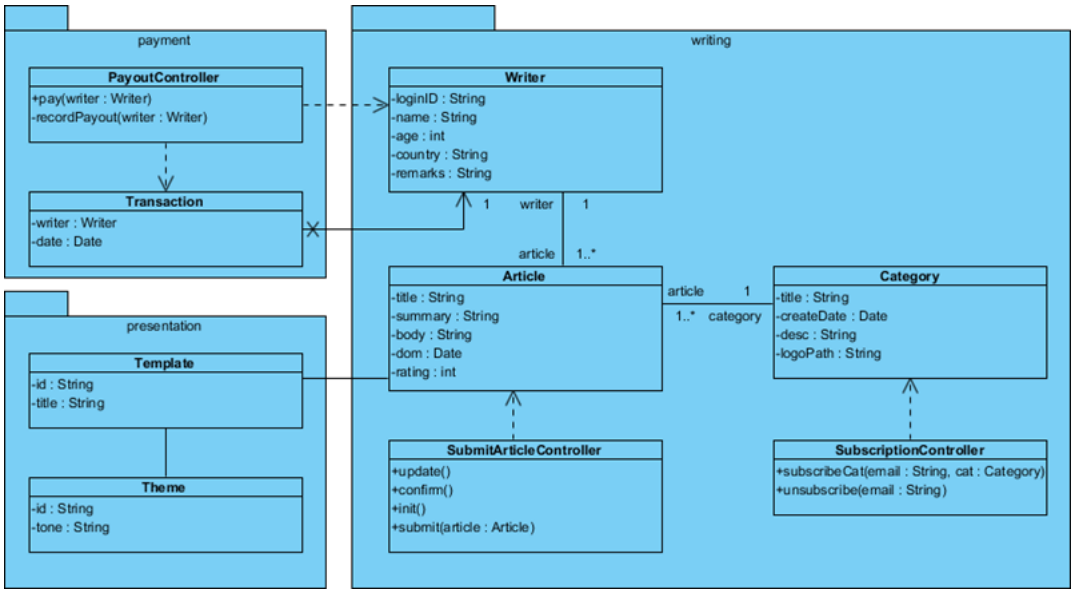
How to Customize Element Templates in Doc. Composer?

Written Date : January 25, 2011

Although Doc. Composer comes with a complete set of built-in element templates, you may still want to customize or even to write your own templates for maximizing the efficiency of document design. You can accomplish this by writing and programming your own element templates. In this article we will show you how to write your own template and use it in building a document.

Getting Started

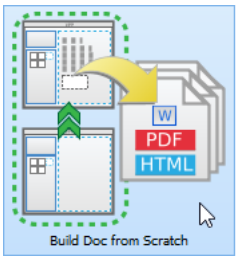
1. Download Article-Writing.vpp. You can also find this file at the bottom of this tutorial, under the **Resources** section.
2. Open the downloaded project file in Visual Paradigm.
3. Open the class diagram *Domain diagram*. The diagram is as below. In this tutorial we are going to write a custom template that lists out the operations' details of all controller classes (i.e. classes with names ended with 'Controller').

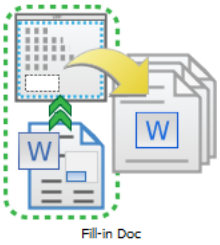


Create an empty document

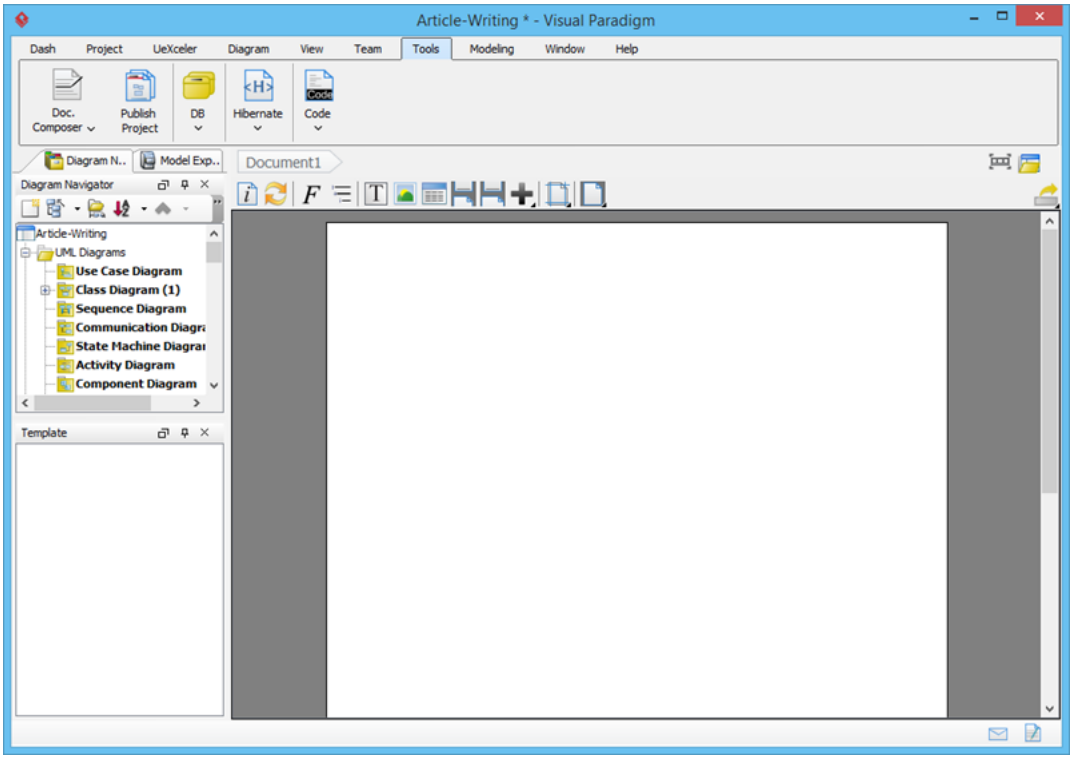
1. Select **Tools > Doc. Composer** from the application toolbar.

2. Click on **Build Doc from Scratch**.





Your screen should look like this:



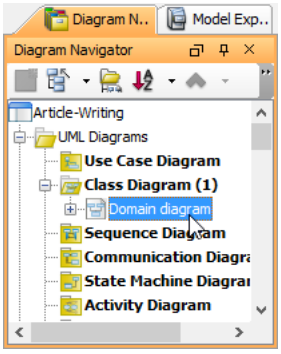
Writing your template

We are going to produce a document that lists out the operations' details of all controller classes (i.e. classes with names ended with 'Controller') in a specific diagram. Here is the structure of document:

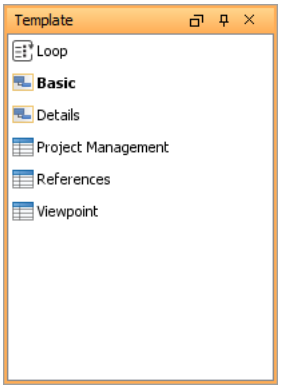
[class diagram image]
[for-each controller class]
[Class name]
[Table of operations]
[/for-each]

We are going to show you how to write an element template to produce such a document. To create a template:

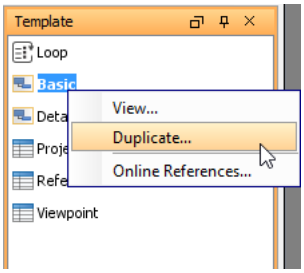
1. Each element template has to be written for a specific kind of project data. It can be a diagram type, a model element type, or even a project. For example, if you want to write a template to list out all use cases in a use case diagram, you should write a template for use case diagram. To begin with, select the type of project data in **Diagram Navigator** or **Model Explorer**. In this case, select the class diagram *Domain diagram* in **Diagram Navigator**.



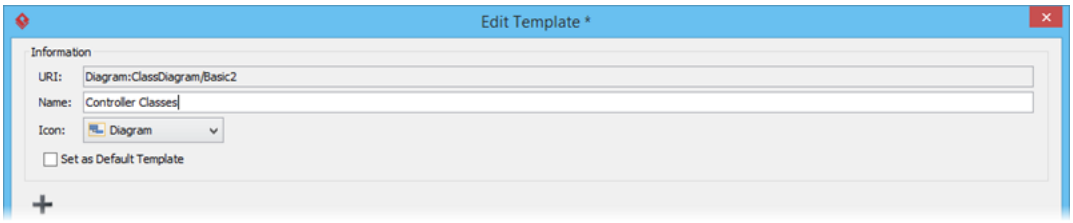
2. The **Template** pane lists the templates available for the selected element type, in this case, class diagram. If your **Template** pane is hidden, select **View > Panes > Property** from the application toolbar (or press **Ctrl+Shift+P**) to show it.



3. You have to duplicate an existing template in order to create your own. Right click on ANY template and select **Duplicate...** from the popup menu.



4. In the **Edit Template** window, enter *Controller Classes* as the template name.

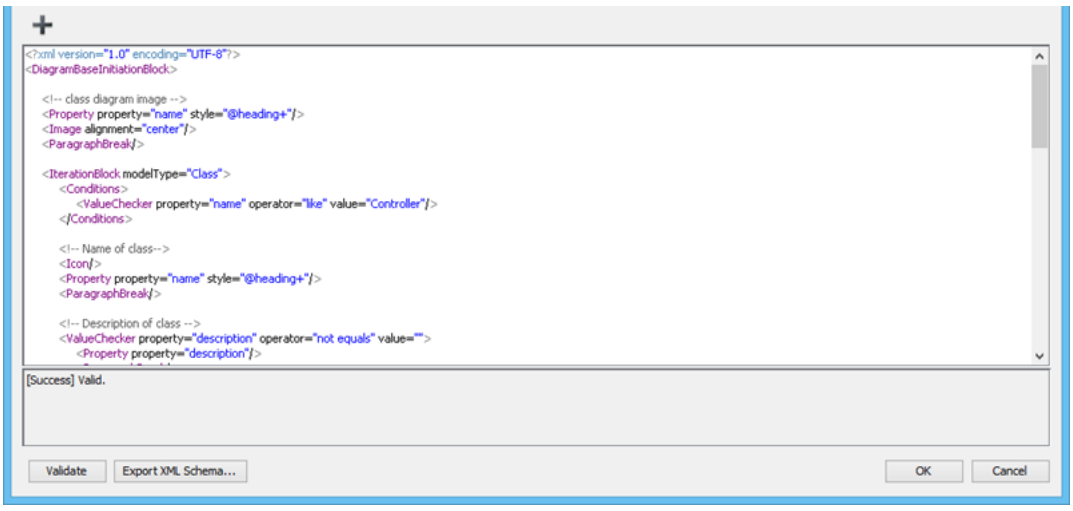


5. Copy the following template content and paste into the template editor.

```
<?xml version="1.0" encoding="UTF-8"?>
<DiagramBaseInitiationBlock>
<!-- class diagram image -->
<Property property="name" style="@heading+"/>
<ParagraphBreak/>
<Image alignment="center"/>
<ParagraphBreak/>
<IterationBlock modelType="Class">
<Conditions>
<ValueChecker property="name" operator="like" value="Controller"/>
</Conditions>
<!-- Name of class-->
<Icon/>
<Property property="name" style="@heading+"/>
<ParagraphBreak/>
<!-- Description of class -->
<ValueChecker property="description" operator="not equals" value="">
<Property property="description"/>
<ParagraphBreak/>
</ValueChecker>
<ElementBaseInitiationBlock>
<!-- Table of operations (show when has operations)-->
<HasChildElementChecker modelType="Operation">
<Text style="@heading+">Operations</Text>
<ParagraphBreak/>
<TableBlock colWidths="35, 65" tableStyle="Summaries">
<TableRow>
<TableCell>
<Text>Name</Text>
</TableCell>
<TableCell>
<Text>Description</Text>
</TableCell>
</TableRow>
<IterationBlock modelType="Operation">
<TableRow>
<TableCell>
<!-- Name of operation-->
```

```
<Property property="name"/>
<!-- Parameters of operation -->
<HasChildElementChecker modelType="Parameter">
<Text> (</Text>
<IterationBlock modelType="Parameter" ignoreLastSeparator="true">
<Property property="name"/>
<HasValueChecker property="type">
<Text> : </Text>
<Property property="type" />
</HasValueChecker>
<Text>, </Text>
</IterationBlock>
<Text>)</Text>
</HasChildElementChecker>
</TableCell>
<TableCell>
<!-- Description of operation-->
<Property property="description" style="Description"/>
</TableCell>
</TableRow>
</IterationBlock>
</TableBlock>
</HasChildElementChecker>
</ElementBaseInitiationBlock>
</IterationBlock>
</DiagramBaseInitiationBlock>
```

Your template editor should look like this:

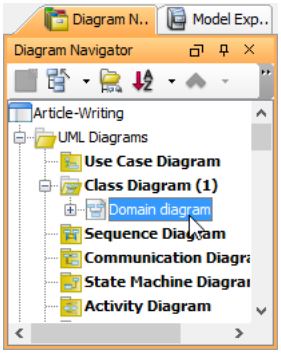


6. Click **OK** to save the changes.

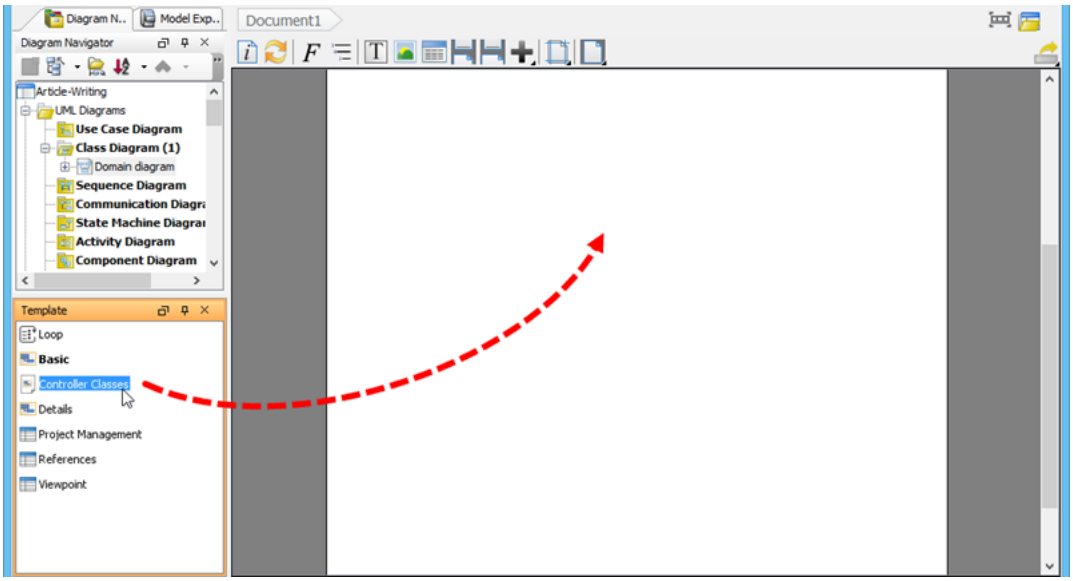
Using your template

Now you've defined a template. Let's develop a document with it.

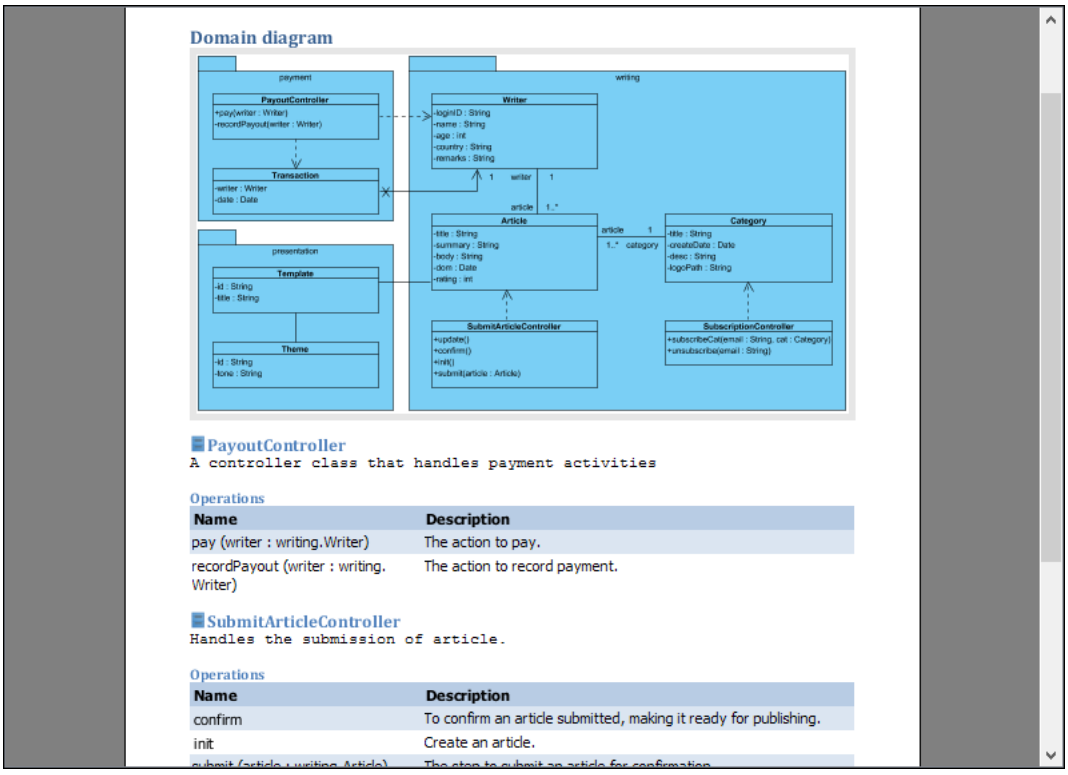
1. Select the class diagram *Domain diagram* from the **Diagram Navigator**.



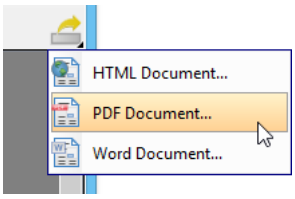
2. Drag the template **Controller Classes** from the **Template** pane and drop it on the document.



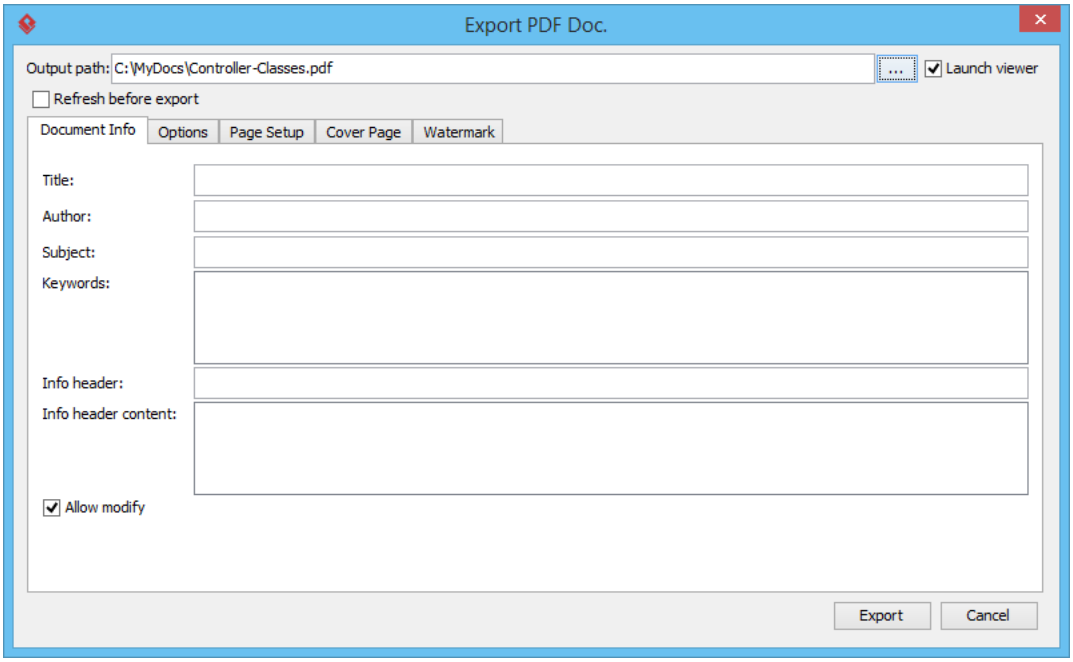
You should see an image of class diagram, along with the details of controller classes presented in the document, like the following.



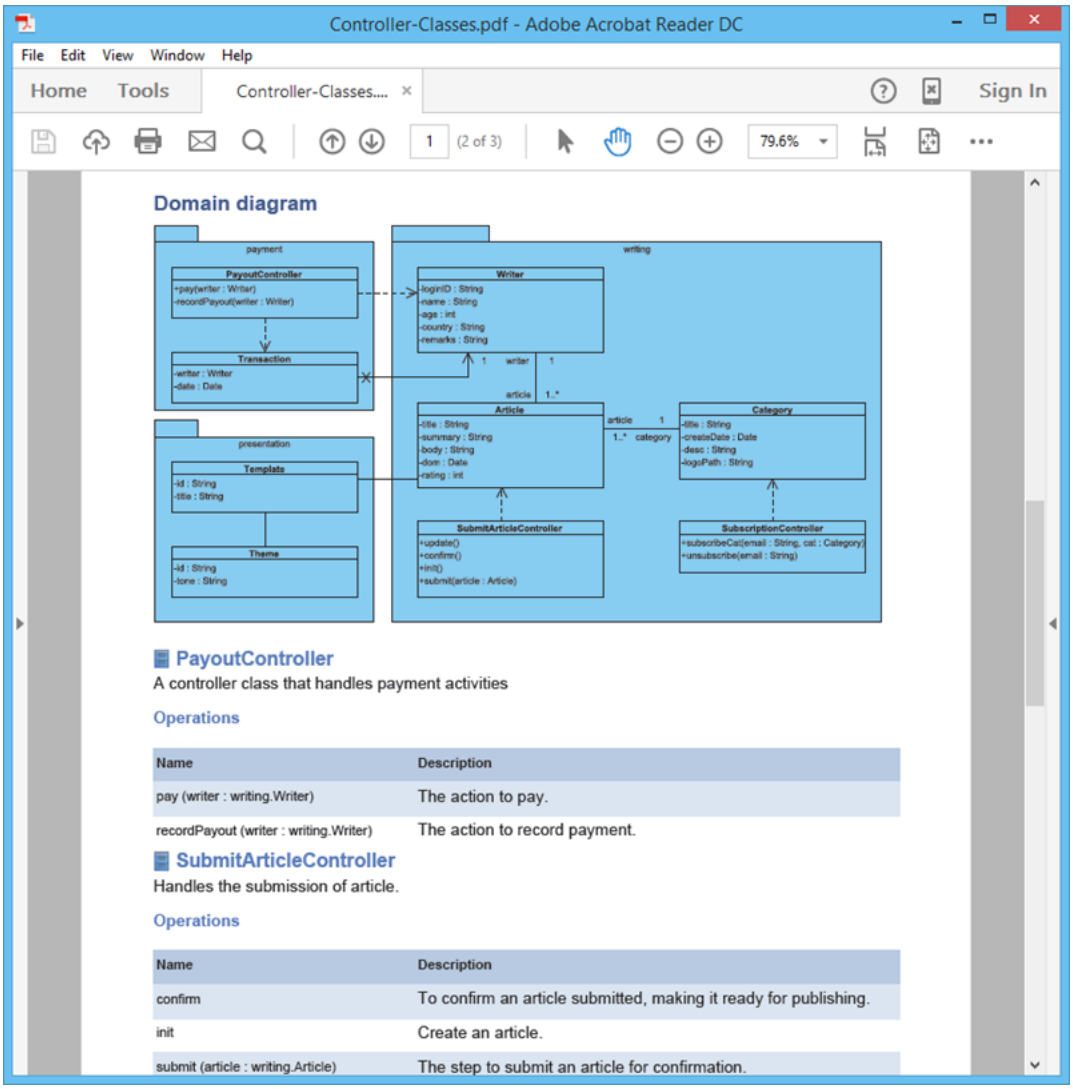
3. Click **Export** at the top-right corner of the document and select **PDF Document** from the pop-up menu.



4. In the **Export PDF Doc.** window, enter the output path.



5. Click **Export**. You obtain a PDF document like this:



Resources

1. Article-Writing.vpp

Related Links

- Tutorial - Writing Software Requirements Specification with Doc. Composer

Visual Paradigm home page
(https://www.visual-paradigm.com/)

Visual Paradigm tutorials
(https://www.visual-paradigm.com/tutorials/)