UML Standard Elements A

This appendix describes the predefined standard elements for UML. The standard elements are organized into categories (stereotypes, tagged values, and constraints) and are alphabetically ordered.

A.1 Stereotypes

The following stereotypes are predefined in the UML. Any stereotype that applies to a specific class in the metamodel also applies to any subclasses of that class.

Name	Applies to	Description
«becomes»	Dependency	Becomes is a stereotyped dependency whose source and target represent the same instance at different points in time, but each with potentially different values, state instance, and roles. A becomes dependency from A to B means that that instance A becomes B with possibly new values, state instance, and roles at a different moment in time/space.
«call»	Dependency	Call is a stereotyped dependency whose source is an operation and whose target is an operation. A call dependency specifies that the source invokes the target operation. A call dependency may connect a source operation to any target operation that is within scope including, but not limited to, operations of the enclosing classifier and operations of other visible classifiers.

Name	Applies to	Description
«copy»	Dependency	Copy is a stereotyped dependency whose source and target are different instances, but each with the same values, state instance, and roles (but a distinct identity). A copy dependency from A to B means that B is an exact copy of A. Future changes in A are not necessarily reflected in B.
«create»	BehavioralFeature	Create is a stereotyped behavioral feature denoting that the designated feature creates an instance of the classifier to which the feature is attached.
	Event	Create is a stereotyped event denoting that the instance enclosing the state machine to which the event type applies is created. Create may only be applied to an initial transition at the topmost level of this state machine, and in fact, this is the only kind of trigger that may be applied to an initial transition.
«destroy»	BehavioralFeature	Delete is a stereotyped behavioral feature denoting that the designated feature destroys an instance of the classifier to which the feature is attached.
	Event	Delete is a stereotyped event denoting that the instance enclosing the state machine to which the event type applies is destroyed.
«deletion»	Refinement	Deletion is a stereotyped refinement having no clients and no sub-refinements.
«derived»	Dependency	Derived is a stereotyped dependency whose source and target are both elements, usually but not necessarily of the same type. A derived dependency specifies that the source is derived from the target, meaning that the source is not manifest, but rather is implicitly derived from the target.
«document»	Component	Document is a stereotyped component representing a document.
«enumeration»	DataType	Enumeration is a stereotyped data type, whose details specify a domain consisting of a set of identifiers that are the possible values of an instance of the data type.
«executable»	Component	Executable is a stereotyped component denoting a program that may be run on a Node.

Name	Applies to	Description
«extends»	Generalization	Extends is a stereotyped generalization between use cases. It specifies that the contents of the extending use case may be added to the related use case. It not only specifies where the contents should be added (extensionPoint), but also if it only should be added if a specified condition (BooleanExpression). When an instance of the related use case reaches the extension point and the condition is fulfilled, the instance continues according to a sequence that is the result of extending the original sequence with the extending sequence at this point. It is required that the ordering of the parts of the extending use case must be fulfilled if its parts are inserted at different places.
«facade»	Package	Facade is a stereotyped package containing nothing but references to model elements owned by another package. It is used to provide a 'public view' of some of the contents of a package. A Façade does not contain any model elements of its own.
«file»	Component	File is a stereotyped component representing a document containing source code or data.
«framework»	Package	Framework is a stereotyped package consisting mainly of patterns.
«friend»	Dependency	Friend is a stereotyped usage dependency whose source is a model element, such as an operation, class, or package, (or operation) and whose target is a different package model element, such as a class or package (or operation). A friend relationship grants the source access to the target regardless of the declared visibility. It extends the visibility of the supplier so that the client can see into the supplier.
«import»	Dependency	Import is a stereotyped dependency between two packages, denoting that the public contents of the target package are added to the namespace of the source package.

Name	Applies to	Description
«implementation Class»	Class	Implementation class is a stereotyped class that is not a type and that represents the implementation of a class in some programming language. An instance may have zero or one implementation classes. This is in contrast to plain general classes, wherein an instance may statically have multiple classes at one time and may gain or lose classes over time and an object (a subtype of instance) may dynamically have multiple classes.
«inherits»	Generalization	Inherits is a stereotyped generalization denoting that instances of the subtype are not substitutable for instance of the supertype.
«instance»	Dependency	Instance is a stereotyped dependency whose source is an instance and whose target is a classifier. An instance dependency from I to C means that I is an instance of C.
«invariant»	Constraint	Invariant is a stereotyped constraint that must be attached to a set of classifiers or relationships, and denotes that the conditions of the constraint must hold for the classifiers or relationships and their instances.
«library»	Component	Library is a stereotyped component representing a static or dynamic library.
«metaclass»	Dependency	Metaclass is a stereotyped dependency whose source and target are both classifiers and denoting that the target is the metaclass of the source.
	Classifer	Metaclass is a stereotyped classifier denoting that the class is a metaclass of some other class.
«postcondition»	Constraint	Postcondition is a stereotyped constraint that must be attached to an operation, and denotes that the conditions of the constraint must hold after the invocation of the operation.
«powertype»	Classifier	Powertype is a stereotyped classifier denoting that the classifier is a metatype, whose instances are subtypes of another type.
	Dependency	Powertype is a stereotyped dependency whose source is a set of generalizations and whose target is a classifier specifying that the target is the powertype of the source.

Name	Applies to	Description
«precondition»	Constraint	Precondition is a stereotyped constraint that must be attached to an operation, and denotes that the conditions of the constraint must hold for the invocation of the operation.
«private»	Generalization	Private is a stereotyped generalization that specifies private inheritance. It hides the inherited features of a class and renders it non-substitutable for declarations of its ancestors.
«process»	Classifier	Process is a stereotyped classifier that is also an active class, representing a heavy-weight flow of control.
«requirement»	Comment	Requirement is a stereotyped comment that states a responsibility or obligation.
«send»	Dependency	Send is a stereotyped dependency whose source is an operation and whose target is a signal, specifying that the source sends the target signal.
«stereotype»	Classifier	Stereotype is a stereotyped classifier, denoting that the classifier serves as a stereotype. This stereotype permits modelers to model stereotype hierarchies.
«stub»	Package	Stub is a stereotyped package representing a package that is incomplete transferred; specifically, a stub provides the public parts of the package, but nothing more.
«subclass»	Generalization	Subclass is a stereotyped generalization denoting that instances of the subtype are not substitutable for instance of the supertype.
«subtraction»	Refinement	Subtraction is a stereotyped refinement having no clients and no sub-refinements.
«subtype»	Generalization	Subtype is a stereotyped generalization that offers no different properties or behavior than basic generalization. This stereotype exists as the opposite of subclass, so that subtyping versus subclassing can be marked explicitly.

Name	Applies to	Description
«system»	Package	System is a stereotyped package that represents a collection of models of the same modeled system. The models contained in the System all describe the modeled system from different viewpoints, the viewpoints not necessarily disjoint. The System makes up a comprehensive specification of the modeled system, it is the top-most construct in the specification. A System also contains all relationships and constraints between model elements contained in different models. These model elements add no semantic information to the connected model elements, since each model shows a complete view of the modeled system. Thus, these model elements do not express information on the modeled system as such, but rather on the models (e.g.,they may be used for requirements tracking). A modeled system may be realized by a set of subordinate modeled systems, each described by its own set of models collected in a separate System. A System can only be contained in a System.
«table»	Component	Table is a stereotyped component representing a data base table.
«thread»	Classifier	Thread is a stereotyped classifier that is also an active class, representing a light-weight flow of control.
«topLevelPackage»	Package	TopLevelPackage is a stereotyped package denoting the top-most package in a model, representing all the non-environmental parts of the model. A TopLevelPackage is at the top of the containment hierarchy in a model.
«type»	Class	Type is a stereotype of Class, meaning that the class is used for specification of a domain of instances (objects) together with the operations applicable to the objects. A type may not contain any methods, but it may have attributes and associations.

Name	Applies to	Description
«useCaseModel»	Model	UseCaseModel is a model that describes a system's functional requirements in terms of a set of use cases and their interactions with actors. It is required that a UseCaseModel only contains use cases and actors and their relationships: extends and uses between use cases, associations between use cases and actors, and generalizations between actors.
«uses»	Generalization	Uses is a stereotyped generalization between use cases. It specifies that the contents of the related use case is included (or used) in the description of the other use case. It is typically used for extracting shared behavior. It requires that the ordering of the parts of the used use case must be fulfilled if its parts are used at different places. Uses may only be defined between use cases.
«utility»	Classifier	Utility is a stereotyped classifier representing a classifier that has no instances, but rather denotes a named collection of non-member attributes and operations, all of which are class-scoped.

A.2 Tagged Values

The following tagged values are predefined in the UML. Any tagged value that applies to a specific class in the metamodel also applies to any subclasses of that class.

Name	Applies to	Description
documentation	Element	Documentation is a comment, description, or explanation of the element to which it is attached.
location	Classifier	Location denotes that the classifier is a part of the given component.
	Component	Location denotes that the component resides on given node.
persistence	Attribute	Persistence denotes the permanence of the state of the attribute, marking it as transitory (its state is destroyed when the instance is destroyed) or persistent (its state is not destroyed when the instance is destroyed).
	Classifier	Persistence denotes the permanence of the state of the classifier, marking it as transitory (its state is destroyed when the instance is destroyed) or persistent (its state is not destroyed when the instance is destroyed).

Name	Applies to	Description
	Instance	Persistence denotes the permanence of the state of the instanced, marking it as transitory (its state is destroyed when the instance is destroyed) or persistent (its state is not destroyed when the instance is destroyed).
responsibility	Classifier	Responsibility is a contract by or an obligation of the classifier.
semantics	Classifier	Semantics is the specification of the meaning of the classifier.
	Operation	Semantics is the specification of the meaning of the operation.

A.3 Constraints

The following constraints are predefined in the UML.

Name	Applies to	Description
association	LinkEnd	Association is a constraint applied to a link-end, specifying that the corresponding instance is visible via association.
broadcast	Request	Broadcast is a constraint applied to a request sent to multiple instances, specifying that it is sent simultaneously to all target instances, in an undefined unspecified order.
complete	Generalization	Complete is a constraint applied to a set of generalizations, specifying that all subtypes have been specified (although some may be elided) and that additional subtypes are not permitted.
disjoint	Generalization	Disjoint is a constraint applied to a set of generalizations, specifying that instance may have no more than one of the given subtypes as a type of the instance. This is the default semantics of generalization.
global	LinkEnd	Global is a constraint applied to a link-end, specifying that the corresponding instance is visible because it is in a global scope relative to the link.
implicit	Association	Implicit is a constraint applied to an association, specifying that the association is not manifest, but rather is only conceptual.
incomplete	Generalization	Incomplete is a constraint applied to a set of generalizations, specifying that not all subtypes have been specified (even if some are elided) and that additional subtypes are permitted. This is the default semantics of generalizations.

Name	Applies to	Description
local	LinkEnd	Local is a constraint applied to a link-end, specifying that the corresponding instance is visible because it is in a local scope relative to the link.
or	Association	Or is a constraint applied to a set of associations, specifying that over that set, only one is manifest for each associated instance. Or is an exclusive (not inclusive) constraint.
overlapping	Generalization	Overlapping is a constraint applied to a set of generalizations, specifying that instances may have more than one of the given subtypes as a type of the instance.
parameter	LinkEnd	Parameter is a constraint applied to a link-end, specifying that the corresponding instance is visible because it is a parameter relative to the link.
self	LinkEnd	Self is a constraint applied to a link-end, specifying that the corresponding instance is visible because it is the dispatcher of a request.
vote	Request	Vote is a constraint applied to a request, specifying that the return value is selected by a majority vote of all the return values returned from multiple instances.

A