Content

[2 Classes 1](#_Toc71815365)

[2.1 [cl.name/] 1](#_Toc71815366)

[3 Data Types 2](#_Toc71815367)

[3.1 [dt.name/] 2](#_Toc71815368)

[4 Enumeration Types 3](#_Toc71815369)

[4.1 [dt.name/] 3](#_Toc71815370)

[5 Primitive Types 3](#_Toc71815371)

[5.1 [dt.name/] 3](#_Toc71815372)

# Classes

## [cl.name/]

[co.\_body.clean()/]

Applied stereotypes:

* [st.name/]

Attributes for [cl.name/]

Table 1: Attributes for [cl.name/]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Type** | **Multiplicity** | **Access** | **Stereotypes** | **Description** |
| [p.name/][if(not p.qualifiedName.contains(cl.name))], Inherited[/if] | [p.type.name/]  [if p.defaultValue->notEmpty()][p.defaultValue.stringValue()/][else]./.[/if] | [if(p.lower=p.upper)]1[else][p.lower/]..[if(p.upper=-1)]\*[else][p.upper/][/if][/if] | [if(not(p.isReadOnly))]RW[else]R[/if] | [st.name/]  • [if oa.name.contains('part')]partOfObjectKey: [p.getValue(st, oa.name).oclAsType(Integer)/]  • [if oa.name.contains('attribute')]AVC: [p.getValue(st, oa.name).oclAsType(EnumerationLiteral).name/]  • [if oa.name.contains('Invariant')]isInvariant: [p.getValue(st, oa.name).oclAsType(Boolean)/]  • [if oa.name.contains('value')]valueRange: [if (not p.getValue(st, oa.name).oclIsUndefined())][p.getValue(st, oa.name).oclAsType(String)/][else]no range constraint[/if]  • [if oa.name.contains('Length')]bitLength: [p.getValue(st, oa.name).oclAsType(EnumerationLiteral).name/]  • [if oa.name.contains('unit')]unit: [if (not p.getValue(st, oa.name).oclIsUndefined())][p.getValue(st, oa.name).oclAsType(String)/][else]no unit defined[/if]  • [if oa.name.contains('support')]support: [p.getValue(st, oa.name).oclAsType(EnumerationLiteral).name/] | [c.\_body.clean()/]  [else][if (p.name.contains ('\_'))] See referenced class  [else] |

# Data Types

## [dt.name/]

[co.\_body.clean()/]

Applied Stereotypes:

* [st.name/]

Attributes for [dt.name/]

Table 1: Attributes for [dt.name/]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Type** | **Multiplicity** | **Access** | **Stereotypes** | **Description** |
| [p.name/] | [p.type.name/]  [if p.defaultValue->notEmpty()][p.defaultValue.stringValue()/][else]./.[/if] | [if(p.lower=p.upper)]1[else][p.lower/]..[if(p.upper=-1)]\*[else][p.upper/][/if][/if] | [if(not(p.isReadOnly))]RW[else]R[/if] | [st.name/]   * [if oa.name.contains('part')]partOfObjectKey: [p.getValue(st, oa.name).oclAsType(Integer)/] * [if oa.name.contains('attribute')]AVC: [p.getValue(st, oa.name).oclAsType(EnumerationLiteral).name/] * [if oa.name.contains('Invariant')]isInvariant: [p.getValue(st, oa.name).oclAsType(Boolean)/] * [if oa.name.contains('value')]valueRange: [if (not p.getValue(st, oa.name).oclIsUndefined())][p.getValue(st, oa.name).oclAsType(String)/][else]no range constraint[/if] * [if oa.name.contains('Length')]bitLength: [p.getValue(st, oa.name).oclAsType(EnumerationLiteral).name/] * [if oa.name.contains('unit')]unit: [if (not p.getValue(st, oa.name).oclIsUndefined())][p.getValue(st, oa.name).oclAsType(String)/][else]no unit defined[/if] * [if oa.name.contains('support')]support: [p.getValue(st, oa.name).oclAsType(EnumerationLiteral).name/] | [c.\_body.clean()/] |

# Enumeration Types

## [dt.name/]

[co.\_body.clean()/]

Contains Enumeration Literals:

* [e.name/]:
  + [co.\_body.clean()/]

# Primitive Types

## [dt.name/]

[co.\_body.clean()/]