

The XmlText FieldType

This FieldType validates and stores formatted text.

Name	Internal name	Expected input
XmlText	ezxmltext	mixed

- Input expectations
 - Example of the FieldType's internal format
 - For XHTML Input
- Input object API
- Value object API
- Validation
- Settings
 - Tag presets

Input expectations

Type	Description	Example
string	XML document in the FieldType internal format as a string.	See the example below.
eZ\Publish\Core\FieldType\XmlText\Input	An instance of the class implementing FieldType abstract <code>Input</code> class.	See the example below.
eZ\Publish\Core\FieldType\XmlText\Value	An instance of the FieldType <code>value</code> object.	See the example below.

Example of the FieldType's internal format

```
<?xml version="1.0" encoding="utf-8"?>
<section
  xmlns:custom="http://ez.no/namespaces/ezpublish3/custom/"
  xmlns:image="http://ez.no/namespaces/ezpublish3/image/"
  xmlns:xhtml="http://ez.no/namespaces/ezpublish3/xhtml/"
  <paragraph>This is a paragraph.</paragraph>
</section>
```

For XHTML Input

The eZ XML output use `` and `` by default, respecting the semantic XHTML notation.

Learn more about ``, ``, ``, `<i>`

- [Read the share.ez.no forum about our choice of semantic tags with eZ XML](#)
- [Learn more about the semantic tags vs the presentational tags.](#)

Input object API

`Input` object is intended as a vector for different input formats. It should accept input value in a foreign format and convert it to the FieldType's internal format.

It should implement abstract `eZ\Publish\Core\FieldType\XmlText\Input` class, which defines only one method:

Method	Description
getInternalRepresentation	The method should return the input value in the internal format.

At the moment there is only one implementation of the `Input` class, `eZ\Publish\Core\FieldType\XmlText\Input\EzXml`, which accepts input value in the internal format, and therefore only performs validation of the input value.

Example of using the Input object

```

...

use eZ\Publish\Core\FieldType\XmlText\Input\EzXml as EzXmlInput;

...

$contentService = $repository->getContentService();
$contentTypeService = $repository->getContentTypeService();

$contentType = $contentTypeService->loadContentTypeByIdentifier( "article" );
$contentCreateStruct = $contentService->newContentCreateStruct( $contentType, "eng-GB"
);

$inputString = <<<EZXML
<?xml version="1.0" encoding="utf-8"?>
<section
  xmlns:custom="http://ez.no/namespaces/ezpublish3/custom/"
  xmlns:image="http://ez.no/namespaces/ezpublish3/image/"
  xmlns:xhtml="http://ez.no/namespaces/ezpublish3/xhtml/">
  <paragraph>This is a paragraph.</paragraph>
</section>
EZXML;

$ezxmlInput = new EzXmlInput( $inputString );

$contentCreateStruct->setField( "description", $ezxmlInput );

...

```

Value object API

`eZ\Publish\Core\FieldType\XmlText\Value` offers following properties:

Property	Type	Description
xml	DOMDocument	Internal format value as an instance of DOMDocument.

Validation

Validation of the internal format is performed in the `eZ\Publish\Core\FieldType\XmlText\Input\EzXml` class.

Settings

Following settings are available:

Name	Type	Default value	Description
<code>numRows</code>	<code>int</code>	10	Defines the number of rows for the online editor in the administration interface.
<code>tagPreset</code>	<code>mixed</code>	<code>Type::TAG_PRESET_DEFAULT</code>	Preset of tags for the online editor in the administration interface.

Tag presets

Following tag presets are available as constants in the `eZ\Publish\Core\FieldType\XmlText` class:

Constant	Description
<code>TAG_PRESET_DEFAULT</code>	Default tag preset.
<code>TAG_PRESET_SIMPLE_FORMATTING</code>	Preset of tags for online editor intended for simple formatting options.

Example of using settings in PHP

```
...  
  
use eZ\Publish\Core\FieldType\XmlText\Type;  
  
...  
  
$contentTypeService = $repository->getContentTypeService();  
$xmltextFieldCreateStruct = $contentTypeService->newFieldDefinitionCreateStruct(  
    "description", "ezxmltext" );  
  
$xmltextFieldCreateStruct->fieldSettings = array(  
    "numRows" => 25,  
    "tagPreset" => Type::TAG_PRESET_SIMPLE_FORMATTING  
);  
  
...
```