First, a Quick Review!

CRC Cards for an ATM

- Class: ATM

Responsibilities	Collaborators
Display Option Menu Ask user for PIN Send PIN to Bank for Validation Display Validation Errors Lock System when PIN Invalid Ask Bank for Account Information Display Account Balance Debit Account Dispense Money Print Receipt Eject Card Articulate Display Display Help Articulate Help	Bank

CRC Cards for an ATM

- Class: Bank

Responsibilities	Collaborators
Validate PIN Access Account Balance Add to Account Balance Debit Account Balance	Account

CRC Cards for an ATM

- Class: Account

Responsibilities	Collaborators
Update Account Balance	

Unified Modelling Language

- **Unified Modelling Language** (UML) was designed to help translate models of software into actual software objects and relationships.

- **UML** is language-independent and extremely expressive.

- We'll use only a small part of UML, specifically **Class Diagrams.** Class Diagrams can be used to represent basic features of OO design.

- The entire UML spec is very extensive and can be found at this link: https://www.omg.org/spec/UML/1.4/About-UML/

UML: Class Diagrams

Each class is represented **by a box** divided in three sections:

Class name, Data members, Methods

Player
-name : String -manaPoints : int -healthPoints: int
+Player(name: String, manaPoints: int, healthPoints: int) +getName() :String +fight(other: Player) : void +toString() : String

UML: Class Diagram Notation

Data Members:

name: type

Methods:

methodName(param1: type1, param2: type2, ...): returnType

Visibility:

– private

+ public

protected

~ package

Static: underline

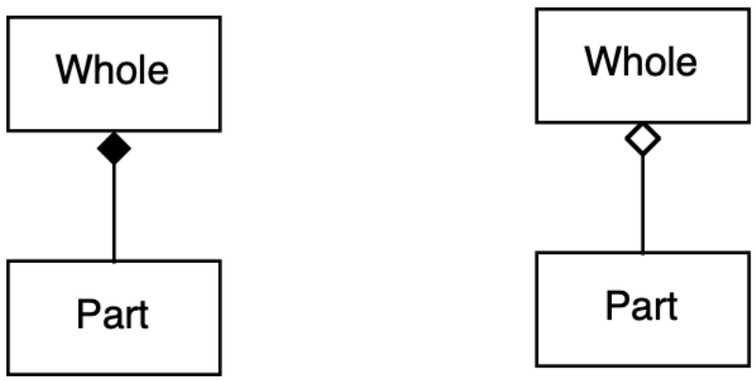
Abstract method: *italic*

Abstract class: *italic* or <<abstract>>

Interface: <<Interface>>

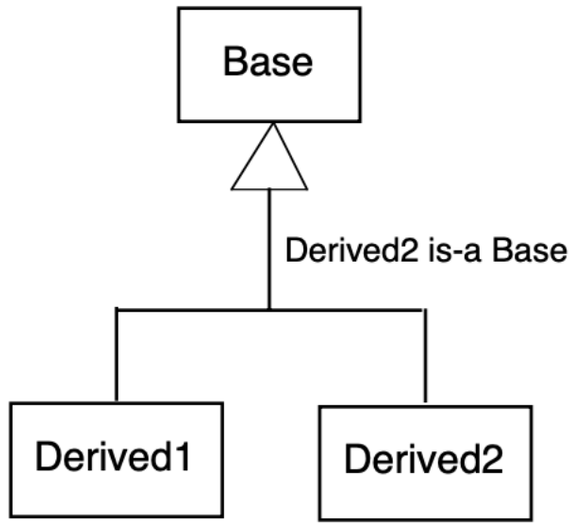
UML: Class Diagram Relationships

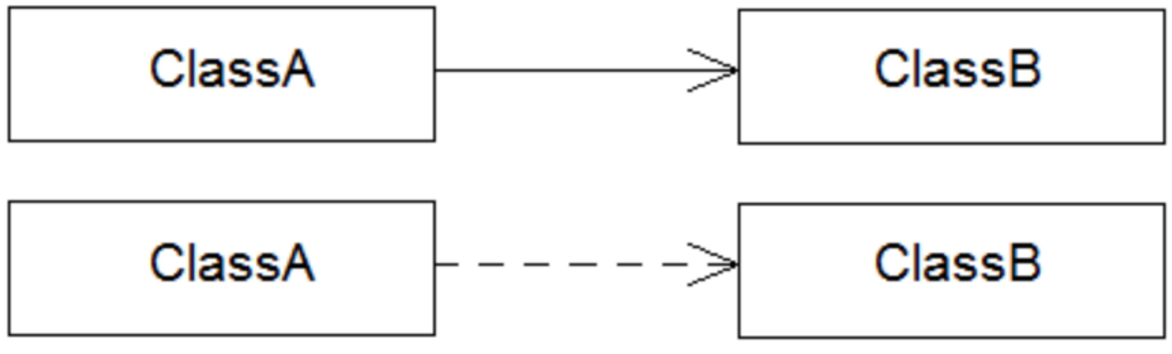
Composition and Aggregation are represented with diamonds.
A composition is a solid diamond, and an aggregation is not.



Inheritance, or **IS-A** relationships,
are represented with triangles

Relationships with i**nterfaces** look similar,
but lines between boxes are dashed.





Associations and **Dependencies** between classes can be represented with arrows.

UML Diagrams

You can use Visual Paradigm to generate your own UML Diagrams

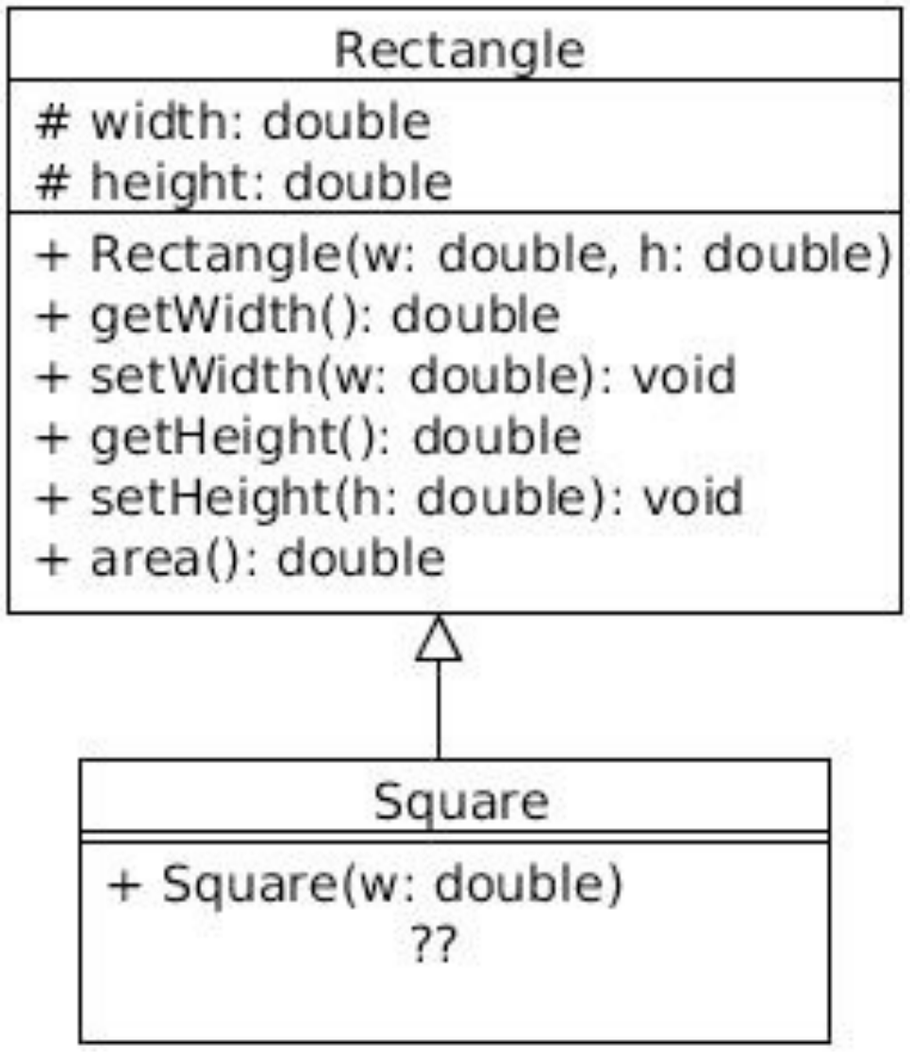
https://ap.visual-paradigm.com/university-of-toronto

We will be asking you to generate UML diagrams for your Group Project.

UNIVERSITY OF TORONTO MISSISSAUGA

UML: What Does it Look Like?

UML can help us sketch relationships between **Square** and **Rectangle** Objects too. This one violates a **SOLID** principle tho. *Do you remember its name?*



Many Design Patterns are communicated in UML

Design Patterns: What are they?

A Design Pattern:

- Describes software solutions and component structures that are common
- Defines software classes, roles and relationships
- Does not depend on any particular programming language
- Must be applied, tailored to create particular solutions

The benefits are many:

- The patterns **work**
- They facilitate **documentation**
- They enhance **communication**
- They promote **re-use**
- They limit **errors**

Classified in four main groups:

Behavioural, Creational, Structural, Concurrency

UNIVERSITY OF TORONTO MISSISSAUGA

Patterns you have seen:

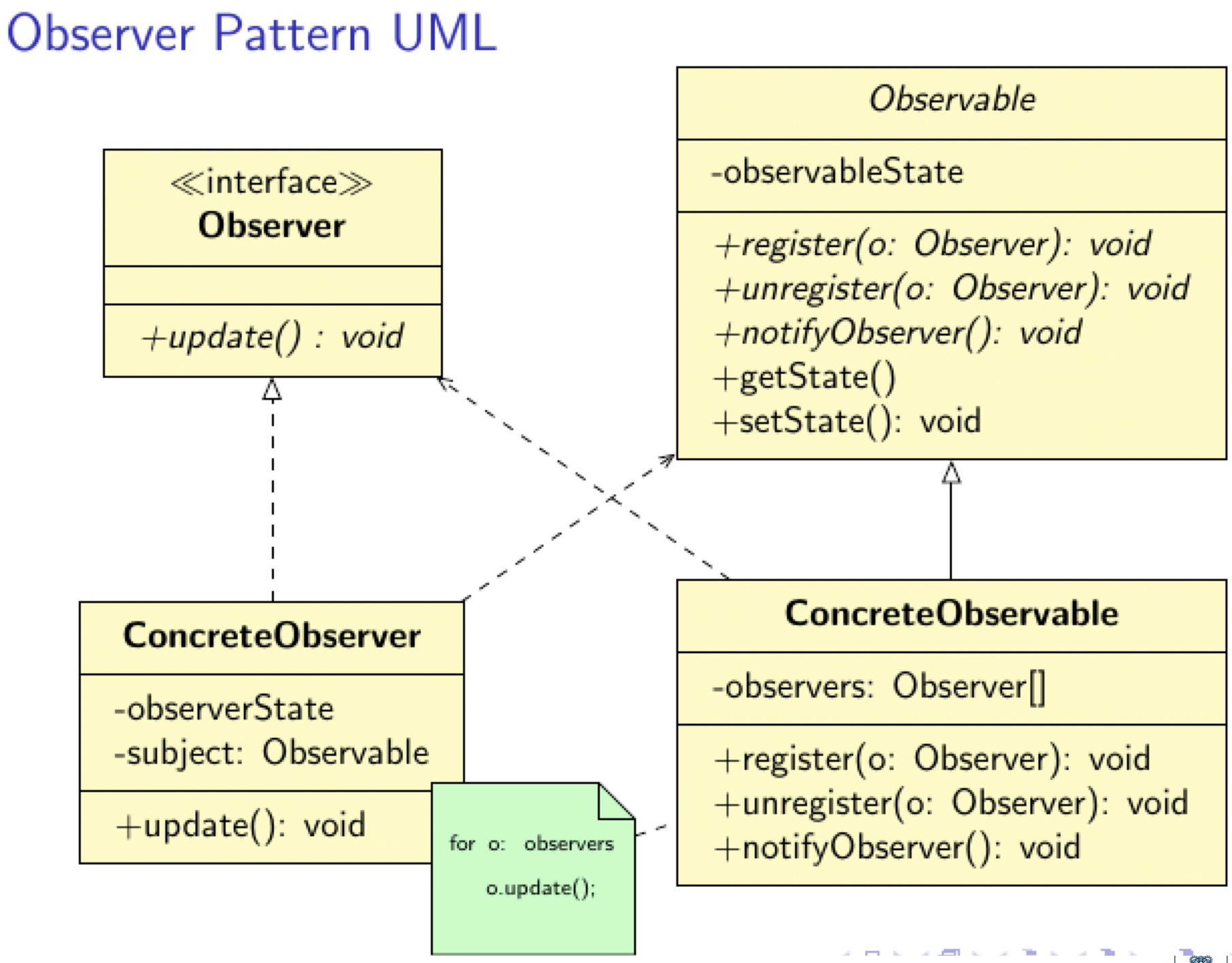
- Assignment 2 separates Views from an underlying Model of program state. This reflects a pattern we will discuss in more detail soon: Model-View-Controller.

- You have seen Iterators and Iterables in past lectures. Iterator is a Design Pattern, too.

- Your lab this week relates to another popular Design Pattern called “Observer”.

- More design patterns have been posted to Quercus; you will want to pick a few that interest or intrigue you for your group project, and you will need to relate these to features your user stories.

Design Patterns: Observer Example



Project Brainstorming Sessions

- Brainstorm project ideas with a TA and an Accessibility Mentor from CNIB.

- Fifteen minute Zoom consults available on: **October 26, 27 and November 1**

- Sign up at this URL

- Present ideas for and/or drafts of user stories and acceptance criteria! Consider user stories that are accessible and engaging to users with low vision.