | | String | Yes | The row descriptor of the seat |
| ColDesc | | String | Yes | The column descriptor of the seat |
| TypeID | | Integer | Yes | The SeatType ID for the seat |
| GroupID | | Integer | Yes | The group ID for the seat |
| GroupDesc | | String | Yes | The group description for the seat |
| GroupPos | | Integer | Yes | The seat position in the group |
| Xpos | | Integer | Yes | The relative X position of the seat in the layout |
| Ypos | | Integer | Yes | The relative Y position of the seat in the layout |
| Width | | Integer | Yes | The relative width of the seat in the layout |
| Height | | Integer | Yes | The relative height of the seat in the layout |
| Angle | | Integer | Yes | The angle of the seat in the layout |

!! NOTE: This call has been slightly modified to support locations that use custom seat pricing, these are locations that have different prices on specific seats in an auditorium. Due to this pricing not being known until the seat selection has been made, this call will return the pricing group for each seat, and a list of pricing groups and tickets. In order to access this data, this call should be made using the SCHEDULE credentials, not the SELLING credentials. Making this call with the selling credentials will not return any pricing data. !!

**Get Seat Plan For Performance – Sample Request:**

```
<Request>
      <Version>1</Version>
      <Command>GETSEATPLANFORPERF</Command>
      <Data>
            <Packet>
                  <PerformanceID>008180000166</PerformanceID>
            </Packet>
      </Data>
</Request>
```

This call allows you to pull the reserved seating layouts for a specific performance, including the status of each seat. It also returns a list of seat types used in the layout.

**Get Seat Plan For Performance – Sample Response (made with SCHEDULE credentials):**

```
<Response>
   <Version>1</Version>
   <Code>-1</Code>
   <Packet>
      <Response>
         <Code>-1</Code>
         <ResponseText>OK</ResponseText>
         <GridRef>
            <Width>10000</Width>
            <Height>10000</Height>
         </GridRef>
         <Layout>
            <Pricing>
               <Tickets>
                  <Ticket>
                     <Code>7</Code>
                     <Name>Student</Name>
                     <Price>0.71</Price>
                     <Tax>0.01</Tax>
                     <Points>8</Points>
                  </Ticket>
                  ...
                  <Ticket>
                     <Code>3</Code>
                     <Name>Senior</Name>
                     <Price>0.31</Price>
                     <Tax>0.01</Tax>
                     <Points>7</Points>
                  </Ticket>
               </Tickets>
               <PriceGroups>
                  <PriceGroup>
                     <Code>1</Code>
```

```
                  <Tickets>
                     <Ticket>
                        <Code>7</Code>
                     </Ticket>
                     ...
                     <Ticket>
                        <Code>3</Code>
                     </Ticket>
                  </Tickets>
               </PriceGroup>
            </PriceGroups>
         </Pricing>
         <SeatTypes>
            <SeatType>
               <ID>10</ID>
               <CustomerName>Recliner</CustomerName>
            </SeatType>
               ...
            <SeatType>
               <ID>1</ID>
               <CustomerName>Wheelchair</CustomerName>
            </SeatType>
         </SeatTypes>
         <Seats>
            <Seat>
               <ColDes>6</ColDes>
               <ColNum>7</ColNum>
               <RowDes>A</RowDes>
               <RowNum>0</RowNum>
               <Section>4</Section>
               <Type>9</Type>
               <GroupID>4</GroupID>
               <Status>0</Status>
               <XPos>3172</XPos>
               <YPos>2833</YPos>
               <Width>473</Width>
               <Height>706</Height>
               <Angle>0</Angle>
               <PriceGroupCode>1</PriceGroupCode>
            </Seat>
               ...
            <Seat>
               <ColDes>20</ColDes>
               <ColNum>20</ColNum>
               <RowDes>H</RowDes>
               <RowNum>7</RowNum>
               <Section>4</Section>
               <Type>10</Type>
               <GroupID>59</GroupID>
               <Status>0</Status>
               <XPos>9061</XPos>
               <YPos>8475</YPos>
               <Width>453</Width>
               <Height>598</Height>
               <Angle>0</Angle>
               <PriceGroupCode>1</PriceGroupCode>
            </Seat>
         </Seats>
      </Layout>
   </Response>
</Packet>
</Response>
```

**Get Seat Plan For Performance – Sample Response (made with SELLING credentials):**

```
<Response>
    <Version>1</Version>
    <Code>-1</Code>
    <Packet>
        <Response>
            <Code>-1</Code>
            <ResponseText>OK</ResponseText>
            <GridRef>
                <Width>10000</Width>
                <Height>10000</Height>
            </GridRef>
            <Layout>
                <SeatTypes>
                    <SeatType>
                        <ID>0</ID>
                        <CustomerName>Normal</CustomerName>
                    </SeatType>
                </SeatTypes>
                <Seats>
                    <Seat>
                        <ColDes>1</ColDes>
                        <ColNum>0</ColNum>
                        <RowDes>E</RowDes>
                        <RowNum>2</RowNum>
                        <Section>21</Section>
                        <Type>0</Type>
                        <GroupID>2</GroupID>
                        <Status>0</Status>
                        <XPos>1627</XPos>
                        <YPos>3057</YPos>
                        <Width>448</Width>
                        <Height>780</Height>
                        <Angle>0</Angle>
                    </Seat>
                    ......
                    <Seat>
                        <ColDes>1</ColDes>
                        <ColNum>0</ColNum>
                        <RowDes>D</RowDes>
                        <RowNum>4</RowNum>
                        <Section>21</Section>
                        <Type>0</Type>
                        <GroupID>3</GroupID>
                        <Status>0</Status>
                        <XPos>1627</XPos>
                        <YPos>4013</YPos>
                        <Width>448</Width>
                        <Height>780</Height>
                        <Angle>0</Angle>
                    </Seat>
                </Seats>
            </Layout>
        </Response>
    </Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <GridRef> |
|---|---|---|---|---|
| Width | | Integer | Yes | The relative grid width |
| Height | | Integer | Yes | The relative grid height |

| Node | Child Node | Type | Required | Description <Layouts> |
|---|---|---|---|---|
| Layout | | Sub-Node | Yes | |
| Pricing | Tickets, PriceGroups | Sub-Node | Maybe | If request was made using SCHEDULE credentials: The <Pricing> node will contain a <Tickets> node and a <PriceGroups> node. |
| Tickets | Ticket | Sub-Node | Maybe | If request was made using SCHEDULE credentials: The <Tickets> node will contain a number of <Ticket> nodes that are the tickets in use in this layout. |
| PriceGroups | PriceGroup | Sub-Node | Maybe | If request was made using SCHEDULE credentials: There will be a <PriceGroup> node for each unique price group in the layout |
| SeatTypes | SeatType | Sub-Node | Yes | There will be a <SeatType> node for each type used in this layout |
| Seats | Seat | Sub-Node | Yes | |

| Node | Child Node | Type | Required | Description <Ticket> |
|---|---|---|---|---|
| Code | | String | Yes | Ticket code used in purchase transactions |
| Name | | String | Yes | Ticket name as it should be displayed on the Internet |
| Price | | Decimal | Yes | Total ticket price |
| Tax | | Decimal | Maybe | Any tax included in the ticket price |
| Points | | Integer | Maybe | Any points that are earned when buying this ticket type |

| Node | Child Node | Type | Required | Description <PriceGroup> |
|---|---|---|---|---|
| Code | | Integer | Yes | The unique ID for this PriceGroup node, this ID is what links the <Seat> to the <PriceGroup> |
| Tickets | Ticket | Sub-Node | Yes | There will be a <Ticket> node for each ticket type that is part of this PriceGroup |
| Ticket | Code | Sub-Node | Yes | Each <Ticket>will contain a <Code> node that links back to the <Tickets><Ticket> nodes under the <Pricing> node. |

| Node | Child Node | Type | Required | Description <SeatType> |
|---|---|---|---|---|
| Type | | Sub-Node | Yes | |
| ID | | Integer | Yes | The ID of the SeatType |
| CustomerName | | String | Yes | The customer name for the SeatType |

| Node | Child Node | Type | Required | Description <Seat> |
|---|---|---|---|---|
| RowNum | | Integer | Yes | The row index of the seat |
| ColNum | | Integer | Yes | The column index of the seat |
| RowDes | | String | Yes | The row descriptor of the seat |
| ColDes | | String | Yes | The column descriptor of the seat |
| Section | | String | Yes | The section ID of the seat |
| Type | | Integer | Yes | The SeatType ID for the seat |
| GroupID | | Integer | Yes | The group ID for the seat |
| Status | | Integer | Yes | 0 = unsold, 1 = sold, 2 = locked, 3 = held, 4 = open tab, 5 = broken |
| Xpos | | Integer | Yes | The relative X position of the seat in the layout |
| Ypos | | Integer | Yes | The relative Y position of the seat in the layout |
| Width | | Integer | Yes | The relative width of the seat in the layout |
| Height | | Integer | Yes | The relative height of the seat in the layout |
| Angle | | Integer | Yes | The angle of the seat in the layout |
| PriceGroupCode | | Integer | Maybe | If request was made using SCHEDULE credentials: This value links back to the appropriate <PriceGroups><PriceGroup><Code>, which shows what <Tickets><Ticket> are available for this seat. |

# Reserved Seating – Check Seat Picks

This call allows for you to verify the point-of-sale system will allow the sale of the seats picked. This is in order to stop customer from leaving too many single seats across the auditorium.

**Reserved Seating – Check Seat Picks – Sample Request:**

```
<Request>
	<Command>CHECKSEATPICKS</Command>
	<Data>
		<Packet>
			<PerformanceID>008691000166</PerformanceID>
			<Seats>
				<Seat>
					<RowIndex>0</RowIndex>
					<ColIndex>0</ColIndex>
				</Seat>
			</Seats>
		</Packet>
	</Data>
</Request>
```

If there are multiple seats to check, just add the appropriate <Seat></Seat> information.

| Node | Child Node | Type | Required | Description <Seats> |
|---|---|---|---|---|
| Seats | Seat | Sub Node | Yes | |

| Node | Child Node | Type | Required | Description <Seat> |
|---|---|---|---|---|
| Row | | Integer | Yes | Row number |
| Col | | Integer | Yes | Column number |

**Reserved Seating – Check Seat Picks – Sample Response:**

```
<Response>
	<Version>1</Version>
	<ValidSelections>1</ValidSelections>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Char | Yes | 1 |
| ValidSelections | | Integer | Yes | 0 = Invalid, 1 = Valid |
| InvalidReason | | String | Maybe | If invalid selection, this will be the reason why. |

**Seating Chart - Sample Request – PerformanceID:**

```
<Request>
	<Version>1</Version>
	<Command>SeatChart</Command>
	<Data>
		<Packet>
			<PerformanceID>CA25640|201010222300|9</PerformanceID>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Command | | String | Yes | SeatChart |
| PerformanceID | | String | Yes | Performance ID for show |

**Seating Chart - Sample Request – MtFilmCode, MtPerformanceID, and ShowTime:**

```
<Request>
	<Version>1</Version>
	<Command>SeatChart</Command>
	<Data>
		<Packet>
			<MtPerformanceID>1</MtPerformanceID>
			<MtFilmCode>123</MtFilmCode>
			<ShowTime>201010201430</ShowTime>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Command | | String | Yes | SeatChart |
| MtPerformanceID | | String | Yes | Movietickets.com performance id for show |
| MtFilmCode | | String | Yes | Movietickets.com film code for title |
| ShowTime | | String | Yes | Performance date/time (yyyyMMddhhmm) |

**Seating Chart - Sample Response:**

```
<Response>
	<Code>0</Code>
	<ResponseText>OK</ResponseText>
	<Layout>
		<Seats>
			<Seat>
				<ColDes>11</ColDes>
				<ColNum>1</ColNum>
				<RowDes>D</RowDes>
				<RowNum>3</RowNum>
				<Section>11</Section>
				<Type>0</Type>
				<Status>0</Status>
			</Seat>
			<Seat>
				<ColDes>10</ColDes>
				<ColNum>1</ColNum>
				<RowDes>F</RowDes>
				<RowNum>6</RowNum>
				<Section>11</Section>
				<Type>0</Type>
				<GroupID>20</GroupID>
				<Status>0</Status>
			</Seat>
		</Seats>
	</Layout>
</Response>
```

| Node | Child Node | Type | Required | Description <Seat> |
|---|---|---|---|---|
| ColDesc | | String | Yes | Column description |
| ColNum | | Integer | Yes | Column number |
| RowDesc | | String | Yes | Row description |
| RowNum | | Integer | Yes | Row number |
| Section | | String | Yes | Section name |
| Type | | Integer | Yes | Type of seat (see seat type list) |
| GroupID | | Integer | No | Grouped seats will have the same GroupID |
| Status | | Integer | Yes | 0 = unsold, 1 = sold, 2 = locked, 3 = held |

# Holding and Releasing Seats

A hold seat request will hold the requested seats in a pending state for 5 minutes. Held seats can, and should, be released if the customer switches to different seats.

**Reserved Seating – Hold Seats – Sample Request:**

```
<Request>
	<Version>1</Version>
	<Command>HoldSeats</Command>
	<Data>
		<Packet>
			<Command>Hold</Command>
			<Performances>
				<Performance>
					<PerformanceID>+21 Strange Title</PerformanceID>
					<Seats>
						<Seat>
							<Row>0</Row>
							<Col>1</Col>
								<Section>All</Section>
						</Seat>
					</Seats>
				</Performance>
			</Performances>
		</Packet>
	</Data>
</Request>
```

If there are multiple seats to hold, just add the appropriate <Seat></Seat> information.
Performances can also be identified using PerformanceID.

| Node | Child Node | Type | Required | Description <Seats> |
|---|---|---|---|---|
| Seats | Seat | Sub Node | Yes | |

| Node | Child Node | Type | Required | Description <Seat> |
|---|---|---|---|---|
| Row | | Integer | Yes | Row number |
| Col | | Integer | Yes | Column number |
| Section | | String | Yes | Section name |

**Reserved Seating – Hold Seats – Sample Response:**

```
<Response>
	<Code>0</Code>
	<ResponseText>OK</ResponseText>
	<TransactionID>80A6F0F0-6747-4717-BF27-E2593E61818C</TransactionID>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Code | | Integer | Yes | 0 = success, or an Error Code |
| ResponseText | | String | Yes | OK |
| TransactionID | | String | Yes | Used to complete transaction or release seats |

**Reserved Seating – Release Seats – Sample Request:**

```
<Request>
	<Version>1</Version>
	<Command>HoldSeats</Command>
	<Data>
		<Packet>
			<Command>Release</Command>
			<TransactionID>18EEF867-F72E-4B47-9C3E-
96CE026475E1</TransactionID>
		</Packet>
	</Data>
</Request>
```

| Node | Child Node | Type | Required | Description <Request> |
|---|---|---|---|---|
| Command | | String | Yes | HoldSeats |
| Data | Packet | Sub Node | Yes | |
| Command | | String | Yes | Release |
| TransactionID | | String | Yes | TransactionID from Hold request |

**Reserved Seating – Release Seats – Sample Response:**

```
<Response>
	<Version>1</Version>
	<Code>0</Code>
	<ResponseText>OK</ResponseText>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Code | | String | Yes | 0 = success, or an Error Code |
| ResponseText | | String | Yes | OK |

# Ticket Purchasing – Reserved Seating

### Ticket Purchase (Reserved) – Title, ShowTime, and AuditoriumID - Sample Request:

```
<Request>
 <Version>1</Version>
 <Command>Buy</Command>
 <Data>
  <Packet>
   <PurchaseTitles>
    <PurchaseTitle>
     <Title>Casino Royale</Title>
     <ShowTime>201012132335</ShowTime>
     <Auditorium>1</Auditorium>
     <Tickets>
      <Ticket>
       <Amount>1</Amount>
       <TypeCode>1|1</TypeCode>
       <HoldSeatTransactionID>4022547A-9FB1-4D24-956A-D98AEB482BDC</HoldSeatTransactionID>
       <Seat>
        <Row>7</Row>
        <Col>9</Col>
       </Seat>
       <Section>1|1</Section>
      </Ticket>
      <Ticket>
       <Amount>1</Amount>
       <TypeCode>1|1</TypeCode>
       <HoldSeatTransactionID>4022547A-9FB1-4D24-956A-D98AEB482BDC</HoldSeatTransactionID>
       <Seat>
        <Row>7</Row>
        <Col>10</Col>
       </Seat>
       <Section>1|1</Section>
      </Ticket>
     </Tickets>
    </PurchaseTitle>
   </PurchaseTitles>
   <Fees>
    <TicketFee>0</TicketFee>
    <TransactionFee>2</TransactionFee>
    <Adjust>0</Adjust>
   </Fees>
   <Payments>
    <Payment>
     <Type>CreditCard</Type>
     <Number>5499990123456781</Number>
     <Expiration>0114</Expiration>
     <AvsStreet>4 Baum</AvsStreet>
     <AvsPostal>30329</AvsPostal>
     <CID>123</CID>
     <NameOnCard>RTS</NameOnCard>
     <ChargeAmount>3</ChargeAmount>
    </Payment>
   </Payments>
  </Packet>
 </Data>
</Request>
```

### Ticket Purchase (Reserved) – PerformanceID - Sample Request:

```
<Request>
 <Version>1</Version>
 <Command>Buy</Command>
 <Data>
  <Packet>
   <PurchaseTitles>
    <PurchaseTitle>
     <PerformanceID>1005233501</PerformanceID>
     <Tickets>
      <Ticket>
       <Amount>1</Amount>
       <TypeCode>1|1</TypeCode>
       <HoldSeatTransactionID>4022547A-9FB1-4D24-956A-D98AEB482BDC</HoldSeatTransactionID>
           <Seat>
        <Row>7</Row>
        <Col>9</Col>
       </Seat>
       <Section>1|1</Section>
      </Ticket>
      <Ticket>
       <Amount>1</Amount>
       <TypeCode>1|1</TypeCode>
       <HoldSeatTransactionID>4022547A-9FB1-4D24-956A-D98AEB482BDC</HoldSeatTransactionID>
       <Seat>
        <Row>7</Row>
        <Col>10</Col>
       </Seat>
       <Section>1|1</Section>
      </Ticket>
     </Tickets>
    </PurchaseTitle>
   </PurchaseTitles>
   <Fees>
    <TicketFee>0</TicketFee>
    <TransactionFee>2</TransactionFee>
    <Adjust>0</Adjust>
   </Fees>
   <Payments>
    <Payment>
     <Type>CreditCard</Type>
     <Number>5499990123456781</Number>
     <Expiration>0114</Expiration>
     <AvsStreet>4 Baum</AvsStreet>
     <AvsPostal>30329</AvsPostal>
     <CID>123</CID>
     <NameOnCard>RTS</NameOnCard>
     <ChargeAmount>3</ChargeAmount>
    </Payment>
   </Payments>
  </Packet>
 </Data>
</Request>
```

### Ticket Purchase (Reserved) – MtFilmCode, ShowTime, and PerformanceID - Sample Request:

```
<Request>
 <Version>1</Version>
 <Command>Buy</Command>
 <Data>
  <Packet>
   <PurchaseTitles>
    <PurchaseTitle>
     <PerformanceID>1005233501</PerformanceID>
     <ShowTime>201010052335</ShowTime>
     <MtFilmCode>20516</MtFilmCode>
     <Tickets>
      <Ticket>
       <Amount>1</Amount>
       <TypeCode>1|1</TypeCode>
       <HoldSeatTransactionID>4022547A-9FB1-4D24-956A-D98AEB482BDC</HoldSeatTransactionID>
       <Seat>
        <Row>7</Row>
        <Col>9</Col>
       </Seat>
       <Section>1|1</Section>
      </Ticket>
      <Ticket>
       <Amount>1</Amount>
       <TypeCode>1|1</TypeCode>
       <HoldSeatTransactionID>4022547A-9FB1-4D24-956A-D98AEB482BDC</HoldSeatTransactionID>
       <Seat>
        <Row>7</Row>
        <Col>10</Col>
       </Seat>
       <Section>1|1</Section>
      </Ticket>
     </Tickets>
    </PurchaseTitle>
   </PurchaseTitles>
   <Fees>
    <TicketFee>0</TicketFee>
    <TransactionFee>2</TransactionFee>
    <Adjust>0</Adjust>
   </Fees>
   <Payments>
    <Payment>
     <Type>CreditCard</Type>
     <Number>5499990123456781</Number>
     <Expiration>0114</Expiration>
     <AvsStreet>4 Baum</AvsStreet>
     <AvsPostal>303529</AvsPostal>
     <CID>123</CID>
     <NameOnCard>RTS</NameOnCard>
     <ChargeAmount>3</ChargeAmount>
    </Payment>
   </Payments>
  </Packet>
 </Data>
</Request>
```

**Ticket Purchase (Reserved) - Sample Response:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<Code>0</Code>
			<ResponseText>OK</ResponseText>
			<TransactionID>1524291</TransactionID>
			<Pickups>
				<Pickup>
					<TicketDay>20101213</TicketDay>
					<PickupNumber/>
					<BarCodes>
						<BarCode>
							<Type>Transaction</Type>
							<CodeType>UPC</CodeType>
							<BarCodeData>525524291791</BarCodeData>
						</BarCode>
					</BarCodes>
				</Pickup>
			</Pickups>
		</Response>
	</Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Code | | Integer | Yes | -1 |
| Packet | Response | Sub Node | Yes | |
| ResponseText | | String | Yes | OK |
| TransactionID | | String | Yes | Transaction receipt number |
| Pickups | Pickup | Sub Node | Yes | |
| TicketDay | | String | Yes | Date of performance (yyyyMMddhhmm) |
| PickupNumber | | String | Yes | Not currently used |
| BarCodes | BarCode | Sub Node | Yes | |
| Type | | String | Yes | Transaction |
| CodeType | | String | Yes | The bar code type: UPC |
| BarCodeData | | String | Yes | The confirmation number to be encoded. BAR CODE IMAGE IS NOT PROVIDED BY THE POS. |

# Concession Prices and Sales

A list of concession items, Per Cap settings, and prices can be downloaded in XML format, from the theatre, using the following URL:

https://<Theatre RTN Number>.formovietickets.com:2235/concessionprices.xml

When selling concession sales it may be necessary to include sales tax in the <ChargeAmount> tag. Insert additional <PurchaseItem> tags if needed.

**Concession Sales – SalesTaxCheck – Sample Request:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<SalesTaxCheck>1</SalesTaxCheck>
			<PurchaseItems>
				<PurchaseItem>
					<ItemName>Candy Item</ItemName>
					<Amount>1</Amount>
				</PurchaseItem>
			</PurchaseItems>
		</Packet>
	</Data>
</Request>
```

**Concession Sales – SalesTaxCheck – Sample Response:**

```
<Response>
	<Status>0</Status>
	<SalesTax>0.24</SalesTax>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Status | | Integer | Yes | 0 = success, or an Error Code |
| SalesTax | | Decimal | Yes | The amount of sales tax on the items in the Request. |

**Concession Sales – Purchase – Sample Request:**

```
<Request>
	<Version>1</Version>
	<Command>Buy</Command>
	<Data>
		<Packet>
			<PurchaseItems>
				<PurchaseItem>
					<ItemName>Candy Item</ItemName>
					<Amount>1</Amount>
				</PurchaseItem>
			</PurchaseItems>
			<Fees>
				<TicketFee>0</TicketFee>
				<TransactionFee>0</TransactionFee>
				<Adjust>0</Adjust>
			</Fees>
			<Payments>
				<Payment>
					<Type>CreditCard</Type>
					<Number>5499990123456781</Number>
					<Expiration>0114</Expiration>
					<AvsStreet>4 Baum</AvsStreet>
					<AvsPostal>30329</AvsPostal>
					<CID>123</CID>
					<NameOnCard>RTS</NameOnCard>
					<ChargeAmount>2.00</ChargeAmount>
				</Payment>
			</Payments>
		</Packet>
	</Data>
</Request>
```

**Concession Sales – Purchase – Sample Response:**

```
<Response>
	<Version>1</Version>
	<Code>-1</Code>
	<Packet>
		<Response>
			<Code>0</Code>
			<ResponseText>OK</ResponseText>
			<TransactionID>1476432</TransactionID>
		</Response>
	</Packet>
</Response>
```

| Node | Child Node | Type | Required | Description <Response> |
|---|---|---|---|---|
| Version | | Integer | Yes | 1 |
| Code | | Integer | Yes | -1 |
| Packet | Response | Sub Node | Yes | |
| Code | | String | Yes | 0 = success, or an Error Code |
| ResponseText | | String | Yes | OK |
| TransactionID | | String | Yes | Confirmation for purchase |

# Error Codes

| Error Code | Error Description |
|---|---|
| 100 | No data packet was decrypted (possible encryption error) |
| 101 | No <Payment> nodes specified |
| 102 | Gift request received, but user does not have rights to sell gifts cards |
| 103 | Films request received, but user does not have rights to sell tickets |
| 104 | No films, gifts, or items in request |
| 106 | Invalid PerformanceID format |
| 107 | Could not find film for PerformanceID |
| 108 | No tickets for film in request |
| 110 | Auditorium is oversold |
| 111 | Charge amount does not match calculated amount by POS |
| 113 | Bad username or password |
| 114 | No gift card prefixes are configured |
| 115 | No gift purchase amount in request |
| 116 | POS could not allocate gift certificate number |
| 117 | Can't add points to a non registered card |
| 118 | Invalid gift card number |
| 119 | Unable to create HostedCheckout PaymentId |
| 120 | Invalid ProcessCompletePostData Format |
| 121 | Invalid HostedCheckout Processor |
| 122 | HostedCheckout Transaction Expired |
| 123 | MPS Payment Not Valid |
| 124 | HostedCheckout Purchase Failed |
| 125 | Invalid HostedCheckout PaymentId |
| 126 | Multiple Payment Validations Error |
| 127 | Charge Amount is Neg |
| 128 | Ticket fee item not configured |
| 129 | Extra fee item not configured |
| 130 | Adjust item not configured |
| 131 | Unknown Ticket Class In Request |
| 132 | Ticket class in not enabled for internet ticketing |
| 133 | Zero Priced Ticket in Request |
| 134 | Payments are specified during check sales tax request |
| 135 | No PickupNumber specified |
| 136 | Refund, Reverse, or Redeem error (see message for specific details) |

| | |
|---|---|
| 137 | Performance is no longer scheduled (or not a reserved seating show) |
| 138 | Non-secure payment on tokenized account |
| 139 | Ticket not available for performance |
| 140 | Reserved ticket purchase error (see message for specific details) |
| 141 | Transaction does not balance error |
| 142 | Reserved ticket purchase error (see message for specific details) |
| 143 | Malformed gift purchase packet (see message for specific details) |
| 144 | Unable to validate GiftCardPIN |
| | |
| 500 | POS could not allocate cash register control (possibly server too busy) |
| | |
| 700 | Unknown error during sale |
| 701 | Not enough money on gift card |
| 702 | Invalid credit card number/expiration, or card declined |
| 703 | Invalid performance |
| 704 | Ticket type is disabled |
| 705 | Ticket serial number file is invalid |
| 706 | Concession item not setup – check ticket/concession link items |
| 707 | Reserved seat sale failed – check the seating chart |
| | |
| 5000 | Exception Error (see message for details). |
| | |
| 9000 | POS Not Licensed |
| 9001 | Packet Parse Error (see message for extra details) |

# Reserved Seating Codes

| Seat Code | Seat Description |
|---|---|
| 0 | Normal Seat |
| 1 | Handicapped Seat |
| 2 | Aisle (the aisle, NOT an aisle seat) |
| 3 | House Seat |
| 4 | Companion Seat |
| 5 | Pillar |
| 6 | Table |
| 7 | Beanbag |
| 8 | Loveseat |
| 9 | Rocker |
| 10 | Recliner |

# Film and Show Bit Field Values

| Info1 | Setting |
|---|---|
| 1 | RTN Display |
| 2 | RTN Sell |
| 4 | Kiosk Display |
| 8 | No Passes |
| 16 | Dolby Digital |
| 32 | THX |
| 64 | DLP |
| 128 | Dubbed in English |
| 256 | Subtitled in English |
| 512 | DTS |
| 1024 | No Discounts |
| 2048 | Stadium Seating |
| 4096 | 3D |
| 8192 | 16MM |
| 16384 | 35MM |

| Info2 | Setting |
|---|---|
| 1 | Movietickets.com Display |
| 2 | Movietickets.com Sell |
| 4 | Rentrak Transfer |
| 8 | IMAX |
| 16 | Flat |
| 32 | Digital |
| 64 | Descriptive Video Service |
| 128 | Subtitled in French |
| 256 | Dubbed in French |
| 512 | French Language Film |
| 1024 | 70MM |
| 2048 | Open Caption |
| 4096 | Closed Caption |
| 8192 | Special Advanced Screening |
| 16384 | Adults Only |

| Info3 | Setting |
|---|---|
| 1 | Sign Display |
| 2 | NOT USED |
| 4 | NOT USED |
| 8 | Directors Hall |
| 16 | RWC |
| 32 | Reserved Seating |
| 64 | Gallery |
| 128 | Lux Level |
| 256 | Premier |
| 512 | Cine Capri |
| 1024 | Cine Art |
| 2048 | Showcase Art |
| 4096 | Surround Sound |
| 8192 | Scope |
| 16384 | DBOX |
| 32768 | PLF |
| 65536 | Dolby Atmos |
| 131072 | RealD 3D |
| 262144 | Sony 4K |
| 524288 | DTSX |
| 1048576 | Auro 3D |
| 2097152 | Fedelio |
| 4194304 | Captiview |
| 8388608 | Audio Description |
| 16777216 | Hearing Impaired Track |
| 33554432 | Sensory Show |

| Info4 | Setting |
|---|---|
| 1 | Reserved |