
Neos SEO Addon Documentation

Release dev-master

The Neos Team

Apr 10, 2019

Contents

1	Installation	3
2	Page title	5
3	Basic meta tags	7
4	Twitter Cards	9
5	Open Graph	11
6	Structured data	13
6.1	Breadcrumb	13
6.2	Website & Searchbox	13
6.3	Social profile	14
7	XML sitemap	15
8	Alternate Language Tag	17
9	Dynamic robots.txt	19
10	Disabling not needed features	21

This documentation covering version dev-master has been rendered at: Apr 10, 2019

CHAPTER 1

Installation

Install the package through composer:

```
composer require neos/seo
```


The default `<title>` tag rendering in the *Neos.Neos:Page* Fusion object is a “reverse breadcrumb” of the regular title field(s). This is done in *head.titleTag.default*.

A new field *titleOverride* is added to *Neos.Neos:Document* via the *Neos.Seo:TitleTagMixin*. The new field is used as the `<title>` tag content if it is filled (see *head.titleTag.content* in *Neos.Neos:Page*).

CHAPTER 3

Basic meta tags

The fields for keywords and description are added to *Neos.Neos:Document* via the *Neos.Seo:SeoMetaTagsMixin*. If they are filled in, `<meta>` tags for their contents will be rendered (see *head.metaTitleTag* and *head.metaDescriptionTag* in *Neos.Neos:Page*).

Two checkboxes allow to set the content for the `<meta name="robots">` tag to any combination of the possible values *follow*, *nofollow*, *index* and *noindex*.

CHAPTER 4

Twitter Cards

The *Neos.Seo:TwitterCardMixin* (added to *Neos.Neos:Document* by default) provides a new inspector tab to configure Twitter Cards on any document. If a Twitter Card is enabled, the related meta tags will be rendered as needed and useful.

The *twitter:site* handle can be configured with the setting *Neos.Seo.twitterCard.siteHandle* by providing a valid Twitter handle:

```
Neos:
  Seo:
    twitterCard:
      siteHandle: '@neoscms'
```

Check the documentation on <https://dev.twitter.com/cards/overview> for more on Twitter Cards.

The *Neos.Seo:OpenGraphMixin* (added to *Neos.Neos:Document* by default) provides a new inspector tab to configure Open Graph on any document. The Open Graph protocol enables any web page to become a rich object in a social graph. The essential ones are:

- *og:type*
- *og:title*
- *og:description*
- *og:image*
- *og:url*

In general Open Graph tags are just shown if they have given data, because otherwise Facebook for example will extract data for the generated view from the site itself. So fallbacks are not needed. If you are not satisfied with the generated view you should define your own. If a Open Graph Type is enabled, the related meta tags will be rendered according to following rules.

- *og:title* is only rendered if it includes data
- *og:description* will use *meta:description* as a fallback or show nothing
- *og:url* the URL of the document
- *og:image* is only rendered if it includes data

For more information please have a look at <http://ogp.me/>.

Structured data

This package provides basic prototypes for structured data and some already implemented elements. You can activate and configure them according to your needs.

The prototypes are also meant to be a very basic standard for your own structured elements. So feel free to adjust and reuse them.

When you activate structured data elements and have them on your live site you should use the Google Search Console to verify that they work as expected: <https://search.google.com/search-console>.

6.1 Breadcrumb

Search engines can output breadcrumbs by passing a structured data representation of a documents rootline. More information is available here: <https://developers.google.com/search/docs/data-types/breadcrumb>.

You can enable this feature with the following code:

```
prototype(Neos.Seo.StructuredData.Container).breadcrumb.@if.enabled = true
```

6.2 Website & Searchbox

To output general information including url and name of your site you can enable the *Website* object:

```
prototype(Neos.Seo.StructuredData.Container).website.@if.enabled = true
```

Additionally if you have a local search on your site you can enable the *Sitelinks Searchbox* object. More information is available here: <https://developers.google.com/search/docs/data-types/sitelinks-searchbox>

You can enable this feature with the following code and by adjusting *targetNode*:

```
prototype(Neos.Seo.StructuredData.Website) {  
    searchAction.targetNode = §{<reference to your search page>}  
}
```

6.3 Social profile

Search engines can output information about the social profile by passing a structured data representation of a documents rootline. Depending whether the site is run by a person or organization some parts need to be configured.

More information is available here: * <https://developers.google.com/search/docs/data-types/social-profile> * <https://developers.google.com/search/docs/data-types/logo>

You can enable this feature with the following code:

```
prototype(Neos.Seo.StructuredData.Container).socialProfile.@if.enabled = true
```

To adjust the profile configure this according to your requirements in your *Settings.yaml*:

```
Neos:
  Seo:
    socialProfile:
      type: 'set either to "Person" or "Organization"'
      logo: 'resource://Vendor.Site/Public/Images/MyLogo.png'
      profiles:
        twitter: 'your twitter name'
        facebook: 'your facebook name'
        instagram: 'your instagram name'
        linkedIn: 'your linkedin name'
        youTube: 'your YouTube channel identifier'
```

XML sitemap

The generation of an XML sitemap to submit to search engines can be used as follows:

The change frequency and priority for each sitemap entry are used as specified in the respective fields added to the SEO tab in the inspector of *Neos.Neos:Document* nodes via the *Neos.Seo:XmlSitemapMixin*. For priority the default value is 0.5 (neutral) and the change frequency is omitted unless specified.

The necessary route to make the sitemap available is automatically included via *Settings.yaml* and will provide the sitemap via *your.domain/sitemap.xml*. See Settings on how to disable or change the route.

To include contained images of pages in the xml sitemap use the following fusion code:

```
prototype(Neos.Seo:XmlSitemap) {
    body.includeImageUrls = true
}
```

Be aware that the sitemap will output all images referenced in a page and it's content. If you reference images that should not render in the frontend you might need to adjust the sitemap according to your needs.

By default all shortcuts are ignored in the sitemap. They inherit from the prototype *Neos.Seo:NoindexMixin*. If you have other document types that should not appear in the sitemap you can also let them inherit from that prototype.

To include alternate language links of pages in the xml sitemap use the following fusion code:

```
prototype(Neos.Seo:XmlSitemap) {
    body.includeAlternateLanguageLinks = true
}
```

Be aware of possible performance issues. Rendering the sitemap with all optional features might be slow for larger installations and needs an optimized *XmlSitemapImplementation* which could use ElasticSearch for example. Alternatively you can change the caching behavior and have a cron job that recreates the sitemap for example once per day.

Alternate Language Tag

The *Alternate Language Tag* provides information that the site is also available in other languages. By default the tags are rendered with the *Neos.Neos:DimensionMenu* and the *language* dimension. Given the Neos Demo Site Package as an example the rendered tags for the homepage would be.

```
<link rel="alternate" hreflang="en_US" href="http://neos.dev/" />
<link rel="alternate" hreflang="en_UK" href="http://neos.dev/uk/" />
```

According to the following dimension settings, there would be a lot more tags expected. However only two variants of the homepage exists, thus only *en_US* and its fallback *en_UK* are rendered.

In case the dimension that contains the language is not named *language* you have to set the alternative name with the property *ContentRepository.dimensionTypes.language*.

```
ContentRepository:
  contentDimensions:
    'language':
      label: 'Language'
      icon: 'icon-language'
      default: 'en_US'
      defaultPreset: 'en_US'
      presets:
        'all': ~
        'en_US':
          label: 'English (US)'
          values: ['en_US']
          uriSegment: 'en'
        'en_UK':
          label: 'English (UK)'
          values: ['en_UK', 'en_US']
          uriSegment: 'uk'
        'de':
          label: 'German'
          values: ['de']
          uriSegment: 'de'
        'fr':
          label: 'French'
          values: ['fr']
          uriSegment: 'fr'
        'nl':
```

(continues on next page)

(continued from previous page)

```
    label: 'Dutch'
    values: ['nl', 'de']
    uriSegment: 'nl'
  'dk':
    label: 'Danish'
    values: ['dk']
    uriSegment: 'dk'
  'lv':
    label: 'Latvian'
    values: ['lv']
    uriSegment: 'lv'
dimensionTypes:
  language: 'language'
```

Dynamic robots.txt

To activate the automatic *robots.txt* you have to delete the *robots.txt* inside the */Web* folder. You also have to edit the *.htaccess*:: Change the line *RewriteRule ^(_Resources/Packages/robots.txt|favicon.ico) - [L]* to *RewriteRule ^(_Resources/Packages/|favicon.ico) - [L]*.

If you don't want to delete 'robots.txt' after every update, you should add following lines to your '.htaccess':

```
# Use Neos robots.txt RewriteCond %{REQUEST_URI} ^/robots.txt RewriteRule (.*) index.php [L]
```

If you use nginx you should disable the following entry if you have it:

```
location = /robots.txt { allow all; log_not_found off; access_log off;
}
```

If you only want to render a subset of the available language dimensions (e.g., if the content is not yet ready) you can configure this in the *Settings.yaml*:

```
Neos:
  Seo:
    robotsTxt:
      # Activate only English and German
      dimensionsPresets: ['en', 'de']
```

You can also add your own definitions to the *robots.txt*. They can be passed by adding them to definitions array. For example to block the GoogleBot from a directory use this fusion code:

```
prototype(Neos.Seo.RobotsTxt) {
  data {
    userAgentGoogleBot = 'User-agent: Googlebot'
    disallowAll = 'Disallow: /private'
    disallowAll.@position = 'after userAgentGoogleBot'
  }
}
```

You should work with the position argument to ensure that everything is where you want it. By default a definition is preconfigured that blocks */neos* for all user agents.

CHAPTER 10

Disabling not needed features

The package provides a number of mixins to help rendering SEO metadata. By default, they are enabled in the *Configuration/NodeTypes.yaml* file, along with an inspector tab:

```
'Neos.Neos:Document':
  superTypes:
    'Neos.Seo:TitleTagMixin': true
    'Neos.Seo:SeoMetaTagsMixin': true
    'Neos.Seo:TwitterCardMixin': true
    'Neos.Seo:CanonicalLinkMixin': true
    'Neos.Seo:OpenGraphMixin': true
    'Neos.Seo:XmlSitemapMixin': true
  ui:
    inspector:
      tabs:
        seo:
          label: 'Neos.Seo:NodeTypes.Document:tabs.seo'
          position: 30
          icon: 'icon-bullseye'

'Neos.Neos:Shortcut':
  superTypes:
    'Neos.Seo:TitleTagMixin': false
    'Neos.Seo:SeoMetaTagsMixin': false
    'Neos.Seo:NoindexMixin': true
    'Neos.Seo:TwitterCardMixin': false
    'Neos.Seo:CanonicalLinkMixin': false
    'Neos.Seo:OpenGraphMixin': false
    'Neos.Seo:XmlSitemapMixin': false
```

Then to enable rendering of all SEO meta tags, the following code is used:

```
prototype(Neos.Neos:Page) {
  htmlTag.attributes.lang = Neos.Seo:LangAttribute
  head {
    titleTag = Neos.Seo:TitleTag
    metaDescriptionTag = Neos.Seo:MetaDescriptionTag
    metaKeywordsTag = Neos.Seo:MetaKeywordsTag
    metaRobotsTag = Neos.Seo:MetaRobotsTag
    canonicalLink = Neos.Seo:CanonicalLink
```

(continues on next page)

(continued from previous page)

```
alternateLanguageLinks = Neos.Seo:AlternateLanguageLinks
twitterCard = Neos.Seo:TwitterCard
openGraphMetaTags = Neos.Seo:OpenGraphMetaTags
}
}
```

If not all of the features are needed in a project, they can be disabled as needed. This example removes OpenGraph support.

Packages/Sites/Acme.AcmeCom/Configuration/NodeTypes.yaml:

```
'Neos.Neos:Document':
  superTypes:
    'Neos.Seo:OpenGraphMixin': false
```

Packages/Sites/Acme.AcmeCom/Resources/Private/Fusion/Root.fusion:

```
prototype (Neos.Neos:Page) .head.openGraphMetaTags >
```