Domain Model Diagramming

Why do we use different objects

- its a good organisational tool (seperation of concerns)
- S**eparation of concerns** (**SoC**) is a design principle for separating a computer program into distinct sections. Each section addresses a separate *concern*, a set of information that affects the code of a computer program
- makes your programs easier to understand

How to break user stories into a diagram

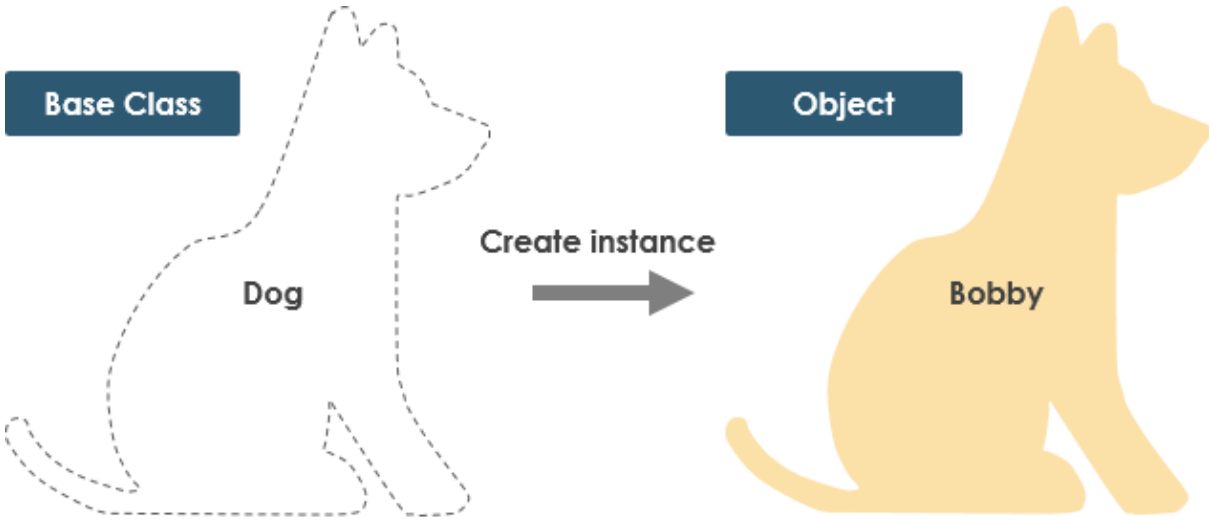
- Noun (attributes) - instances of the class
- Verb (Methods)

UML Class Diagrams

A Class is a blueprint for an object. Objects and classes go hand in hand, we use classes to create objects. So a class describes what an object will be, but it isn't the object itself.

In fact, classes describe the type of objects, while objects are usable instances of classes. Each Object was built from the same set of blueprints and therefore contains the same components (properties and methods). The standard meaning is that an object is an instance of a class and object - Objects have states and behaviors.

e.g

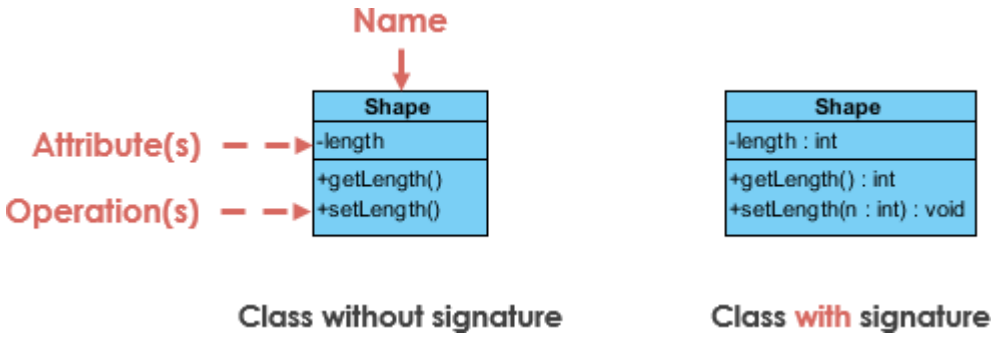


Properties	Methods
Color	Sit
Eye Color	Lay Down
Height	Shake
Length	Come
Weight	

Property Values	Methods
Color: Yellow	Sit
Eye Color: Brown	Lay Down
Height: 17 in	Shake
Length: 35 in	Come
Weight: 24 pounds	

UML Class Notation

A class represent a concept which encapsulates state (**attributes**) and behaviour (**operations**). Each attribute has a type. Each **operation** has a **signature**. *The class name is the **only mandatory information***.



Class Name:

- The name of the class appears in the first partition.

Class Attributes:

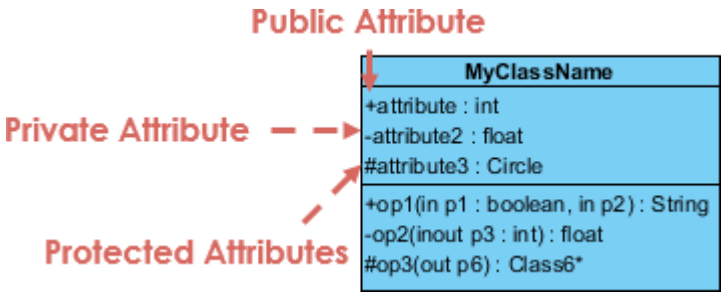
- Attributes are shown in the second partition.
- The attribute type is shown after the colon.
- Attributes map onto member variables (data members) in code.

Class Operations (Methods):

- Operations are shown in the third partition. They are services the class provides.
- The return type of a method is shown after the colon at the end of the method signature.
- The return type of method parameters are shown after the colon following the parameter name. Operations map onto class methods in code

Class Visibility

The +, - and # symbols before an attribute and operation name in a class denote the visibility of the attribute and operation.



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

Relationships between classes

UML is not just about pretty pictures. If used correctly, UML precisely conveys how code should be implemented from diagrams. If precisely interpreted, the implemented code will correctly reflect the intent of the designer. Can you describe what each of the relationships mean relative to your target programming language shown in the Figure below?

If you can't yet recognize them, no problem this section is meant to help you to understand UML class relationships. A class may be involved in one or more relationships with other classes. A relationship can be one of the following types: