Unit Revision
Object-Oriented Programming

Dr. Viet Vo
vvo@swin.edu.au
Department of Computing Technologies
School of Science, Computing and Engineering Technologies

Learning Outcomes

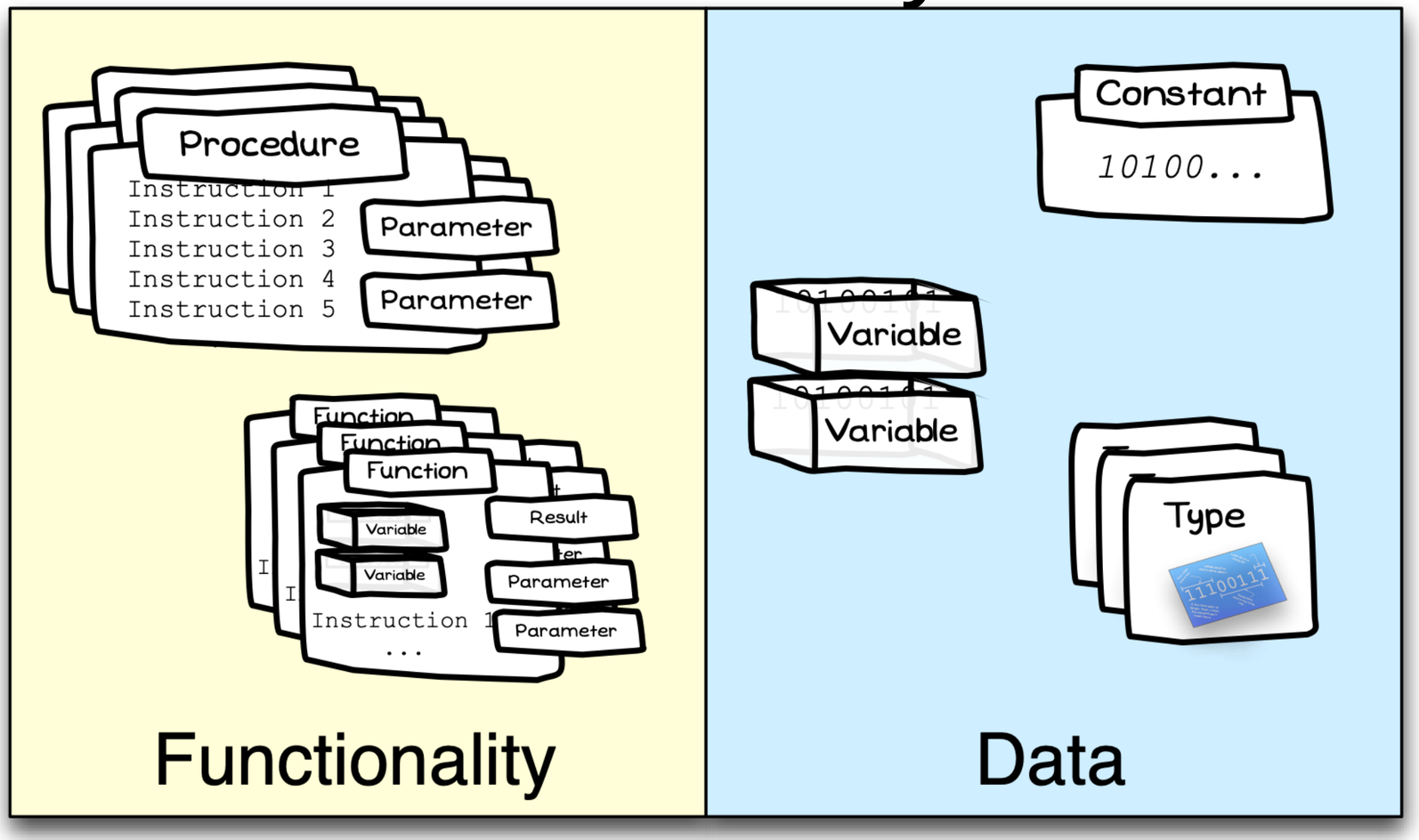
- Review important topics in OOP
- Portfolio summary submission

SWINBURNE UNIVERSITY OF TECHNOLOGY

Unit Schedule

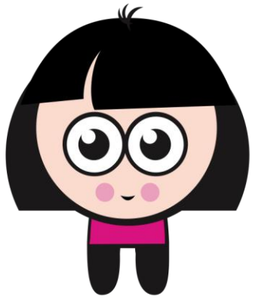
Week	Topics	Assessment	Tasks included in Portfolio
1	Introducing Objects		1.1P and 1.2P
2	Framework Classes, Unit Testing, and UML Class Diagrams		2.1P, 2.2P, 2.3P, and 2.4P
3	Collaboration, Memory, and UML Sequence Diagram		3.1P, 3.2P, and 3.3P
4	Inheritance and Polymorphism		4.1P, 4.2P,
5	Interfaces and Exceptions		5.1P and 5.2P, and 5.3C
6	Responsibility Driven Design		6.1P and 6.2P, and 6.3D and 6.4D and 6,5HD, and 6.6HD
7	Common Mistakes and Test Preparation		7.1P
8	Principles of Good Design	**Hurdle Test (48-hours)**	
9	General Responsibility Assignment Software Patterns		9.1P, and 9.2C, and 9.3HD and 9.4HD
10	Design Patterns		10.1C
11	Other Object Oriented Languages		11.1P
12	Object Oriented Programming in a Nutshell		**Portfolio Submission and Assessment**

Week 1: Program procedurally by organising code into separate artefacts for data and functionality

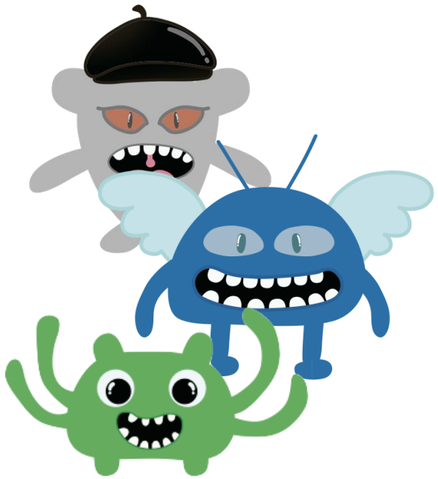


Week 1: With Objects, you create entities that encapsulate **both** functionality and data — they know and can do things

Week 1: Use abstraction to classify the different kinds of roles objects will play in your software



classify → Player

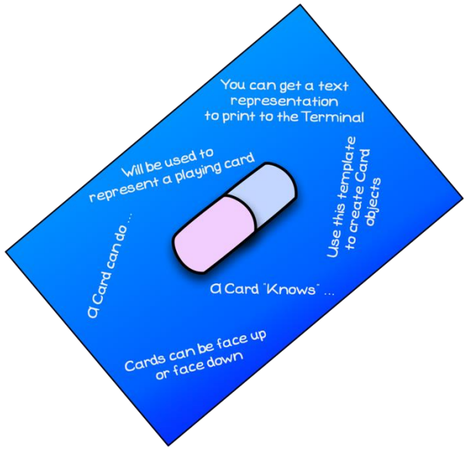


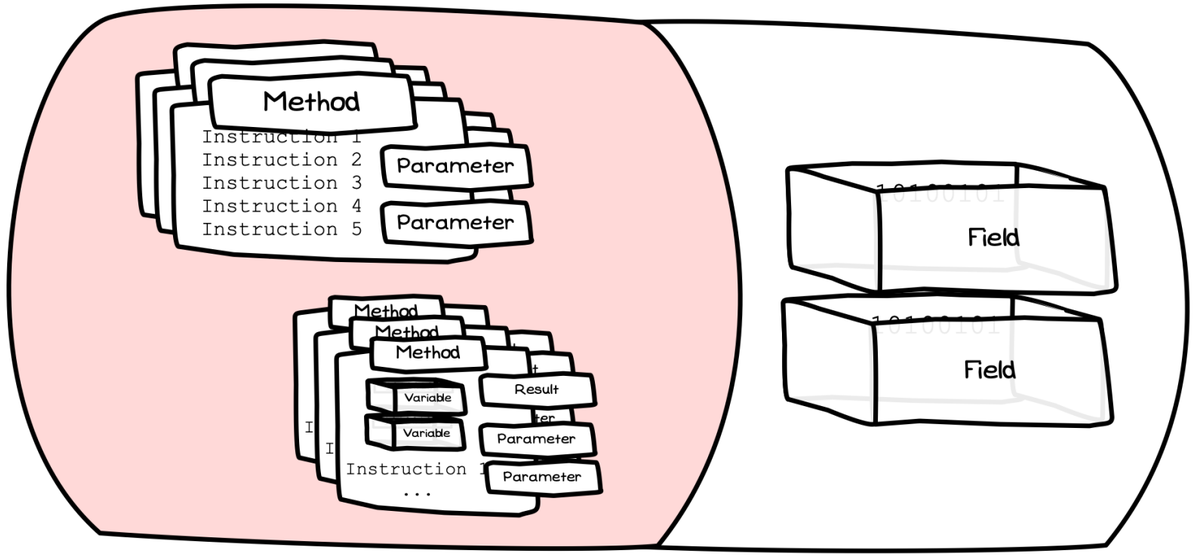
classify → Alien

Use Abstraction (Classification) to define object classes

SWINBURNE UNIVERSITY OF TECHNOLOGY

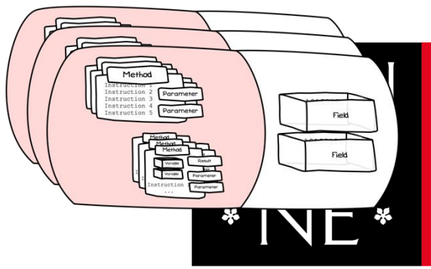
Week 2: Add special methods called constructors to initialise your objects when created





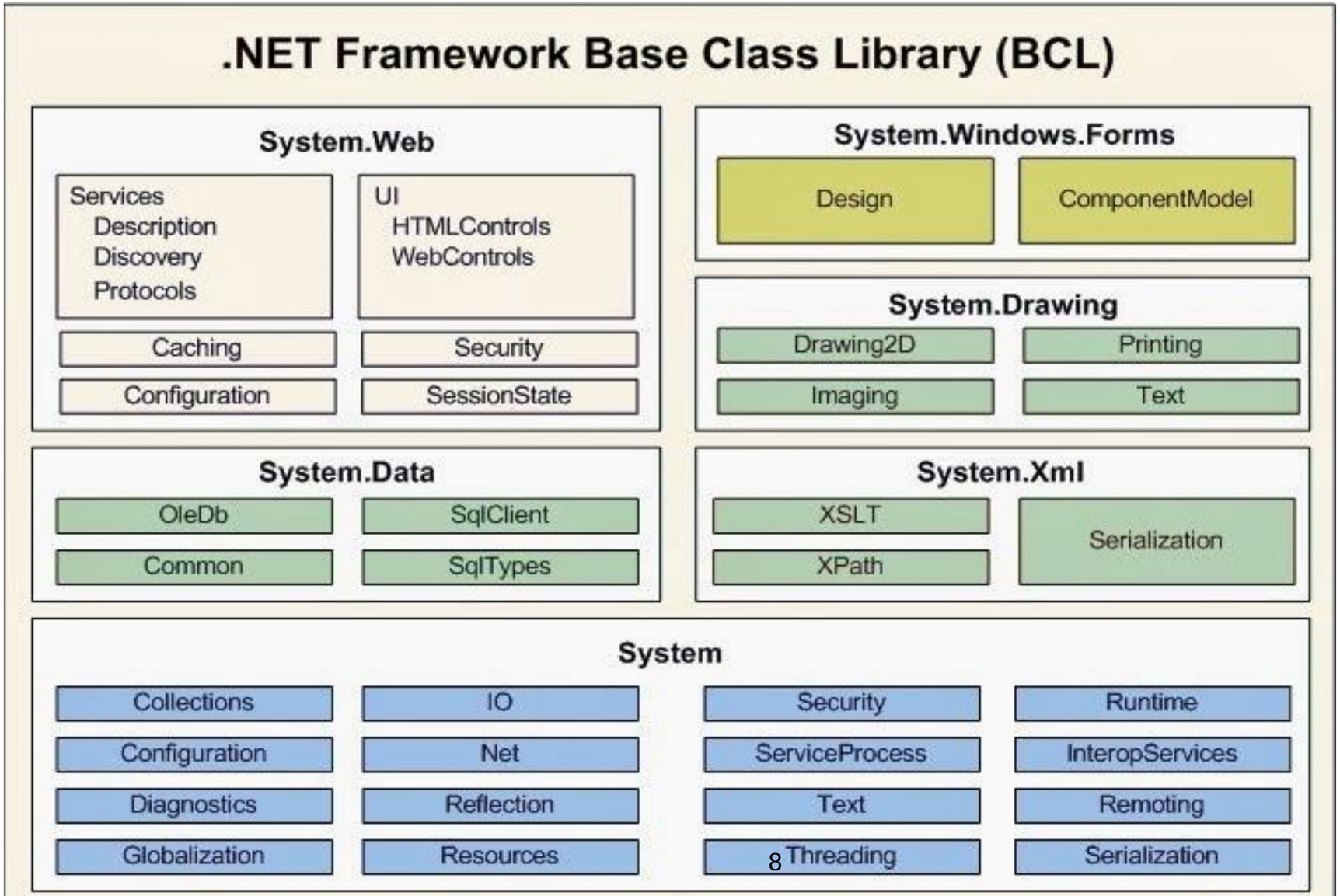
Constructors are declared within the class

These define how to create/initialise the objects.



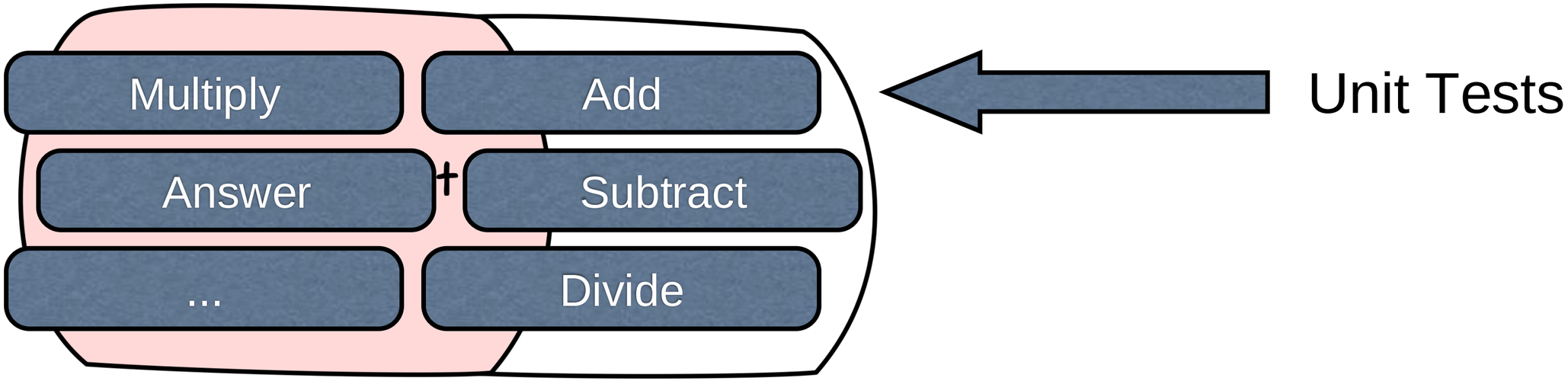
SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 2: . Net Frameworks cover an extensive range of features.



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 2: Unit Test



Week 2: Create Testing Classes

```
namespace Task1_2
{
    public class StudentTests
    {
        [Test]
        public void StudentName_ShouldBeSetCorrectly()
        {
            // Arrange
            Student student = new Student("John Doe", 1);

            // Act
            string name = student.Name;

            // Assert
            ClassicAssert.AreEqual("John Doe", name);
        }
```


SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 3: Object collaboration

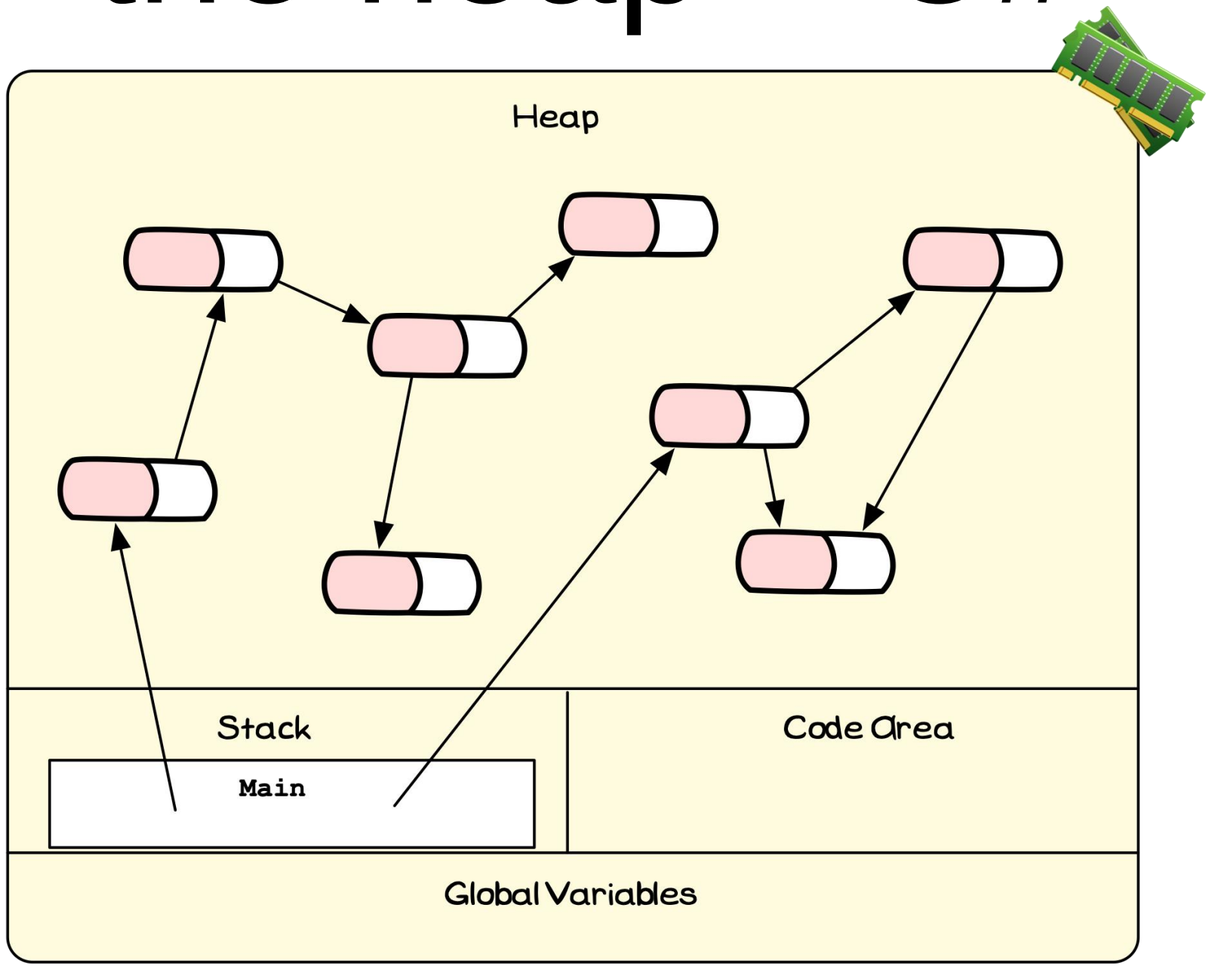
Association

Aggregation

Dependence

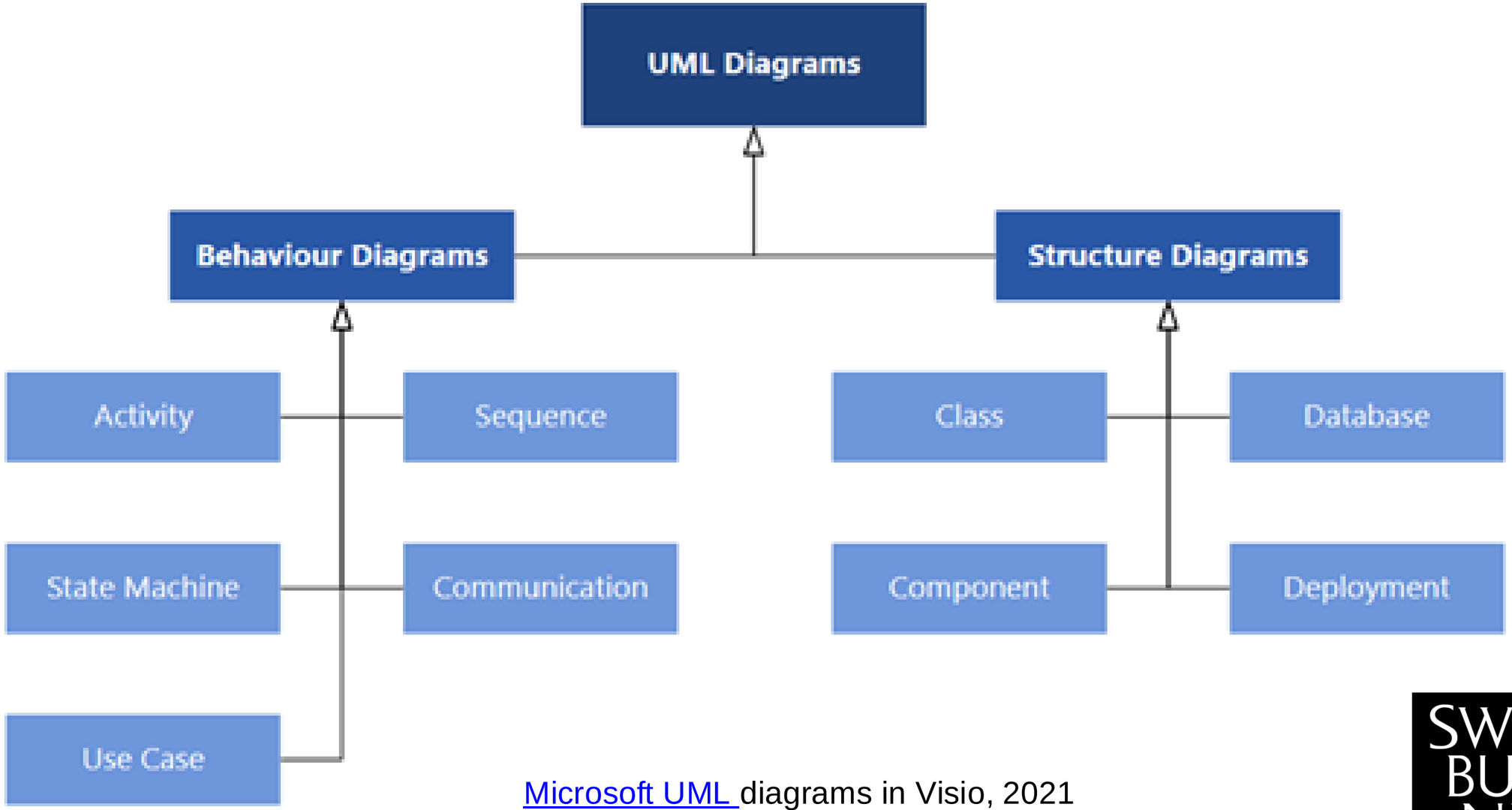
SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 3: At runtime objects exist on the heap – C#



SWINBURNE UNIVERSITY OF TECHNOLOGY

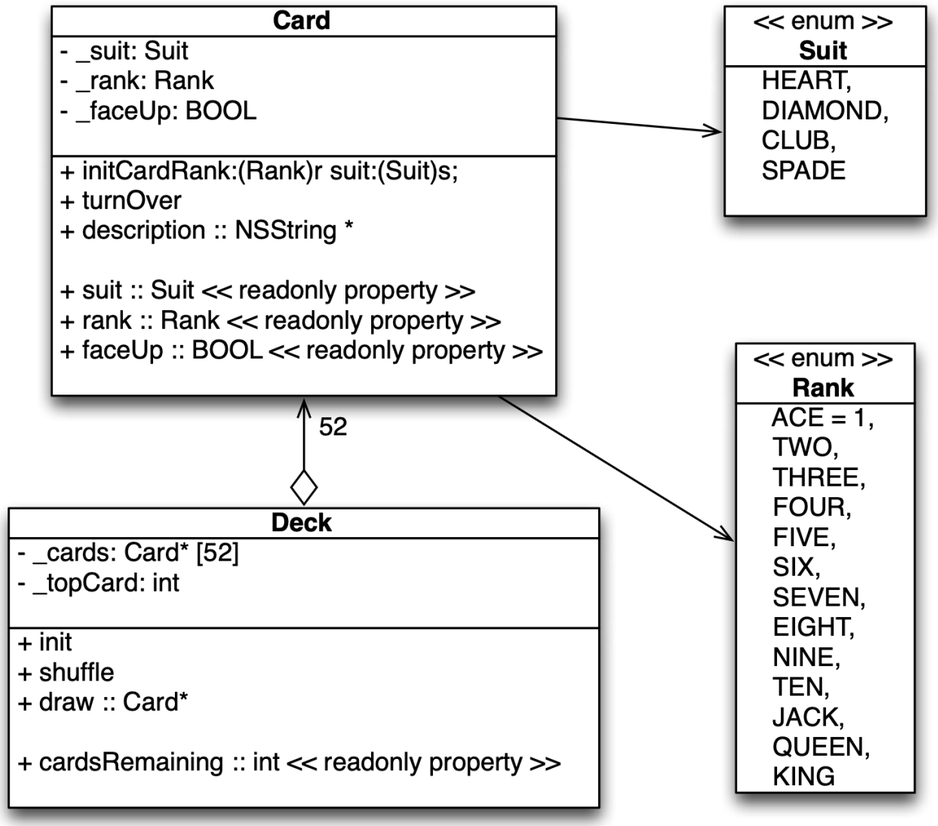
Week 3: Different Types of Diagrams



Microsoft UML diagrams in Visio, 2021

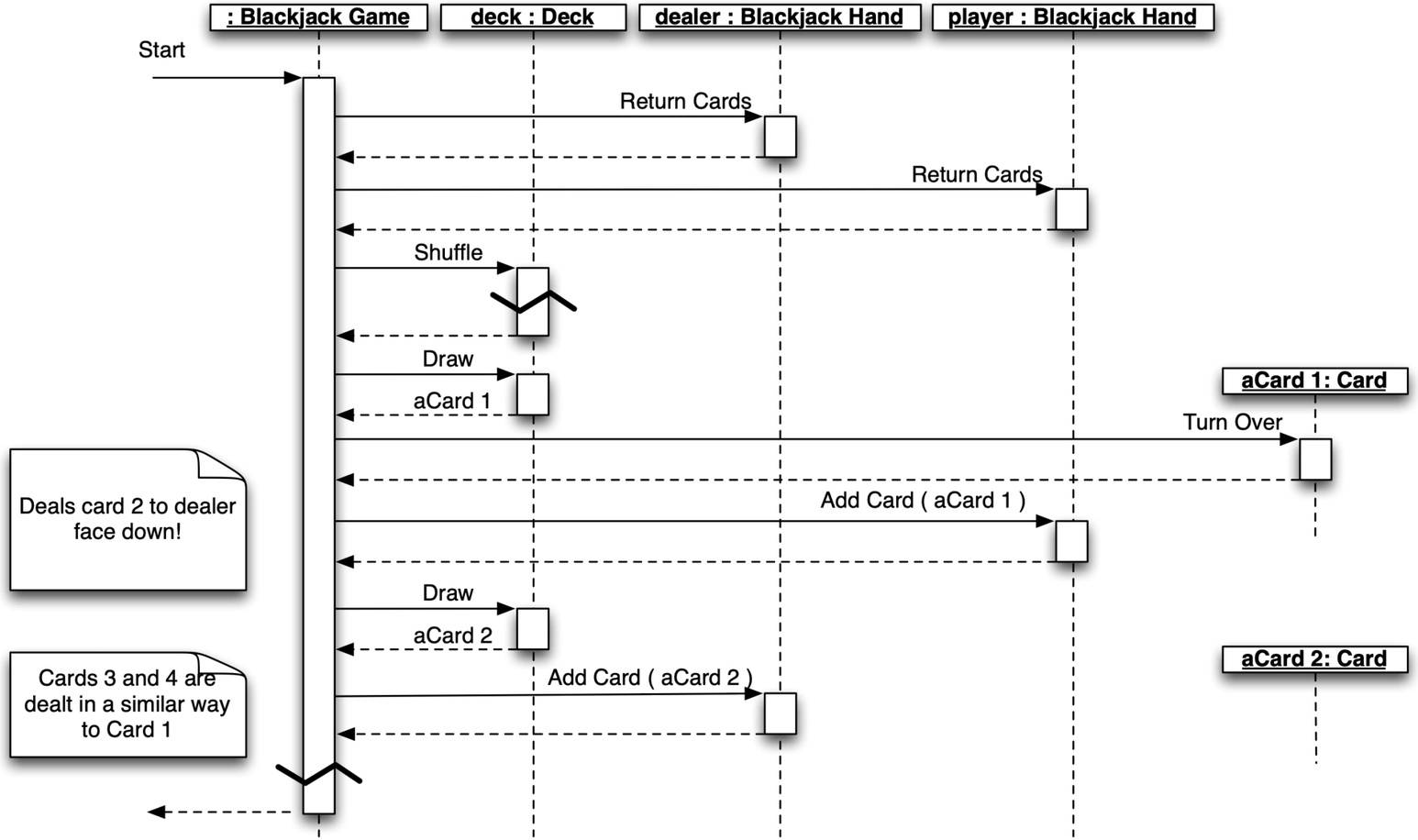
SWINBURNE UNIVERSITY OF TECHNOLOGY

Communicate the static structure of your program using a class diagram



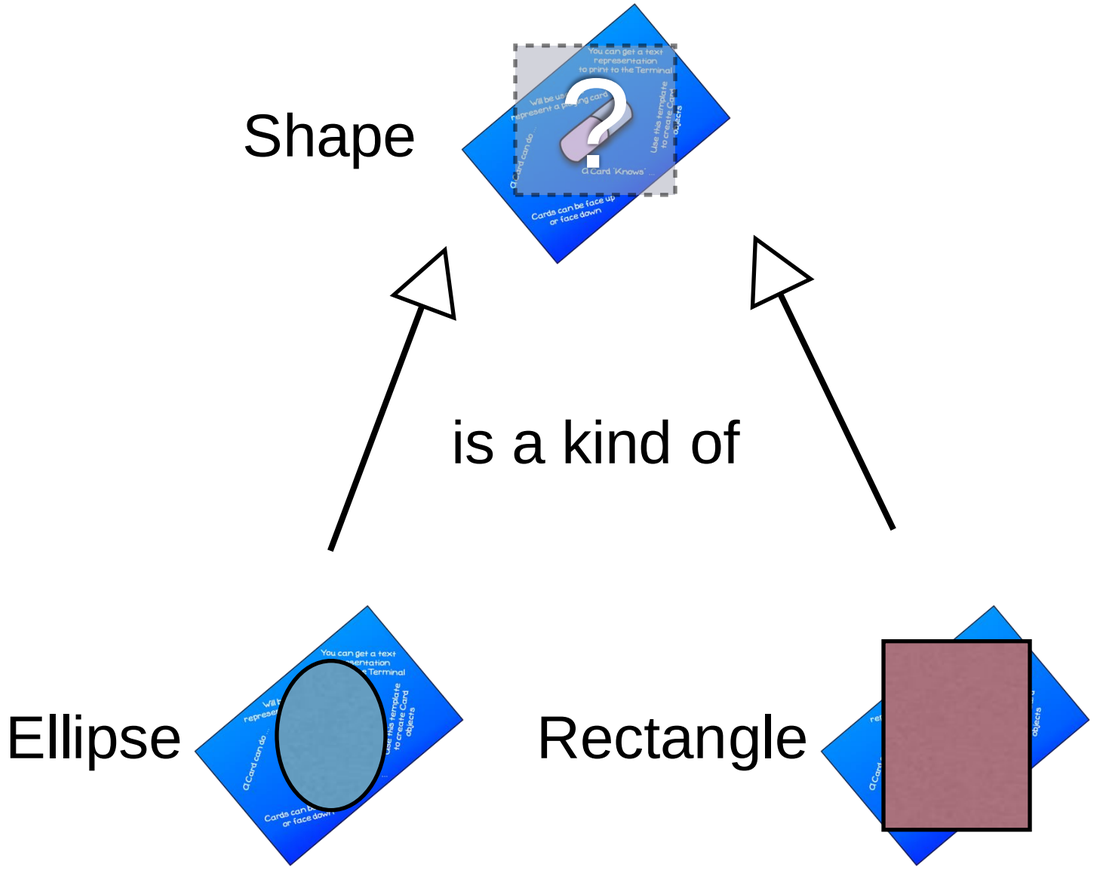
SWINBURNE UNIVERSITY OF TECHNOLOGY

Sequence Diagram : Communicate interactions using a sequence diagram



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 4: Use inheritance to model **is-a** kinds of relationships



SWINBURNE UNIVERSITY OF TECHNOLOGY

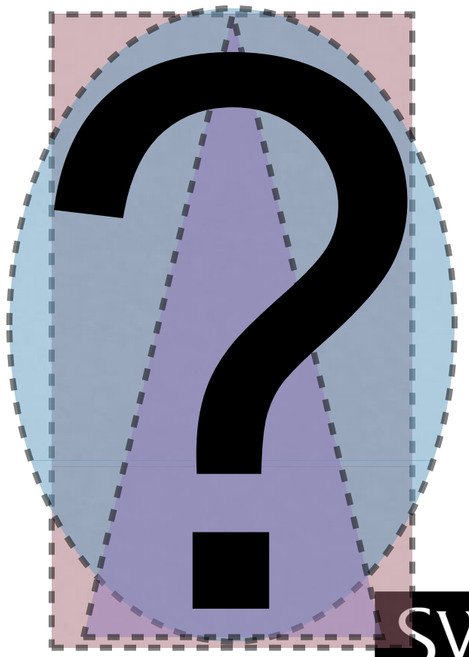
Week 4: Use child objects where the parent is expected

- Polymorphism – a Greek word that means “many-shaped”
- As C# developers, we do not know which specific type of shapes the end-user may want to instantiate in runtime
- No matter which shape selected, it still has a ‘Draw’ behaviour
- Polymorphism can help that with correct objects

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 4: Flexibility: Refer to a parent class, but get child objects… they work as expected!

`Shape s` $\xrightarrow{\text{Draw}}$

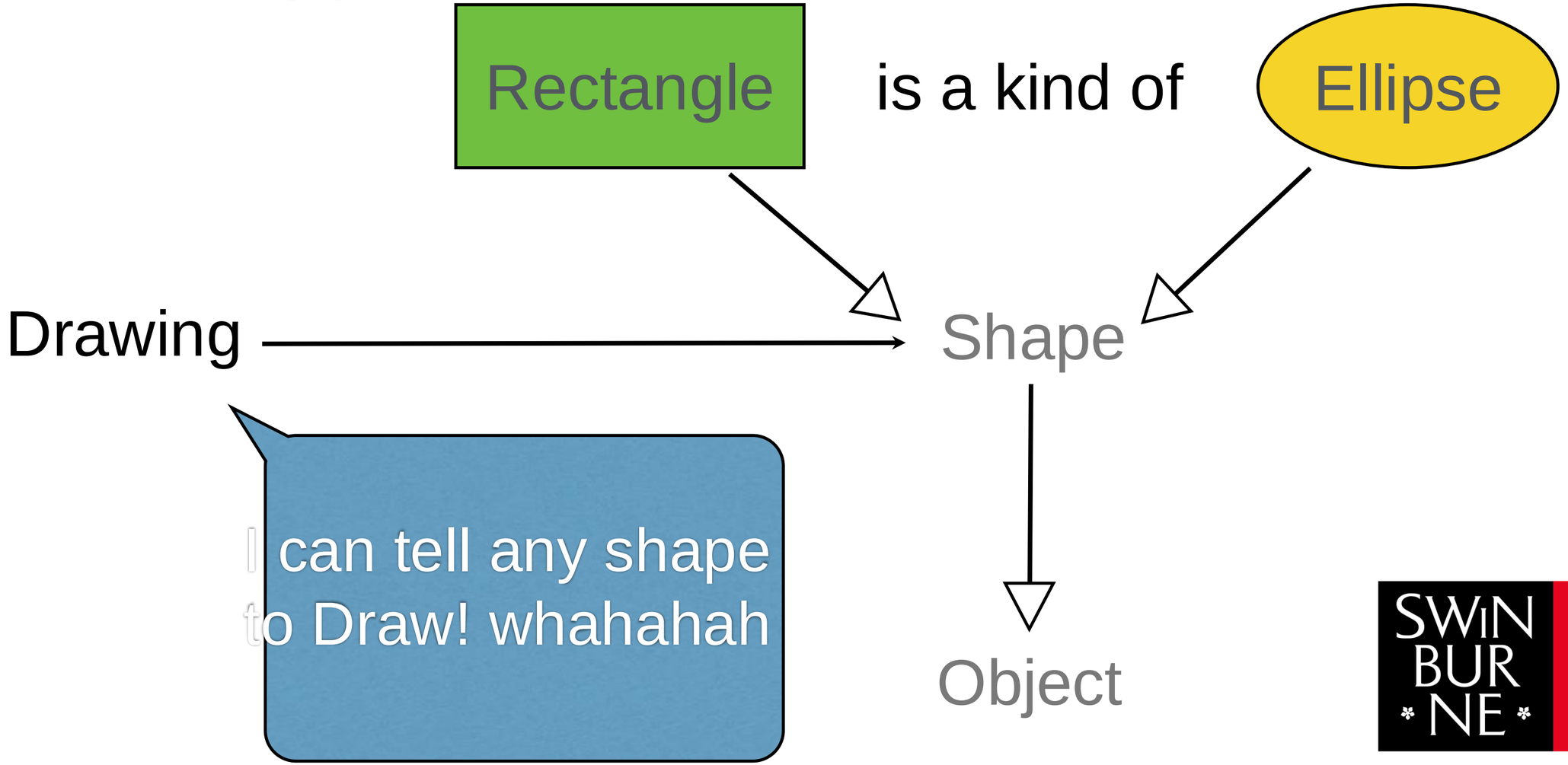


Shape mywhich = Rectangle/Triangle/Ellipse
mywhich.Draw()

SWINBURNE UNIVERSITY OF TECHNOLOGY

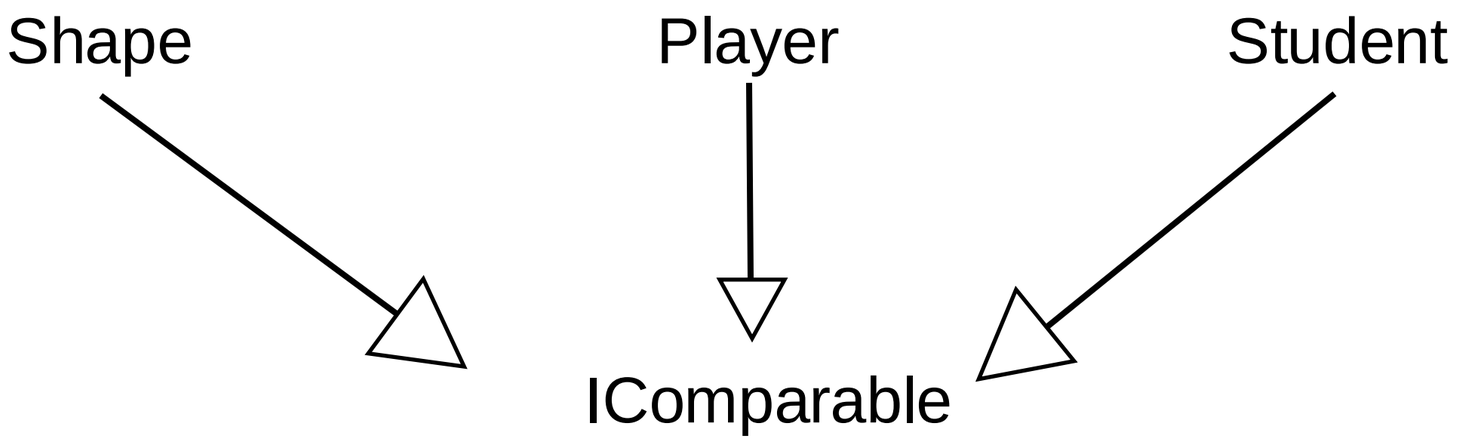
Week 5: Interfaces

Developers use inheritance to create families of types with common features



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 5: Polymorphism means objects of this type can now be used anywhere the interface is needed




```
List<IComparable> myGenericList = new List< IComparable >();
Shape myShape = new Shape();
Player myPlayer = new Player();
myGenericList.Add(myShape);
```

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 5: Use catch block to deal with the error

```
try
{
    ...
}
catch (System.Exception e)
{
    // cleanup
    ...
}
```

Ok... it threw an exception. I need to clean up this mess!

Example →

```
string filePath = Console.ReadLine();
try
{
    StreamReader reader = new StreamReader(filePath)
     Console.WriteLine("File opened successfully.");
}
catch (FileNotFoundException ex)
{
     Console.WriteLine(”Error: File not found");
}
catch (IOException ex)
{
     Console.WriteLine(”Error while accessing file");
}
catch (System.Exception ex)
{
     Console.WriteLine(”Unexpected");
}
```

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 6: Responsibility Driven Design

- Creates effective OO designs using Roles, Responsibilities, and Collaborations
- Design before code = better code + faster written
- Emphasizes behavioral modeling
- Turns software requirements in to OO software

Wirfs-Brock, Rebecca et al. “Object Design: Roles, Responsibilities, and Collaborations.” (2002).

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 6: Responsibility Driven Design

Step 1: Define the purpose for objects in your program using **Roles**

Step 2: Define **responsibilities** for each candidate role

Step 3: **Collaborate** with other objects to meet responsibilities

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 6: Use the different kinds of relationships to help identify possible links

- Association
- Aggregation
- Dependency

→

- where to use
- clarify using software requirement/business analysis

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 6: Use scenarios to test how your model responds to events and implements features

Chess Game

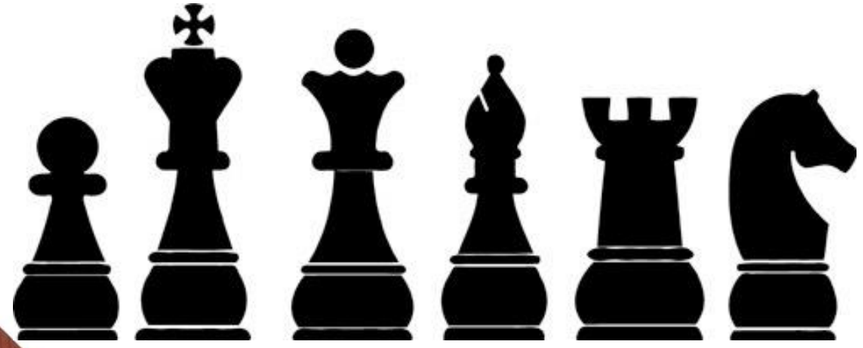
knows its Board
knows its Pieces
knows its Players
can start the game
can process a players turn

Board, setup.

Rook, exist.

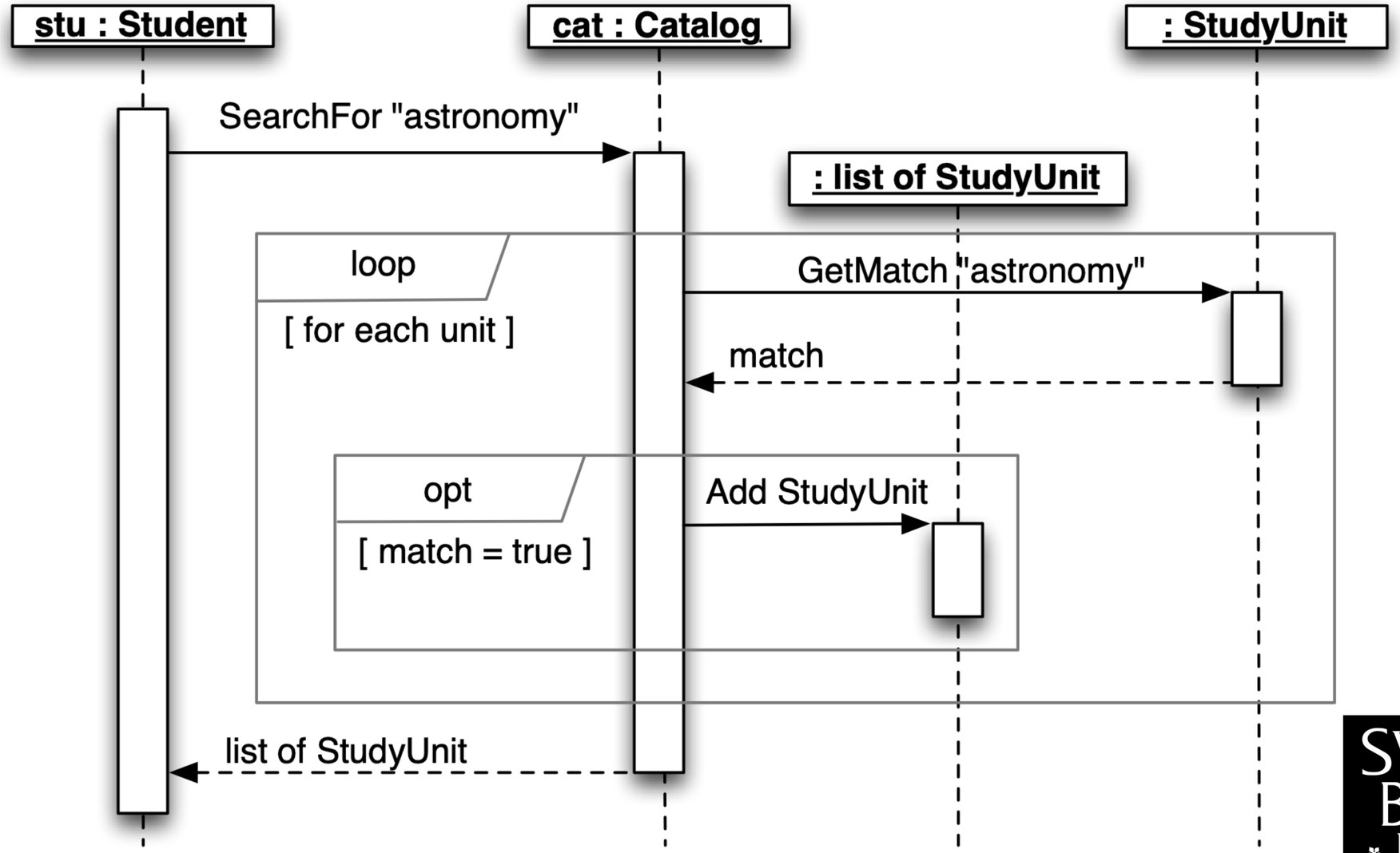
Rook, this is your King.

Cell, hold this Rook.



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 6: Use Sequence Diagram to model control flow logic



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 7: Methods can have the same name as long as they have different parameters

```
public static void DrawRectangle(Color clr, Rectangle rect)...
public static void DrawRectangle(Color clr, Rectangle rect, DrawingOptions opts)...
public static void DrawRectangle(Color clr, double x, double y, double width, double height)...
public static void DrawRectangle(Color clr, double x, double y, double width, double height, DrawingOptions opts)
```

```
1 of 4   void SplashKit.DrawRectangle(Color clr, Rectangle rect)

SplashKit.DrawRectangle()
```

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 7: Scope tells us **where** a variable can be accessed

I know about a variable called ‘i’!

```
public void MethodOne()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }
}
```

I know about a different ‘i’! Neat!

```
public void MethodTwo()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }
}
```

Week 8: Good design is often described in terms of design goals

Extensible!

Robust!

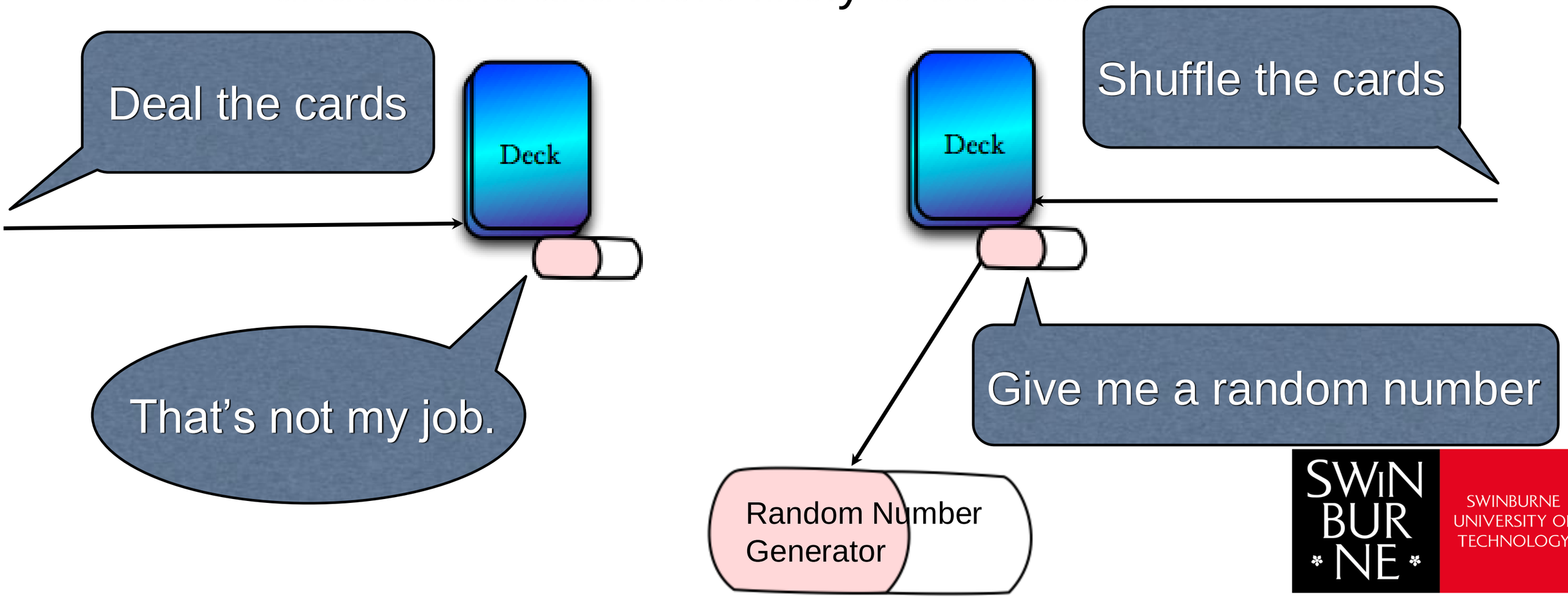
Flexible!

Modular!

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 8: Guideline1. Classes should be lazy

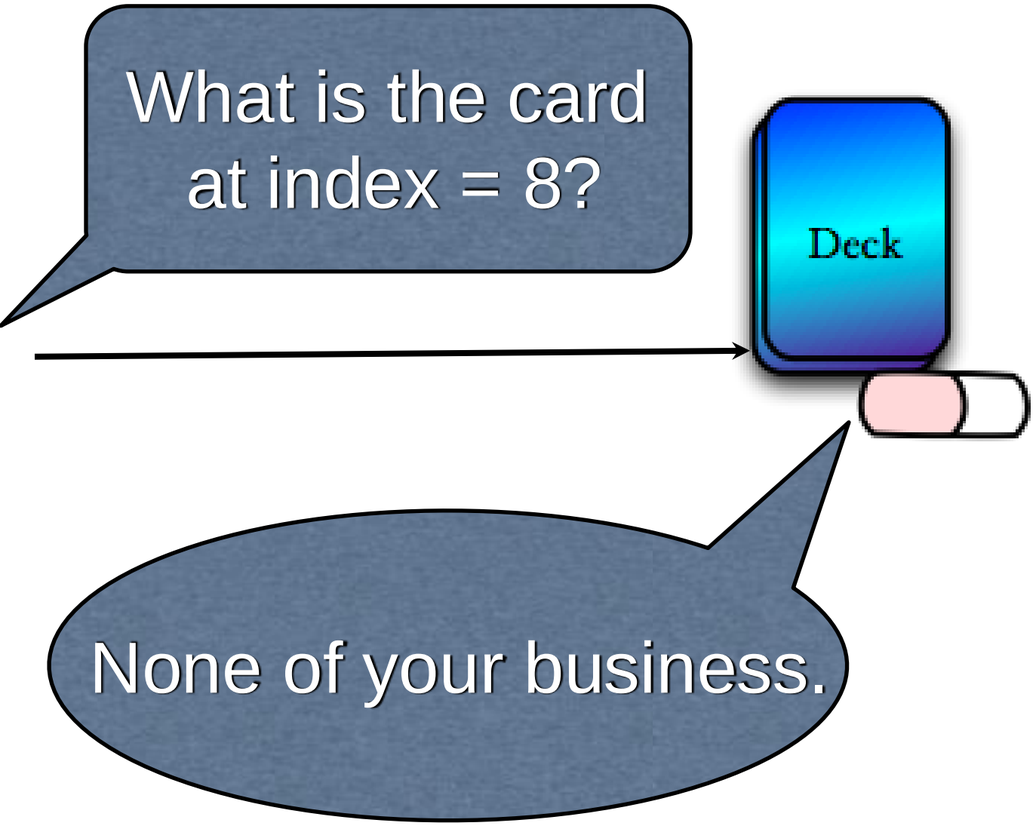
Focused classes with well-defined responsibilities are easier to understand and more likely to be reuseable

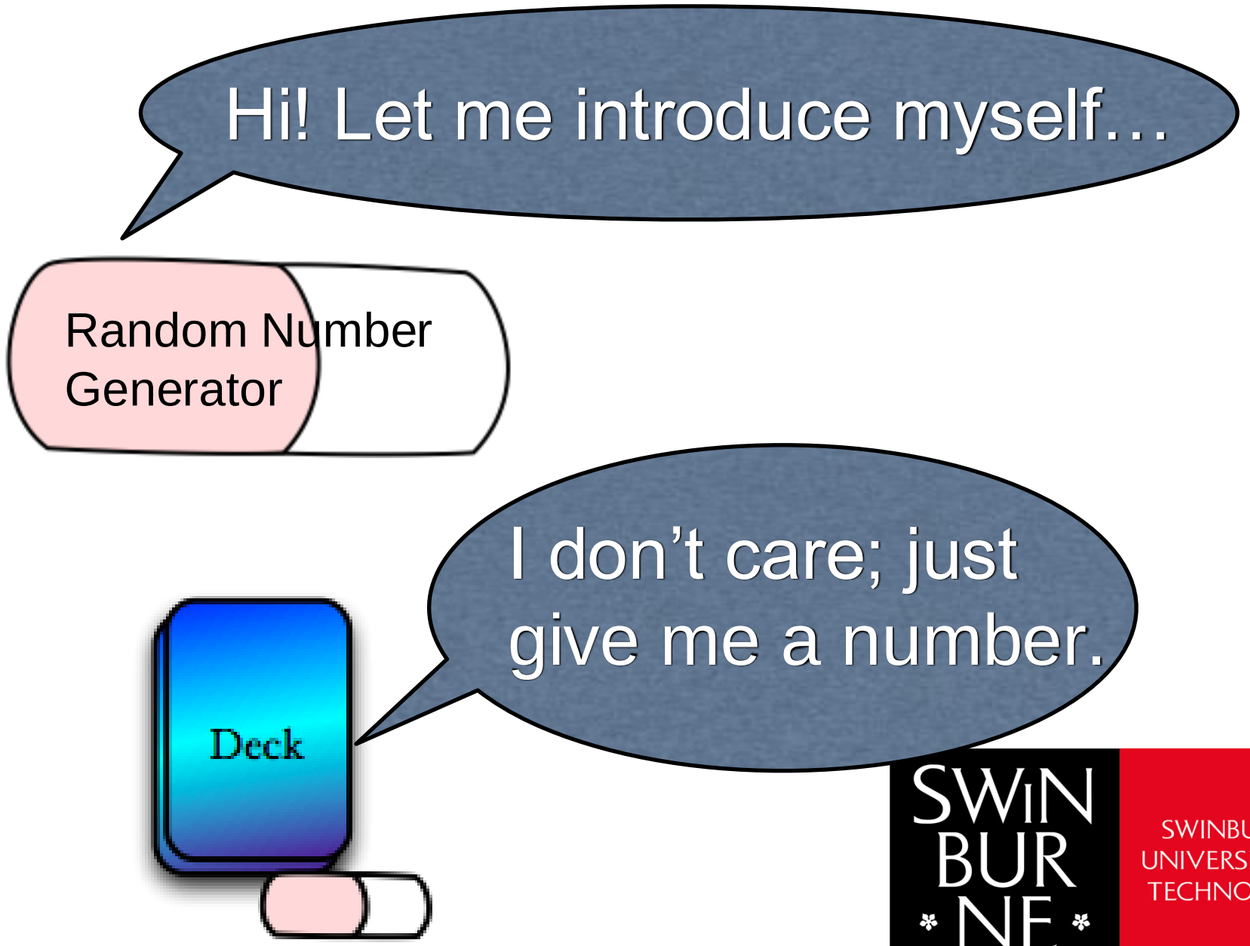


SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 8: Guideline 2. Classes should be antisocial

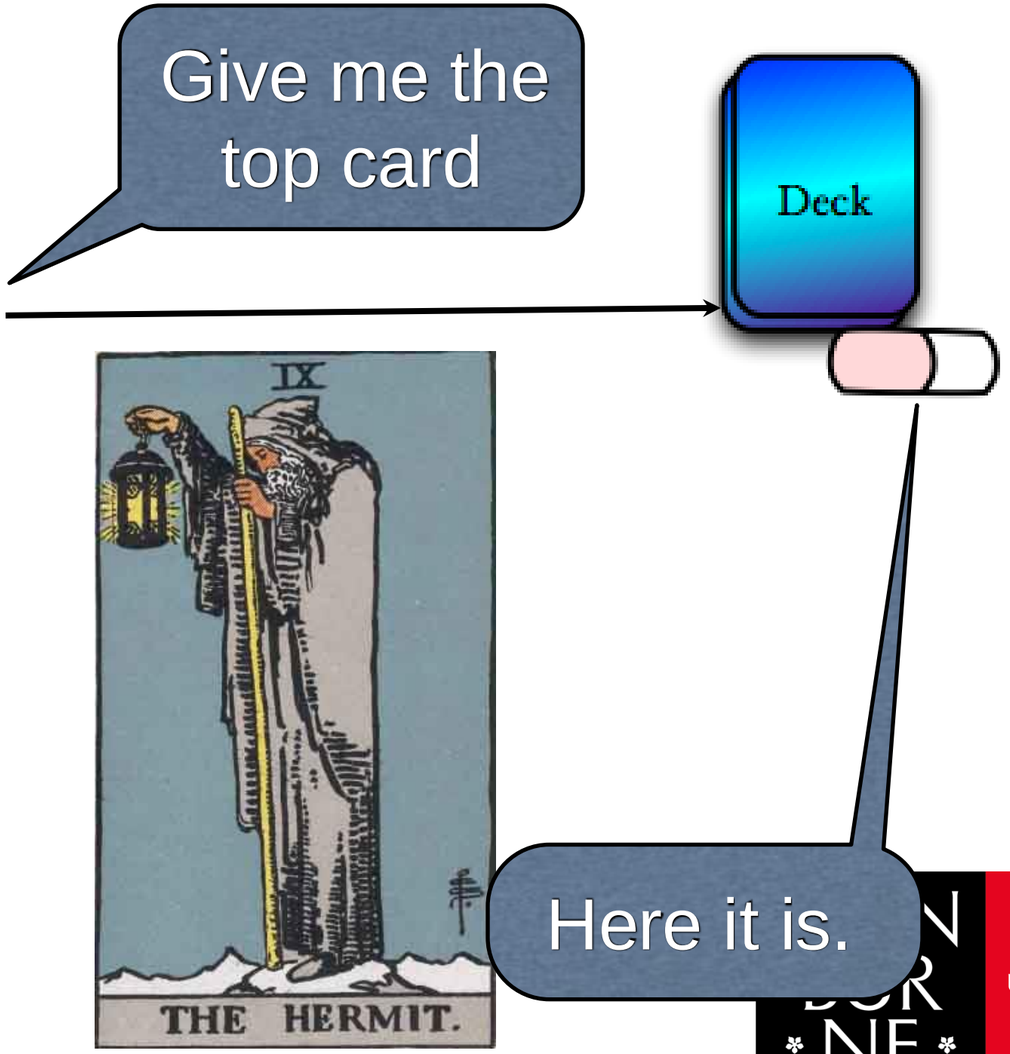
Class should keep details hidden, only share what is needed





SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 8: Guideline 3. Derived classes should be conformist



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 8: Apply three simple rules to help evaluate object-oriented designs

Laziness
Anti-sociality
Conformity

imply to

Extensibility
Flexibility
Robustness
Modularity

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 9: General Responsibility Assignment Software Patterns

- Software patterns provide optimised, reusable templates to solve problems

- Good design choices

  Extensibility
  Flexibility
  Robustness
  Modularity

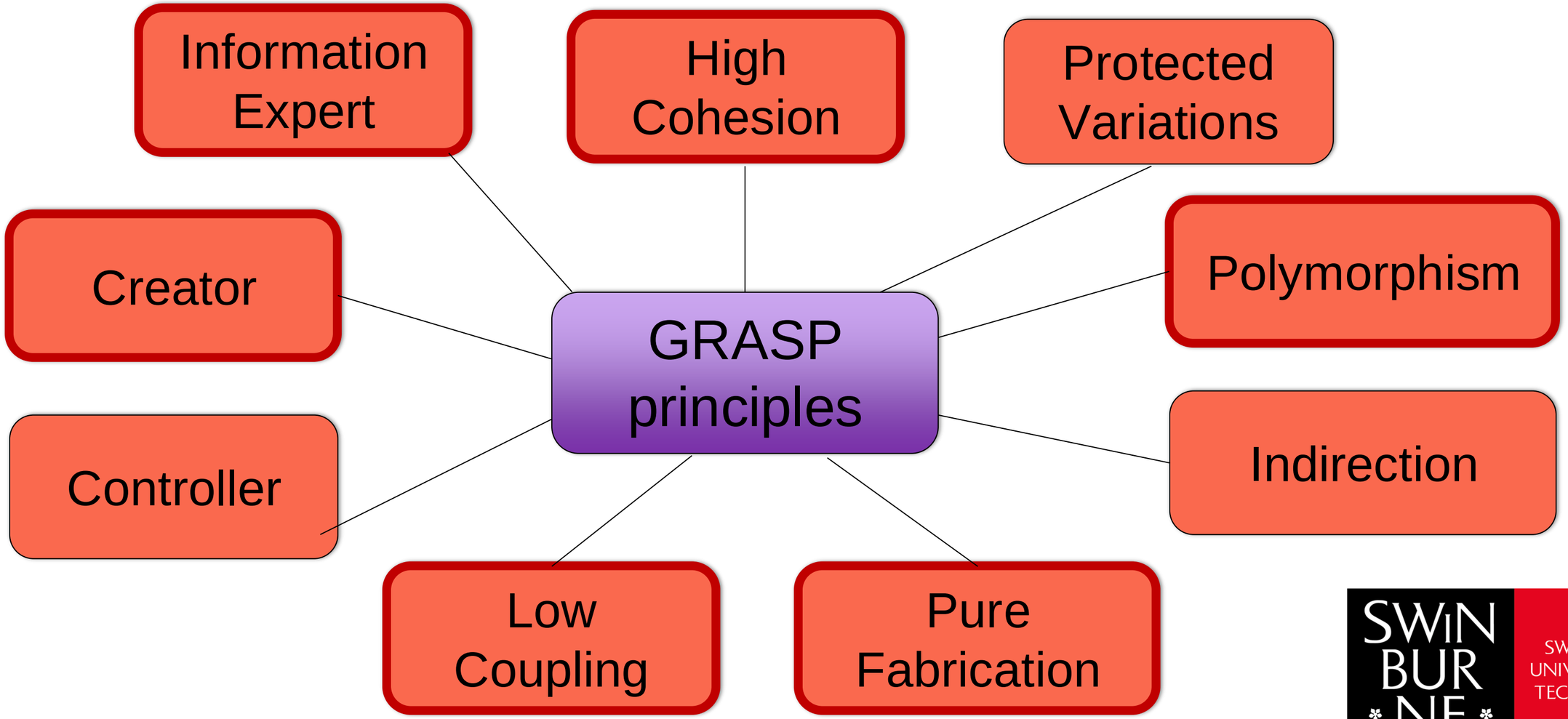
- GRASP

  •set of patterns containing guidelines and principles

  •provides a systematic approach to assign responsibilities to class and objects

SWINBURNE UNIVERSITY OF TECHNOLOGY

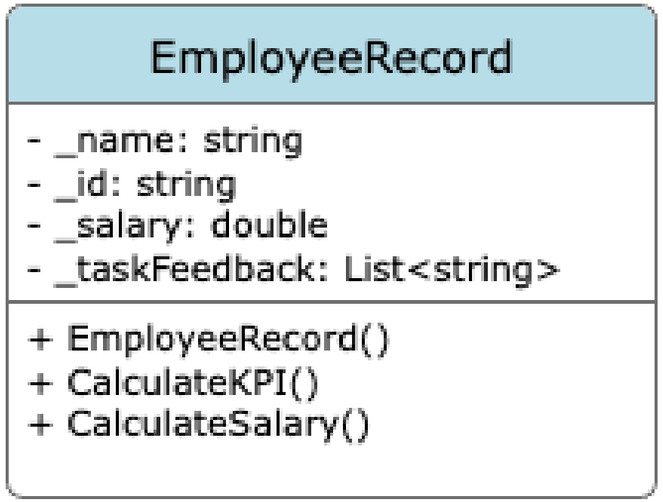
Week 9: General Responsibility Assignment Software Patterns



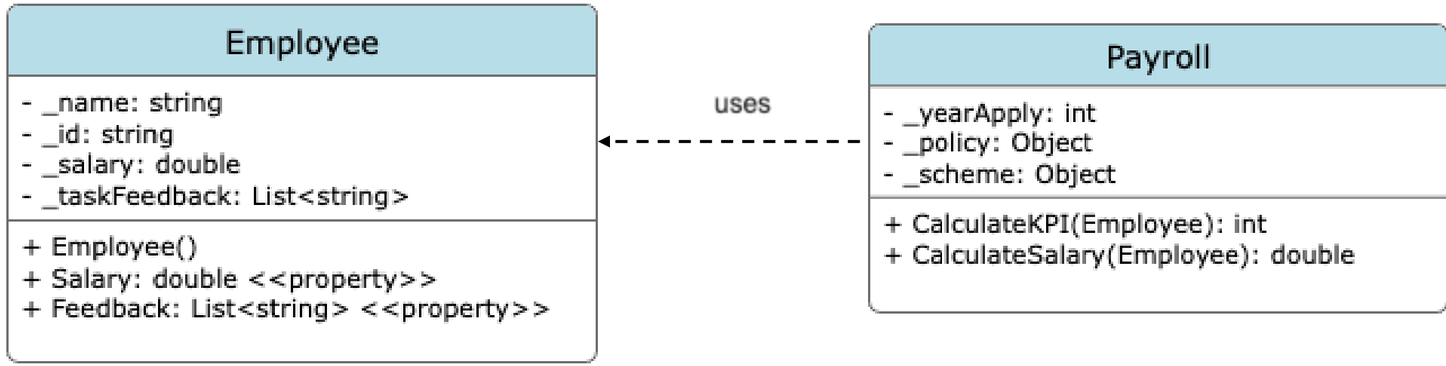
SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 9: Cohesion

The degree of strongly related functionality are described within the classes



Low Cohesion

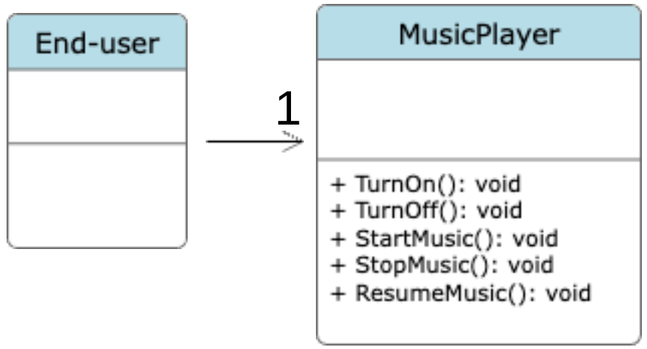


High Cohesion

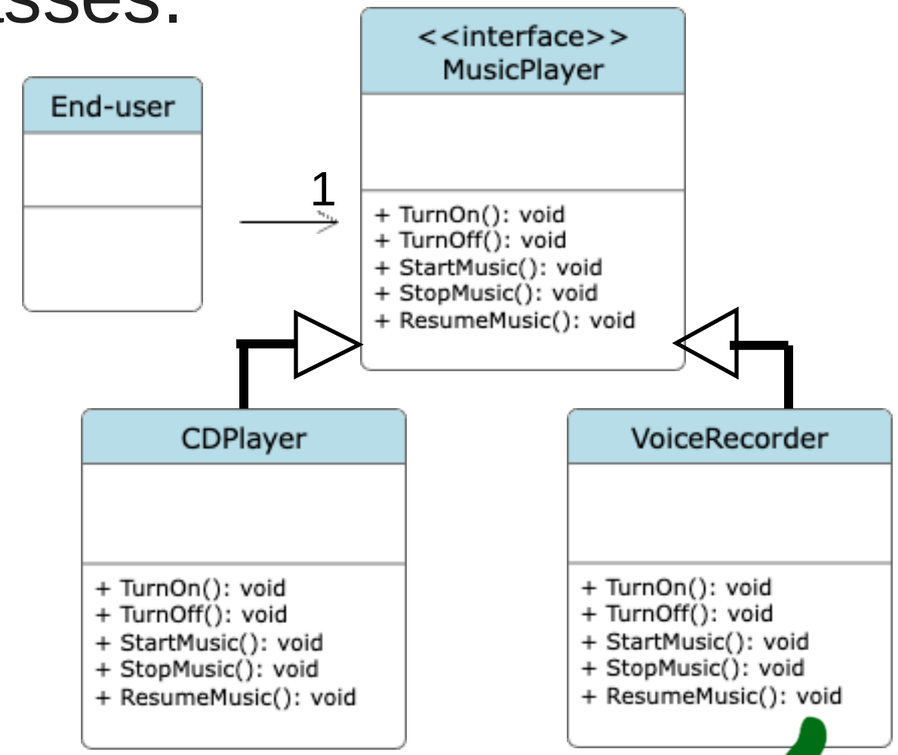
SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 9: Coupling

The degree of dependence among classes.



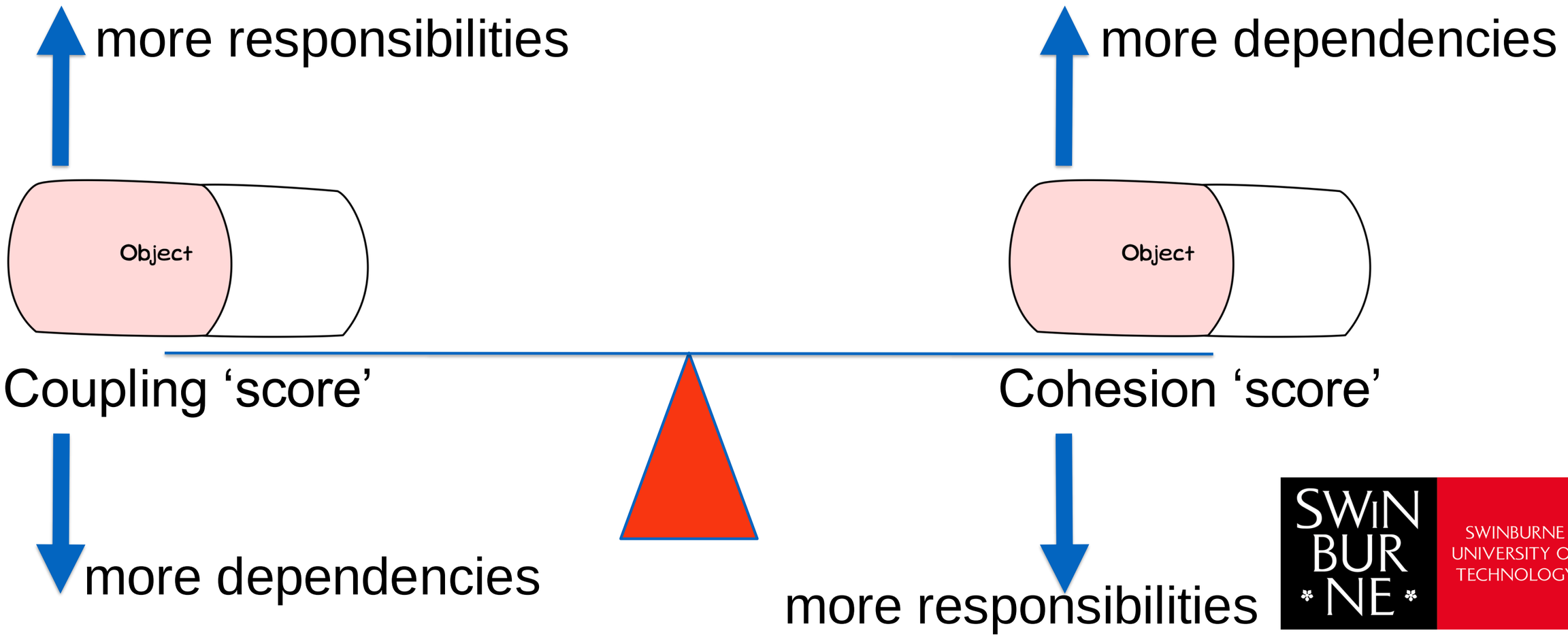
Tight Coupling



Loose Coupling

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 9: Maintain a balance between coupling and cohesion



SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 10: Classifications of Design Patterns

- **Structural Design Patterns:** deal with relationships between objects, making it easier for these entities to work together

- **Creational Patterns:** provide instantiation mechanisms, making it easier to create objects in a way that suits the situation

- **Behavioural Patterns:** dictate communications between objects, increasing the ease and flexibility of object communication

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 10: Classifications of Design Patterns

Creational

Singleton
Factory
Builder
Prototype
…

Structural

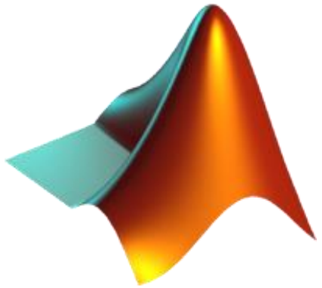
Adapter
Composite
Bridge
Proxy
Decorator
Façade
…

Behavioural

Strategy
Observer
State
Command
Visitor
…

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 11: Approach new languages armed with the principles of OOP



SWIN BUR NE

SWINBURNE UNIVERSITY OF TECHNOLOGY

Portfolio Submission

SWINBURNE UNIVERSITY OF TECHNOLOGY

Double check your task submissions and feedback

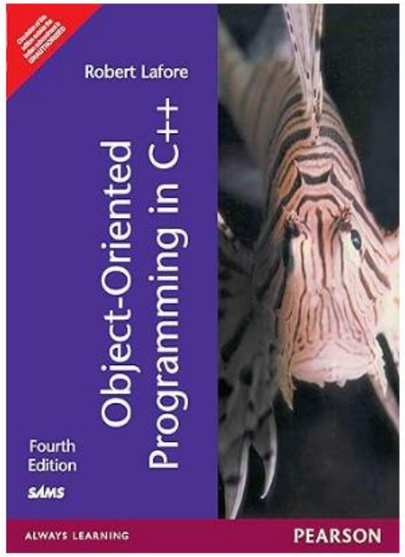
If not receiving feedbacks, contact tutors asap

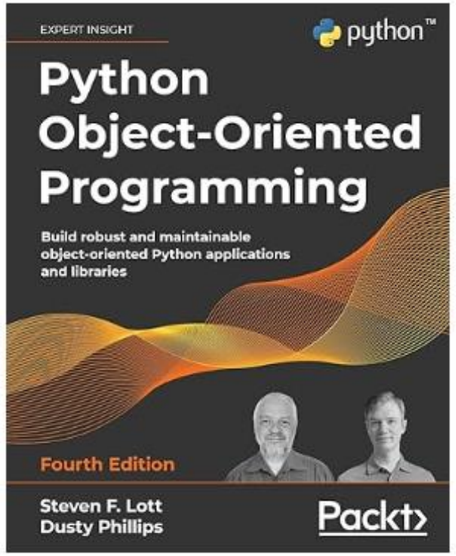
Academic Integrity Check

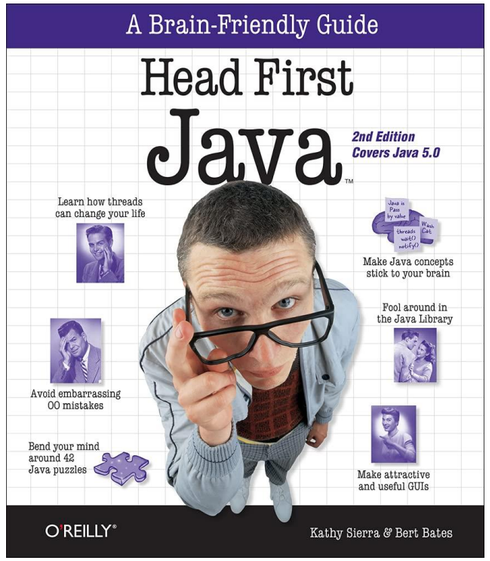
Custom Project Demonstration

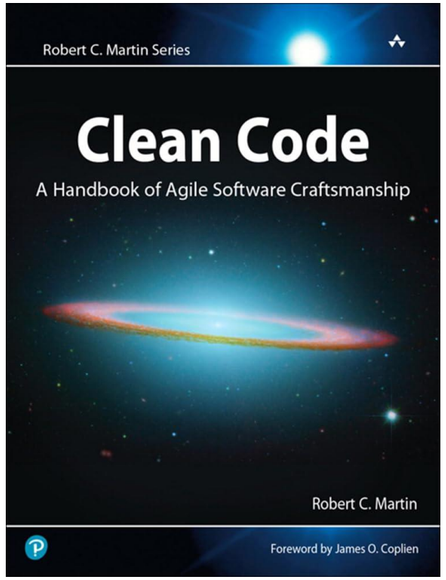
SWINBURNE UNIVERSITY OF TECHNOLOGY

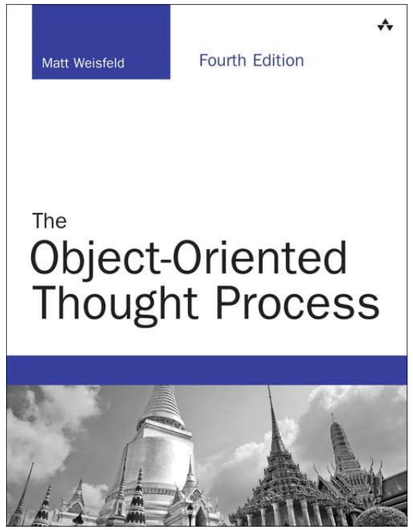
Recommended Books

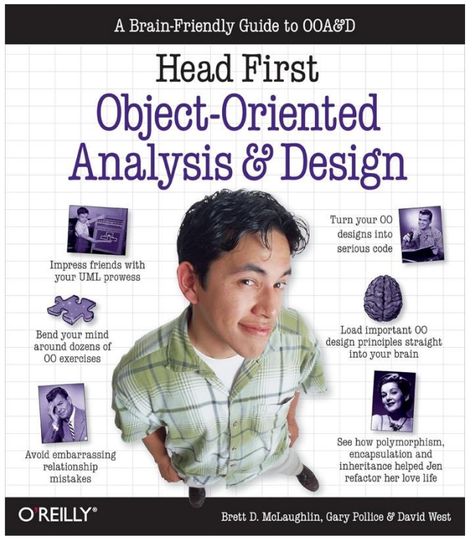


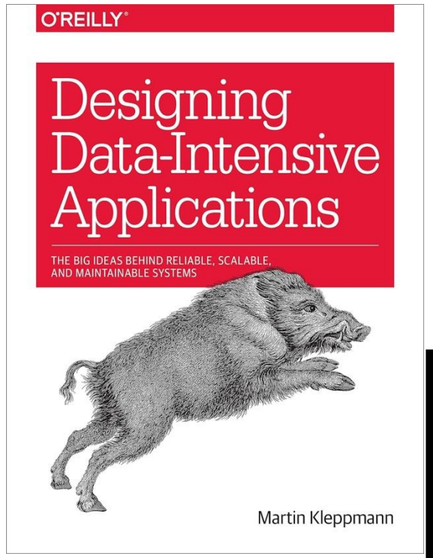












SWINBURNE UNIVERSITY OF TECHNOLOGY