Connection to the properties file containing model path <drop/>

<properties path=’${project\_loc}/template/conf.properties’ /> <drop/>

Ouput path is located in property file <drop/>

<config>

<output path=’${outputPath}’ />

</config>

<context model=’${model}’ element=’{0}’ importedBundles=’gmf;papyrus’/>

<gendoc>[self.oclAsType(uml::NamedElement).name.clean()/]</gendoc>

Document generated by Gendoc (<http://www.eclipse.org/gendoc>)

Help and support at : <https://www.eclipse.org/forums/index.php/f/286/>

Contenu

<gendoc> <drop/>

[self.ownedElement->filter(Model)->sortedBy(name).modelDocumentation()/]<drop/>

</gendoc> <drop/>

# chapter : This fragment generates recursively chapters for each sub packages <drop/>

<fragment name=’chapter’ importedFragments=’displayComment;title;diagrams’ > <drop/>

<arg name=’e’ type=’uml::Package’/> <drop/>

<arg name=’val’ type=’Integer’/> <drop/>

Display the diagrams of the package <drop/>

[self.diagrams (val + 1)/] <drop/>

</fragment>

Fragments declaration <drop/>

# title : This fragment generates the correct style for headings <drop/>

<fragment name=’title’> <drop/>

<arg name=’s’ type=’String’/> <drop/>

<arg name=’e’ type=’Integer’/> <drop/>

[if (e = 1)] <drop/>

# [s/]

[elseif (e = 2)] <drop/>

## [s/]

[elseif (e = 3)] <drop/>

### [s/]

[elseif (e = 4)] <drop/>

#### [s/]

[elseif (e = 5)] <drop/>

##### [s/]

[elseif (e = 6)] <drop/>

###### [s/]

[/if] <drop/>

</fragment> <drop/>

# displayComment : This fragment handles correctly the comments <drop/>

<fragment name=’displayComment’> <drop/>

<arg name=’e’ type=’uml::Element’/> <drop/>

[for (co:uml::Comment | e.ownedComment)]<drop/>

<dropEmpty>[co.\_body.clean()/]</dropEmpty>

[/for]<drop/>

</fragment> <drop/>

# diagrams : generates the diagrams of the given element <drop/>

<fragment name=’diagrams’ importedBundles=’gmf;papyrus’ importedFragments=’title;displayComment’> <drop/>

<arg name=’e’ type=’uml::Element’/> <drop/>

<arg name=’val’ type=’Integer’/> <drop/>

[for (d : notation::Diagram | e.getPapyrusDiagrams())] <drop/>

The diagram is display if it is not empty <drop/>

[if (not d.isDiagramEmpty())] <drop/>

[d.name.clean().title(val)/]

<image object=’[d.getDiagram()/]’ maxW=’true’> <drop/>

Figure 1 : [d.name.clean()/]

</image> <drop/>

For each element in the diagram which is not a package the diagram is displayed <drop/>

[for (eobject : uml::Element | d.getElementsInDiagram())] <drop/>

[if (eobject.oclIsKindOf(uml::NamedElement))] <drop/>

[eobject.oclAsType(uml::NamedElement).name.clean().title(val)/] <drop/>

[eobject.displayComment()/] <drop/>

[/if] <drop/>

[/for] <drop/>

[/if] <drop/>

[/for] <drop/>

</fragment> <drop/>

# ’diagramDocumentation’: generates the documentation <drop/>

<fragment name=’diagramDocumentation’ importedBundles=’gmf;papyrus’ importedFragments=’documentElementsInDiagram;title;displayComment’> <drop/>

<arg name=’d’ type= ‘notation::Diagram’/> <drop/>

[d.name.clean().title(2)/]<drop/>

<image object=’[d.getDiagram()/]’ maxW=’true’> <drop/>

</image> <drop/>

[d.documentElementsInDiagram()/]<drop/>

</fragment> <drop/>

# documentElementsInDiagram: Document elements in diagrams <drop/>

<fragment name=‘documentElementsInDiagram’ importedBundles=’gmf;papyrus’ importedFragments=’title;displayComment’> <drop/>

<arg name=’d’ type= ‘notation::Diagram’/> <drop/>

[for (eobject : uml::Element | d.getElementsInDiagram()->filter(uml:: NamedElement)->sortedBy(name))] <drop/>

[eobject.oclAsType(uml::NamedElement).name.clean().title(3)/]<drop/>

[eobject.displayComment()/]<drop/>

[/for]<drop/>

</fragment> <drop/>

# modelDocumentation: Document elements in diagrams <drop/>

<fragment name=‘modelDocumentation’ importedBundles=’gmf;papyrus’ importedFragments=‘diagramDocumentation;packageDocumentation;title’> <drop/>

<arg name=’m’ type= ‘Model’/> <drop/>

[m.name.clean().title(1)/]<drop/>

[for (d:Diagram | m.getPapyrusDiagrams()->sortedBy(name))]<drop/>

[d.diagramDocumentation()/]<drop/>

[/for] <drop/>

[for (u:UseCase | m.ownedElement->filter(UseCase)->sortedBy(name))]<drop/>

[u.usecaseDocumentation()/]<drop/>

[/for]<drop/>

[for (p:Package | m.ownedElement->filter(Package)->sortedBy(name))]<drop/>

[p.packageDocumentation()/]<drop/>

[/for]<drop/>

[for (c:Component | m.ownedElement->filter(Component)->sortedBy(name))]<drop/>

[c.componentDocumentation()/]<drop/>

[/for]<drop/>

[for (n:Node | m.ownedElement->filter(Node)->sortedBy(name))]<drop/>

[n.nodeDocumentation()/]<drop/>

[/for]<drop/>

</fragment><drop/>

# usecaseDocumentation: Document elements in diagrams <drop/>

<fragment name=‘usecaseDocumentation’ importedBundles=’gmf;papyrus’ importedFragments=’diagramDocumentation’> <drop/>

<arg name=’p’ type= ‘UseCase’ /> <drop/>

[for (d:Diagram | p.getPapyrusDiagrams()->sortedBy(name))]<drop/>

… [d.diagramDocumentation()/]<drop/>

[/for]<drop/>

[for (inner\_usecase:UseCase | p.ownedElement->filter(UseCase)->sortedBy(name))]<drop/>

[inner\_usecase.usecaseDocumentation()/]<drop/>

[/for] <drop/>

</fragment><drop/>

# packageDocumentation: Document elements in diagrams <drop/>

<fragment name=‘packageDocumentation’ importedBundles=’gmf;papyrus’ importedFragments=’diagramDocumentation’> <drop/>

<arg name=’p’ type= ‘Package’ /> <drop/>

[for (d:Diagram | p.getPapyrusDiagrams()->sortedBy(name))]<drop/>

… [d.diagramDocumentation()/]<drop/>

[/for]<drop/>

[for (inner\_package:Package | p.ownedElement->filter(Package)->sortedBy(name))]<drop/>

[inner\_package.packageDocumentation()/]<drop/>

[/for] <drop/>

</fragment><drop/>

# componentDocumentation: Document elements in diagrams <drop/>

<fragment name=‘componentDocumentation’ importedBundles=’gmf;papyrus’ importedFragments=’diagramDocumentation’> <drop/>

<arg name=’c’ type= ‘Component’ /> <drop/>

[for (d:Diagram | c.getPapyrusDiagrams()->sortedBy(name))]<drop/>

… [d.diagramDocumentation()/]<drop/>

[d.name.clean().title(2)/]<drop/>

[/for]<drop/>

[for (inner\_component:Component | c.ownedElement->filter(Component)->sortedBy(name))]<drop/>

[inner\_component.componentDocumentation()/]<drop/>

[/for] <drop/>

</fragment><drop/>

# nodeDocumentation: Document elements in diagrams <drop/>

<fragment name=‘nodeDocumentation’ importedBundles=’gmf;papyrus’ importedFragments=’diagramDocumentation’> <drop/>

<arg name=’n’ type= ‘Node’ /> <drop/>

[for (d:Diagram | n.getPapyrusDiagrams()->sortedBy(name))]<drop/>

… [d.diagramDocumentation()/]<drop/>

[/for]<drop/>

[for (inner\_node:Node| n.ownedElement->filter(Component)->sortedBy(name))]<drop/>

[inner\_node.nodeDocumentation()/]<drop/>

[/for] <drop/>

</fragment><drop/>