UML Cheatsheet

UML is a mechanism for communication. It is intended to convey the meaningful parts of your application. Include the data which will help someone understand your code, not everything must be included (unless it's an exam, then include everything).

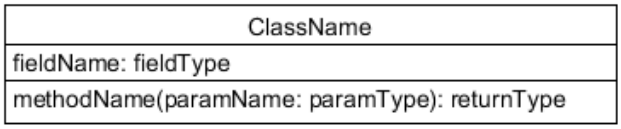
Representing Classes

The basic method for representing fields and methods is:

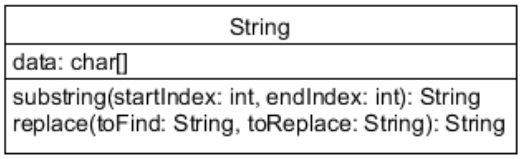
Fields:	**Methods:**
name: Type	name(paramName1: Type, paramName2: Type): returnType

Below is a general template for representing classes, and a small representation of the String class. If you're representing an interface, put <<interface>> above the class name, for an abstract class, put the name in *italics*.

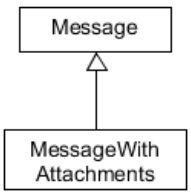
Template:



String Representation:

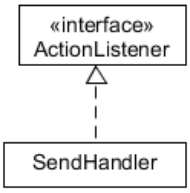


Arrows:



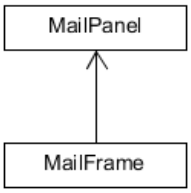
Inheritance
(is-a)

MessageWithAttachment is-a Message



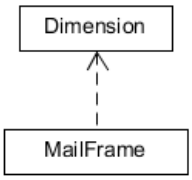
Interface Implementation
(is-a)

SendHandler is-an ActionListener



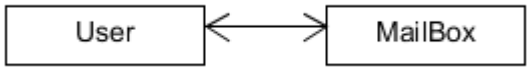
Association
(has-a field,
or has a field that uses the type as a type parameter)

MailFrame has-a MailPanel field

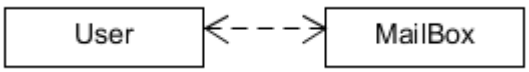


Dependency
(depends on, or uses but is not a field)

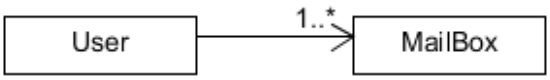
MailFrame depends-on Dimension



Two-Way Association
User has a MailBox field AND
MailBox has a User field



Two-Way Dependency
User depends on MailBox AND
MailBox depends on User
(ex. A MailBox method takes a User as a parameter, or vice versa)



Cardinality
How many items are in a relationship
Say: "A User has one-to-many MailBoxes"