Learn Neos

Fusion

In a Nutshell

for Neos 4.3 LTS

Download digital version, give feedback, report errors

http://bit.ly/fusion-nuts

Table of Contents

Version 4.3.0

Comments

```
# Line comment
// Another line comment
/*
 * Multi-line comment
 */
```

Simple Values

```
book = "The guide"
answer = 42
enableImprobabilityDrive = true
```

Expressions

```
answer = ${Math.floor(85 / 2)}
title = ${q(node).property('title')}
```

Objects

```
output = Neos.Fusion:Value {
  value = 'Hello, world!'
}
```

Property Definitions

```
output.world = "world!"
output {
  hello = "Hello"
}
prototype(Acme.Demo:Book).title = "Fusion – In a Nutshell"
```

Prototype Extension

```
prototype(Acme.Demo:Book) < prototype(Neos.Fusion:Template) {
  templatePath = 'Book.html'
}
```

Prototype Nesting

```
prototype(Acme.Demo:Book) {
  # Specify properties for all Tag objects rendered inside Book
  prototype(Neos.Fusion:Tag).attributes.class = 'book'
}
```

Clear Property

```
output.world >
```

Includes

```
include: SingleFile.fusion
include: 'Component/**/*.fusion'
include: "resource://Acme.Demo/Private/Fusion/Book.fusion"
```

Domain Specific Language

```
value = dslIdentifier`special meaning for my preprocessor`
```

Namespaces

```
namespace d=Acme.Demo
output = d:Book
```

@if.[key]

Conditional rendering of object (renders `null` if any condition is `false`)

```
image = Neos.Neos:ImageUri {
  asset = ${q(node).property('image')}
  @if.hasAsset = ${q(node).property('image') ? true : false}
}
```

@process.[key]

Apply processors after evaluation

```
example.@process.myName = ${String.toUpperCase(value)}
```

@context.[key]

Set or override context variable values by named key

```
example.@context.myVariable = ${q(node).parent()}
```

@apply.[key]

Dynamically define properties from an associative array

```
teasers = Neos.Fusion:Loop {
    collection = ${teaserList}
    itemName = 'teaser'
    itemRenderer = Vendor.Site:Teaser {
        @apply.teaser = ${teaser}
        fixedProperty = 'Foo'
    }
}
```

@position

Order definitions in `Array`, `@process` or `Case`

```
<numerical-order> | start (<weight>) | end (<weight>) | before <key> (<weight>) | after <key> (<weight>)
```

```
output = Neos.Fusion:Array {
  key1 = 'First'
  key1.@position = 'start 10'
  key2 = 'Second'
  key2.@position = 'before key1'
  key3 = 'Third'
  key3.@position = 'start 20'
}
# output: ThirdSecondFirst
```

@cache

Define caching behavior for an object

mode	string	embed | cached | dynamic | uncached
entryIdentifier	array	Named values to build the entry identifier (which is used to identify each cache entry – different content needs a different identifier)
entryTags	array	List of tags for the cache entry (which is used to flush entries when data changes)
entryDiscriminator	string | boolean	Discriminator for dynamic cached objects (which is used to identify separate cache entries on evaluation – e.g. by using request arguments in an expression). If the expression evaluates to false the object will be uncached.
context	array	List of context variable names to store for modes cached or dynamic
maximumLifetime	integer	Maximum lifetime for nearest cached entry: null (default), 0 (unlimited) or n seconds

```
cachedExample.@cache {
  mode = 'cached'
  entryIdentifier.node = ${node}
  entryTags.nodeType = ${Neos.Caching.nodeTypeTag('My.Site:Post')}
}
```

```
dynamicExample.@cache {
  mode = 'dynamic'
  entryIdentifier.node = ${node}
  entryDiscriminator = ${request.arguments.pagination}
  context {
    1 = 'node'
    2 = 'documentNode'
  }
  entryTags.parents = ${Neos.Caching.nodeTag(q(node).parents())}
}
```

Tags flushed by node changes:

Everything | NodeType _ [My.Site:NodeTypeName] (for all super types) | Node _ [identifier] | DescendantOf _ [identifier]

@class

Set implementation class for object

```
@class = 'Acme\\Demo\\Fusion\\MyObjectImplementation'
```

@exceptionHandler

Set exception handler class for an object that catches exceptions during rendering.

See implementations of AbstractRenderingExceptionHandler for existing handlers.

Array deprecated

Render multiple ordered definitions, returns string

[key]	string	A nested definition that evaluates to a string
[key].@position		Define the ordering of the element
@ignoreProperties	array	List of properties ignored by rendering

```
output = Neos.Fusion:Array {
  world = 'world!'
  hello = Neos.Fusion:Value {
    value = 'Hello'
    @position = 'start'
  }
  goodbye = 'Bye!'
  @ignoreProperties = ${['goodbye']}
}
# Rendered output: Helloworld!
```

Neos.Fusion:Array has been renamed to Neos.Fusion:Join, which adds a @glue meta property. The old name is deprecated and will be removed in a future major version.

Attributes

Render HTML tag attributes in an extensible way

[key]	mixed	The attribute name [key] and value, array values will be joined with whitespace
@allowEmpty	boolean	Whether empty attributes (HTML5 syntax) should be allowed

```
output = Neos.Fusion:Attributes {
  id = 'message'
  class = Neos.Fusion:RawArray {
    alert = 'alert'
    alert-error = 'alert-error'
  }
}
```

Augmenter

Modify HTML content to add attributes (standalone or as a processor)

content	string	HTML content to be augmented

fallbackTagName	string	Adds single tag around content if none or multiple are present, defaults to div
[key]	mixed	Nested definitions will be added as HTML attributes

```
output = Neos.Fusion:Augmenter {
  content = '<p>Lorem ipsum</p>'
  data-identifier = ${node.identifier}
}
```

CanRender new

Check whether a Fusion prototype can be rendered

type	string	The prototype name that is checked

```
canRender = Neos.Fusion:CanRender {
  type = 'My.Package:Prototype'
}
```

Case

Conditionally render one of multiple object definitions or paths

[key]	object	Each item of the case is a Neos.Fusion:Matcher object
[key].condition	boolean	The first condition that evaluates to true will be used (required)
[key].type	string	Object type to render (as string)
[key].element	array	Properties for the object instance (when using type)
[key].renderPath	string	Path to render (overrules type)
[key].renderer	mixed	Simple value, expression or object (overrules renderPath and type)
[key].@position		Define the ordering of the condition

```
page.body.content.header = Neos.Fusion:Case {
  landingPage {
    condition = ${q(node).property('landingPage')}
    type = 'My.Package:LandingPage'
  }
  default {
    condition = true
    @position = 'end'
    renderer = My.Package:DefaultPage {
      myProperty = 'foo'
    }
  }
}
```

Collection deprecated

Render each item in a collection, returns string

collection	array	The collection to render (Iterable, required)
itemName	string	Context variable name for each item value (required)
itemKey	string	Context variable name for each item key (optional)
iterationName	string	Context variable name for iteration information with properties index, cycle, isFirst, isLast
itemRenderer | content	mixed	The item rendering definition: simple value, expression or object (required)

```
output = Neos.Fusion:Collection {
  collection = ${q(node).children()}
  itemName = 'childNode'
  itemRenderer = Neos.Fusion:Tag {
    tagName = 'p'
    content = ${q(childNode).property('title')}
  }
}
```

Neos.Fusion:Collection has been renamed to Neos.Fusion:Loop, which adds a @glue meta property and uses property items instead of collection. The old name is deprecated and will be removed in a future major version.

Component

Atomic Fusion component: evaluates properties as `props` and evaluates to `renderer`

renderer	mixed	Rendering definition for evaluation (required)
[key]	mixed	Properties to evaluate as `props` context variable

```
prototype(My.Site:Component.Headline) < prototype(Neos.Fusion:Component) {
  # props
  level = 'h1'
  text = "Hey, I'm a headline"

  # rendering definition
  renderer = Neos.Fusion:Tag {
    tagName = ${props.level}
    content = ${props.text}
  }
}
```

DataStructure new

Evaluate nested definitions as an array (opposed to string for Neos.Fusion:Join), returns `array`

[key]	mixed	A nested definition (simple value, expression or object), [key] will be used for the resulting array key
[key].@position	string/integer	Define the ordering of the element

```
output = Neos.Fusion:Attributes {
  class = Neos.Fusion:DataStructure {
    alert-error = 'alert-error'
    alert = 'alert'
    alert.@position = 'start'
  }
}
```

Debug

Output all properties or `value` for debugging

value	mixed	Single value to debug (optional)
title	string	Title for the debug output (optional)
plaintext	boolean	Output result as plaintext (defaults to false)
[key]	mixed	Properties to debug

```
myValue.@process.debug = Neos.Fusion:Debug {
  title = 'Debug the title prop of my page'
  value = ${q(node).property('title')}
}
```

Http.Message

Generate a HTTP response message (based on Array)

httpResponseHead		Set status and headers for the response (instance of Http.ResponseHead)
statusCode	integer	HTTP status code of the response, defaults to 200
headers.[key]	string	Define HTTP headers to set on the response, key will be used as header name
[key]	string	Nested definitions to render (see Array)

```
page = Neos.Fusion:Http.Message {
  httpResponseHead {
    statusCode = 404
    headers.Content-Type = 'application/json'
  }
}
```

Join new

Render multiple ordered definitions and concatenate the results, returns string

[key]	string	A nested definition that evaluates to a string
[key].@position		Define the ordering of the element
@ignoreProperties	array	List of properties ignored by rendering
@glue	string	The glue used to join the items together (default '')

```
output = Neos.Fusion:Join {
  world = 'world!'
  hello = Neos.Fusion:Value {
    value = 'Hello'
    @position = 'start'
  }
  @glue = '|'
}
# Rendered output: Hello|world!
```

Loop new

Render each item in items **using** itemRenderer**, returns** string

items	array/Iterable	The array or iterable to iterate over (required)
itemName	string	Context variable name for each item value (defaults to item)
itemKey	string	Context variable name for each item key (defaults to itemKey)
iterationName	string	Context variable name for iteration information with properties index, cycle, isFirst, isLast (defaults to iterator)
itemRenderer | content	mixed	The item rendering definition: simple value, expression or object (required)
@glue	string	The glue used to join the items together (defaults to '')

```
output = Neos.Fusion:Loop {
  items = ${[1, 2, 3]}
  itemName = 'element'
  itemRenderer = ${element * 2}
  @glue = '|'
}
# Rendered output: 2|4|6
```

Map new

Render each item in items **using** itemRenderer**, returns** array **(opposed to string for Neos.Fusion:Loop)**

items	array/Iterable	The array or iterable to iterate over (required)
itemName	string	Context variable name for each item value (defaults to item)
itemKey	string	Context variable name for each item key (defaults to itemKey)
iterationName	string	Context variable name for iteration information with properties index, cycle, isFirst, isLast (defaults to iterator)
itemRenderer | content	mixed	The item rendering definition: simple value, expression or object (required)

```
output = Neos.Fusion:Map {
  items = ${[1, 2, 3]}
  itemName = 'element'
  itemRenderer = Neos.Fusion:Template {
    templatePath = 'resource://...'
    element = ${element}
  }
}
```

RawArray deprecated

Evaluate nested definitions, returns array

[key]	mixed	A nested definition
[key].@position		Define the ordering of the element
@ignoreProperties	array	List of properties ignored for evaluation

```
output = Neos.Fusion:Attributes {
  class = Neos.Fusion:RawArray {
    alert-error = 'alert-error'
    alert = 'alert'
    alert.@position = 'start'
  }
}
```

Neos.Fusion:RawArray has been renamed to Neos.Fusion:DataStructure. The old name is deprecated and will be removed in a future major version.

RawCollection deprecated

Evaluate each item in a collection, returns array

collection	array	The collection to evaluate (Iterable, required)
itemName	string	Context variable name for each item value (required)
itemKey	string	Context variable name for each item key (optional)
iterationName	string	Context variable name for iteration information with properties index, cycle, isFirst, isLast
itemRenderer | content	mixed	The item rendering definition: simple value, expression or object (required)

```
output = Neos.Fusion:RawCollection {
  collection = ${q(node).children()}
  itemName = 'childNode'
  itemRenderer = Neos.Fusion:RawArray {
    title = ${q(childNode).property('title')}
  }
  @process.toJson = ${Json.stringify(value)}
}
```

ⓘ Neos.Fusion:RawCollection has been renamed to Neos.Fusion:Map, which uses property items instead of collection. The old name is deprecated and will be removed in a future major version.

Reduce new

Reduce the given items to a single value by using itemRenderer

items	array/Iterable	The array or iterable to iterate over (required)
itemName	string	Context variable name for each item value (defaults to item)
carryName	string	Context variable that contains the result of the last iteration (defaults to carry)
itemKey	string	Context variable name for each item key (defaults to itemKey)
iterationName	string	Context variable name for iteration information with properties index, cycle, isFirst, isLast (defaults to iterator)
itemReducer	mixed	The reducer definition (simple value, expression or object) that will be applied for every item. (required)
initialValue	mixed	The value that is passed to the first iteration or returned if the items are empty (defaults to null)

```
output = Neos.Fusion:Reduce {
  @context {
    initialValue = 'foo'
    other = '-'
  }
  items = ${['bar', 'batz']}
  itemName = 'element'
  carryName = 'carry'
  initialValue = ${initialValue}
  iterationName = 'iteration'
  itemReducer = ${carry + '::' + element + other + iteration.index}
}
# Rendered output: foo::bar-0::batz-1
```

Renderer

Dynamic evaluation by renderer **definition,** renderPath **or** type

type	string	Object type to render (as string)
element	object	Properties for the object instance (when using type)
renderPath	string	Path to render (overrules type)
renderer	mixed	Simple value, expression or object (overrules renderPath and type)

```
output = Neos.Fusion:Renderer {
  type = 'Neos.Fusion:Value'
  element.value = 42
}
```

ResourceUri

Build a URI to a package resource (e.g. JS or CSS)

path	string	Path to resource, either a path relative to Public and package or a resource:// URI
package	string	The package key (e.g. 'My.Package')
resource	Resource	Resource object instead of path and package
localize	boolean	Whether resource localization should be used, defaults to true

```
scriptInclude = Neos.Fusion:Tag {
  tagName = 'script'
  attributes.src = Neos.Fusion:ResourceUri {
    path = 'resource://Neos.Twitter.Bootstrap/Public/3.3/js/bootstrap.min.js'
  }
}
```

Tag

Render an HTML tag with attributes and body (if given)

tagName	string	Tag name, defaults to `div`
content	string	The inner content of the element
attributes.[key]	string	Tag attributes, the `Tag` prototype defines an `Attributes` object by default
omitClosingTag	boolean	Whether to leave out the closing tag, defaults to `false`
selfClosingTag	boolean	Whether the tag is self closing, will be guessed from the tag name by default

```
output = Neos.Fusion:Tag {
  tagName = 'meta'
  attributes.charset = 'UTF-8'
}
```

Template

Render a Fluid template

templatePath	string	The template path and filename (required)
partialRootPath	string	Path to partials
layoutRootPath	string	Path to layout
sectionName	string	The Fluid `<f:section>` to render (if given)
[key]	mixed	Define a variable with name `key` in Fluid

```
prototype(My.Site:SomePrototype) < prototype(Neos.Fusion:Template) {
  templatePath = 'resource://My.Site/Private/Templates/Prototypes/SomePrototype.html'
  someVariable = ${q(node).property('someProperty')}
}
```

UriBuilder

Build a URI to a controller action

package	string	The package key (e.g. 'My.Package')
subpackage	string	The subpackage, defaults to empty
controller	string	The controller name (e.g. 'Registration')
action	string	The action name (e.g. 'new')
arguments	array	Arguments to the action by named key
format	string	An optional request format (e.g. 'html')
section	string	An optional fragment (hash) for the URI
additionalParams	array	Additional URI query parameters
addQueryString	boolean	Whether to keep current query parameters
argumentsToBeExcludedFrom-QueryString	array	Query parameters to exclude for addQueryString
absolute	boolean	Whether to create an absolute URI

```
uri = Neos.Fusion:UriBuilder {
  package = 'My.Package'
  controller = 'Registration'
  action = 'new'
  absolute = true
}
```

Value

Evaluate any value as a Fusion object

value	mixed	A simple value, expression or object (required)

```
output = Neos.Fusion:Value {
  value = 'A static text'
  @process.h1Tag = ${'<h1>' + value + '</h1>'}
}
```

BreadcrumbMenu

Render a breadcrumb menu (all Document ancestors up to the current node), **extends** Menu

> See Menu for supported properties

```
page.body {
  parts.breadcrumb = Neos.Neos:BreadcrumbMenu
}
```

Content

Base type to render content nodes, extends Template

templatePath	string	The template path and filename, defaults to 'resource://[packageKey]/Private/Templates/NodeTypes/[nodeType].html'
[key]	mixed	Variables for the node template, all node type properties are set by default
attributes	Attributes	Extensible attributes for the template (have to be used as <div{attributes -> f:format.raw()}> in template)

```
prototype(My.Site:MyContent) < prototype(Neos.Neos:Content) {
  templatePath = 'resource://My.Site/Private/Template/NodeTypes/Other.html'
  # Auto-generated for all node type properties
  # title = ${q(node).property('title')}
}
```

ContentCase

Render a content node from context variable node, **extends** Case

default	Matcher	Default matcher that renders a node with a prototype [My.Site:NodeTypeName]
[key]	object	Additional matchers for extending the default rendering, see Case

```
output = Neos.Neos:ContentCase {
  @context.node = ${q(node).property('myReference')}
}
```

ContentCollection

Render content in a ContentCollection **node, extends** Tag

nodePath	string	The relative node path of the ContentCollection (e.g. 'main') (required)
tagName	string	Tag name for the collection, defaults to 'div'
attributes	Attributes	Attributes for the collection tag
content	object	Instance of ContentCollectionRenderer
@cache	array	Content cache properties, mode defaults to 'cached'
@context.node	NodeInterface	ContentCollection Node to render, defaults to nodePath of current node

A ContentCollection is cached by default. A nested ContentCollection uses cache mode embed.

A Neos.Neos:ContentElementWrapping processor is added by default for editing metadata.

```
prototype(My.Site:Section) {
  content = Neos.Neos:ContentCollection {
    nodePath = 'content'
  }
}
```

```
page.body {
  # Render a shared, editable footer
  parts.footer = Neos.Neos:ContentCollection {
    nodePath = 'footer'
    @context.node = ${Neos.Node.nearestContentCollection(site, this.nodePath)}
  }
}
```

ContentCollectionRenderer

Helper to render a ContentCollection **node, extends** Collection

collection	array	Array or iterable of NodeInterface, defaults to children of node context variable
itemRenderer	object	Renderer for content nodes, defaults to ContentCase
itemName	string	Item name, defaults to 'node'
iterationName	string	Iteration name, defaults to 'iterator'

See Collection for all supported properties

ContentComponent

Atomic Fusion mapping component: evaluates properties as `props` **and evaluates to** `renderer`**, extends** `Component`

renderer	mixed	Rendering definition for evaluation (required)
[key]	mixed	Properties to evaluate as `props` context variable

```
prototype(My.Site:Content.Teaser) < prototype(Neos.Neos:ContentComponent) {
  renderer = My.Site:Component.Teaser {
    title = Neos.Neos:Editable {
      property = 'title'
    }
    subtitle = ${q(node).property('subtitle')}
    content = Neos.Neos:ContentCollection {
      nodePath = 'content'
    }
  }
}
```

ContentElementEditable

Processor for inline editing metadata of a single node property

property	string	Node property that should be editable (required)
node	NodeInterface	The node, defaults to `node` context variable

```
output = Neos.Fusion:Tag {
  tagName = 'h2'
  content = ${q(node).property('title')}
  @process.editable = Neos.Neos:ContentElementEditable {
    property = 'title'
  }
}
```

ContentElementWrapping

Processor for editing metadata of a node

node	NodeInterface	The node, defaults to `node` context variable

This processor is added by default when using `Content`, `ContentComponent` or `ContentCollection`.

```
prototype(My.Site:MyContent) < prototype(Neos.Fusion:Tag) {
  tagName = 'img'
  attributes.src = Neos.Neos:ImageUri {
    asset = ${q(node).property('image')}
    @if.hasAsset = ${q(node).property('image') ? true : false}
  }
  @process.wrapping = Neos.Neos:ContentElementWrapping
}
```

ConvertUris

Processor to convert internal node and asset URIs to public URIs

externalLinkTarget	string	Override the target attribute for external links, defaults to '_blank'. Disable with empty value.
resourceLinkTarget	string	Override the target attribute for resource links, defaults to '_blank'. Disable with empty value.
forceConversion	boolean	Whether to convert URIs of non-public workspaces, defaults to false
absolute	boolean	Generate absolute node URIs, defaults to false
value	string	The value to convert, defaults to value context variable
node	NodeInterface	The current node as reference, defaults to node context variable

This processor should be added to all inline editable properties and when using the LinkEditor.

```
prototype(My.Site:MyContent) {
  title.@process.convertUris = Neos.Neos:ConvertUris {
    absolute = true
  }
}
```

DimensionsMenu

Render a menu to other node variants (with different dimension values)

dimension	string	Show only presets of a single dimension (optional)
presets	array	Show only a subset or specific order of presets when using dimension (optional)
includeAllPresets	boolean	Include all presets – not only allowed combinations, defaults to false

templatePath	string	Override the template path
renderHiddenInIndex	boolean	Whether nodes with hiddenInIndex should be rendered, defaults to false
attributes	Attributes	Extensible attributes for the whole menu
normal.attributes	Attributes	Attributes for *normal* state
active.attributes	Attributes	Attributes for *active* state
current.attributes	Attributes	Attributes for *current* state
absent.attributes	Attributes	Attributes for *absent* state

Menu item properties

Each item represents a combination of allowed dimension presets (e.g. *language* and *country*).

node	NodeInterface	A node that should be used to link to the item (with resolved shortcuts)
state	string	'normal', 'active' (ancestor of current node), 'current' (the current document node) or 'absent' (no variant found)
label	string	Dimension preset label
dimensions	array	Dimension values of the preset combination indexed by dimension name
targetDimensions	array	Target dimensions, indexed by dimension name with array of value, label and isPinnedDimension

```
variantMenu = Neos.Neos:DimensionsMenu
```

```
languageMenu = Neos.Neos:DimensionsMenu {
  dimension = 'language'
  presets = ${['en_US', 'de_DE']}
}
```

Editable

Atomic fusion component for an editable property

node	NodeInterface	The node, defaults to node context variable
property	string	Node property that should be editable (required)
block	boolean	Render a block ( ) or inline element (), defaults to true

```
prototype(My.Site:Content.MyContent) < prototype(Neos.Neos:ContentComponent) {
  renderer = My.Site:Component.MyContent {
    title = Neos.Neos:Editable {
      property = 'title'
    }
  }
}
```

ImageTag

Render an image tag for an asset (thumbnail)

asset	AssetInterface	A Image, ImageVariant or other AssetInterface
*		All other ImageUri properties
attributes	Attributes	Image tag attributes

```
logoUri = Neos.Neos:ImageUri {
  asset = ${q(node).property('image')}
  width = 100
  height = 100
  allowCropping = true
  allowUpScaling = true
}
```

ImageUri

Build a URI to a (thumbnail) image for an asset

asset	AssetInterface	A Image, ImageVariant or other AssetInterface
width	integer	Desired width of the image (optional)
maximumWidth	integer	Restrict image to a maximum width, defaults to 2560
height	integer	Desired height of the image (optional)
maximumHeight	integer	Restrict image to a maximum height, defaults to 2560
allowCropping	boolean	Crop the image if the target aspect ratio does not match, defaults to false
allowUpScaling	boolean	Allows to scale the resulting image larger than the original, defaults to false
quality	integer	Desired quality of the image (for JPEG images) between 1 and 100, defaults to null

preset	string	Image configuration preset (optional – if set, all other resize attributes are ignored)
async	boolean	Returns an asynchronous image URI if the requested images was not generated yet, defaults to false

```
logoUri = Neos.Neos:ImageUri {
  asset = ${q(node).property('logo')}
  width = 200
  height = 200
  allowCropping = true
}
```

Menu

Render a menu with items for nodes, extends Template

templatePath	string	Override the template path
entryLevel	integer	Start the menu at the given depth
maximumLevels	integer	Restrict the maximum depth of items in the menu (relative to entryLevel)
startingPoint	NodeInterface	The parent node of the first menu level (defaults to node context variable)
lastLevel	integer	Restrict the menu depth by node depth (relative to site node)
filter	string	Filter items by node type, defaults to 'Neos.Neos:Document'
renderHiddenInIndex	boolean	Whether nodes with hiddenInIndex should be rendered, defaults to false
itemCollection	array	Explicitly set the node items for the menu (alternative to startingPoint and levels)
attributes	Attributes	Extensible attributes for the whole menu
normal.attributes	Attributes	Attributes for *normal* state
active.attributes	Attributes	Attributes for *active* state
current.attributes	Attributes	Attributes for *current* state

Menu item properties

node	NodeInterface	A node that should be used to link to the item (with resolved shortcuts)

originalNode	NodeInterface	The unresolved node for the menu item
state	string	'normal', 'active' (ancestor of current node) or 'current' (the current document node)
label	string	Full label of the node
menuLevel	integer	Menu level of the item
subItems	array	Array of sub items in the menu

```
body.parts.mainMenu = Neos.Neos:Menu {
  entryLevel = 1
  maximumLevels = 3
  filter = 'Neos.Neos:Document,!My.Site:Document.Post'
}
```

NodeUri

Build a URI to a node

node	NodeInterface | string	A node instance or node path (absolute or relative), empty for current document node
format	string	An optional request format (e.g. 'html')
section	string	An optional fragment (hash) for the URI
additionalParams	array	Additional URI query parameters
addQueryString	boolean	Whether to keep current query parameters
argumentsToBeExcludedFromQueryString	array	Query parameters to exclude for addQueryString
absolute	boolean	Whether to create an absolute URI
baseNodeName	string	Base node context variable for relative paths, defaults to 'documentNode'

```
siteUri = Neos.Neos:NodeUri {
  node = ${site}
}
```

Page

Render an HTML document with metadata for Neos

doctype	string	HTML doctype, defaults to <!DOCTYPE html>

htmlTag	Tag	The opening <html> tag
attributes.[key]	Attributes	Attributes for the <html> tag
headTag	Tag	The opening <head> tag
head	Array	Content inside <head>
titleTag	Tag	The <title> tag
javascripts	Array	Head JS includes (script tags)
stylesheets	Array	Head CSS includes (link tags)
bodyTag	Tag	The opening <body> tag
attributes.[key]	Attributes	Attributes for <body> tag
body	Template	The main body content
templatePath	string	Path to the body Fluid template (required)
[key]	mixed	Variables for the body template
javascripts	Array	Body footer JS includes (script tags)

```
# Render content from main ContentCollection
# Output in Fluid template with {content.main -> f:format.raw()}
page.body {
  content.main = PrimaryContent {
    nodePath = 'main'
  }
}
```

```
# Include a stylesheet from a <f:section> in the template
page.head.stylesheets.mySite = Neos.Fusion:Template {
  templatePath = 'resource://My.Site/Private/Templates/Page/Default.html'
  sectionName = 'stylesheets'
  node = ${node}
}
```

Plugin

Base type to render plugin nodes or static plugins

package	string	The package key (e.g. 'My.Package') (required)
subpackage	string	The subpackage, defaults to empty
controller	array	The controller name (e.g. 'Registration')
action	string	The action name (e.g. 'new')

argumentNamespace	string	Namespace for action arguments, will be resolved from node type by default
[key]	mixed	Pass an internal argument to the action (access with _key)

Plugins are based on Flow controllers that can implement arbitrary logic.

```
prototype(My.Package:FlickrPlugin) < prototype(Neos.Neos:Plugin) {
  package = 'My.Package'
  controller = 'Flickr'
  action = 'tagStream'
}
```

PrimaryContent

Render the primary ContentCollection **of a document, extends** Case

nodePath	string	The relative node path of the ContentCollection (e.g. 'main') (required)
default	array	Default case that renders a ContentCollection
[key]	array	Additional conditions (e.g. to handle specific document types), see Case

```
page.body {
  content {
    main = PrimaryContent {
      nodePath = 'main'
    }
  }
}
```

> AFX is a domain specific language for Fusion with a compact tag based syntax

Eel Expression

```
renderer = afx`{String.toUpperCase(props.headline)}`
```

Tags with Attributes

```
renderer = afx`
  <Vendor.Site:Prototype type="headline">
    <h1 class="headline">{props.headline}</h1>
    <div @if.hasDetails={props.details}>{props.details}</div>
  </Vendor.Site:Prototype>
`
```

Properties Spread

Pass multiple properties to a tag using Fusion @apply

```
renderer = afx`
  <Vendor.Site:Component {...expression} title="Foo" />
`
```

@path

Define the Fusion property path of a child inside the parent object (should be used in favor of @children and @key)

```
renderer = afx`
  <Vendor.Site:ExampleContainer>
    <h2 @path="header">{props.title}</h1>
    <p @path="body">{props.description}</p>
  </Vendor.Site:Prototype>
`
```

@children

Sets a Fusion property that will receive children of the tag (defaults to content)

```
renderer = afx`
  <Neos.Fusion:Loop collection={['one','two','three']} @children="itemRenderer">
    <p>{item}</p>
  </Neos.Fusion:Loop>
`
```

new @children="itemRenderer" can be omitted for Neos.Fusion:Loop or Neos.Fusion:Map.

@key

Define an explicit property name when having *multiple* children

```
renderer = afx`
  <div class="teaser">
    <h2 @key="title">{props.title}</h2>
    <p @key="content">{props.content}</p>
  </div>
`
```

> Eel expressions are valid JavaScript syntax
> JavaScript expression syntax can be used for Eel

Literals

```
42 | 1.5 | false | true | null |
'A string' | "Another \"quoted\" string"
```

Variables

```
${myVariable}
```

Traversal

```
${node.contextPath}
```

Method Calls

```
${String.substr("Hello world!", 6, 5)}
${q(node).property('title')}
```

Offset Access

```
${String.split("Hello world!", ' ')[1]}
${items[index + 1]}
```

Property Access (this)

```
prototype(My.Site:Prototype) {
  _content = ${q(node).property('content')}
  croppedContent = ${String.crop(this._content, 100)}
  @if.hasContent = ${this._content ? true : false}
}
```

Mathematical operators

```
+ | - | * | / | %
```

Comparisons

```
== | != | < | <= | > | >=
```

Boolean operators

```
! | && | ||
```

Conditional operator

```
${conditionExpression ? thenExpression : elseExpression}
```

Array and Object Literals

```
${Array.join(['first', 'second', 'last'], ', ')}
${q(node).context({'invisibleContentShown': true})}
```

Function Literals new

```
${Array.map([1, 2, 3, 4], x => x * x)}
${Array.reduce(
  [1, 2, 3, 4],
  (accumulator, currentValue) => accumulator + currentValue,
  1
)}
```

Array.[function]

Array helpers based on JavaScript specification (where applicable), including EcmaScript 6 proposals.

concat (array1, array2, *array_*)	Concatenate arrays or values to a new array
every (array, callback)	Check if all elements in an array pass a test given by the callback
filter (array, callback)	Filter an array by a test given as the callback, passing each element and key as arguments
first (array)	Get the first element of an array
flip (array)	Exchanges keys with associated values
indexOf (array, search, *fromIndex*)	Get first index of search, -1 if not present
isEmpty (array)	Check if an array is empty
join (array, *separator*)	Join values of an array with a separator
keys (array)	Get the array keys
last (array)	Get the last element of an array
length (array)	Get the length of an array
map (array, callback)	Apply the callback to each element of the array
pop (array)	Removes the last element from an array
push (array, element)	Insert one or more elements at the end
random (array)	Picks a random element from the array
range (start, end, *step*)	Create an array containing a range of elements
reduce (array, callback, initialValue)	Apply the callback to each element of the array and accumulate a single value
reverse (array)	Returns an array in reverse order
set (array, key, value)	Set the specified key in the the array
shift (array)	Remove the first element of an array
shuffle (array, *preserveKeys*)	Shuffle an array
slice (array, begin, *end*)	Extract a portion of an indexed array
some (array, callback)	Check if at least one element in an array passes a test given by the callback
sort (array)	Sorts an array
splice (array, offset, *length*, *replacements*)	Replaces a range by the given replacements
unshift (array, element)	Insert one or more elements at the beginning

```
@if.notEmpty = ${Array.length(myArray) > 0}
```

```
# Check if some item matches a condition
@if.hasItems = ${Array.some(myArray, x => x > 4)}
```

```
# Transform node properties with map
value = ${
  Array.join(
    Array.map(q(site).children('[instanceof Neos.Neos:Document]').get(), n => q(n).property('title'))
  )
}
```

```
@process.filterNotBlank = ${Array.filter(value, x => !String.isBlank(x))}
```

Configuration.[function]

setting (settingPath)	Return Flow settings for the given path

```
${Configuration.setting('My.Site.tracking.enabled')}
```

Date.[function]

add (date, interval)	Add interval, returns new DateTime
create (time)	Get a date object by given date or time format
dayOfMonth (dateTime)	Get the day of month of a date
diff (dateA, dateB)	Difference between dates as DateInterval
format (date, format)	Format date (or interval) to string with format
formatCldr (date, cldrFormat, *locale*)	Format a date to string with *CLDR* format
hour (dateTime)	Get the hour of a date (24 hour format)
minute (dateTime)	Get the minute of a date
month (dateTime)	Get the month of a date
now ()	Get the current date and time
parse (string, format)	Parse date from string with format to DateTime
second (dateTime)	Get the second of a date
subtract (date, interval)	Subtract interval, returns new DateTime
today ()	Get the current date
year (dateTime)	Get the year of a date

```
${Date.format(q(node).property('publishedAt'), 'd.m.Y')}
```

Json.[function]

parse (json, *associativeArrays*)	JSON decode the given string
stringify (value)	JSON encode the given value

```
# Convert RawArray or RawCollection to JSON string
@process.toJson = ${Json.stringify(value)}
```

Math.[function]

Mathematical helpers based on JavaScript specification (excerpt)

abs (x)	Absolute value of x (non-negative)
ceil (x)	Smallest integer greater than or equal to x
floor (x)	Largest integer less than or equal to x
isNaN (x)	Test if the given value is not a number
max (x, y_)	Returns the largest argument
min (x, y_)	Returns the smallest argument
pow (x, y)	Power of x by y
random ()	Random number between 0 (incl.) and 1 (excl.)
randomInt (min, max)	Random int between min and max (incl.)
round (subject, *precision*)	Rounds the subject to the given precision
sign (x)	Sign of x (–1, 1 or 0)
sqrt (x)	Square root of x
trunc (x)	Remove any fractional digits (ceil or floor)

See neos.readthedocs.io for a complete reference of Math helpers.

Neos.Array.[function]

Functional programming array helpers

filter (set, filterProperty)	Filter an array of objects, by only keeping the elements where each object's $filterProperty evaluates to true
filterNegated (set, filterProperty)	Filter an array of objects, by only keeping the elements where each object's $filterProperty evaluates to false
groupBy (set, groupingKey)	The input is assumed to be an array or Collection of objects. Groups this input by the $groupingKey property of each element

Neos.Caching.[function]

Caching helpers to make cache tag generation easier

descendantOfTag (nodes)	Generate a @cache entry tag for descendants of a node, an array of nodes or a FlowQuery result
nodeTag (nodes)	Generate a @cache entry tag for a single node, array of nodes or a FlowQuery result
nodeTypeTag (nodeType)	Generate an @cache entry tag for a node type

```
@cache.entryTags.refs = ${Neos.Caching.descendantOfTag(q(node).property('references'))}
```

Neos.Link.[function]

Linking and routing helpers

convertUriToObject (uri, *contextNode*)	Get node or asset object by internal URI
getScheme (uri)	Get scheme of internal URI (i.e. 'node' or 'asset')
hasSupportedScheme (uri)	Test if scheme is supported for linking
resolveAssetUri (uri)	Resolve public URI for asset:// URI
resolveNodeUri (uri, contextNode, controllerContext)	Resolve public URI for node:// URI

```
linkedNode = ${Neos.Link.convertUriToObject(nodeUri, node)}
```

Neos.Node.[function]

Node helpers

nearestContentCollection (node, nodePath)	Check if node is a collection, find collection by nodePath otherwise, else throw exception
isOfType (node, nodeType)	If this node type or any of the direct or indirect super types has the given name

Security.[function]

Helper for security related information

getAccount ()	Get the Account of the first authenticated token
hasRole (roleIdentifier)	Test if at least one authenticated account has the given role

String.[function]

String helpers based on JavaScript specification (where applicable)

base64decode (string, *strict*)	Implementation of PHP base64_decode
base64encode (string, *strict*)	Implementation of PHP base64_encode
charAt (string, index)	Get the character at a specific position
chr (value)	Generate a single-byte string from a number
crop (string, maxChars, *suffix*)	Crop a string to maxChars, add suffix if cropped
cropAtSentence (string, maxChars, *suffix*)	Crop a string to maxChars at sentence
cropAtWord (string, maxChars, *suffix*)	Crop a string to maxChars at word boundary
endsWith (string, search, *position*)	Test if a string ends with the given search string
firstLetterToLowerCase (string)	Lowercase the first letter of a string
firstLetterToUpperCase (string)	Uppercase the first letter of a string
htmlSpecialChars (string, *preserveEntities*)	Convert special characters to HTML entities
indexOf (string, search, *fromIndex*)	Find the first position of a substring

isBlank (string)	Test if a string is blank or whitespace only
lastIndexOf (string, search, *toIndex*)	Find the last position of a substring
length (string)	Get the length of a string
md5 (string)	Calculate the MD5 checksum of the given string
nl2br (string)	Insert HTML line breaks before all newlines in a string
ord (string)	Convert the first byte of a string to a value between 0 and 255
pregMatch (string, pattern)	Match a regular expression (PREG)
pregMatchAll (string, pattern)	Match a global regular expression (PREG)
pregReplace (string, pattern, replace)	Replace by regular expression (PREG)
pregSplit (string, pattern, *limit*)	Split string by regular expression (PREG)
rawUrlDecode (string)	Decode string from URLs (RFC 3986)
rawUrlEncode (string)	Encode string for URLs (RFC 3986)
replace (string, search, replace)	Replace occurrences of search by replace
split (string, *separator*, *limit*)	Split a string by a separator
startsWith (string, search, *position*)	Test if a string starts with the given search string
stripTags (string, *allowableTags*)	Strip all HTML tags from the given string
substr (string, start, *length*)	Return characters from start up to given length
substring (string, start, *end*)	Return characters from start to end index
toBoolean (string)	Convert a string to boolean
toFloat (string)	Convert a string to float
toInteger (string)	Convert a string to integer
toLowerCase (string)	Lowercase a string
toString (value)	Convert the given value to a string
toUpperCase (string)	Uppercase a string
trim (string, *charlist*)	Trim whitespace at the beginning and end
wordCount (string)	Get count of words in string

```
# E.g. for '75-25'
colWidth = ${String.split(q(node).parent().property('layout'), '-')[index]}
```

```
page = ${String.toInteger(request.arguments['page'])}
```

Translation.[function]

Translation helpers

id (id)	Build a translation by id
value (value)	Build a translation by original label
translate (id, *originalLabel*, *arguments*, *source*, *package*, *quantity*, *locale*)	Get the translated value for an id or original label

Translation builder helpers using id(...) or value(...):

value (value)	Set the original translation value
arguments (arguments)	Set the arguments
source (source)	Set the source (name of file with translations)
package (package)	Set target package key (defaults to current)
quantity (quantity)	Set the quantity (number to find plural form)
locale (locale)	Set the locale by identifier
translate (overrides)	Get the translation, parameters can be overridden

```
# Get a translation by id using the builder API
${Translation.id('myLabel').source('Main').translate()}
```

Type.[function]

Type helpers

className (variable)	Get class name of variable (null if not an object)
getType (variable)	Get the variable type
instance (variable, expectedObjectType)	Is the given variable of the provided object type
isArray (variable)	Is the given variable an array
isBoolean (variable)	Is the given variable boolean
isFloat (variable)	Is the given variable a float
isInteger (variable)	Is the given variable an integer
isNumeric (variable)	Is the given variable numeric
isObject (variable)	Is the given variable an object
isScalar (variable)	Is the given variable a scalar
isString (variable)	Is the given variable a string
typeof (variable)	Get the variable type

```
@if.assetNotEmpty = ${Type.isObject(asset)}
```

- The Eel helper q(...) is used to wrap any node, object, array or Iterable in FlowQuery
- Most operations traverse node or object graphs and return a new FlowQuery result
- final operations return single or array values not wrapped in a FlowQuery result
- FlowQuery always works on a set of items (like jQuery does), .get() unwraps a result

A special filter syntax can be used for Filter arguments of FlowQuery operations:

Attribute filters

instanceof | !instanceof — Checks if an item is (not) an instance of the given node type

```
childDocs = ${q(node).children('[instanceof Neos.Neos:Document]')}
```

= | != — Strict (in-) equality of property and value

```
menuItems = ${q(node).children('[showInMenu=true]')}
```

< | <= | > | >= — Comparison of value and operand with less than, less than or equal, greater than, greater or equal

```
largeItems = ${q(node).find('[size>=4]')}
```

^= — Value starts with operand (string-based)

```
sizeItems = ${q(node).find('[style^="size-"]')}
```

$= — Value ends with operand (string-based)

```
xsItems = ${q(node).find('[style$="-xs"]')}
```

*= — Value contains operand (string-based)

```
jsItems = ${q(node).find('[language*="js"]')}
```

Node name filter

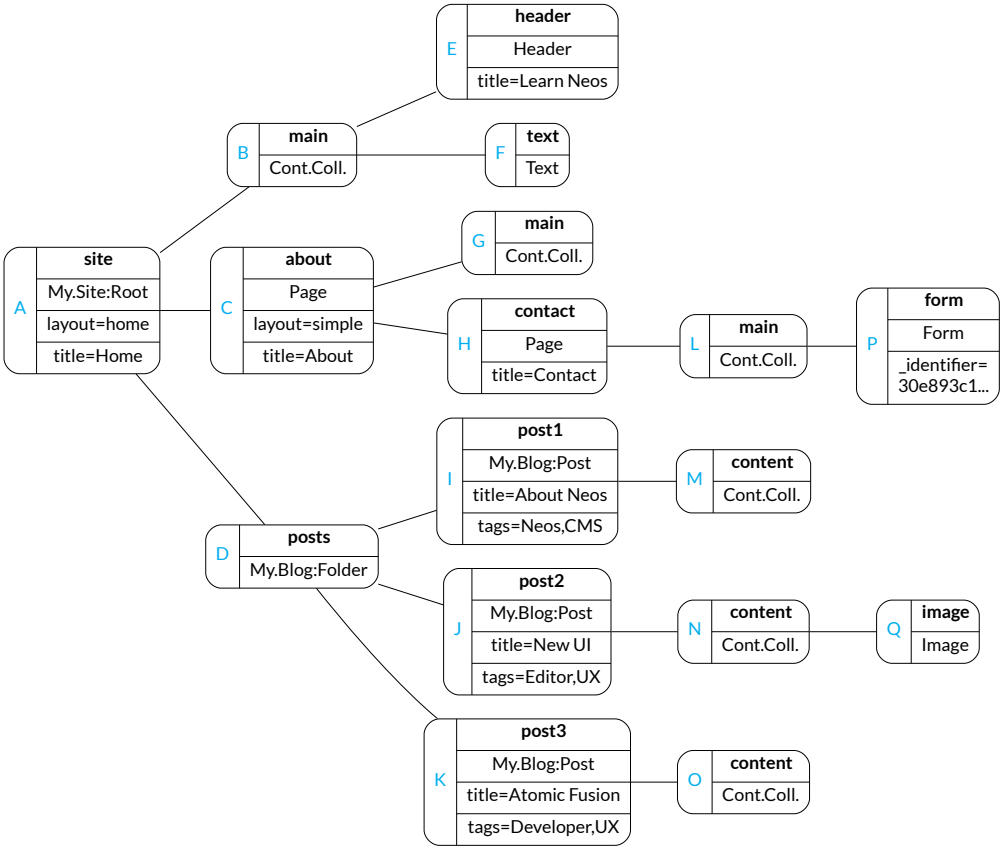
name — Filter nodes by node name

```
mainContentCollection = ${q(node).children('main').get(0)}
```

Combining filters

name[filter] — Node name and attribute filter
[filterA][filterB] — All filters have to match (AND)
[filterA],[filterB] — At least one filter has to match (OR)

```
shouldLink = ${q(node).is('[instanceof My.Blog:Post],[instanceof My.Blog:Author]')}
```



Node structure used in the following code examples

.add(other)

Adds all items of another FlowQuery to the current result (appended to the end)

other	FlowQuery	FlowQuery results to add

```
# node: H
nodes = ${q(node).add(q(node).parents())}
# nodes: H, C, A
```

.cacheLifetime()

final **Returns the minimum cache lifetime for all nodes in the result as seconds or** null **if no** hiddenBeforeDateTime **or** hiddenAfterDateTime **properties are set, returns integer**

```
@cache.maximumLifetime = ${q(node).children().cacheLifetime()}
```

.children(*filter*)

Get child nodes (for each node in the result)

filter	Filter	Optional filter to select child nodes

```
nodes = ${q(site).children('[instanceof Neos.Neos:Document][!instanceof My.Blog:Folder]')}
# nodes: C
```

.closest(filter)

Get the first node that matches `filter` **starting from node and traversing all parents** (for each node in the result)

filter	Filter	Filter to find closest nodes

```
# node: H
baseLayout = ${q(node).closest('[layout]').property('layout')}
# baseLayout: simple
```

.context(properties)

Get nodes in result from another CR context (e.g. workspace, dimension values)**, given by** `properties` (if a variant exists in the context)

properties	array	CR context properties (see `ContentContextFactory`)

```
removedNodes = ${
  q(node).context({'removedContentShown': true}).children('[_removed = true]')
}
```

.count(*filter*)

final **Count items in result with optional filter, returns integer**

filter	Filter	Optional filter for items to count

```
# node: D
numberOfArticles = ${q(node).children().count()}
# numberOfArticles: 2
```

.filter(filter)

Keep items in the result matching filter

filter	Filter	Filter for items

```
# node: C
nodes = ${q(node).children().filter('[instanceof Neos.Neos:ContentCollection]')}
# nodes: G
```

.find(filter)

Find nodes by name, path, identifier or node type filter (recursively)

filter	Filter	Filter to select nodes (see examples)

```
byNodeName = ${q(site).find('main')}
# byNodeName: B
byRelativePath = ${q(site).find('main/text')}
# byRelativePath: F
byAbsolutePath = ${q(site).find('/sites/site/about')}
# byAbsolutePath: C
byIdentifier = ${q(site).find('#30e893c1...')}
# byIdentifier: P
byNodeType = ${q(site).find('[instanceof My.Blog:Post]')}
# byNodeType: I, J, K
byMultipleNodeTypes = ${q(site).find('[instanceof My.Blog:Folder],[instanceof My.Blog:Post]')}
# byMultipleNodeTypes: D, I, J, K
byNodeTypeWithFilter = ${q(site).find('[instanceof My.Blog:Post][tags*="UX"]')}
# byNodeTypeWithFilter: J, K
```

.first()

Reduce result to first item (or leave empty)

```
# node: D
nodes = ${q(node).children().first().find('content')}
# nodes: M
```

.get(*index*)

final **Unwrap result as array or get single item, returns single value or array**

index	integer	Optional item index in the result

```
# node: D
post = ${q(node).children().get(1)}
# post: J <- NodeInterface
```

```
# node: D
posts = ${q(node).children().get()}
# posts: [I, J, K] <- array of NodeInterface
```

.has(selector)

Reduce the result to nodes having a child that matches a filter or one of the given nodes

selector		A filter, array, object, Traversable or FlowQuery

```
# node: D
nodes = ${q(node).children().find('content').has('[instanceof Neos.NodeTypes:Image]').parent()}
# nodes: J
```

.is(*filter*)

final **Test if at least one item exists in the result** (matching an optional filter)**, returns boolean**

filter	Filter	Optional filter for matching items

```
condition = ${q(site).is('[instanceof My.Site:Root]')}
# condition: true
```

.last()

Reduce the current result to the last element (or leave empty)

```
# node: D
title = ${q(node).children().last().property('title')}
# title: Atomic Fusion <- string
```

.next(*filter*)

Get the following sibling with an optional filter (for each node in the result)

filter	Filter	If given, a following sibling has to match the filter

```
# node: I
nextPost = ${q(node).next().get(0)}
# nextPost: J <- NodeInterface
```

.nextAll(*filter*)

Get all following siblings with an optional filter (for each node in the result)

filter	Filter	If given, following siblings have to match the filter

```
# node: I
posts = ${q(node).nextAll()}
# posts: J, K
```

.nextUntil(selector, *filter*)

Get following siblings until a node matches selector **with an optional filter** (for each node in the result)

selector	Filter	Following siblings are taken until the selector is matched (which is not included)
filter	Filter	If given, siblings have to match the filter

```
# node: I
posts = ${q(node).nextUntil('[tags*="Developer"]')}
# posts: J
```

.parent(*filter*)

Get the parent node with an optional filter (for each node in the result)

filter	Filter	If given, a parent has to match the filter

```
# node: H
title = ${q(node).parent().property('title')}
# title: About <- string
```

.parents(*filter*)

Get all ancestors (up to and including the site node) **with an optional filter, direct parent comes first** (for each node in the result)

filter	Filter	If given, keep only ancestors matching the filter

```
# node: P
nodes = ${q(node).parents('[instanceof Neos.Neos:Document]')}
# nodes: H, C, A
```

.parentsUntil(selector, *filter*)

Get ancestors up to a node matching selector **with an optional filter, direct parent comes first** (for each node in the result)

selector	Filter	Ancestors are taken until the selector is matched (which is not included)
filter	Filter	If given, keep only ancestors matching the filter

```
# node: P
nodes = ${q(node).parentsUntil('[instanceof My.Site:Root]')}
# nodes: H, C
```

.prev(*filter*)

Get the previous sibling with an optional filter (for each node in the result)

filter	Filter	If given, a previous sibling has to match the filter

```
# node: J
prevTitle = ${q(node).prev('[instanceof My.Blog:Post]').property('title')}
# prevTitle: About Neos <- string
```

.prevAll(*filter*)

Get all previous siblings with an optional filter, direct sibling comes first (for each node in the result)

filter	Filter	If given, previous siblings have to match the filter

```
# node: K
posts = ${q(node).prevAll()}
# posts: J, I
```

.prevUntil(selector, *filter*)

Get previous siblings until a node matches selector **with an optional filter, direct sibling comes first** (for each node in the result)

selector	Filter	Previous siblings are taken until the selector is matched (which is not included)
filter	Filter	If given, siblings have to match the filter

```
# node: K
posts = ${q(node).prevUntil('[tags*="CMS"]')}
# posts: J
```

.property(propertyName)

final **Get the value of** `propertyName` **for the first item in the result, returns mixed**

propertyName	string	Name of the property, internal properties can be accessed with an underscore prefix (e.g. `_hidden`)

```
# node: C
title = ${q(node).property('title') || 'A default title'}
# title: About <- string
```

```
# node: P
identifier = ${q(node).property('_identifier')}
# identifier: 30e893c1... <- string
```

.remove(items) new

Removes all given items from the current result

items	Array, FlowQuery or Object	Items to remove

```
# node of type post: J
 posts = ${q(site).find('[instanceof post]').remove(node)}
 # posts: I, K
```

.siblings(*filter*)

Get all nodes on the same level excluding the current node and matching an optional filter
(for each node in the result)

filter	Filter	If given, siblings have to match the filter

```
# node: J
posts = ${q(node).siblings()}
# posts: I, K
```

.slice(start, *end*)

Reduce the result to indices from `start` **to** `end`

start	integer	Start index of items (starting from `0`), a negative value indicates an offset from the end
end	integer	Optional end index of items (defaults to last item), a negative value indicates an offset from the end

```
# node: D
posts = ${q(node).children('[instanceof My.Blog:Post]').slice(0, -1)}
# posts: I, J
```

.sort(propertyName, direction)

Sort nodes in the result by propertyName **in** direction

propertyName	string	Name of the property to use for sorting
direction	string	Order items ascending 'ASC' or descending 'DESC'

```
# node: D
posts = ${q(site).find('[instanceof My.Blog:Post]').sort('title', 'ASC')}
# posts: I, K, J
```

Need more help?

> www.neos.io/docs-and-support/documentation.html
Browse the official documentation for Neos and Flow

> discuss.neos.io
Ask a question in the official forum

> slack.neos.io
Join Neos on Slack and chat with the community

> learn-neos.com
Find Fusion tutorials and exercises

> neos@networkteam.com
Or get in touch with us about your Neos project

NEED HELP?

- Development, support or trainings – we'll boost up your Neos project
- Keen to learn more? Check out **networkteam.com/needhelp**

networkteam

Don't panic

learn-neos.com

networkteam