**File**

#######################################################

#

# %**fileName**%

# Python implementation of the %**elemType**% %**className**%

# Generated by Enterprise Architect

# Created on: %**eaDateTime**%

%**if** **classAuthor** != ""%

# Original author: %**classAuthor**%

%**endIf**%

#

#######################################################

%***ImportSection***%\n

%**list**="Namespace" @separator="\n\n" @indent=""%

**Namespace**

%**list**="Namespace" @separator="\n\n"%

%**list**="Class" @separator="\n\n"%

**Class**

%***ClassDeclaration***%

%***ClassNotes***%

%***ClassBody***%

**Class Notes**

%**PI**=""%

%**if** **genOptGenComments** != "T"%

%**endTemplate**%

$notes = %**REPLACE**(**classNotes**, "\n", "\n ")%

%**if** $notes != ""%

%qt%%qt%%qt%

$notes = %**WRAP\_COMMENT**($notes, **genOptWrapComment**, "", " ")%

%**TRIM**($notes)%\n

%qt%%qt%%qt%

%**endIf**%

**Class Base**

%**classBaseName**%

**Class Interface**

%***ClassBase***%

**Class Body**

$attr=%**list**="Attribute" @separator="\n" @indent=" "%

$ops=%**list**="Operation" @separator="\n\n" @indent=" "%

$iclasses=%**list**="InnerClass" @separator="\n\n" @indent=" "%

%**if** $iclasses != ""%

$iclasses\n

%**endIf**%

$attr

$ops

%**if** $attr == "" **and** $ops == "" **and** $iclasses == ""%

pass

%**endIf**%

**Class Declaration**

$dec=%**REPLACE**(**classTag**:"decorators","@","\n@")%

%**TRIM**($dec)%

class %**className**%%***ClassInherits***%:

**Class Inherits**

%**PI**=""%

$bases=%**list**="ClassBase" @separator=", "%

$interfaces=%**list**="ClassInterface" @separator=", "%

%**if** $bases != "" **and** $interfaces != ""%

($bases, $interfaces)

%**elseIf** $bases != ""%

($bases)

%**elseIf** $interfaces != ""%

($interfaces)

%**endIf**%

**Attribute**

%**if** **attInitial** != ""%

%***AttributeNotes***%

%**PI**=""%

%***AttributeDeclaration***%

%**endIf**%

**Attribute Notes**

%**if** **genOptGenComments** != "T"%

%**endTemplate**%

%**WRAP\_COMMENT**(**attNotes**, **genOptWrapComment**, "", "# ")%

**Attribute Declaration**

%**PI**=""%

%**if** **attScope** == "Private"%

\_\_

%**endIf**%

%**attName**% = %**attInitial**%

**Import**

$package = %**importPackagePath**%

$file = %**importFileName**%

%**if** $file != ""%

$imports = %**list**="ImportImpl" @separator="," $package == **importPackagePath** **and** $file == **importFileName**%

$pos = %**FIND**($file, ".py")%

%**if** $pos != "-1"%

$file=%**LEFT**($file,$pos)%

%**else**%

$file=%**importClassName**%

%**endIf**%

%**else**%

$imports = %***ImportImpl***%

$file = %**importClassName**%

%**endIf**%

$imports = %**REMOVE\_DUPLICATES**($imports,",")%

$imports = %**REPLACE**($imports, ",", ", ")%

%**PI**=" "%

%**if** $package!= ""%

from $package.$file

%**endIf**%

import $imports

**Import Impl**

%**importClassName**%

**Import Section**

$imports = %**fileImports**%

$imports += %**fileHeaders**%

$imports += %**list**="Import" @separator="\n" **importFromAggregation** == "T" **or** **importFromAssociation** == "T" **or** **importFromAtt** == "T" **or** **importFromGeneralization** == "T" **or** **importFromMeth** == "T" **or** **importFromParam** == "T" **or** **importFromRealization** == "T" **and** **importInFile** != "T"%

%**REMOVE\_DUPLICATES**($imports, "\n")%

**Linked Attribute**

%***LinkedAttributeNotes***%

%***LinkedAttributeDeclaration***%

**Linked Attribute Notes**

%**if** **genOptGenComments** != "T"%

%**endTemplate**%

%**WRAP\_COMMENT**(**linkAttNotes**, **genOptWrapComment**, "", "# ")%

**Linked Attribute Declaration**

%**PI**=""%

%**if** **linkAttAccess** == "Private"%

\_\_

%**endIf**%

%**if** **linkAttRole** != ""%

%**linkAttRole**%

%**else**%

%**REPLACE**(**genOptDefaultAssocAttName**,"$LinkClass",**linkAttName**)%

%**endIf**%

= %**linkAttName**%()\n

**Linked Class Base**

%**linkParentName**%

**Linked Class Interface**

%***LinkedClassBase***%

**Operation**

%***OperationDeclaration***%

%***OperationNotes***%

%***OperationBody***%

**Operation Notes**

%**PI**=""%

%**if** **genOptGenComments** != "T"%

%**endTemplate**%

$notes = %**REPLACE**(**opNotes**, "\n", "\n ")%

%**if** $notes != ""%

%qt%%qt%%qt%

$notes = %**WRAP\_COMMENT**($notes, **genOptWrapComment**, "", " ")%

%**TRIM**($notes)%\n

%qt%%qt%%qt%

%**endIf**%

**Operation Declaration**

$dec=%**REPLACE**(**opTag**:"decorators","@","\n@")%

%**TRIM**($dec)%

%**PI**=" "%

$params = %**list**="Parameter" @separator=", "%

def

%**PI**=""%

%**if** **opScope** == "Private"%

\_\_

%**endIf**%

%**opName**%

($params):

**Operation Body**

$behaviour = %**WRAP\_COMMENT**(opBehaviour, "40", " ", "# ")%

%**if** $behaviour != ""%

$behaviour\n\n

%**endIf**%

pass

**Parameter**

%**PI**=""%

%**paramName**%

%**if** **paramDefault** != "" **and** **paramDefault** != "<none>"%

= %**paramDefault**%

%**endIf**%

**List Macro**

If you need to loop or iterate through a set of Objects that are contained within or are under the current object, you can do so using the **%list** macro. This macro performs an iterative pass on all the objects in the scope of the current template, and calls another template to process each one.

The basic structure is:

%list=<TemplateName> @separator=<string> @indent=<string> ( <conditions> ) %

where <*string*> is a double-quoted literal string and <*TemplateName*> can be one of the following template names:

Attribute, AttributeImpl, Class, ClassBase, ClassImpl, ClassInterface, Constraint, Custom Template (custom templates enable you to define your own templates), Effort, InnerClass, InnerClassImpl, LinkedFile, Metric, Namespace, Operation, OperationImpl, Parameter, Problem, Requirement, Resource, Risk, Scenario, Test

<*conditions*> is optional and looks the same as the conditions for *if* and *elseIf* statements.

Example

In a Class transform, the Class might contain multiple Attributes; this example calls the **Attribute** transform and outputs the result of processing the transform for each attribute of the Class in scope. The resultant list separates its items with a single new line and indents them two spaces respectively. If the Class in scope had any **stereotyped** attributes, they would be generated using the appropriately specialized template.

%list="Attribute" @separator="\n" @indent="  "%

The **separator** attribute, denoted above by *@separator*, specifies the space that should be used between the list items, excluding the last item in the list.

The **indent** attribute, denoted by *@indent*, specifies the space by which each line in the generated output should be indented.

Special Cases

There are some special cases to consider when using the **%list** macro:

|  |  |
| --- | --- |
|  | If the **Attribute** template is used as an argument to the **%list** macro, this also generates attributes derived from Associations by executing the appropriate **LinkedAttribute** template |
|  | If the **ClassBase** template is used as an argument to the **%list** macro, this also generates Class bases derived from links in the model by executing the appropriate **LinkedClassBase** template |

|  |  |
| --- | --- |
|  | If the**ClassInterface** template is used as an argument to the **%list** macro, this also generates Class bases derived from links in the model by executing the appropriate **LinkedClassInterface** template |
|  | If**InnerClass** or ***InnerClassImpl*** is used as an argument to the **%list** macro, these Classes are generated using the **Class** and **ClassImpl** templates respectively; these arguments direct that the templates should be processed based on the inner Classes of the Class in scope |