

## Last time ...

### Data Analysis, Design and Representation

- Simple data
  - 1 23 3.5 "Joe's Crab Shack"
- Compound data

```
class Fish {
  int weight;
  int age;
  String color;
}

Fish(int weight, int age, String color) {
  this.weight=weight;
  this.age=age;
  this.color=color;
}

//Give the difference between givenSize and this fish's weight
int amountBigger(int givenSize){
  ...
}

//Determine if this fish's weight is larger than givenSize
boolean biggerThan(int givenSize){
  ...
}
```

Fish
int weight
int age
String color

## Last time ...

### Design recipe

- Analyze data
- Purpose/Header
- Examples
- Template
- Body
- Test

## Design and Representation Matching

Fish
int weight
int age
String color
int amountBigger(int givenSize)
boolean biggerThan(int givenSize)

```
class Fish {
  int weight;
  int age;
  String color;
}

Fish(int weight, int age, String color) {
  this.weight=weight;
  this.age=age;
  this.color=color;
}

int amountBigger(int givenSize){
  ...
}

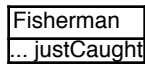
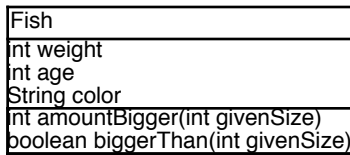
boolean biggerThan(int givenSize){
  ...
}
}
```

## Using classes as data

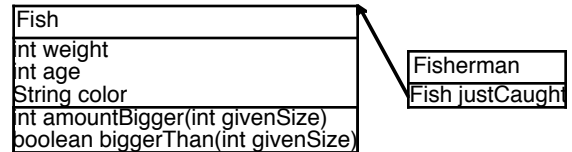
- All fishermen want to know whether their fish weighs more than a given weight

### Using classes as data

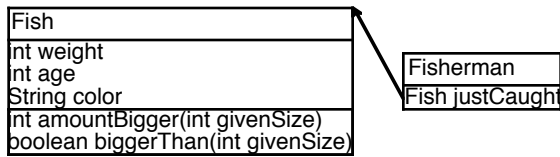
- All fishermen want to know whether their fish weighs more than a given weight



### Fish and Fisherman

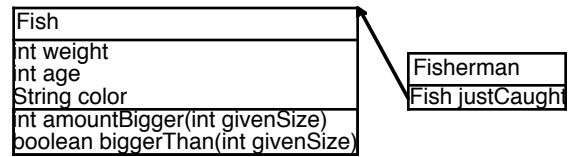


### Fish and Fisherman



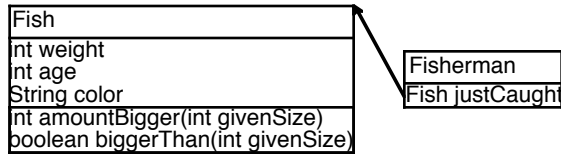
- Design recipe: template step
  - Record the pieces of information available

### Fish and Fisherman



- Design recipe: template step
  - Record the information available and wherever there is an arrow from data in the design, expect a method call

## Fish and Fisherman




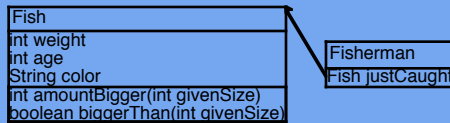
- Design recipe: template step
  - Record the information available and wherever there is an arrow from data in the design, expect a method call
- Template for Fisherman: // ... justCaught.FishMethod( ... ) ...

## Back to Fisherman measuring fish

- All fishermen want to know whether their fish weighs more than a given weight
- Practice

## Data

- Simple data
  - 1 23 3.5 "Joe's Crab Shack"
- Compound data
  - 
- Compound data using other compound data



## More Practice

Is a Fisherman's fish larger than a given Fish?

### Feeding a fish

- An aquarium manager needs to feed the fish.
  - When a fish is fed, we produce a fish which weighs a given amount more

### Feeding a fish

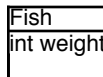
- An aquarium manager needs to feed the fish.
  - When a fish is fed, we produce a fish which weighs a given amount more

`Fish ( weight = 4 )`

+ 2 -->

`Fish ( weight = 6 )`

### Feeding a fish



### Feeding a fish

```
class Fish {  
    int weight;  
  
    Fish( int weight ) {  
        this.weight = weight;  
    }  
}
```

### Feeding a fish

```
class Fish {
  int weight;

  Fish( int weight ) {
    this.weight = weight;
  }

  //To produce a fish that is w heavier than this Fish
  Fish feedFish( int w ) {
  }
}
```

### Feeding a fish

```
class Fish {
  int weight;

  Fish( int weight ) {
    this.weight = weight;
  }

  //Examples
  //new Fish(4).feedFish(2) -> new Fish(6)
  //To produce a fish that is w heavier than this Fish
  Fish feedFish( int w ) {
  }
}
```

### Feeding a fish

```
class Fish {
  int weight;

  Fish( int weight ) {
    this.weight = weight;
  }

  //Examples
  //new Fish(4).feedFish(2) -> new Fish(6)
  //To produce a fish that is w heavier than this Fish
  Fish feedFish( int w ) {
    ... this.weight ....
  }
}
```

### Feeding a fish

```
class Fish {
  int weight;

  Fish( int weight ) {
    this.weight = weight;
  }

  //Examples
  //new Fish(4).feedFish(2) -> new Fish(6)
  //To produce a fish that is w heavier than this Fish
  Fish feedFish( int w ) {
    return new Fish(this.weight + w);
  }
}
```

### Feeding a fish

```
class Fish {
    int weight;

    Fish( int weight ) {
        this.weight = weight;
    }

    //Examples
    //new Fish(4).feedFish(2) -> new Fish(6)
    //To produce a fish that is w heavier than this Fish
    Fish feedFish( int w ) {
        return new Fish(this.weight + w);
    }
}
new Fish(4).feedFish(2)
```

### Practice

- A fisherman catches a new fish
- A fisherman cuts a piece off his fish, decreasing the weight

### Representing Fish food

- Fish at the aquarium eat Plants and Plankton, update feedFish accordingly
  - A plant's weight varies by how many leaves it has
  - A plankton can't be eaten if the Fish can't catch it

### Representing Fish food

- Fish at the aquarium eat Plants and Plankton, update feedFish accordingly
  - A plant's weight varies by how many leaves it has
  - A plankton can't be eaten if the Fish can't catch it
- Design Plant and Plankton

### Plant & Plankton

Plant
int leaves

Plankton
int weight
boolean moves

### Updating feedFish

Fish
int weight
• Fish feedFish(... givenFood)

### How can we write feedFish?

- Make Fish eat Food

Food
boolean isMoving()
• int getWeight()

### How can we write feedFish?

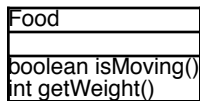
- Make Fish eat Food

Food
boolean isMoving()
• int getWeight()

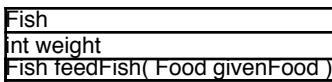
Fish
int weight
• Fish feedFish( Food givenFood )

### How can we write feedFish?

- Make Fish eat Food

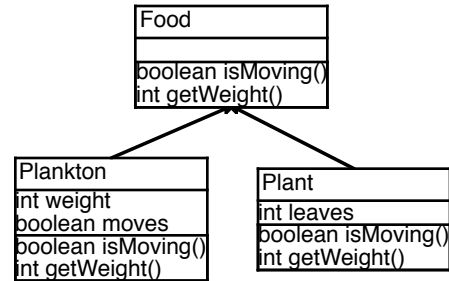


- 

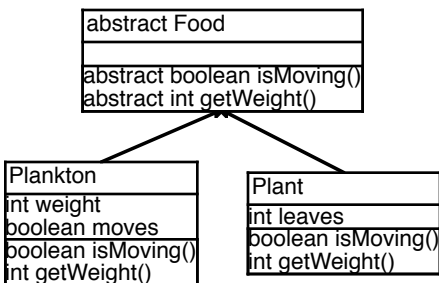


- How to connect Plants and Plankton to Food?

### Food, Plants & Plankton



### Food, Plants & Plankton



### Representing Food in Java - inheritance

```

abstract class Food {
    //To determine if the food moves
    abstract boolean isMoving();
    //To report the weight of the food
    abstract int getWeight();
}

class Plankton extends Food {
    int weight;
    boolean moving;
    Plankton(int weight,boolean moving){
        this.weight = weight;
        this.moving = moving;
    }
    // To determine if the Plankton moves
    // new Plankton(3,true).isMoving() == true
    boolean isMoving() {
        return ...
    }
    //To report the weight of the Plankton
    //new Plankton(3,true).getWeight() == 3
    int getWeight() { ... }
}
    
```

## isMoving

- When is a Plant moving?
- When is a Plankton moving?
- How much do they weigh?

## Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
    }
}
```

## Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //new Fish(2).feedFish(new Plant(3)) -> new Fish(8)
    //new Fish(3).feedFish(new Plankton(3,true)) -> new Fish(3)
    //new Fish(3).feedFish(new Plankton(3,false)) -> new Fish(6)
    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
    }
}
```

## Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //new Fish(2).feedFish(new Plant(3)) -> new Fish(8)
    //new Fish(3).feedFish(new Plankton(3,true)) -> new Fish(3)
    //new Fish(3).feedFish(new Plankton(3,false)) -> new Fish(6)
    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
        // ... this.weight ... givenFood ...
    }
}
```

### Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //new Fish(2).feedFish(new Plant(3)) -> new Fish(8)
    //new Fish(3).feedFish(new Plankton(3,true)) -> new Fish(3)
    //new Fish(3).feedFish(new Plankton(3,false)) -> new Fish(6)
    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
        // this.weight ... givenFood ...
        if ( ... )
            ....
        else
            ....
    }
}
```

### Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //new Fish(2).feedFish(new Plant(3)) -> new Fish(8)
    //new Fish(3).feedFish(new Plankton(3,true)) -> new Fish(3)
    //new Fish(3).feedFish(new Plankton(3,false)) -> new Fish(6)
    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
        // this.weight ... givenFood ...
        if ( givenFood.isMoving() )
            ....
        else
            ....
    }
}
```

### Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //new Fish(2).feedFish(new Plant(3)) -> new Fish(8)
    //new Fish(3).feedFish(new Plankton(3,true)) -> new Fish(3)
    //new Fish(3).feedFish(new Plankton(3,false)) -> new Fish(6)
    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
        // this.weight ... givenFood ...
        if ( givenFood.isMoving() )
            return this;
        else
            ....
    }
}
```

### Back to updating Fish's feed

Fish eat Food, but they can't eat food that they can't catch

```
class Fish {
    int weight;
    Fish(int weight) {
        this.weight = weight;
    }

    //new Fish(2).feedFish(new Plant(3)) -> new Fish(8)
    //new Fish(3).feedFish(new Plankton(3,true)) -> new Fish(3)
    //new Fish(3).feedFish(new Plankton(3,false)) -> new Fish(6)
    //To produce a fish larger than this fish, based on givenFood
    Fish feed( Food givenFood ) {
        // this.weight ... givenFood ...
        if ( givenFood.isMoving() )
            return this;
        else
            return new Fish(this.weight + givenFood.getWeight());
    }
}
```

### Fisherman, Fish, and Tires

- Fishermen don't always catch fish...
  - Sometimes they catch tires instead

### Fisherman, Fish, and Tires

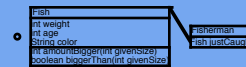
- Fishermen don't always catch fish...
  - Sometimes they catch tires instead
- A fisherman wants to eat his catch, he needs to know if it can be eaten first

### Fisherman, Fish, and Tires

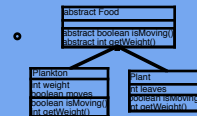
- Fishermen don't always catch fish...
  - Sometimes they catch tires instead
- A fisherman wants to eat his catch, he needs to know if it can be eaten first
- Practice

### Data

- Simple data
  - 1 23 3.5 "Joe's Crab Shack"
- Compound data
- Compound data using other compound data



- Conditional Compound Data



**New practice**

- A pizza can have toppings
  - Anchovies
  - Olives
  - Extra Cheese

**New practice**

- A pizza can have toppings
  - Anchovies
  - Olives
  - Extra Cheese
- A restaurant worker wants the of price a one-topping pizza, where all toppings have different prices

**More than one Fish**

- Fishermen normally catch more than one Fish
- How can we improve Fisherman?

**More than one Fish**

- Fishermen normally catch more than one Fish
- How can we improve Fisherman?

Fisherman
Fish justCaught
Fish caughtBeforeThat
...

### More than one Fish

- Fishermen normally catch more than one Fish
- How can we improve Fisherman?

Fisherman
Fish justCaught
Fish caughtBeforeThat
...

- What about the Fisherman who only caught one Fish, or 3 Fish

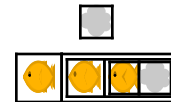
### A Catch of Fish

- Need to represent 0 Fish caught, 1 Fish caught, 2 Fish, etc

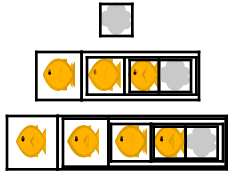
### Stretchy boxes



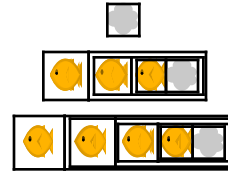
### Stretchy boxes



### Stretchy boxes



### Stretchy boxes

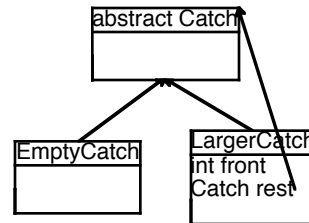


- To make things easier, let's represent a Fish as its weight for now

### A Catch

- A Catch is either
  - an Empty Catch
  - or one weight and a Catch

### A Catch



### Catch in Java

```
abstract class Catch {  
}  
  
class EmptyCatch extends Catch {  
    EmptyCatch() {}  
}  
  
class LargerCatch extends Catch {  
    int front;  
    Catch rest;  
    LargerCatch( int front, Catch rest) {  
        this.front = front;  
        this.rest = rest;  
    }  
}
```

### Making a Catch

- A fisherman has caught 0 fish
  - new EmptyCatch()

### Making a Catch

- A fisherman has caught 0 fish
  - new EmptyCatch()
- A fisherman has caught 1 fish, weighing 3
  - new LargerCatch(3, new Catch())

### Making a Catch

- A fisherman has caught 0 fish
  - new EmptyCatch()
- A fisherman has caught 1 fish, weighing 3
  - new LargerCatch(3, new Catch())
- A fisherman has caught 2 fish, weighing 3 and 5 pounds

## How heavy is the catch?

```
abstract class Catch {
    //Determines the total weight of this catch
    abstract int weight();
}

class EmptyCatch extends Catch {
    EmptyCatch() {}
    //Determines the total weight of this EmptyCatch
    int weight() {
        return 0;
    }
}

class LargerCatch extends Catch {
    int front;
    Catch rest;
    LargerCatch( int front, Catch rest) {
        this.front = front;
        this.rest = rest;
    }
    //Determines the total weight of this LargerCatch
    int weight() {
        return this.front + this.rest.weight();
    }
}
```

## Data

- Simple data
- Compound data
- Compound data using other compound data



- Conditional Compound Data



- Recursive Data

