# Basic Pricing

Presenter’s Name
Presenter’s Title

## Safe Harbor Statement

## Oracle Training Materials – Usage Agreement

## Agenda

- Commerce Pricing Overview
- Using Price Lists
- Displaying Pricing

## Learning Objectives

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

- Understand the different pricing elements in ATG
- List the differences between static and dynamic pricing and know when to use them
- Learn how static and dynamic pricing works
- Describe how pricing services work
- Understand the different pricing models in ATG
- Learn about price lists and volume pricing
- Display static and dynamic pricing for an item on a JSP Page

# Section 1:

# Commerce Pricing Overview

## ATG Commerce Pricing Services

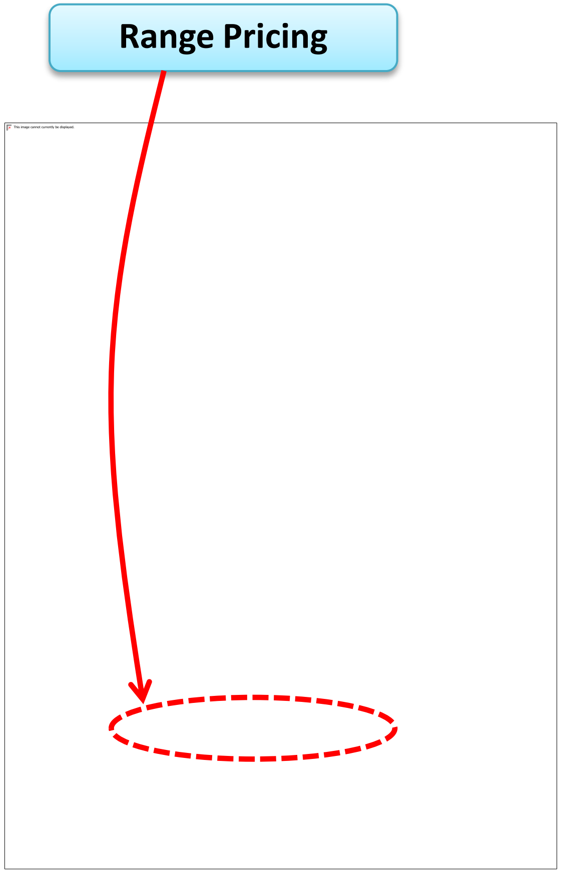
- ATG Commerce pricing services provide a flexible system for personalizing the prices of items.
- You can tailor prices and generate them dynamically on demand.
- Pricing services include a set of standard features that are designed to handle the pricing demands of most websites.
- These services can be extended to handled specific business requirements as needed.

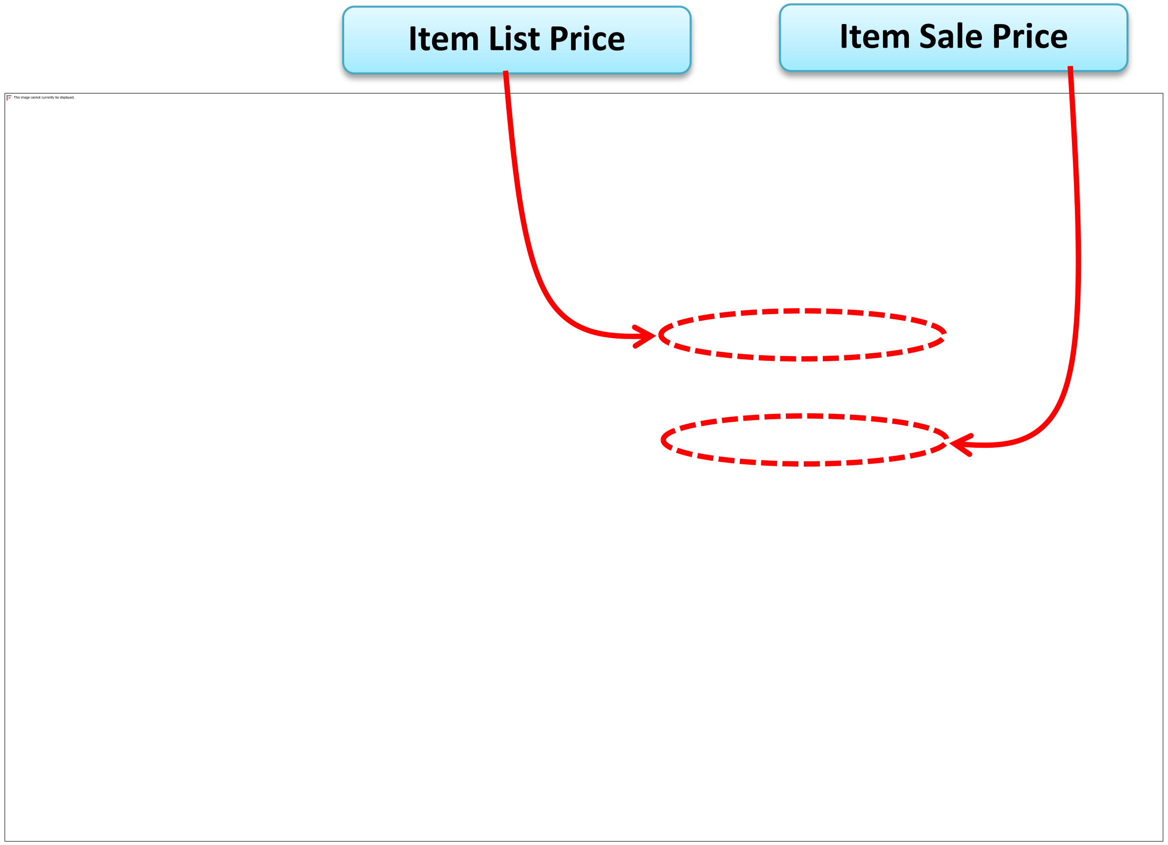
## An Example of Pricing

- Joe Bruin visits an eCommerce website.
- He navigates to the product list page where pricing is presented for a list of products, typically a range of prices.
- He then navigates to the product detail page where he is presented with **SKU prices.**
- He puts a SKU in the cart and a promotion is activated reducing the price and displaying a **dynamic price**.
- In the checkout pages, Joe selects a shipping method and a **shipping price** is added.
- Depending on the state, there is also a **tax price** added.
- The order is then summed up to get an **order price** for the entire order.

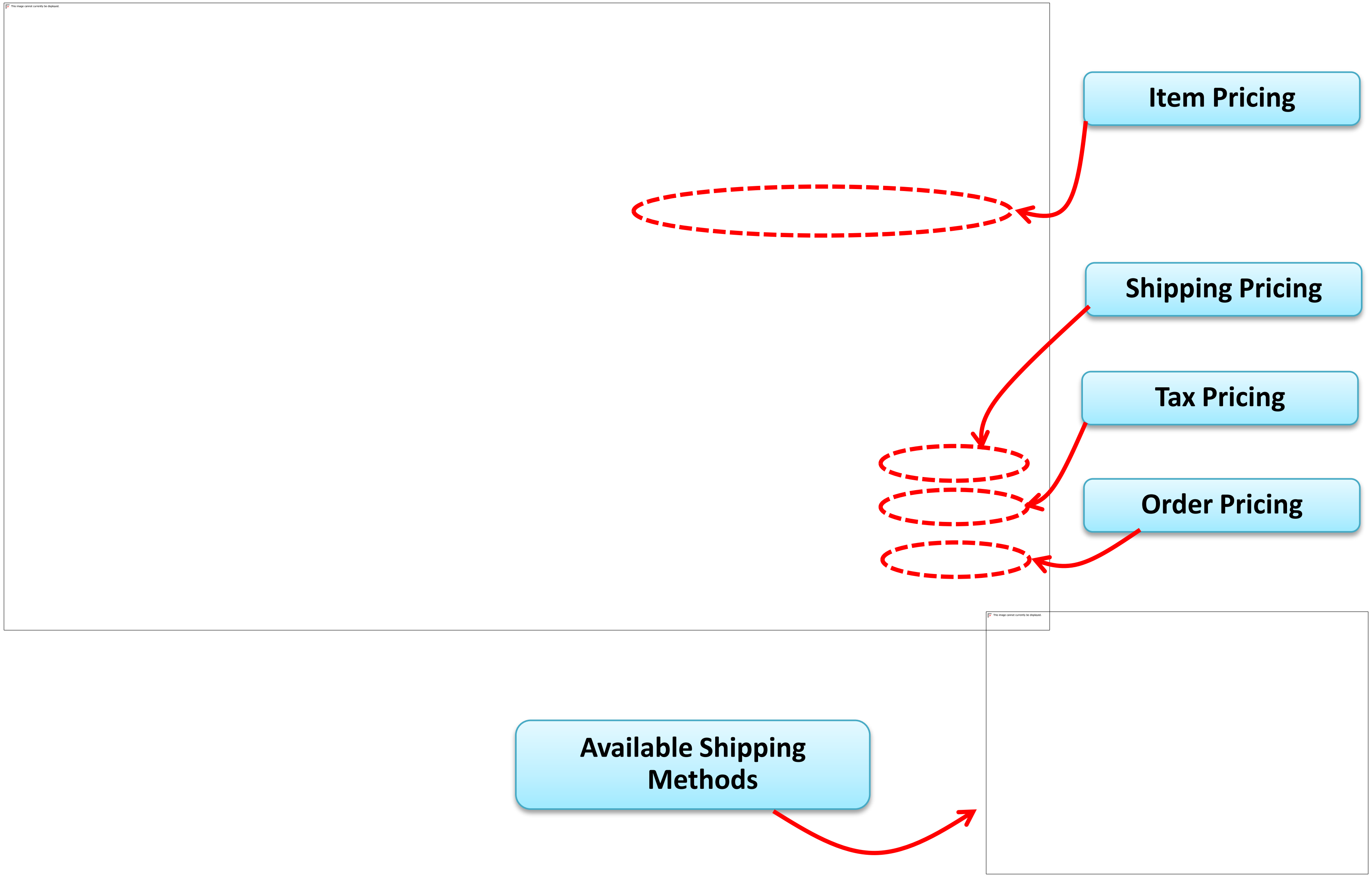
# Example of Pricing Elements (1)

- Example of range pricing is shown on the left.
- Example of list and sale pricing is shown on the right.





# Example of Pricing Elements (2)



## Pricing Elements

- ATG provides support for pricing four commerce objects:
  - Item,
  - Order,
  - Shipping,
  - Tax.
- ATG's pricing mechanism relies on pricing each of the four items in a very similar way.
- Each object has corresponding price holder classes, engines, calculators, qualifiers, and discount types.

## Static Pricing vs. Dynamic Pricing

- **Static pricing** refers to displaying a merchandiser specified price such as a list price and/or sale price to the user.
  - Static price does not take into account the promotions that the user may qualify for.
  - Joe saw static pricing on product list and detail pages.
- **Dynamic pricing** refers to programmatically determining a price based on the merchandiser specified price but also taking into account any promotions.
  - Dynamic prices run through the pricing engine and incurs a performance overhead.
  - Joe was shown the dynamic pricing on the cart and checkout pages.

## Static and Dynamic Pricing

- The static in static pricing simply refers to the fact that the pricing has not passed through a pricing engine and been adjusted for promotions.
  - The prices still come from a repository that is based on a SQL database.
  - Static pricing is the same for users using the same price list.
- Dynamic pricing has passed through the pricing engine and been adjusted for promotions.
  - Dynamic pricing is usually different for each user based on the promotions they qualify for.
- ATG recommends using static pricing on product list and product detail pages for performance considerations.

## How Static Pricing Works

- Typically, static pricing works by the merchandiser setting the price on a product and/or SKU.
    - When set directly on the SKU repository item, it is referred to as **SKU based pricing.**
    - In a price list that may be different for different users, it is referred to as **price list based pricing**.
- The website will simply access the appropriate repository item and print the price out on the page.

```
 List price: <dsp:valueof param="sku.listPrice“
                  converter="currency"/><br />
 <dsp:droplet name="/atg/dynamo/droplet/Switch">
   <dsp:param param="sku.onSale" name="value"/>
   <dsp:oparam name="true">
      On sale for
      <dsp:valueof param="sku.salePrice" converter="currency“ />!
   </dsp:oparam>
</dsp:droplet>
```

## How Dynamic Pricing Works

- For Items commerce object
  - The listPrice/salePrice of a SKU is assigned.
  - The price is discounted based on any global or individual promotions for the user.
- Orders
  - The order subtotal is calculated based on total cost of items.
  - Any order level discounts are applied to adjust the order price.
- Shipping price
  - The shipping price is assigned based on shipping method and items in the cart.
  - Any shipping level discounts are applied to adjust the price.
- Tax
  - Calculate any state, federal, or other taxes based on items in cart.

## Dynamically Pricing a Commerce Object

- Each commerce object that needs to be priced passes through a pricing engine that is specific to the item.
- There are four pricing engines:
  - **ItemPricingEngine** prices an item,
  - **OrderPricingEngine** prices an order,
  - **ShippingPricingEngine** prices shipping,
  - **TaxPricingEngine** calculates the tax.
- Each engine generates specific Pricing Holding classes:
  - **ItemPriceInfo** object holds item price,
  - **OrderPriceInfo** object holds order price,
  - **ShippingPriceInfo** holds shipping price,
  - **TaxPriceInfo** holds tax price.
- All pricing engines take PriceModelHolder which holds the global and user specific promotions.

This image cannot currently be displayed.

# Pricing a Commerce Object

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.

## Pricing a Commerce Object

- Each type of pricing object has the same basic structure:
  - A pricing engine.
  - One or more calculators specific to the pricing engine.
  - A helper method in the qualifier service.
  - An item-descriptor in the promotions repository.
  - A price Info price holding class to return the price.
- Example for an item pricing:
  - An item pricing engine.
  - An item list price calculator, an item sale price calculator, and an item discount calculator.
  - The findQualifyingItems call in atg.commerce.pricing.Qualifier.
  - The item discount item-descriptor in the repository.
  - An ItemPriceInfo class to hold the intermediate and final price.

## How Pricing Services Generate Prices

- Before pricing happens, the following steps take place:
  - A PricingModelHolder instance is created at the start of the customer session.
  - It contains a merged cache of global and user specific promotions.
- When an action prompts a pricing operation, the following steps occur:
  - A droplet or call requests a commerce object to be priced.
  - PricingTools is invoked. It gets the instance of PricingModelHolder.
  - PricingTools then invokes the pricing engine passing to it the object and the PricingModelHolder.
  - Pricing engines process the object creating a priceInfo object.
  - The priceInfo object is passed through various calculators.
  - The final priceInfo  is returned to the requesting operation.

## Pricing Services Generating Prices

This image cannot currently be displayed.

# How Pricing Engine Works

- The pricing engine goes through a series of steps to price the object passed to it. For an item:
  - First the **pre-calculators** are called. The pre-calculators assign a list and sale price to the item.
  - The appropriate method in the **qualifier service** is called. This returns a list of qualified items that specify what and how much discount to apply to the items. The promotions in the PriceModelHolder are used to determine this.
  - The **discount price calculators** are then called in sequence. They adjust the price as specified in the qualified items list and updated the priceInfo object.
  - Finally, the **post calculators** are called. They make final checks and update any prices are necessary.
- The final PriceInfo object is returned from the engine.
- The pricing mechanism is similar for all commerce objects.

## Pricing Engine, Price Holding Classes, and Calculators

- **Pricing engines** are responsible for:
  - Retrieving any promotions that are available to the site visitor,
  - Determining which pricing calculator generates the price,
  - Invoking the calculator in the correct order.
- **PriceInfo** price holding classes
  - Store price information about different objects.
  - In addition to the price it also holds how to interpret the amount and how it was calculated.
- **Pricing calculators** are used for:
  - Indentify and compute price for a SKU.
  - Using the information from the pricing engine and the qualifier service to determine price.

## Promotions, PMDL, and Qualifiers

- **Promotions** in ATG Commerce
  - Are a way to encourage a user to make a purchase by highlighting and offering discounts on products and services.
  - Example: Specific amount off a particular product.
- Pricing Model Definition Language (**PMDL**)
  - Specify promotions in ATG Commerce.
  - They describes the conditions under which the promotion should take effect.
- **Qualifier service** is a nucleus component
  - It is a helper class for discount calculators.
  - It determines which things should change and how they should be discounted.
  - Qualifier does not modify the actual price. That part is performed by the Pricing Calculators.

## PricingAdjustment, PricingContext, and PricingModel

- **PricingAdjustment** represents an element of a price's audit trail.
  - Describes why and how a particular price was changed.
  - Includes a description of the change and the amount.
  - Contains the promotion (if any) that triggered the change.
- **PricingContext** is the context under which an object was priced.
  - It contains the items being priced, order, site, promotion, locale, and the profile used when a price was calculated.
- **PricingModel** describes a discount.
  - It is a repository item that holds the promotion.
  - Includes a PMDL rule and the discount type and amount.

## PricingTools and PriceModelHolder

- **PricingTools** is a helper class.
  - It is the main way business-layer logic interacts with pricing engines and other pricing classes.
  - Contains translation functions that identify which currencyCode goes with which locale.
  - Determines pricing locale.

- **PricingModelHolder** is used by PricingTools to perform pricing.
  - A session scoped component that is created at the beginning of user session.
  - It holds the merged list customer's active promotions and global promotions.
  - Reload user's promotions when a new promotion becomes available.

## Section 1 ✓
## Check Your Understanding

When a price comes from different price lists for different users, is it static or dynamic pricing?

**Answer:**

**Static pricing. Dynamic pricing has promotions applied to static price.**

## Section 1 ✔
## Check Your Understanding

What are the four commerce objects that ATG prices?

**Answer:**

**Item, order, shipping, and tax.**

## Section 1 ✓
## Check Your Understanding

Name a few components used to price the commerce objects.

**Answer:**

**Engine, price holding class, calculators, qualifiers.**

## Section 1 ✓
## Check Your Understanding

How is the dynamic order price calculated?

**Answer:**

**The order subtotal is calculated based on the dynamic item prices. Any relevant discount is applied to it.**

## Section 1 ✔

## Check Your Understanding

What is a PMDL?

**Answer:**

**It is an XML document used to represent a promotion in ATG.**

## Section 1 ✔
## Check Your Understanding

What is a PricingModelHolder?

**Answer:**

**A session scoped component that holds the merged list customers’ active promotions and global promotions.**

## Summary

- ATG Commerce pricing services provide a flexible system for personalizing the prices of items.
- ATG prices four commerce objects: item, order, shipping, and tax.
- Each object has corresponding engines, price holder classes, calculators, qualifiers, and discount types.
- Each object is priced in a very similar process using item specific implementations.
- Static pricing is directly from the repository. Dynamic pricing applies promotions to the static price.

## Section 2:

## Using Price Lists

## Pricing Schemes in ATG

- In ATG Commerce, three pricing models are supported:
  - SKU based pricing,
  - Price list based pricing,
  - A combination of SKU and price list based pricing.
- SKU based pricing stores the price directly on the SKU.
- Price list based pricing stores the prices in a separate list. You can have multiple price lists.
- Unlike SKU based pricing, you can offer customers different pricing for the same items.
- Use a combination when most of the prices are common for all your customers with few exceptions.
- Price list based pricing is the most typical pricing that is used on commerce sites.

This image cannot currently be displayed.

## SKU and Price List Based Pricing

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.

## Overview of Setting Up Price Lists

- Create price list with the following properties:
  - Name,
  - Base price list,
  - Start date,
  - End date,
  - Locale.
- Typically, you create two price lists:
  - List price list,
  - Sale price list.
- How to price items using price lists:
  - Assign a list and sale price list to a user.
  - Price an item with a price list.
  - View a price through JSP code.

## Assigning a Price List to a User

- Assigning a price list to the user is similar to assigning catalogs to the profile.
- It is performed in the DAF servlet pipeline using the component PriceListProfilePropertySetter.
- The dafpipeline/ProfilePropertyServlet:

```
profilePropertySetters+= \
          /atg/userprofiling/CatalogProfilePropertySetter,\
          /atg/userprofiling/PriceListProfilePropertySetter
```

- The price lists are stored in the profile:
  - priceList property stores the list price list,
  - salePriceList stores the sale price list.
- The PriceListProfilePropertySetter uses:
  - Current sites defaultListPriceList and defaultSalePriceList
  - Otherwise use PriceListManagers, DefaultPriceListId, and DefaultSalePriceListId.

## Pricing Models in Price Lists

- The following pricing schemes can be used:
  - List pricing,
  - Sale pricing,
  - Bulk pricing,
  - Tiered pricing.
- Each SKU can have any or all of the above. In addition a combination of the above with SKU based pricing can also be used.
- If price is not found in the price list, ATG looks up the price in the **base price list**, for the SKU.
- Bulk and tiered pricing are together referred to as volume pricing.

## Volume Pricing

- **Bulk pricing** calculates the price of a product based on the minimum quantity. As an example:
  - Purchase up to 10 shirts for $20 each.
  - Purchase 11 to 20 shirts for $15 each.
  - Purchase 21 or more for $10 each.
  - If you purchased 15 shirts, the cost would be 15*$15 = $225.
  - If you purchased 25 shirts, the cost would be 25*$10 = $250.
- **Tiered pricing** calculates the price of a product using fixed quantity or weight at different pricing levels.
  - Purchase up to 10 shirts for $20 each.
  - Cost of shirts from 11 through 20 for $15 each.
  - Cost of shirts from 21 and up for $10 each.
  - If you purchase 15 shirts, the cost would be 10*$20+5*$15 = $275.
  - 25 shirts would cost 10*$20 + 10*$15 + 5*$10 = $400.

## PriceListManager

- PriceListManager maintains the price lists.
- You can get a price from a price list by product, by SKU, or by product and SKU.
- PriceListManager's getPrice method is used during the pricing of an order.
- PriceListManager maintains DefaultPriceListID and DefaultSalePriceListID which represent the list and sale price lists.
  - Used for assigning the price lists to profiles.
  - Also used if there are no price lists assigned to the profile.

## Price List Calculators

- The ItemPriceListCalculator components contain all the functionality common to all pricing schemes for list prices..
- ItemPriceListCalculator ‘s  PricingSchemaName property contains sub calculators for each list price scheme.
  - listPrice: ItemListPriceCalculator
  - bulkPrice: ItemBulkPriceCalculator
  - tieredPrice: ItemTierPriceCalculator
- For sales price based on price list, developers can create a component called ItemSalePriceCalculator.
- ItemSalePriceCalculator ‘s PricingSchemaName property can contain sub calculators for each sale price scheme.
  - listPrice: SalePriceListsListCalculator
  - bulkPrice: SalePriceListsBulkCalculator
  - tieredPrice: SalePriceListsTieredCalculator

## Configuring ATG Commerce to Use Price Lists

- ATG Commerce ships with SKU based pricing.
- To configure price lists, configure the ItemPricingEngine to use the appropriate pre calculator.
- In the ItemPricingEngine.properties, configure:

```
preCalculators=\
        calculators/ItemPriceListCalculator,\
        calculators/ItemPriceListSaleCalculator,\
        calculators/ConfigurableItemPriceListCalculator,\
        calculators/ConfigurableItemPriceListSaleCalculator
```

- The pre calculator assigns list and sale prices for regular and configurable SKUs.

## Using a Combination of Price Lists and SKU Based Pricing

- You can also use price lists in combination with SKU-based pricing.
- In calculators/ItemPriceListCalculator component configure:

```
noPriceIsError=false
noPriceCalculator=\
  /atg/commerce/pricing/calculators/ItemListPriceCalculator
```

- You may also want to configure the sale price calculator to default to SKU based sales price.

## Section 2 ✔

## Check Your Understanding

What are the three pricing models supported by ATG?

**Answer:**

**SKU based, price list based, and hybrid.**

## Section 2 ✓
## Check Your Understanding

Under what conditions would you use a price list based pricing model?

**Answer:**

**If you need different pricing for different segments of users or you need volume pricing.**

## Section 2 ✓
## Check Your Understanding

Name the component that needs to be added to the ProfilePropertyServlet to enable price list assignment to the users.

**Answer:**

**The PriceListProfilePropertySetter is the component that can be used to assign price lists to the user.**

## Section 2 ✓
## Check Your Understanding

What is volume pricing?

**Answer:**

**Bulk and tiered pricing are together referred to as volume pricing.**

## Section 2 ✔
## Check Your Understanding

Name the item price calculators used with price lists.

**Answer:**

**ItemListPriceCalculator, SalePriceListsListCalculator, and other volume pricing calculators.**

## Summary

- ATG suports three pricing models: SKU based, price list based, and hybrid.
- Price lists have to be assigned to a user for every session. ATG provides components to facilitate this.
- Bulk pricing calculates the price of a product based on the minimum quantity.
- Tiered pricing calculates the price of a product using fixed quantity or weight at different pricing levels.
- PriceListManager maintains the price lists and provides a default price list for assignment to the users.

## Section 3:

## Displaying Pricing

## Pricing Servlet Beans

- The following droplets are used for static pricing:
  - **PriceDroplet** returns the price for a given product or SKU.
  - **ComplexPriceDroplet** takes a complex price and returns the levels contained within it.
  - **PriceRangeDroplet** takes a product and determines the highest and lowest price for a range of SKUs associated with the product.
- The following droplets are used for dynamic pricing:
  - **PriceItem** servlet bean is used to dynamically price a single item by taking the promotions into account.
  - **PriceEachItem** servlet bean is used to price dynamically a collection of items by taking promotions into account.
- The following droplets are used for displaying shipping:
  - **AvailableShippingMethods** servlet bean displays available shipping methods for a particular shipping group.

## Static Pricing: PriceDroplet

- The PriceDroplet returns a price for a given product or SKU using price list (optional).
- It is for static price and does not use the PricingEngine.
- You can retrieve the price directly from the SKU object if the pricing is SKU based.

```
<dsp:droplet name="PriceDroplet">
  <dsp:param name="sku" value="sku" />
  <dsp:param name="product" value="product" />
  <dsp:oparam name="output">
    <dsp:droplet name="/atg/dynamo/droplet/Switch">
      <dsp:param name="value" param="price.pricingScheme" />
      <dsp:oparam name="listPrice">
        <dsp:valueof param="price.listPrice" converter="currency">
            no price</dsp:valueof>
      </dsp:oparam>
    </dsp:droplet>
  </dsp:oparam>
</dsp:droplet>
```

## Static Pricing: Use SKU Objects

- When the prices are not from price lists, you can simply get the prices from the SKU object.

```
 List price: <dsp:valueof param="sku.listPrice“
                  converter="currency"/><br />
 <dsp:droplet name="/atg/dynamo/droplet/Switch">
   <dsp:param param="sku.onSale" name="value"/>
   <dsp:oparam name="true">
      On sale for
      <dsp:valueof param="sku.salePrice" converter="currency“ />!
   </dsp:oparam>
</dsp:droplet>
```

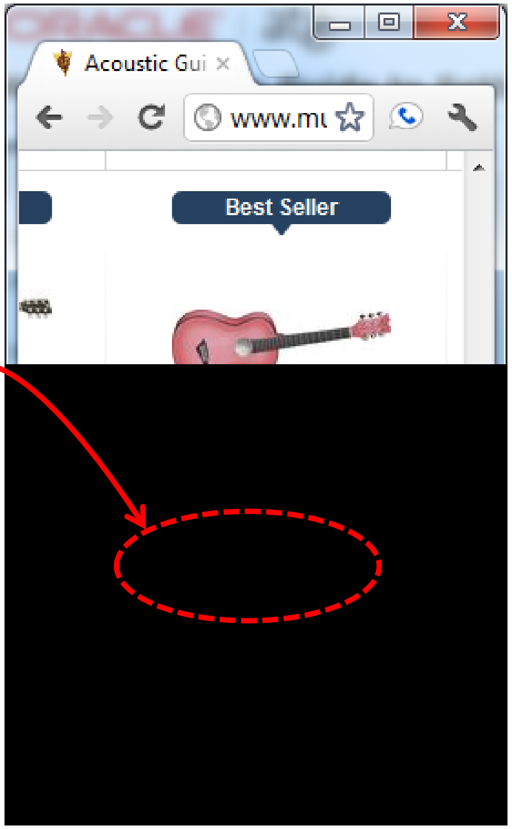
## Static Pricing: ComplexPriceDroplet

- The ComplexPriceDroplet takes a complex price and returns the levels contained within it.

```
<dsp:droplet name="ComplexPriceDroplet">
  <dsp:param param="complexPrice" name="complexPrice"/>
  <dsp:oparam name="output">
    <table border=1>
    <dsp:droplet name="For">
      <dsp:param param="numLevels" name="howMany"/>
      <dsp:param value="index" name="indexName"/>
      <dsp:oparam name="output">
         <tr><td>
           <dsp:valueof param="leveMinimums[index]"/> -
           <dsp:valueof param="levelMaximums[index]“ />
         </td><td>
           <dsp:valueof param="prices[index]"/>
         </td></tr>
      </dsp:oparam>
    </dsp:droplet>
    </table>
  </dsp:oparam>
</dsp:droplet>
```

# Static Pricing: PriceRangeDroplet

- The PriceRangeDroplet takes a product and determines the highest and lowest price for a range of SKUs associated with the product.
- It is used for product list pages to display a range of pricing for the product.



```
<dsp:droplet name="PriceRangeDroplet">
  <dsp:param name="productId" value="productId">
  <dsp:param name="priceList" bean="priceList">
  <dsp:oparam name="output">
        <dsp:valueof param="lowestPrice" /> -
        <dsp:valueof param="highestPrice" />
  </dsp:oparam>
</dsp:droplet>
```

## Dynamic Pricing: PriceItem Droplet

- PriceItem servlet bean is used to dynamically price a single item by taking promotions into account.
- It only applies item level promotions.
- The pricing is run through the pricing engine.

```
<dsp:droplet name="PriceItem">
  <dsp:param param="sku" name="item"/>
  <dsp:param param="product" name="product"/>
  <dsp:oparam name="output">
    <dsp:valueof param="element.priceInfo.amount"
       converter="currency">no price</dsp:valueof>
  </dsp:oparam>
</dsp:droplet>
```

## Dynamic Pricing: PriceEachItem

- The PriceEachItem servlet bean is to dynamically price a collection of items by taking promotions into account.

```
<dsp:droplet name="PriceEachItem">
  <dsp:param param="product.childSKUs" name="items"/>
  <dsp:oparam name="output">
    <dsp:droplet name="/atg/dynamo/droplet/ForEach">
      <dsp:param param="element" name="array"/>
      <dsp:param value="pricedItem" name="elementName"/>
      <dsp:oparam name="output">
        <dsp:valueof param="pricedItem.auxiliaryData.catalogRef.displayName"/> -
         List price: <dsp:valueof param="pricedItem.priceInfo.listPrice"
                  converter="currency">no price</dsp:valueof> <br />
        <dsp:droplet name="Switch">
          <dsp:param param="pricedItem.priceInfo.onSale" name="value"/>
          <dsp:oparam name="true">
             On sale for <dsp:valueof param="pricedItem.priceInfo.salePrice"
                      converter="currency"/>!
          </dsp:oparam>
        </dsp:droplet><BR />
      </dsp:oparam>
    </dsp:droplet>
  </dsp:oparam>
</dsp:droplet>
```

## Shipping: AvailableShippingMethods

- The AvailableShippingMethods servlet bean displays available shipping methods for a particular shipping group.
- It is used to present shipping options to the user.

```
<dsp:droplet name="AvailableShippingMethods">
  <dsp:param name="shippingGroup"
                    bean="ShoppingCartModifier.shippingGroup" />
  <dsp:oparam name="output">
    <dsp:select bean="ShoppingCartModifier.shippingGroup.shippingMethod">
      <dsp:droplet name="ForEach">
        <dsp:param param="availableShippingMethods" name="array"/>
        <dsp:oparam name="output">
          <dsp:getvalueof id="option1" param="element"
                idtype="java.lang.String">
            <dsp:option value="<%=option1%>"/>
          </dsp:getvalueof> <dsp:valueof param="element"/>
        </dsp:oparam>
      </dsp:droplet>
    </dsp:select>
  </dsp:oparam>
</dsp:droplet>
```

# Displaying Fully Discounted Pricing

- In the shopping cart, you would need to typically display fully discounted pricing for the items.
- Use the cartModifierFormHandler.order object to iterate through the items and display the price from the priceInfo objects.
- You can use the four priceInfo objects to display the discounted item, shipping, tax, and order pricing.
- The section on price info objects covers the details you can get from these objects.
- For performance considerations, you should display static pricing in the product list and detail pages.

## Section 3 ✔
## Check Your Understanding

What is the PriceDroplet used for?

**Answer:**

**The PriceDroplet returns a price for a given product or SKU using price lists.**

## Section 3 ✔
## Check Your Understanding

What droplet would you use to return levels contained within a volume price?

**Answer:**

**The ComplexPriceDroplet returns levels contained in the complex price.**

## Section 3 ✔

## Check Your Understanding

What are the key inputs to a PriceItem droplet?

**Answer:**

**The PriceItem droplet takes SKU and quantity.**

## Section 3 ✓
## Check Your Understanding

What is the element output parameter of PriceEachItem contain?

**Answer:**

**It is an array of PriceItems that have pricing and reference to SKU.**

## Section 3 ✔ Check Your Understanding

Name the droplet used to list shipping methods.

**Answer:**

**The AvailableShippingMethods servlet bean displays available shipping methods for a particular shipping group.**

## Summary

- ATG provides servlet beans to display static and dynamic pricing.
- PriceDroplet, ComplexPriceDroplet, and PriceRangeDroplet are used to display static pricing.
- You can also display static pricing by simply accessing the repository item.
- PriceItem and PriceEachItem are used for displaying dynamic pricing.
- AvailableShippingMethod is used to display available shipping methods for a particular shipping group.

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.

<Insert Picture Here>

# Q&A

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.

This image cannot currently be displayed.