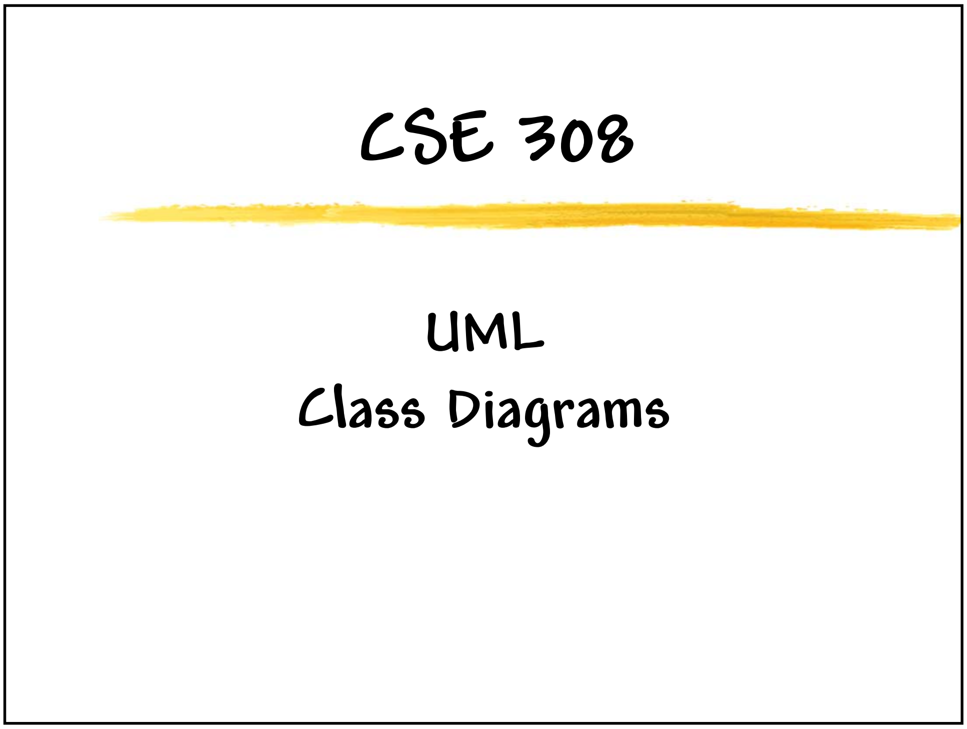
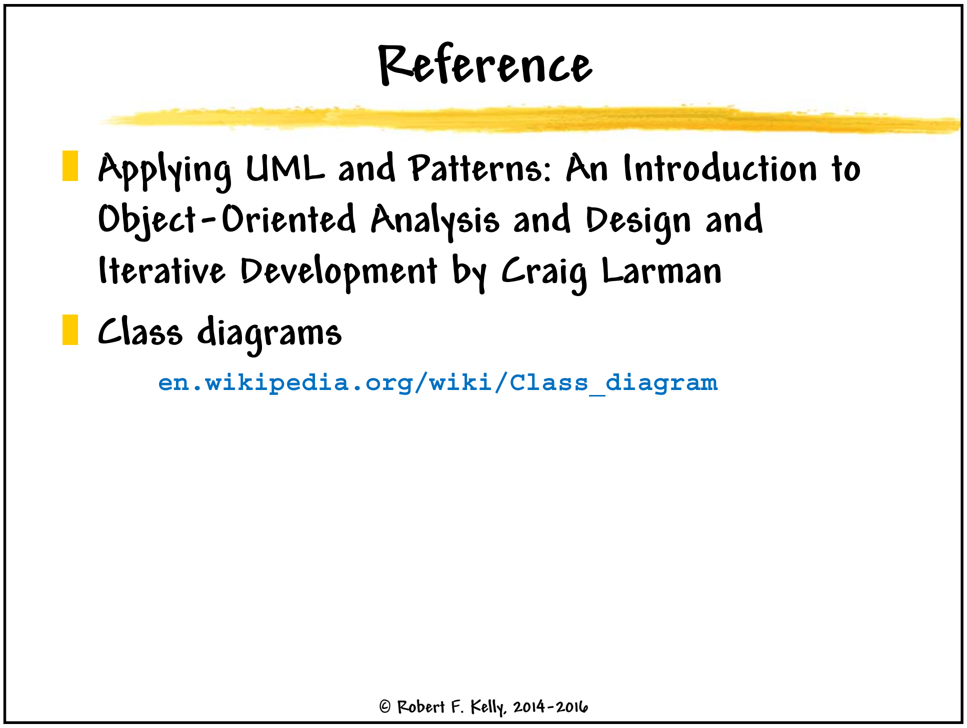


CSE 308

UML
Class Diagrams



Reference

- Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development by Craig Larman
- Class diagrams

  en.wikipedia.org/wiki/Class_diagram

Class Diagram

- Goal is to convey information about the static structure of your application domain
- Best if build iteratively
- Frequently, it is not a precise representation of the software structure
- Conventions you follow are largely tool and software organization based

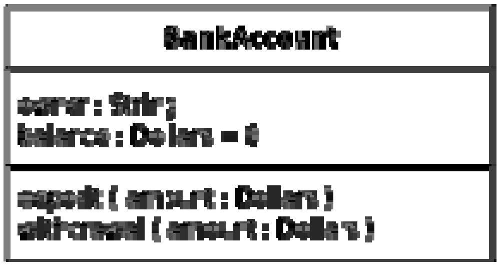
Note that Violet is a decent UML tool (and is free), but it is not quite ready for prime-time

Class Notation

- A class descriptor contains three parts
  - Name
  - Attributes
  - Methods

Class name is upper camel case
Attribute and method names are lower camel case

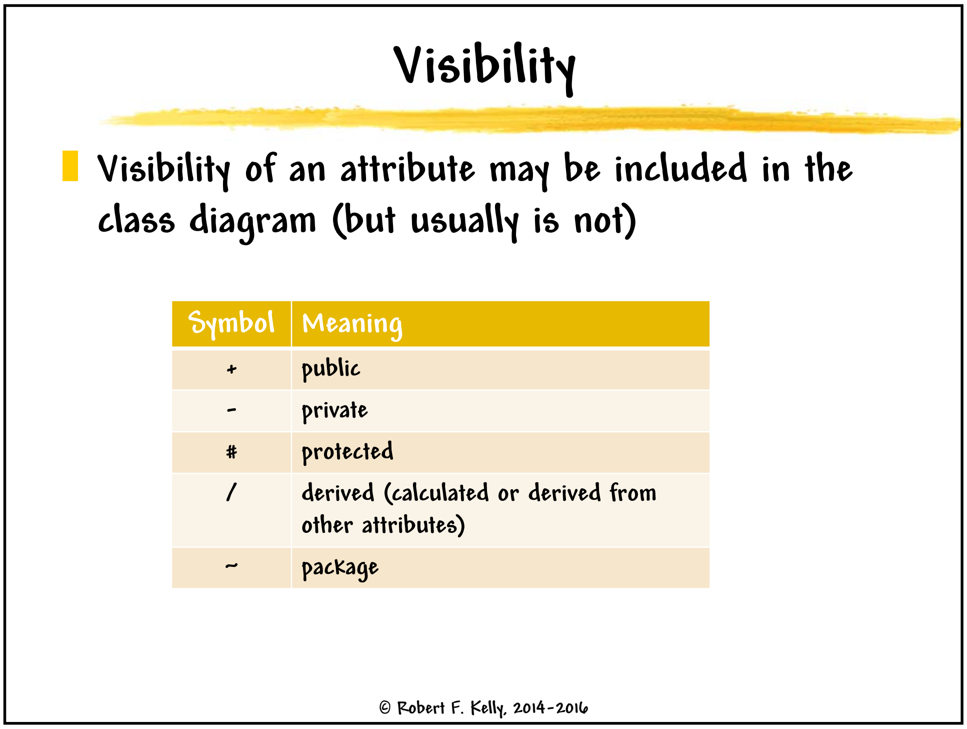
Attribute type info is not required, and will probably not be in your first pass at a class diagram



BankAccount

owner : String
balance : Dollars = 0

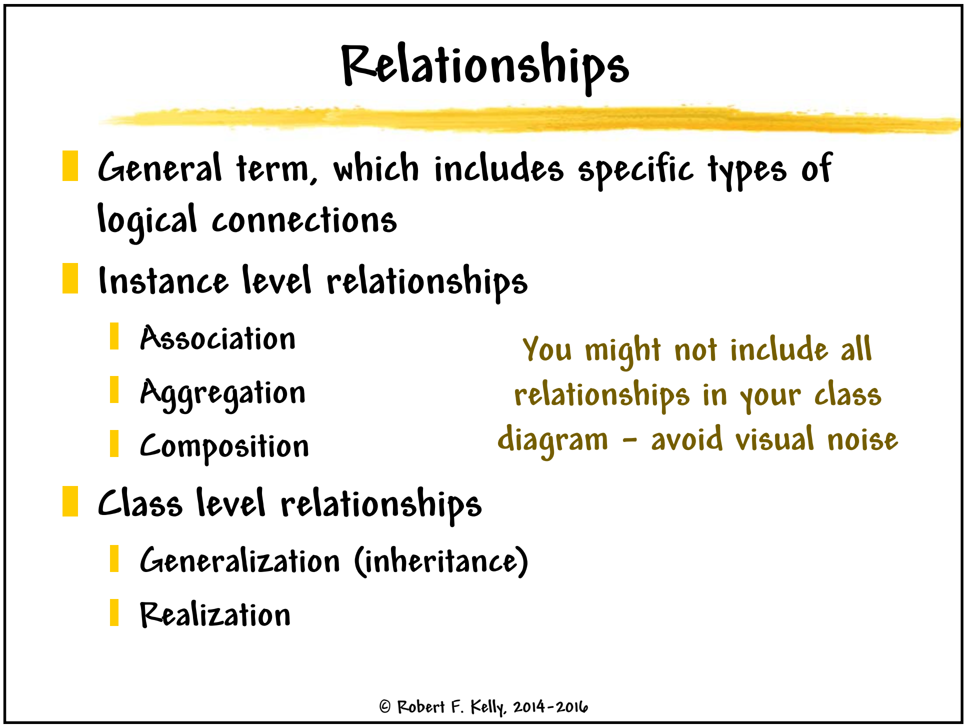
deposit (amount : Dollars)
withdrawal (amount : Dollars)



Visibility

- Visibility of an attribute may be included in the class diagram (but usually is not)

Symbol	Meaning
+	public
-	private
#	protected
/	derived (calculated or derived from other attributes)
~	package



Relationships

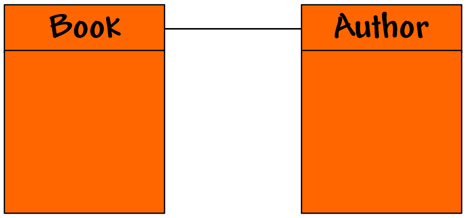
- General term, which includes specific types of logical connections
- Instance level relationships
  - Association
  - Aggregation
  - Composition
- Class level relationships
  - Generalization (inheritance)
  - Realization

You might not include all relationships in your class diagram - avoid visual noise

Association

- Not a statement about data flows, key relationships, etc.
- At least one class makes reference to the other
- Used when the relationship is not transient
- Options
  - Named
  - Multiplicity
  - Arrowhead (showing ownership)
  - Other properties

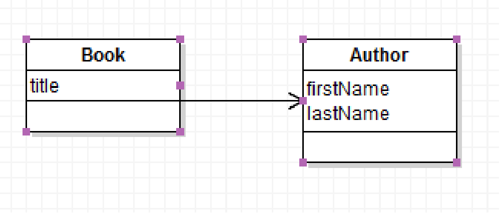
A Book has an Author



Association Arrowhead

- Note that Violet uses an arrowhead connector (not the pure association)
- Usually means that the class at the tail of the arrow has an attribute of the type (Class name) shown at the end of the arrow
- Domain UML associations do not use arrowheads (SW UML does)

You can use text at the end of the arrow to show the attribute name



How to Express Attributes

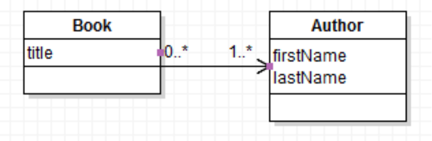
- Choices
  - Attribute text
  - Association lines
- Guidelines
  - Attribute text for primitive types
  - Attribute text for library class types
  - Association lines for class types

Multiplicity

- Multiplicity in an association indicates the number of instances
- Multiplicity symbol is often tool-related

Symbol	Instances
0..1	No instances or one instance
1	Exactly one instance
0..*	Zero or more instances
1..*	One or more instances
3,5,8	Exactly 3, 5, or 8

Violet allows you to add labels to an association



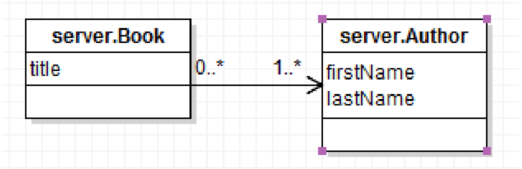
Note, you might not include related classes in attributes

Methods

- Typically, developed later in the design phase
- Sequence diagrams are very helpful in determining needed methods
- No need to include obvious methods (e.g., getters and setters)
- Class diagram might include parameters and return type

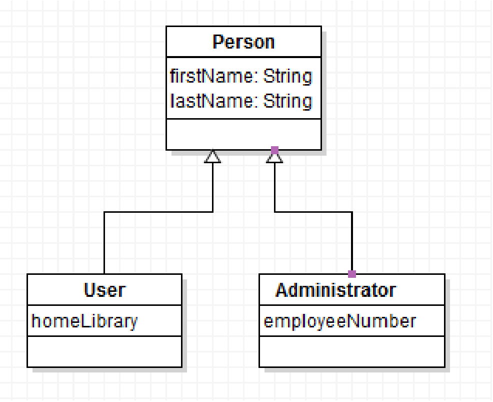
Package

- If you are showing multiple packages in a single class diagram, either
  - Surround the package classes with a dashed border
  - Include your package identifier in the Class name
- Be sure that your packages are organized logically to maximize cohesion
- Do not use the default package



Inheritance

- More general term is Generalization



Realization

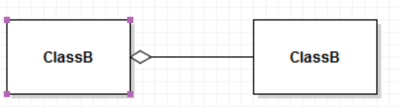
- One class realizes the behavior of the other class
- Satisfies an interface
- Shown with an association line with a triangle
- Possibly shown more clearly with Interface notation

Keywords

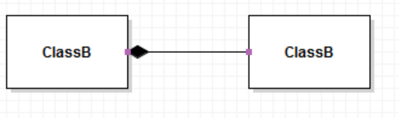
- Textual adornment to categorize a model element
- Can be shown in double brackets (<<…>>) or curly braces ({…})
- Examples
  - Interface
  - Abstract

Aggregation and Composition

- Aggregation
  - Variant of association
  - Shows containment of classes
  - No consistent usage - don't use it



- Composition
  - Variant of association
  - Shows that an instance of the part belongs to only one composite instance at a time


ClassB
ClassB