Base Distribution - Task # 44955

Story # 44921 (Accepted): Decide TypoScript and TYPO3CR Naming

Status:	Resolved	**Priority:**	Should have
Author:	Sebastian Kurfuerst	**Category:**	
Created:	2013-01-30	**Assigned To:**	Sebastian Kurfuerst
Updated:	2013-01-30	**Due date:**	
Subject:	decide Node Type Definition naming		
Description			

Related issues:

related to Base Distribution - Story # 44982: Refactor Node Type Definition a...	**Resolved**	**2013-01-30**

History

#1 - 2013-01-30 10:19 - Sebastian Kurfuerst

- Status changed from New to Accepted

#2 - 2013-01-30 10:19 - Sebastian Kurfuerst

Node type definition NOW

The following options are allowed:

* `superTypes`: array of parent node types which are inherited from

* `childNodes`: When the given content type is created, the subnodes listed underneath
  here are automatically created. Example::

```
childNodes:
  column0:
    type: 'TYPO3.Neos.ContentTypes:ContentCollection'
    autoCreated: true (only thing supported in Neos currently)
    mandatory: true/false # TODO for validation lateron
```

* `isAbstract`: true/false cannot be instanciated directly

* `isMixin`: true/false is a mixin

* `properties`: list of properties for this content type. For each property,
  the following settings can be done:

  * `type`: type of the property. One of:
    * `boolean`
    * `string`
    * `date` -> TODO: DateTime
    * `enum` -> TODO: Get rid of that. it's a string with a special editor.
    * `TYPO3\Media\Domain\Model\ImageVariant`

    * additional types can be used, as long as they are defined inside
      `TYPO3.Neos.userInterface` (TODO: bad name!) or specify a JavaScript
      class inside `userInterface.class`

  * `defaultValue`: (TODO: renamed from "default") (optional) Default value. If a new node is created of this type, then it
    is initialized with this value.

  * `ui`:
    * `label`: (OBSOLETE for 1.1)
    * `inspector`:
      * `group`:
      * `position`: (TODO: position should work as in TypoScript)
      * `editor`:
      * `editorOptions`:
    * `reloadIfChanged`:
    * `inlineEditable` (true/false)

* `ui`:

  * `label`: (OBSOLETE for 1.1)
  * `icon`
  * `darkIcon`
  * `group`
  * `inlineEditable` (if FALSE, the non-editable-overlay is shown)
  * `inspector`
    * `groups`

Node Type Definition beforehand

The following options are allowed:

* `superTypes`: array of parent content types which are inherited from

* `label`: The human-readable label of the content type

* `icon`: (optional) The path to the icon of the content element on dark background

* `darkIcon`: (optional) The path to the icon of the content element on light background

* `group`: (optional) Name of the group in which the content type belongs. It can only be created through the user interface if `group` is defined and it is valid.

  All valid groups are listed inside `TYPO3.Neos.contentTypeGroups`

* `nonEditableOverlay`: (boolean, optional). If TRUE, a non-editable overlay is shown in the backend of Neos.

* `properties`: list of properties for this content type. For each property, the following settings can be done:

  * `type`: type of the property. One of:
    * `boolean`
    * `string`
    * `date`
    * `enum`
    * `TYPO3\Media\Domain\Model\ImageVariant`
    * additional types can be used, as long as they are defined inside `TYPO3.Neos.userInterface` (TODO: bad name!) or specify a JavaScript class inside `userInterface.class`

  * `label`: Label of the property

  * `default`: (optional) Default value. If a new node is created of this type, then it is initialized with this value.

  * `group`: (optional) Name of the *property group* this property is categorized into in the content editing user interface. If none is given, the property is not editable through the property inspector of the user interface.

    The value here must reference a configured property group in the `groups` configuration element of this content element.

  * `priority`: (optional, integer) controls the sorting of the property inside the given `group`. The highest priority is rendered on top (TODO: inconsistent with @position in TypoScript...). Only makes sense if `group` is specified.

  * `reloadOnChange`: (optional) If TRUE, the whole content element needs to be re-rendered on the server side if the value changes. This only works for properties which are displayed inside the property inspector, i.e. for properties which have a `group` set.

  * `options`: (optional) Specify type-specific further options for the given content type. Currently only used if `type=enum`.

  * `userInterface`: (optional) Contains user interface related properties of the property; used for the property inspector. Currently the only supported sub-property is `class`, where an `Ember.View` can be specified which is rendered inside the property inspector.

* `groups`: list of groups inside the *property inspector* for this content type.

Each group has the following settings:

  * `label`: Displayed label of the group
  * `priority`: (integer) Controls the sorting of the groups. The highest-priority group is rendered on top (TODO: inconsistent with @position in TypoScript)

* `inlineEditableProperties`: Array of property names which are editable directly on the page through Create.JS / Aloha. (TODO: check that all properties editable through Aloha / Create.JS ARE indeed marked as inlineEditableProperties; and the other way around as well)

* `structure`: When the given content type is created, the subnodes listed underneath here are automatically created. Example::

```
structure:
  column0:
    type: 'TYPO3.Neos.ContentTypes:Section'
```

Here is an example content type::

```
TYPO3:
  TYPO3CR:
    contentTypes:
      'My.Package:SpecialImageWithTitle':
        label: 'Image'
        superTypes: ['TYPO3.Neos.ContentTypes:ContentObject']
        group: 'General'
        icon: 'Images/Icons/White/picture_icon-16.png'
        darkIcon: 'Images/Icons/Black/picture_icon-16.png'
        properties:
          # the "title" property is not specified here, but is
          # inherited from TYPO3.Neos.ContentTypes:ContentObject
          image:
            type: TYPO3\Media\Domain\Model\ImageVariant
            label: 'Image'
            group: 'image'
            reloadOnChange: true
          imagePosition:
            type: enum
            label: 'Image Position'
            group: 'image'
            default: 'left'
            options:
              values:
                'left':
                  label: 'Left Align'
                'right':
                  label: 'Right Align'
        groups:
          image:
            label: 'Image'
            priority: 5
        inlineEditableProperties: ['title']
```

#3 - 2013-01-30 10:19 - Sebastian Kurfuerst

- Status changed from Accepted to Under Review

#4 - 2013-01-30 16:37 - Sebastian Kurfuerst

- Status changed from Under Review to Resolved