Meta-Metamodel (M3)

In this document we describe the Meta-Metamodel to be used by LionWeb **version 2024.1**. The Meta-Metamodel is called LionCore.

Table of Contents

1. Introduction . . . 2
1.1. Goals . . . 2
1.2. Languages supported . . . 2
1.3. What kind of models should be expressible? . . . 2
2. Overview . . . 3
3. Definition of the meta-metamodel . . . 4
3.1. Concepts . . . 4
3.1.1. Language . . . 4
3.1.2. Concept . . . 5
3.1.3. Annotation . . . 5
3.1.4. Interface . . . 7
3.1.5. PrimitiveType . . . 8
3.1.6. Enumeration . . . 8
3.1.7. EnumerationLiteral . . . 9
3.1.8. StructuredDataType . . . 9
3.1.9. Field . . . 10
3.1.10. Containment . . . 10
3.1.11. Reference . . . 11
3.1.12. Property . . . 12
3.2. Abstract concepts . . . 12
3.2.1. LanguageEntity . . . 12
3.2.2. Classifier . . . 13
3.2.3. DataType . . . 14
3.2.4. Feature . . . 14
3.2.5. Link . . . 15
3.3. Interfaces . . . 16
3.3.1. IKeyed . . . 16
3.4. Pre-defined keys and resolveInfos . . . 17
3.4.1. Keys and resolveInfos of M3 elements . . . 17
3.4.2. Keys and resolveInfos of built-in elements . . . 19
3.5. Built-in elements . . . 19
3.5.1. Concepts . . . 20
3.5.2. Interfaces . . . 20

3.5.3. Primitive types . . . 20
3.6. Supporting terminology . . . 20
3.6.1. Multiplicity . . . 20
3.6.2. Partitions . . . 21
3.6.3. Identifiers . . . 21
3.6.4. Keys . . . 22
3.6.5. Namespaces . . . 22
4. Other considerations . . . 22
4.1. Reflection . . . 22
4.2. Generics . . . 22
4.3. References to language elements . . . 22
4.4. Union or intersection types . . . 23
4.5. Operations . . . 23
5. Reference models . . . 23
5.1. Meta-meta model . . . 23
5.2. Pre-defined elements . . . 75
6. Comparison with other meta-metamodels . . . 82
6.1. Comparison with Ecore . . . 83
6.2. Comparison with MPS . . . 83

1. Introduction

1.1. Goals

The goal is to define a meta-metamodel that can be used in different contexts and implemented from different languages.

The approach taken would be conservative: we want to provide boring and proven infrastructure, so that innovation can be built on top of it.

This will be based on the experience that we as a community had with the meta-metamodel used in EMF and MPS mainly. Suggestions based on the experience obtained with other meta-metamodels are also very welcome.

1.2. Languages supported

We aim to have the initial implementations being available in Java and Typescript. We are interested in implementing it in other languages too, but not as part of the initial effort.

1.3. What kind of models should be expressible?

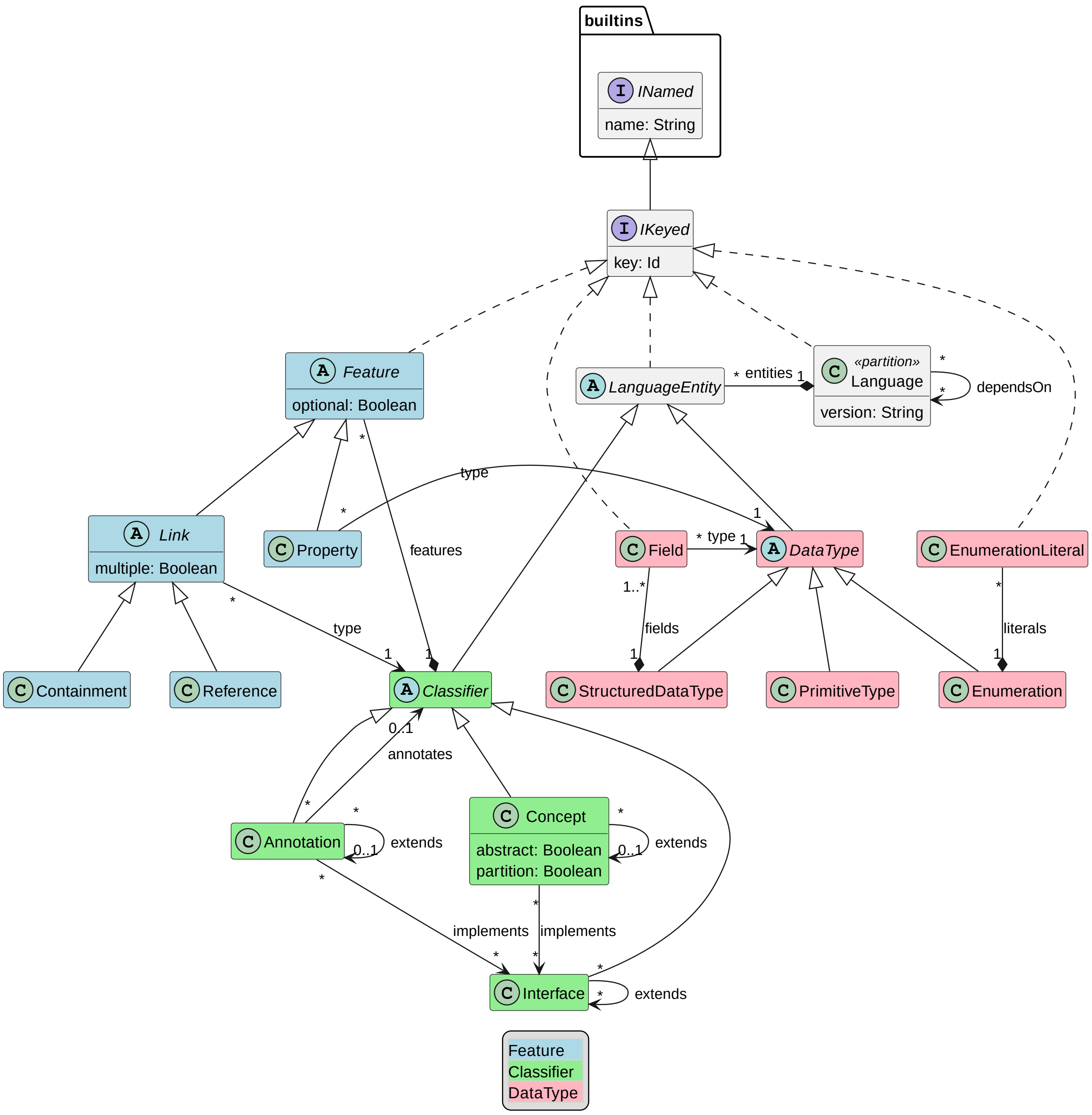
Any kind of model. In other words, models specified using metamodels expressed through this Meta-Metamodel should not make any assumptions on the node being obtained from parsing text or a graphical modeling tool or a projectional editor. All these possible origins should be supported, with specific extensions were necessary.

For example, metamodels should be expressed to define:

- The Abstract Syntax Tree (AST) of existing textual programming languages such as Java, RPG, Python, or SAS
- The Abstract Syntax Tree (AST) of DSLs backed by projectional editors. For example, DSLs used to define tax calculations or automation rules for email marketing systems

2. Overview

Let’s first see a representation of the meta-metamodel using (a tiny subset of) the UML’s class diagram notation.



3. Definition of the meta-metamodel

In this section we describe the single elements composing the Meta-Metamodel. We will list elements by their type: classes, abstract classes and interfaces.

3.1. Concepts

The concepts are Language, Concept, Annotation, Interface, PrimitiveType, Enumeration, EnumerationLiteral, StructuredDataType, Field, Containment, Reference, and Property.

3.1.1. Language

A Language[1] will provide the Concepts necessary to describe ideas in a particular domain together with supporting elements necessary for the definition of those Concepts.

It also represents the namespace for LanguageEntities.

Example

For example, a Language for accounting could collect several Concepts such as *Invoice*, *Customer*, *InvoiceLine*, *Product*. It could also contain related elements necessary for the definitions of the concepts. For example, a DataType named Currency.

EMF & MPS equivalent

A Language in LionWeb will be roughly equivalent to an `EPackage` or the contents of the *structure aspect* of an MPS Language.

A Language will not have a URI or a prefix, differently from EPackages.

A Language will have a version string, different from MPS Languages' version number.

Differently from EPackages and MPS Languages, there is no way to group language elements. EPackages have instead sub-packages and MPS Languages have virtual folders. For this use case, different Languages could be used instead.

Characteristics

A Language has a `name`, a `key`[2], and a `version`[3][4], similar to MPS Languages.

Each Language will contain a list of Language entities in its `entities` containment.

A Language is an INamed (as it has a name) and an IKeyed (as it has a key)[5].

A Language can depend on other Languages via `dependsOn` reference.[6] Dependencies must be explicitly declared.[7]

Constraints

`version` can be any non-empty string.[8][9]

Language elements contained in a Language are allowed to refer to these other Language elements:[7]

- within the same language
- within declared dependencies
- within *transitive* dependencies.

  Example: if Language *B* depends on Language *A*, and Language *C* depends on *B*, then Language *C* can also refer to members of Language *A* without explicitly declaring a dependency.

A Language CAN declare a transitive dependency explicitly. In the example above, Language *C* CAN declare an explicit dependency on Language *A*.

As any Language depends (at least transitively and implicitly) on built-ins elements, a Language CAN declare a dependency on builtins — but *does not* need to.[10]

3.1.2. Concept

A Concept represents a category of entities sharing the same structure.

Example

For example, *Invoice* would be a Concept. Single entities could be Concept instances, such as Invoice #1/2022.

EMF & MPS equivalent

A Concept is roughly equivalent to an `EClass` (with the `isInterface` flag set to `false`) or an MPS's `ConceptDeclaration`.

Characteristics

A Concept has a `name` and an `key`.

A Concept can be concrete (i.e., instantiable) or abstract, marked by boolean `abstract` property.

A Concept can be marked as partition via the boolean `partition` property.[11]

A Concept is a Classifier (as it has features). It is indirectly a LanguageEntity (as it is a top level element in a Language), an INamed (as it has a name), a IKeyed (as it has a key).

Each Concept `extends` zero or one Concepts. If no Concepts are explicitly extended, the Concept will implicitly extend the Concept Node. `Node` is the only concept that truly does not extend any Concept.

A Concept `implements` zero or more Interfaces.

A Concept can have any number of `features`, given it is a Classifier.

Constraints

A Concept MUST NOT extend itself, or form circles via `extends`.

3.1.3. Annotation

An Annotation is an additional piece of information attached to potentially any node, sharing the node's lifecycle. The annotated node (or its concept) does not need to know about the annotation.[12] Annotations CAN be attached to other Annotations (although this structure is hard to comprehend, and should be used with caution).

Annotations should only have limited content because the more complex the annotation is, the more likely it should not be part of the annotated node's lifecycle — We might want to reuse the complex annotation somewhere else.
Example: In MPS, the editor of a concept can be seen as an annotation of that concept. But even if we deleted the concept, we might want to reuse the editor for another (similar) concept.

Annotations' contents should be orthogonal to the annotated node, because actual content should be part of the original concepts, and unrelated contents should be somewhere else.
Example: Let's assume our Language defines a `Date` PrimitiveType. We might annotate `Date` with `@JavaImplementation(java.util.Date)` and `@TypeScriptImplementation(JsJoda.LocalDate)`. Our core model stays platform-independent, but we maintain the implementation details at the right place — if we ever deleted `Date`, all related information would be gone without further cleanup.

Example

We design a fancy documentation language, and can annotate any other node with such documentation.

EMF & MPS equivalent

An Annotation is equivalent to MPS' `NodeAttribute`. In EMF, an `Adapter` can be used for similar purposes on M1 models. On M2 models (i.e. Ecore), an Annotation corresponds to an `EAnnotation`.

Characteristics

An Annotation has a `name` and an `key`.

An Annotation is a Classifier, and indirectly a LanguageEntity (as it is a top level element in a Language), an INamed (as it has a name), a IKeyed (as it has a key).

Each Annotation specifies which `Classifiers` it `annotates`. If it should be applicable to any node, it annotates Node.

Annotations can specify which Annotation it `extends` and which interfaces it `implements`. An Annotation CAN have any number of `features`, given it is a Classifier.

Constraints

We CANNOT redefine `annotates` in a sub-annotation (i.e. an annotation that `extends` another).[13]

We MUST attach an instance of an Annotation only to instances of the Concept the Annotation `annotates` (and sub-concepts thereof). We can attach be zero, one, or more instances to a single annotated node.[14]

We MUST NOT use instances of Concepts as annotations to any other node.[15]

Annotation examples

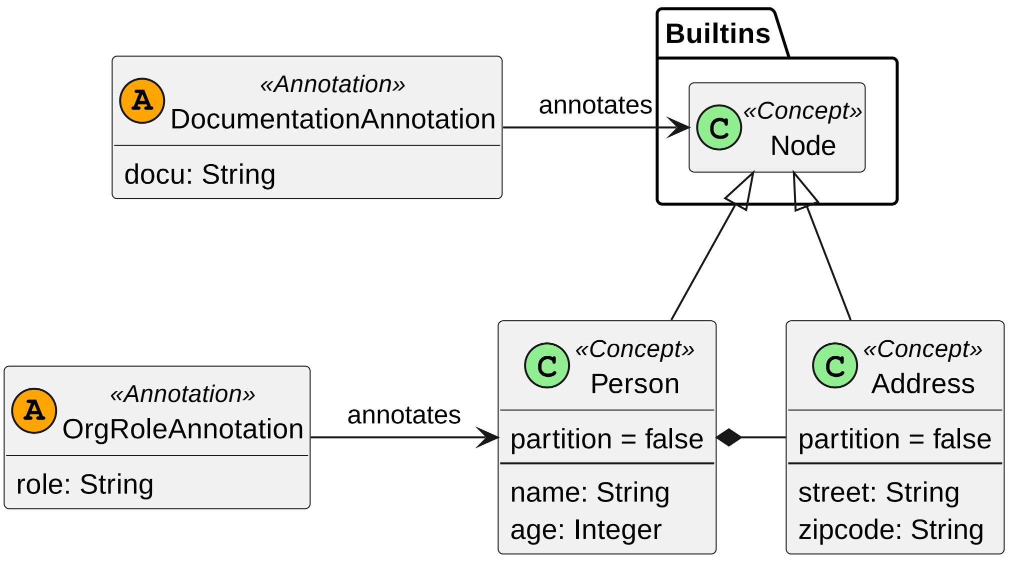


Figure 1. Language

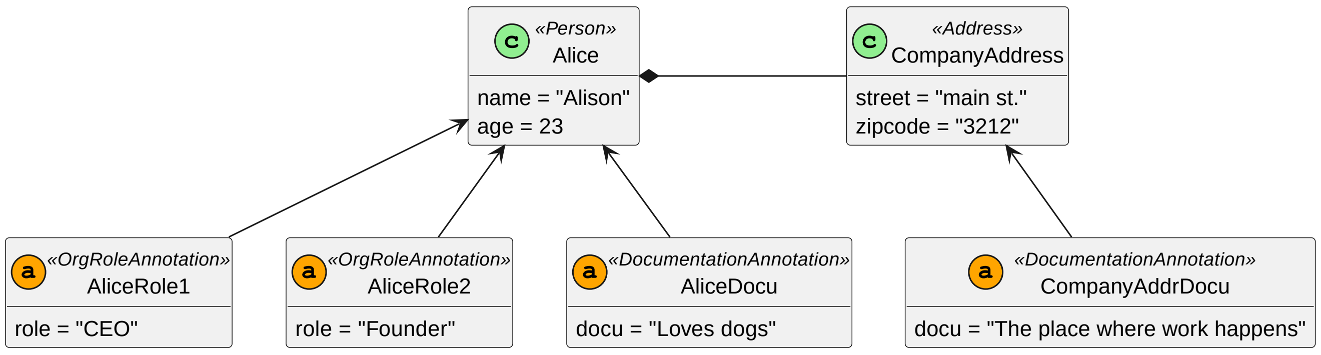


Figure 2. Valid instance

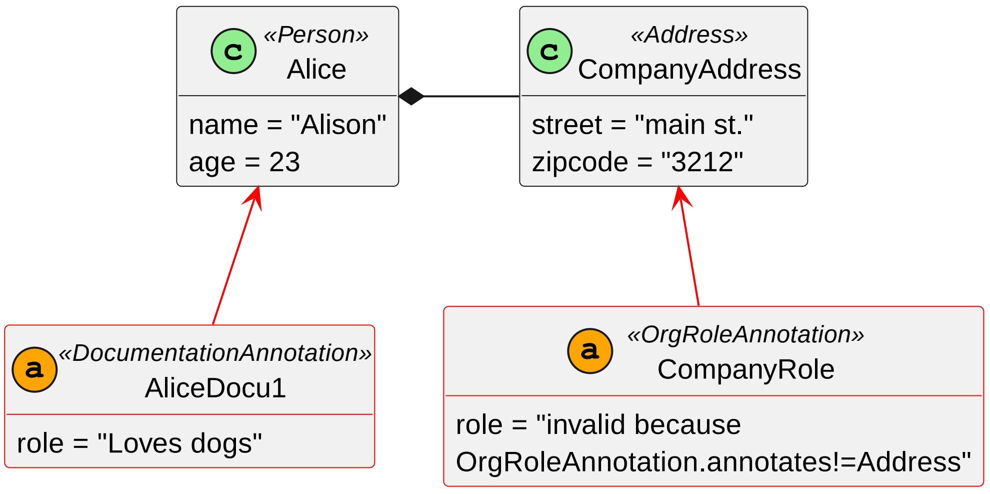


Figure 3. Invalid instance

3.1.4. Interface

An Interface[16] represents a category of entities sharing some similar characteristics.

Example

For example, `Named` would be an Interface.

EMF & MPS equivalent

An Interface in LionWeb will be roughly equivalent to an `EClass` (with the `isInterface` flag set to `true`) or an MPS's `ConceptInterfaceDeclaration`.

Characteristics

An Interface has a `name` and an `key`.

An Interface is an Classifier (as it has features). It is indirectly a LanguageEntity (as it is a top level element in a Language), an INamed (as it has a name), a IKeyed (as it has a key).

Each Interface `extends` zero or more Interfaces.

An Interface can have any number of `features`, given it is a Classifier.

Constraints

3.1.5. PrimitiveType

This represents an arbitrary primitive value, which is not an Enumeration.

Example

BooleanType, *NumberType*, and *StringType* are common PrimitiveTypes.

EMF & MPS equivalent

A PrimitiveType is similar to Ecore's `EDataType` and to MPS' `PrimitiveDataTypeDeclaration`.

Differently from ECore's `EDataType` PrimitiveType has no flag `serializable`, and it does not inherit fields such as `instanceClassName`, `instanceClass`, or `defaultValue`.

Characteristics

A PrimitiveType has a `name` and an `key`.

A PrimitiveType is a DataType (as it can be used as type of a Property). It is indirectly a LanguageEntity (as it is a top level element in a Language), an INamed (as it has a name) and a IKeyed (as it has a key).

The correspondence between a PrimitiveType and an implementation class on a specific platforms can be specified through annotations, but it is not specified on the PrimitiveType itself.

Constraints

We consider a PrimitiveType as opaque value type without structure or identity.

3.1.6. Enumeration

A primitive value with finite, pre-defined, known set of possible values.

Example

DaysOfWeek or *PlayingCardSuit* are common Enumerations.

EMF & MPS equivalent

An Enumeration is similar to Ecore's `EEnum` and to MPS' `EnumerationDeclaration`.

Differently from ECore's `EEnum` Enumeration has no flag `serializable`, and it does not inherit fields such as `instanceClassName`, `instanceClass`, or `defaultValue`.

Characteristics

An Enumeration has a `name` and an `key`.

An Enumeration contains EnumerationLiterals in its `literals` containment. It also represents the namespace for the EnumerationLiterals.

An Enumeration is a DataType (as it can be used as type of a Property). It is indirectly a LanguageEntity (as it is a top level element in a Language), an INamed (as it has a name) and a IKeyed (as it has a key).

Constraints

Implementations should use native representations for Enumerations, e.g. a `enum` in a Java implementation.

3.1.7. EnumerationLiteral

One of the possible values of an Enumeration.

Example

Monday, *Tuesday*, *Wednesday*, *Thursday*, *Friday*, *Saturday* and *Sunday* are all EnumerationLiterals of the *DaysOfWeek* Enumeration.

EMF & MPS equivalent

An EnumerationLiteral is similar to Ecore's `EEnumLiteral` and to MPS' `EnumerationMemberDeclaration`.

Characteristics

An EnumerationLiteral has a `name` and an `key`.

An EnumerationLiteral is a IKeyed (as it has a key) and indirectly an INamed (as it has a name).

Constraints

Each EnumerationLiteral must have a unique `name` within the Enumeration.

Each EnumerationLiteral belongs to one and only one Enumeration.

3.1.8. StructuredDataType

Represents a collection of named instances of Data Types. They are meant to support a small composite of values that semantically form a unit. Instances of StructuredDataTypes have no identity, are always copied by value, and SHOULD be immutable.[17] Two instances of a StructuredDataType that hold the same values for all fields of that StructuredDataType are interchangeable. (Instances of Classifiers have an identity, through `id`.)

Example

We could represent a "Complex number { real part, imaginary part }" or "Color { red, green, blue }" as StructuredDataType.

EMF & MPS equivalent

No direct similarities. We could simulate it by a custom `EDatatype` in ECore.

Characteristics

A StructuredDataType contains Fields in its `fields` containment. It also represents the namespace for the fields.

A StructuredDataType has a `name` and an `key`.

A StructuredDataType is a DataType (as it can be used as type of a Property). It is indirectly a LanguageEntity (as it is a top level element in a Language), an INamed (as it has a name) and a IKeyed (as it has a key).

Constraints

M1 Instances of a StructuredDataType SHOULD be immutable.

An M1 instance of StructuredDataType MUST have non-null values for all its fields.

A StructuredDataType MUST NOT (directly or indirectly) contain a field with itself as `type`.[17]

3.1.9. Field

Represents one part of a StructuredDataType.[17]

Example

In the StructuredDataType "Complex number", we would have two fields "real part" and "imaginary part". A StructuredDataType "Color" had the fields "red", "green", and "blue".

EMF & MPS equivalent

No direct similarities.

Characteristics

A Field has a `name` and an `key`.

A Field refers its `type`, which is a DataType.

A Field is an IKeyed (as it has a key) and indirectly an INamed (as it has a name).

Constraints

Each Field MUST have a unique `name` within its containing StructuredDataType.

Each Field belongs to one and only one StructuredDataType.

3.1.10. Containment

Represents a relation between a containing Classifier and a contained Classifier.

Example

Between an *IfStatement* and its *condition* there is a Containment relation.

EMF & MPS equivalent

A Containment is similar to an ECore's `EReference` with the `containment` flag set to `true`. Differently from an `EReference` there is no `container` flag and `resolveProxies` flag.

A Containment is similar to an MPS's `LinkDeclaration` with `metaClass` having value `aggregation`. Differently from a `LinkDeclaration` there is no field `unordered`.

Characteristics

A Containment has a `name` and an `key`. It can be marked as `optional` and `multiple`.

A Containment refers its `type`, which is a Classifier.

A Containment is a Link (as it describes a relation between two Classifiers). It is indirectly a Feature (as it describes the characteristics of a Classifier), an INamed (as it has a name) and a IKeyed (as it has a key).

Constraints

Each node, except Partitions, must be contained once in exactly one Containment.

The implicit inverse relation of a Containment is called *parent*. Example: An IfStatement *contains* its condition. Then, the condition's *parent* is the IfStatement.

Instances in containments MUST NOT form a direct or indirect circle: If node *a* contains node *b*, node *b* MUST NOT contain *a* (direct containment circle). If the same node *b* contains node *c*, *c* MUST NOT contain *a* (indirect containment circle). However, Concept *A* can have a Containment of type *A*. Concept *A* with Containment of type *B*, with Concept *B* having a Containment of type *A*, is also valid.

We MUST NOT use Annotations as type of a Containment. We MUST put instances of Concepts into a containment on M1 level; we MUST NOT use instances of Annotations.[15]

3.1.11. Reference

Represents a relation between a referring Classifier and referred Classifier.

Example

VariableReference may have a Reference to a *VariableDeclaration*.

EMF & MPS equivalent

A Reference is similar to an ECore's `EReference` with the `containment` flag set to `false`. Differently from an `EReference` there is no `container` flag and `resolveProxies` flag.

A Reference is similar to an MPS's `LinkDeclaration` with `metaClass` having value `reference`. Differently from a `LinkDeclaration` there is no field `unordered`.

Characteristics

A Reference has a `name` and an `key`. It can be marked as `optional` and `multiple`.

A Containment refers its `type`, which is a Classifier.

A Reference is a Link (as it describes a relation between two Classifiers). It is indirectly a Feature (as it describes the characteristics of a Classifier), an INamed (as it has a name) and a IKeyed (as it has a key).

Constraints

A multiple Reference MAY refer to the same target more than once. Example: We have nodes *p1* and *p2* of Concept Procedure. Concept StackTrace has a multiple reference "frames" of type Procedure. An instance of StackTrace can have frames refer to *[p1, p2, p1]*.

3.1.12. Property

This indicates a simple value associated to an entity.

Example

For example, an *Invoice* could have a *date* or an *amount*.

EMF & MPS equivalent

A Property is similar to Ecore's `EAttribute`.

A Property is similar to MPS's `AttributeDeclaration`.

Characteristics

A Property has a `name` and an `key`. It can be marked as `optional`.

A Property refers its `type`, which is a DataType.

A Property is a Feature (as it describes the characteristics of a Classifier). It is indirectly a IKeyed (as it has a key) and an INamed (as it has a name).

Constraints

Properties are always single-valued.[18].

3.2. Abstract concepts

The abstract concepts are LanguageEntity, Classifier, DataType, Feature, and Link.

3.2.1. LanguageEntity

A LanguageEntity is an entity with an identity directly contained in a Language.

Example

For example, *Invoice*, *Currency*, *Named*, or *String* could be LanguageEntities.

EMF & MPS equivalent

LanguageEntity is similar to Ecore's `EClassifier`.

LanguageEntity is similar to MPS' `IStructureElement`. The difference is that `IStructureElement` includes also elements that cannot appear as top level elements of a structure aspects, such as `LinkDeclaration`, `PropertyDeclaration`, and `EnumerationMemberDeclaration`.

Characteristics

A LanguageEntity has a `name` and an `key`.

A LanguageEntity is a IKeyed (as it has a key), and indirectly an INamed (as it has a name).

A LanguageEntity can be one of:

- Annotation
- Concept
- Interface
- Enumeration
- PrimitiveType
- StructuredDataType

Constraints

Each LanguageEntity must have a unique `name` within the Language.

Each LanguageEntity belongs to one and only one Language.

3.2.2. Classifier

Something which can own Features.[19]

Example

A Concept can have several features.

EMF & MPS equivalent

Classifier is similar to `EClass` in Ecore (which is used both for classes and interfaces) and to `AbstractConceptDeclaration` in MPS.

Characteristics

A Classifier has a `name` and an `key`.

It also represents the namespace for Features.

A Classifier owns any number of Features in `features` containment.

A Classifier can be one of:

- Annotation
- Concept
- Interface

A Classifier is a LanguageEntity (as it is a top level element in a Language). It is indirectly a IKeyed (as it has a key) and an INamed (as it has a name).

3.2.3. DataType

A type of value which has no relevant identity in the context of a model.

Example

A *Currency* or a *Date* type.

EMF & MPS equivalent

It is similar to Ecore’s `EDataType`.

It is similar to MPS’ `DataTypeDeclaration`.

Characteristics

A DataType has a `name` and an `key`.

A DataType is a LanguageEntity (as it is a top level element in a Language). It is indirectly a IKeyed (as it has a key) and an INamed (as it has a name).

A DataType can be one of:

- PrimitiveType
- Enumeration
- StructuredDataType

Constraints

We consider a DataType as value type without identity. Implementations can choose to copy, compare, or process DataType values by value.

3.2.4. Feature

A Feature represents a characteristic or some form of data associated with a particular Classifier.

Example

For example, an *Invoice* can have an associated *date*, a *number*, a connection with a *customer*, and it can contain *InvoiceLines*. All of this information is represented by features.

EMF & MPS equivalent

A Feature in LionWeb will be roughly equivalent to an `EStructuralFeature` or to the combination of `Properties` and `Links` (both containment and reference links) in MPS.

Differently from Ecore’s `EStructureFeature`, Features do not have flags such as `changeable`, `volatile`, `transient`, or `unsettable`. They have no `default value`.

Different from MPS' `Link`, Features CANNOT be specialized.[20]

Characteristics

A Feature has a `name` and an `key`.

A Feature can be set to `optional` or required.

A Feature is a IKeyed (as it has a key) and indirectly an INamed (as it has a name).

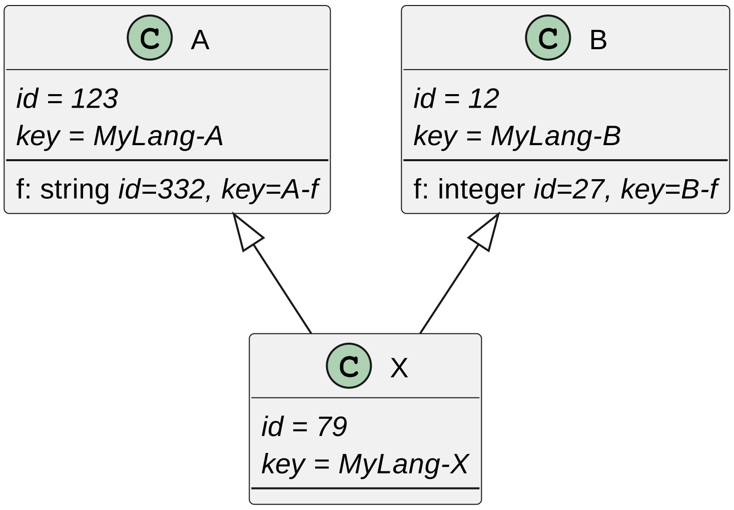
A Feature can either be one of:

- Property
- Containment
- Reference

Constraints

Each Feature MUST have a unique name within a specific Classifier, including all (directly or indirectly) inherited Features.[21]

We CAN have non-unique *inherited* feature *names*, as in this example:



However, within LionWeb we always use a Feature's *id* or *key* to for identification. Thus, LionWeb can deal with the name clash.

If a host language cannot handle this situation, the generator towards that language needs to resolve it.[22]

3.2.5. Link

Represent a connection to a Classifier.

Example

An *Invoice* can be connected to its *InvoiceLines* and to a *Customer*.

EMF & MPS equivalent

It is similar to Ecore's `EReference`.

It is similar to MPS' `LinkDeclaration`.

Characteristics

A Link has a `name` and an `key`. It can be marked as `optional`.

A Link can have `multiple` or only a single targets.

A Link refers its `type`, which is a Classifier.

A Link is a Feature (as it describes the characteristics of a Classifier). It is indirectly a IKeyed (as it has a key) and an INamed (as it has a name).

A Link can be either a Containment, or a Reference.

Constraints

We do NOT support link specialization.[20]

3.3. Interfaces

The interfaces are IKeyed.

3.3.1. IKeyed

Something with a name that has a key.[5][23]

Example

A Concept *Invoice*, contained in a Language `com.foo.Accounting`.

EMF & MPS equivalent

n/a

Characteristics

An IKeyed has a `name` and an `key`.[2]

A IKeyed is an INamed (as it has a name).

All elements of the Meta-Metamodel realize `IKeyed`.

Constraints

A IKeyed’s `name` MUST be unique within the namespace (i.e. Language, Classifier, or Enumeration). The name MUST be a valid programming language identifier[24].

More specifically, we allow Java identifiers with the following modifications:

- We do NOT allow `$` (dollar sign)
- *ReservedKeyword*, *BooleanLiteral*, and *NullLiteral* (as per Java identifier spec) are allowed identifiers.

Effectively:

- Names MUST NOT start with a number.
- Names MUST NOT be empty.

- Names MUST NOT contain spaces.
- Names CAN use Unicode characters, numbers, and underscore.

NOTE These restrictions only apply to names of Language elements (i.e. M3 concepts/M2 instances). Any language that uses `INamed` on its own can establish their own constraints.

Refer to Keys for more constraints.

3.4. Pre-defined keys and resolveInfos

We can serialize languages as regular nodes. To keep the serialized form of the language compatible with multiple LionWeb versions. we SHOULD serialize references to M3 elements and built-ins *without* target reference id, and include the resolveInfo as listed in this section.[25]

3.4.1. Keys and resolveInfos of M3 elements

The language itself has key `LionCore-M3`[26] and is named `LionCore_M3`[27].

M3 element	Concept	Key	ResolveInfo
Annotation	Concept	`Annotation`	`LionWeb.LionCore_M3.Annotation`
Annotation.annotates	Reference	`Annotation-annotates`	`LionWeb.LionCore_M3.Annotation.annotates`
Annotation.extends	Reference	`Annotation-extends`	`LionWeb.LionCore_M3.Annotation.extends`
Annotation.implements	Reference	`Annotation-implements`	`LionWeb.LionCore_M3.Annotation.implements`
Classifier	Concept	`Classifier`	`LionWeb.LionCore_M3.Classifier`
Classifier.features	Containment	`Classifier-features`	`LionWeb.LionCore_M3.Classifier.features`
Concept	Concept	`Concept`	`LionWeb.LionCore_M3.Concept`
Concept.abstract	Property	`Concept-abstract`	`LionWeb.LionCore_M3.Concept.abstract`
Concept.extends	Reference	`Concept-extends`	`LionWeb.LionCore_M3.Concept.extends`
Concept.implements	Reference	`Concept-implements`	`LionWeb.LionCore_M3.Concept.implements`
Containment	Concept	`Containment`	`LionWeb.LionCore_M3.Containment`
DataType	Concept	`DataType`	`LionWeb.LionCore_M3.DataType`
Enumeration	Concept	`Enumeration`	`LionWeb.LionCore_M3.Enumeration`

M3 element	Concept	Key	ResolveInfo
Enumeration.literals	Containment	`Enumeration-literals`	`LionWeb.LionCore_M3.Enumeration.literals`
EnumerationLiteral	Concept	`EnumerationLiteral`	`LionWeb.LionCore_M3.EnumerationLiteral`
Feature	Concept	`Feature`	`LionWeb.LionCore_M3.Feature`
Feature.optional	Property	`Feature-optional`	`LionWeb.LionCore_M3.Feature.optional`
Field	Concept	`Field`	`LionWeb.LionCore_M3.Field`
Field.type	Reference	`Field-type`	`LionWeb.LionCore_M3.Field.type`
IKeyed	Interface	`IKeyed`	`LionWeb.LionCore_M3.IKeyed`
IKeyed.key	Property	`IKeyed-key`	`LionWeb.LionCore_M3.IKeyed.key`
Interface	Concept	`Interface`	`LionWeb.LionCore_M3.Interface`
Interface.extends	Reference	`Interface-extends`	`LionWeb.LionCore_M3.Interface.extends`
Language	Concept	`Language`	`LionWeb.LionCore_M3.Language`
Language.dependsOn	Reference	`Language-dependsOn`	`LionWeb.LionCore_M3.Language.dependsOn`
Language.entities	Containment	`Language-entities`	`LionWeb.LionCore_M3.Language.entities`
Language.version	Property	`Language-version`	`LionWeb.LionCore_M3.Language.version`
LanguageEntity	Concept	`LanguageEntity`	`LionWeb.LionCore_M3.LanguageEntity`
Link	Concept	`Link`	`LionWeb.LionCore_M3.Link`
Link.multiple	Property	`Link-multiple`	`LionWeb.LionCore_M3.Link.multiple`
Link.type	Reference	`Link-type`	`LionWeb.LionCore_M3.Link.type`
PrimitiveType	Concept	`PrimitiveType`	`LionWeb.LionCore_M3.PrimitiveType`
Property	Concept	`Property`	`LionWeb.LionCore_M3.Property`
Property.type	Reference	`Property-type`	`LionWeb.LionCore_M3.Property.type`
Reference	Concept	`Reference`	`LionWeb.LionCore_M3.Reference`

M3 element	Concept	Key	ResolveInfo
StructuredDataType	Concept	StructuredDataType	LionWeb.LionCore_M3.StructuredDataType
StructuredDataType.fields	Containment	StructuredDataType-fields	LionWeb.LionCore_M3.StructuredDataType.fields

3.4.2. Keys and resolveInfos of built-in elements

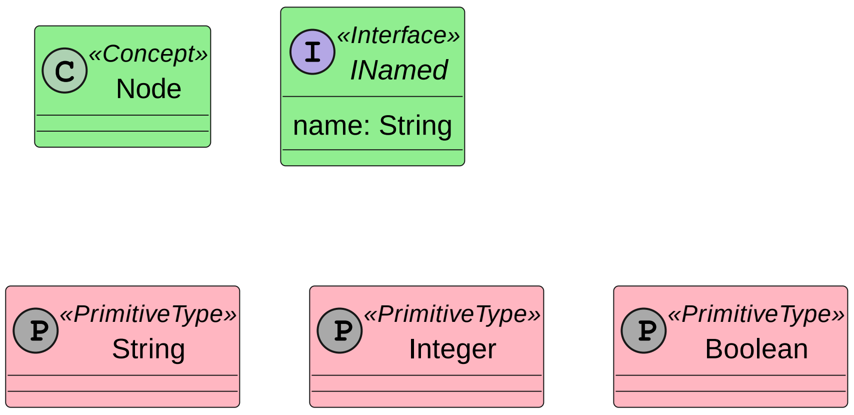
The language hosting built-in elements has id and key LionCore-builtins[28], its name is LionCore_builtins[27].

Every language implicitly depends on this language.[29] Thus, the ids in this language MUST be stable. This means the id MUST be identical to the key for each node in this language.

Instance	Concept	Key	ResolveInfo
Boolean	PrimitiveType	LionCore-builtins-Boolean	LionWeb.LionCore_builtins.Boolean
INamed	Interface	LionCore-builtins-INamed	LionWeb.LionCore_builtins.INamed
INamed.name	Property	LionCore-builtins-INamed-name	LionWeb.LionCore_builtins.INamed.name
Integer	PrimitiveType	LionCore-builtins-Integer	LionWeb.LionCore_builtins.Integer
Node	Concept	LionCore-builtins-Node	LionWeb.LionCore_builtins.Node
String	PrimitiveType	LionCore-builtins-String	LionWeb.LionCore_builtins.String

3.5. Built-in elements

Each LionWeb implementation ships with a set of built-in elements, akin to a *standard library*.[27] These elements are part of the M2 language LionCore_builtins. It can be used in any user-defined Language.[30]



3.5.1. Concepts

- Node[31], an *abstract* Concept that's the (explicit or implicit) the ancestor of all concepts.

3.5.2. Interfaces

- INamed[32], an Interface with one Property called name of type String.

  If a Reference.type targets a Classifier that implements INamed, implementations SHOULD use the target's name property as default resolveInfo value.

3.5.3. Primitive types

Some primitive types will be widely used, so it makes sense to pre-define them.[33]

- Boolean with exactly two values: true and false (Details)
- String, an arbitrary-length series of Unicode characters (Details)
- Integer, an arbitrary-length positive or negative integer number (Details)

3.6. Supporting terminology

3.6.1. Multiplicity

Multiplicity describes how many targets a Link must and can have.

Example

Common multiplicities are 1 (meaning there MUST be exactly one target), 0..1 (meaning there CAN be exactly one target), 0..* (meaning there CAN be zero or more targets), and 1..* (meaning there MUST be at least one target, but there CAN be more than one targets).

EMF & MPS equivalent

In Ecore there is no equivalent as lowerBound and upperBound can be set independently.

This is equivalent to MPS' Cardinality, which has the four values mentioned as example.

Characteristics

LionCore represents multiplicity as the two booleans optional (whether there MUST be at least one target) and multiple (whether there CAN be more than one target).

Multiplicity	optional	multiple
1	true	false
0..1	false	false
0..*	true	true
1..*	false	true

3.6.2. Partitions

Each node that does not have a parent node MUST be of a Concept with partition flag set to `true`. This implies that every node is contained in exactly one partition, namely the partition defined by its root node.[11] Partitions CANNOT be nested.

EMF & MPS equivalent

A partition is similar to ECore’s `Resource`.

A partition is similar to MPS' `model`.

3.6.3. Identifiers

Valid characters

Ids can only contain these symbols:

- lowercase latin characters: `a`..`z`
- uppercase latin characters: `A`..`Z`
- arabic numerals: `0`..`9`
- underscore: `_`
- hyphen: `-`

This is the same character set as Base64url variant.

Representation

Ids are represented by a string, containing only valid characters (as defined above). An id string is NOT padded, also not by whitespaces. An id string does NOT contain any terminating symbols (compared to some BASE64 variants); this does not affect internal representation in a specific implementation language, e.g. C-style \0-terminated strings.

Scope

Node ids MUST be unique within their id-space.

Id-space

An id-space is a realm that guarantees the uniqueness of all ids within. Typically, this means one repository.

An id-space has an id as defined above. Uniqueness of id-space ids is out of scope of LionWeb specification.

In LionWeb (the protocol), id-spaces are NOT hierarchical. An implementation might choose to use hierarchical id-spaces internally.

Identification

A node can be identified relative to its id-space by the node's id. To globally identify a node, we use the combination of the id-space id and the node id.

3.6.4. Keys

We use keys when we refer from *instance level* to *meta level*.[34] Refer to References to language elements for a list of all usages.

Keys are modeled via IKeyed.key. Keys MUST be valid Identifiers.

A key SHOULD be globally unique, and MUST be unique within an Id-space, i.e. the Language.[26] For approximate global uniqueness, we SHOULD adopt Java's package naming scheme, based on domain names. As we don't allow dots (`.`) in ids, we SHOULD use dashes (`-`) instead.

3.6.5. Namespaces

A Namespace implements INamed and can have descendants that implement INamed.

Typically, a namespace enforces some constraints on the contained names, like uniqueness within the same namespace, or what's considered a valid name.

We can calculate a fully qualified name by concatenating the namespaces of all the ancestors up to the top level ancestor. Future versions might support this directly.[35]

4. Other considerations

4.1. Reflection

Reflection describes the ability of each Meta-Metamodel instance to access the definition of the Meta-Metamodel element from which it has been instantiated.

It is important to offer this functionality also in consideration that some implementation languages may not offer reflection capabilities that could be used as an alternative.

4.2. Generics

Generics are not directly supported by this proposal. We could solve some needs through specialization of features in derived classes. We could alternatively also imagine using specific annotations for supporting this.

In general Generics complicate the solution and MPS can live without them. Also, in StarLasu we never encountered the need for them so far.

4.3. References to language elements

From a language, we refer to all language elements by their id.[9] This includes references to

- other languages
- extended Concepts
- extended or implemented Interfaces
- Types of Properties or Links

As built-ins is just another language, we refer to its members by their id, just as any other language members.

From an instance, we refer to its defining language element by the language element's Keys[34] This includes references to

- Language usage
- Annotation or Concept instance
- Property assignment
- Containment assignment
- Reference assignment
- EnumerationLiteral value

4.4. Union or intersection types

These are not supported.

4.5. Operations

Operations are not represented in the Meta-Metamodel.

5. Reference models

5.1. Meta-meta model

The LionCore model, aka LionWeb M3. It is defined by means of itself, as outlined by Meta-Object Facility.

```
{
  "serializationFormatVersion": "2024.1",
  "languages": [
    {
      "key": "LionCore-M3",
      "version": "2024.1"
    },
    {
      "key": "LionCore-builtins",
      "version": "2024.1"
    }
  ],
  "nodes": [
    {
      "id": "-id-LionCore-M3",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Language"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "LionCore_M3"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Language-version"
          },
          "value": "2024.1"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "LionCore-M3"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Language-entities"
          },
          "children": [
            "-id-Annotation",
            "-id-Concept",
            "-id-Interface",
            "-id-Containment",
            "-id-DataType",
            "-id-Enumeration",
            "-id-EnumerationLiteral",
            "-id-Feature",
            "-id-Field",
            "-id-Classifier",
            "-id-Link",
            "-id-Language",
            "-id-LanguageEntity",
            "-id-IKeyed",
            "-id-PrimitiveType",
            "-id-Property",
            "-id-Reference",
            "-id-StructuredDataType"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Language-dependsOn"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": null
    },
    {
      "id": "-id-Annotation",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Annotation"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Annotation"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-Annotation-annotates",
            "-id-Annotation-extends",
            "-id-Annotation-implements"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Classifier",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-Annotation-annotates",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "annotates"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Annotation-annotates"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Classifier",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Annotation"
    },
    {
      "id": "-id-Annotation-extends",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "extends"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Annotation-extends"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Annotation",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Annotation"
    },
    {
      "id": "-id-Annotation-implements",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "implements"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Annotation-implements"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Interface",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Annotation"
    },
    {
      "id": "-id-Concept",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Concept"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Concept"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-Concept-abstract",
            "-id-Concept-partition",
            "-id-Concept-extends",
            "-id-Concept-implements"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Classifier",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-Concept-abstract",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Property"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "abstract"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Concept-abstract"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
```

```json
            "key": "Property-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_builtins.Boolean",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Concept"
    },
    {
      "id": "-id-Concept-extends",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "extends"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Concept-extends"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Concept",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Concept"
    },
    {
      "id": "-id-Concept-implements",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "implements"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Concept-implements"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Interface",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Concept"
    },
    {
      "id": "-id-Concept-partition",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Property"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "partition"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Concept-partition"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Property-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_builtins.Boolean",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Concept"
    },
    {
      "id": "-id-Interface",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Interface"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Interface"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-Interface-extends"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Classifier",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-Interface-extends",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "extends"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Interface-extends"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Interface",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Interface"
    },
    {
      "id": "-id-Containment",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Containment"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Containment"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": []
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Link",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-DataType",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "DataType"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "DataType"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": []
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.LanguageEntity",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-Enumeration",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Enumeration"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Enumeration"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-Enumeration-literals"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.DataType",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-Enumeration-literals",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Containment"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "literals"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Enumeration-literals"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.EnumerationLiteral",
              "reference": null
            }
          ]
```

```
      }
    ],
    "annotations": [],
    "parent": "-id-Enumeration"
  },
  {
    "id": "-id-EnumerationLiteral",
    "classifier": {
      "language": "LionCore-M3",
      "version": "2024.1",
      "key": "Concept"
    },
    "properties": [
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-abstract"
        },
        "value": "false"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-partition"
        },
        "value": "false"
      },
      {
        "property": {
          "language": "LionCore-builtins",
          "version": "2024.1",
          "key": "LionCore-builtins-INamed-name"
        },
        "value": "EnumerationLiteral"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "IKeyed-key"
        },
        "value": "EnumerationLiteral"
      }
    ],
    "containments": [
      {
        "containment": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Classifier-features"
        },
        "children": []
      }
    ],
    "references": [
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-extends"
        },
        "targets": []
      },
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-implements"
        },
        "targets": [
          {
            "resolveInfo": "LionWeb.LionCore_M3.IKeyed",
            "reference": null
          }
        ]
      }
    ],
    "annotations": [],
    "parent": "-id-LionCore-M3"
  },
  {
    "id": "-id-Feature",
    "classifier": {
      "language": "LionCore-M3",
      "version": "2024.1",
      "key": "Concept"
    },
    "properties": [
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-abstract"
        },
        "value": "true"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
              "key": "Concept-partition"
            },
            "value": "false"
          },
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "Feature"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "Feature"
          }
        ],
        "containments": [
          {
            "containment": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Classifier-features"
            },
            "children": [
              "-id-Feature-optional"
            ]
          }
        ],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-extends"
            },
            "targets": []
          },
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-implements"
            },
            "targets": [
              {
                "resolveInfo": "LionWeb.LionCore_M3.IKeyed",
                "reference": null
              }
            ]
          }
        ],
        "annotations": [],
        "parent": "-id-LionCore-M3"
      },
      {
        "id": "-id-Feature-optional",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Property"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Feature-optional"
            },
            "value": "false"
          },
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "optional"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "Feature-optional"
          }
        ],
        "containments": [],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Property-type"
            },
            "targets": [
              {
                "resolveInfo": "LionWeb.LionCore_builtins.Boolean",
                "reference": null
              }
            ]
          }
        ],
        "annotations": [],
        "parent": "-id-Feature"
      },
      {
        "id": "-id-Field",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-abstract"
            },
            "value": "false"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-partition"
            },
            "value": "false"
          },
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "Field"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "Field"
          }
        ],
        "containments": [
          {
            "containment": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Classifier-features"
            },
            "children": [
              "-id-Field-type"
            ]
          }
        ],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-extends"
            },
            "targets": []
          },
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-implements"
            },
            "targets": [
              {
                "resolveInfo": "LionWeb.LionCore_M3.IKeyed",
                "reference": null
              }
            ]
          }
        ],
        "annotations": [],
        "parent": "-id-LionCore-M3"
      },
      {
        "id": "-id-Field-type",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Reference"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Link-multiple"
            },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "type"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Field-type"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.DataType",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Field"
    },
    {
      "id": "-id-Classifier",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Classifier"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Classifier"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-Classifier-features"
          ]
        }
      ],
      "references": [
        {
          "reference": {
```

```
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-extends"
      },
      "targets": [
        {
          "resolveInfo": "LionWeb.LionCore_M3.LanguageEntity",
          "reference": null
        }
      ]
    },
    {
      "reference": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-implements"
      },
      "targets": []
    }
  ],
  "annotations": [],
  "parent": "-id-LionCore-M3"
},
{
  "id": "-id-Classifier-features",
  "classifier": {
    "language": "LionCore-M3",
    "version": "2024.1",
    "key": "Containment"
  },
  "properties": [
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Link-multiple"
      },
      "value": "true"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Feature-optional"
      },
      "value": "true"
    },
    {
      "property": {
        "language": "LionCore-builtins",
        "version": "2024.1",
        "key": "LionCore-builtins-INamed-name"
      },
      "value": "features"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "IKeyed-key"
      },
      "value": "Classifier-features"
    }
  ],
  "containments": [],
  "references": [
    {
      "reference": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Link-type"
      },
      "targets": [
        {
          "resolveInfo": "LionWeb.LionCore_M3.Feature",
          "reference": null
        }
      ]
    }
  ],
  "annotations": [],
  "parent": "-id-Classifier"
},
{
  "id": "-id-Link",
  "classifier": {
    "language": "LionCore-M3",
    "version": "2024.1",
    "key": "Concept"
  },
  "properties": [
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-abstract"
      },
      "value": "true"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-partition"
      },
      "value": "false"
    },
    {
      "property": {
        "language": "LionCore-builtins",
        "version": "2024.1",
        "key": "LionCore-builtins-INamed-name"
      },
      "value": "Link"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "IKeyed-key"
      },
      "value": "Link"
    }
  ],
  "containments": [
    {
      "containment": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Classifier-features"
      },
      "children": [
        "-id-Link-multiple",
        "-id-Link-type"
      ]
    }
  ],
  "references": [
    {
      "reference": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-extends"
      },
      "targets": [
        {
          "resolveInfo": "LionWeb.LionCore_M3.Feature",
          "reference": null
        }
      ]
    },
    {
      "reference": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-implements"
      },
      "targets": []
    }
  ],
  "annotations": [],
  "parent": "-id-LionCore-M3"
},
{
  "id": "-id-Link-multiple",
  "classifier": {
    "language": "LionCore-M3",
    "version": "2024.1",
    "key": "Property"
  },
  "properties": [
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Feature-optional"
      },
      "value": "false"
    },
    {
      "property": {
        "language": "LionCore-builtins",
        "version": "2024.1",
        "key": "LionCore-builtins-INamed-name"
      },
      "value": "multiple"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "IKeyed-key"
      },
      "value": "Link-multiple"
    }
  ],
  "containments": [],
  "references": [
    {
      "reference": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Property-type"
      },
      "targets": [
        {
          "resolveInfo": "LionWeb.LionCore_builtins.Boolean",
          "reference": null
        }
      ]
    }
  ],
  "annotations": [],
  "parent": "-id-Link"
},
{
  "id": "-id-Link-type",
  "classifier": {
    "language": "LionCore-M3",
    "version": "2024.1",
    "key": "Reference"
  },
  "properties": [
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Link-multiple"
      },
      "value": "false"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Feature-optional"
      },
      "value": "false"
    },
    {
      "property": {
        "language": "LionCore-builtins",
        "version": "2024.1",
        "key": "LionCore-builtins-INamed-name"
      },
      "value": "type"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "IKeyed-key"
      },
      "value": "Link-type"
    }
  ],
  "containments": [],
  "references": [
    {
      "reference": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Link-type"
      },
      "targets": [
        {
          "resolveInfo": "LionWeb.LionCore_M3.Classifier",
          "reference": null
        }
      ]
    }
  ],
  "annotations": [],
  "parent": "-id-Link"
},
{
  "id": "-id-Language",
  "classifier": {
    "language": "LionCore-M3",
    "version": "2024.1",
    "key": "Concept"
  },
  "properties": [
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-abstract"
      },
      "value": "false"
    },
    {
      "property": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept-partition"
      },
      "value": "true"
    },
    {
      "property": {
        "language": "LionCore-builtins",
        "version": "2024.1",
        "key": "LionCore-builtins-INamed-name"
      },
      "value": "Language"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "IKeyed-key"
        },
        "value": "Language"
      }
    ],
    "containments": [
      {
        "containment": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Classifier-features"
        },
        "children": [
          "-id-Language-version",
          "-id-Language-dependsOn",
          "-id-Language-entities"
        ]
      }
    ],
    "references": [
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-extends"
        },
        "targets": []
      },
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-implements"
        },
        "targets": [
          {
            "resolveInfo": "LionWeb.LionCore_M3.IKeyed",
            "reference": null
          }
        ]
      }
    ],
    "annotations": [],
    "parent": "-id-LionCore-M3"
  },
  {
    "id": "-id-Language-dependsOn",
    "classifier": {
      "language": "LionCore-M3",
      "version": "2024.1",
      "key": "Reference"
    },
    "properties": [
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Link-multiple"
        },
        "value": "true"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Feature-optional"
        },
        "value": "true"
      },
      {
        "property": {
          "language": "LionCore-builtins",
          "version": "2024.1",
          "key": "LionCore-builtins-INamed-name"
        },
        "value": "dependsOn"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "IKeyed-key"
        },
        "value": "Language-dependsOn"
      }
    ],
    "containments": [],
    "references": [
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Link-type"
        },
        "targets": [
          {
            "resolveInfo": "LionWeb.LionCore_M3.Language",
            "reference": null
          }
        ]
      }
    ],
    "annotations": [],
    "parent": "-id-Language"
  },
  {
    "id": "-id-Language-entities",
    "classifier": {
      "language": "LionCore-M3",
      "version": "2024.1",
      "key": "Containment"
    },
    "properties": [
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Link-multiple"
        },
        "value": "true"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Feature-optional"
        },
        "value": "true"
      },
      {
        "property": {
          "language": "LionCore-builtins",
          "version": "2024.1",
          "key": "LionCore-builtins-INamed-name"
        },
        "value": "entities"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "IKeyed-key"
        },
        "value": "Language-entities"
      }
    ],
    "containments": [],
    "references": [
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Link-type"
        },
        "targets": [
          {
            "resolveInfo": "LionWeb.LionCore_M3.LanguageEntity",
            "reference": null
          }
        ]
      }
    ],
    "annotations": [],
    "parent": "-id-Language"
  },
  {
    "id": "-id-Language-version",
    "classifier": {
      "language": "LionCore-M3",
      "version": "2024.1",
      "key": "Property"
    },
    "properties": [
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Feature-optional"
        },
        "value": "false"
      },
      {
        "property": {
          "language": "LionCore-builtins",
          "version": "2024.1",
          "key": "LionCore-builtins-INamed-name"
        },
        "value": "version"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "IKeyed-key"
        },
        "value": "Language-version"
      }
    ],
    "containments": [],
    "references": [
      {
        "reference": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Property-type"
        },
        "targets": [
          {
            "resolveInfo": "LionWeb.LionCore_builtins.String",
            "reference": null
          }
        ]
      }
    ],
    "annotations": [],
    "parent": "-id-Language"
  },
  {
    "id": "-id-LanguageEntity",
    "classifier": {
      "language": "LionCore-M3",
      "version": "2024.1",
      "key": "Concept"
    },
    "properties": [
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-abstract"
        },
        "value": "true"
      },
      {
        "property": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Concept-partition"
        },
        "value": "false"
      },
      {
        "property": {
          "language": "LionCore-builtins",
          "version": "2024.1",
          "key": "LionCore-builtins-INamed-name"
        },
        "value": "LanguageEntity"
      },
      {
```

```
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "LanguageEntity"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": []
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": []
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.IKeyed",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-IKeyed",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Interface"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "IKeyed"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "IKeyed"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-IKeyed-key"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Interface-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_builtins.INamed",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-IKeyed-key",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Property"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "key"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "IKeyed-key"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Property-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_builtins.String",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-IKeyed"
    },
    {
      "id": "-id-PrimitiveType",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "PrimitiveType"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "PrimitiveType"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": []
        }
      ],
      "references": [
        {
```

```
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.DataType",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-Property",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
```

```json
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "Property"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "Property"
          }
        ],
        "containments": [
          {
            "containment": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Classifier-features"
            },
            "children": [
              "-id-Property-type"
            ]
          }
        ],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-extends"
            },
            "targets": [
              {
                "resolveInfo": "LionWeb.LionCore_M3.Feature",
                "reference": null
              }
            ]
          },
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-implements"
            },
            "targets": []
          }
        ],
        "annotations": [],
        "parent": "-id-LionCore-M3"
```

```
    },
    {
      "id": "-id-Property-type",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Reference"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "type"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Property-type"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.DataType",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-Property"
    },
    {
      "id": "-id-Reference",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Reference"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "Reference"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": []
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Link",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-StructuredDataType",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-abstract"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-partition"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "StructuredDataType"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "StructuredDataType"
        }
      ],
      "containments": [
        {
          "containment": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Classifier-features"
          },
          "children": [
            "-id-StructuredDataType-fields"
          ]
        }
      ],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-extends"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.DataType",
              "reference": null
            }
          ]
        },
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Concept-implements"
          },
          "targets": []
        }
      ],
      "annotations": [],
      "parent": "-id-LionCore-M3"
    },
    {
      "id": "-id-StructuredDataType-fields",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Containment"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-multiple"
          },
          "value": "true"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Feature-optional"
          },
          "value": "false"
        },
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "fields"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "StructuredDataType-fields"
        }
      ],
      "containments": [],
      "references": [
        {
          "reference": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "Link-type"
          },
          "targets": [
            {
              "resolveInfo": "LionWeb.LionCore_M3.Field",
              "reference": null
            }
          ]
        }
      ],
      "annotations": [],
      "parent": "-id-StructuredDataType"
    }
  ]
}
```

5.2. Pre-defined elements

The LionCore built-in elements.

```
{
  "serializationFormatVersion": "2024.1",
  "languages": [
    {
      "key": "LionCore-M3",
      "version": "2024.1"
    },
    {
      "key": "LionCore-builtins",
      "version": "2024.1"
    }
  ],
  "nodes": [
    {
      "id": "LionCore-builtins",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Language"
      },
        "properties": [
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "LionCore_builtins"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Language-version"
            },
            "value": "2024.1"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "LionCore-builtins"
          }
        ],
        "containments": [
          {
            "containment": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Language-entities"
            },
            "children": [
              "LionCore-builtins-String",
              "LionCore-builtins-Boolean",
              "LionCore-builtins-Integer",
              "LionCore-builtins-Node",
              "LionCore-builtins-INamed"
            ]
          }
        ],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Language-dependsOn"
            },
            "targets": []
          }
        ],
        "annotations": [],
        "parent": null
      },
      {
        "id": "LionCore-builtins-String",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "PrimitiveType"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "String"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "LionCore-builtins-String"
          }
        ],
        "containments": [],
        "references": [],
        "annotations": [],
        "parent": "LionCore-builtins"
      },
      {
        "id": "LionCore-builtins-Boolean",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "PrimitiveType"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "Boolean"
          },
          {
```

```
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "LionCore-builtins-Boolean"
        }
      ],
      "containments": [],
      "references": [],
      "annotations": [],
      "parent": "LionCore-builtins"
    },
    {
      "id": "LionCore-builtins-Integer",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "PrimitiveType"
      },
      "properties": [
        {
          "property": {
            "language": "LionCore-builtins",
            "version": "2024.1",
            "key": "LionCore-builtins-INamed-name"
          },
          "value": "Integer"
        },
        {
          "property": {
            "language": "LionCore-M3",
            "version": "2024.1",
            "key": "IKeyed-key"
          },
          "value": "LionCore-builtins-Integer"
        }
      ],
      "containments": [],
      "references": [],
      "annotations": [],
      "parent": "LionCore-builtins"
    },
    {
      "id": "LionCore-builtins-Node",
      "classifier": {
        "language": "LionCore-M3",
        "version": "2024.1",
        "key": "Concept"
      },
      "properties": [
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-abstract"
            },
            "value": "true"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-partition"
            },
            "value": "false"
          },
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "Node"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "LionCore-builtins-Node"
          }
        ],
        "containments": [
          {
            "containment": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Classifier-features"
            },
            "children": []
          }
        ],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-extends"
            },
            "targets": []
          },
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Concept-implements"
            },
            "targets": []
          }
        ],
        "annotations": [],
        "parent": "LionCore-builtins"
      },
      {
        "id": "LionCore-builtins-INamed",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Interface"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "INamed"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "LionCore-builtins-INamed"
          }
        ],
        "containments": [
          {
            "containment": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Classifier-features"
            },
            "children": [
              "LionCore-builtins-INamed-name"
            ]
          }
        ],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Interface-extends"
            },
            "targets": []
          }
        ],
        "annotations": [],
        "parent": "LionCore-builtins"
      },
      {
        "id": "LionCore-builtins-INamed-name",
        "classifier": {
          "language": "LionCore-M3",
          "version": "2024.1",
          "key": "Property"
        },
        "properties": [
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Feature-optional"
            },
            "value": "false"
          },
          {
            "property": {
              "language": "LionCore-builtins",
              "version": "2024.1",
              "key": "LionCore-builtins-INamed-name"
            },
            "value": "name"
          },
          {
            "property": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "IKeyed-key"
            },
            "value": "LionCore-builtins-INamed-name"
          }
        ],
        "containments": [],
        "references": [
          {
            "reference": {
              "language": "LionCore-M3",
              "version": "2024.1",
              "key": "Property-type"
            },
            "targets": [
              {
                "resolveInfo": "LionWeb.LionCore_builtins.String",
                "reference": null
              }
            ]
          }
        ],
        "annotations": [],
        "parent": "LionCore-builtins-INamed"
      }
    ]
  }
```

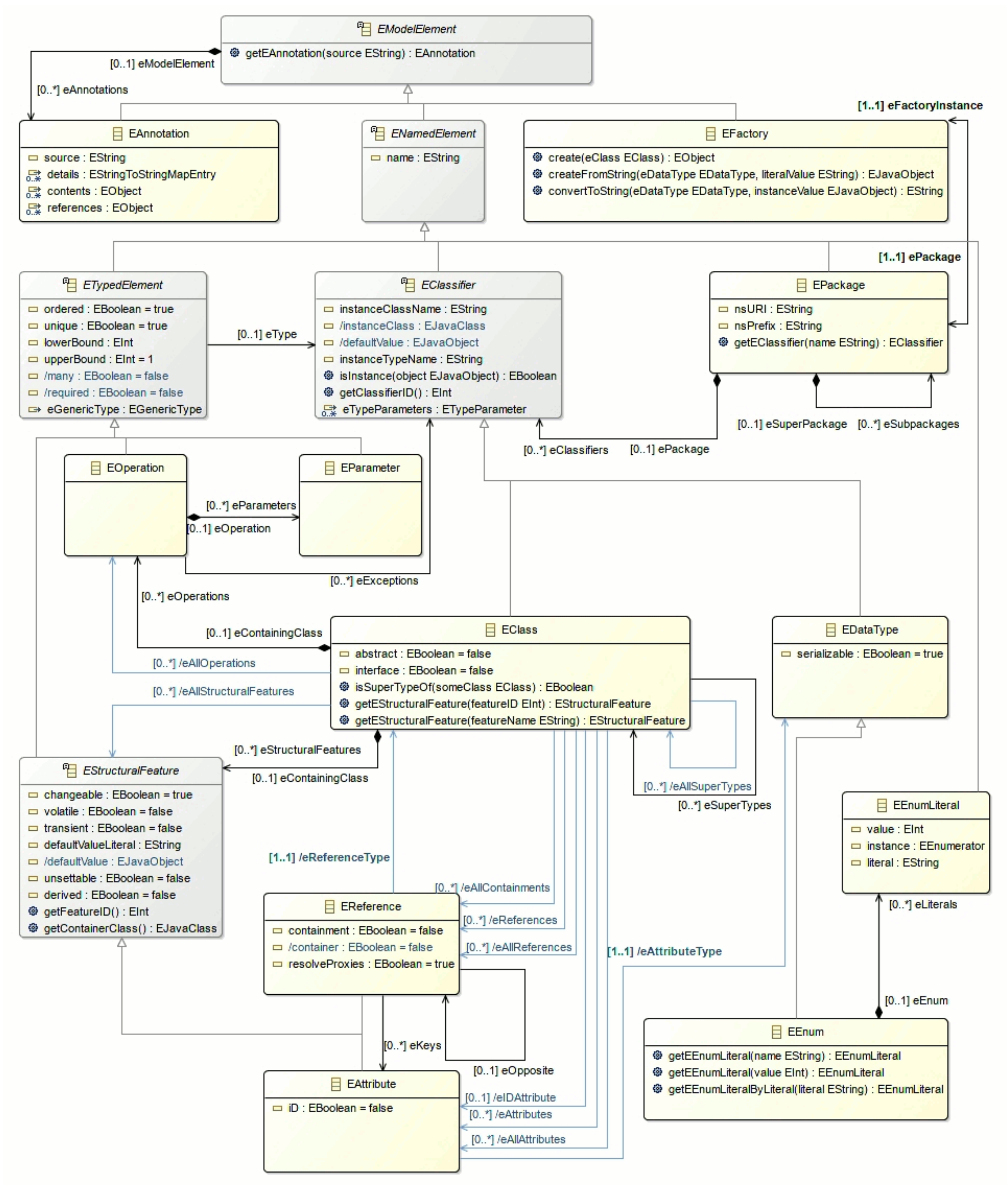
6. Comparison with other meta-metamodels

Main difference: we aim for multiple implementations on different platforms, we want to serve both textual and projectional languages and editors.

LionCore	Ecore	MPS
Language	EPackage	Language's structure aspect (docs, javadoc)
Annotation	Adapter (M1) / EAnnotation (M2)	NodeAttribute (docs)
Concept	EClass	ConceptDeclaration (docs, javadoc)
Interface	EClass	InterfaceConceptDeclaration (docs, javadoc)
PrimitiveType	EDataType	PrimitiveDataTypeDeclaration (javadoc)
Enumeration	EEnum	EnumerationDeclaration (docs, javadoc)
EnumerationLiteral	EEnumLiteral	EnumerationMemberLiteral (docs, javadoc)
Containment	EReference	LinkDeclaration (docs, javadoc)
Reference	EReference	LinkDeclaration (docs, javadoc)
Property	EAttribute	PropertyDeclaration (docs, javadoc)
IKeyed	—	—
LanguageEntity	EClassifier	IStructureElement (javadoc)
Classifier	EClass	AbstractConceptDeclaration (javadoc)
DataType	EDataType	DataTypeDeclaration (javadoc)
Feature	EStructuralFeature	(docs, javadoc)
Link	EReference	LinkDeclaration (javadoc)

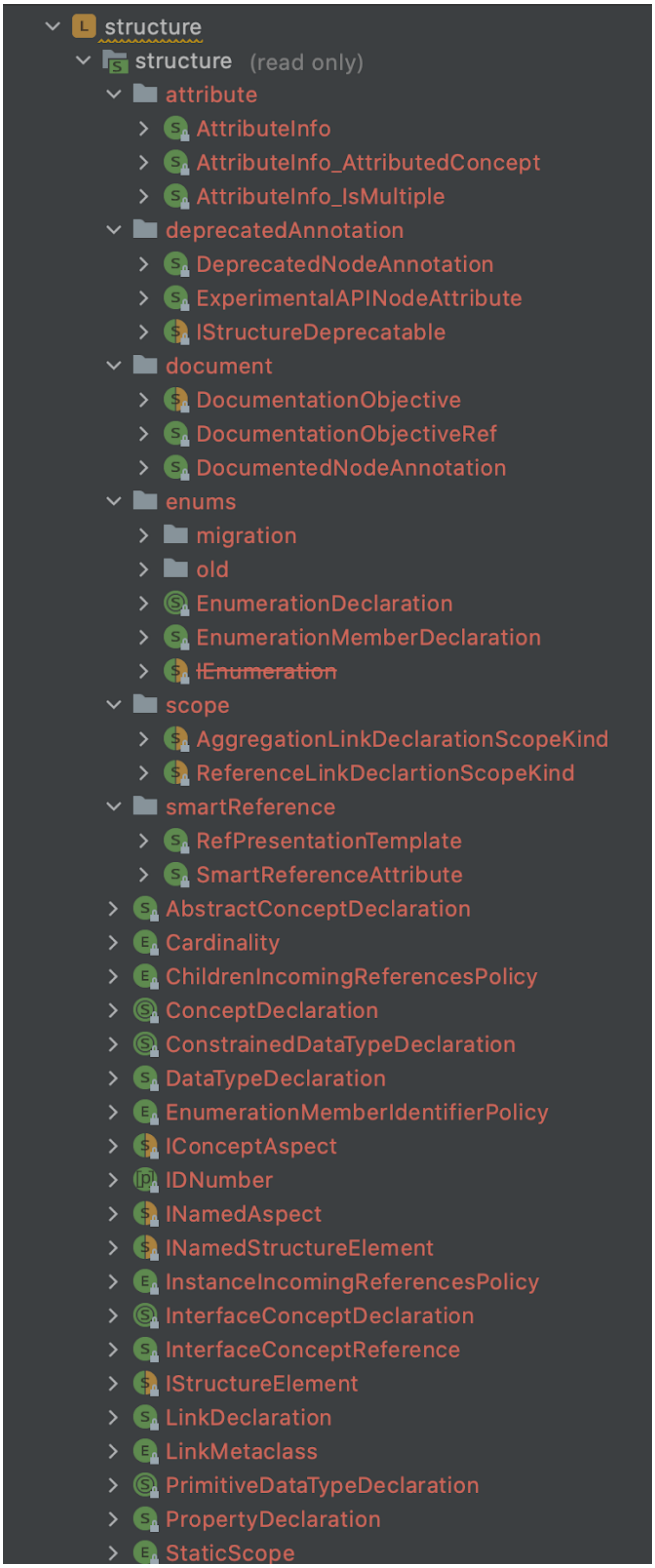
6.1. Comparison with Ecore

javadoc



EModelElement
getEAnnotation(source EString) : EAnnotation
[0..1] eModelElement
[0..*] eAnnotations
[1..1] eFactoryInstance
EAnnotation
source : EString
details : EStringToStringMapEntry
contents : EObject
references : EObject
ENamedElement
name : EString
EFactory
create(eClass EClass) : EObject
createFromString(eDataType EDataType, literalValue EString) : EJavaObject
convertToString(eDataType EDataType, instanceValue EJavaObject) : EString
[1..1] ePackage
ETypedElement
ordered : EBoolean = true
unique : EBoolean = true
lowerBound : EInt
upperBound : EInt = 1
/many : EBoolean = false
/required : EBoolean = false
eGenericType : EGenericType
[0..1] eType
EClassifier
instanceClassName : EString
/instanceClass : EJavaClass
/defaultValue : EJavaObject
instanceTypeName : EString
isInstance(object EJavaObject) : EBoolean
getClassifierID() : EInt
eTypeParameters : ETypeParameter
EPackage
nsURI : EString
nsPrefix : EString
getEClassifier(name EString) : EClassifier
[0..1] eSuperPackage
[0..*] eSubpackages
[0..*] eClassifiers
[0..1] ePackage
EOperation
EParameter
[0..*] eParameters
[0..1] eOperation
[0..*] eExceptions
[0..*] eOperations
[0..1] eContainingClass
[0..*] /eAllOperations
[0..*] /eAllStructuralFeatures
EClass
abstract : EBoolean = false
interface : EBoolean = false
isSuperTypeOf(someClass EClass) : EBoolean
getEStructuralFeature(featureID EInt) : EStructuralFeature
getEStructuralFeature(featureName EString) : EStructuralFeature
EDataType
serializable : EBoolean = true
[0..*] eStructuralFeatures
[0..1] eContainingClass
[0..*] /eAllSuperTypes
[0..*] eSuperTypes
EStructuralFeature
changeable : EBoolean = true
volatile : EBoolean = false
transient : EBoolean = false
defaultValueLiteral : EString
/defaultValue : EJavaObject
unsettable : EBoolean = false
derived : EBoolean = false
getFeatureID() : EInt
getContainerClass() : EJavaClass
[1..1] /eReferenceType
EEnumLiteral
value : EInt
instance : EEnumerator
literal : EString
[0..*] /eAllContainments
EReference
containment : EBoolean = false
/container : EBoolean = false
resolveProxies : EBoolean = true
[0..*] /eReferences
[0..*] /eAllReferences
[1..1] /eAttributeType
[0..*] eLiterals
[0..1] eEnum
[0..*] eKeys
[0..1] eOpposite
EEnum
getEEnumLiteral(name EString) : EEnumLiteral
getEEnumLiteral(value EInt) : EEnumLiteral
getEEnumLiteralByLiteral(literal EString) : EEnumLiteral
EAttribute
iD : EBoolean = false
[0..1] /eIDAttribute
[0..*] /eAttributes
[0..*] /eAllAttributes

6.2. Comparison with MPS



```
structure
  structure (read only)
    attribute
      AttributeInfo
      AttributeInfo_AttributedConcept
      AttributeInfo_IsMultiple
    deprecatedAnnotation
      DeprecatedNodeAnnotation
      ExperimentalAPINodeAttribute
      IStructureDeprecatable
    document
      DocumentationObjective
      DocumentationObjectiveRef
      DocumentedNodeAnnotation
    enums
      migration
      old
      EnumerationDeclaration
      EnumerationMemberDeclaration
      ~~IEnumeration~~
    scope
      AggregationLinkDeclarationScopeKind
      ReferenceLinkDeclartionScopeKind
    smartReference
      RefPresentationTemplate
      SmartReferenceAttribute
    AbstractConceptDeclaration
    Cardinality
    ChildrenIncomingReferencesPolicy
    ConceptDeclaration
    ConstrainedDataTypeDeclaration
    DataTypeDeclaration
    EnumerationMemberIdentifierPolicy
    IConceptAspect
    IDNumber
    INamedAspect
    INamedStructureElement
    InstanceIncomingReferencesPolicy
    InterfaceConceptDeclaration
    InterfaceConceptReference
    IStructureElement
    LinkDeclaration
    LinkMetaclass
    PrimitiveDataTypeDeclaration
    PropertyDeclaration
    StaticScope
```

[1] Rename M3 Metamodel to Language? #78

[2] Rename M3 property id → key #90

[3] Is version part of M3 Metamodel? #7

[4] Add version property to M3 Metamodel #92

[5] Shall we have `IKeyed` interface in M3? #142

[6] Is `Language.dependsOn` a `UsedLanguage`? #145

[7] Metamodel dependencies: explicit, transitive? #50

[8] What does Language.version mean semantically? #130

[9] How to refer from one language to another? #131

[10] Details on builtin language #153

[11] Repo API: Do we need model partitions? #29

[12] If and how to represent Annotations in M3 #13

[13] Details on Annotations #154

[14] Can we have multiple instances of the same Annotation associated to a certain Node? #32

[15] We disallow concept nodes in annoations / annotation instances in containments #320

[16] Rename `ConceptInterface`? #190

[17] Value Types #265

[18] We do not allow multiple property values #289

[19] Rename FeaturesContainer to Classifier #105

[20] Which parts of a link can be specialized? #8

[21] Disallow redefining / overriding inherited feature #139

[22] Name clashes during inheritance #97

[23] Is M3 `NamespacedEntity` an abstract concept or interface? #143

[24] Allowed characters for names in metamodels #48

[25] Make LionCore M3 and builtins language definitions have version-dependent IDs #279

[26] Requirements on metamodel keys #91

[27] Change builtins language name? #195

[28] Key of builtin stdlib #141

[29] Discussion on implicitly importing stdlib

[30] Metalevel of builtin standard library #196

[31] Do we need to represent BaseConcept? #71

[32] Introducing the builtin interface INamed #86

[33] Supported built-in primitive types #9

[34] Metamodel.id/NamespacedEntity.id vs. Node id #80

[35] Rework NamespaceProvider and NamespacedEntity in M3 #146