uRapidFlow Customization

Events

A general way to add observers to the events is:

1. Declare an observer in your custom module configuration, etc/events.xml or etc/adminhtml/events.xml (any new file)

An example: `Acme/Module/etc/events.xml`

MyCustom_Module.xml

```
<?xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="urn:magento:framework:Event/etc/events.x
sd">
  <event name="urapidflow_profile_reindex_after">
    <observer name="urapidflow"
instance="Acme\Module\Observer\ProductUrlUpdateObserver"/>
  </event>
</config>
```

`Acme/Module/Observer/ProductUrlUpdateObserver.php`

Observer.php

```
<?php

namespace Acme\Module\Observer;

use Magento\Framework\Event\ObserverInterface;

class ProductUrlUpdateObserver implements ObserverInterface
{
    /**
     * @var \Unirgy\RapidFlow\Helper\Url
     */
    protected $helper;

    /**
     * CategoryUrlUpdateObserver constructor.
     * @param \Unirgy\RapidFlow\Helper\Url $helper
     */
    public function __construct(\Unirgy\RapidFlow\Helper\Url $helper)
    {
        $this->helper = $helper;
    }

    public function execute(\Magento\Framework\Event\Observer
$observer)
    {
        /** @var \Unirgy\RapidFlow\Model\Profile $profile */
        $profile = $observer->getData('profile');
        try {
            $this->helper->updateProductsUrlRewrites();
        } catch (\Magento\Framework\Exception\AlreadyExistsException
$e) {
            $profile->getLogger()->error($e->getLogMessage());
        }
    }
}
```

Product Import Events

There are 5 stages during product import process where a custom logic can be plugged in as an event observer:

- `urapidflow_product_import_after_fetch` - after fetching new and old data
- `urapidflow_product_import_after_validate` - after validating the new data
- `urapidflow_product_import_after_diff` - after finding the difference between new and old data
- `urapidflow_product_import_after_save` - after all changes has been saved
- `urapidflow_product_import_after_rtidx` - after real-time reindex ran (if enabled in profile)

The events are fired for each page of data (by default 100 rows), and each stage has more vars available than previous, and including all vars that were available before.

urapidflow_product_import_after_fetch

Vars

- `profile` - instance of current Unirgy_RapidFlow_Model_Profile
- `new_data` - rows fetched from CSV file (including any default values for missing columns, as specified in the profile)
- `old_data` - product information for matching SKUs fetched from DB
- `skus` - mapping of SKUs in the current page to product IDs
- `attr_value_ids` - EAV attribute value IDs for fetched old data

Example

This example observer will change weight values of “2kg” and “470g” into correct decimal weight value.

`Acme/Module/Observer/WeightUpdateObserver.php`

WeightUpdateObserver.php

```php
<?php

class WeightUpdateObserver implements ObserverInterface
{
    public function execute(Observer $observer)
    {
        $vars = $observer->getEvent()->getVars();
        foreach ($vars['new_data'] as &$row) {
            if (strpos($row['weight'], 'kg')!==false) {
                $row['weight'] = intval($row['weight']);
            } elseif (strpos($row['weight'], 'g')!==false) {
                $row['weight'] = intval($row['weight'])/1000;
            }
        }
        unset($row);
    }
}
```

urapidflow_product_import_after_validate

Vars

- `valid` - which products passed or not validation

urapidflow_product_import_after_diff

Vars

- `insert_entity` - new products to be created
- `change_attr` - changed attributes
- `change_website` - changed product-website associations
- `change_stock` - changed inventory stock information
- `change_category_product` - changed product-category associations

Product Export Events

There are 2 stages during product import process where a custom logic can be plugged in as an event observer:

- `urapidflow_catalog_product_export_before_format` - after fetching data from db and before formating for output
- `urapidflow_catalog_product_export_before_output` - after formating for output and right before writing to file

The events are fired for each page of data (by default 100 rows), and each stage has more vars available than previous, and including all vars that were available before.

urapidflow_catalog_product_export_before_format

Vars

- `profile`
- `products`
- `fields`

urapidflow_catalog_product_export_before_output

Vars

- `profile`
- `products`
- `fields`
- `rows`