Managing Inventories on asset custom field value change -
Getting amount from screen - Example 3

It is possible to track quantity of an asset while it is added to the issue custom field or removed from it. Amount of the inventory will be taken from another field on the screen. Please follow the steps of the following example.

- Define a Quantity attribute
- Add it to the desired form
- Create multiple assets and set Quantity values (i.e. 50, 40, 30)
- Define a Text Custom Field to set amount of inventories
- Define a screen to Update Asset
- Define a new transition with Update Asset screen
- Define the post function to update quantity for the Update Asset transition
  - Post function Groovy Script
- Test it on Post Function Create Screen and publish workflow
- Update assets with Update Asset transition

In this example, we'll define a global transition with a custom screen (Update Asset) to increase or decrease quantity value of an asset when it is added or removed from asset custom field. Asset won't be updated when asset won't added to the issue or removed from it.

Define a Quantity attribute

![Edit Attribute](image)

Add it to the desired form

![Edit Form Phone](image)

Create multiple assets and set Quantity values (i.e. 50, 40, 30)
Define a Text Custom Field to set amount of inventories

**Name:** Quantities of Assets  
**Description:** "Asset Id:Quantity" pairs to change Asset’s actual quantity, example: 2001:5, 2002:2
Add "Assets" and "Quantities of Assets" fields to the new screen. Remove asset custom field from edit and create screens. Only this screen will update assets on transition. View Issue screen will have Asset custom field as read-only.

**Configure Screen**

This page shows the way the fields are organized on **Update Asset** screen.

Note: when the screen is shown to the user only non-hidden fields that the user has permissions to edit will be actually displayed.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assets</td>
<td>AIP Assets And Inventory</td>
</tr>
<tr>
<td></td>
<td>Plugin Custom Field</td>
</tr>
<tr>
<td>Quantities of Assets</td>
<td>Text Field (single line)</td>
</tr>
</tbody>
</table>

AP: Project Management Screen Scheme [DEFAULT]

These 2 issue types... ...use this screen scheme

- Task [DEFAULT]
- Sub-task

<table>
<thead>
<tr>
<th>Operation</th>
<th>Screen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Issue</td>
<td>AP: Project Management Create Issue Screen</td>
</tr>
<tr>
<td>Edit Issue</td>
<td>AP: Project Management Edit Issue Screen</td>
</tr>
<tr>
<td>View Issue</td>
<td>AP: Project Management View Issue Screen</td>
</tr>
</tbody>
</table>

**Define a new transition with Update Asset screen**

In this example it is defined as All-to-self. You can define for a specific status as well.
Define the post function to update quantity for the Update Asset transition

Select Add "[AIP] - Update Asset workflow post function" to the workflows to update an asset object on transition. And select Quantity field. Leave form parameter as blank to set any matching form.
**Post function Groovy Script**

```
import inventoryplugin.entity.InventoryItem;
import java.util.List;
import com.atlassian.jira.issue.Issue;
import org.gradle.language.jvm.tasks.JvmExec;

// Add a quantity custom field to the transition
String quantityCustomField = getCustomField("customfield_10001");

// Get the custom field value from the transition
String fieldValue = getCustomFieldValue(issueParam);

// Update the custom field value
issue.setValue(quantityCustomField, fieldValue);
```

**Attributes to be updated**

- `Quantity` - Use default groovy script

**Add another attribute**

Add an attribute and write the groovy script to update an Asset and Inventory Plugin asset object attribute. You can define multiple configurations which will match the asset custom field value to update. If issue has more than one asset, all of them are updated when configuration matches.
import inventoryplugin.entity.JipInventoryItem;
import org.apache.commons.lang3.StringUtils;
import com.atlassian.jira.issue.Issue;

// "Quantities of Assets" text custom field let us set amount of the inventories in the transition
// customfield_10501 is the id of the custom field.
// We expect a pattern "Asset Id:Quantity" pairs, example: 2001:5, 2002:2
String getTextCustomFieldValue(Issue issueParam) {
    def customFieldManager = ComponentAccessor.getCustomFieldManager()
    def cField = customFieldManager.getCustomFieldObject("customfield_10501")
    return issueParam.getCustomFieldValue(cField)
}

// if asset not found or an exception occurs, "0" will return
Integer getAmountOfAssetInThisTransition(Issue issueParam) {
    try {
        Integer result = 0;
        def quantityFieldOfIssue = getTextCustomFieldValue(issueParam);
        if (quantityFieldOfIssue != null) {
            def assetPairs = quantityFieldOfIssue.split(',');
            assetPairs.each { assetPair ->
                try {
                    def (assetId, assetQuantity) = assetPair.trim().split(':');
                    if (assetId.toString() == asset.ID.toString()) result = assetQuantity.trim() as Integer;
                }
                catch (Exception e) {
                    // do nothing
                }
            }
        }
        return result;
    }
    catch (Exception e) {
        return 0
    }
}

// as field is a Text type we need to convert value to int and return as String
String getNewQuantityValueOfAsset() {
    def stringValue = StringUtils.trim(aipUtils.getAttributeValueAsStringByName(asset, 'Quantity'));
    if (stringValue != null && stringValue.isInteger()) {
        Integer prevAmount = getAmountOfAssetInThisTransition(originalIssue);
        Integer newAmount = getAmountOfAssetInThisTransition(issue);
        if (prevAmount == null) prevAmount = 0;
        if (newAmount == null) newAmount = 0;
        int intValue = stringValue as Integer
        if (assetStatus == 'added') {
            intValue = intValue - newAmount;
        } else if (assetStatus == 'removed') {
            intValue = intValue + prevAmount;
        } else if (assetStatus == 'noChange') {
            // asset was already in the field. We'll try to calculate the amount of the change.
            if (prevAmount != newAmount) {
                intValue = intValue - (newAmount - prevAmount);
            }
        }
        return intValue as String
    } else {
        return stringValue
    }
}
return getNewQuantityValueOfAsset();
• aipUtils.getAttributeValueAsStringByName returns the Quantity attribute value of the asset. **Quantity will calculated as 0 if nothing found for asset in “Quantities of Assets” field**

• "return null" result won’t update asset. If you want to clear attribute value return empty String "return ‘’"

• This script will be executed for each assets of all custom fields that matches the configuration. Assets also includes removed assets to let you control inventory. If you want to do nothing for assets removed from issue you can use if (assetStatus == ‘removed’) return null

• Don’t forget to change custom field id. In this example it is customfield_10501

**Test it on Post Function Create Screen and publish workflow**

Update asset “Quantities of Assets” field for test

```groovy
intValue = intValue - newAmount;
else if (assetStatus == 'removed') {
    intValue = intValue + prevAmount;
}
```

This is the default script to be executed for the options that has no specific groovy script. Type script returning the value to update asset attribute

If you do not want to change the attribute value return null.

If you want to clear the attribute value return empty string: return ‘’

**Script will be actually executed, if you modify anything please test objects (issue, asset, etc.)!**

**Result:** 50

**Publish workflow and now ready to try it with test issues and test assets.**
Update assets with Update Asset transition

First: Issue has no assets

Asset Project / AP-1

This is your first task

Details

- Type: Task
- Priority: Medium
- Status: IN PROGRESS
- Resolution: Unresolved
- Labels: None

Description

Issues are the things you do in a project. In business projects, issues are called tasks.

3 Phones have 50, 40, 30 Quantity values
Delete "Quantities of Assets" value with Update Asset transition before test.

Update issue Issue updated with transition
Assets: MP-001 #2102 [Phone], MP-002 #2101 [Phone]
Quantities of Assets: 2102:4, 2101:3

Issue updated and 2 phones are updated with 46 and 37 quantity values (before 50 and 40)
Now 3rd asset is added

Update Asset

Assets: MP-001 #2102 [Phone] ×, MP-002 #2101 [Phone] ×, MP-003 #2100 [Phone] ×

Links issues with assets

Quantities of Assets: 2102:4, 2101:3, 2100:5

“Asset Id:Quantity” pairs to change Asset’s actual quantity, example: 2001:5, 2002:2

Comment

Quantity values after update is 46, 37, 25

Now 2 assets are removed
Quantity values after update is 46, 40, 30
Now change inventory amount from 4 to 7

Update Asset

Assets: MP-001 #2102 [Phone] ✗

Links issues with assets

Quantities of Assets: 2102:7

*Asset Id:Quantity* pairs to change Asset's actual quantity, example: 2001:5, 2002:2

Comment:

Quantity values after update is 43, 40, 30
Now remove last asset from issue

**Update Asset**

- **Assets:** Start typing to search by name or #ID (e.g. #123)
- **Quantities of Assets:** Links issues with assets
  - "Asset Id:Quantity" pairs to change Asset's actual quantity, example: 2001:5, 2002:2
- **Comment:**

  ![Comment Section]

  - Visual
  - Text

  Viewable by All Users

Quantity values after update is 50, 40, 30 - as we started at the beginning.
<table>
<thead>
<tr>
<th><strong>Asset ID</strong></th>
<th><strong>Asset Name</strong></th>
<th><strong>Form</strong></th>
<th><strong>Vendor</strong></th>
<th><strong>Model</strong></th>
<th><strong>Quantity</strong></th>
<th><strong>Serial Number</strong></th>
<th><strong>IMEI</strong></th>
<th><strong>Screen Size</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>#2102</td>
<td>MP-001</td>
<td>Phone</td>
<td>AcBel</td>
<td>iPhone 7 Plus 32 gb</td>
<td>50</td>
<td>1234234 234</td>
<td>w234234234</td>
<td>5.5&quot;</td>
</tr>
<tr>
<td>#2101</td>
<td>MP-002</td>
<td>Phone</td>
<td>Samsung</td>
<td>S8</td>
<td>40</td>
<td>1231231 23</td>
<td>234234234</td>
<td>5.7&quot;</td>
</tr>
<tr>
<td>#2100</td>
<td>MP-003</td>
<td>Phone</td>
<td>Sony</td>
<td>Xperia XZ3</td>
<td>30</td>
<td>1231231 23</td>
<td>1231231 2312</td>
<td>4.7&quot;</td>
</tr>
</tbody>
</table>