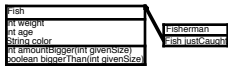
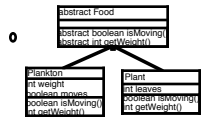


Where were we?

- Introduced new forms data
 - Compound data that uses other compound data



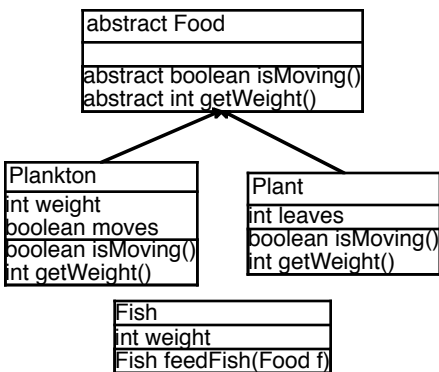
- Conditional compound data



Recap cont.

- Methods using two pieces of compound data
 - An arrow from data in the design means we need a method call in the template
 - Fisherman may only access the methods of his Fish
- Methods that produce compound data
 - Fish feedFish(int w) { ... }
 - Fisherman catchFish(Fish f) { ... }
- abstract methods for Conditional compound data
 - must be implemented using information in the subclasses

Updating feedFish with Food



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)
  //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
      ....
  }
}
```

if

- If Sally has enough money, she can buy a new car. Otherwise she'll have to keep her old one

```
if (enoughMoney())  
    return new Car();  
else  
    return oldCar;
```

if

```
if ( 1<2 )  
    return 3+2;  
else  
    return 3-2;  
  
if ( true )  
    return 3+2;  
else  
    return 3-2;  
  
return 3+2;
```

if

```
if ( 10<2 )  
    return 3+2;  
else  
    return 3-2;  
  
if ( false )  
    return 3+2;  
else  
    return 3-2;  
  
return 3-2;
```

Finishing feedFish

```
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  
            ....  
    }  
}
```

Finishing feedFish

```
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
      ....
  }
}
```

Finishing feedFish

```
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 new Fish(this.weight);
    else
      ....
  }
}
```

Finishing feedFish

```
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 new Fish(this.weight);
    else
      return new Fish(this.weight + givenFood.getWeight());
  }
}
```

Fisherman, Fish, and Tires

- Fishermen don't always catch fish...
 - Sometimes they catch tires instead
- A fisherman wants to know if his catch is valuable, a Tire in good condition is valuable, and an edible Fish is valuable
- When fishermen catch valuable items, they say "Yippee". Otherwise they say "boohoo"
- Practice

More than one Fish

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

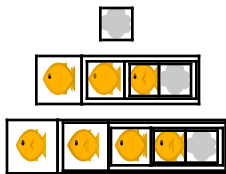
Fisherman
Fish justCaught
Fish caughtBeforeThat

- 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

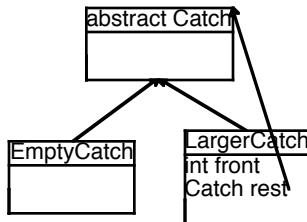


- To make things easier, let's represent a Fish as its weight for now

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

Data

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



- Conditional Compound Data

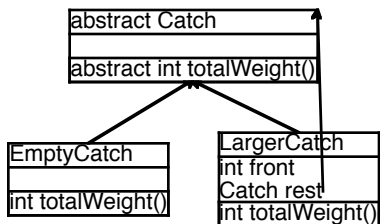


- Recursive Data



Recursive computations

- The fisherman wants to know how heavy his catch is



Designing totalWeight()

- Purpose & header

```
abstract class Catch {
    //To report the total weight of this Catch
    abstract int totalWeight();
}

class EmptyCatch extends Catch {
    EmptyCatch() {}
    //To report the total weight of this EmptyCatch
    int totalWeight() {}
}

class LargerCatch extends Catch {
    int front;
    Catch rest;
    LargerCatch( int front, Catch rest) {
        this.front = front;
        this.rest = rest;
    }
    //To report the total weight of this LargerCatch
    int totalWeight() {}
}
```

Examples

```
abstract class Catch {
    //To report the total weight of this Catch
    abstract int totalWeight();
}

class EmptyCatch extends Catch {
    EmptyCatch() {}
    // new EmptyCatch().totalWeight() == 0
    //To report the total weight of this EmptyCatch
    int totalWeight() {
    }
}

class LargerCatch extends Catch {
    int front;
    Catch rest;
    LargerCatch( int front, Catch rest) {
        this.front = front;
        this.rest = rest;
    }
    //new LargerCatch(3,new EmptyCatch()).totalWeight() == 3
    //new LargerCatch(5,new LargerCatch(3,new EmptyCatch())) == 8
    //To report the total weight of this LargerCatch
    int totalWeight() {
    }
}
```

Body - template

```
abstract class Catch {
    //To report the total weight of this Catch
    abstract int totalWeight();
}

class EmptyCatch extends Catch {
    // new EmptyCatch().totalWeight() == 0
    //To report the total weight of this EmptyCatch
    int totalWeight() {
        //...
    }
}

class LargerCatch extends Catch {
    int front;
    Catch rest;
    //new LargerCatch(3,new EmptyCatch()).totalWeight() == 3
    //new LargerCatch(5,new LargerCatch(3,new EmptyCatch())) == 8
    //To report the total weight of this LargerCatch
    int totalWeight() {
        //... this.front ... this.rest.CatchMethod(...)
    }
}
```

Body - EmptyCatch

```
abstract class Catch {
    //To report the total weight of this Catch
    abstract int totalWeight();
}

class EmptyCatch extends Catch {
    // new EmptyCatch().totalWeight() == 0
    //To report the total weight of this EmptyCatch
    int totalWeight() {
        return 0;
    }
}

class LargerCatch extends Catch {
    //new LargerCatch(3,new EmptyCatch()).totalWeight() == 3
    //new LargerCatch(5,new LargerCatch(3,new EmptyCatch())) == 8
    //To report the total weight of this LargerCatch
    int totalWeight() {
        //... this.front ... this.rest.CatchMethod(...)
    }
}
```

Body - LargerCatch

```
abstract class Catch {
    //To report the total weight of this Catch
    abstract int totalWeight();
}

class EmptyCatch extends Catch {
    // new EmptyCatch().totalWeight() == 0
    //To report the total weight of this EmptyCatch
    int totalWeight() {
        return 0;
    }
}

class LargerCatch extends Catch {
    //new LargerCatch(3,new EmptyCatch()).totalWeight() == 3
    //new LargerCatch(5,new LargerCatch(3,new EmptyCatch())) == 8
    //To report the total weight of this LargerCatch
    int totalWeight() {
        //... this.front ... this.rest.CatchMethod( ... )
        return this.front + this.rest.totalWeight();
    }
}
```

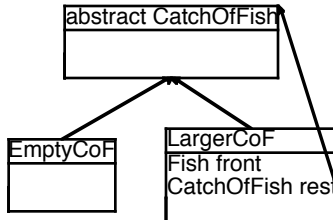
Stepping through

- new LargerCatch(3, new EmptyCatch()).totalWeight()
 - return LargerCatch.front + EmptyCatch.rest.totalWeight();
 - return 3 + LargerCatch.rest.totalWeight();
 - return 3 + EmptyCatch.totalWeight();
 - return 3 + return 0;
 - return 3 + 0;
 - return 3;
 - 3

How many fish are in the catch?

- practice

Putting the Fish in Catch



Putting the Fish in Catch

```
abstract class CatchOfFish {
}

class EmptyCoF extends CatchOfFish {
    EmptyCoF() {}
}

class LargerCoF extends CatchOfFish {
    Fish front;
    CatchOfFish rest;
    LargerCoF( Fish front, CatchOfFish rest) {
        this.front = front;
        this.rest = rest;
    }
}
```

Methods on lists

- Are all of the weights over 10 pounds? - allOver10lbs

Building a new catch

Our fisherman (who is a Fish lover) wants to feed all of his Fish, producing a new Catch of fish that all weigh more

```
abstract class Catch {
    //To produce a Catch with weights w larger than this Catch
    abstract Catch feedAll( int w );
}

class EmptyCatch extends Catch {
    //To produce a Catch with weights w larger than this EmptyCatch
    Catch feedAll( int w ) {
    }
}

class LargerCatch extends Catch {
    //To produce a Catch with weights w larger than this LargerCatch
    Catch feedAll( int w ) {
    }
}
```

Finishing feedAll in DrScheme

A final feedAll - for reference

```
abstract class Catch {
  //To produce a Catch with weights w larger than this Catch
  abstract Catch feedAll( int w );
}
class EmptyCatch extends Catch {
  //new EmptyCatch().feedAll(4) -> new EmptyCatch()
  //To produce a Catch with weights w larger than this EmptyCatch
  Catch feedAll( int w ) {
    // ...
    return new EmptyCatch();
  }
}
class LargerCatch extends Catch {
  //new LargerCatch(4,new LargerCatch( 3, new EmptyCatch())).feedAll(3)
  //  new LargerCatch( 7, new LargerCatch( 6, new EmptyCatch() ))
  //To produce a Catch with weights w larger than this LargerCatch
  Catch feedAll( int w ) {
    // ... this.first ... this.rest.CatchMethod( ... )
    return new LargerCatch( this.first + w,this.rest.feedAll(w);
  }
}
```

A new example

- Represent a Pizza that can have more than one topping
- Calculate the full cost of a pizza.
- Determine which of two pizzas is healthier (has fewer calories)