Inventory Management

Presenter's Name
Presenter's Title

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Oracle Training Materials – Usage Agreement

Use of this Site ("Site") or Materials constitutes agreement with the following terms and conditions:

1. Oracle Corporation ("Oracle") is pleased to allow its business partner ("Partner") to download and copy the information, documents, and the online training courses (collectively, "Materials") found on this Site. The use of the Materials is restricted to the non-commercial, internal training of the Partner's employees only. The Materials may not be used for training, promotion, or sales to customers or other partners or third parties.

2. All the Materials are trademarks of Oracle and are proprietary information of Oracle. Partner or other third party at no time has any right to resell, redistribute or create derivative works from the Materials.

3. Oracle disclaims any warranties or representations as to the accuracy or completeness of any Materials. Materials are provided "as is" without warranty of any kind, either express or implied, including without limitation warranties of merchantability, fitness for a particular purpose, and non-infringement.

4. Under no circumstances shall Oracle or the Oracle Authorized Delivery Partner be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this Site of Materials. As a condition of use of the Materials, Partner agrees to indemnify Oracle from and against any and all actions, claims, losses, damages, liabilities and expenses (including reasonable attorneys' fees) arising out of Partner's use of the Materials.

5. Reference materials including but not limited to those identified in the Boot Camp manifest can not be redistributed in any format without Oracle written consent.

Agenda

- Introduction to Inventory System
- Inventory Classes and Implementations

Learning Objectives

At the end of this lesson you should be able to:

- Use ATG's inventory system.
- Understand stock levels and thresholds
- Learn about the inventory repository
- Administer the inventory repository from the Dyn Admin Console
- Display inventory on a web page using inventory droplets
- Learn about the inventory classes and implementations
- Use the inventory manager to perform operations on the inventory

Section 1:

Introduction to Inventory System

Inventory System Overview

- The Inventory Framework facilitates querying and inventory management for your sites.
- Interaction with the Inventory System is vital to the various stages of an electronic purchase.
- The Inventory System provides a complete set of methods to support inventory handling.
- The Inventory System allows developers to:
  - Add inventory to SKUs when new inventory arrives.
  - Remove items from inventory.
  - Notify the store of a customer's intent to purchase an item that is not currently in stock (backorder).
  - Notify the store of a customer's intent to purchase an item that has never been in stock (preorder).

What is Inventory?

- Inventory, in the context of ATG, is a list and quantity of SKUs available in stock, pre-orderable stock, or back-orderable stock by a business. It is usually also called stock.
- The eCommerce System interacts with the Inventory Management as follows:
  - When a customer puts an item in the cart, the eCommerce System queries the Inventory Management System to figure out if the order can be fulfilled and when it can be fulfilled (backorder, preorder).
  - When an order is placed, the eCommerce system informs the Inventory Management System that a purchase has occurred, so that the stock levels can be decremented.
  - When an item that is out of stock is back in stock or the stock levels change for any reason (error), Inventory Management informs the eCommerce System of the change.

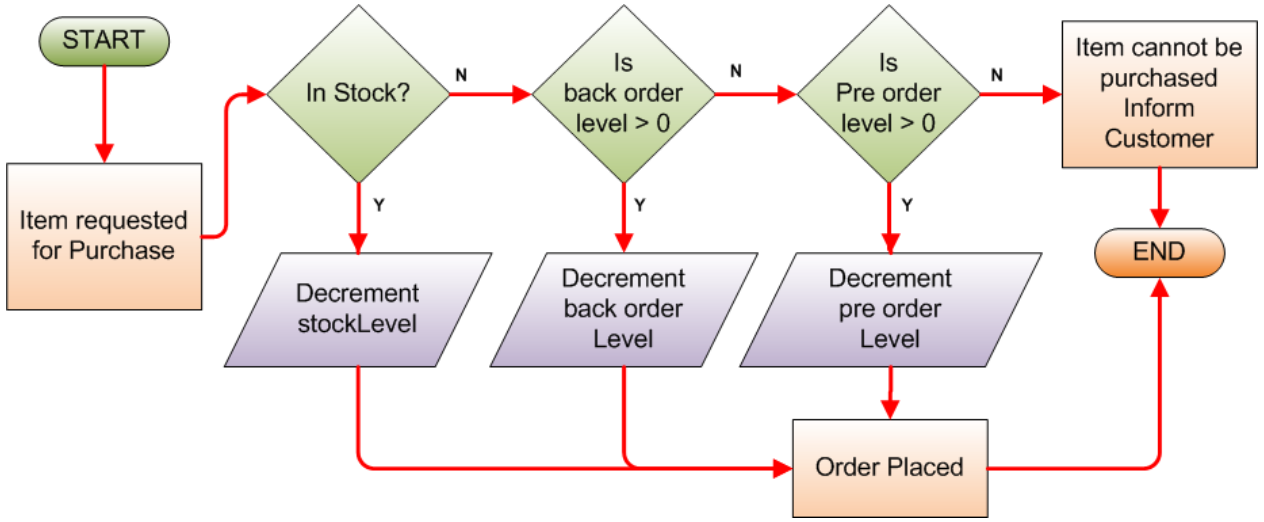
STOCK LEVELS AND THRESHOLDS

Understanding Stock Levels (1)

- **In stock**
  - Items that can be purchased currently on the site.
  - This is stored in **`stockLevel`** field.
- **preorder**
  - Inventory for items that are **not released** yet.
  - Web sites allow customers to place an order for unreleased products with the expectation of fulfilling it at a pre-specified time called the release date.
  - **`preorderLevel`** is the number of items that can be preordered.
- **Backorder**
  - Inventory that you currently do not have but are expecting to receive shortly.
  - Websites allow customers to still place the order for the item with an expectation that it will be fulfilled shortly.
  - **`backorderLevel`** is the number of items that may be backordered.

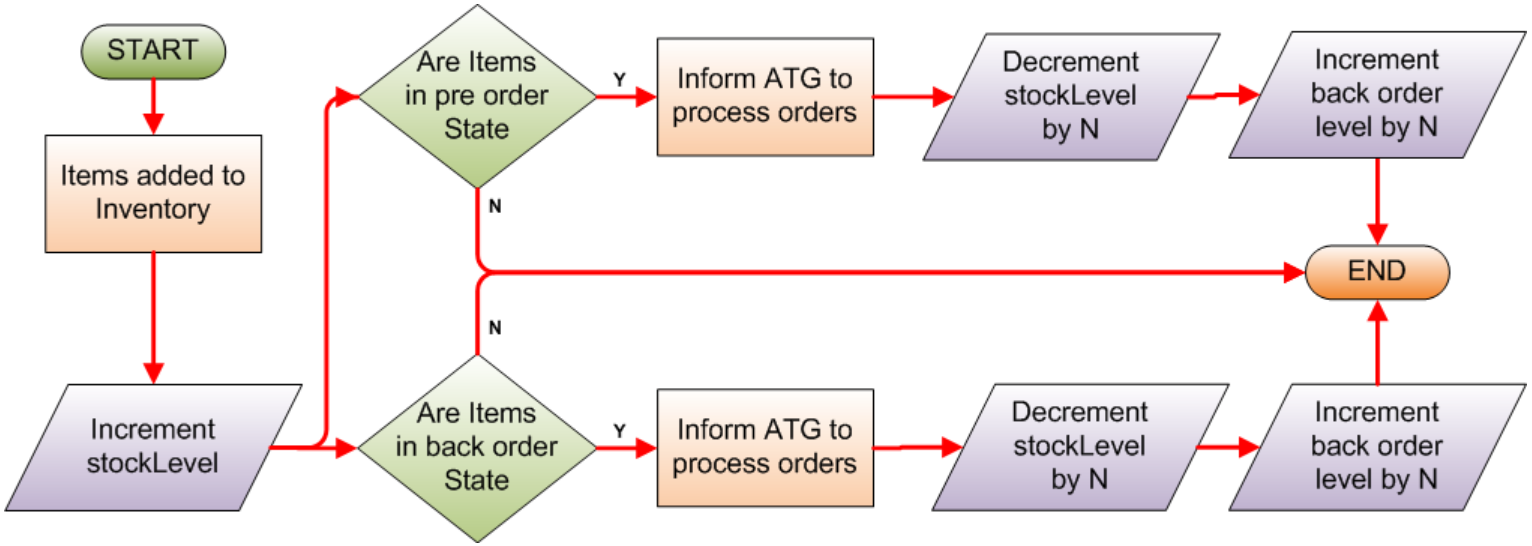
Understanding Stock Levels (2)

- When an item is ordered, ATG first tries to allocate the order from stock.
- If not, the backorderLevel and preorderLevel are checked and allocation is made.
- Typically, the SKU is back-orderable or pre-orderable, but not both.



Understanding Stock Level (3)

- When new inventory is added, the stock level is increased.
- A message is sent to the fulfillment systems to clear out any preorders and backorders.
- Note that the backorder and preorder levels are increased as much as possible to its original value.



Inventory Thresholds

- For each stock level, there are thresholds associated:
  - **`stockThreshold`** for stockLevel,
  - **`preorderThreshold`** for preorderLevel,
  - **`backorderThreshold`** for backorderLevel.
- When the stock level reaches the threshold value, an InventoryThresholdReached event is generated.
- Administrators and developers can make use of this to take automated and manual actions.
- A common use is also to prevent customers from putting the item in cart when threshold is reached.
  - Customers who already have the SKU in cart can check out.
  - Prevents orders from going to a HOLD status.
  - This implementation needs custom coding.

INVENTORY REPOSITORY

Inventory Repository

- ATG Commerce comes with the inventory repository which stores the stock level, availability information, and other inventory data.
- The inventory repository is stored in a separate repository from the product catalog.
- The API methods in the **RepositoryInventoryManager** take the SKU ID and use it to look up the inventory in the Inventory repository.
- The inventory definition is stored in atg/commerce/inventory/inventory.xml.
- Like any other repository, you can extend the inventory repository to add your own properties.

Inventory Repository Item Properties

- The following are key inventory repository item properties:
  - **`displayName`** and **`description`**,
  - **`catalogRefId`** is the SKU ID in the product catalog,
  - **`stockLevel`**, **`backorderLevel`**, and **`preorderLevel`**,
  - **`stockThreshold`**, **`backorderThreshold`**, and **`preorderThreshold`**,
  - **`availabilityDate`** and **`availabilityStatus`**.
- AvailabilityStatus can be:
  - INSTOCK
  - OUTOFSTOCK
  - DISCONTINUED
  - PREORDERABLE
  - BACKORDERABLE
  - DERIVED
- DERIVED is the default availability status.

ADMINISTERING INVENTORY

Administrator Activities Using the Inventory System

- Using the Dynamo Administration Center (DYN Admin), administrators can interact with the inventory manager.
- Administrator can:
  - **View** inventory levels for specific items for purchase, backorder, or preorder.
  - **Update** inventory levels for stock, preorder, and backorder as well as threshold values. They can also update availability status and availability date.
  - **Notify** systems by sending an inventory update notification.
- The update inventory interface accepts the property to update, the value, and if you wish to increment or decrement to set a value.
- This is for exceptional use case only. The normal mechanism should be a feed based interaction.

Example of Using the Console

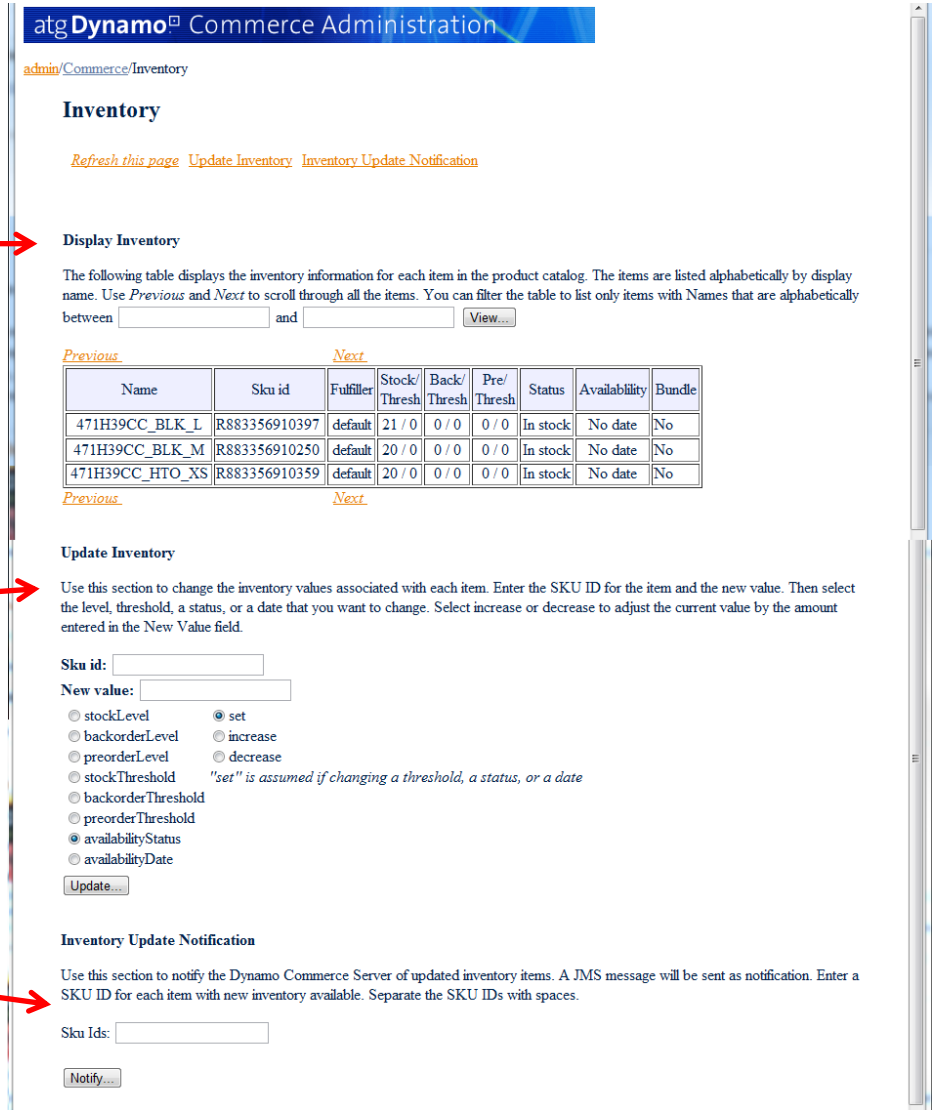
- Joe has discovered a serious issue in the warehouse. 100 dream weaver shirts that were supposed to be stock have been misplaced.
- Procurement confirms that another shipment of 100 more of these shirts are on their way and are expected to arrive the next Monday.
- Joe decides to:
  - Decrease the stockLevel to zero.
  - Increase the backorderLevel to 100.
  - Set availabilityDate to next Monday.
  - Send out an inventory update notification so that the relevant processes run.
- When new stock arrives, the inventory system will send a message to eCommerce site and everything is back to normal.

The Dyn Admin Console

Display Inventory

Update Inventory

Send Inventory Notification



INVENTORY LOOKUP DROPLET

InventoryLookup Droplet

- The InventoryLookup droplet gives access to inventory information in a JSP.
- Input parameters:
  - **`itemId`** – The itemId is the catalogRefId of the product catalog SKU whose inventory information will be retrieved.
  - **`useCache`** (Boolean) – If set to true, cached data will be retrieved.
- Open Parameter:
  - **`output`**
- Output Parameters:
  - **`availabilityDate`**, **`availabilityStatus`**, and **`availabilityStatusMsg`**,
  - **`stockLevel`**, **`preoderLevel`**, and **`backorderLevel`**
  - **`stockThreshold`**, **`preorderThreshold`**, and **`backorderThreshold`**.

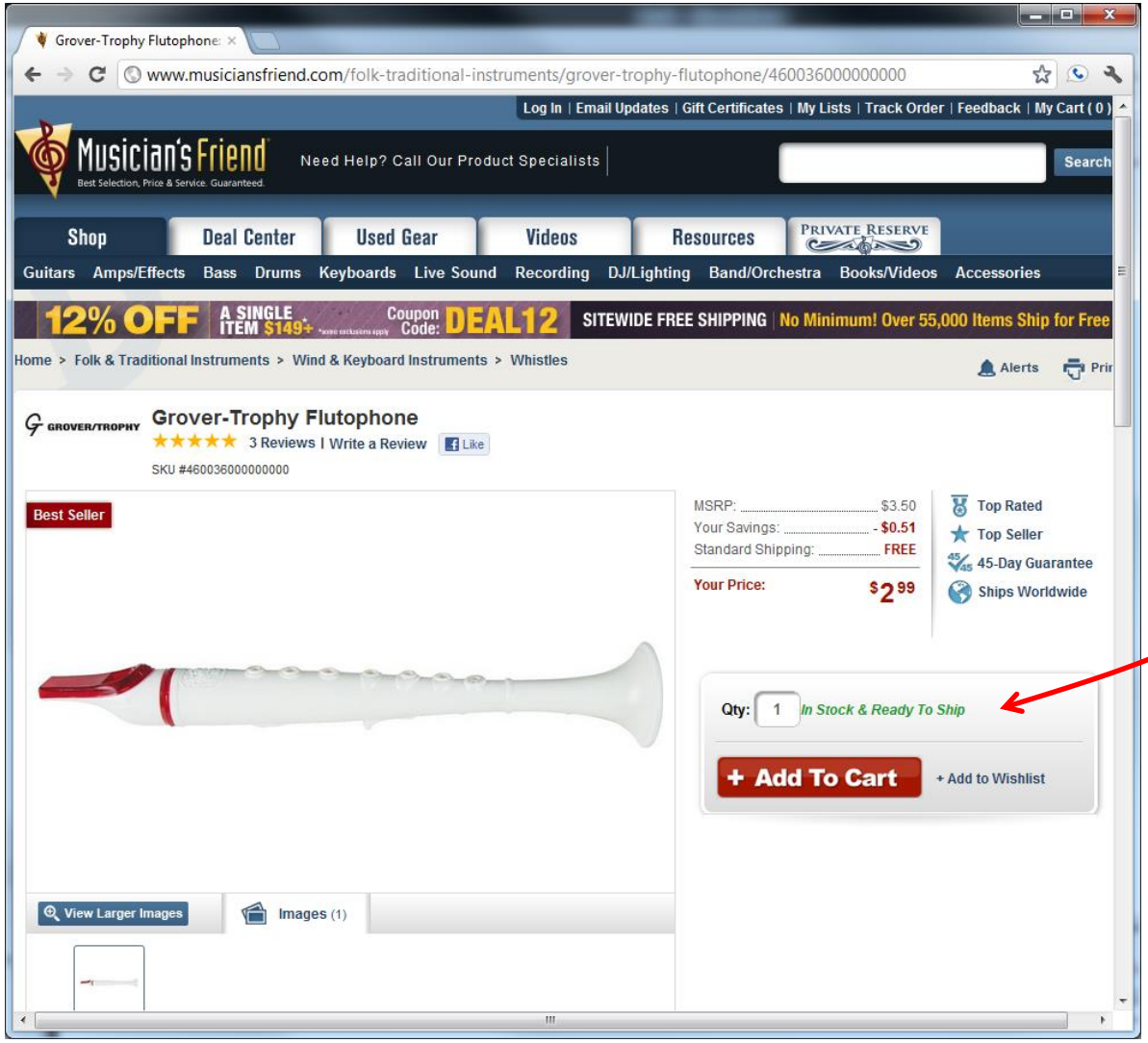
InventoryLookup Example

- The following code is an example of using the InventoryLookup droplet:

```
<dsp:droplet name="/atg/commerce/inventory/InventoryLookup">
  <dsp:param param="link.item.repositoryId" name="itemId"/>
  <dsp:param value="true" name="useCache"/>
  <dsp:oparam name="output">
     This item is
      <dsp:valueof
           param="inventoryInfo.availabilityStatusMsg"/>
     <br />There are
      <dsp:valueof param="inventoryInfo.stockLevel"/>
     left in the inventory.<br>
 </dsp:oparam>
</dsp:droplet>
```

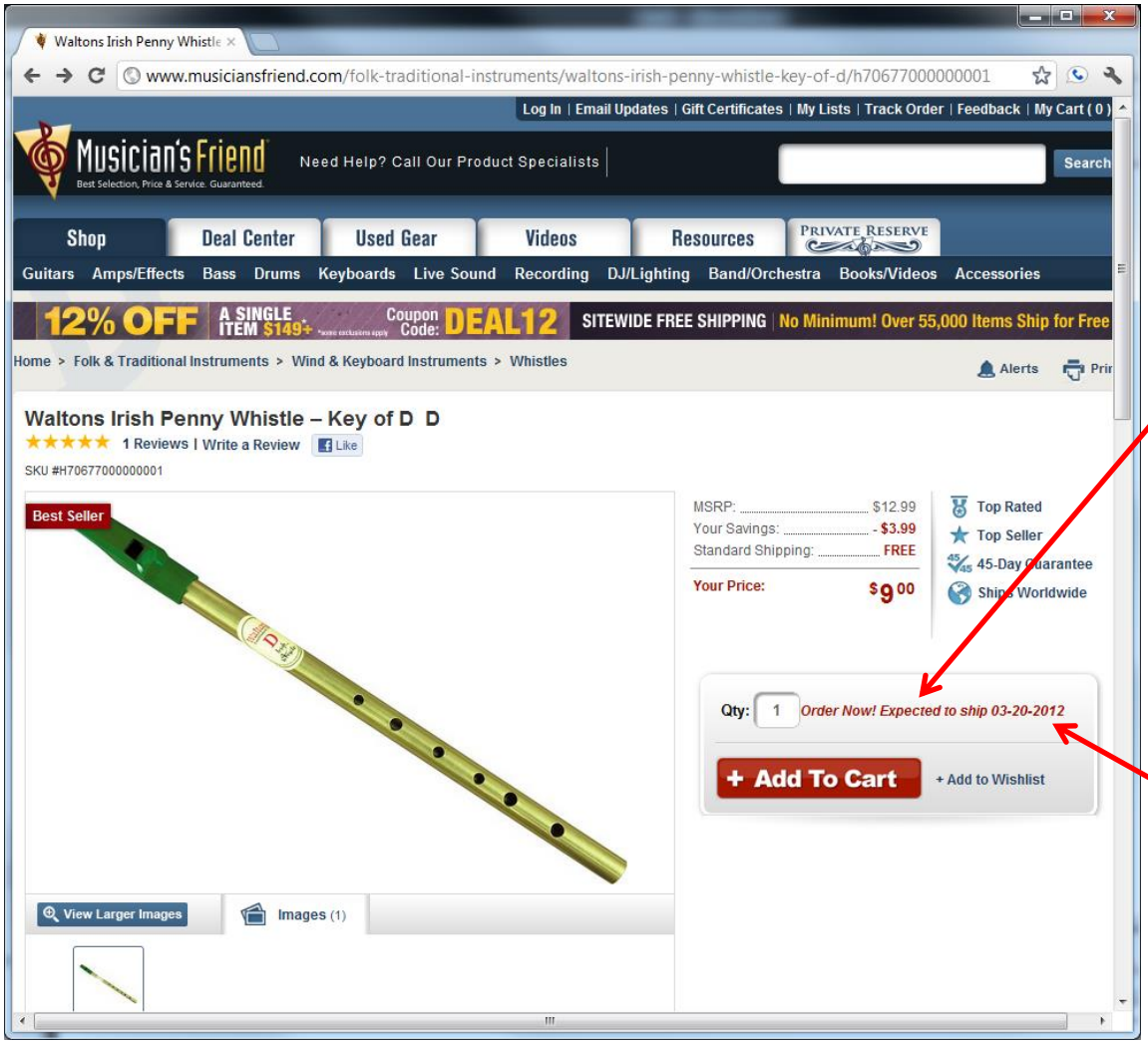
EXAMPLE OF INVENTORY USAGE

In Stock Check



In Stock Message

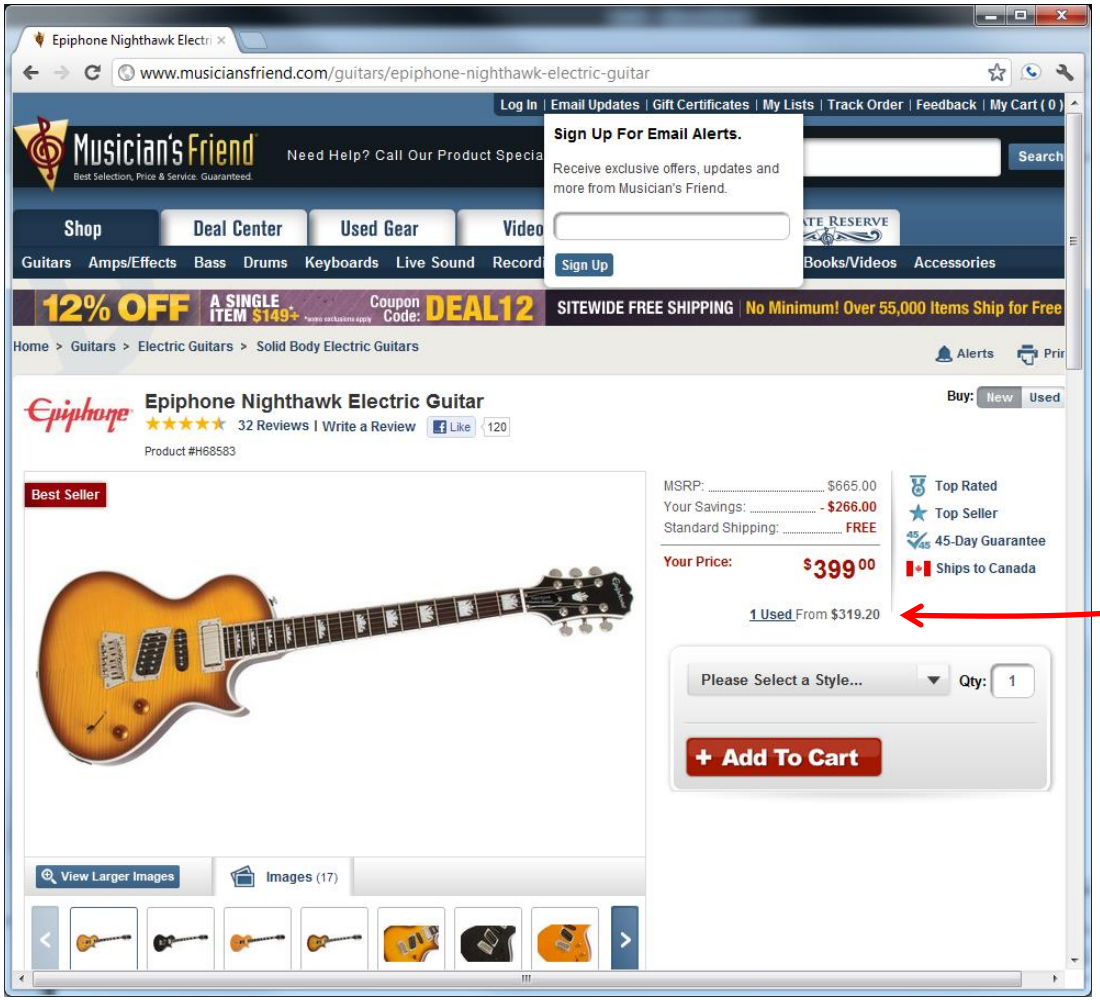
Availability Status and Date



Availability Status

Availability Date

Additional Inventory Types



Used Inventory

Section 1 
Check Your Understanding

What are the types of output parameters that the InventoryLookDroplet exposes?

Answer:

It exposes output parameters like availability information, stock levels, thresholds, etc.

Section 1 
Check Your Understanding

What are the types of activities that can be performed in the Inventory section of the Dynamo Admin console.

Answer:

You can view and update the inventory and send update notifications.

Section 1 ✔
Check Your Understanding

Name some states that the availability status can be set to.

Answer:

INSTOCK, OUTOFSTOCK, DISCONTINUED, PREORDERABLE, BACKORDERABLE, DERIVED.

Section 1 
Check Your Understanding

What is a stock level threshold?

Answer:

The stock level threshold for in stock, backordered, or preordered level is a stock level that when reached causes the InventoryThresholdReached event to be generated.

Section 1
Check Your Understanding

User Tommy places an order for 10 shirts that are backordered. The initial backorder inventory was 100 which falls to 90 after the order. When the stock arrives for the shirt and the order is fulfilled, what is the final backorder level?

Answer:

It is 100. The fulfillment system will invoke purchaseFromBackorder which will cause the backorder inventory level to go back up to 100.

Summary

- The Inventory Framework facilitates querying and inventory management for your sites.
- ATG supports three inventory levels by default: in stock, preorder, and backorder.
- It also supports threshold levels for each of the stock levels.
- ATG Commerce comes with the Inventory repository which stores the stock level, availability information, and other inventory data.
- Administrators can use the Dyn Admin interface to administer the repository.
- Inventory Lookup droplet allows querying the inventory to be displayed to the customer.

Section 2:

Inventory Classes and Implementations

INVENTORY CLASSES

Inventory Classes

- The inventory API includes the following:
  - **InventoryManager** interface is a public interface that contains all the inventory system functionality.
  - **InventoryException** class is thrown by most of the methods.
- The InventoryManager interface is implemented by:
  - **AbstractInventoryManagerImpl** removes all methods except purchase for simple implementations.
  - **NoInventoryManager** implements a placeholder that always returns IN STOCK response.
  - **RepositoryInventoryManager** is an implementation that is based on SQL repositories.
  - **CachingInventoryManager** provides caching and delegates the calls to other inventory managers.
  - **LocalizingInventoryManager** provides a locale-based (multiple) inventory functionality.

Inventory Manager

- InventoryManager is the public interface that contains all of the inventory system functionality.
- To use the inventory management system, you can:
  - Use the provided InventoryManager implementations,
  - Extend the provided implementations,
  - Implement the interface and write your own manager.
- InventoryManager function typically returns a status code:
  - INVENTORY_STATUS_SUCCEED (0)
  - INVENTORY_STATUS_FAIL (1)
  - INVENTORY_STATUS_INSUFFICIENT_SUPPLY (2)
  - INVENTORY_STATUS_ITEM_NOT_FOUND (3)
- In addition, they throw InventoryException and sometimes MissingInventoryItemException.

Inventory Manager Methods - Purchase

- Three methods are available for purchase:

Method	Description
`purchase`	• Called by the Inventory system when the user completes a purchase. • Decreases the inventory (stocklevel) of the item. • Returns unsuccessfully if inventory is not available. • This method is required by any implementation of the inventory system.
`preorder`	• Called by Inventory system when the user completes a purchase of a preordered item. • Decreases the preorderLevel of the item. • Returns unsuccessfully if the preorderLevel is lower than the quantity being preordered.
`backorder`	• Called by Inventory system when the user completes a purchase of a backordered item. • Decreases the backorderLevel of the item. • Returns unsuccessfully if the backorderLevel is lower than the quantity being backordered.

Inventory Manager Methods – Purchasing Off of Pre and Backorder

- Two methods are available to purchase off of pre and backorder.
- They perform the same function but increase the appropriate stock level.
- Assume you had 10 backorder items.
  - User places an order for 10 reducing backorder level to 0.
  - When fulfillment system calls purchaseOffBackOrder later when inventory is available, inventory is reduced by 10 and backOrderLevel is increased back to 10.

Method	Description
`purchaseOffBackorder`	• Does the same thing as purchase and increases the backorderLevel.
`purchaseOffPreorder`	• Does the same thing as purchase and increases the preorderLevel.

Set, Increase, and Decrease Stock Level Methods

- For each level (stock, backorder, and preorder) there are 3 methods to set, increase, and decrease the level.
- In addition, there is a method to set the threshold.

Stock	Backorder	Preorder
setStockLevel	setBackorderLevel	setPreorderLevel
increaseStockLevel	increaseBackorderLevel	increasePreorderLevel
decreaseStockLevel	decreaseBackorderLevel	decreasePreorderLevel
setStockThreshold	setBackorderThreshold	setPreorderThreshold

- Set sets it to a fixed number. Increase and decrease do the operation specified on the current level.

Inventory Manager - Query Methods

- The query methods return the relevant information being requested.

Method	Description
queryStocklevel	Returns the number of items available for purchase (stocklevel).
queryBackorderLevel	Returns the number of items available for backorder (backorderLevel).
queryPreorderLevel	Returns the number of items available for preorder (preorderLevel).
queryStockThreshold	Returns the stockThreshold.
queryBackorderThreshold	Returns the backorderThreshold.
queryPreorderThreshold	Returns the preorderThreshold.
queryAvailabilityDate	Returns the date when this item will become available.
queryAvailabilityStatus	Determines whether an item is in stock, out of stock, backorderable, or preorderable. Used when determining which method to call : purchase, backorder, or preorder.

Inventory Manager – Availability Status (1)

- Availability status specifies whether an item is in stock, out of stock, backorderable, or preorderable.
- It is used when determining which method to call: **`purchase`**, **`backorder`**, or **`preorder`**.
- There are two methods for availability status:
  - queryAvailabilityStatus,
  - setAvailabilityStatus.
- The states for availability status are:
  - AVAILABILITY_STATUS_IN_STOCK,
  - AVAILABILITY_STATUS_OUT_OF_STOCK,
  - AVAILABILITY_STATUS_PREORDERABLE,
  - AVAILABILITY_STATUS_BACKORDERABLE,
  - AVAILABILITY_STATUS_DERIVED,
  - AVAILABILITY_STATUS_DISCONTINUED.

Inventory Manager – Availability Status (2)

- AVAILABILITY_STATUS_DERIVED causes the queryAvailabilityStatus to calculate the actual status based on the values of the item's stockLevel, backorderLevel, and preorderLevel.
- If the availability status is set to anything other than DERIVED, it will result in that status being returned even if stock level changes.
  - For example, if you have 10 items and set the status to AVAILABILITY_STATUS_IN_STOCK. When the 10 items sell out and make the level 0, the status query will still return the same status AVAILABILITY_STATUS_IN_STOCK.
- AVAILABILITY_STATUS_DERIVED should be used in most cases while dealing with OOTB Inventory Manager implementations.

Inventory Manager – Other Methods

- There are two methods available for availabilityDate:
  - **`queryAvailabilityDate,`**
  - **`setAvailabilityDate.`**
- Availability date is the date on which the item will become available.
- It is used to inform the customer about availability.

- **`inventoryWasUpdated`** is a helper method that is called when a set of items is added to inventory.
- It does not change the state of the inventory.
- It is a convenient way of notifying interested systems of a large update.

Inventory JMS Messages

- The InventoryManager creates JMS messages for the following events:
  - **`InventoryThresholdReached`** is generated when the levels for stock, backorder, and preorder fall below their thresholds. The ID number and the property hit the threshold is included in the message.
  - **`UpdateInventory`** message is sent when stock level increases. The IDs of all items that have stock available and were previously unavailable are included.
- These messages can be captured in Scenario engine or custom event listeners to perform specific business functions.
  - In a simplistic example, an email message can be sent to the merchandiser.

INVENTORY MANAGER IMPLEMENTATIONS

ATG Inventory Manager Implementations

- ATG Commerce includes the following implementations of the InventoryManager out-of-the-box:
  - **AbstractInventoryManagerImpl** removes all methods except purchase for simple implementations.
  - **NoInventoryManager** implements a placeholder that always returns IN STOCK response.
  - **RepositoryInventoryManager** is an implementation that is based on SQL repositories.
  - **CachingInventoryManager** provides caching and delegates the calls to other inventory managers.
  - **LocalizingInventoryManager** provides a locale-based (multiple) inventory functionality.
- You can and most typically will extend the OOTB implementation of the InventoryManager.

AbstractInventoryManagerImpl

- AbstractInventoryManagerImpl is an abstract class.
- It removed all the InventoryManager methods except purchase. This method is abstract.
- It provides a simple way for users to purchase items online, without all the extra features such as backorder, preorders, etc.
- Developers must extend this class and implement the purchase method to use this functionality.

No Inventory Manager

- NoInventoryManager is an implementation of the InventoryManager interface intended as a placeholder.
- It is useful in cases where no Inventory Manager functionality is required, but an inventory manager of some kind is needed for property setting.
- It allows components that require an InventoryManager to function without actually having an inventory system on the backend.
- All methods of this implementation return INVENTORY_STATUS_SUCCEED indicating that all items are in stock indefinitely and infinitely.

Repository Inventory Manager

- RepositoryInventoryManager implements all the methods of the InventoryManager interface.
- By default it uses the SQL repository to store inventory information.
  - The Inventory repository defines one repository item type: an inventory item.
  - Each SKU for which inventory is managed requires a corresponding inventory item to be created in the Inventory repository.
- This inventory manager broadcasts events when levels are at a configurable critical level and when it is notified of updated inventory.
- The repository InventoryManager can then be configured to extract inventory manager information from the third party repository.

Caching Inventory Manager

- Caching inventory manager caches any read-only data for quick display to the site user.
- It is configured in UncachedInventoryManager and a cache.
- The uncachedInventoryManager property should refer to the RepositoryInventoryManager.
- All calls are sent to the uncachedInventoryManager component except the query methods.
- It provides flush methods to clear the cache.
- This inventory manager does not actually perform inventory management. It relies on another implementation to accomplish that part.
- You can use this inventory manager with your own implementation to gain caching capability.

Localizing Inventory Manager

- LocalizingInventoryManager is an implementation of the InventoryManager interface.
- It is used when you want to have more than one set of inventory data and therefore more than one InventoryManager.
- Determines which data/ manager set to use based on your customer's locale.
- When a call is made, this manager calls a second method with the same name but passes in the customer's locale.
- Like the caching inventory manager, it relies on one or more inventory managers to do the actual work.

USING THE INVENTORY MANAGER

Example: Display Item's Availability

```
int availability =
      inventoryManager.queryAvailabilityStatus(SKUId);
Date availabityDate =
      inventoryMangager.queryAvailabilityDate(SKUId);
```

- Query the availabilityStatus and availabilityDate from the manager.
- Typically, you can use the Inventory Lookup droplet to do the same activity.
- If you need to take an action or display status in a Java class, you can use the code snippet above.

Example: Allocating Items for an Order

- The following code snippet shows how to interact with the Inventory Manager while allocating items for an order that has been placed by a customer.

```
String SKUId = "sku123";
long quantity = 5;
// state comes from  Shipping Group Relationship
int itemState = getItemState();
InventoryManager inventory = getInventoryManager();

int status;
if (itemState == INITIAL) // normal case
  status = inventory.purchase(SKUId, quantity);
else if(itemState == BACK_ORDERED)
  status = inventory.purchaseOffBackorder(SKUId, quantity);
else if(itemState == PRE_ORDERED)
  status = inventory.purchaseOffPreorder(SKUId, quantity)
```

Example: Cancelling an Order

- The following code snippet shows how to interact with the Inventory Manager while cancelling an order.

```
String SKUId = "sku123";
long quantity = 5;
// getItemState comes from shipping group relationship
int itemState = getItemState();
InventoryManager inventory = getInventoryManager();

int status;
if(itemState == PENDING_DELIVERY) // normal case
  status = inventory.increaseStockLevel(SKUId, quantity);
else if(itemState == BACK_ORDERED)
  status = inventory.increaseBackorderLevel(SKUId,quantity);
else if(itemState == PRE_ORDERED)
  status = inventory.increasePreorderLevel(SKUId, quantity);
```

Section 1 
Check Your Understanding

The availability status of a SKU is set to AVAILABILITY_STATUS_BACKORDERABLE. When new stock arrives and is added to the inventory, what is the new state of the SKU?

Answer:

It is still AVAILABILITY_STATUS_BACKORDERABLE. You should use AVAILABILITY_STATUS_DERIVED to automatically compute the status based on stock levels.

Section 1 ✔
Check Your Understanding

What are the methods used for increasing stock level?

Answer:

increaseStockLevel, increaseBackorderLevel, and increasePreorderLevel.

Section 1 
Check Your Understanding

What methods does the Caching Inventory Manager handle?

Answer:

It handles query methods and returns cached valued. The rest are delegated to uncachedInventoryManager.

Section 1 
Check Your Understanding

To use the SQL inventory repository which InventoryManager implementation should you configure?

Answer:

You should use the RepositoryInventoryManager implementation.

Section 1 ✔
Check Your Understanding

What are the three methods used for purchasing inventory?

Answer:

purchase, purchaseOffBackorder, and purchaseOffPreorder.

Summary

- **InventoryManager Interface** is a public interface that contains all the inventory system functionality.
- It throws **InventoryException** class from most methods. Some methods also throw MissingInventoryItemException.
- **AbstractInventoryManagerImpl** is an abstract class that removed all the InventoryManager methods except purchase. This method is abstract.
- **NoInventoryManager** is a placeholder implementation that always returns success for inventory queries.
- **RepositoryInventoryManager** is the most commonly used implementation that uses SQL repository to store inventory.
- **CachingInventoryManager** relies on other inventory managers for core functions while providing caching capability for queries.
- **LocalizingInventoryManager** relies on locale to pick one or more other inventory managers to perform the functions.

Q&A

ORACLE IS THE INFORMATION COMPANY


ORACLE®