Dynamic Attributes

- All hybris Models are automatically generated during the hybris Platform build, thus they cannot hold any custom changes.
- Dynamic Attributes enable you to add attributes to the hybris Models, and to create custom logic behind them, without touching the hybris Model class itself.
- In contrast to attributes with the persistence type set to **property** that are persisted in a database, Dynamic Attributes have non-persistent values.
- The fact that there is some custom logic written in a separate class is absolutely transparent to the user.

Overview of Dynamic Attributes:

- A Dynamic Attribute is defined for a type in the **items.xml** file. You just need to set the **persistence type** of the attribute to **dynamic**. For each Dynamic Attribute, a Spring bean ID known as **attributeHandler** is automatically generated. However, you may provide the custom bean ID.
- If you do not provide the custom bean id for **attributeHandler** object is: *ItemtypeCode_attributeQualifier* **AttributeHandler**. For example, the bean ID for the Dynamic Attribute with the qualifier **myHandler** for the **dynamic** type is **dynamic_myHandler** .

Provide a custom **attributeHandler** by assigning a bean id of the class that implements the **DynamicAttributeHandler** interface and holds the logic:

Example:

murchandise-Items.xml:

I had created one new dynamic attribute in my Customer itemtype as follows.

```
<attribute type="java.lang.String" qualifier="longName">

<persistence type="dynamic"attributeHandler="myHandler/>
        <modifiers read="true" write="true" optional="true" unique="false"/>
     </attribute>
```

Task:Create A Dynamic Attribute LongName which is dynamically Adding the FirstName and LastName and Also Spliting LongName into FirstName and LastName

1.Create new **Item** type with two attributes, which **persistence type** is set to **property**. Their values are persisted in the database.

murchandise-Items.xml:

```
 <itemtype code="CustomDynamicAttribute"
                 extends="GenericItem"
                 jaloclass="com.lycamobile.core.jalo.CustomDynamicAttribute"
                 autocreate="true"
                 generate="true">
```

```
          <deployment table="CustomDynamicAttribute" typecode="12004"
propertytable="CustomDynamicAttributeProps"/>
<attributes>
      <attribute autocreate="true" qualifier="firstName"
type="java.lang.String">
         <modifiers read="true" write="true" search="false" optional="true" />
         <persistence type="property" />
      </attribute>
      <attribute autocreate="true" qualifier="lastName"
type="java.lang.String">
         <modifiers read="true" write="true" search="false" optional="true" />
         <persistence type="property" />
      </attribute>
   </attributes>
</itemtype>
```

2.Create an attribute that can compute its value in the memory, by using values from persisted attributes and return the result. For this you may use the Dynamic Attribute.

Add the new Dynamic Attribute named **longName**, which uses existing attributes **firstName** and **lastName** to compute their values.Set the **persistence type** of new attribute to **dynamic.**

```
 <attribute type="java.lang.String" qualifier="longName">

 <persistence type="dynamic"attributeHandler="myHandler/>
         <modifiers read="true" write="true" optional="true" unique="false"/>
      </attribute>
```

Implement the DynamicAttributeHandler:

Create a new class named **DynamicAttributesStringSample** that implements **DynamicAttributeHandler** interface.

```
public class MyHandler implements DynamicAttributeHandler<String,
CustomDynamicAttributeModel>

{
	private final Logger LOG = Logger.getLogger("MyHandler.class");

	public static final String VALUE_DELIMITER = " ";

	/*
	 * (non-Javadoc)
	 *
	 * @see
	 *
de.hybris.platform.servicelayer.model.attribute.DynamicAttributeHandler#get
(de.hybris.platform.servicelayer.model
	 * .AbstractItemModel)
	 */
	@Override
	public String get(final CustomDynamicAttributeModel item)
	{

		if (item == null)
		{
			throw new IllegalArgumentException("Item model is
required");
		}
		return item.getFirstName() + VALUE_DELIMITER +
item.getLastName();

	}

	/*
	 * (non-Javadoc)
	 *
	 * @see
	 *
de.hybris.platform.servicelayer.model.attribute.DynamicAttributeHandler#set
(de.hybris.platform.servicelayer.model
	 * .AbstractItemModel, java.lang.Object)
	 */
	@Override
	public void set(final CustomDynamicAttributeModel item, final String
value)
	{
		if (item != null && value != null)
		{
			final String[] split = value.split(VALUE_DELIMITER);
```

```
                item.setFirstName(split[0]);
                item.setLastName(split[1]);

                item.setLongName("item.setFirstName(split[0])" + " " +
"item.setFirstName(split[0])");
                LOG.info(item);}}
```

Register the Spring Bean:

In exrension-spring.xml

```
<bean id="myHandler" class="com.lycamobile.dynamic.MyHandler"   />
```

Update Running System

Update running system using the hybris Administration Console .

Sources are compiled and in the CustomerModel class and proper getter&setter are generated.

- Now go to hmc and check the dynamic attribute as follows.

- In hmc goto users and select customers option in that.

- In search box search for the Id(****@gmail.com)

- The customer details for that id are displayed.

- Now double click and open it.

- GO to Administration tab and check for the Dynamic attribute name which you are created.