

ITEC 120

Lecture 30
Association / Aggregation

Review

- Questions?
- OO Conversion

Association / Aggregation


Objectives

- Connect objects together
 - Sole source
 - Community source

Association / Aggregation


Relationships

- Objects Attributes ↔ Objects in RL



Radius
ColorScheme

- Relationships in RL



Jimmy's Ball

Association / Aggregation

Key

- Relationships store a way for information to flow between parties (1 vs 2way)
- Arrays
 - This variable allows you access X sources of info
- Variables
 - This variable points to Y information

Association / Aggregation

Aggregation

Relationship type #1

```

public static void main(String[] args)
{
    Jimmy one = new Jimmy();
    one.brag();
}

public class Jimmy
{
    public double height;
    private Ball myBall;
    Jimmy()
    { myBall = new Ball(); }
    public void brag()
    { System.out.println("My ball is " + myBall.radius + "high"); }
}

public class Ball
{
    int radius;
    String colorscheme;
}
    
```

- This object has a


Has a

Association / Aggregation

Another example

```


public class SodaMachine
{
    public void stock(); //Ensures 15 of each type
    public void sell(); //Sell a soda
    private mountainDew[] dewes;
    private cocacola[] colas;
    private sprite[] sprites;
}
    
```



Association / Aggregation

Benefits

- Enforces physical restrictions in code



The only way to get a soda is when you put money in


Real life
If you break the rule, the police stop you

Virtual life
If you break the rule, javac stops you

Association / Aggregation

Problem

30 machines x \$2500 each == \$75,000
Every 3-4 years!



- Sharing
 - Expensive resource
 - Multiple parties sharing
 - Aggregation doesn't cut it

Computer labs are expensive
I can't have my own lab, I have to share with other professors!

Association / Aggregation

Association

```

public class ComputerLab
{
    Mac[] machines;
}

public class Professor
{
    private ComputerLab myLab;
    public void setLab(ComputerLab aLab)
    {
        myLab = aLab;
    }
}

public static void main(String[] args)
{
    ComputerLab DH225 = new ComputerLab();
    Professor[] itec = new Professor[16];
    itec[0] = new Professor();
    itec[1] = new Professor();
    itec[0].setLab(DH225);
    itec[1].setLab(DH225);
}
    
```

Shared by many

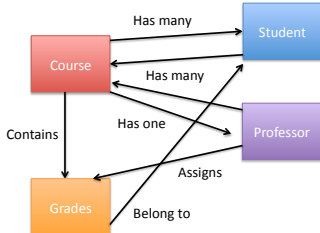
Association / Aggregation

Splitting hairs

- Syntax wise
 - Aggregation = Private variable w/out a setter
 - Association = Private/Public variable w/ a setter/getter
- Meaning wise
 - Non-trivial
 - Exclusive relationship
 - Shared resource

Association / Aggregation

Case study



```

classDiagram
    Course --> Student : Has many
    Course --> Professor : Has many
    Professor --> Grades : Assigns
    Professor --> Course : Belong to
    Course --> Grades : Contains
    
```

Association / Aggregation

Case study

*Assume all public variables

- Student record system in Java

```

public class Student
{
    private String Name;
    public String ferpaget/SetName();
    Courses[] listOfClasses;
}

public class Grades
{
    Student aStudent;
    double num;
}

public class Course
{
    String name;
    Student[] students;
    Grade[] studentGrades;
    Professor prof;
}

public class Professor
{
    String name;
    Course[] courses;
}
    
```

Association / Aggregation

Creation

```

Professor Ray = new Professor();
Ray.name = "Dr. Ray";
Student[] students = new Student[35];
//Code to fill student array
Course ITEC120 = new Course();
ITEC120.prof = Ray;
ITEC120.students = students;
Grades[] grades = new Grades[35];
grades[0] = new Grades();
grades[0].aStudent = students[0];
grades[0].num= 80;
//Etc...
ITEC120.studentGrades = grades;
    
```

Associates

Association / Aggregation

Usage

- For each student in ITEC 120 compute the average grade in all of their courses

```

double theGrades=0.0;
for (int i=0; i<ITEC120.students.length; i++)
{
    double gradeTotal=0.0;
    for (int j=0; j<ITEC120.students[i].studentGrades.length; j++)
    {
        gradeTotal+= ITEC120.students[i].studentGrades[j].num;
    }
    gradeTotal/=ITEC120.students[i].studentGrades.length;
    theGrades += gradeTotal;
}
System.out.println(theGrades/ITEC120.students.length);
    
```

Association / Aggregation

Only possible b/c of association

Case study 2

```

classDiagram
    class Bank
    class Driver
    class BankAccounts
    class Signers
    Bank --> Driver
    Bank --> BankAccounts : Has many
    BankAccounts --> Signers : Have many
    
```

Association / Aggregation

Over-engineering

```

public double[] amount;
public String[] names;
//Read in information
for (int i=0; i<amount.length; i++)
{
    if (amount[i] < 0)
    {
        amount[i] -= 50;
        System.out.println(
            names[i].substring(0,names[i].indexOf(' '))
            + " is overdrawn");
    }
}
    
```

Parallel arrays
Loop
Conditional

```

classDiagram
    class Signer
    class BankAccount
    class Bank
    class Driver
    Signer --> BankAccount
    BankAccount --> Bank
    Bank --> Driver
    
```

Versus

Immense complexity / overhead for the problem

Association / Aggregation

Which tool?

Setup time: <1 min

Setup time: 5-30 mins

Simple problem

Big problem

Association / Aggregation

Review

- Relationships are important
- Association
- Aggregation

Association / Aggregation