Product Labels Manual

Getting Started

Welcome to the **Catalog Labels** documentation.

You will find everything you need to set up your **Catalog Labels** so that you can support nearly any marketing strategy with attractive labels on your products' images.

Go ahead, dive in!

Firstly, please check out our extension in the My Downloadable Products section of our store. Learn how to install extension, and proceed with Quick Start, which will guide you to set up your product labels.

How to install the extension

1. Back up your store database and web directory.
2. Log in to the SSH console on your server and navigate to the root directory of the Magento 2 store.
3. Copy the installation instructions from the page My Downloadable Products to the SSH console and press ENTER.
4. Run the command `php -f bin/magento module:enable Mirasvit_Core Mirasvit_CatalogLabel` for enable extension.
5. Run the command `php -f bin/magento setup:upgrade` to install extension.
6. Run the command `php -f bin/magento cache:clean` for clean cache.
7. Deploy static view files

```
rm -rf pub/static/frontend/*; rm -rf pub/static/backend/*; rm -rf var/view_preprocessed/*; php -f bin/magento setup:static-content:deploy
```

Quick Start

Our Catalog Label extension is a simple yet powerful tool for marking your product according to your marketing and sale policy.

It is quite intuitive and works our-of-box, but there are some things to be tuned up.

1.

Start with **Placeholders**. They are blocks that overlap product images and allow labels to be displayed.

2. Create **Labels**, which will display certain attribute values or combined conditions, allowing you to plan complex campaigns and mark promoted products.

3. Check your labels' appearance. If additional styles at Labels are not helpful, there are **some tricks** that you can use to fit labels to your store.

This should be a good start.

Refer to the corresponding sections of this manual to learn more.

Catalog Labels Configuration

Each Catalog Labels extension has its own section at **Stores -> Configuration -> Mirasvit Extensions -> Catalog Label**.

However, for now, this section features only one setting:

- **Flush dependent pages cache after creating new product**

When this option is set to **Yes**, all pages where this product are displayed or mentioned will be automatically flushed (i.e. their cached versions will be purged). It allows you to quickly update part of your store when a new arrival comes without the entire cache regenerating. It is extremely useful if you have a very complex store, and/or caching is a serious matter of productivity.

If you wish to fit your product labels to your theme, refer to the Labels section.

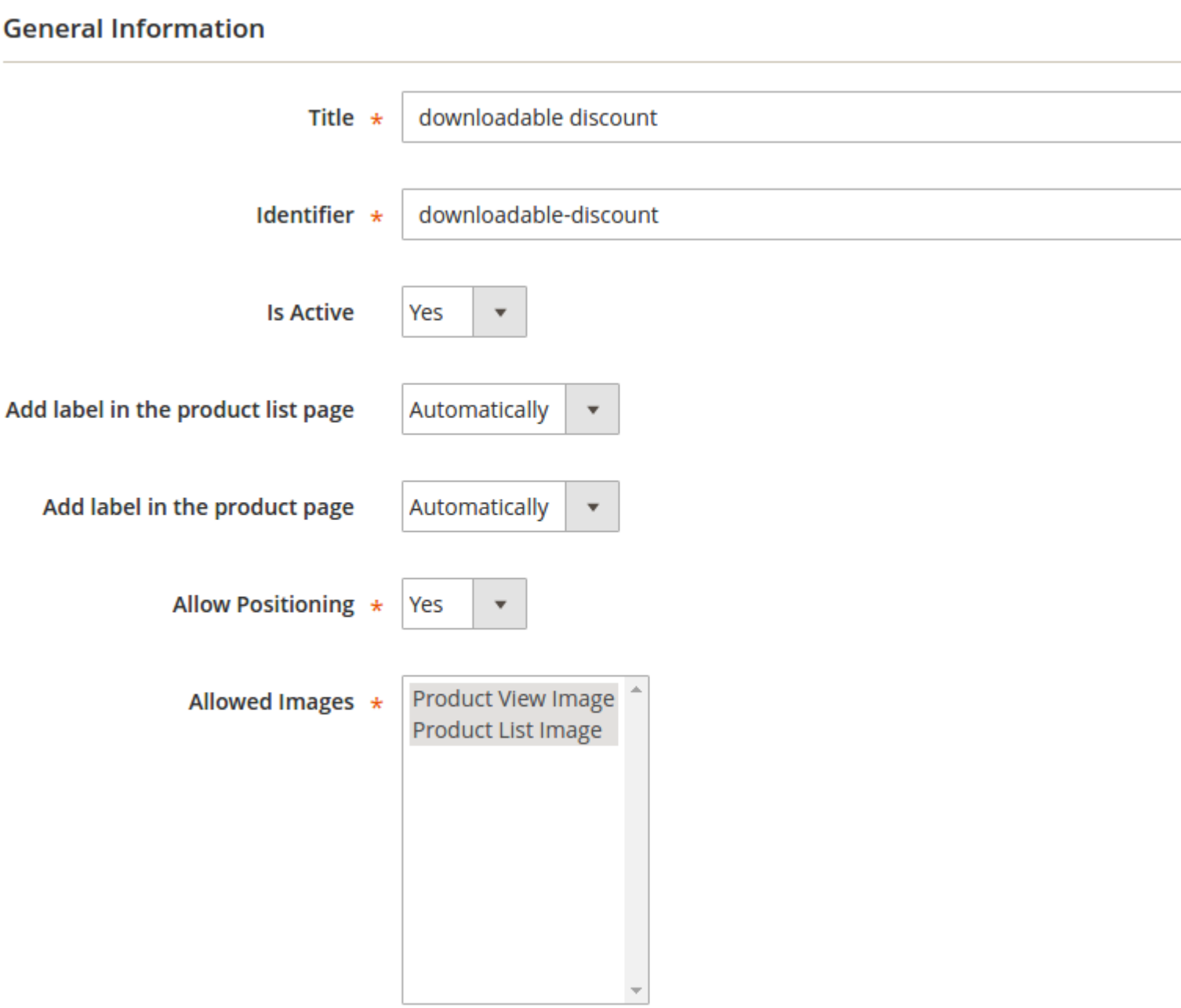
Manage Placeholders

Placeholders are the most basic building blocks of our extension. They should be created before any **Label** is created.

All of them are located at **Marketing -> Promotions -> Product Labels -> Manage Placeholders** section. From there, you can edit, remove, or create a new Placeholder.

How to Create New Placeholder

Visit **Marketing -> Promotions -> Product Labels -> Manage Placeholders** and press the **Add New** button. You will be brought to the Creation Page, as shown on the screenshot.



This page contains only one section, **General Information**, which consists of the following fields:

- **Title** - a sensical title of placeholder.
- **Identifier** - The identifier (code) of placeholder that is used to display labels, bound to this placeholder at required positions.
- **Is Active** - indicates whether this placeholder is active, and the bound labels are eligible for display.
- **Add label in the product list page** - Sets a method for adding labels to products in the product list page (such as Category or Search Results):
    - **Automatically** - allows for adding product labels to the product list page automatically.
    - **Manually** - allows for manual labels placing (default method, but preferable only when automatic does not work).
- **Add label in the product page** - Sets the method for adding labels to products in the product page:
    - **Automatically** - allows for adding product labels to the product page automatically.
    - **Manually** - allows for manual labels placing (default method, but preferable only when automatic does not work).

- **Allow Positioning** - when enabled, this option allows you to change labels positioning, bound to the placeholder.
- **Allowed Images** - sets the allowed types of images where the product label needs to be applied. There are two default types:
    - **Product View Images** - the images are displayed as part of Product Info.
    - **Product List Image** - the thumbnails are used for building the Catalog or Search Result page.

Manage Labels

Product Labels are markers which are placed on top of product images at Catalog and Product pages. Using these marks, you can easily promote your new product or boost sales during marketing campaigns.

All Labels are located at the **Marketing -> Product Labels -> Manage Labels** section. From there, you can edit, remove, or create a new label.

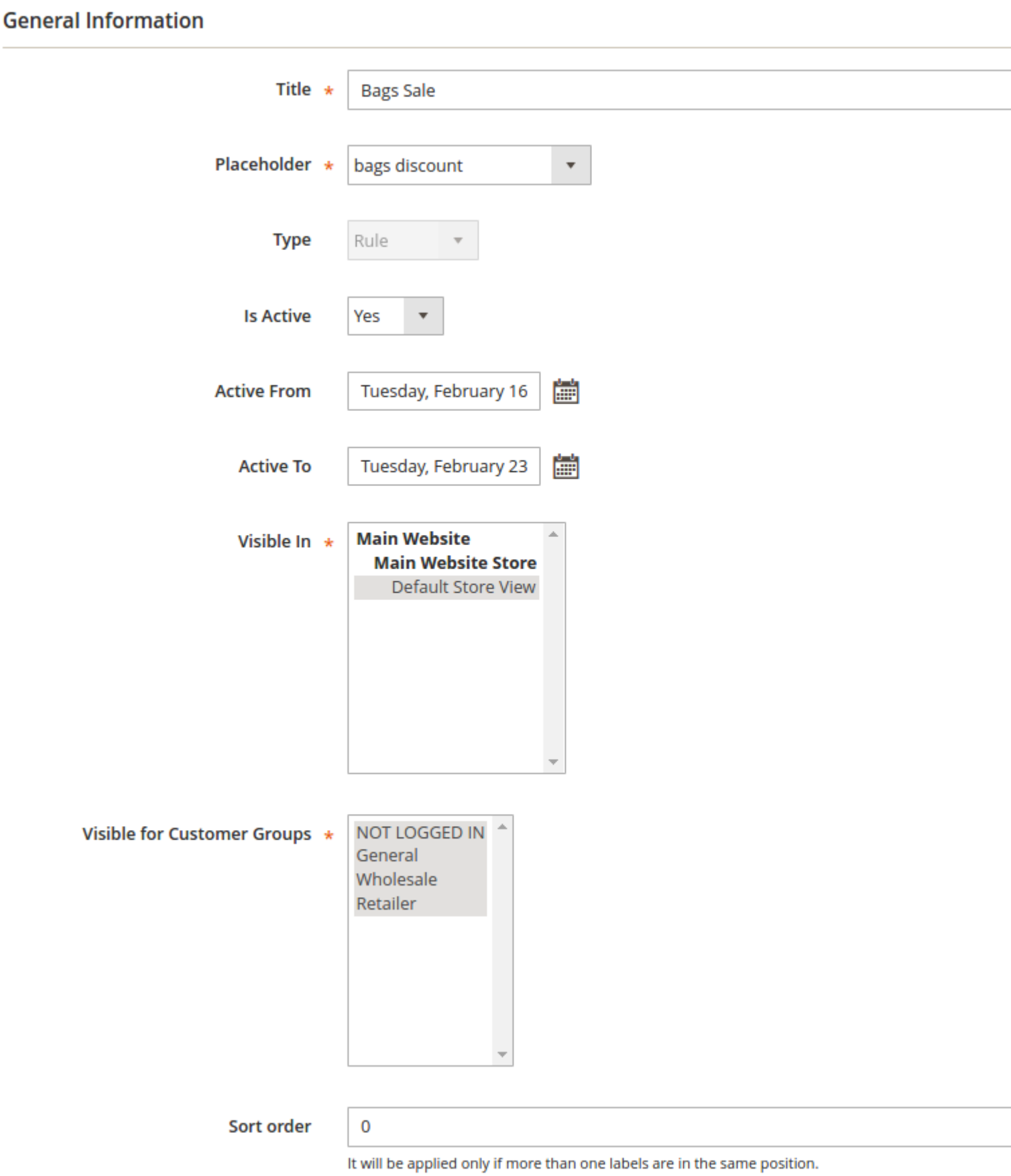
How to Create New Label

Visit the **Marketing -> Product Labels -> Manage Labels** section and press the **Add New** button. Labels creation is broken into two Stages, one of which selects the Label Type, and the second gives details of their application.

The first stage consists of the following fields:

- **Title** - a logical title of the label.
- **Placeholder** - a placeholder which will be used as a framework for your label. Read more at the Placeholders section.
- **Relation Type** - a type of label which needs to be created.
    - **Attribute** - a simple label that is displayed when the product has a selected attribute value.
    - **Rule** - a complex label that uses conditions to determine when a label needs to be applied.
- To proceed to the next stage, press the **Continue** button.

The next stage contains detailed information about label displaying. **General Information** is displayed for both types of Label, and consists of the following fields:



- **Is Active** - indicates whether the label is active and ready to apply.
- **Active From**, **Active To** - time bounds when a label should be applied. Very useful for planning Sales campaigns.
- **Visible In** - stores where the label needs to be applied.

- **Visible for Customer Groups** - customer groups who are eligible to see labels.
- **Sort Order** - priority order in which labels are applied to products.

Attribute-based Labels Setup

If **Attribute** was selected in the previous stage of **Relation Type**, the second stage will feature an additional field in the **General Tab**:

- **Attribute** - selects the attribute that is used as a condition to display certain labels. These attributes can be viewed and even created at **Stores -> Attributes -> Product** section.

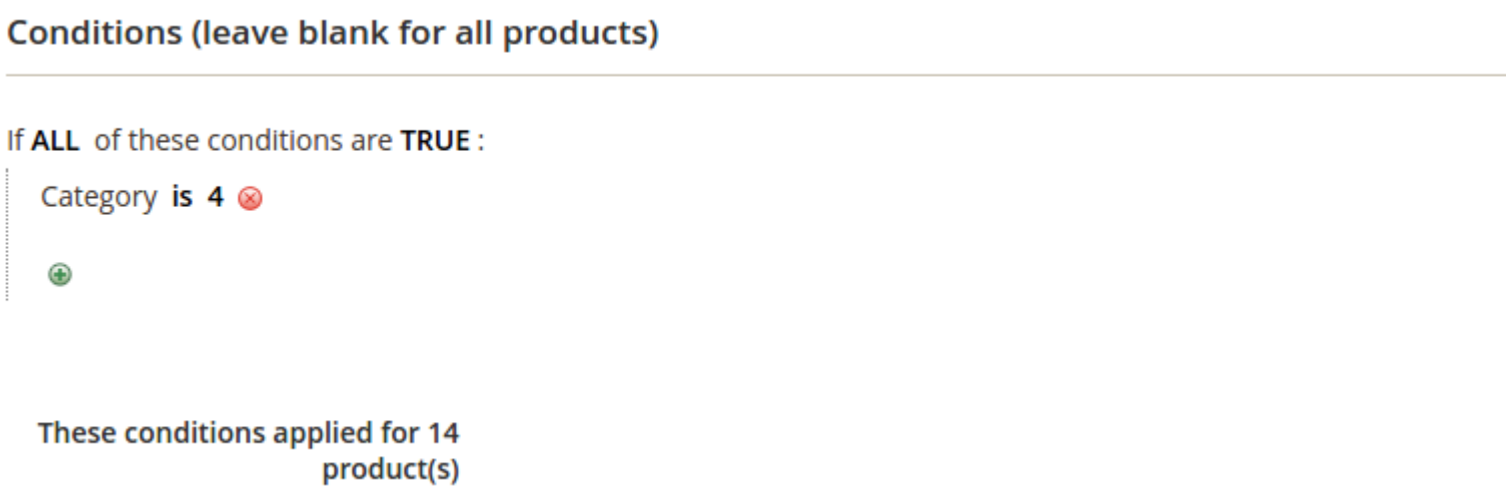
You can bind different labels for different attribute values, using the additional tab **Gallery** that is featured in this type of Label. It can contain zero or more rows with the following label definitions:

- **Option** - a value of the selected attribute, which triggers a label display.
- **Product List Image** - image will be displayed overlapping product images at list pages, such as Category or Search Results.
- **Title** - logical title for a label (for product list image).
- **Description** - short description of label's purpose (for product list image).
- **URL** - URL associated with the label which is useful for campaigns (for product list image).
- **Product View Image** - image which will be displayed overlapping product images at the Product View pages.
- **Title** - logical title for a label (for product view image).
- **Description** - short description of the label's purpose (for product view image).
- **URL** - URL associated with the label, which is useful for campaigns (for product view image).

Rule-based Labels Setup

If **Rule** was selected as the **Relation Type** in the previous Stage, the Second Stage will feature two additional tabs: **Conditions** and **Design**.

At the **Conditions** tab, you will see a conditional block, which allows you to filter products to which labels should be applied, as shown on the screenshot:



All conditionals should be enclosed in the global mode block. They have four possible global modes of applying, shown in the special header `If *[apply mode]* of these conditions are *[validation mode]*`:

Applying modes define, when rule shall be triggered:

- **ALL** - implies that the rule will be executed only when all conditions are strictly met;
- **ANY** - implies that the rule will be executed only when one or more (but not all) of conditions are met;

Validation modes define which result can produce each condition to be registered as "met":

- **TRUE** - implies that the conditions need to be valid.
- **FALSE** - implies that the conditions need to be invalid.

You can also define multiple nested mode blocks by selecting the **Conditions Combination** option. These modes allow for creating flexible condition sets to satisfy a policy of any complexity.

Once you have selected global mode (or left it as default), press the green (+) button, and select a condition type. There are three categories of available conditions:

- **Product**

  Contains three base attributes which are used for identifying products:

  - **Attribute Set** - a set of attributes defining a particular type of product (e.g. jewelry, for example). All these sets can be seen at **Stores -> Attributes -> Attribute Set**.
  - **Category** - one or more categories where the product needs to be registered.
  - **SKU** - direct value, or pattern (using **contains** operator), that needs to be in the product's SKU code.

- **Product Attribute**

  Contains all attributes, defined at **Stores -> Attributes -> Product**, the most interesting of which are:

  - **New** - allows to check whether the product is New.
  - **Format** - a digital form of product (useful for downloadable products)

- **Product Additional**

  Contains special attributes that can be used for building complex campaigns:

  - **Created At** - checks how many days ago the product was added to the store.
  - **Updated At** - checks how many days ago the product was last changed.
  - **Percent Discount** - analyzes which discount was applied to this particular product.

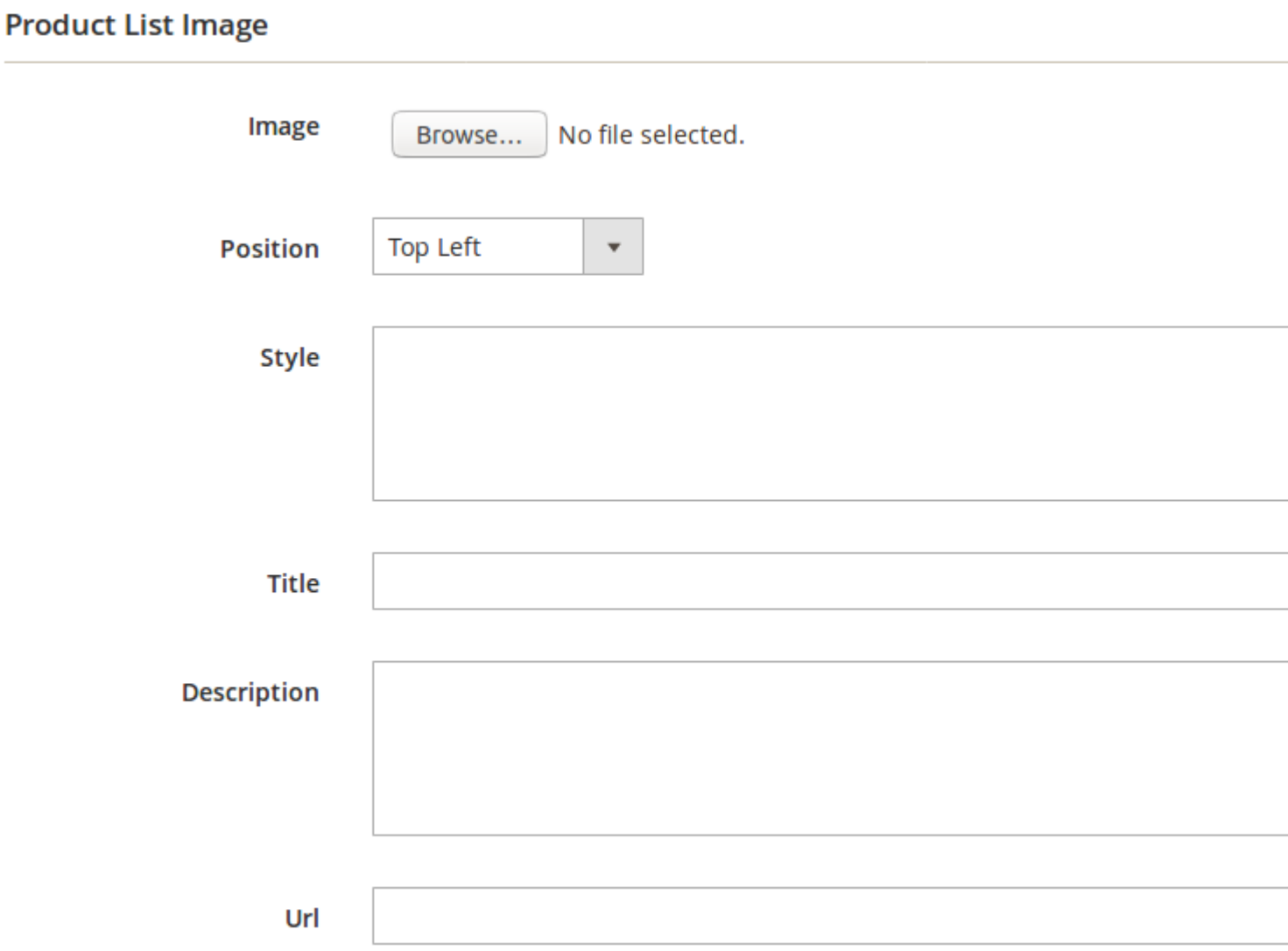
    Note

    Only discounts that have been applied directly to the products are analyzed. They are created at **Marketing -> Catalog Price Rules**

.

- **Price - Final Price** - is the final price, calculated after the application of all discounts and special price rules, which customers will actually pay.
- **Quantity** - the quantity of a product which is currently in stock. This is useful when you need to mark the product as sold out.
- **Set As New** - checks whether the product was set as new (including old products, returned as new arrivals).

At the **Design** tab, you can specify label appearance options:



- **Image** - uploads label which will be added to the product image.
- **Position** - sets the position of the label on the product image.

Example

The Position is calculated from two directions: **vertical** and **horizontal**.
Vertical are:

- **Left**
- **Center**
- **Right**

Horizontal are:

- **Top**
- **Middle**
- **Bottom**

If you wish to position the label to the upper right corner, select in the drop-down list **Top Right** option.

- **Style** - additional CSS styles that should be applied to the label.
- **Title** - sets title on the label.
- **Description** - sets label description.
- **Url** - sets URL key of label.

Display Labels within a Custom Theme

Some themes have different layouts that display products in their own unique formats. In this case, you need to adjust templates manually. Take a look at the real example below.

Assuming that `$_product` contains a current product object, you will need to insert the following code to make our extension place Catalog Label.

Example

```
<?php
    $objectManager = \Magento\Framework\App\ObjectManager::getInstance();
    $labels = $objectManager->create('Mirasvit\CatalogLabel\Model\ResourceModel
                            ->addActiveFilter()
                            ->addStoreFilter($_product->getStore());
    /** @var \Mirasvit\CatalogLabel\Model\Label $label */
    foreach ($labels as $label) {
        $placeholder = $label->getPlaceholder();
        if ($placeholder->getIsAutoForList()) {
            echo $block->getLayout()->createBlock('\Mirasvit\CatalogLabel\Block
                ->setType('list')
                ->setTemplate('product/badge.phtml')
                ->setPlaceholderCode($placeholder->getCode())
                ->setProduct($_product)
                ->setWidth(null)
                ->setHeight(null)
                ->toHtml();
        }
    }
```

```
?>
```

A Real Example

Consider the stantard template `Magento/Catalog/view/frontend/templates/product/list.phtml`, which contains the following code:

```
<?php foreach ($_productCollection as $_product): ?>
    <?php /* @escapeNotVerified */ echo($iterator++ == 1) ? '<li class="item pr
    <div class="product-item-info" data-container="product-grid">
        <?php
        $productImage = $block->getImage($_product, $image);
        if ($pos != null) {
            $position = ' style="left:' . $productImage->getWidth() . 'px;'
                . 'top:' . $productImage->getHeight() . 'px;"';
        }
        ?>
        <?php // Product Image ?>
        <a href="<?php /* @escapeNotVerified */ echo $_product->getProductUrl()
            <?php echo $productImage->toHtml(); ?>
        </a>
        <div class="product details product-item-details">
            ...
<?php endforeach; ?>
```

We will insert our label overlay code right after the `<div class="product details product-item-details">` line, inside the products iteration cycle:

```
<?php
    $objectManager = \Magento\Framework\App\ObjectManager::getInstance();
    $labels = $objectManager->create('Mirasvit\CatalogLabel\Model\ResourceModel
                            ->addActiveFilter()
                            ->addStoreFilter($_product->getStore());
    /** @var \Mirasvit\CatalogLabel\Model\Label $label */
    foreach ($labels as $label) {
        $placeholder = $label->getPlaceholder();
        if ($placeholder->getIsAutoForList()) {
            echo $block->getLayout()->createBlock('\Mirasvit\CatalogLabel\Block
                ->setType('list')
                ->setTemplate('product/badge.phtml')
                ->setPlaceholderCode($placeholder->getCode())
                ->setProduct($_product)
                ->setWidth(null)
                ->setHeight(null)
                ->toHtml();
        }
    }
?>
```

This code will create a block which will overlap the image, creating a place for our label.

The same approach can be applied to other Magento 2 custom themes.

How to upgrade an extension

To upgrade an extension, take the following steps:

1. Back up your store database and web directory.
2. Log in to the SSH console of your server and navigate to the root directory of the Magento 2 store.
3. Run the command `composer require mirasvit/module-cataloglabel: --update-with-dependencies` to update the current extension with all dependencies.

   Note

   In some cases, the command above is not applicable, or you are unable to update just the current module, and need to upgrade all Mirasvit modules in a bundle. In this case, the command above will have no effect.

   Instead, run the `composer update mirasvit/*` command. It will update all Mirasvit modules installed in your store.

4. Run the command `php -f bin/magento setup:upgrade` to install updates.
5. Run the command `php -f bin/magento cache:clean` for a clean cache.
6. Deploy static view files

   ```
   rm -rf pub/static/frontend/*; rm -rf pub/static/backend/*; rm -rf var/view_preprocessed/*; php -f bin/magento setup:static-content:deploy
   ```

Disabling Extension

Temporary Disabling

To temporarily disable the extension, please follow the next steps:

1. Log in to the SSH console on your server and navigate to the root directory of the Magento 2 store.
2. Run the command `php -f bin/magento module:disable Mirasvit_CatalogLabel` to disable extension.
3. Log in to Magento backend and refresh the store cache (if enabled).

Extension Removing

To uninstall the extension, please take the following steps:

1. Log in to the SSH console on your server and navigate to the root directory of the Magento 2 store.
2. Run the command `composer remove mirasvit/module-cataloglabel` to remove the extension.
3. Log in to Magento backend and refresh store cache (if enabled).

Change Log

1.1.25

(2021-11-26)

Fixed

- getDisplays issue

1.1.24

(2021-11-25)

Improvements

- Speed up the getDisplays() method

1.1.23

(2021-08-31)

Improvements

- Added "Is Salable" product rule

1.1.22

(2021-03-02)

Fixed

- Apply persent discount rule issue

1.1.21

(2021-03-02)

Fixed

- Re-saving dates problem

1.1.20

(2020-10-19)

Fixed

- Small spelling fixes

1.1.19

(2020-07-30)

Improvements

- Support of Magento 2.4

Fixed

- missing product in ImageBuilder

1.1.18

(2020-06-18)

Fixed

- Missing products when attribute rule enabled

1.1.17

(2020-05-27)

Fixed

- Unable to save image
- Cannot instantiate abstract class issue
- Multiple labels display issue; added out of stock option t rules
- Invalid template file error in system.log

1.1.16

(2020-03-17)

Fixed

- Wrong rule processing when multistore inventory enabled

1.1.15

(2020-01-20)

Feature

- Display labels without images

1.1.14

(2019-05-29)

Fixed

- Position issue on product list

1.1.13

(2019-01-23)

Fixed

- M2.3. Product Label does not show on catalog page

1.1.12

(2019-01-03)

Fixed

- M2.1. Solved compilation issue

1.1.11

(2018-11-29)

Improvements

- M2.3 support

1.1.11

(2018-11-29)

Improvements

- M2.3 support

1.1.10

(2018-11-28)

Fixed

- support of magento 2.3

1.1.9

(2018-08-16)

Fixed

- Installation issue with area code

1.1.8

(2018-07-18)

Fixed

- Fixed an issue with not clickable link for Label on Product List page

---

1.1.7

(2018-07-13)

Fixed

- Fixed Percent Discount

1.1.6

(2018-07-09)

Fixed

- Properly display labels based on Percent Discount condition

---

1.1.5

(2018-06-02)

Fixed

- Fixed Percent Discount if Advanced Pricing->Special Price is given in percent

---

1.1.4

(2018-06-13)

Fixed

- Fixed an issue with unexpected label on product page

1.1.3

(2018-05-31)

Fixed

- Fixed an issue related to the rule cancellation while cron starts

---

1.1.2

(2018-05-21)

Fixed

- Fixed Attribute Gallery styles

1.1.1

(2018-04-26)

Fixed

- Fix big amount of memory usage
- Fixed Percent Discount rule for bundle products

1.1.0

(2018-03-16)

Improvements

- Flush dependent pages cache after new product creating

1.0.16

(2018-03-09)

Fixed

- Fixed an error "Fatal error: Uncaught Error: Call to undefined method Magento\CatalogRule\Model\ResourceModel\Rule::calcProductPriceRule()"

1.0.15

(2018-01-29)

Fixed

- Fixed bottom label position for products that have a single image with no thumbnails displayed.

1.0.14

(2017-10-17)

Fixed

- Magento 2.2 compatibility

1.0.13

(2017-10-17)

Fixed

- Magento 2.2 compatibility

1.0.11

(2017-06-20)

Documentation

- Online User Manual updated

1.0.10

(2017-05-18)

Fixed

- Fixed usability issue: "Style" field made adjustable for "Manage Labels" Admin Panel page
- Fixed "Notice: Undefined variable: percent" for some stores

---

1.0.9

(2017-03-29)

Improvements

- Added possibility to define additional CSS styles for labels from Admin Panel

Fixed

- Fixed design issue - labels removed from shopping cart page

---

1.0.8

(2017-01-12)

Fixed

- Fixed design issue

1.0.7

(2016-12-27)

Improvement

- Added product attribute "Set as New" to label conditions

1.0.6

(2016-12-06)

Fixes

- Fixed cron errors

1.0.5

(2016-10-18)

Improvement

- Updated docs

1.0.4

(2016-09-05)

Fixes

- Fixed an issue when badge description is breaking product list view

---

1.0.3

(2016-06-30)

Fixes

- Fixed an issue when cataloglabel is displaying in minicart after product was added to cart from product list

---

1.0.2

(2016-06-30)

Fixes

- Compatibility to Magento 2.1

1.0.0

(2016-02-17)

- Initial release