**Content Types**

The content types should always be present. This issue should solve the problems when code is based on content types which are not available.

The requirements are using the following documentation on the Symfony website.

<http://symfony.com/doc/3.4/components/config/resources.html>

Here is explained how to `Load Resources` in your project. In the example is spoken about the Yaml loader. We would prefer the Xml loader.

The XML file for specific project content types can be placed into the `app/config/resources/` directory, named like `content\_types.xml`

For example:

* `app/config/content\_types.xml`
* `app/config/pages.xml` (future)
* `app/config/blocks.xml` (future)

The other structures within the Integrated bundles should be for example:

* `integrated/content-bundle/Resources/config/<directory>/content\_types.xml`
* `integrated/content-bundle/Resources/config/<directory>/blocks.xml` (future)
* `integrated/content-bundle/Resources/config/<directory>/pages.xml` (future)

The directories can be registred with the CompilerPass. To check what kind of documents are set into the XML files, it’s an option to add the type of the document in the CompilerPass for example:

$definition->addMethodCall(**’registerResource’**, [**ContentType:class**, **'@IntegratedContentBundle/Resources/config/<directory>/content\_types.xml'**]);

The XML should contain the content types which are required within the application. The only properties off the content type need to be defined in the XML are:

* id (unique identifier)
  + There is no prefix for `content types`. There should always be a check if the unique id is already available within the collection. When the content type already exists, nothing will happen with the one from the XML.
* class ( Class document)
* name (for displaying in listings)
  + This property can be left out, it’s not really required. However it’s easy and more readable.

The content types are stored into the database, in the collection `content\_types`.  
  
The content types can’t be deleted in Integrated when they are added from the XML. Add a property named ‘locked in the document, which is default false, except when it’s loaded from the configuration. When you are browsing in the overview of content types it’s not possible to delete the content types which have the property set disabled to true. The name of the `content type` can be changed by clicking on the edit button. The locked property shouldn’t be set in the Database. Also check the “undefined scenario part” at the end of this document.

There is an existing ContentTypeManager which can be changed with the new way of loading the `content types`.

* \Integrated\Bundle\ContentBundle\Doctrine\ContentTypeManager

This manager is used several other files:

* \Integrated\Bundle\ContentBundle\Tests\Menu\ContentTypeMenuBuilderTest
* \Integrated\Bundle\ContentBundle\Provider\MediaProvider
* \Integrated\Bundle\ContentBundle\Menu\ContentTypeMenuBuilder
* \Integrated\Bundle\ContentBundle\Form\Type\Job\CompanyJobType
* \Integrated\Bundle\ContentBundle\Form\Type\AuthorType

The remove action in Integrated isn’t possible when it’s `locked`. This locked property is only available in the XML and not in the database.

* \Integrated\Bundle\ContentBundle\Controller\ContentTypeController

**Undefined scenario**

There is a possibility when there is already a content type added for example:

* id news
* class ContentType/Video
* label Video News

In the XML is added:

* id news
* class ContentType/News
* label News

What to do in this situation isn’t described in this document. Because the unique identifier is “id” the Content Type manager doesn’t do anything with this.

**No changes**

The code parts below doesn’t need to be changed, because the content types will be added into the database

There are several places to change the code to work with this new way.

The location of the file is:

* \Integrated\Bundle\ContentBundle\Controller\ContentTypeController

Code in the “indexAction” needs to work with the existing Content Type manager.   
There is also a ParamConverter, but this doesn’t need to be changed.