LIFERAY®

CONFIGURING AND CUSTOMIZING ALLOWED CONTENT RULES

ALLOWING CONTENT IN THE EDITOR

- It’s possible to configure the `allowedContent` of an editor through an OSGi module.
- For more information about the allowed content rules, please see the documentation on *CKEditor*: *http://docs.ckeditor.com/#!/guide/dev_allowed_content_rules*

EXTENDING THE EDITOR’S CONFIGURATION

- To modify the editor’s configuration, you can create a module that has a component that implements the `EditorConfigContributor` interface.
- When you implement this interface, your module will provide a service that modifies the editors you’d like to change.

EXTENDING CONFIGURATION EXAMPLE

- As an example, you can create a generic OSGi module using your favorite third-party tool or use the *Blade CLI*.
- You can create a unique package name in the module's `src` directory and create a new Java class in that package.
- The class should extend the `BaseEditorConfigContributor` class.
- Directly above the class's declaration, you can insert a component annotation:

```
@Component(
    property = {

    },
    service = EditorConfigContributor.class
)
```

CHANGING THE ALLOWED CONTENT RULES

- For this example, you could change the `allowedContent` of the Web Content's AlloyEditor. The declaration would look like this:

```
import com.liferay.journal.constants.JournalPortletKeys;
@Component(
    property = {
        "editor.name=alloyeditor",
        "javax.portlet.name=" + JournalPortletKeys.JOURNAL,
        "service.ranking:Integer=100"
    },
    service = EditorConfigContributor.class
)
public class CustomJournalMediaEditorConfigContributor
    extends BaseEditorConfigContributor {
}
```

SPECIFYING CHANGES

- Next, changes need to be specified.
- Add the following method to the new class:

```
@Override
public void populateConfigJSONObject(
    JSONObject jsonObject, Map<String, Object> inputEditorTaglibAttributes,
    ThemeDisplay themeDisplay,
    RequestBackedPortletURLFactory requestBackedPortletURLFactory) {
    ...
}
```

- In the `populateConfigJSONObject` method, the element value can be added for `allowedContent`, `extraPlugins`, and `disallowedContent` through the `jsonObject` parameter:

```
jsonObject.put("allowedContent", "p ");
jsonObject.put("disallowedContent", "h1 h2 h3 h4 h5 h6 ");
```

- It should look like the `01-configuration-example.java` in the exercises folder.

FINISHING UP

- Now if we create a new web content article, the AlloyEditor will disallow all header tags except for <p> tags.

Notes: