Elastic Search Ultimate Manual

Welcome to the Elastic Search Ultimate Guide!

Here you will find everything you need to set up a better search experience.

The Elastic Search Ultimate Extension includes Elastic Search, Search Spell-Correction, and Search AutoComplete, giving you a powerful way to search through your store.

First, Please find your extension in **My account / My Downloadable Products / View & Download**. Then, start with the section titled: Installation. It is best to follow our guide step-by-step to configure the best search results.

Go ahead, dive in!

Learn about Initial extension's setup:

- Installation
- Quick Start
- Upgrading
- Disabling

Welcome to the Elastic Search Ultimate Guide!

Here you will find everything you need to set up a better search experience.

First, please find your extension in your account in **My Downloadable Products** section. Then, start with Installation and Quick Start option. It is best to follow our step-by-step guide in order to configure the best search results.

Go ahead, dive in!

Learn about the initial setup:

- Installation
- Quick Start

Installation

In this article you will find two possible ways of our extension's installation.

Installation via composer (preferably)

We recommend this installation method because Composer doesn't allow to overwrite files.

1. Backup your store's database and web directory.
2. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
3. Copy the installation instructions from the page **My Downloadable Products / View & Download** to the SSH console.
4. Run command `php -f bin/magento module:enable Mirasvit_Core Mirasvit_Search Mirasvit_SearchMysql Mirasvit_SearchElastic Mirasvit_SearchAutocomplete Mirasvit_Misspell Mirasvit_SearchLanding Mirasvit_Report Mirasvit_SearchReport` to enable the extension.
5. Run command `php -f bin/magento setup:upgrade` to install the extension.
6. Run command `php -f bin/magento cache:clean` to clean the cache.
7. Deploy static view files
   ```
   rm -rf pub/static/*
   rm -rf var/view_preprocessed/*
   php -f bin/magento setup:static-content:deploy
   ```
8. Run search spell-correction reindex `php -f bin/magento indexer:reindex mst_misspell`
9. Go to **System / Search Indexes**, configure Product index and run the reindex for it.

Installation via direct file upload

You can also install the extension via direct files uploading.

1. Go to **My Downloadable Products / View & Download**
2. Unpack .zip package and copy contents to magento root directory
3. Run command `composer require elasticsearch/elasticsearch:~5.1` to install required libraries.
4. Run command `php -f bin/magento module:enable Mirasvit_Core Mirasvit_Search Mirasvit_SearchMysql Mirasvit_SearchElastic Mirasvit_SearchAutocomplete Mirasvit_Misspell Mirasvit_SearchLanding Mirasvit_Report Mirasvit_SearchReport` to enable the extension.
5. Run command `php -f bin/magento setup:upgrade` to install the extension.
6. Run command `php -f bin/magento cache:clean` to clean the cache.
7. Deploy static view files
   ```
   rm -rf pub/static/*
   rm -rf var/view_preprocessed/*
   php -f bin/magento setup:static-content:deploy
   ```
8. Run search spell-correction reindex `php -f bin/magento indexer:reindex`

`mst_misspell`

9. Go to **System / Search Indexes**, configure Product index and run the reindex for it.

Learn about the initial setup:

- Quick Start

Quick Start

As you've successfully completed installation of Elastic Search Ultimate, we will guide you through the main steps required to start efficiently using our extension.

1. **Make the content of your store searchable by configuring different Search Indexes.**
   - Go to **System / Search / Manage Indexes**
   - Enable or add new indexes that you want to be searchable
   - After that please reindex search indexes by running the command `php -f bin/magento indexer:reindex catalogsearch_fulltext`
2. **Configure Global Search Options and Additional Search Options**
3. **Display search results the way you want it with the Autocomplete option!**
   - Open Search Autocomplete configuration in **System / Search Management / Settings / Search Autocomplete**.
   - Enable required indexes and set proper results limit.
4. **Enable Search Spell Correction**

   Please reindex spell correction by running the command `php -f bin/magento indexer:reindex mst_misspell`

Please follow our guide step by step to get the best search result!

Core Search Settings

Here you can quickly navigate across all functionality settings we have. Please use the list below to navigate.

This section covers all topics, necessary for working with indices, and consists of the following subsections:

- **Search Indexes Settings**
  - Managing Indexes
  - Adding New Index
  - Product Index
  - Category Index

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).

- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

  Note

  This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

   - **Title** - title of the search index. It will be used as tab header at search display page.
   - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
     - **Magento Indexes**
       - Product
       - Category
       - CMS Page
       - Attribute
     - **Custom Search Indexes**
       - Wordpress Blog
     - **Mirasvit Extensions**
       - Blog MX
       - Knowledge Base
       - Gift Registry
     - **Magefun Blog Extension**
     - **Mageplaza Blog Extension**
     - **Ves Extensions**
       - Blog
       - Ves Brand
     - **Amasty**
       - Blog
       - FAQ
     - **Blackbird Content Manager**

    - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
    - **Active** - sets, whether index should be activated.
3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
    - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
    - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
  - If WordPress is installed on the same database, the correct value is `default`.
  - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

    Example

    Typical database connection should look similar to this:

```
    'db' => array(
    'table_prefix' => '',
    'connection' => array(
        'default' => array(
            'host' => 'localhost',
            'dbname' => 'store',
            'username' => 'root',
            'password' => 'password',
            'active' => '1',
        ),
    ),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

```
http://example.com/blog/{post_name}.html
http://example.com/blog/?p={ID}
http://example.com/{category_slug}/{post_name}.html
```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

  Note

  This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

   - **Title** - title of the search index. It will be used as tab header at search display page.
   - **Type** - shows index type (searchable content type). Some values of this field will trigger specific

options. Pick a link from type list below to know more:
    - **Magento Indexes**
        - Product
        - Category
        - CMS Page
        - Attribute
    - **Custom Search Indexes**
        - Wordpress Blog
    - **Mirasvit Extensions**
        - Blog MX
        - Knowledge Base
        - Gift Registry
    - **Magefun Blog Extension**
    - **Mageplaza Blog Extension**
    - **Ves Extensions**
        - Blog
        - Ves Brand
    - **Amasty**
        - Blog
        - FAQ
    - **Blackbird Content Manager**
  - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
  - **Active** - sets, whether index should be activated.
3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
  - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
  - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
  - **Relevance** - sorting by maximum relevance with search request
  - **Name** - sorting by names in alphanumeric order.
  - **Creation Time** - sorting by date of adding products to store
  - **Price 0-9** - sorting from cheapest to most expensive.
  - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
  - If WordPress is installed on the same database, the correct value is `default`.
  - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

    Example

    Typical database connection should look similar to this:

```
'db' => array(
'table_prefix' => '',
'connection' => array(
    'default' => array(
        'host' => 'localhost',
        'dbname' => 'store',
        'username' => 'root',
        'password' => 'password',
        'active' => '1',
    ),
),
```

```
),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

  ```
  http://example.com/blog/{post_name}.html
  http://example.com/blog/?p={ID}
  http://example.com/{category_slug}/{post_name}.html
  ```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).

- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

Note

This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

    - **Title** - title of the search index. It will be used as tab header at search display page.
    - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
        - **Magento Indexes**
            - Product
            - Category
            - CMS Page
            - Attribute
        - **Custom Search Indexes**
            - Wordpress Blog
        - **Mirasvit Extensions**
            - Blog MX
            - Knowledge Base
            - Gift Registry
        - **Magefun Blog Extension**
        - **Mageplaza Blog Extension**
        - **Ves Extensions**
            - Blog
            - Ves Brand
        - **Amasty**
            - Blog
            - FAQ
        - **Blackbird Content Manager**
    - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
    - **Active** - sets, whether index should be activated.
3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes,

where extension should conduct search. Each row consist of the following fields:
    - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
    - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
    - If WordPress is installed on the same database, the correct value is `default`.
    - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

      Example

      Typical database connection should look similar to this:

```
'db' => array(
'table_prefix' => '',
'connection' => array(
    'default' => array(
        'host' => 'localhost',
        'dbname' => 'store',
        'username' => 'root',
        'password' => 'password',
        'active' => '1',
    ),
),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

```
http://example.com/blog/{post_name}.html
http://example.com/blog/?p={ID}
http://example.com/{category_slug}/{post_name}.html
```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

- CMS Page Index
- Attribute Index
- Wordpress Blog Index

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

  Note

  This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

   - **Title** - title of the search index. It will be used as tab header at search display page.
   - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
     - **Magento Indexes**
       - Product
       - Category
       - CMS Page
       - Attribute
     - **Custom Search Indexes**

        - Wordpress Blog
    - **Mirasvit Extensions**
        - Blog MX
        - Knowledge Base
        - Gift Registry
    - **Magefun Blog Extension**
    - **Mageplaza Blog Extension**
    - **Ves Extensions**
        - Blog
        - Ves Brand
    - **Amasty**
        - Blog
        - FAQ
    - **Blackbird Content Manager**
  - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
  - **Active** - sets, whether index should be activated.
3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
  - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
  - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.

- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
  - If WordPress is installed on the same database, the correct value is `default`.
  - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

    Example

    Typical database connection should look similar to this:

```
    'db' => array(
    'table_prefix' => '',
    'connection' => array(
        'default' => array(
            'host' => 'localhost',
            'dbname' => 'store',
            'username' => 'root',
            'password' => 'password',
            'active' => '1',
        ),
    ),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

```
http://example.com/blog/{post_name}.html
http://example.com/blog/?p={ID}
http://example.com/{category_slug}/{post_name}.html
```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

> **Note**
>
> This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1.

To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:
    - **Title** - title of the search index. It will be used as tab header at search display page.
    - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
        - **Magento Indexes**
            - Product
            - Category
            - CMS Page
            - Attribute
        - **Custom Search Indexes**
            - Wordpress Blog
        - **Mirasvit Extensions**
            - Blog MX
            - Knowledge Base
            - Gift Registry
        - **Magefun Blog Extension**
        - **Mageplaza Blog Extension**
        - **Ves Extensions**
            - Blog
            - Ves Brand
        - **Amasty**
            - Blog
            - FAQ
        - **Blackbird Content Manager**
    - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
    - **Active** - sets, whether index should be activated.
3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
    - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
    - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
  - If WordPress is installed on the same database, the correct value is `default`.
  - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

    Example

    Typical database connection should look similar to this:

```
    'db' => array(
    'table_prefix' => '',
    'connection' => array(
        'default' => array(
            'host' => 'localhost',
            'dbname' => 'store',
```

```
            'username' => 'root',
            'password' => 'password',
            'active' => '1',
        ),
    ),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

  ```
  http://example.com/blog/{post_name}.html
  http://example.com/blog/?p={ID}
  http://example.com/{category_slug}/{post_name}.html
  ```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

  Note

  This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

   - **Title** - title of the search index. It will be used as tab header at search display page.
   - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
     - **Magento Indexes**
       - Product
       - Category
       - CMS Page
       - Attribute
     - **Custom Search Indexes**
       - Wordpress Blog
     - **Mirasvit Extensions**
       - Blog MX
       - Knowledge Base
       - Gift Registry
     - **Magefun Blog Extension**
     - **Mageplaza Blog Extension**
     - **Ves Extensions**
       - Blog
       - Ves Brand
     - **Amasty**
       - Blog
       - FAQ
     - **Blackbird Content Manager**
   - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
   - **Active** - sets, whether index should be activated.
3.

Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
    - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
    - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
    - If WordPress is installed on the same database, the correct value is `default`.
    - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

      Example

      Typical database connection should look similar to this:

```
'db' => array(
'table_prefix' => '',
'connection' => array(
    'default' => array(
        'host' => 'localhost',
        'dbname' => 'store',
        'username' => 'root',
        'password' => 'password',
        'active' => '1',
    ),
),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

```
http://example.com/blog/{post_name}.html
http://example.com/blog/?p={ID}
http://example.com/{category_slug}/{post_name}.html
```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

- Managing Indexes
- Adding New Index

- Product Index
- Category Index
- CMS Page Index
- Attribute Index
- Wordpress Blog Index

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

  Note

  This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

   - **Title** - title of the search index. It will be used as tab header at search display page.
   - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
     - **Magento Indexes**
       - Product
       - Category
       - CMS Page

        - Attribute
    - **Custom Search Indexes**
        - Wordpress Blog
    - **Mirasvit Extensions**
        - Blog MX
        - Knowledge Base
        - Gift Registry
    - **Magefun Blog Extension**
    - **Mageplaza Blog Extension**
    - **Ves Extensions**
        - Blog
        - Ves Brand
    - **Amasty**
        - Blog
        - FAQ
    - **Blackbird Content Manager**
  - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
  - **Active** - sets, whether index should be activated.
3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
   - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
   - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.
5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).

- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
  - If WordPress is installed on the same database, the correct value is `default`.
  - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

    Example

    Typical database connection should look similar to this:

```
    'db' => array(
    'table_prefix' => '',
    'connection' => array(
        'default' => array(
            'host' => 'localhost',
            'dbname' => 'store',
            'username' => 'root',
            'password' => 'password',
            'active' => '1',
        ),
    ),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

```
http://example.com/blog/{post_name}.html
http://example.com/blog/?p={ID}
http://example.com/{category_slug}/{post_name}.html
```

Back to top

Configure Search Indexes

Search Indexes are the most important part of your search subsystem. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document on your store, which would require considerable time and computing power.

This section covers all topics, necessary for working with indexes, and consists of the following subsections:

Managing Indexes

Our extension can combine all indexes, existing in your configuration, to boost search and give your customers the most relevant results. It brings them all to a single grid, located at **System -> Search Management -> Search Indexes**, from where you can configure them.

Each index, added to this grid, displayes the following properties:

- **Title** - title of the search index.
- **Type** - shows index type (searchable content type - read more at **Adding New Index** subsection).
- **Position** - the position of the index in the search results. Search results will be organized in tabs according to this property.
- **Status** - indicates, whether current index is ready for search. **Disabled** value means, that index will be excluded from search.

Additional **Action** column provides common actions, that can be performed directly from grid, such as:

- **Edit** - edit index settings (default action).
- **Reindex** - run manual reindexing for selected index.
- **Delete** - remove index from Mirasvit Search extension. %

> **Note**
>
> This action will completely remove this index from your store, so if you wish index to be excluded from search - just change its status to **Disable**.

Back to top

Adding and Configuring New Index

1. To add a new index to Mirasvit Search extension, go to **System -> Search Management -> Search Indexes** and press **Add New Index**.

2. Index record creation are divided into two stages: setting common settings and specific, which depend from their type. Common settings are shown in **General Settings** subsection:

    - **Title** - title of the search index. It will be used as tab header at search display page.
    - **Type** - shows index type (searchable content type). Some values of this field will trigger specific options. Pick a link from type list below to know more:
        - **Magento Indexes**
            - Product
            - Category
            - CMS Page
            - Attribute
        - **Custom Search Indexes**
            - Wordpress Blog
        - **Mirasvit Extensions**
            - Blog MX
            - Knowledge Base
            - Gift Registry
        - **Magefun Blog Extension**
        - **Mageplaza Blog Extension**
        - **Ves Extensions**
            - Blog
            - Ves Brand
        - **Amasty**
            - Blog
            - FAQ
        - **Blackbird Content Manager**
    - **Position** - the position of the index in the search results. Extension will sort tabs on search results page based on position.
    - **Active** - sets, whether index should be activated.

3. Press **Save and Continue Edit** to proceed to index configuration stage.

4. Add **Searchable Attributes** to the type-dependent option list, with rows corresponding to attributes, where extension should conduct search. Each row consist of the following fields:
    - **Attribute** - attribute name. It is picked from properties of selected index type. For example, if **Product** type is selected - then attributes would be **Product Name**, **SKU**, **Price**, **Tax Class** and so on.
    - **Weight** - sort order, which defines importance of each attribute for product relevancy. The maximum weight is 10 (highest priority), the minimum weight is 0(lowest priority). Each index type comes with a predefined set of searchable options, that will be displayed after completing the first stage. There should be **at least one searchable attribute**, otherwise search will not work properly.

5. **Properties** - type-dependent specific options section. Read more below, or pick a link from type values, described in **(2)** step.

6. Save index and activate it to include to search.

On installation three indexes will be automatically created and configured: **Product**, **Category** and **CMS Page** .

Back to top

Product Index

Product Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Specific options of this type will be shown on **Properies** section of Index edit page:

- **Search by Parent Categories Name** - include to search all parent categories (useful, when store has wide categories tree).
- **Search by child products** - include to search associated products from Bundled, Grouped and Configurable products.
- **Search by Product ID** - enable search by product id (entity_id attribute, which is not listed as searchable by Magento).
- **Search by custom options** - enable search by custom options (defined additionally to existing ones).
- **Push "out of stock" products to the end** - force sorting of search results by stocks inventory, so 'out of stocks' products will be displayed last.
- **Search only by active categories** - display only products, which are assigned to at least one active category
- **Force sort order by** - overrides default sort order of search results by one of these options:
    - **Relevance** - sorting by maximum relevance with search request
    - **Name** - sorting by names in alphanumeric order.
    - **Creation Time** - sorting by date of adding products to store
    - **Price 0-9** - sorting from cheapest to most expensive.
    - **Price 9-0** - sorting from expensive to cheapest.

Back to top

Category Index

Category Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's no specific options for this type of index.

Back to top

CMS Index

CMS Page Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

There's only one specific option for this type on **Properies** section of Index edit page:

- **Ignored CMS Pages** - defines, on which CMS pages search should not be performed. You can select zero or more pages here via checkbox drop-down list.

Back to top

Attribute Index

Unlike of other indexes, this one can be created only for specific attribute, which should be displayed as separate section in Search Results.

This attribute should be previously enabled for Advanced Search. It can be done at **Stores -> Attributes -> Product** grid. Pick up desired attribute, then jump to **Storefront Properties** subsection and then make them available for search by setting to **Yes** two options: **Use in Search** and **Visible in Advanced Search**.

Note

Attribute index can work only with attributes, that can be **indexed**, e. q. they belong to selectable type.

To see type of Product Attribute, visit **Stores -> Attributes -> Product** grid, pick up attribute record, and see **Catalog Input Type for Store Owner** field. Selectable types are **Multiple Select** and **Dropdown**. Only attributes of this type can be indexed.

If you wish to use attributes like **Author**, or similar, you have to make them selectable first, and then make them available for search as above.

After saving product you can configure Attribute Index for this attribute at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

Back to top

Wordpress Blog Search Index

Wordpress Blog Index can be created at **System -> Search Management -> Search Indexes** grid. Read more here about adding new index.

- **Database Connection Name** - connection name of the wordpress database.
  - If WordPress is installed on the same database, the correct value is `default`.
  - If WordPress is installed on the separate database, you need to create a new connection in file `app/etc/env.php`.

Example

Typical database connection should look similar to this:

```
    'db' => array(
    'table_prefix' => '',
    'connection' => array(
        'default' => array(
            'host' => 'localhost',
            'dbname' => 'store',
            'username' => 'root',
            'password' => 'password',
            'active' => '1',
        ),
    ),   ),
```

- **Table Prefix** - the prefix for the wordpress tables (wp_ by default).
- **Url Template** - the full URL for your posts with dynamical variables.

  Typical base urls should look like example below below:

  ```
  http://example.com/blog/{post_name}.html
  http://example.com/blog/?p={ID}
  http://example.com/{category_slug}/{post_name}.html
  ```

Back to top

Implementing Custom Search Index

Sometimes it's need to have specific type of Index, which is either not included to our extension, or belongs to some third-party extension. In this case custom index can be implemented, using the following instructions:

1. Clone the example module from repository http://github.com/mirasvit/module-search-extended
2. Go to `app/code/Mirasvit/SearchExtended/Index/` and rename subpath `Magento/Review/Review/` to the required one `([provider]/[module]/[entity])`
3. Change class names in file `app/code/Mirasvit/SearchExtended/Index/[provider]/[module]/[entity]/Index.p`
   - Rename class
   - Set your values to `getName()`, `getPrimaryKey()` and `getIdentifier()` methods
4. Configure the attributes you want to get in `getAttributes()` method
5. Change methods `buildSearchCollection()` and `getSearchableEntities()`
6. Change registration for new index in file `app/code/Mirasvit/SearchExtended/etc/di.xml`
7. Adjust layout file `app/code/Mirasvit/SearchExtended/view/frontend/layout/catalogsearch_result_`

Rename template name/path and adjust it
`/app/code/Mirasvit/SearchExtended/view/frontend/templates/index/magento/rev`

8. Enable module and Clear magento cache

If everything was correct, you can add index of your custom type like any regular index.

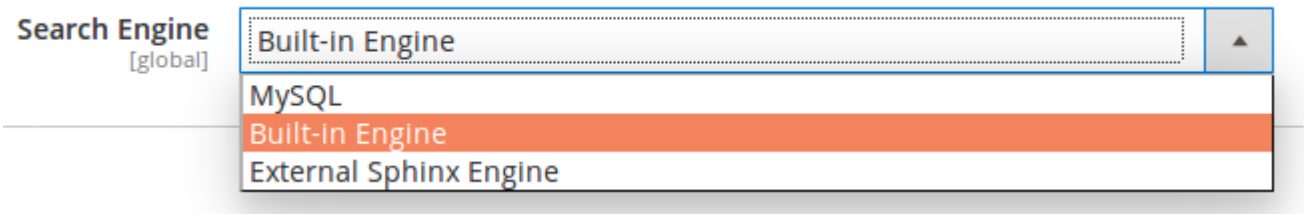
1. If you use SSU please go to: `/vendor/mirasvit/module-search-autocomplete/src/SearchAutocomplete/Model/Index` folder
2. Create folder/file structure `<Company>/<Extension>/<EntityType>.php` i.e. `/vendor/mirasvit/module-search-autocomplete/src/SearchAutocomplete/Model/Index/Ves/Blog/Post.php`
3. Open `/vendor/mirasvit/module-search-autocomplete/src/SearchAutocomplete/etc/di.xml` and add item to type `name="Mirasvit\SearchAutocomplete\Model\Index\Pool"` arguments

Configure Global Search Settings

This section describes, how you can customize and greatly improve the relevance of your search results by configuring Search Settings.

The most important part is **Global Search Configuration**. It is located at **System -> Search Management -> Settings -> Mirasvit Extensions -> Search**, and divided into the following sections:

- **Search Engine Configuration**
- **Search Settings**
- **Multi-store Search Result**



Search Engine Configuration

Our extension allows you to power up search either with default Magento search engine, or with external engine. Option **Search Engine** selects, which engine should be in charge, and has three possible values:

- **MySQL** - the native magento engine.
- **Built-in search engine** - will use an internal search algorithm of our extension.

Note

Built-in search engine mode **does not** require installation of Sphinx Engine on your server, but you will still receive the same features as with the Sphinx Engine. However, you can experience a slower search speed than with the Sphinx Engine (only for more then 20K products).

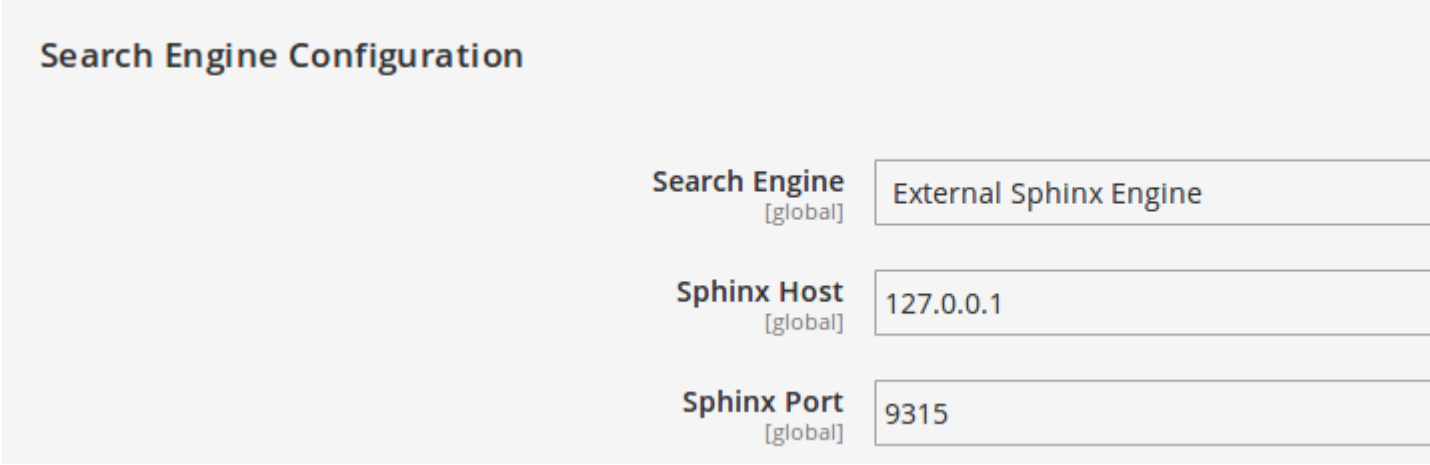
Third possible value depends from precise extension, that you're using. Mirasvit provides two search applications, that share this option, but support different search engines.

- **Sphinx Search Ultimate**

  Sphinx Search Ultimate, as it derives from its name, allows you to use Sphinx Engine on the dedicated server, or on the same server of your store.

  Sphinx is an open source full text search server, which features high performance, relevance (aka search quality), and integration simplicity. It's written in C++ and runs on Linux (RedHat, Ubuntu, etc), Windows, MacOS, Solaris, FreeBSD, and a few other systems. It is better used for stores with products quantity below 50k and without need of layered navigation or aggregated search requests. Read more about this engine key features.

  Sphinx Search Ultimate adds to the option **Global Search Configuration** -> **Search Engine Configuration** -> **Search Engine** possible value **External Sphinx Engine**.



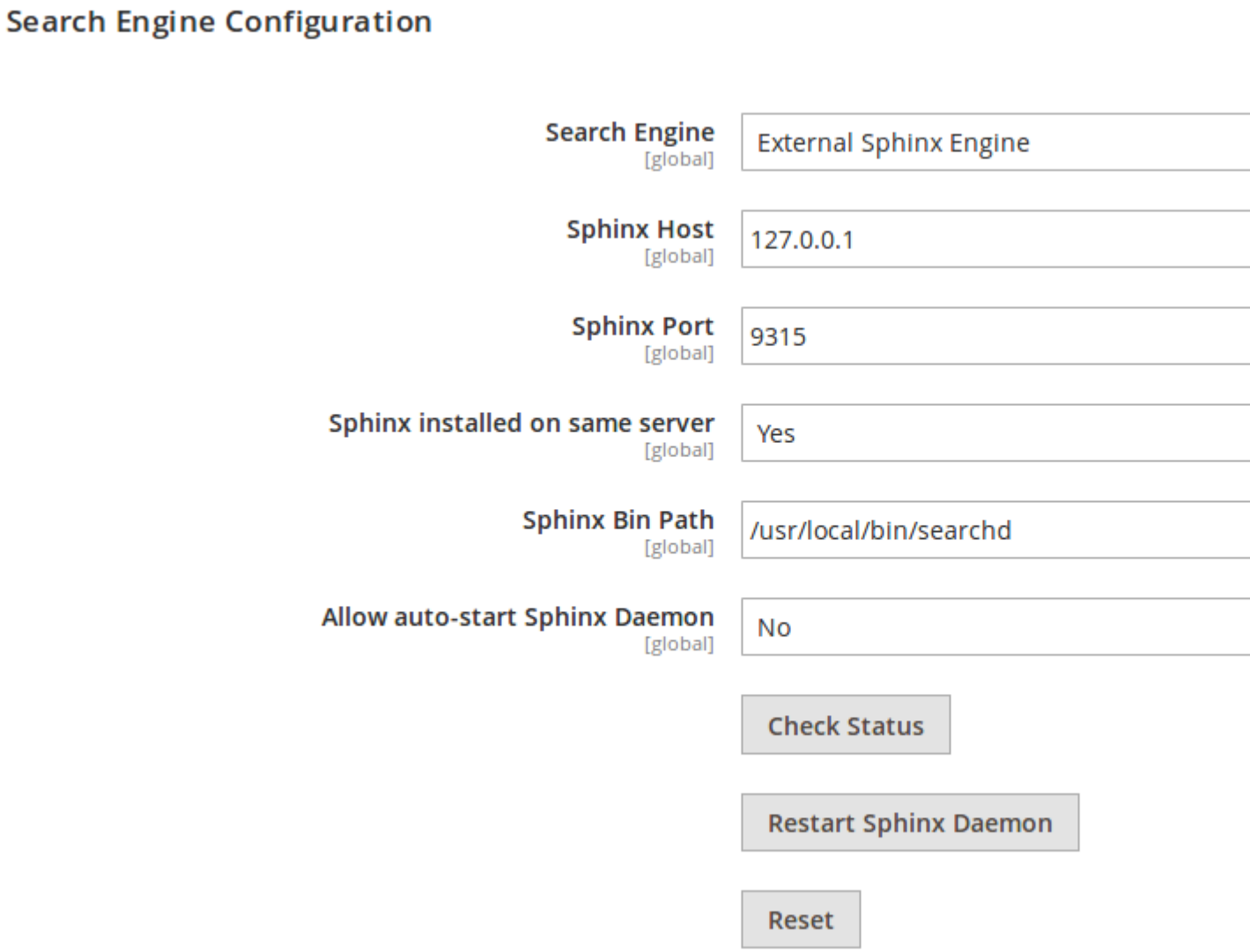
  Note

  To start with, please, make sure that you have installed Sphinx Search Engine. To do this please follow our installation guide

  External Sphinx Engine also triggers additional options for configuring and managing Sphinx Daemon:

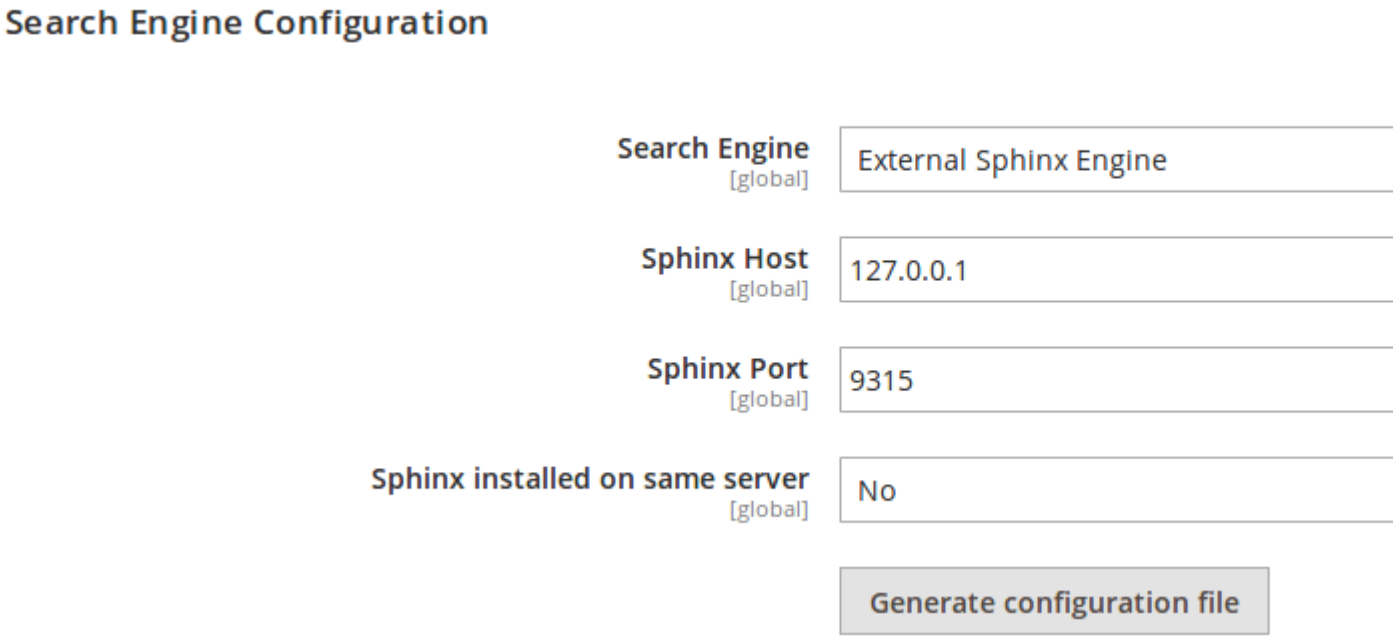
  - **Sphinx Host** - sphinx daemon host (localhost by default).
  - **Sphinx Port** - sphinx daemon port (any free port, like 9811, 9812).
  - **Sphinx installed on same server** - triggers appearance of additional features of Sphinx Daemon. Can have two different modes:

  For Sphinx installed on the same server with your Magento store :



- **Yes** - defines, that Sphinx works on the same server, as store and database. Triggers the following additional options and additional buttons, which allows to manage daemon:
  - **Sphinx Bin Path** - defines name and location of sphinx daemon. By default it's **searchd**.
  - **Allow auto-start Sphinx Daemon** - sets auto-starting daemon with Magento's store. Useful, when you can have unexpected server power-off (for example, for maintenance purpose).
  - **Check Status** - button, that allows to view current daemon status
  - **Restart Sphinx Daemon** - button, that allows to restart daemon directly from Magento Configuration pane.
  - **Reset** - button, that allows reset daemon current search task.

For Sphinx installed on the dedicated (remote) server :



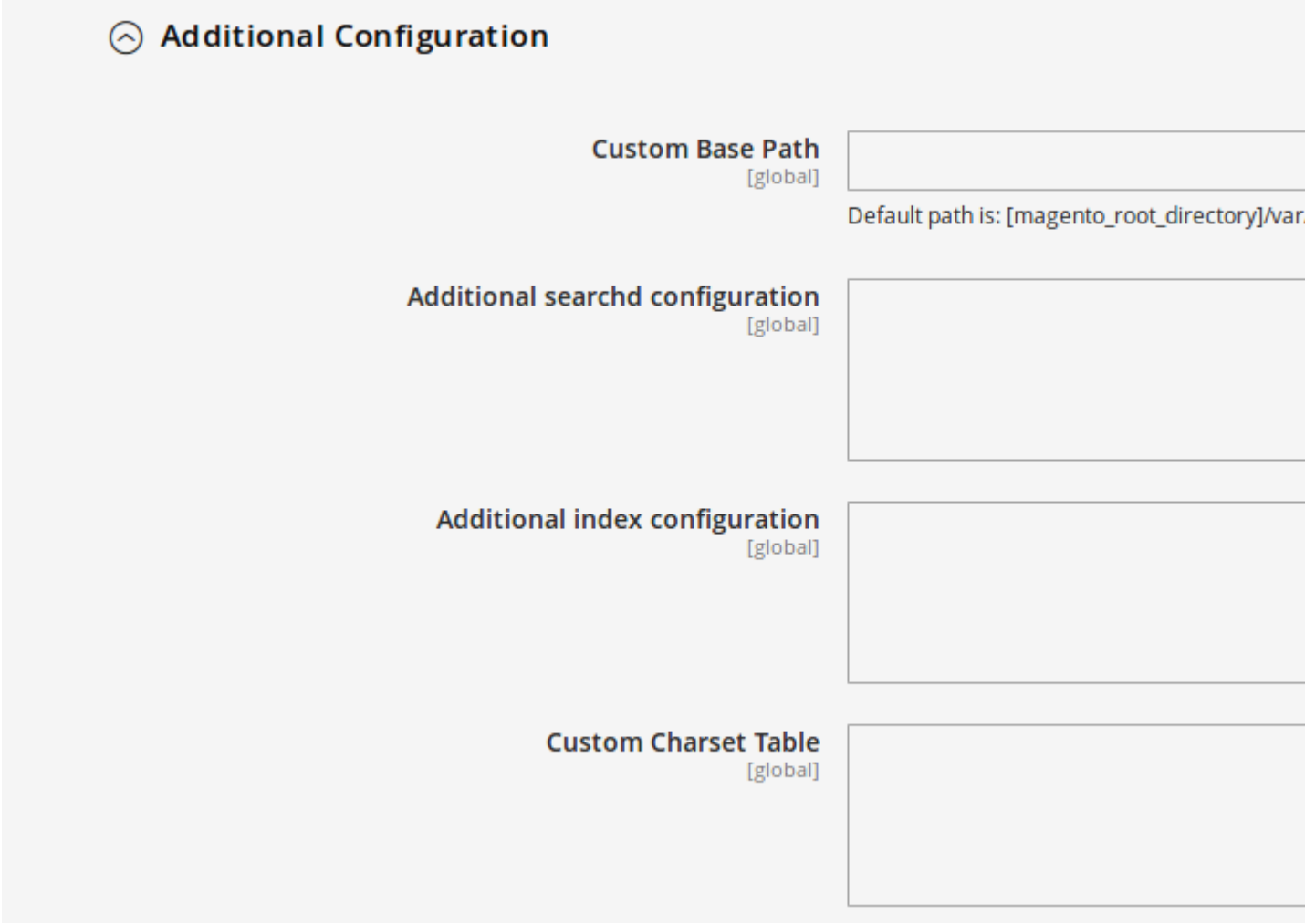
- **No** - defines, that Sphinx works on separate or dedicated server.
  - **Generate configuration file** - button, that allows to generate Sphinx config file to copy to your remote (dedicated) server.

Note

If you already installed **Sphinx Search engine** please learn more about connection with Sphinx Engine.

Search Engine Configuration section contains **Additional Configuration** subsection, visible for **External Sphinx engine** only. It allows you to tune up Sphinx configuration file, and contains the

following settings:



- **Custom Base Path** - defines custom path to Sphinx, if it was installed not to the default `[magento_root_directory]/var/sphinx/` location.
- **Additional searchd configuration** - defines additional parameters to `searchd` Search Daemon. Read more about it here.
- **Additional index configuration** - allows to add settings to the Sphinx index configuration. Read more about it here.
- **Custom Charset Table** - allows to add character sets to the Sphinx configuration file. Read more about it here.

- **Elastic Search Ultimate**

  Elastic Search Ultimate, as it derives from its name, allows you to use Elastic Engine on the dedicated server, or on the same server of your store.

  Elasticsearch is a distributed, RESTful search and analytics engine capable of solving a growing number of use cases, written on Java so it can be run virtually anywhere. It is best used for stores with more 50k of products and/or support of Layered Navigation. Read more about its key features.

  Elastic Search Ultimate adds to the option **Global Search Configuration** -> **Search Engine**

  Configuration -> **Search Engine** possible value **Elasticsearch Engine**

search_engine_elastic.png

. Image not found or type unknown

Note

To start with, please, make sure that you have installed Elastic Search Engine. To do this please follow our installation guide

Elasticsearch Engine also triggers additional options for configuring and managing Sphinx Daemon:

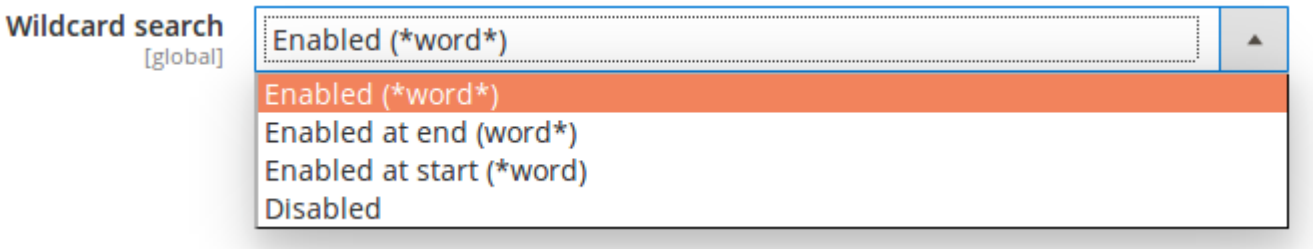
- **Elasticsearch Host** - elastic host (`127.0.0.1` by default).
- **Elasticsearch Port** - elastic port (any free port, but typically `9200`).
- **Elasticsearch Index Prefix** - specifies index name for current Magento store.

It also features two buttons, that allows you to check Elastic Search connection:

- **Check Status** - button, that allows to view current Elastic status
- **Reset** - button, that resets Elastic current search tasks.

Back to Top

Search Settings



- **Wildcard search** - allows customer to search the product by part of the word, marking unknown part with asterisk (*). There's four different wildcard modes available:
  - **Enabled (*word*)** - fully enables wildcards.
  - **Enabled at end (*word*)** - partially enables wildcards, allowing to search by first part of keyword.
  - **Enabled at start (*word*)** - partially enables wildcards, allowing to search by last part of keyword.
  - **Disabled** - totally disables wildcards.

    Note
    Wildcards enabling slightly reduces the relevance of search and increases the number of search results.

- **Enable redirect from 404 to search results** - if option is enabled, customer will be redirected to the store search results of the broken URL text instead of the "404 Not Found" page.
- **Redirect if there is a Single Result** - if the search query results only have one match, the customer will be immediately taken to to corrresponding product page.

- **Enable Google Sitelinks Search** - if option is enabled, the extension shows the Sitelink Search Box on the Google search results page. After enabling the option, the search box will be shown only after Google reindexing.
- **Enable search terms highlighting** - if option is enabled, search query word(s) will be highlighted in the search results.
- **Display Related Search Terms** - if option is enabled, related search terms will be displayed on the search result page.
- **Max number of items in the result** - sets the maximum number of items in the search result. Set 0 to disable limitation.
- **Wildcard Exceptions** - the list of keywords (characters) for which wildcard search can not apply.
- **Replace words in search query** - two-column list of auto-replace. When evaluating search extension will seek keywords from **Find word** columns, and automatically replace with the one from **Replace With** column. Column **Find word** can contain more than one keyword, separated by comma.
- **Not' words** - words from this list invert search. E. q. appearance of these words in search automatically treated as "exclude results with this word".
- **Long Tail Expressions** - allows you to receive the correct search results for words that contain dashes or any other non-alphabetic symbols. Read more in Long Tail Configuration section.
- **Minimum number of characters** - to search-specifies the minimum amount of characters, which triggers autocomplete drop-down list. It works only when **Autocomplete** is installed and enabled.
- **Match mode** - overrides default Magento mode of search with one of the following options:
    - **AND** - this mode is **default**. Elements (e. q. products, pages) matched only when all requested keywords are found in respective attributes.
    - **OR** - defines, that elements matched only when at least one of requested keywords is found.

Back to Top

Multi-store Search Result

You need multi-store search, when you have store-dependant elements - for example, products, that are visible only at specific storeviews - but wish to allows customers to search simultaneously on all your stores.

- **Enable Multi-Store Search Results** - enables multi-store search. Search results will be displayes in tabs, each of which corresponds with one of your storeviews. This option triggers one additional option:
    - **Stores** - allows you to select, which storeviews should be included to multi-store search.

Back to Top

Configure "Long-Tail" Search

This section describes the Long-Tail Search feature, that will allow you to have correct search results for words that contain dashes or other non-alphabetic symbols. You can also replace on-the fly the most typical errors customers can make in complex product names.

What is Long-Tail Search?

For example, we have a product `Canon PowerShot SX500 IS`. But customer can request `Canon PowerShot SX-500IS`, which default search will not find, because it differs from actual product label.

It's because Magento by default during reindex uses only correct product labels from database, and thus, index will contain only them - making products with complex names "ineligible" for search.

This is where "Long-tail" search come. During reindex and search this feature recognizes the keywords rather by pattern and replaces it either to the empty or some other characters, "correcting" customer's request on-the fly.

In example above the `SX500 IS` can be converted to the `SX500IS` and during the search, the `SX-500IS` also be converted to the `SX500IS` by replacing '-' symbol to empty char.

This way search will be able to find products by several combinations of spelling the product's name.

Back to Top

Configuring Long-Tail Search

Go to **System / Search Management / Settings / Mirasvit Extensions / Search**
In the section **Search Settings** go to the option **Long tail**.
There you can set up regular expressions to receive required search results.

- **Match Expression** - the regular expression(s) that parses words for further replacing.

  Parsing goes for search index, during an indexing process, and goes for search phrases during search. E.g. `/([a-zA-Z0-9]*[\-\/][a-zA-Z0-9]*[\-\/]*[a-zA-Z0-9]*)/`

- **Replace Expression** - the regular expression(s) to parse characters to be replaced. Parsing goes in the results of "Match Expression". E.g. `/[\-\/]/`
- **Replace Char** - the character to replace values founded by "Replace Expression". E.g. `empty value`.

Back to Top

Configuring Long-Tail Search

Here is some of most useful cases of long-tail search, implemented as corresponding rules.

- **Automatically remove '-' symbol from product names**

  Create a rule with the following parameters:

  - **Match Expression** - `/[a-zA-Z0-9]*-[a-zA-Z0-9]*/`
    Matched text: `SX500-123`, `GLX-11A`, `GLZX-VXV`, `GLZ/123`, `GLZV 123`, `CNC-PWR1`
  - **Replace Expression** `-/-/`

- **Replace Char** - empty
  Result text: `SX500123, GLX11A, GLZXVXV, GLZ/123, GLZV-123-123, CNCPWR1`

- **Automatically remove '-' and '/' symbols from product names**

  Create a rule with the following parameters:

  - **Match Expression** - `/[a-zA-Z0-9]*[ \-\/][a-zA-Z0-9]*/`
    Matched text: `SX500-123, GLX-11A, GLZX-VXV, GLZ/123, GLZV 123, CNC-PWR1`
  - **Replace Expression** - `/[ \-\/]/`
  - **Replace Char** - empty
    Result text: `SX500123, GLX11A, GLZXVXV, GLZ123, GLZV123, CNCPWR1`

- **Automatically make solid all products names with separators**

  Create a rule with the following parameters:

  - **Match Expression** - `/[a-zA-Z0-9]*[-\/][a-zA-Z0-9]*([-\/][a-zA-Z0-9]*)?/`
    Matched text: `SX500-123, GLX-11A, GLZX-VXV, GLZ/123, GLZV-123-123, CNC-PWR1`
  - **Replace Expression** - `/[-\/]/`
  - **Replace Char** - empty
    Result text: `SX500123, GLX11A, GLZXVXV, GLZ123, GLZV123123, CNCPWR1`

- **Automatically fix misspelled product's name**

  Create a rule with the following parameters:

  - **Match Expression** - `/([a-zA-Z0-9]*[\- ][a-zA-Z0-9]*[\-][a-zA-Z0-9]*)/`
    Matched text: `VHC68B-80, VHC-68B-80, VHC68B80`
  - **Replace Expression** - `/[\- ]/`
  - **Replace Char** - empty
    Result text: `VHC68B80`

Back to Top

Moving Long-Tail Expressions from M1 to M2

Long-Tail expressions, which are used in Search Sphinx for M1 and M2 sliightly differ.

In M1 Search Sphinx you can enter one or more expressions to match, separated by '|' character. In M2 you can not.

Consider the following expression for Search Sphinx for M1:

Example

Match Expression: `/[a-zA-Z0-9][ -/][a-zA-Z0-9]([ -/][a-zA-Z0-9]*)?/|/[a-zA-Z]{1,3}[0-9]{1,3}/`
Replace Expression:`/[ -/]/|/([a-zA-Z]{1,3})([0-9]{1,3})/`
Replace Char:`$1 $2`

It actually contains two separate regexps to match: `/[a-zA-Z0-9][ -/][a-zA-Z0-9]([ -/][a-zA-Z0-9]*)?/` and `/[a-zA-Z]{1,3}[0-9]{1,3}/` with respective separate expressions for replace.

You need either to reformat that expression, so it will match in single expression, or rewrite this rule as a set of two:

- **First rule**

  This rule will implement the first part of original M1 expression.

  - **Match Expression**: `/[a-zA-Z0-9][ -/][a-zA-Z0-9]([ -/][a-zA-Z0-9]*)?/`
  - **Replace Expression**:`/[ -/]/`
  - **Replace Char**:`$1 $2`
- **Second rule**

  This rule will implement the second part of original M1 expression.

  - **Match Expression**: `/[a-zA-Z]{1,3}[0-9]{1,3}/`
  - **Replace Expression**:`/([a-zA-Z]{1,3})([0-9]{1,3})/`
  - **Replace Char**:`$1 $2`

Back to Top

Manage Landing Pages

Landing search page is special search result page, with a static URL, where customers are redirected on using some search expression.

Let us have a large number of frequently asked (or just a promotional set) models of Samsung phones with black coat. So we create a separate promotional page, say, `http://store.com/black-samsung-phone.html`, and bind it to the search phrase "black samsung phone". Then, when customer will request a black Samsung phone, it will be immediately sent to your special page.

Also it supports the following logic .We create a separate promotional page, say, `http://store.com/black-samsung-phone.html`, and bind it to the search phrase "black samsung phone". When customer will go to this (specific) URL search results for "black samsung phone" will be immediately built on it.

All such a pages can be managed from **System -> Search Management -> Manage Landing Pages** grid.

Adding New Landing Page

- Go to **System / Search Management / Manage Landing Pages** and press **Add New** button.
- On creation page fill the following fieds:
  - **Query Text** - the key phrase, which should bring customer to landing page (ex. black samsung phone)
  - **URL Key** - relative path to landing page. For example, if URL key is `shoes/all`, then full URL would be `https://example/shoes/all/`.
  - **Active** - activates or deactivated redirect to landing page.
  - **Page Title** - overrides title of that page with yours.
  - **Meta Keywords** - meta keywords, that can be used by search crawlers.
  - **Meta Description** - meta description, that can be used by search crawlers.
  - **Layout Update XML** - overrides XML layout of landing page.
- Save and activate landing page.

Manage Synonyms

Synonyms are keywords with the same or similar meaning. All of them are located at **System -> Search Management -> Manage Synonyms** section.

You can either manually add synonyms, or import them from YAML-formatted file.

Adding New Synonym

- Go to **System -> Search Management -> Manage Synonyms** grid and press **Add New Synonym** button.
- On creation page, fill the following fields:
  - **Term** - is the keyword, which customer could enter to the search box, and which will be replaced with keyword
  - **Synonyms** - comma-separated list of synonyms. It should contain at least one keyword. Each of them should match the following requirements:
    1. It should consist of one word, and only of alphanumeric characters (e. q. without spaces, dashes, slashes and so on).
    2. It should have length, greater than 1 character.
    3. Max length of synonyms list equals 255 symbols.
  - **Store View** - allows to select, where defined synonyms will be applied.
- Save record.

Example

Assume, that we have on our stores a set of watches, and need them to be found on "clock" keyword. So we setup Synonym as:

```
  Term: clock  Synonyms: watch
```

Then, if customer issues "clock" query, all watches will be found.

Importing Synonyms

Our extension uses YAML file format for synonyms importing. It should resemble the following format:

```
-
    term: [TERM_1]
    synonyms: [SYN_1]
-
    term: [TERM_2]    synonyms: [SYN_2]
```

Name of this file should be equal to your language code in capital case. Codes can be found here, use column **639-1** for that.

Example

Let's create a synonyms file for English locale. Name of such a file would be `EN.yaml`, and it's content should be:

```
-
    term: abiogenesis
    synonyms: autogenesis,autogeny,spontaneous generation
-
    term: abject
    synonyms: low,miserable,scummy,scurvy,resigned,unhopeful
-
    term: abjection
    synonyms: abasement,degradation
-
    term: abjectly    synonyms: resignedly
```

To import synonyms, perform the following steps:

- Place your custom YAML file to `[magento_root]/` folder.
- Go to **System -> Search Management -> Manage Synonyms** and press **Import Synonyms** button.
- **Dictionary** field defines locale (language), to which synonyms are imported. All dictionaries should exist, and have at least one record, since imported data are appended to existing.
- **Store View** defines storeviews, where imported synonyms will be applied.
- Press **Import** to import and apply synonyms.

Manage Stopwords

Stopwords are words that have little lexical meaning or ambiguous meaning and are not useful during the search (ex. and, or, the, a, with, etc). Therefore, these words should be removed from search phrases to make them relevant.

You can either manually add stopwords, or import them from YAML-formatted file.

Adding New Stopword

- Go to **System -> Search Management -> Manage Stopwords** grid and press **Add New Stopword** button.
- On creation page, fill the following fields:
  - **Stopword** - is the keyword, which should be removed from search requests.
  - **Store View** - allows to select, where defined synonyms will be applied.
- Save record.

Importing Stopwords

Our extension uses YAML file format for stopwords importing. It should resemble the following format:

```
[ID_1]:[Stopword_1]
[ID_2]:[Stopword_2]
[ID_3]:[Stopword_3]
```

Name of this file should be equal to your language code in capital case. Codes can be found here, use column **639-1** for that.

Example

Let's create a stopwords file for English locale. Name of such a file would be `EN.yaml`, and it's content should be:

```
1:"but"
2:"now"
3:"what"    4:"except"
```

To import stopwords from such a file, perform the following steps:

- Place your custom YAML file to the special `[magento_root]/` folder.
- Go to **System -> Search Management -> Manage Stopwords** and press **Import Stopwords** button.
- **Dictionary** field defines locale (language), for which stopwords are imported. It is picked from the name of your YAML import files.
- **Store View** defines storeview, where imported stopwords should be applied.
- Press **Import** to import and apply stopwords.

Score Boost Rules

Score Boost Rules are powerful tool, which allows you to affect relevancy of search results, depending on certain conditions.

When Mirasvit Search extension builds search results, it groups them by indexes' position and their position in **System -> Search Management -> Settings -> Search Autocomplete -> Searchable Content**. Groups can include Products, Pages, Categories and so on - they can be defined in **System -> Search Management -> Search Indexes**.

But inside these groups items are listed strictly by their relevance to search query, which calculated for each item separately as **item rank**. Position in search results list depends from this rank value.

Score Boost Rules allows you to increase or decrease this rank depending on item properties, which allow you to move certain products to the top or botton of list, which is extremely useful for promotion and marketing purposes.

Creating a new Rule

To create a new Score Boost Rule, navigate to **System -> Search Management -> Score Boost Rules** section and press **Add New Rule** button.

You need to define the following properties to create a Rule/

- **Title** - sensical title of the Rule
- **Active** - whether Rule is active and should be applied to Search Results
- **Active (date)** - a time period, when Rule should apply to Search Results. Leave empty to have Rule always applicable.
- **Store** - storeviews, where current Rule should be applicable.
- **Score Factor** - score adjustment, that should be added or subtracted from rating, generated by search engine.
    - **Action** - action, that should be performed. Can have only two possible values.
        - **Increase By**
        - **Decrease By**
    - **Rank Adjustment** - numerical value, that should be added or subtracted from rating.
    - **Metric** - defines, how Rank Adjustment shall be used for adjustment. Can have two possible values:
        - **Points** - in this case Rank Adjustment just added to the actual rating.
        - **Times** - in this case actual rating is multiplied by Rank Adjustment. Used to rocket-jump products to the top (for example, promotional products).
    - **Parameter** - defines, which rating shall be adjusted by the Rule.
        - **Initial Score** - rating, which was generated by search engine.
        - **Product Popularity** - popularity rating, that is defined as quantity of orders with products, that meet conditions below.
        - **Product Rating** - product rating, that is defined as quantity of reviews for products, that meet conditions below

Conditions are broken into two parts.

- **Apply the rule only for following products** - allows you to define, which combination of products makes Rule apply.
- **Apply the rule only when the following conditions are met** - allows you to filter **Search Query**, to which Rule shall apply.

Both of them use the same pattern, as other rules in Magento 2, and enclosed into logical blocks `If ALL of these conditions are TRUE/FALSE` (products meet conditions, when all of them apply) or `If ANY of these conditions are TRUE/FALSE` (product shall meet only one of defined conditions).

Here are few useful examples, that demonstrate, how Score Boost Rules work.

Examples

- **Erin Recommends Promo**

  This example allows you to move products, that were recommended by your editorial board (it is defined by custom attribute **Erin Recommends**), to the top of search results.

  Title: `Erin Recommends Promo` **Score Factor**: **Increase by** `10` **points Initial Score Apply the rule only for following products:**

  - `Erin Recommends is Yes`

- **Analog Watches to the End**

  This rule drops to the very bottom all analog watches, when customer search includes "watch" keyword.

  Title: `Analog Watches to End` **Score Factor**: **Decrease by** `2` **times Initial Score Apply the rule only for following products:**

  - `Product Name contains analog` **Apply the rule only when the following conditions are met:**
  - `Search Query contains watch`

- **New Products Promo**

  This example allows you to uplift promotional products higher than others, but not necessary at the top.

  Title: `New Products Promo` **Score Factor**: **increase by** `5` **points Initial Score Apply the rule only for following products:**

  - `New is Yes`

Customize Search Weight

Our extension arranges relevance of found products using Global Settings. But sometimes (for example, for promotional purposes) you need to forcibly move one or more specific products to the top, or vice versa, to the bottom of search results.

It can be done via special option **Search Weight**, added by our extension to the general settings of the Product Edit Pages.

Joust Duffle Bag

Store View: All Store Views

Enable Product [website] — Yes

Attribute Set — Bag

Product Name * [store view] — Joust Duffle Bag

SKU * [global] — 24-MB01

Price * [global] — $ 34.00

Advanced Pricing

Tax Class [website] — Taxable Goods

Quantity [global] — 100

Advanced Inventory

Stock Status [global] — In Stock

Weight [global] — lbs — This item has wei

Categories [global] — Gear × Bags ×

Description [store view] — Show / Hide Editor

Paragraph Font

This weight is the relative position, where product will be placed on search result page. It ranges from 100 (product or category will always appear at the top of search results list) to -100 (product or category will always appear at the bottom of search results list).

Installing Elastic Search Engine

If you would like to use our extension with Elasticsearch Engine, you should install it first.

Elastic Search should be installed differently in different platforms. If you use UNIX or Linux-based system, you can use one of the following commands:

- `cat /etc/*-release`
- `cat /proc/version`
- `hostnamectl`

Note

Important Note: Elastic Search requires Java, so make sure **openJDK** or **Oracle JDK** is installed before proceed.

Depending on your platform, displayed by command above, you need to pick up one of the following procedures:

- **Ubuntu and other Debian-enabled OS**
- **CentOS and other RPM-enabled OS**
- **Install from gzip package**

If you have unusual configuration, or use non-Linux setup, please, refer to official user manual how to install the elastic engine in such case.

After installation is complete, refer to **How to check and manage Elastic Search Service** subsection.

Ubuntu and other Debian-enabled OS

If you have installed Ubuntu, or other system with Debian package manager, execute the following commands:

- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.deb`
- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.deb.sha512`
- `shasum -a 512 -c elasticsearch-6.2.2.deb.sha512`
- `sudo dpkg -i elasticsearch-6.2.2.deb`
- `sudo update-rc.d elasticsearch defaults 95 10`
- `sudo -i service elasticsearch start`

CentOS and other RPM-enabled OS

If you have installed CentOS, or other system with RPM package manager, execute the following commands:

- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.1.1.rpm`
- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.1.1.rpm.sha512`
- `shasum -a 512 -c elasticsearch-6.1.1.rpm.sha512`
- `sudo rpm --install elasticsearch-6.1.1.rpm`
- `sudo chkconfig --add elasticsearch`
- `sudo -i service elasticsearch start`

Install from gzip package

If you have Linux-based system, but not from above distributions, or you just wish to make it run on-demand, use the following commands:

- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.tar.gz`
- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.tar.gz.sha512`
- `shasum -a 512 -c elasticsearch-6.2.2.tar.gz.sha512`
- `tar -xzf elasticsearch-6.2.2.tar.gz`
- `cd elasticsearch-6.2.2/`
- `./bin/elasticsearch`

How to check and manage Elastic Search Service

Once you had installed Elastic Search using one of above procedures, you need to check, whether it actually installed and running. Use this command to get current status:

- `sudo -i service elasticsearch status`

Output should return: `elasticsearch is running`

You can also visit **Stores -> Configuration -> Mirasvit Extensions -> Search -> Search Engine Configuration**, then select **Elasticsearch Engine** at **Search Engine** option.

You will see a **Check status** button on displayed subpanel. If Elastic Search is properly installed, you will receive output like below:

Example

Elasticsearch is running.

```
name: nyYUXv5
cluster_name: elasticsearch
cluster_uuid: EFGeuFOBSP64M9q0N8ST2Q
version:
     number: 6.2.2
     build_hash: 10b1edd
     build_date: 2018-02-16T19:01:30.685723Z
```

```
     build_snapshot:
     lucene_version: 7.2.1
     minimum_wire_compatibility_version: 5.6.0
     minimum_index_compatibility_version: 5.0.0
tagline: You Know, for Search
_shards:
     total: 0
     successful: 0
     failed: 0
_all:
     primaries:
     total:
indices:
{"error":{"root_cause":[{"type":"index_not_found_exception","reason":"no such i
```

You can also send a request to your store's Elastic Search Port (see Connecting Elastic Search Engine). By default it is **9200**. If Elastic Search is properly installed, you will receive the following output:

Example

URL: http://store.com:9200/

```
{
  "name" : "nyYUXv5",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "EFGeuFOBSP64M9q0N8ST2Q",
  "version" : {
    "number" : "6.2.2",
    "build_hash" : "10b1edd",
    "build_date" : "2018-02-16T19:01:30.685723Z",
    "build_snapshot" : false,
    "lucene_version" : "7.2.1",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"}
```

If at least one of the tests above passed with correct output, you had successfully installed Elastic Search Engine on your store.

If you need to manually restart Elastic Search, use command `sudo -i service elasticsearch restart`.

If you need to manually stop Elastic Search, use command `sudo -i service elasticsearch stop`.

Connecting Elasticsearch Engine

Make sure, that you had installed Elastic Search prior to configuring it.

Note

Supported Elastic Search Engine versions are **5.2.+**, **5.5.+** or **6.+**.

If you had installed it, and checked, whether it is running, visit **System -> Search Management -> Settings** subsection.

1. Set **Elastic Search Engine** in option **Search Engine**, and subpanel with connection settings will appear:
    - **Elasticsearch Host** - Elastic Search host (localhost, or 127.0.0.1 by default).
    - **Elasticsearch Port** - Elastic Search Query port (9200 by default).
    - **Elasticsearch Index Prefix** - prefix for store index. Should be changed if you have multiple magento installations on server, each with respective prefix.
2. Check correctness of configuration by pressing **Check Status** button. It should return output, as described in **How to check and manage Elastic Search Service** subsection.
3. If parameters are incorrect, use **Reset** button to return to defaults.
4. Save configuration, if you sure, that configuration is fine.

Note

After configuration saving, you need to run reindex:

- At **System -> Search Management -> Search Indices** or
- Using CLI interface with a command `php bin/magento indexer:reindex catalogsearch_fulltext` from your store's root directory.

Installing Elastic Search Engine

If you would like to use our extension with Elasticsearch Engine, you should install it first.

Elastic Search should be installed differently in different platforms. If you use UNIX or Linux-based system, you can use one of the following commands:

- `cat /etc/*-release`
- `cat /proc/version`
- `hostnamectl`

Note

Important Note: Elastic Search requires Java, so make sure **openJDK** or **Oracle JDK** is installed before proceed.

Depending on your platform, displayed by command above, you need to pick up one of the following

procedures:

- **Ubuntu and other Debian-enabled OS**
- **CentOS and other RPM-enabled OS**
- **Install from gzip package**

If you have unusual configuration, or use non-Linux setup, please, refer to official user manual how to install the elastic engine in such case.

After installation is complete, refer to **How to check and manage Elastic Search Service** subsection.

Ubuntu and other Debian-enabled OS

If you have installed Ubuntu, or other system with Debian package manager, execute the following commands:

- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.deb`
- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.deb.sha512`
- `shasum -a 512 -c elasticsearch-6.2.2.deb.sha512`
- `sudo dpkg -i elasticsearch-6.2.2.deb`
- `sudo update-rc.d elasticsearch defaults 95 10`
- `sudo -i service elasticsearch start`

CentOS and other RPM-enabled OS

If you have installed CentOS, or other system with RPM package manager, execute the following commands:

- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.1.1.rpm`
- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.1.1.rpm.sha512`
- `shasum -a 512 -c elasticsearch-6.1.1.rpm.sha512`
- `sudo rpm --install elasticsearch-6.1.1.rpm`
- `sudo chkconfig --add elasticsearch`
- `sudo -i service elasticsearch start`

Install from gzip package

If you have Linux-based system, but not from above distributions, or you just wish to make it run on-demand, use the following commands:

- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.tar.gz`
- `wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.2.2.tar.gz.sha512`
- `shasum -a 512 -c elasticsearch-6.2.2.tar.gz.sha512`
- `tar -xzf elasticsearch-6.2.2.tar.gz`

- `cd elasticsearch-6.2.2/`
- `./bin/elasticsearch`

How to check and manage Elastic Search Service

Once you had installed Elastic Search using one of above procedures, you need to check, whether it actually installed and running. Use this command to get current status:

- `sudo -i service elasticsearch status`

Output should return: `elasticsearch is running`

You can also visit **Stores -> Configuration -> Mirasvit Extensions -> Search -> Search Engine Configuration**, then select **Elasticsearch Engine** at **Search Engine** option.

You will see a **Check status** button on displayed subpanel. If Elastic Search is properly installed, you will receive output like below:

Example

Elasticsearch is running.

```
name: nyYUXv5
cluster_name: elasticsearch
cluster_uuid: EFGeuFOBSP64M9q0N8ST2Q
version:
     number: 6.2.2
     build_hash: 10b1edd
     build_date: 2018-02-16T19:01:30.685723Z
     build_snapshot:
     lucene_version: 7.2.1
     minimum_wire_compatibility_version: 5.6.0
     minimum_index_compatibility_version: 5.0.0
tagline: You Know, for Search
_shards:
     total: 0
     successful: 0
     failed: 0
_all:
     primaries:
     total:
indices:
{"error":{"root_cause":[{"type":"index_not_found_exception","reason":"no such i
```

You can also send a request to your store's Elastic Search Port (see Connecting Elastic Search Engine). By default it is **9200**. If Elastic Search is properly installed, you will receive the following output:

Example

URL: http://store.com:9200/

```
{
  "name" : "nyYUXv5",
  "cluster_name" : "elasticsearch",
```

```
"cluster_uuid" : "EFGeuFOBSP64M9q0N8ST2Q",
"version" : {
  "number" : "6.2.2",
  "build_hash" : "10b1edd",
  "build_date" : "2018-02-16T19:01:30.685723Z",
  "build_snapshot" : false,
  "lucene_version" : "7.2.1",
  "minimum_wire_compatibility_version" : "5.6.0",
  "minimum_index_compatibility_version" : "5.0.0"
},
"tagline" : "You Know, for Search"}
```

If at least one of the tests above passed with correct output, you had successfully installed Elastic Search Engine on your store.

If you need to manually restart Elastic Search, use command `sudo -i service elasticsearch restart`.

If you need to manually stop Elastic Search, use command `sudo -i service elasticsearch stop`.

Configuring Autocomplete & Suggest

All Autocomplete settings are located at **System / Settings / Mirasvit Extensions / Search Autocomplete** section.

It consists of the following sections:

- **General Configuration**
- **Hot searches**

General Configuration

This section also breaks into lesser subsections, and contains the following options:

- **General Configuration** - defines basic application settings
  - **Minimum number of characters to search** - specifies the minimum amount of characters, which customer should enter to trigger autocomplete.
  - **The delay to start finding** - specifies delay between triggering autocomplete. (by option above) and beginning of actual search.

    Note

    Our extension actually begins to search for possible autocompletions and suggestions only when both conditions match:
    - customer had entered minimal required number of characters;
    - there were no actions during specified delay period.
- **Fast Mode** - Option allows to increase search speed due to the excluding Magento 2 from the autocomplete search at the initialization stage.

Note

- This option is available for Sphinx search and Elastic engines only;
- The disadvantages include the increased indexing time of the search index, and the lack of some search capabilities, such as a long-tail, wildcard exceptions etc.

- **Searchable content** - list of search indexes, where search is performed, and results displayed as autocomplete options. Indices are either taken from standard Magento, or if extension is installed as part of **Advanced Search Sphinx Pro** or **Sphinx Search Ultimate** - from corresponding **Indexes** grid.

  - **Index** - name of index, which can be included to autocomplete.
  - **Is Enabled** - includes current index to autocomplete
  - **Max Number of results** - the maximum number of results, which should be displayed in autocomplete drop-down widget.

  Note
  You can drag and drop rows in this list to define order, in which results from different indices will be displayed in autocomplete drop-down.

- **Enable TypeAhead** - enables auto-suggestion feature. Our extension collects information about most popular search queries and their results, groups them and stores separately. When autocomplete is triggered and **TypeAhead** option enabled, our application automatically searches for typed term and displays suggestion of found most relevant query.

  Tip
  Use Right Arrow button to quickly turn auto-suggestion to full autocomplete query and save autocomplete time.

- **Product Settings** - defines content and appearance of autocomplete individual product information cells.

  - **Show Price** - displays price of product.
  - **Show Thumbnail** - displays small thumbnail of product image.
  - **Show Rating** - displays number of reviews and approval rating (so-called star rating).
  - **Show Description** - displays short excerpt from product's description.
  - **Show SKU** - displays SKU of the product.
  - **Show "Add to cart"** - displays shortcut button for quick purchasing products.
  - **Optimize autocomplete view for small screen size** - allows optimization of autocomplete layout to small screen sizes. **Note:** may require additional style fixing at **Appearance** section.

- **Appearance** - contains only one field, which defines custom appearance of autocomplete widget.

  - **Additional CSS Styles** - custom CSS styles, that should be applied either to entire drop-down, or to individual product cells. It is extremely powerful tool, which allows you to fit our Autocomplete extension to almost any theme.

    Example

    To customize individual product cell in autocomplete drop-down, use the following expression:

```
.searchautocomplete__item-magento_catalog_product
{
    // Your extended styles}
```

It will be added to our stylesheet.

Hot Searches

Hot searches are the most popular queries, which were requested by customers. If current customer request includes such a query, autocomplete can highlight them and put to the top of drop-down.

- **Search queries** - allows to override Hot Searches by adding here special keywords (comma-separated), that should be counted as hot. It is very useful during promotional campaigns.
- **Ignored words** - allows to exclude from Hot Searches certain keywords. It is also a list of comma-separated words.
- **Max Number of queries** - the maximum allowed number of Hot Searches, which should be displayed on autocomplete drop-down.

FAQ

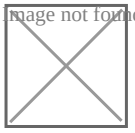
This section describes the most common problems, that customers report, and how they can be resolved:

How to make the autocomplete dropdown scrollable and smaller for mobile devices

For this navigate to the **Stores > Settings > Configuration > Mirasvit Extensions > Developer > CSS Settings** and add the css styles below to the **Additional CSS Styles** field:

```
@media screen and (max-width: 767px) {
    .searchautocomplete__autocomplete {
        max-height: 200px;
        overflow-y: scroll;
    }}
```

`max-width: 767px` - is the maximum width of the device for which these styles are applied.



How to make the autocomplete show a product price including or excluding tax

For this navigate to the **Stores > Settings > Configuration > Sales > Tax > Price Display Settings** and

switch the option **Display Product Prices In Catalog** to **Excluding Tax** - to display price without taxes or any other option to display price including tax.

Configure Search Spell Correction

All configuration options are located at **Store -> Configuration -> Mirasvit Extensions -> Search Spell Correction** section.

There's only two options for now. In both our extension analyzes customer's request and tries to find product, whose names are most close to the original request.

- **Enable spell correction** - enables automatic spelling correction.

  Example

  Let us assume, that your store have '*Samsung*' products in catalog.

  When customer accidentally misspells `Samsang phone`, default Magento search will return nothing, since you have no such a product.

  But with this option enabled, customer will be notified about potential misspell and will see results for the corrected search phrase `Samsung phone`.

- **Enable fallback search** - enables searching for partial request satisfaction, when there's no results for original request.

  Example

  Let us assume, that customer puts a phrase `red samsung phone` to the search, but you have only `samsung phone` product.

  If store has no such a product, default Magento search also will return nothing.

  But with this option enabled, customer will be notified about error, and and receive results by the correct search phrase `samsung phone`.

Reports

With detailed search reports, you are able to check how relative the search is to your customers.

From this information you will be able to fine-tune your search configuration so that your customers will be led to the products they need.

You can find reports in **System / Search Management / Reports**

You can check the Search Report by:

Search Volume

- Total Searches / Popularity
- Unique Searches - number of unique searches (search phrases)
- Users - number of unique sessions with searches
- Engagement % - the percent of users that opened product from search page

You can group or filter it

- By exact date
- By hour
- By day
- By week
- By month
- By year

Search Terms

- Total Searches
- Popularity
- Engagement %

Tip
You can export Reports to CSV, Excel or XML formats

Extension Upgrading

To upgrade the extension follow these steps:

1. Backup your store's database and web directory.
2. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
   If extension was installed via:
   - **Composer**: run command `` `composer require mirasvit/module-search-elastic-ultimate --update-with-dependencies` `` to update current extension with all dependencies.

Note

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

Run instead `composer update mirasvit/*` command. It will update all Mirasvit modules, installed on your store.

- **Direct file upload**: download new extension package from our store and copy contents to root Magento directory

3. Run command `php -f bin/magento setup:upgrade` to install updates.
4. Deploy static view files `rm -rf pub/static/*; rm -rf var/view_preprocessed/*; php -f bin/magento setup:static-content:deploy`
5. Run command `php -f bin/magento cache:clean` to clean the cache.
6. Reset Elasticsearch engine from backend and run the following command `php -f bin/magento indexer:reindex catalogsearch_fulltext` to reindex search indexes

Extension Disabling

Temporarily Disable

To temporarily disable the extension please follow these steps:

1. Login to the SSH console of your server and navigate the to root directory of the Magento 2 store.
2. Run the command `php -f bin/magento module:disable Mirasvit_Search Mirasvit_SearchMysql Mirasvit_SearchElastic Mirasvit_SearchAutocomplete Mirasvit_Misspell Mirasvit_SearchLanding Mirasvit_SearchReport` to disabled the extension.

Remove the Extension

To uninstall the extension please follow these steps:

1. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
2. Disable extension.
3. Run command `composer remove mirasvit/module-search-elastic-ultimate` to remove the extension.

Change Log

Search Spell Correction [mirasvit/module-misspell]

1.0.32

(2019-08-13)

Fixed

- Marketplace compatibility

1.0.31

(2019-05-27)

Fixed

- Generators cannot return values using “return”

1.0.29

(2019-02-12)

Fixed

- Allowed memory size error

1.0.28

(2018-11-29)

Fixed

- Compatibility with Magento 2.3

1.0.27

(2018-10-01)

Fixed

- ECHO

1.0.26

(2018-09-19)

Fixed

- Issue with first suggesting in some cases

1.0.24

(2018-05-31)

Fixed

- Issue with indexation cyrilic terms

1.0.23

(2018-04-11)

Fixed

- Issue with error 22003

1.0.22

(2017-12-25)

Improvements

- Integrated with Search Autocomplete
- Added Reindex validator

1.0.21

(2017-12-13)

Improvements

- Fallback search logic

1.0.20

(2017-11-17)

Fixed

- Issue with _cl table

1.0.19

(2017-10-26)

Fixed

- Possible issue with null values during indexation

1.0.18

(2017-09-28)

Fixed

- Issue with calculation number of results for suggested search phrase

---

1.0.17

(2017-09-26)

Fixed

- M2.2
- Issue with highlighting

1.0.16

(2017-08-09)

Fixed

- Issue with check zero result

1.0.15

(2017-07-12)

Fixed

- Issue with Changelog changes

1.0.14

(2017-07-10)

Improvements

- Fallback search logic

1.0.13

(2017-06-20)

Fixed

- Compatibility issue with Amasty Shopby

1.0.12

(2017-05-10)

Improvements

- Remove spell correction index if it disabled

1.0.11

(2017-04-11)

Improvements

- Switched to API interfaces

1.0.10

(2017-02-20)

Improvements

- Changed all string fuctions to mb_*

1.0.9

(2017-02-03)

Improvements

- Added Recurring setup script for check fulltext indices

1.0.8

(2016-11-21)

Improvements

- Compatibility with M 2.2.0

1.0.7

(2016-06-24)

Fixed

- Compatibility with Magento 2.1

1.0.6

(2016-06-16)

Fixed

- Fixed an issue with changing index mode for misspell index

1.0.5

(2016-04-27)

Improvements

- Improved extension performance
- i18n

Documentation

- Updated installation steps

1.0.4

(2016-02-23)

Fixed

- Fixed an issue with segmentation fault during reindex (PHP7)

1.0.3

(2016-02-07)

Documentation

- Added user manual

Search Autocomplete & Suggest [mirasvit/module-search-autocomplete]

1.1.96

(2019-08-08)

Fixed

- Issue with wrong layer

1.1.95

(2019-08-06)

Fixed

- Prices issue for multistore setup in 'Fast Mode'
- Product thumbnails issue in 'Fast Mode'

1.1.94

(2019-07-31)

Fixed

- Issue with autocomplete visibility, even if cart popoup was showed

1.1.93

(2019-07-30)

Features

- Fishpig Glossary index support

Fixed

- native search form active state
- nested environment emulation error
- reindex speedup
- Blinking autocomplete box with multiple search forms on the same page

---

1.1.92

(2019-06-19)

Fixed

- Render html entities on server side
- KB article typo in template
- Remove .active when on autocomplete miss focus

1.1.91

(2019-04-26)

Fixed

- conflict with IE 10

Improvements

- Added message after fast mode enable

1.1.90

(2019-04-24)

Fixed

- Ensure search autocomplete Fast Mode config file on reindex

- ## Display Fast mode indexes in correct order

1.1.89

(2019-04-12)

Fixed

- ## incorrect module conflict declaration

1.1.88

(2019-04-08)

Fixed

- Similar results in multiple attribute indexes

1.1.87

(2019-04-01)

Fixed

- Translations for search in stores with fast mode

Improvements

- ## Improved weighting, ability to use advansed search options, synonyms, stopwords

1.1.86

(2019-03-13)

Fixed

- Search in stores with fast mode

Search Elastic [mirasvit/module-search-elastic]

1.2.46

(2019-08-13)

Fixed

- Reindex issue when MSI disabled

1.2.45

(2019-07-31)

Fixed

- Advanced search issue
- Indexing with multi source inventory

1.2.44

(2019-07-04)

Fixed

- ## Reindex speed issue

Improvements

- Ability to reset search indexes for current store

1.2.43

(2019-06-27)

Fixed

-

Magento 2.3.2 compatibility

1.2.42

(2019-05-22)

Fixed

- ### missing filters when layered navigation multiselect enabled

1.2.41

(2019-05-15)

Fixed

- Magento EE indexation issue
- ### Search autocomplete fast mode

1.2.40

(2019-04-24)

Fixed

- ### Search Autocomplete Fast mode missing indexes

1.2.39

(2019-04-08)

Fixed

- Aggregations based on parent attributes
- Aggregations Dynamic bucket
- Catalog search and catalog categories downtime on search reindex

- ## Issue with synonyms in Search Autocomplete Fast mode

1.2.38

(2019-04-01)

Fixed

- Indexing issues

Improvements

- Ability to use advansed search options, synonyms, stopwords in fast mode

---

1.2.37

(2019-03-28)

Fixed

- support of magento 2.3.1

1.2.36

(2019-03-21)

Fixed

- search and filter using child products attributes

1.2.35

(2019-03-13)

Fixed

- Search in stores with fast mode

Search Landing Page [mirasvit/module-search-landing]

1.0.7

(2019-06-04)

Fixed

- Issue with different url keys for landing pages on different stores

---

1.0.6

(2018-11-29)

Fixed

- Compatibility with Magento 2.3

1.0.5

(2018-10-10)

Improvements

- Multistore

1.0.4

(2018-04-12)

Features

- Allow redirect by search term to url key

1.0.3

(2017-09-26)

Fixed

- M2.2

1.0.2

(2017-07-25)

Fixed

- Issue with static tests

1.0.1

(2017-05-03)

Fixed

- Issue with UI

1.0.0

(2017-05-03)

- Initial release

Search Report [mirasvit/module-search-report]

1.0.5

(2018-08-21)

Fixed

- Report settings do not work

1.0.4

(2018-04-20)

Fixed

- Issue with report by search terms

1.0.3

(2018-02-14)

Improvements

- Switched to new module-report version

Fixed

- Added details for secure cookies added details for secure cookies

---

1.0.2

(2017-09-26)

Fixed

- M2.2

1.0.1

(2017-07-21)

Fixed

- Possible issue with "Circular dependency"
