Exploring - Classes

Classes and Class Diagrams are probably the most commonly used bit of UML, so they get their own document.

Version: 0.0.6.1

All the content in this document is pulled from an EA model. This content is saved in a Tagged value of the Document element. This document also has a Table of Contents, which gets updated each time the document is re-generated.

Contents

Section: Introduction

1 Classes

Because the Class Diagram in used for all kinds of analysis, design and build activities, so it gets a document all of its own, to show how all aspects of your EA models can be documented.

1.1 Basic class diagram with relationships

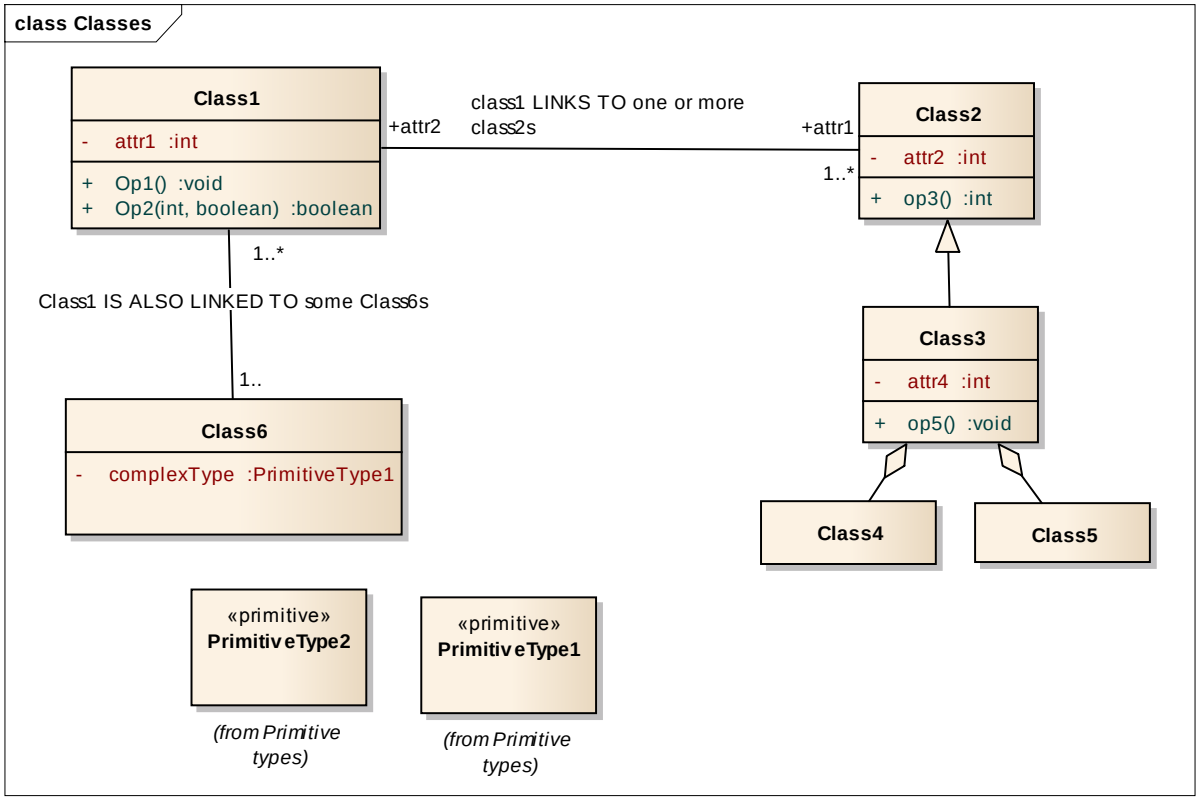

Figure 1 : Classes

This is a simple UML Class Diagram, which contains most of the commonly-used features of class diagrams:

- *Associations, with names and multiplicities*
- *Inheritance relationships*
- *Aggregation relationships*

1.1.1 Class1

Superclass: none

Sub-classes: none

1.1.1.1 Attributes

Attribute	Visibility
attr1	Private

1.1.1.2 Methods

Name	Visibility
Op1	Public
Op2	Public

1.1.1.3 Related Classes

Relationship Type	Relationship name	Cardinality	Class	Role
Association	class1 LINKS TO one or more class2s	1..*	Class2	attr1
Association	Class1 IS ALSO LINKED TO some Class6s	1..	Class6	

1.1.2 Class2

Superclass: none

Sub-classes: none

1.1.2.1 Attributes

Attribute	Visibility
attr2	Private

1.1.2.2 Methods

Name	Visibility
op3	Public

1.1.2.3 Related Classes

Relationship Type	Relationship name	Cardinality	Class	Role
Association	class1 LINKS TO one or more class2s		Class1	attr2

1.1.3 Class3

Superclass: Class2

Sub-classes: none

1.1.3.1 Attributes

Attribute	Visibility
attr2 (Class2)	Private
attr4	Private

1.1.3.2 Methods

Name	Visibility
op3 (Class2)	Public
op5	Public

1.1.4 Class4

Superclass: none

Sub-classes: none

1.1.5 Class5

Superclass: none

Sub-classes: none

1.1.6 Class6

Superclass: none

Sub-classes: none

1.1.6.1 Attributes

Attribute	Visibility
complexType	Private

1.1.6.2 Related Classes

Relationship Type	Relationship name	Cardinality	Class	Role
Association	Class1 IS ALSO LINKED TO some Class6s	1..*	Class1	

1.1.7 Primitive types

Primitive Type
PrimitiveType2
PrimitiveType1

1.2 Detailed Class

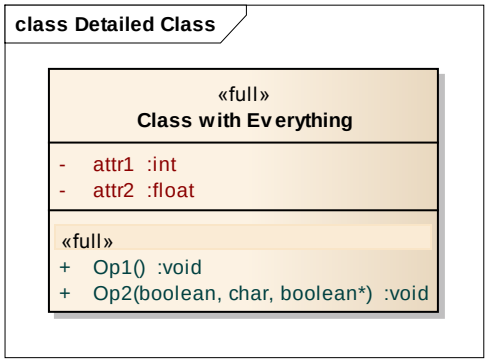

Figure 2 : Detailed Class

This class has all of the aspects of a Class which EA allows:

- *Attributes*
- *Operations (Methods)*
- *Those methods have parameters*

1.2.1 Class with Everything

Superclass: none

Sub-classes: none

Stereotype: full

1.2.1.1 Attributes

Attribute	Visibility
attr1	Private
attr2	Private

1.2.1.2 Methods

1.2.1.2.1 Op1

Visibility: Public

1.2.1.2.2 Op2

Visibility: Public

Parameter Summary: (in) param1: boolean, (in) param2: char, (inout) param3: boolean

Precondition summary: A Precondition: This is a Pre-condition to Op2

1.2.1.2.2.1 Parameters

Parameter	Description
param1	This parameter even has a description
param2	so does this one
param3	

1.2.1.2.2.2 Preconditions

Name	Description
A Precondition	This is a Pre-condition to Op2

1.3 Class and Table mapping

This shows a really useful, and not often used part of EA - it's ability to let us link not just Element to Elements, but Elements to Attributes/Operations, and Attributes/Operations to other ones.

This means we can achieve attribute-level traceability through a model.

In this example, we have created some links between the Attributes in a Class, and the columns in a table. In this way, we can show how the Table saves instances of the class (this is just an example, to show what is possible, not a serious attempt at a Class/Table mapping!)

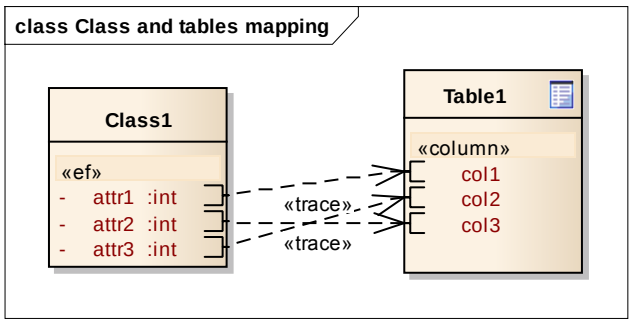

Figure 3 : Class and tables mapping

This shows how the Attributes of the Class are mapped to the columns of the table.

1.3.1 Class1

Superclass: none

Sub-classes: none

1.3.1.1 Attributes

Attribute name	Type	Trace forward to Table::column
attr1	int	Table1:: col1
attr2	int	Table1:: col3
attr3	int	Table1:: col2

1.3.2 Table1

1.3.2.1 Columns

Attribute	Visibility	Traces back to Class Attribute
col1	Public	Class1:: attr1
col2	Public	Class1:: attr3
col3	Public	Class1:: attr2

1.4 More Element Feature traceability

In this example, we have linked some User Interface Widgets with the data attributes and operations to which they map.

If you look into the model, you might notice that it doesn't matter what the relationship type is between the related Element Features: we have used both Dependency and Realization.

Using this technique, we can map the details of the UI in great detail to the attributes and methods in the 'Data layer' class.

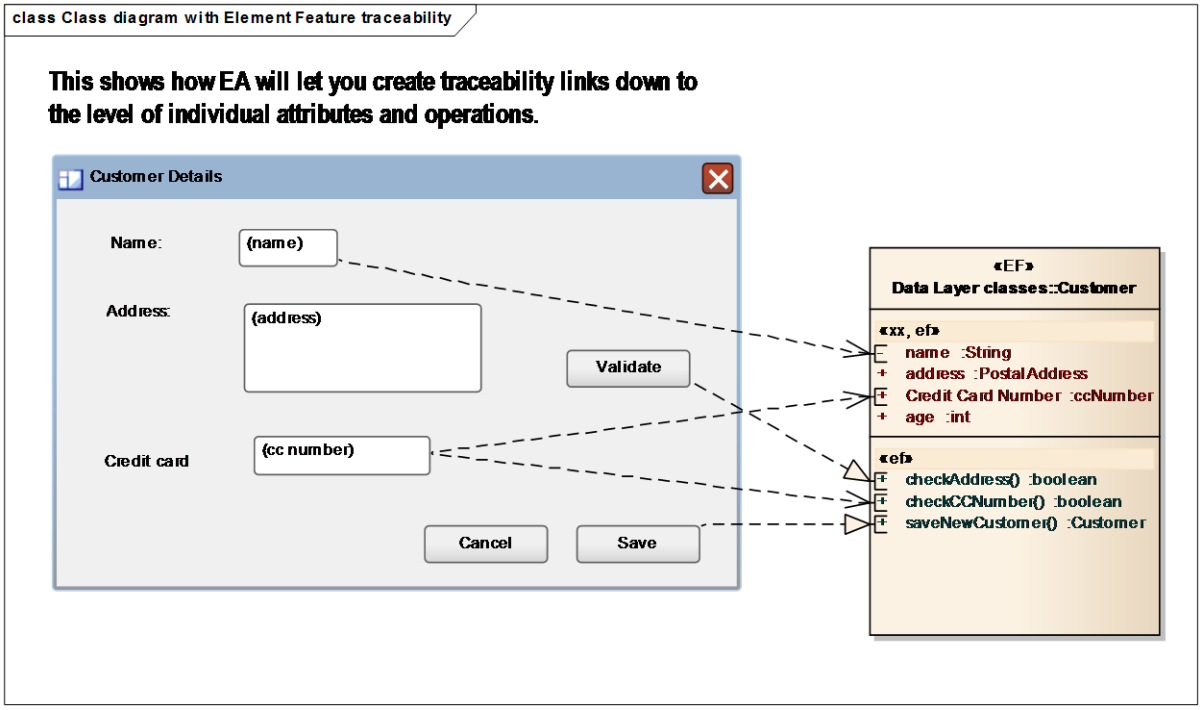

Figure 4 : Class diagram with Element Feature traceability

1.4.1 Data Layer classes

1.4.1.1 Customer

1.4.1.1.1 Attributes

Attribute	Visibility	Trace to UI
name	Private	(name)
address	Public	
Credit Card Number	Public	(cc number)
age	Public	

1.4.1.1.2 Methods

Method	Visibility	Element Feature links
checkAddress	Public	Validate
checkCCNumber	Public	(cc number)
saveNewCustomer	Public	Save

1.4.2 User Interface

1.4.2.1 Customer Details

Stereotype	Name	Element Feature links

Stereotype	Name	Element Feature links
win32StaticText	Name:	
win32StaticText	Credit card number:	
win32Button	Validate	Customer:: checkAddress
win32Button	Save	Customer:: saveNewCustomer
win32Button	Cancel	
win32Edit	(cc number)	Customer:: checkCCNumber, Customer:: Credit Card Number
win32Edit	(address)	
win32StaticText	Address:	
win32Edit	(name)	Customer:: name

End