

# EcoreDoc User Guide

Niko Stotz

Version 0.9.0-SNAPSHOT

# Table of Contents

1. Overview .....	1
2. Java API .....	1
3. Maven Plugin .....	2
4. Standalone Command-Line Tool .....	4
5. Eclipse Plug-in .....	6
6. EcoreDoc Metamodel Annotation .....	6
6.1. Ecore Annotation .....	6
6.2. Xcore Annotation .....	7
7. Generator Configuration .....	7
7.1. Abstract Class AEReferenceConfig .....	8
7.2. Class EAttributeConfig .....	8
7.3. Class EClassConfig .....	9
7.4. Class EContainmentConfig .....	11
7.5. Class EDataTypeConfig .....	11
7.6. Class EEnumConfig .....	12
7.7. Class EEnumLiteralConfig .....	13
7.8. Class EOperationConfig .....	14
7.9. Class EPackageConfig .....	15
7.10. Class EParameterConfig .....	16
7.11. Class EReferenceConfig .....	17
7.12. Class EcoreDocGeneratorConfig .....	18
7.13. Interface IDefaultValueConfig .....	25
7.14. Interface IEAttributeConfig .....	26
7.15. Interface IEClassConfig .....	26
7.16. Interface IEClassifierConfig .....	28
7.17. Interface IEDatatypeConfig .....	28
7.18. Interface IEEnumConfig .....	28
7.19. Interface IEEnumLiteralConfig .....	28
7.20. Interface IENamedElementConfig .....	29
7.21. Interface IEOperationConfig .....	31
7.22. Interface IEPackageConfig .....	31
7.23. Interface IEParameterConfig .....	32
7.24. Interface IEReferenceConfig .....	32
7.25. Interface IEStructuralFeatureConfig .....	33
7.26. Interface IETypedElementConfig .....	33
8. Versions .....	33
9. Known Issues .....	34

Generates [AsciiDoctor](#) files to document Ecore metamodels, similar to [JavaDoc](#). AsciiDoctor can be rendered as HTML, PDF, or Eclipse Help. EcoreDoc can be used as [Maven](#) Plugin, standalone command-line tool, Java API, or Eclipse Plug-in.

# 1. Overview

EcoreDoc's [Java API](#) works on a list of [EClassifiers](#). [Maven Plugin](#), [Standalone Command-Line Tool](#) and [Eclipse Plug-in](#) take all [EClassifiers](#) from one or more [\\*.ecore](#) or [\\*.xcore](#) files.

EcoreDoc creates one output document containing all passed [EClassifiers](#). They are grouped by containing [EPackage](#). The output document contains documentation, all properties of supported elements, and cross-references to all usages of each element.

EcoreDoc currently supports the following elements:

- [EPackage](#)
- [EDatatype](#)
- [EEnum](#)
- [EEnumLiteral](#)
- [EClass](#)
- [EAttribute](#)
- [EReference](#)
- [EOperation](#)
- [EParameter](#)

EcoreDoc is highly configurable via the [Generator Configuration](#).

The homepage and repository of EcoreDoc can be found at <https://gitlab.manatree.io/MDEAssets/EcoreDoc>. Please use the issue tracker at this site for any feature requests or bugs.

# 2. Java API

The [Java API](#) is available as [Maven artifact `com.altran.general.emf.ecoredoc:com.altran.general.emf.ecoredoc.generator`](#) or [OSGi bundle `com.altran.general.emf.ecoredoc.generator.ebr`](#).

The [Generator Configuration](#) is contained in [Maven artifact `com.altran.general.emf.ecoredoc:com.altran.general.emf.ecoredoc.generator.config`](#) or [OSGi bundle `com.altran.general.emf.ecoredoc.generator.config.ebr`](#).

The main interface is [com.altran.general.emf.ecoredoc.generator.EcoreDocGenerator](#). The constructor takes the list of [EClassifiers](#) to generate documentation for.

The [getConfig\(\)](#) method returns a fully initialized [com.altran.general.emf.ecoredoc.generator.config.EcoreDocGeneratorConfig](#) that can be changed to adjust the [Generator Configuration](#).

The `generate()` method returns a `CharSequence` containing the complete AsciiDoctor output document.

## 3. Maven Plugin

The Maven Plugin is available as Maven artifact `com.altran.general.emf.ecoredoc:ecoredoc-maven-plugin`.

It supports the following configuration settings:

### **inputFiles (required)**

The list of Ecore metamodel files to create documentation for.

### **outputFile (required)**

The output file to write the generated AsciiDoctor document to.

By convention, the file extension is `.adoc`.



If the file exists, it will be overwritten and a warning is emitted.

### **resolve (default: false)**

Whether EcoreDoc should explicitly try to resolve all references in the *inputFiles*. Might be necessary for highly interconnected metamodels.

### **config (default: unchanged default config)**

Customized [Generator Configuration](#).

The *config* contents stricly follow the structure and naming relative to [Class EcoreDocGeneratorConfig](#), easiest explained with an example.

Assume the *inputFiles* contain two `EPackage`s, namely `EPackage1` and `EPackage2`.

`EPackage1` contains, among others, two `EClasses`, named `MyEClass` and `Class3`. The latter one contains, among others, the `EAttribute` named `specialNumber`.

`EPackage1` also contains an `EEnum` named `Enum1`.

*pom.xml*

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <artifactId>my-artifact-id</artifactId>

  <dependencies>
    <dependency>
      <groupId>com.altran.general.emf.ecoredoc</groupId>
      <artifactId>ecoredoc-maven-plugin</artifactId>
```

```

</dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>com.altran.general.emf.ecoredoc</groupId>
      <artifactId>ecoredoc-maven-plugin</artifactId>

      <configuration>
        <resolve>true</resolve>

        <config>
          <renderDefaults>false</renderDefaults>
          <documentTitle>This is the title of my document</documentTitle>
          <ePackages>
            <ePackage>
              <targetEPackage>EPackage1</targetEPackage>
              <eClasses>
                <eClass>
                  <targetEClass>MyEClass</targetEClass>
                  <repeatInherited>false</repeatInherited>
                </eClass>
                <eClass>
                  <targetEClass>Class3</targetEClass>
                  <eAttributes>
                    <eAttribute>
                      <targetEAttribute>specialNumber</targetEAttribute>
                      <render>false</render>
                    </eAttribute>
                  </eAttributes>
                </eClass>
              </eClasses>
            <eEnums>
              <eEnum>
                <targetEEnum>Enum1</targetEEnum>
                <renderDefaults>true</renderDefaults>
              </eEnum>
            </eEnums>
          </ePackage>
          <ePackage>
            <targetEPackage>EPackage2</targetEPackage>
            <renderDefaults>true</renderDefaults>
          </ePackage>
        </ePackages>
      </config>
      <inputFiles>
        <inputFile>EPackage1.ecore</inputFile>
        <inputFile>EPackage2.ecore</inputFile>
      </inputFiles>
      <outputFile>output.adoc</outputFile>
    </plugin>
  </plugins>
</build>

```

```

        </configuration>
      </plugin>
    </plugins>
  </build>
</project>

```

This example sets the following configuration:

- `renderDefaults` for all contents: `true`
- `documentTitle`: `This is the title of my document`
- `repeatInherited` for `MyEClass`: `false`
- `render` for `specialNumber`: `false`
- `renderDefaults` for `Enum1`: `true`
- `renderDefaults` for `EPackage2`: `true`

## 4. Standalone Command-Line Tool

The standalone command-line tool is available as Maven artifact `com.altran.general.emf.ecoredoc:com.altran.general.emf.ecoredoc.standalone`.

Use the following command to invoke. Please replace `${ecoredoc-version}` with your version of EcoreDoc:

```
java -jar com.altran.general.emf.ecoredoc.standalone-${ecoredoc-version}-jar-with-dependencies.jar <options>
```

If invoked without options, it will print the following help:

Generates reference documentation for ecore models.

The output is inspired by JavaDoc and formatted in AsciiDoctor format. AsciiDoctor can easily be rendered to HTML, PDF, or Eclipse help.

Usage:

EcoreDocGenerator [parameters] [List of ecore files to generate]

If unspecified, the output file name will be "<firstEcoreFile.ecore>.adoc"

Parameters:

```
-r,
--resolve: Resolve external references
```

```

-o <outputFile>,
--output <outputFile>: Specify output file name.

--documentTitle <title>: Set title of output document

--positionEDataTypes <pos>: Set rendering position of all EDataTypes within EPackage

--positionEEnums <pos>:      Set rendering position of all EEnums within EPackage

--positionEClasses <pos>:    Set rendering position of all EClasses within EPackage

[+|-]defaults:  [Enable|disable] rendering of default values

[+|-]bounds:    [Enable|disable] rendering of multiplicity bounds
                (overwrites defaults parameter)

[+|-]inherited: [Enable|disable] repetition of inherited features

[+|-]useCases:  [Enable|disable] rendering of use cases
                (references to other usages of this element)

[+|-]subTypes:  [Enable|disable] rendering of sub-types

[+|-]superTypes: [Enable|disable] rendering of super-types

```

#### Examples:

```

EcoreDocGenerator my.ecore
Generates the documentation of my.ecore into my.ecore.adoc

```

```

EcoreDocGenerator some/path/to/my.ecore other.ecore
Generates the documentation of some/path/to/my.ecore and other.ecore
into some/path/to/my.ecore.adoc

```

```

EcoreDocGenerator -r my.ecore
Tries to resolve all external references in my.ecore and
generates the documentation of my.ecore and referenced models into my.ecore.adoc

```

```

EcoreDocGenerator -defaults +bounds my.ecore
Generates the documentation of my.ecore and referenced models into my.ecore.adoc
without rendering default values, but still rendering multiplicity bounds

```

```

EcoreDocGenerator --positionEClasses 1 --positionEEnums 2 --positionEDataTypes 3
my.ecore

```

Generates the documentation of my.ecore and referenced models into my.ecore.adoc with all EClasses first, then all EEnums, and finally all EDataTypes

```
EcoreDocGenerator -o output.adoc my.ecore other.ecore
```

Generates the documentation of my.ecore and other.ecore into output.adoc

## 5. Eclipse Plug-in

The Eclipse Plug-in is available as Feature `com.altran.general.emf.ecoredoc.ui.feature`.

It provides a context menu entry for one or more `*..ecore` / `*.xcore` files in the following views:

- Project Explorer
- Package Explorer
- Model Explorer

The command creates one output file next to the first selected input file, named `<firstInputFile.ecore>.adoc`. The output file contains the documentation of all selected metamodels.

## 6. EcoreDoc Metamodel Annotation

Any of the [Generator Configuration](#) options can be used as Ecore Annotation. These options will be used by default; any external options take precedence over annotation options.

EcoreDoc will throw an `IllegalArgumentException` if an EcoreDoc annotation contains an illegal value.

### 6.1. Ecore Annotation

Create an `EAnnotation` on the annotated element with source

```
http://altran.com/general/emf/ecoredoc/generator/config/0.1
```

Within this annotation, create one key/value pair for each option.



The source identifier might change in the future! However, it should be possible to maintain backwards compability.



Property	Value
References	
Source	http://altran.com/general/emf/ecoredoc/generator/config/0.1

Example Ecore annotation on EAttribute **id** (sets *render* for **id** to **false**)

## 6.2. Xcore Annotation

First, register the annotation.

*EcoreDoc* annotation registration

```
annotation "http://altran.com/general/emf/ecoredoc/generator/config/0.1" as EcoreDoc
```

Afterwards, we can use the annotation as usual.

Example Xcore annotation on EAttribute **name** (sets *render* for **name** to **false**)

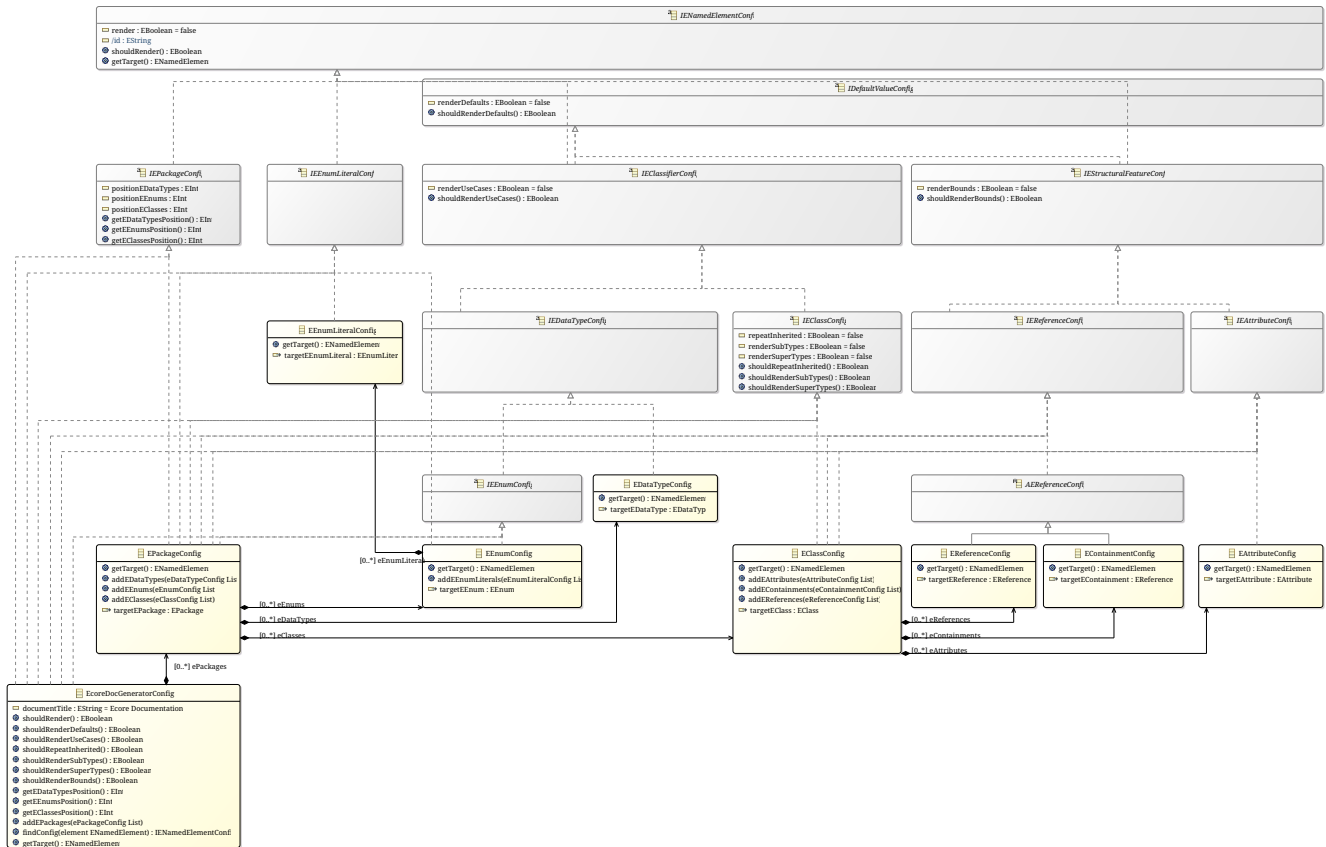
```
@EcoreDoc(
    render="false"
)
String name
```

## 7. Generator Configuration

The generator configuration is an Ecore metamodel, so we obviously use EcoreDoc to create the documentation listed below.

The most important parts are:

- Class [EcoreDocGeneratorConfig](#) as model root, also describing the customization hierarchy
- Interface [INamedElementConfig](#), implemented by all elements
- Interface [IDefaultValueConfig](#), implemented by all elements except Class [EEnumLiteralConfig](#)
- Interface [IEClassifierConfig](#)
- Interface [IEClassConfig](#)
- Interface [IEPackageConfig](#)
- Interface [IEStructuralFeatureConfig](#)



Generator Configuration Class Diagram

## 7.1. Abstract Class AEReferenceConfig

### Super-types

- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)
- [config.IReferenceConfig](#)
- [config.IStructuralFeatureConfig](#)
- [config.ITypedElementConfig](#)

## 7.2. Class EAttributeConfig

### Super-types

- [config.IDefaultValueConfig](#)
- [config.IEAttributeConfig](#)
- [config.INamedElementConfig](#)
- [config.IStructuralFeatureConfig](#)
- [config.ITypedElementConfig](#)

### References

Name	Type	Properties	Description
<a href="#">targetEAttribute</a>	<a href="#">ecore.EAttribute</a>	[0..1]	

### Operations

Name	Aspect and Type	Properties	Description
<b>getTarget()</b> ▲ config. INamedElementConfig. getTarget()	returns ecore. ENamedElement	[0..1]	
targetEAttribute			

Used at

- config.EClassConfig.eAttributes

## 7.3. Class EClassConfig

Super-types

- config.IDefaultValueConfig
- config IEAttributeConfig
- config.EClassConfig
- config.EClassifierConfig
- config.INamedElementConfig
- config.IEOperationConfig
- config.IEParameterConfig
- config.IEReferenceConfig
- config.IEStructuralFeatureConfig
- config.IETypedElementConfig

Containments

Name	Type	Properties	Description
eAttributes	config. EAttributeConfig	[0..*]	
eContainments	config. EContainmentConfig	[0..*]	
eOperations	config. EOperationConfig	[0..*]	
eReferences	config. EReferenceConfig	[0..*]	

References

Name	Type	Properties	Description
targetEClass	ecore.EClass	[0..1]	

Operations

Name	Aspect and Type	Properties	Description
addEAttributes( eAttributeConfig)	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBean Helper#setProperty()} to handle ELists correctly.
	eAttributeConfig config.List	[0..1]	
	EAttributes += eAttributeConfig		
addEContainments( eContainmentConfig )	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBean Helper#setProperty()} to handle ELists correctly.
	eContainmentConfig config.List	[0..1]	
	EContainments += eContainmentConfig		
addEOperations( eOperationConfig)	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBean Helper#setProperty()} to handle ELists correctly.
	eOperationConfig config.List	[0..1]	
	EOperations += eOperationConfig		
addEReferences( eReferenceConfig)	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBean Helper#setProperty()} to handle ELists correctly.
	eReferenceConfig config.List	[0..1]	
	EReferences += eReferenceConfig		

Name	Aspect and Type	Properties	Description
<b>getTarget()</b>  ▲ config. INamedElementConfig. getTarget()	<i>returns</i>  ecore. ENamedElement	[0..1]	
<div>targetEClass</div>			

Used at

- config.EPackageConfig.eClasses

## 7.4. Class EContainmentConfig

Super-types

- config.AEReferenceConfig
- config.IDefaultValueConfig
- config.INamedElementConfig
- config.IEReferenceConfig
- config.IEStructuralFeatureConfig
- config.IETypedElementConfig

References

Name	Type	Properties	Description
targetEContainment	ecore.EReference	[0..1]	

Operations

Name	Aspect and Type	Properties	Description
<b>getTarget()</b>  ▲ config. INamedElementConfig. getTarget()	<i>returns</i>  ecore. ENamedElement	[0..1]	
<div>targetEContainment</div>			

Used at

- config.EClassConfig.eContainments

## 7.5. Class EDataTypeConfig

Super-types

- config.IDefaultValueConfig
- config.IEClassifierConfig
- config.IEDataTypeConfig
- config.INamedElementConfig

### References

Name	Type	Properties	Description
<code>targetEDataType</code>	<code>ecore.EDataType</code>	<code>[0..1]</code>	

### Operations

Name	Aspect and Type	Properties	Description
<code>getTarget()</code>  ▲ <code>config.INamedElementConfig.getTarget()</code>	<i>returns</i> <code>ecore.ENamedElement</code>	<code>[0..1]</code>	
targetEDataType			

### Used at

- `config.EPackageConfig.eDataTypes`

## 7.6. Class EEnumConfig

### Super-types

- `config.IDefaultValueConfig`
- `config.IEClassifierConfig`
- `config.IEDataTypeConfig`
- `config.IEEnumConfig`
- `config.IEEnumLiteralConfig`
- `config.INamedElementConfig`

### Containments

Name	Type	Properties	Description
<code>eEnumLiterals</code>	<code>config.EEnumLiteralConfig</code>	<code>[0..*]</code>	

### References

Name	Type	Properties	Description
<code>targetEEnum</code>	<code>ecore.EEnum</code>	<code>[0..1]</code>	

### Operations

Name	Aspect and Type	Properties	Description
addEEnumLiterals( eEnumLiteralConfig )	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBeanHelper#setProperty()} to handle ELists correctly.
	eEnumLiteralConfig config.List	[0..1]	
	EEnumLiterals += eEnumLiteralConfig		
getTarget()  ▲ config. INamedElementConf ig.getTarget()	returns ecore. ENamedElement	[0..1]	
	targetEEnum		

Used at

- [config.EPackageConfig.eEnums](#)

## 7.7. Class EEnumLiteralConfig

Super-types

- [config.IEEnumLiteralConfig](#)
- [config.INamedElementConfig](#)

References

Name	Type	Properties	Description
targetEEnumLiteral	ecore.EEnumLiteral	[0..1]	

Operations

Name	Aspect and Type	Properties	Description
getTarget()  ▲ config. INamedElementConf ig.getTarget()	returns ecore. ENamedElement	[0..1]	
	targetEEnumLiteral		

Used at

- [config.EEnumConfig.eEnumLiterals](#)

## 7.8. Class EOperationConfig

### Super-types

- [config.AEReferenceConfig](#)
- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)
- [config.IEReferenceConfig](#)
- [config.IEStructuralFeatureConfig](#)
- [config.ITypedElementConfig](#)

### Containments

Name	Type	Properties	Description
eParameters	<a href="#">config.EParameterConfig</a>	[0..*]	

### References

Name	Type	Properties	Description
targetEOperation	<a href="#">ecore.EOperation</a>	[0..1]	

### Operations

Name	Aspect and Type	Properties	Description
addEParameters( <a href="#">eParameterConfig</a> )	returns void	[0..1]	Helper method for <a href="#">{@linkplain org.eclipse.sisu.plexus.CompositeBeanHelper#setProperty()}</a> to handle ELists correctly.
	<a href="#">eParameterConfig</a> <a href="#">config.List</a>	[0..1]	
	EParameters += eParameterConfig		
getId()  ▲ <a href="#">config.INamedElementConfig.id</a>	returns <a href="#">EString</a>	[0..1]	
	targetEOperation?.joinId		
getTarget()  ▲ <a href="#">config.INamedElementConfig.getTarget()</a>	returns <a href="#">ecore.ENamedElement</a>	[0..1]	
	targetEOperation		



Name	Aspect and Type	Properties	Description
joinId(eOperation)	returns	[0..1]	
	EString		
	eOperation ecore.EOperation	[0..1]	
eOperation.name + eOperation.EParameters.map[(EType?.eContainer as ENamedElement)?.name + "_" + EType?.name].join(".")			

Used at

- [config.EClassConfig.eOperations](#)

## 7.9. Class EPackageConfig

Super-types

- [config.IDefaultValueConfig](#)
- [config.IEAttributeConfig](#)
- [config.IEClassConfig](#)
- [config.IEClassifierConfig](#)
- [config.IEDataTypeConfig](#)
- [config.IEEnumConfig](#)
- [config.IEEnumLiteralConfig](#)
- [config.IENamedElementConfig](#)
- [config.IEOperationConfig](#)
- [config.IEPackageConfig](#)
- [config.IEParameterConfig](#)
- [config.IEReferenceConfig](#)
- [config.IEStructuralFeatureConfig](#)
- [config.IETypedElementConfig](#)

Containments

Name	Type	Properties	Description
eClasses	<a href="#">config.EClassConfig</a>	[0..*]	
eDataTypes	<a href="#">config.EDatatypeConfig</a>	[0..*]	
eEnums	<a href="#">config.EEnumConfig</a>	[0..*]	

References

Name	Type	Properties	Description
targetEPackage	<a href="#">ecore.EPackage</a>	[0..1]	

Operations

Name	Aspect and Type	Properties	Description
addEClasses( eClassConfig)	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBeanHelper#setProperty()} to handle ELists correctly.
	eClassConfig config.List	[0..1]	
	EClasses += eClassConfig		
addEDataTypes( eDataTypeConfig)	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBeanHelper#setProperty()} to handle ELists correctly.
	eDataTypeConfig config.List	[0..1]	
	EDataTypes += eDataTypeConfig		
addEEnums( eEnumConfig)	returns void	[0..1]	Helper method for {@linkplain org.eclipse.sisu.plexus.CompositeBeanHelper#setProperty()} to handle ELists correctly.
	eEnumConfig config.List	[0..1]	
	EEnums += eEnumConfig		
getTarget()  ▲ config. IENamedElementConf ig.getTarget()	returns ecore. ENamedElement	[0..1]	
	targetEPackage		

Used at

- [config.EcoreDocGeneratorConfig.ePackages](#)

## 7.10. Class EParameterConfig


Super-types

- [config.AEReferenceConfig](#)
- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)
- [config.IEReferenceConfig](#)
- [config.IEStructuralFeatureConfig](#)
- [config.ITypedElementConfig](#)

#### References

Name	Type	Properties	Description
<a href="#">targetEParameter</a>	<a href="#">ecore.EParameter</a>	[0..1]	

#### Operations

Name	Aspect and Type	Properties	Description
<a href="#">getTarget()</a>  <a href="#">config.INamedElementConfig.getTarget()</a>	returns <a href="#">ecore.ENamedElement</a>	[0..1]	
<div>targetEParameter</div>			

#### Used at

- [config.EOperationConfig.eParameters](#)

## 7.11. Class EReferenceConfig


#### Super-types

- [config.AEReferenceConfig](#)
- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)
- [config.IEReferenceConfig](#)
- [config.IEStructuralFeatureConfig](#)
- [config.ITypedElementConfig](#)

#### References

Name	Type	Properties	Description
<a href="#">targetEReference</a>	<a href="#">ecore.EReference</a>	[0..1]	

#### Operations

Name	Aspect and Type	Properties	Description
<code>getTarget()</code>  <code>config.INamedElementConfig.getTarget()</code>	<i>returns</i> <code>ecore.ENamedElement</code>	<code>[0..1]</code>	
<div>targetEReference</div>			

Used at

- `config.EClassConfig.eReferences`

## 7.12. Class EcoreDocGeneratorConfig

Root for the detailed EcoreDocGenerator configuration.

The configuration allows to specify configuration options for each element and all its contained elements. It always chooses the most specific configuration setting.

Example:

```
EcoreDocGeneratorConfig * renderDefaults: {unset, defaults to true} * repeatInherited: false +
EPackage1 * renderDefaults: false + EClass1 + EAttribute1 * renderDefaults: true + EAttribute2 {no
custom config} + EClass2 extends EClass1 + EPackage2 * repeatInherited: true + EClass3 extends
EClass1 + EClass4 + EAttribute3 * renderDefaults: true * repeatInherited: false
```

Result:

EPackage1

```
renderDefaults false
repeatInherited false
```

EClass1

```
renderDefaults false
repeatInherited false
```

EAttribute1

```
renderDefaults true
repeatInherited false
```

EAttribute2

renderDefaults false  
repeatInherited false

EClass2

renderDefaults false  
repeatInherited false

EPackage2

renderDefaults true  
repeatInherited true

EClass3

renderDefaults true  
repeatInherited true

EClass4

renderDefaults true  
repeatInherited true

EAttribute3

renderDefaults true  
repeatInherited false

#### *Super-types*

- [config.IDefaultValueConfig](#)
- [config.IEAttributeConfig](#)
- [config.IEClassConfig](#)

- [config.IEClassifierConfig](#)
- [config.IEDataTypeConfig](#)
- [config.IEEnumConfig](#)
- [config.IEEnumLiteralConfig](#)
- [config.IENamedElementConfig](#)
- [config.IEOperationConfig](#)
- [config.IEPackageConfig](#)
- [config.IEParameterConfig](#)
- [config.IEReferenceConfig](#)
- [config.IEStructuralFeatureConfig](#)
- [config.IETypedElementConfig](#)

#### Attributes

Name	Type	Properties	Description
<a href="#">documentTitle</a>	<a href="#">EString</a>	<a href="#">[0..1]</a> <i>Default: Ecore Documentation</i>	Title of the generated document. defaults to Ecore Documentation.

#### Containments

Name	Type	Properties	Description
<a href="#">ePackages</a>	<a href="#">config.EPackageConfig</a>	<a href="#">[0..*]</a>	

#### Operations

Name	Aspect and Type	Properties	Description
<a href="#">addEPackages( ePackageConfig)</a>	<i>returns</i> <a href="#">void</a>	<a href="#">[0..1]</a>	Helper method for <code>{@linkplain org.eclipse.sisu.plexus.CompositeBeanHelper#setProperty()}</code> to handle ELists correctly.
	<a href="#">ePackageConfig</a> <a href="#">config.List</a>	<a href="#">[0..1]</a>	
	EPackages += ePackageConfig		



19	element	[0..1]	
	ecore.		
Name	Aspect and Type	Properties	Description

```

switch (element) {
  EPackage:
    EPackages

  EEnum:
    EPackages
    .map[EEnums]
    .flatten

  EDataType:
    EPackages
    .map[EDataTypes]
    .flatten

  EEnumLiteral:
    EPackages
    .map[EEnums]
    .flatten
    .map[EEnumLiterals]
    .flatten

  EClass:
    EPackages
    .map[EClasses]
    .flatten

  EAttribute:
    EPackages
    .map[EClasses]
    .flatten
    .map[EAttributes]
    .flatten

  EReference case (element.isContainment):
    EPackages
    .map[EClasses]
    .flatten
    .map[EContainments]
    .flatten

  EReference case (!element.isContainment):
    EPackages
    .map[EClasses]
    .flatten
    .map[EReferences]
    .flatten

  EOperation:
    EPackages
    .map[EClasses]

```



Name	Aspect and Type	Properties	Description
<b>getEClassesPosition()</b>  ▲ config. IEPackageConfig. getEClassesPosition()	returns EInt	[0..1]	Sets default for positionEClasses = 3.
<pre> if (isSetPositionEClasses) {     positionEClasses } else {     3 } </pre>			
<b>getEDataTypesPosition()</b>  ▲ config. IEPackageConfig. getEDataTypesPosition()	returns EInt	[0..1]	Sets default for positionEDataTypes = 1.
<pre> if (isSetPositionEDataTypes) {     positionEDataTypes } else {     1 } </pre>			
<b>getEEnumsPosition()</b>  ▲ config. IEPackageConfig. getEEnumsPosition()	returns EInt	[0..1]	Sets default for positionEEnums = 2.
<pre> if (isSetPositionEEnums) {     positionEEnums } else {     2 } </pre>			
<b>getTarget()</b>  ▲ config. IENamedElementConfig. getTarget()	returns ecore. ENamedElement	[0..1]	
<pre> null </pre>			
<b>shouldRender()</b>  ▲ config. IENamedElementConfig. shouldRender()	returns EBoolean	[0..1]	Sets default for render = true.
<pre> if (isSetRender) {     render } else {     true } </pre>			

Name	Aspect and Type	Properties	Description
<b>shouldRenderBounds()</b>  ▲ config. ITypedElementConfig. shouldRenderBounds()	returns EBoolean	[0..1]	Sets default for renderBounds = shouldRenderDefaults().
<pre> if (isSetRenderBounds) {     renderBounds } else {     shouldRenderDefaults() } </pre>			
<b>shouldRenderDefaults()</b>  ▲ config. IDefaultValueConfig. shouldRenderDefaults()	returns EBoolean	[0..1]	Sets default for renderDefaults = true.
<pre> if (isSetRenderDefaults) {     renderDefaults } else {     true } </pre>			
<b>shouldRenderSubTypes()</b>  ▲ config. IEClassConfig. shouldRenderSubTypes()	returns EBoolean	[0..1]	Sets default for renderSubTypes = true.
<pre> if (isSetRenderSubTypes) {     renderSubTypes } else {     true } </pre>			
<b>shouldRenderSuperTypes()</b>  ▲ config. IEClassConfig. shouldRenderSuperTypes()	returns EBoolean	[0..1]	Sets default for RenderSuperTypes = true.
<pre> if (isSetRenderSuperTypes) {     renderSuperTypes } else {     true } </pre>			

Name	Aspect and Type	Properties	Description
<b>shouldRenderUseCases()</b>  ▲ config. IEClassifierConfig. shouldRenderUseCases()	returns EBoolean	[0..1]	Sets default for renderUseCases = true.
<pre> if (isSetRenderUseCases) {     renderUseCases } else {     true } </pre>			
<b>shouldRepeatInherited()</b>  ▲ config. IEClassConfig. shouldRepeatInherited()	returns EBoolean	[0..1]	Sets default for repeatInherited = true.
<pre> if (isSetRepeatInherited) {     repeatInherited } else {     true } </pre>			

## 7.13. Interface IDefaultValueConfig

### Attributes

Name	Type	Properties	Description
<b>renderDefaults</b>	EBoolean	[0..1] unsettable	<p>Whether properties should be rendered at their default values.</p> <p>Example: If EReference.ordered = true (the default value), the ordered property of that EReference will not be rendered if renderDefaults = false.</p>

### Operations

Name	Aspect and Type	Properties	Description
<code>shouldRenderDefaults()</code>  ▼ <code>config.EcoreDocGeneratorConfig.shouldRenderDefaults()</code>	returns EBoolean	[0..1]	Traverses the tree to find the most specific renderDefaults setting.
<pre> if (isSetRenderDefaults) {     renderDefaults } else {     (eContainer as IDefaultValueConfig).shouldRenderDefaults() } </pre>			

## 7.14. Interface IEAttributeConfig

Super-types

- `config.IDefaultValueConfig`
- `config.INamedElementConfig`
- `config.IStructuralFeatureConfig`
- `config.ITypedElementConfig`

## 7.15. Interface IEClassConfig

Super-types

- `config.IDefaultValueConfig`
- `config.IEClassifierConfig`
- `config.INamedElementConfig`

Attributes

Name	Type	Properties	Description
<code>renderSubTypes</code>	EBoolean	[0..1] unsettable	Whether the list of sub-types should be rendered.
<code>renderSuperTypes</code>	EBoolean	[0..1] unsettable	Whether the list of super-types should be rendered.

Name	Type	Properties	Description
repeatInherited	EBoolean	[0..1] unsettable	Whether inherited features should be repeated.  Example: EClass1 has an EAttribute name=attr1. EClass2 extends EClass1. If repeatInherited = true for EClass2, attr1 will be listed in the section of EClass1 and EClass2. Otherwise, attr1 will only be listed in the section of EClass1.

### Operations

Name	Aspect and Type	Properties	Description
shouldRenderSubTypes()  ▼ config. EcoreDocGeneratorConfig. shouldRenderSubTypes()	returns EBoolean	[0..1]	Traverses the tree to find the most specific renderSubTypes setting.
<pre> if (isSetRenderSubTypes) {     renderSubTypes } else {     (eContainer as IEClassConfig).shouldRenderSubTypes() } </pre>			
shouldRenderSuperTypes()  ▼ config. EcoreDocGeneratorConfig. shouldRenderSuperTypes()	returns EBoolean	[0..1]	Traverses the tree to find the most specific renderSuperTypes setting.
<pre> if (isSetRenderSuperTypes) {     renderSuperTypes } else {     (eContainer as IEClassConfig).shouldRenderSuperTypes() } </pre>			
shouldRepeatInherited()  ▼ config. EcoreDocGeneratorConfig. shouldRepeatInherited()	returns EBoolean	[0..1]	Traverses the tree to find the most specific repeatInherited setting.
<pre> if (isSetRepeatInherited) {     repeatInherited } else {     (eContainer as IEClassConfig).shouldRepeatInherited() } </pre>			

## 7.16. Interface IEClassifierConfig

### Super-types

- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)

### Attributes

Name	Type	Properties	Description
<code>renderUseCases</code>	<code>EBoolean</code>	<code>[0..1]</code> unsettable	Whether use cases (references to other usages of this element) should be rendered.

### Operations

Name	Aspect and Type	Properties	Description
<code>shouldRenderUseCases()</code>  ▼ <a href="#">config.EcoreDocGeneratorConfig.shouldRenderUseCases()</a>	<code>returns EBoolean</code>	<code>[0..1]</code>	Traverses the tree to find the most specific <code>renderUseCases</code> setting.  <pre>if (isSetRenderUseCases) {     renderUseCases } else {     (eContainer as IEClassifierConfig).shouldRenderUseCases() }</pre>

## 7.17. Interface IEDataTypeConfig

### Super-types

- [config.IDefaultValueConfig](#)
- [config.IEClassifierConfig](#)
- [config.INamedElementConfig](#)

## 7.18. Interface IEnumConfig

### Super-types

- [config.IDefaultValueConfig](#)
- [config.IEClassifierConfig](#)
- [config.IEDataTypeConfig](#)
- [config.INamedElementConfig](#)

## 7.19. Interface IEnumLiteralConfig

### Super-types

- [config.INamedElementConfig](#)

## 7.20. Interface INamedElementConfig

### Attributes

Name	Type	Properties	Description
<code>render</code>	<code>EBoolean</code>	<code>[0..1]</code> unsettable	Whether this element should be rendered at all.

### Operations

Name	Aspect and Type	Properties	Description
<b>abstract</b> <b>getTarget()</b>  ▼ config. EAttributeConfig. getTarget()  ▼ config. EClassConfig. getTarget()  ▼ config. EContainmentConfig. getTarget()  ▼ config. EDatatypeConfig. getTarget()  ▼ config. EEnumConfig. getTarget()  ▼ config. EEnumLiteralConfig. getTarget()  ▼ config. EOperationConfig. getTarget()  ▼ config. EPackageConfig. getTarget()  ▼ config. EParameterConfig. getTarget()  ▼ config. EReferenceConfig. getTarget()  ▼ config. EcoreDocGeneratorC onfig.getTarget()	returns ecore. ENamedElement	[0..1]	Link to the configured element.



Name	Aspect and Type	Properties	Description
<b>shouldRender()</b>  ▼ <a href="#">config.EcoreDocGeneratorConfig.shouldRender()</a>	returns <b>EBoolean</b>	<b>[0..1]</b>	Traverses the tree to find the most specific render setting.
<pre> if (isSetRender) {     render } else {     (eContainer as INamedElementConfig).shouldRender() } </pre>			

Used at

- [config.EcoreDocGeneratorConfig.findConfig\(element\)](#)

## 7.21. Interface IEOperationConfig

Super-types

- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)
- [config.ITypedElementConfig](#)

## 7.22. Interface IEPackageConfig

Super-types

- [config.INamedElementConfig](#)

Attributes

Name	Type	Properties	Description
<b>positionEClasses</b>	<b>EInt</b>	<b>[0..1]</b> unsettable	Rendering position of all EClasses within an EPackage.
<b>positionEDataTypes</b>	<b>EInt</b>	<b>[0..1]</b> unsettable	Rendering position of all EDataTypes within an EPackage.
<b>positionEEnums</b>	<b>EInt</b>	<b>[0..1]</b> unsettable	Rendering position of all EEnums within an EPackage.

Operations

Name	Aspect and Type	Properties	Description
<b>getEClassesPosition()</b>  ▼ config. EcoreDocGeneratorC onfig. getEClassesPosition()	returns EInt	[0..1]	Traverses the tree to find the most specific positionEClasses setting.
<pre> if (isSetPositionEClasses) {     positionEClasses } else {     (eContainer as IEPackageConfig).getEClassesPosition() } </pre>			
<b>getEDataTypesPosition()</b>  ▼ config. EcoreDocGeneratorC onfig. getEDataTypesPosition()	returns EInt	[0..1]	Traverses the tree to find the most specific positionEDataTypes setting.
<pre> if (isSetPositionEDataTypes) {     positionEDataTypes } else {     (eContainer as IEPackageConfig).getEDataTypesPosition() } </pre>			
<b>getEEnumsPosition()</b>  ▼ config. EcoreDocGeneratorC onfig. getEEnumsPosition()	returns EInt	[0..1]	Traverses the tree to find the most specific positionEEnums setting.
<pre> if (isSetPositionEEnums) {     positionEEnums } else {     (eContainer as IEPackageConfig).getEEnumsPosition() } </pre>			

## 7.23. Interface IEPackageConfig

Super-types

- config.IDefaultValueConfig
- config.INamedElementConfig
- config.ITypedElementConfig

## 7.24. Interface IEReferenceConfig

Super-types

- config.IDefaultValueConfig
- config.INamedElementConfig
- config.IStructuralFeatureConfig

- [config.ITypedElementConfig](#)

## 7.25. Interface IEStructuralFeatureConfig

Super-types

- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)
- [config.ITypedElementConfig](#)

## 7.26. Interface ITypedElementConfig

Super-types

- [config.IDefaultValueConfig](#)
- [config.INamedElementConfig](#)

Attributes

Name	Type	Properties	Description
<code>renderBounds</code>	<code>EBoolean</code>	<code>[0..1]</code> unsettable	Whether multiplicity bounds should be rendered, even if they are at their default values and <code>renderDefaults = false</code> .

Operations

Name	Aspect and Type	Properties	Description
<code>shouldRenderBounds()</code>  ▼ <code>config.EcoreDocGeneratorC</code> <code>onfig.shouldRenderBounds()</code>	<code>returns</code> <code>EBoolean</code>	<code>[0..1]</code>	Traverses the tree to find the most specific <code>renderBounds</code> setting.
<pre> if (isSetRenderBounds) {     renderBounds } else if(isSetRenderDefaults) {     renderDefaults } else {     (eContainer as ITypedElementConfig).shouldRenderBounds() } </pre>			

## 8. Versions

This asset in version 0.9.0-SNAPSHOT was developed using the following components and versions.

**Eclipse**

4.7.3a (Oxygen 3a)

**Google Guava**

19.0

**Apache Commons Lang3**

3.1

**Apache Commons IO**

2.2

**Apache Maven**

3.3.9

**Eclipse Ecore**

2.12.0

**Eclipse Xcore**

1.3.1

**Eclipse Tycho**

1.2.0

## 9. Known Issues

- If HTML is used in Ecore documentation, the PDF rendering can be faulty ([Issue #12](#))
- **EAnnotations** are missing from the documentation ([Issue #15](#))