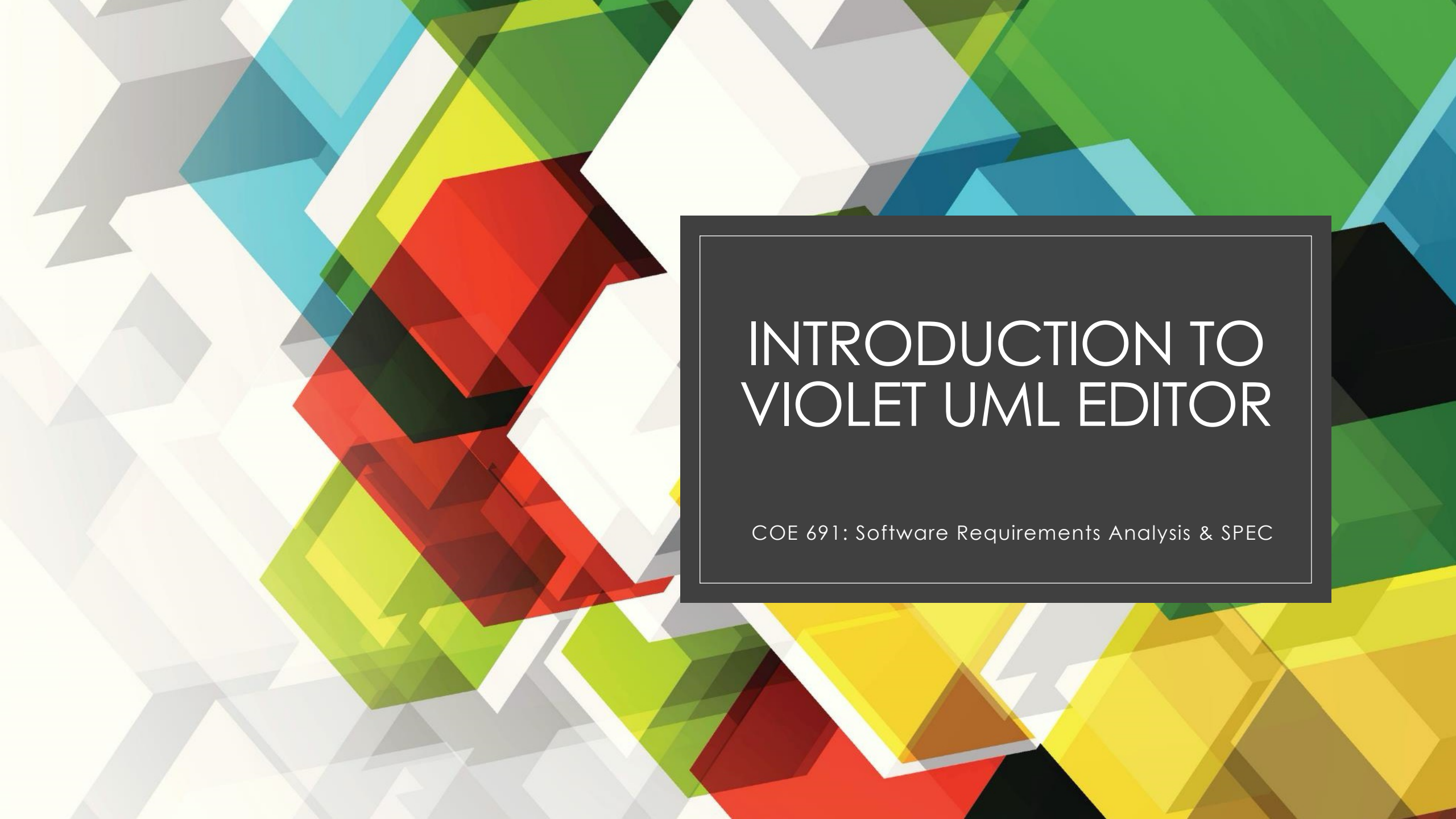


INTRODUCTION TO VIOLET UML EDITOR

COE 691: Software Requirements Analysis & SPEC

What is the Violet UML Editor?

POWERFUL UML MODELLING SOFTWARE

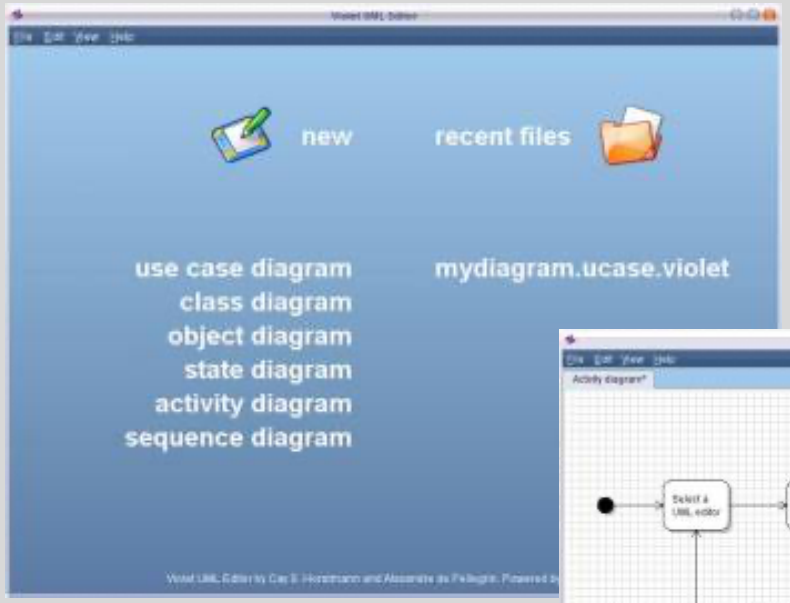
VERY EASY TO LEARN AND USE

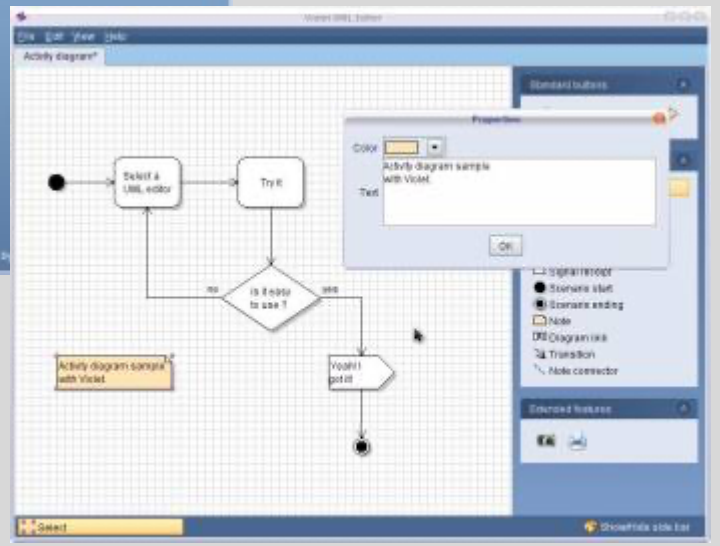
CROSS-PLATFORM

FREE

Violet UML Editor

- Violet is a Java program.
- Violet is open source.
- Violet can be used in many ways: a single application, an applet, via Java web start, and as an Eclipse plugin (fully-integrated into Eclipse).





Installation Steps

- If you do not already have Java on your machine, follow this link: https://www.java.com/en/download/ to download and setup Java.

- The official Violet Documentation at http://alexdp.free.fr/violetumleditor/page.php?id=en:installation offers step-by-step instructions to get Violet up and running.

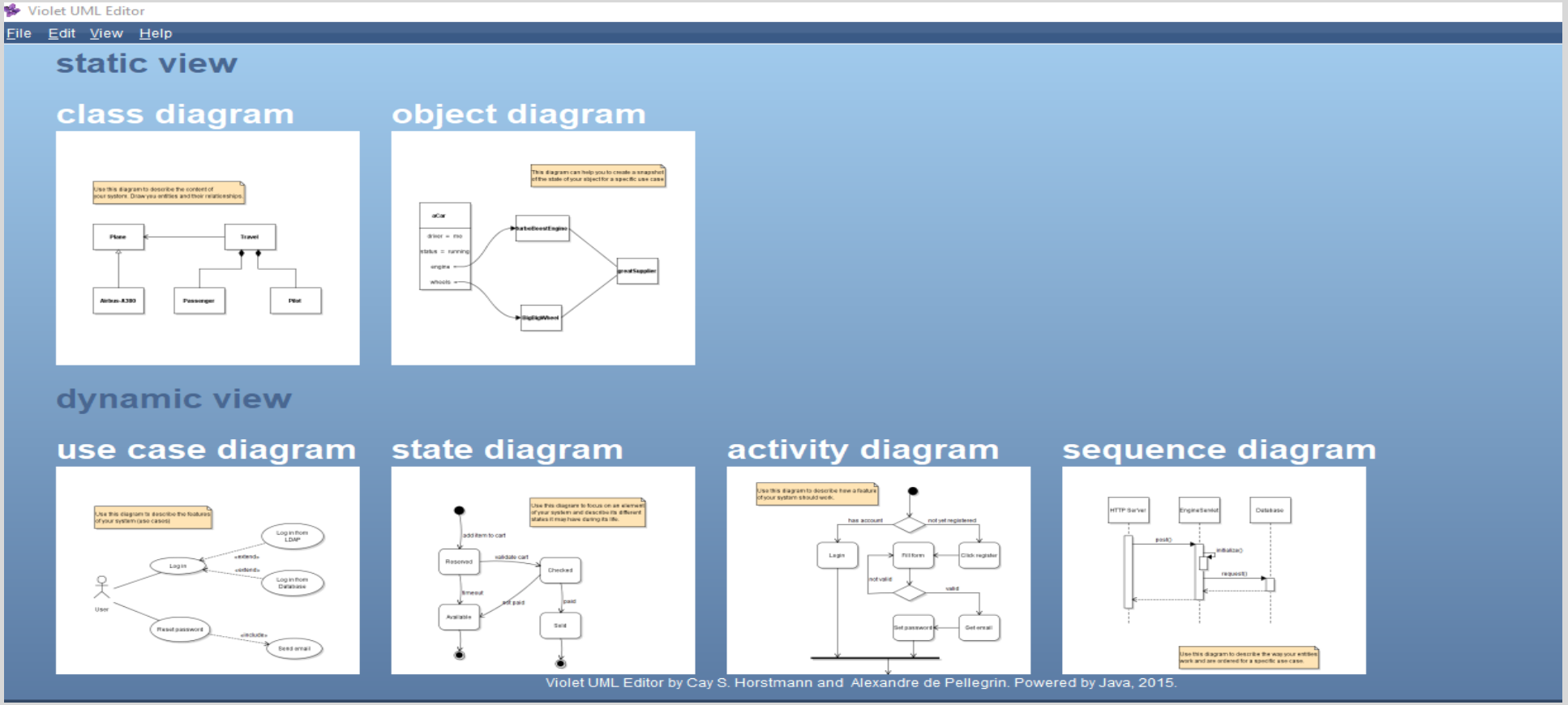
- In general, download file ***com.horstmann.violet-2.0.1.jar*** from this Source Forge site: http://sourceforge.net/projects/violet to get the Violet editor. The runnable jar will execute an install on your machine when prompt.

Violet only offers capabilities to create the following UML diagrams:

- Use case diagrams
- Class diagrams
- Object diagrams
- State diagrams
- Activity diagrams
- Sequence diagrams

We will focus on Use case and Class diagrams for this tutorial.

This is the main page of the editor

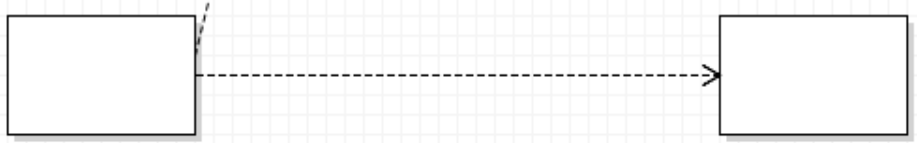


You just need to select your desired UML diagram to create a new template with the appropriate tooling.

Violet UML Editor

File Edit View Help

1.Class diagram*



****
To select one of the tools, you just need to select that tool and double-click on the editor template.
****

****
Buttons to undo, redo, zoom, zoom out, cut, copy, and paste.
****

****
In general, the template for each UML diagram is the same, where each template is equipped with the necessary tools in the "Diagram tools" that correspond to the functionality of that specific UML diagram.
****

****
There is also a tool that allows the user to create notes, similar to the ones on this template. Each note can be connected to an icon using the "Note connector" tool in the "Diagram tools".
****

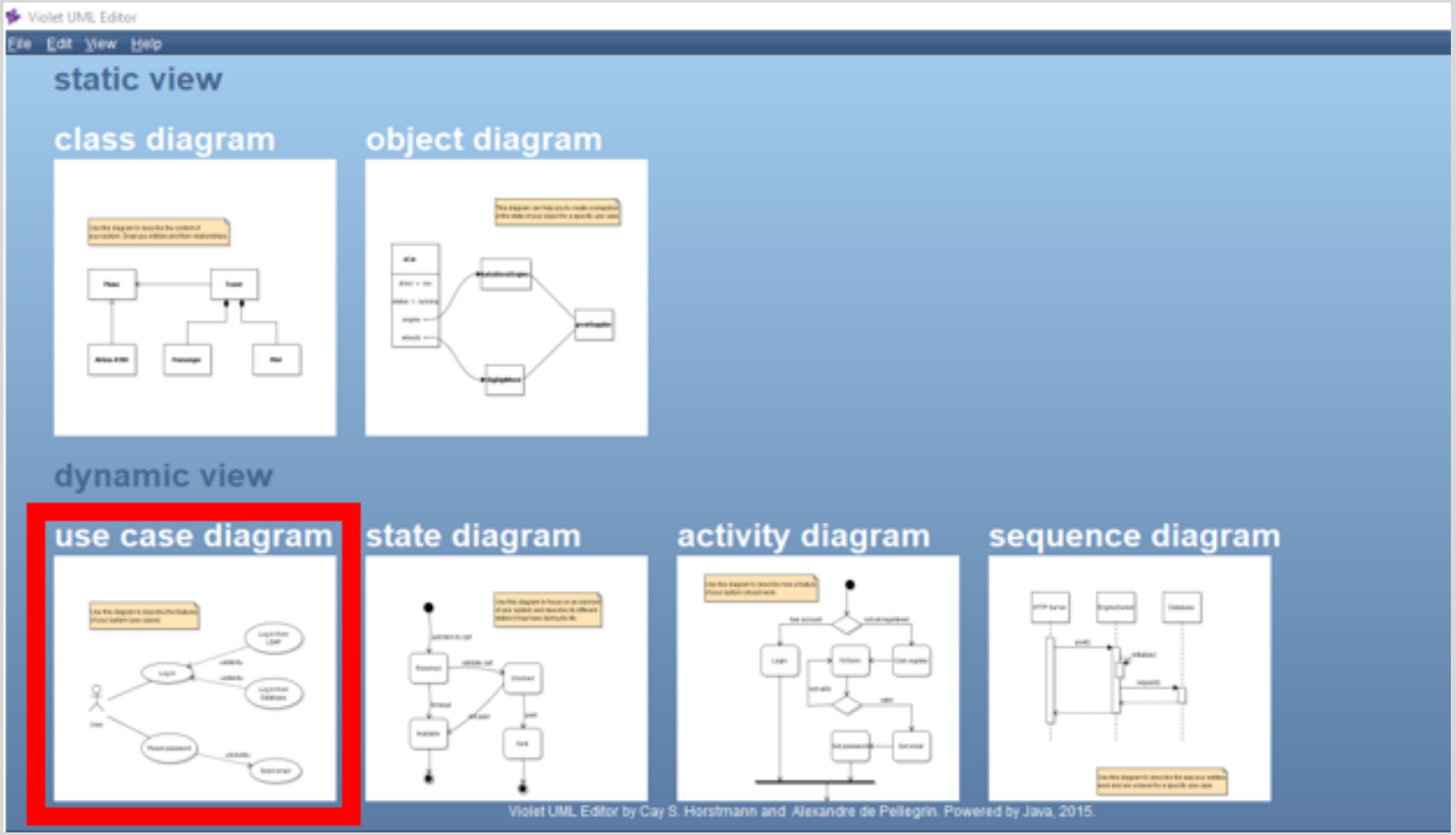
Standard buttons

Diagram tools

- Select
- Class
- Interface
- Package
- Note
- Depends on
- Inherits from
- Implements interface
- Is associated with
- Is an aggregate of
- Is composed of
- Note connector

Extended functions

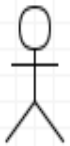
How to Create a Use Case Diagram



Violet UML Editor

File Edit View Help

1.Use case diagram*



Actor

Standard buttons

Diagram tools

Select

Actor

Use case

Note

Interaction

«extend» : precises the beh...

«include» : includes call to

Generalization : is a more s...

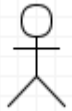
Note connector

Extended functions

Violet UML Editor

File Edit View Help

1.Use case diagram*



Actor

Standard buttons

Diagram tools

- Select
- Actor
- Use case
- Note
- Interaction
- «extend» : precises the beh...
- «include» : includes call to
- Generalization : is a more s...
- Note connector

Extended functions

Violet UML Editor

File Edit View Help

1.Use case diagram*

Right-click on the Use case to edit its name by choosing "Properties".
You can also double-click the Use case and the Properties window automatically pops up.

Actor

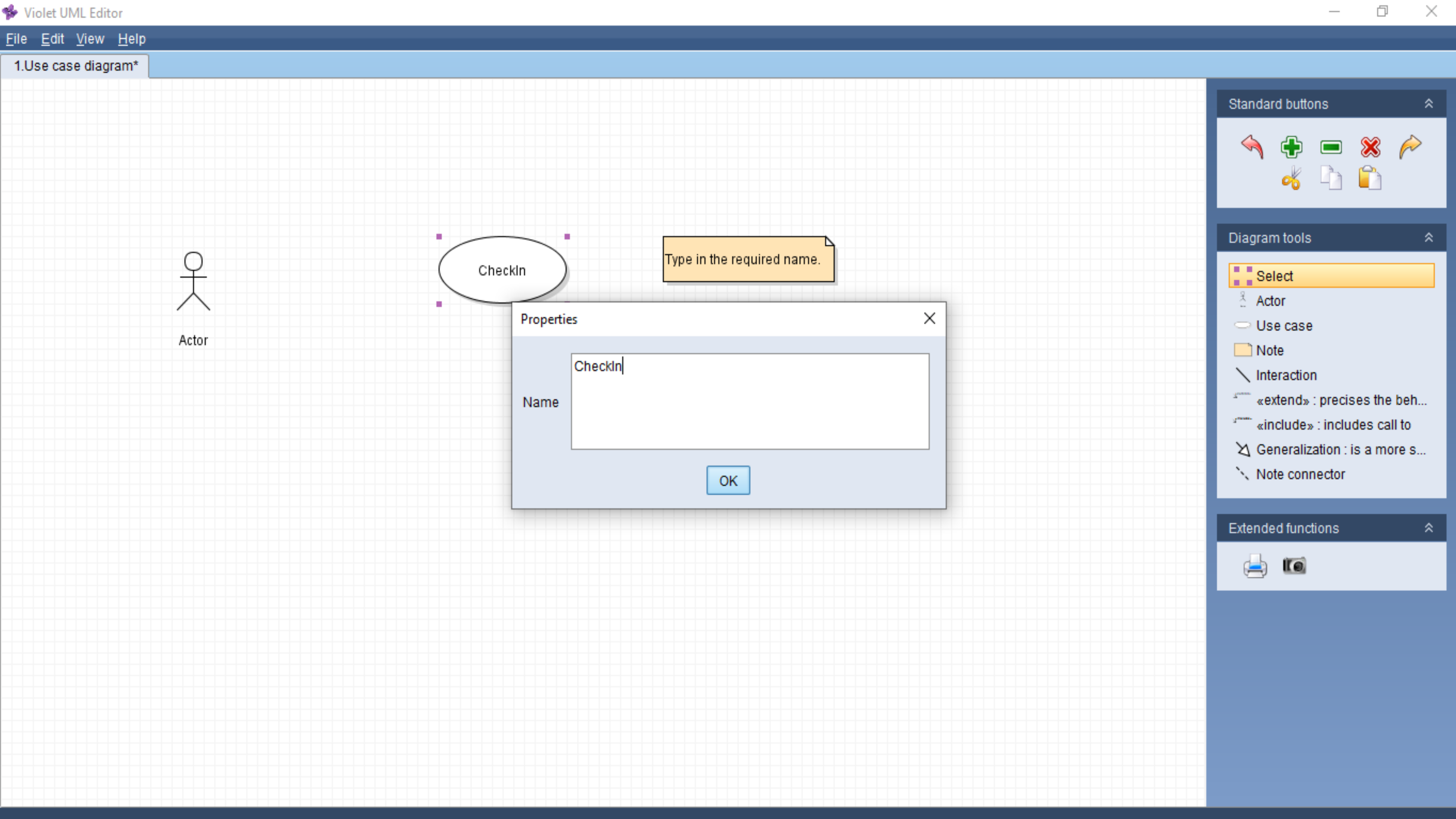
Undo Ctrl-Z
Redo Ctrl-Y
Properties Ctrl-Enter
Cut Ctrl-X
Copy Ctrl-C
Paste Ctrl-V
Delete Delete
Select All Ctrl-A

Standard buttons

Diagram tools

Select
Actor
Use case
Note
Interaction
«extend» : precises the beh...
«include» : includes call to
Generalization : is a more s...
Note connector

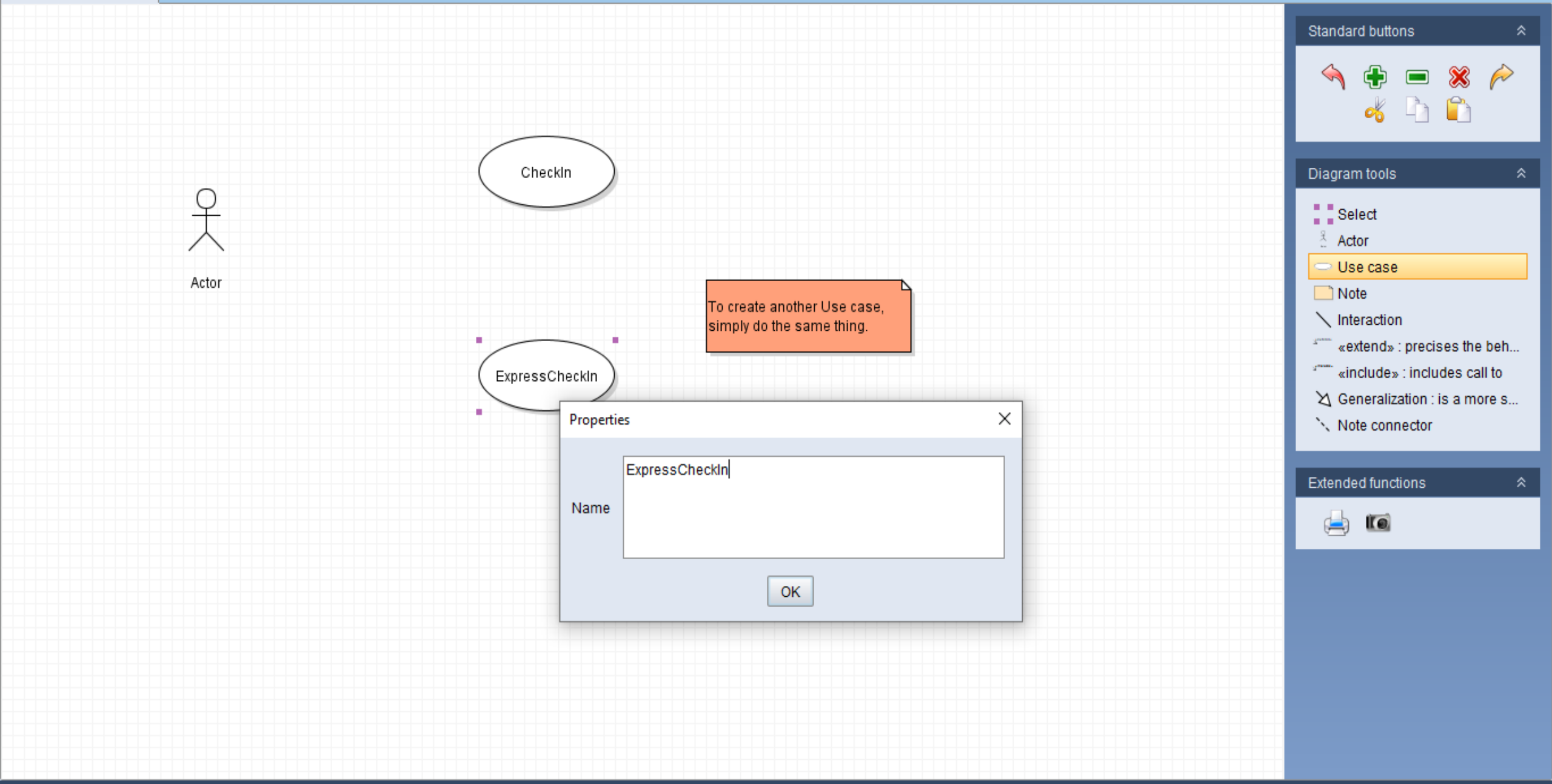
Extended functions


Violet UML Editor
File Edit View Help
1.Use case diagram*
Actor
CheckIn
Type in the required name.
Properties
Name
CheckIn
OK
Standard buttons
Diagram tools
Select
Actor
Use case
Note
Interaction
«extend» : precises the beh...
«include» : includes call to
Generalization : is a more s...
Note connector
Extended functions

Violet UML Editor

File Edit View Help

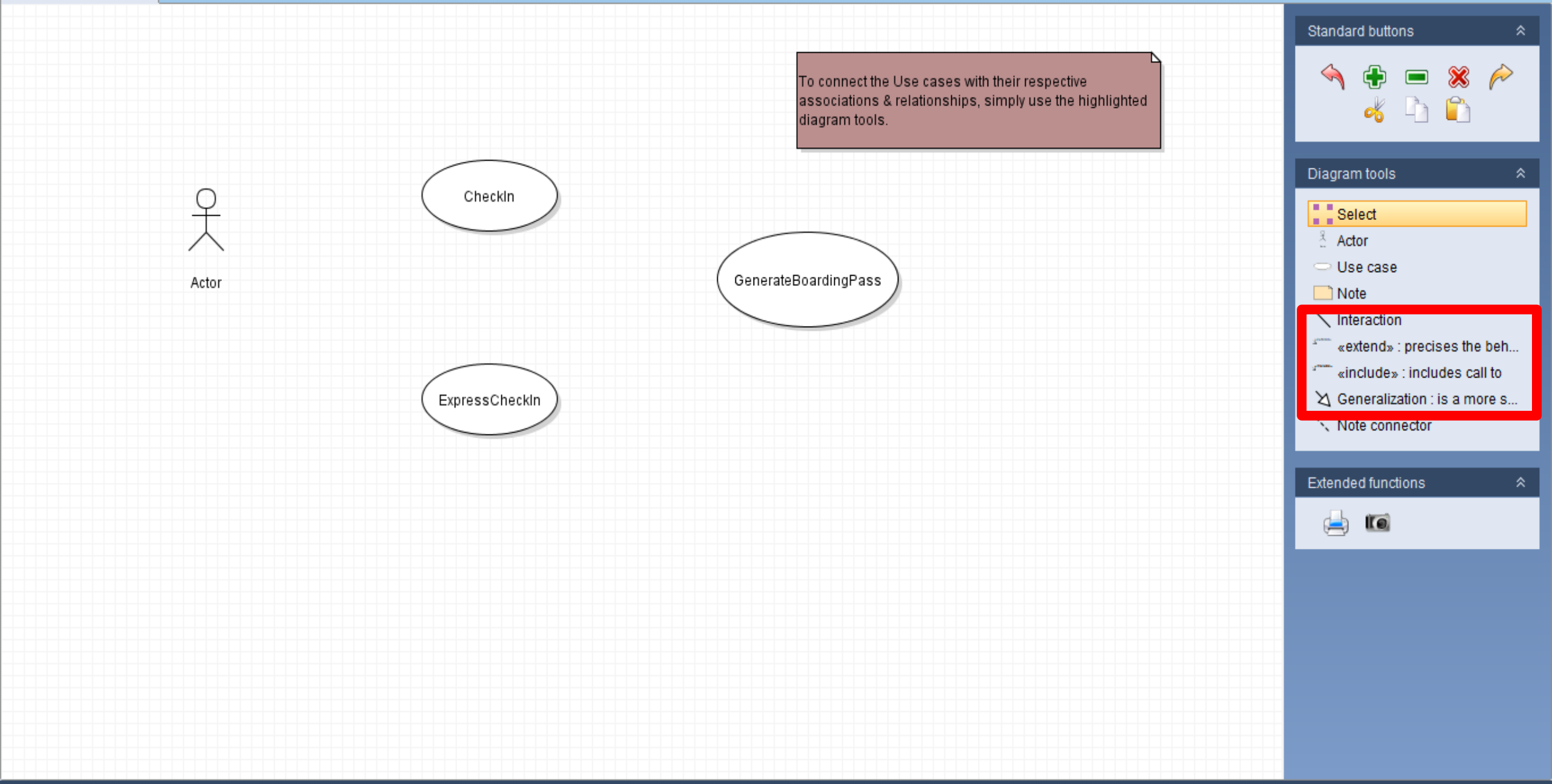
1.Use case diagram*



Violet UML Editor

File Edit View Help

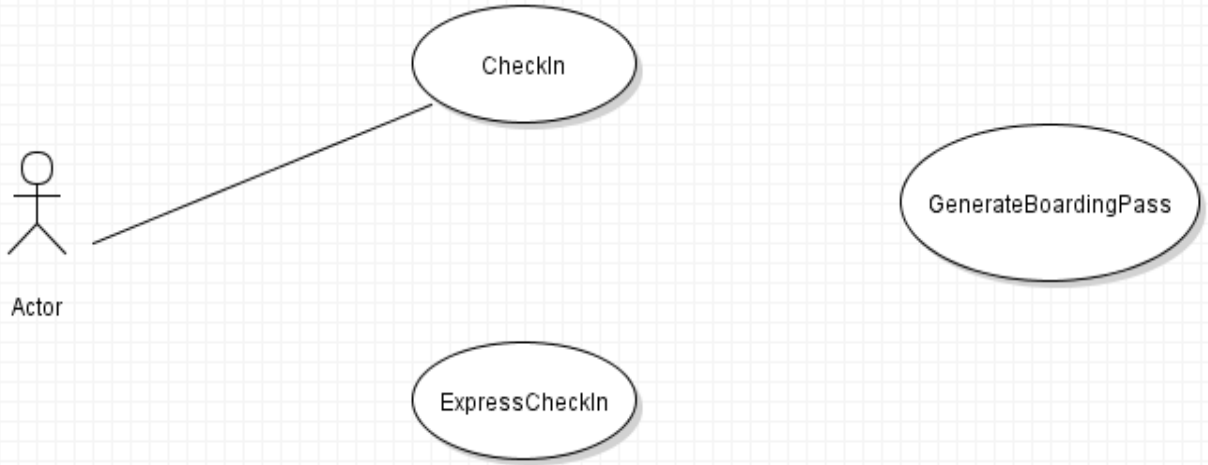
1.Use case diagram*



Violet UML Editor

File Edit View Help

1.Use case diagram*



Standard buttons

Diagram tools

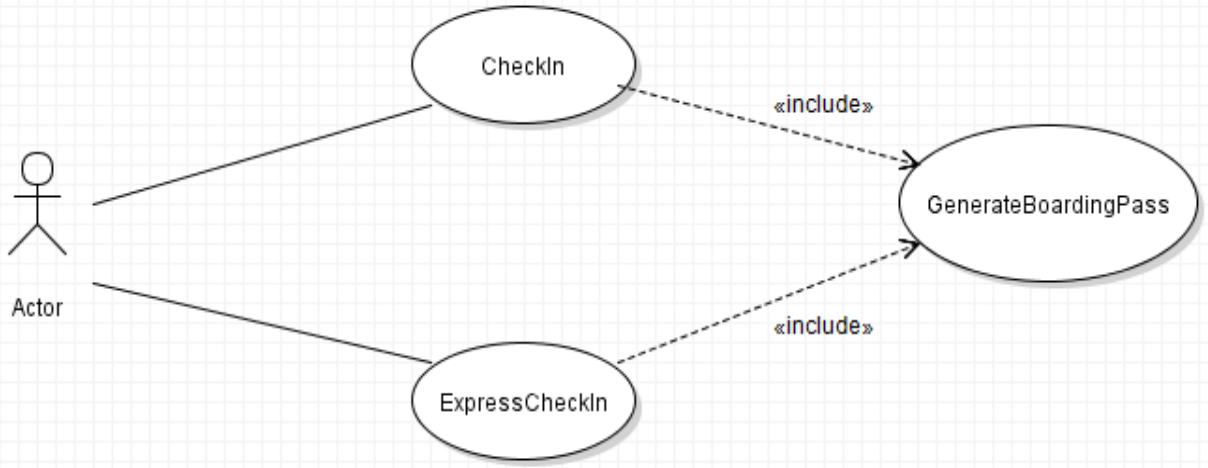
- Select
- Actor
- Use case
- Note
- Interaction
- «extend» : precises the beh...
- «include» : includes call to
- Generalization : is a more s...
- Note connector

Extended functions

Violet UML Editor

File Edit View Help

1.Use case diagram*



Standard buttons

Diagram tools

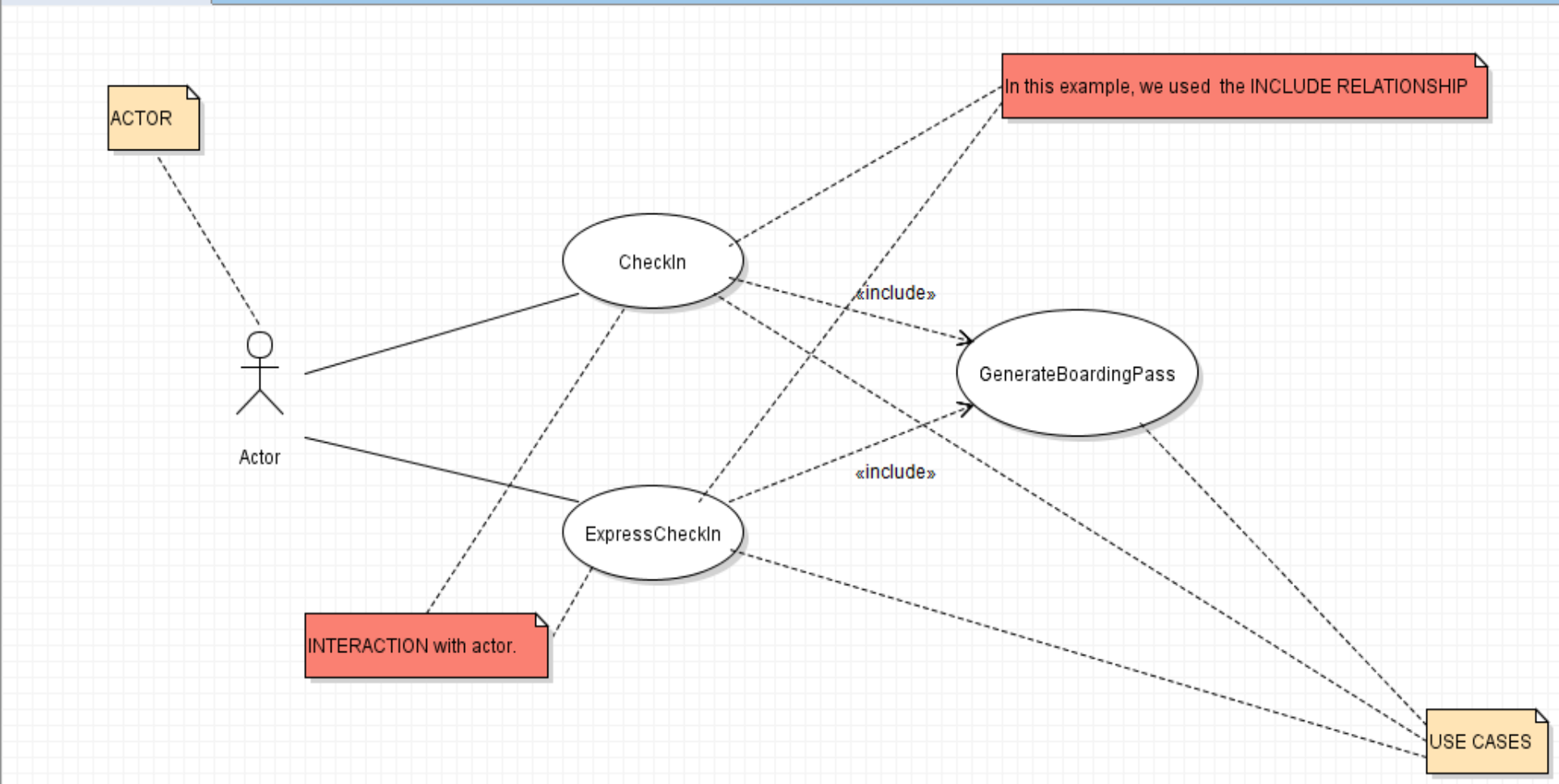
- Select
- Actor
- Use case
- Note
- Interaction
- «extend» : precises the beh...
- «include» : includes call to
- Generalization : is a more s...
- Note connector

Extended functions

Violet UML Editor

File Edit View Help

1.Use case diagram*



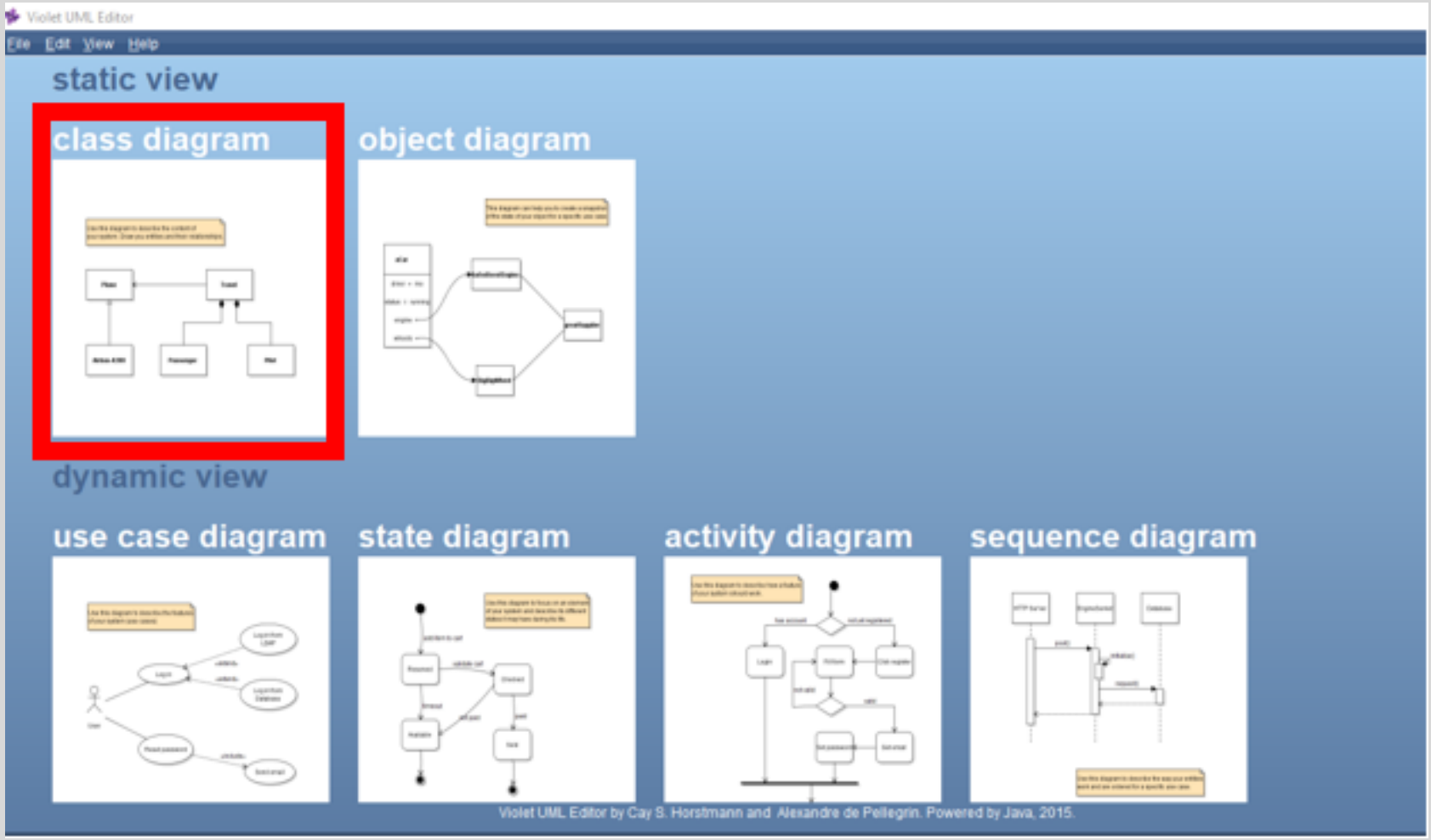
Standard buttons

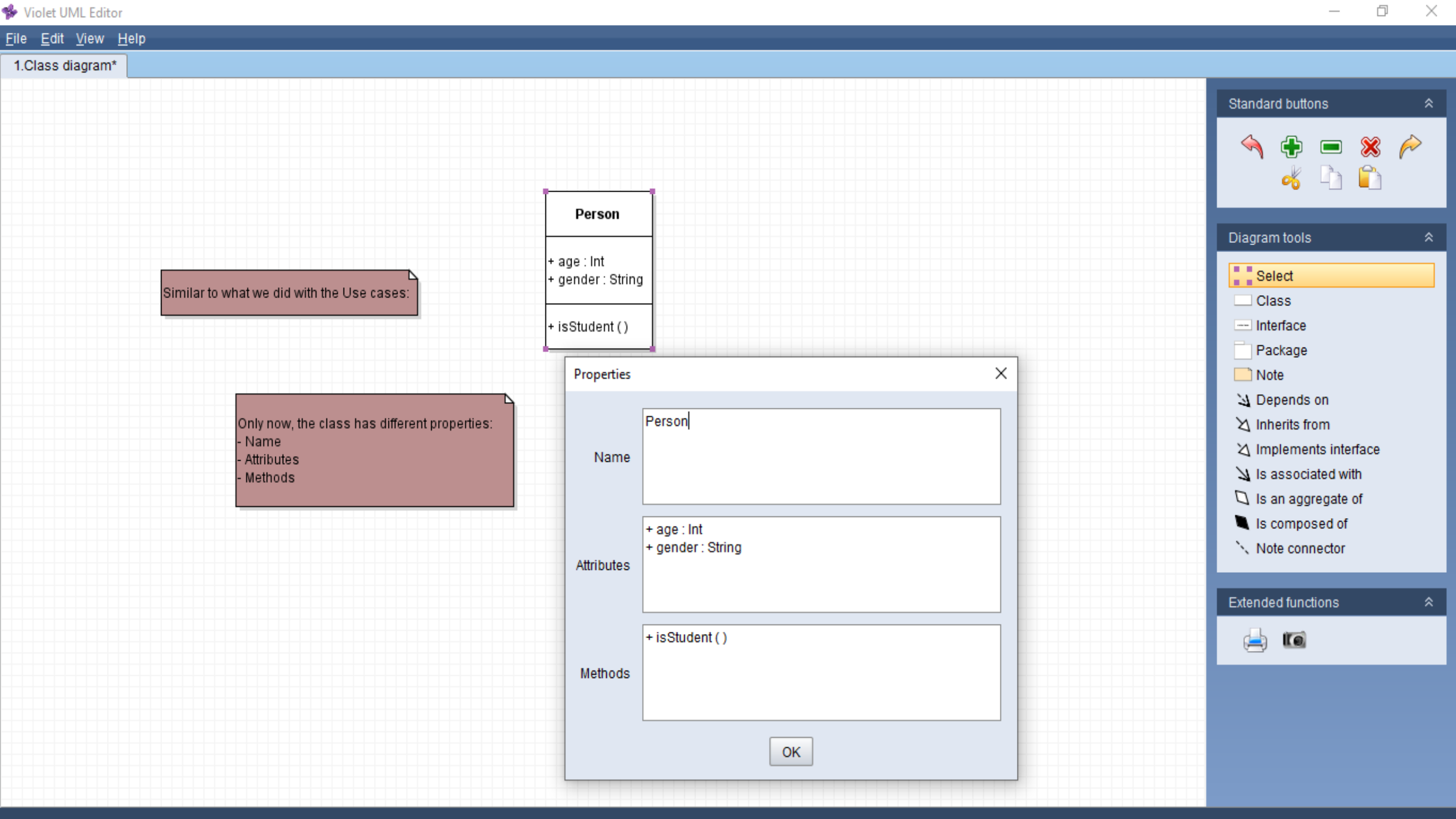
Diagram tools

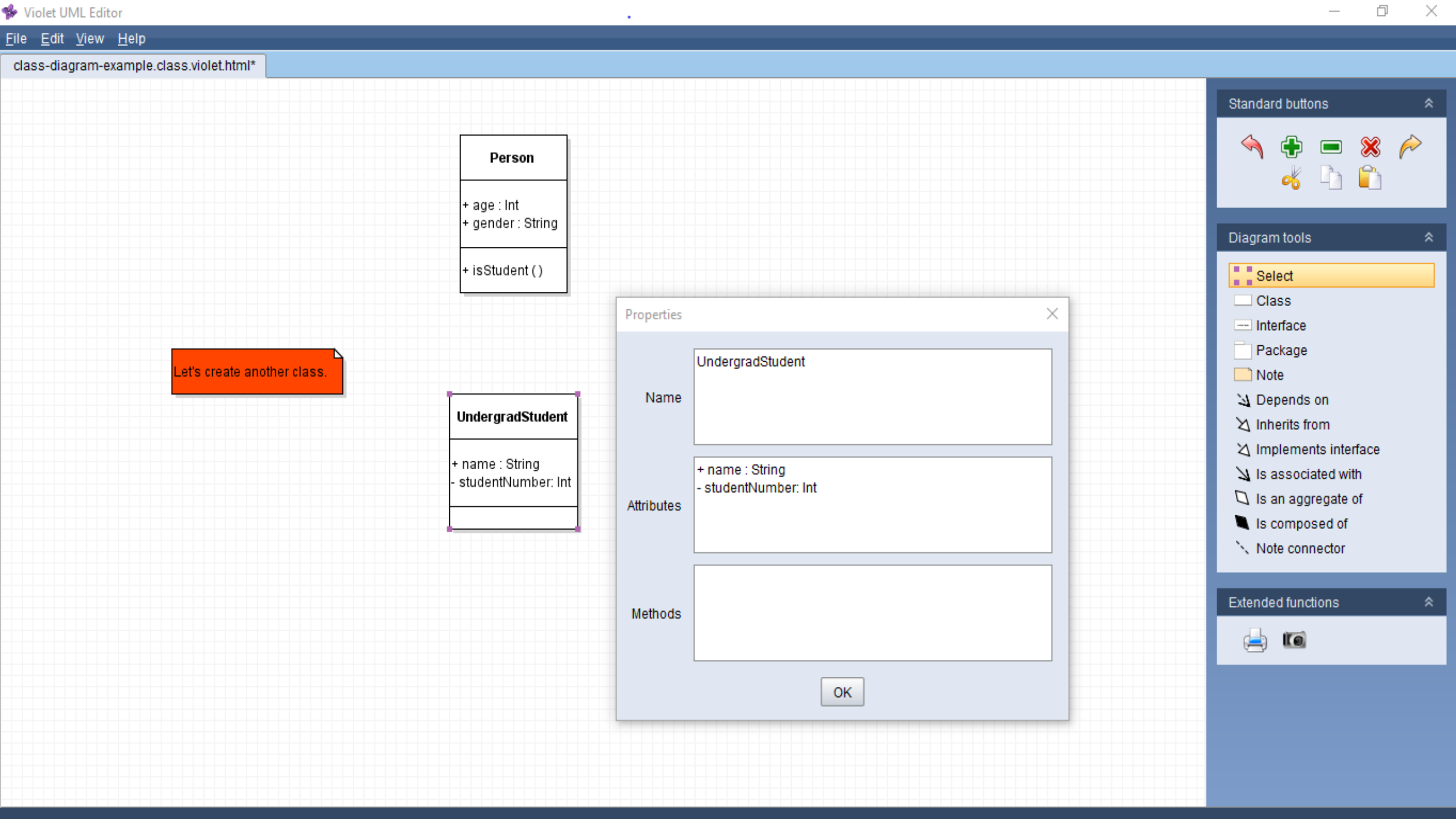
- Select
- Actor
- Use case
- Note
- Interaction
- «extend» : precises the beh...
- «include» : includes call to
- Generalization : is a more s...
- Note connector

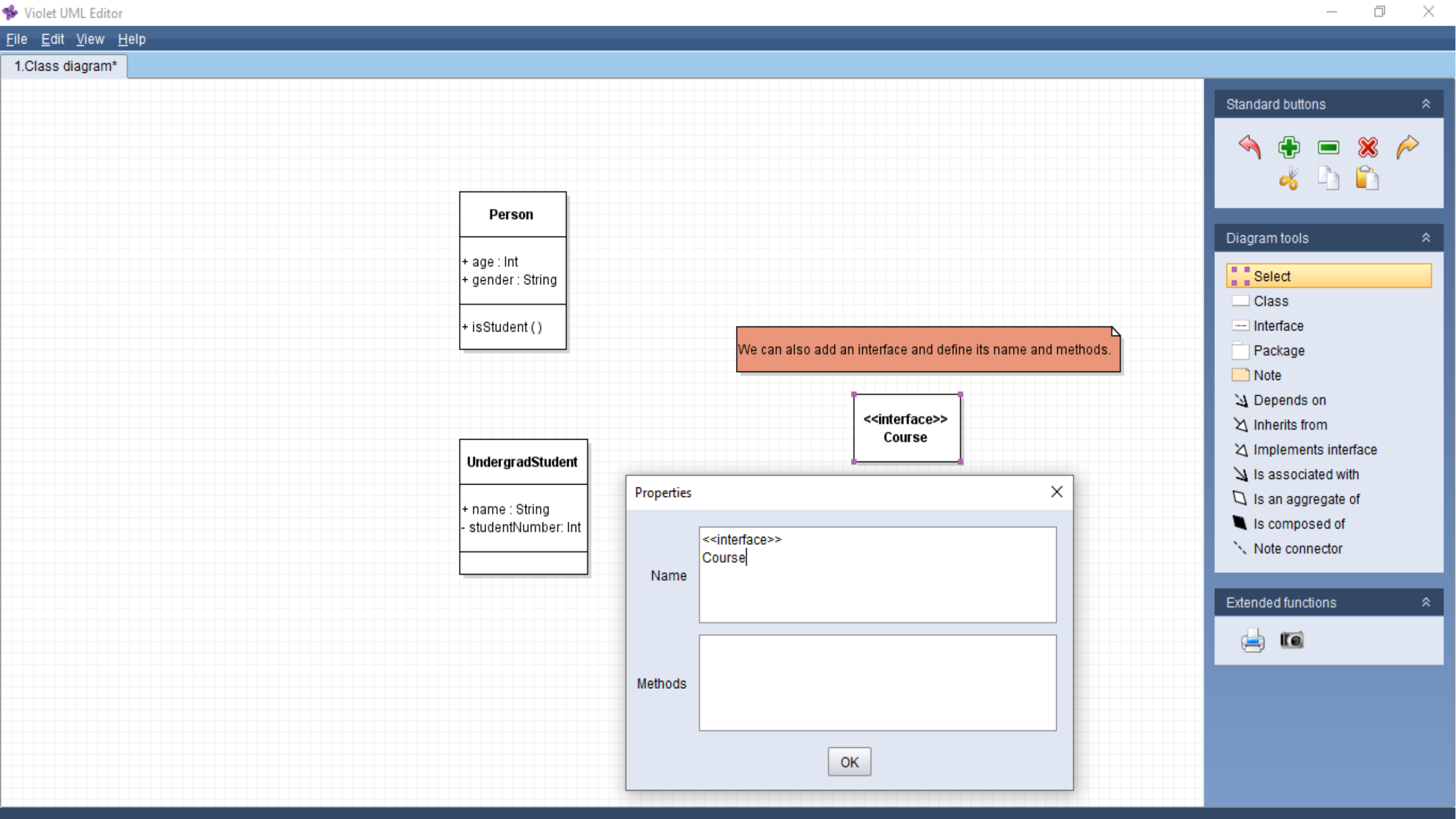
Extended functions

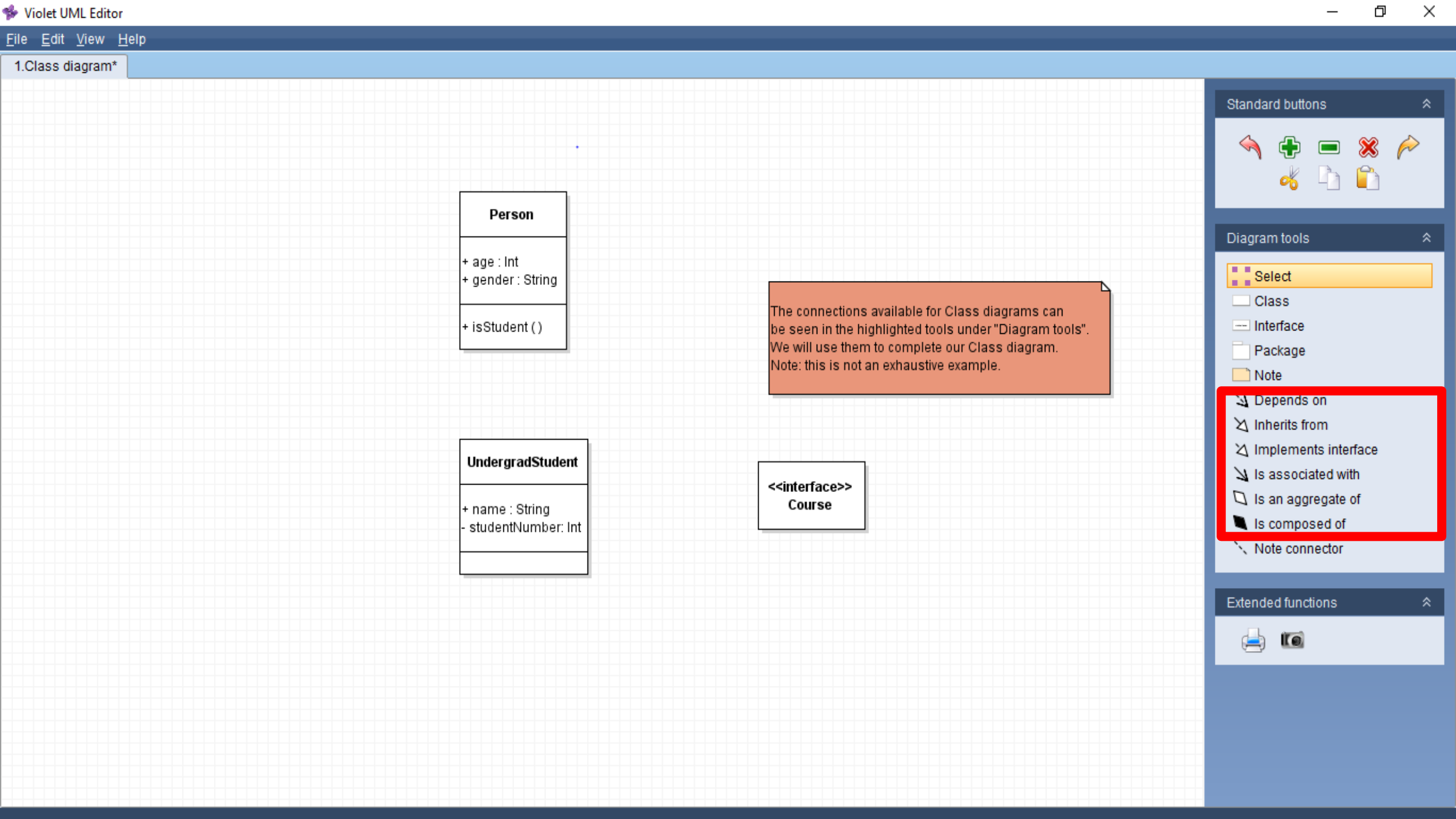
How to Create a Class Diagram

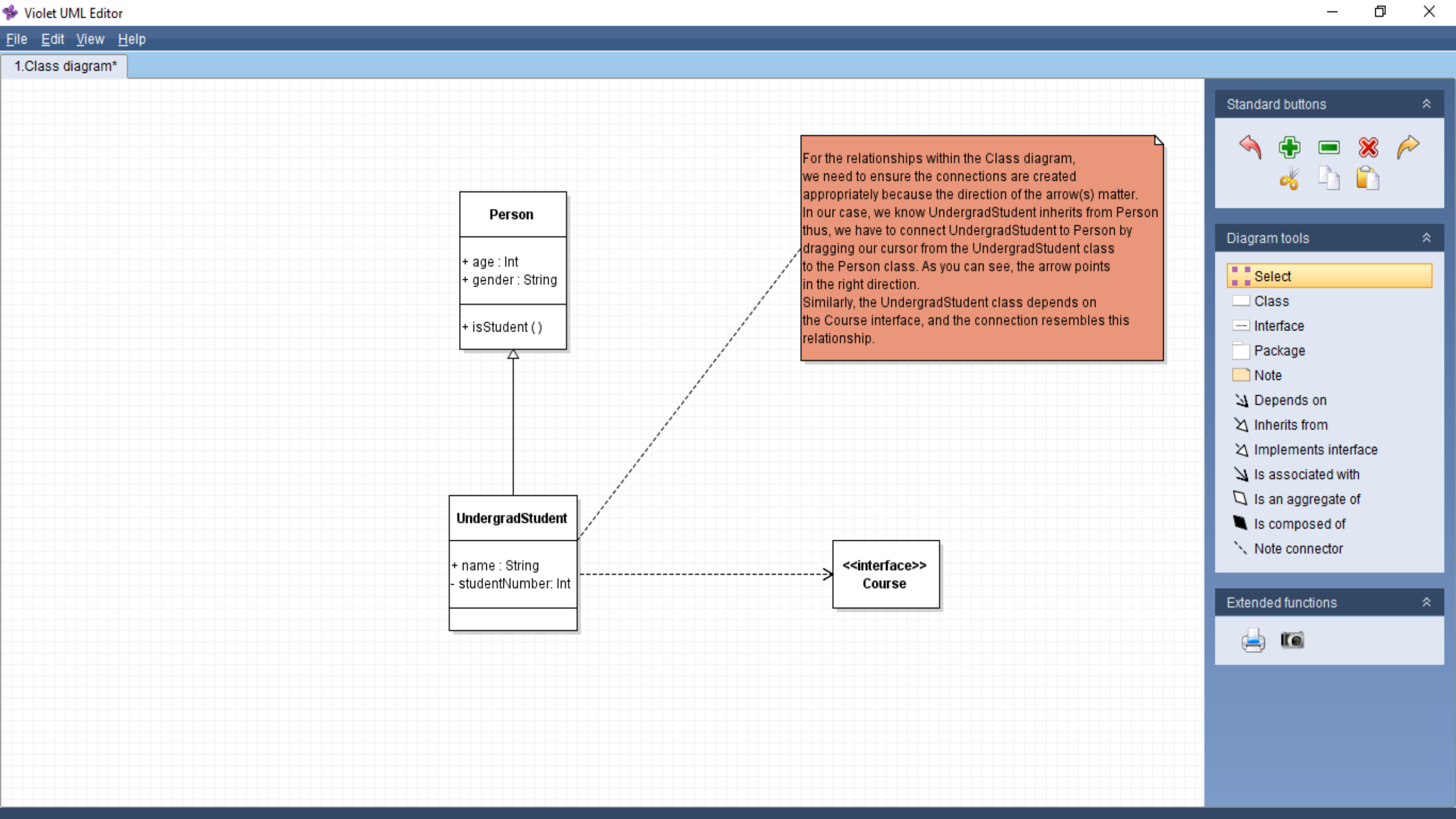


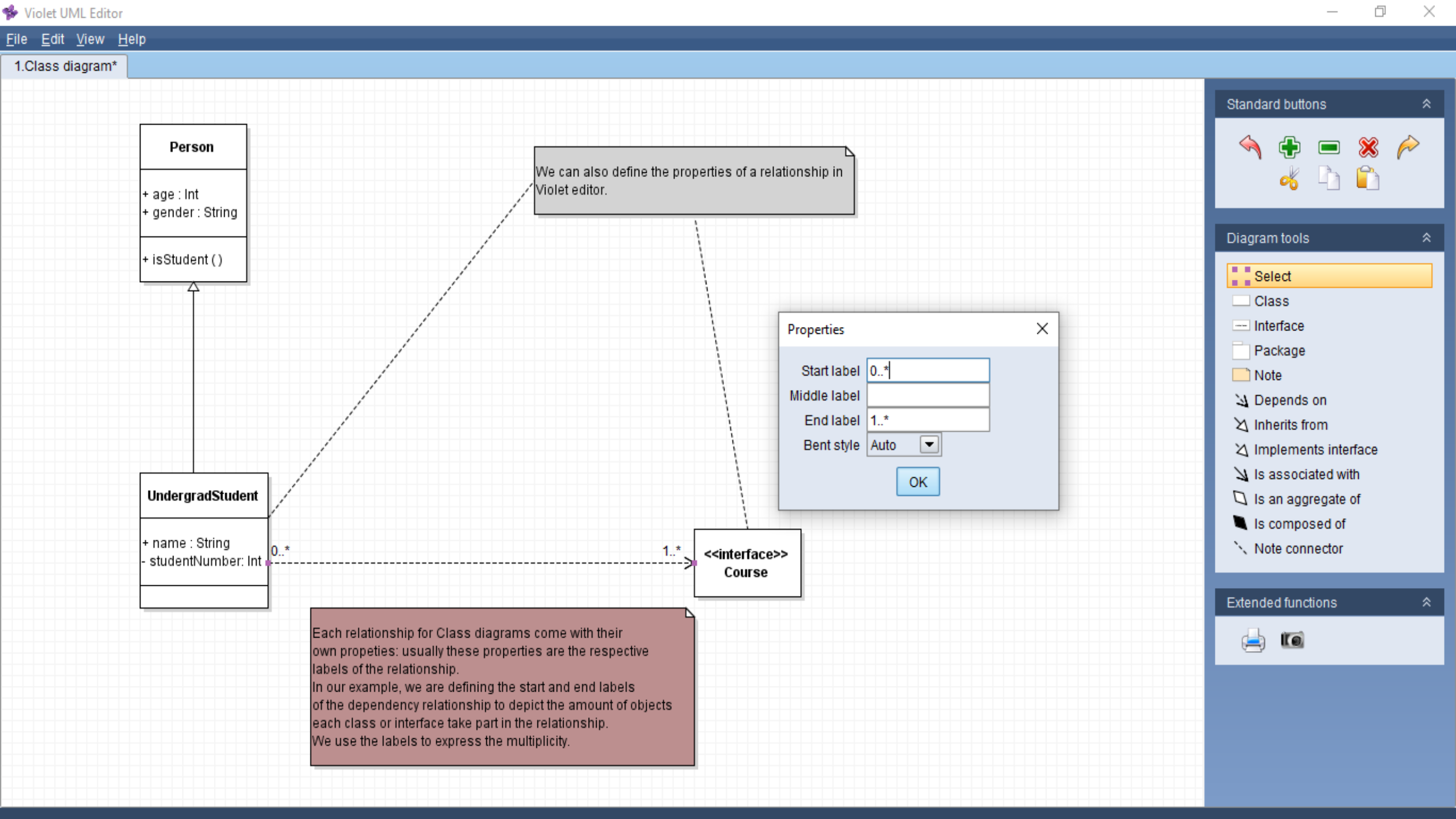

Violet UML Editor
File Edit View Help
1.Class diagram*
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions
Person
+ age : Int
+ gender : String
+ isStudent ( )
Similar to what we did with the Use cases:
Only now, the class has different properties:
- Name
- Attributes
- Methods
Properties
Name
Person
Attributes
+ age : Int
+ gender : String
Methods
+ isStudent ( )
OK


Violet UML Editor
File Edit View Help
class-diagram-example.class.violet.html*
Person
+ age : Int
+ gender : String
+ isStudent ( )
Let's create another class.
UndergradStudent
+ name : String
- studentNumber: Int
Properties
Name
UndergradStudent
Attributes
+ name : String
- studentNumber: Int
Methods
OK
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions


Violet UML Editor
File Edit View Help
1.Class diagram*
Person
+ age : Int
+ gender : String
+ isStudent ( )
We can also add an interface and define its name and methods.
<<interface>>
Course
UndergradStudent
+ name : String
- studentNumber: Int
Properties
Name
<<interface>>
Course
Methods
OK
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions


Violet UML Editor
File Edit View Help
1.Class diagram*
Person
+ age : Int
+ gender : String
+ isStudent ( )
The connections available for Class diagrams can be seen in the highlighted tools under "Diagram tools". We will use them to complete our Class diagram. Note: this is not an exhaustive example.
UndergradStudent
+ name : String
- studentNumber: Int
<<interface>>
Course
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions

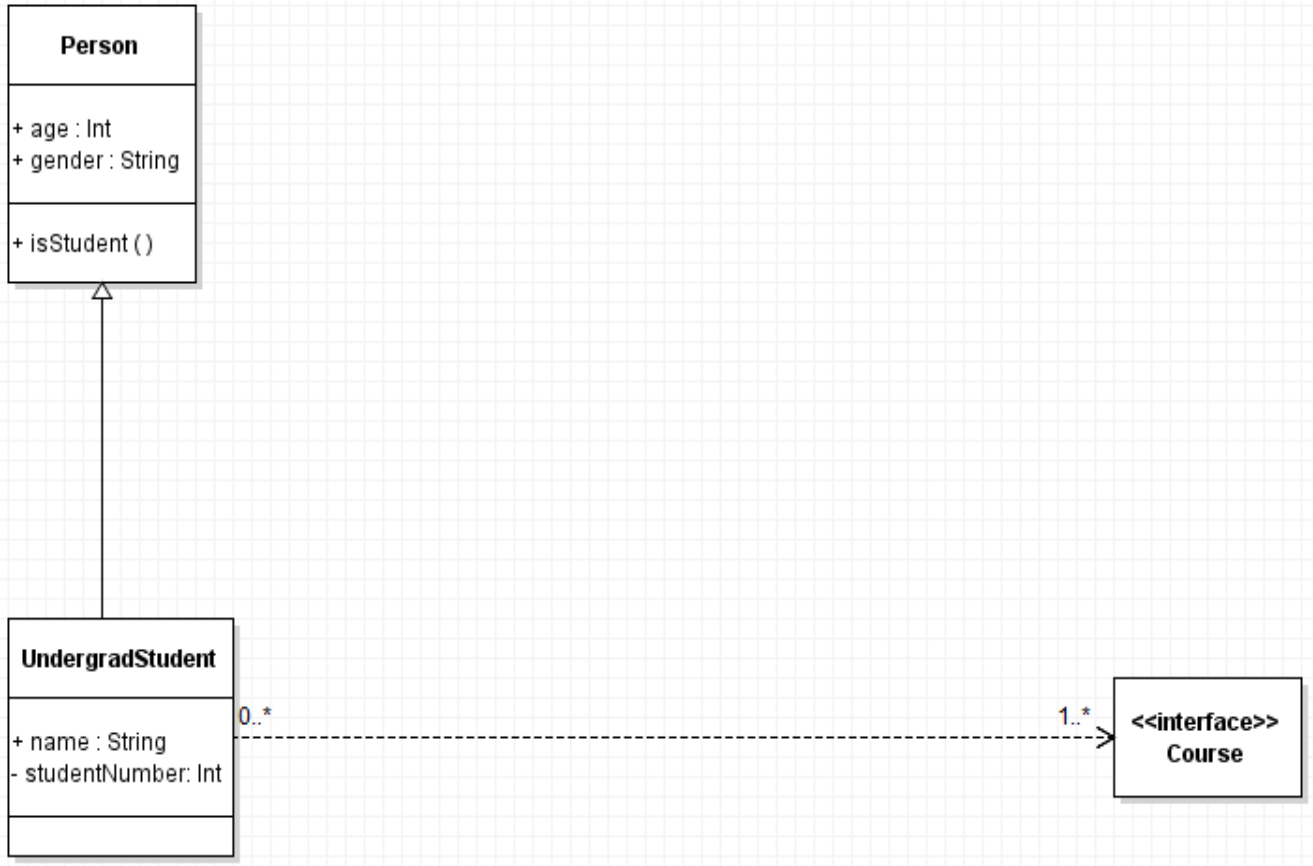

Violet UML Editor
File Edit View Help
1.Class diagram*
Person
+ age : Int
+ gender : String
+ isStudent ( )
For the relationships within the Class diagram,
we need to ensure the connections are created
appropriately because the direction of the arrow(s) matter.
In our case, we know UndergradStudent inherits from Person
thus, we have to connect UndergradStudent to Person by
dragging our cursor from the UndergradStudent class
to the Person class. As you can see, the arrow points
in the right direction.
Similarly, the UndergradStudent class depends on
the Course interface, and the connection resembles this
relationship.
UndergradStudent
+ name : String
- studentNumber: Int
<<interface>>
Course
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions


Violet UML Editor
File Edit View Help
1.Class diagram*
Person
+ age : Int
+ gender : String
+ isStudent ( )
We can also define the properties of a relationship in Violet editor.
Properties
Start label 0..*
Middle label
End label 1..*
Bent style Auto
OK
UndergradStudent
+ name : String
- studentNumber: Int
0..*
1..*
<<interface>>
Course
Each relationship for Class diagrams come with their own propeties: usually these properties are the respective labels of the relationship.
In our example, we are defining the start and end labels of the dependency relationship to depict the amount of objects each class or interface take part in the relationship.
We use the labels to express the multiplicity.
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions

Violet UML Editor

File Edit View Help

1.Class diagram*

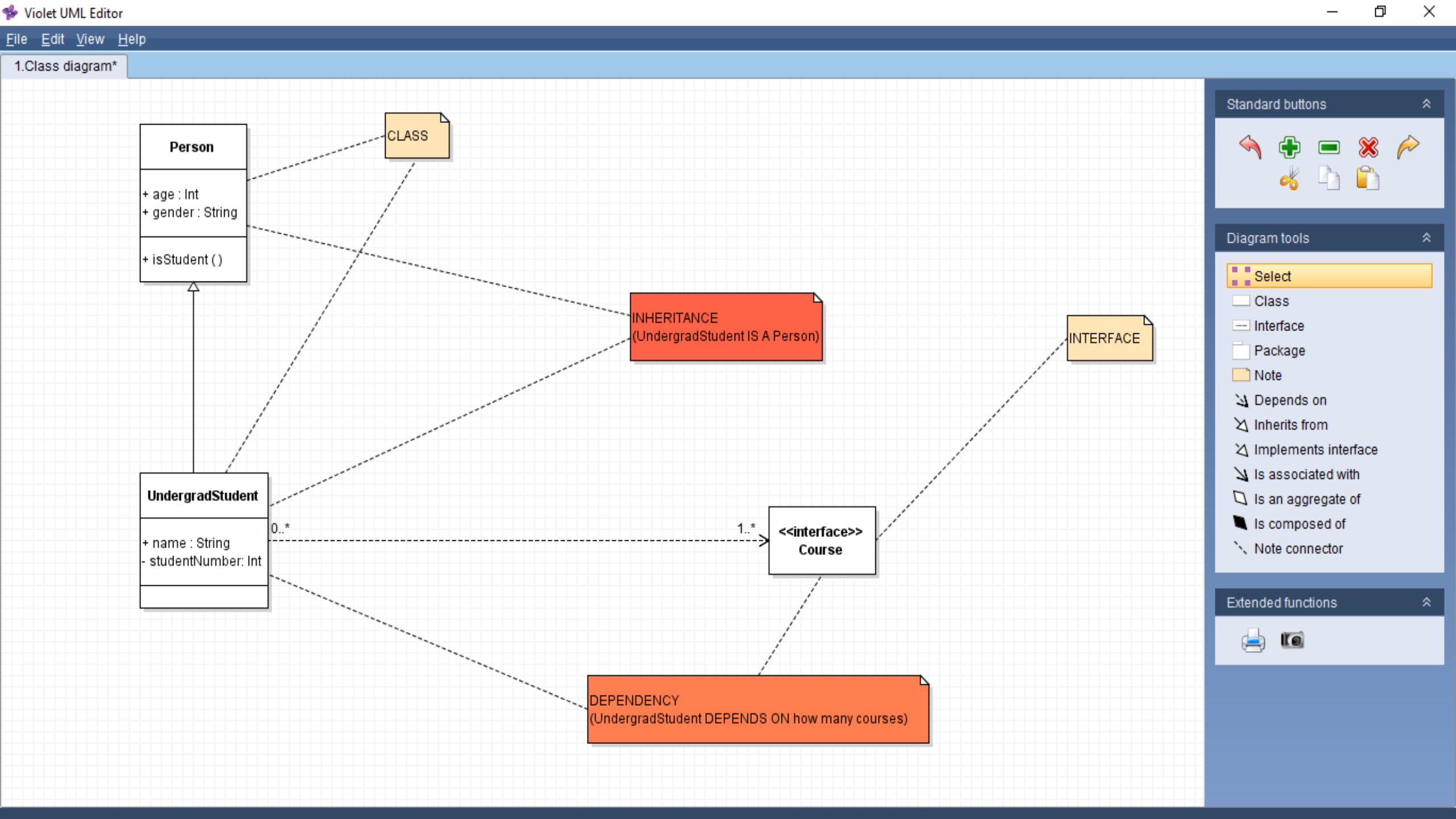


Standard buttons

Diagram tools

- Select
- Class
- Interface
- Package
- Note
- Depends on
- Inherits from
- Implements interface
- Is associated with
- Is an aggregate of
- Is composed of
- Note connector

Extended functions


Violet UML Editor
File Edit View Help
1.Class diagram*
Person
+ age : Int
+ gender : String
+ isStudent ( )
CLASS
INHERITANCE
(UndergradStudent IS A Person)
INTERFACE
UndergradStudent
+ name : String
- studentNumber: Int
0..*
1..*
<<interface>>
Course
DEPENDENCY
(UndergradStudent DEPENDS ON how many courses)
Standard buttons
Diagram tools
Select
Class
Interface
Package
Note
Depends on
Inherits from
Implements interface
Is associated with
Is an aggregate of
Is composed of
Note connector
Extended functions

Support

Visit the official Violet editor documentation for help:
http://alexdp.free.fr/violetumleditor/page.php

You can also visit the official GitHub repo at:
https://github.com/violetumleditor/violetumleditor.