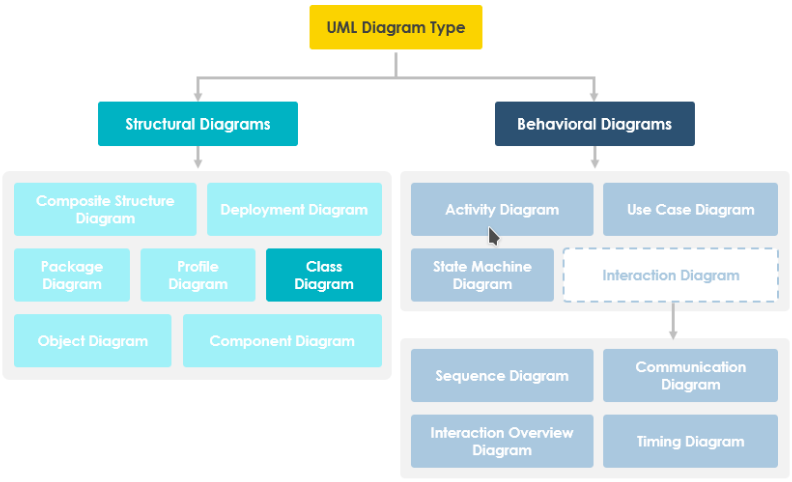
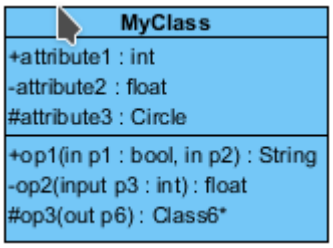
Diagrama de Classe



Purpose of Class Diagrams



Shows static structure of classifiers in a system

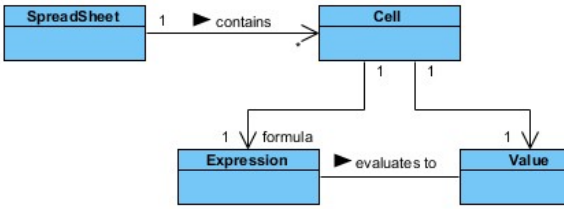
Diagram provides a basic notation for other structure diagrams prescribed by UML

Helpful for developers and other team members too

Business Analysts can use class diagrams to model systems from a business perspective

Relationship Names

- Names of relationships are written in the middle of the association line.
- Good relation names make sense when you read them out loud:
  - "Every spreadsheet **contains** some number of cells",
  - "an expression **evaluates to** a value"
- They often have a **small arrowhead to show the direction** in which direction to read the relationship, e.g., expressions evaluate to values, but values do not evaluate to expressions.



Navigability

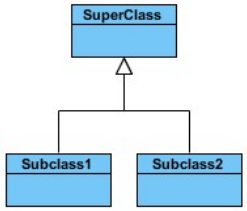
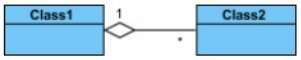
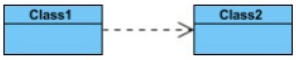
The arrows indicate whether, given one instance participating in a relationship, it is possible to determine the instances of the other class that are related to it.

The diagram above suggests that,

- Given a spreadsheet, we can locate all of the cells that it contains, but that
  - we cannot determine from a cell in what spreadsheet it is contained.
- Given a cell, we can obtain the related expression and value, but
  - given a value (or expression) we cannot find the cell of which those are attributes.

Relationship - Roles

- A role is a directional purpose of an association.
- Roles are written at the ends of an association line and describe the purpose played by that class in the relationship.
  - E.g., A cell is related to an expression. The nature of the relationship is that the expression is the **formula** of the cell.

Relationship Type	Graphical Representation
Inheritance (or Generalization): • Represents an "is-a" relationship. • An abstract class name is shown in italics. • SubClass1 and SubClass2 are specializations of Super Class. • A solid line with a hollow arrowhead that point from the child to the parent class	
Simple Association: • A structural link between two peer classes. • There is an association between Class1 and Class2 • A solid line connecting two classes	
Aggregation: A special type of association. It represents a "part of" relationship. • Class2 is part of Class1. • Many instances (denoted by the *) of Class2 can be associated with Class1. • Objects of Class1 and Class2 have separate lifetimes. • A solid line with an unfilled diamond at the association end connected to the class of composite	
Composition: A special type of aggregation where parts are destroyed when the whole is destroyed. • Objects of Class2 live and die with Class1. • Class2 cannot stand by itself. • A solid line with a filled diamond at the association connected to the class of composite	
Dependency: • Exists between two classes if the changes to the definition of one may cause changes to the other (but not the other way around). • Class1 depends on Class2 • A dashed line with an open arrow	

- + denotes public attributes or operations
- - denotes private attributes or operations
- # denotes protected attributes or operations
- ~ denotes package attributes or operations

Access Right	public (+)	private (-)	protected (#)	Package (~)
Members of the same class	yes	yes	yes	yes
Members of derived classes	yes	no	yes	yes
Members of any other class	yes	no	no	in same package