Allow custom rendering of inline images

**Step 1 “Create a listener and register it”:**Create a listener for example the ContentImageListener like the example in Codeblock 1.

After you created the listener you have to register the listener. See the example in Codeblock 2.

**Step 2: “Create a template”**Add the directory objects/image in the default theme configuration.

You can also add the templates in the specific themes. The themeManager -> locatieTemplate function loops threw these paths to lookup in this directory for a template.

The name of the template is added in the class of the element you want to replace. This attribute is prefixed with template-image-[name-of-the-template]

For example a video.html.twig. You also need to add a fallback template just in case there is no template available.

**Step 3: “Call the Twig extension”**

{{ article.content|integrated\_content }}

**Codeblock 1:**

**class** ContentImageListener  
{  
 */\*\*  
 \** ***@var*** *ThemeManager  
 \*/* **protected $themeManager**;  
  
 */\*\*  
 \** ***@var*** *ObjectManager  
 \*/* **protected $objectManager**;  
  
 */\*\*  
 \** ***@var*** *EngineInterface  
 \*/* **protected $templating**;  
  
 */\*\*  
 \** ***@var*** *SluggerInterface  
 \*/* **protected $slugger**;  
  
 */\*\*  
 \** ***@param*** *ThemeManager $themeManager  
 \** ***@param*** *ObjectManager $objectManager  
 \** ***@param*** *EngineInterface $templating  
 \** ***@param*** *SluggerInterface $slugger  
 \*/* **public function** \_\_construct(  
 ThemeManager $themeManager,  
 ObjectManager $objectManager,  
 EngineInterface $templating,  
 SluggerInterface $slugger  
 ) {  
 $this->**themeManager** = $themeManager;  
 $this->**objectManager** = $objectManager;  
 $this->**templating** = $templating;  
 $this->**slugger** = $slugger;  
 }  
  
 */\*\*  
 \** ***@param*** *ContentEvent $contentEvent  
 \** ***@return*** *string  
 \*/* **public function** replaceImages(ContentEvent $contentEvent)  
 {  
 $content = *preg\_replace\_callback*(  
 **'/\<img.\*?data\-integrated\-id\="(.+?)".\*?\>/'**,  
 **function** ($matches) {  
 **return** $this->findImages($matches);  
 },  
 $contentEvent->getContent()  
 );  
  
 $contentEvent->setContent($content);  
 }  
  
 */\*\*  
 \** ***@param*** *array $matches  
 \** ***@return*** *null|string  
 \*/* **protected function** findImages($matches)  
 {  
 **if** ($file = $this->**objectManager**->find(Content::***class***, $matches[1])) {  
 $class = **''**;  
 **if** (*preg\_match*(**'/class="(.\*?)"/'**, $matches[0], $imgClass)) {  
 $class = $imgClass[1];  
 }  
 **return** $this->getTemplate($file, $class);  
 }  
 **return** $matches[0];  
 }  
  
 */\*\*  
 \** ***@param*** *Content $file  
 \** ***@param*** *string $class  
 \** ***@return*** *null|string  
 \*/* **protected function** getTemplate(Content $file, $class = **''**)  
 {  
 **if** ($template = $this->getViewFromClass($class)) {  
 **return** $this->**templating**->renderResponse(  
 $template,  
 [**'document'** => $file, **'class'** => $class]  
 )->getContent();  
 }  
  
 **return null**;  
 }  
  
 */\*\*  
 \** ***@param*** *string $class  
 \** ***@return*** *string  
 \*/* **protected function** getViewFromClass($class = **''**)  
 {  
 **if** (*preg\_match*(**'/template-image-(.\*?)(\s|$)/'**, $class, $views)) {  
 $view = $this->**slugger**->slugify($views[1], **'\_'**);  
  
 **if** ($template = $this->**themeManager**->locateTemplate(**'objects/image/'** . $view . **'.html.twig'**)) {  
 **return** $template;  
 }  
 }  
  
 **return** $this->**themeManager**->locateTemplate(**'objects/image/default.html.twig'**);  
 }  
}

**Codeblock 2:**

<**parameter key="integrated\_content.event\_listener.content\_image\_listener.class"**>Integrated\Bundle\ContentBundle\EventListener\Objects\ContentImageListener</**parameter**>

<**service id="integrated\_content.event\_listener.content\_image\_listener" class="%integrated\_content.event\_listener.content\_image\_listener.class%"**>  
 <**argument type="service" id="integrated\_theme.templating.theme\_manager"** />  
 <**argument type="service" id="doctrine.odm.mongodb.document\_manager"** />  
 <**argument type="service" id="templating"** />  
 <**argument type="service" id="integrated\_slug.slugger"** />  
 <**tag name="kernel.event\_listener" event="content.event" method="replaceImages"** />  
</**service**>