RADFORD UNIVERSITY

ITEC 120

Lecture 31
Design Patterns

Review

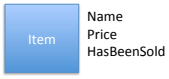
- Questions?
- Association / Aggregation

Objectives

- Describe methods of solving problems
- Templates for creating code
- Example usage

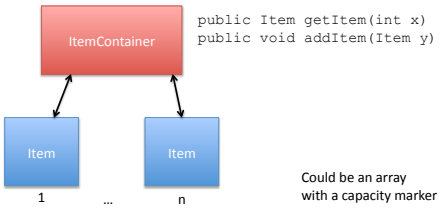
Bottom up

- Object
  - Contains information about a real life or virtual concept
  - May or may not contain arrays



Container

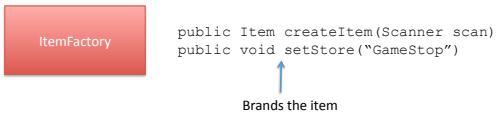
- Holds and provides access to objects



Could be an array with a capacity marker

Factory

- Creates objects, may configure them



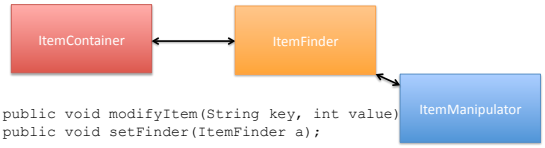
Brands the item

Finder


```
public Item findItem(String name)
public void setContainer(ItemContainer a)
```

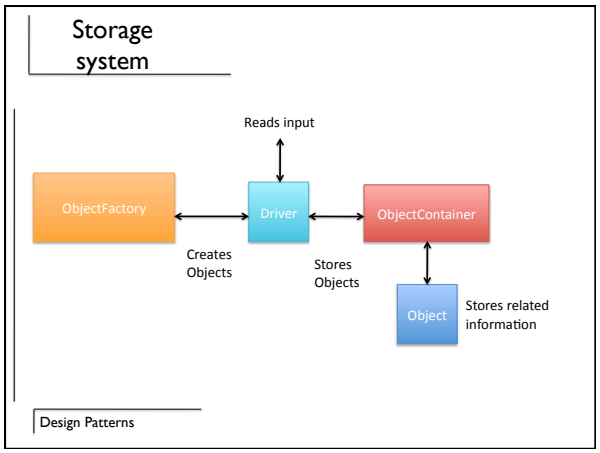
Search engine for Objects

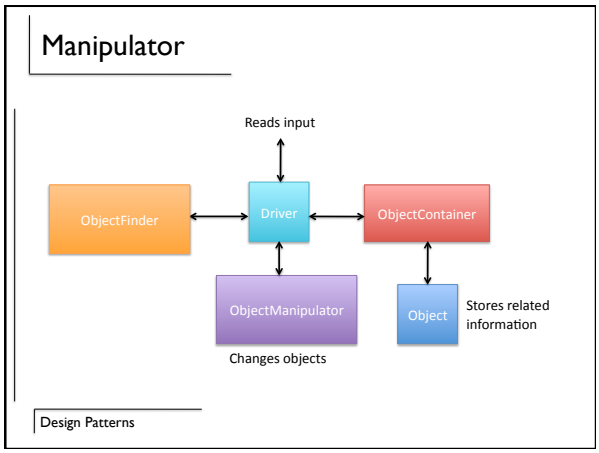
Manipulator

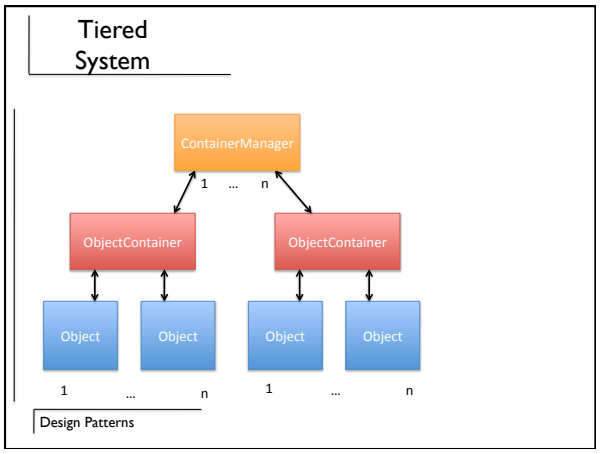


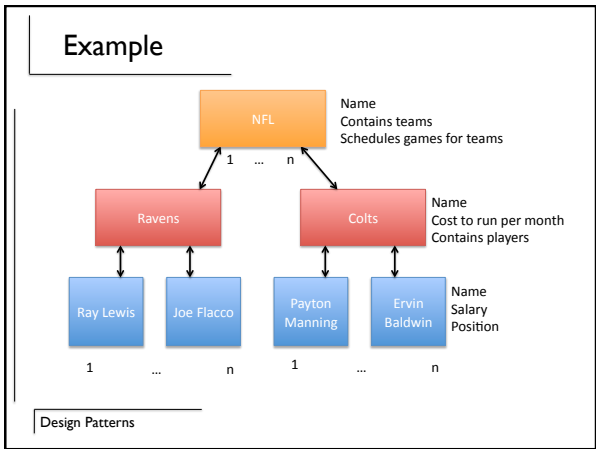

```
public void modifyItem(String key, int value)
public void setFinder(ItemFinder a);
```

Easy way to update objects w/out having to worry about an array each time


Storage system
Reads input
ObjectFactory
Driver
ObjectContainer
Creates Objects
Stores Objects
Object
Stores related information
Design Patterns


Manipulator
Reads input
ObjectFinder
Driver
ObjectContainer
ObjectManipulator
Changes objects
Object
Stores related information
Design Patterns


Tiered System
ContainerManager
1 ... n
ObjectContainer
ObjectContainer
Object
Object
Object
Object
1 ... n
1 ... n
Design Patterns


Example
NFL
Name
Contains teams
Schedules games for teams
1 ... n
Ravens
Colts
Name
Cost to run per month
Contains players
Ray Lewis
Joe Flacco
Payton Manning
Ervin Baldwin
Name
Salary
Position
1 ... n
1 ... n
Design Patterns

Case study

- The world of Pokemon

Cute looking monsters
Cartoon teenagers following
standard clichés that use monsters
Collect / Train monsters for league battles
League = person w/ specific monsters
Teenagers compete against leagues

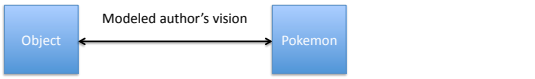
Design Patterns

Why is it popular

- Game + products

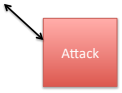
Design Patterns

Design




```
public class Pokemon
{
    public int HP;
    public int level;
    public String name;
    public String type;
    public Attack[] attacks;
}
```

Name
HP
Type
Array Of Attacks




```
public class Attack
{
    public int damage;
    public String name;
    public String type;
}
```

Name
Amount of Damage
Element type

Design Patterns

Integration

The game works by storing all of the information in a file and using the Scanner class to fill in the information

```
Attack punch = new Attack();
punch.name = "Punch";
punch.type = "normal";
punch.damage = 10;
Attack fire = new Attack();
fire.name = "fireBreath";
fire.type = "fire";
fire.damage=20;
```

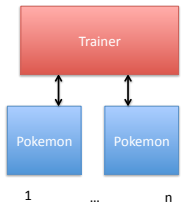
Association

```
Pokemon one= new Pokemon();
one.name = "Charizard";
one.type = "fire";
one.hp=20;
one.level=1;
one.attacks = new Attack[2];
one.attack[0] = punch;
one.attack[1] = fire;

Pokemon two = new Pokemon();
two.name="rat";
two.type="normal";
two.hp=10;
two.level=1;
two.attacks = new Attack[1];
two.attacks[0] = punch;
```

Design Patterns

Design

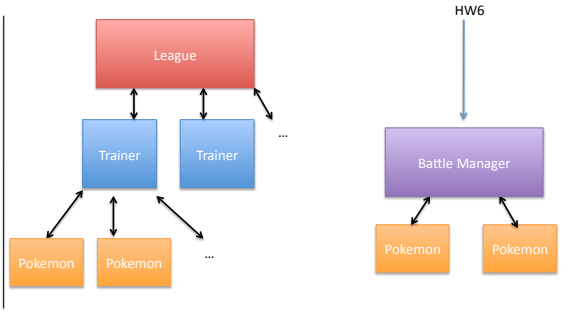


Name
Pokémon Collection

```
public class Trainer
{
    String name;
    Pokemon[] monsters;
}

Trainer ash = new Trainer();
ash.monsters = new Pokemon[2];
ash.monsters[0] = one;
ash.monsters[1] = two;
```


Design



Code

```
public class League
{
    public void addTrainer(Trainer recruit)
    {
        trainers[numTrainers] = recruit;
        numTrainers++;
    }
    public void challenge(Trainer challenger)
    {
        int win=0;
        for (int i=0; i<trainers.length; i++)
        {   win = manager.battle(trainers[i], challenger);
            if (win == 1) break;  }
        System.out.println("0 home, 1 challenge:  " + win);
    }
    BattleManager manager;
    Trainer[] trainers;
    int numTrainers;}
```


Summary

- Patterns are your friend
  - Nails, screws, bolts, hammers, screwdrivers, and wrenches
- Combinations
- Problem ⇔ Design ⇔ Patterns ⇔ Code