COMP 401
INTERFACES

Instructor: Prasun Dewan
PREREQUISITE

- State Properties
INTERFACES

- Define contracts between our users and implementers
- Optional – they may not be used
- Good style to use them
**Motivation: Two Ways of Doing the BMI Spreadsheet**

- ABMISpreadsheet is one way to implement the spreadsheet user-interface
- Let us create AnotherBMISpreadsheet to illustrate another way
- Difference is in number of variables used
public class ABMISpreadsheet {
    double height;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
    }
    double weight;
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
    }
    public double getBMI() {
        return weight/(height*height);
    }
}
ABMISpreadsheet Instance

weight
getWeight() calls
setWeight() calls
writes
new weight

height
getHeight() calls
setHeight() calls
writes
new height

getBMI() calls
reads

ObjectEditor
Another BMISpreadsheet

AnotherBMISpreadsheet Instance

weight
- reads
- writes
  - calls
  - new weight

setWeight()
- calls

getHeight()
- calls

setHeight()
- calls

bmi
- reads
  - calls
  - new height

getBMI()
- calls

ObjectEditor
public void setWeight(double newWeight) {
    weight = newWeight;
    bmi = weight / (height*height);
}
**setHeight()**

```java
public void setHeight(double newHeight) {
    height = newHeight;
    bmi = weight / (height * height);
}
```
METHODS THAT CHANGE

ABMISpreadsheet Instance

weight

setWeight()

writes

calls

new weight

height

setHeight()

writes

calls

new height

bmi

getBMI()

reads

ObjectEditor
public double getBMI() {
    return bmi;
}
public class AnotherBMISpreadsheet {
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = weight/(height*height);
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = weight/(height*height);
    }
    public double getBMI() {
        return bmi;
    }
}
ABMISpreadsheet Instance

- **weight**
  - read
  - write
  - `getWeight()`
  - `setWeight()`

- **height**
  - read
  - write
  - `getHeight()`
  - `setHeight()`

- **bmi**
  - read

ObjectEditor

- **calls**
- **writes**

- **new weight**
- **new height**

**GRAPHICAL ALGORITHM**
public class AnotherBMISpreadsheet {
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = weight/(height*height);
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = weight/(height*height);
    }
    public double getBMI() {
        return bmi;
    }
}
OBJECTEDITOR USER INTERFACES
**SIMILARITIES IN THE TWO CLASSES**

**public class** AnotherBMISpreadsheet {
    **double** height, weight, bmi;
**public double** getHeight() {
    return height;
}
**public void** setHeight(**double** newHeight) {
    height = newHeight;
    bmi = weight/(height*height);
}
**public double** getWeight() {
    return weight;
}
**public void** setWeight(**double** newWeight) {
    weight = newWeight;
    bmi = weight/(height*height);
}
**public double** getBMI() {
    return bmi;
}
**}

**public class** ABMISpreadsheet {
    **double** height;
**public double** getHeight() {
    return height;
}
**public void** setHeight(**double** newHeight) {
    height = newHeight;
}
**double** weight;
**public double** getWeight() {
    return weight;
}
**public void** setWeight(**double** newWeight) {
    weight = newWeight;
}
**public double** getBMI() {
    return weight/(height*height);
}
**}

---

**Same headers, signatures**

**Signature = Header – parameter names**

**OE Beautified Signature**

---

**[ABMISpreadsheet](#)**

**Parameters of Calculate BMI**

**Parameter 1:**
- **Type:** double
- **Value:** 74.98

**Parameter 2:**
- **Type:** double
- **Value:** 1.94

**Calculate BMI(double,double)**
REAL-WORLD ANALOGY

Corvette Specification

manufactured by

manufactured by

implements

implements
**INTERFACE**

```java
public interface BMISpreadsheet {
    public double getWeight();
    public void setWeight(double newVal);

    public double getHeight();
    public void setHeight(double newVal);

    public double getBMI();
}
```
public class AnotherBMISpreadsheet implements BMISpreadsheet {
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = weight/(height*height);
    }
    ...
}

public interface BMISpreadsheet {
    public double getWeight();
    public void setWeight(double newVal);
    public double getHeight();
    public void setHeight(double newVal);
    public double getBMI();
}
**Interface**

- BMISpreadsheet implements ABMISpreadsheet
- ABMISpreadsheet instance of ABMISpreadsheet instance
- AnotherBMISpreadsheet implements ABMISpreadsheet
- AnotherBMISpreadsheet instance of AnotherBMISpreadsheet instance
Using Interfaces to Classify

BMISpreadsheet implements

ABMISpreadsheet

instance of

BMISpreadsheet instance

AnotherBMISpreadsheet

instance of

BMISpreadsheet instance

BMISpreadsheet instance
USING CAR SPECIFICATIONS TO CLASSIFY

Corvette Specification

Corvette manufactured by

Corvette

Corvette

Corvette manufactured by

Corvette

Corvette
Using Interfaces to Type

```java
public class BMISpreadsheetUser {
    public static void main(String[] args) {
        BMISpreadsheet bmiSpreadsheet = new ABMISpreadsheet(1.77, 75);
        System.out.println(bmiSpreadsheet.getBMI());
        bmiSpreadsheet = new AnotherBMISpreadsheet();
        bmiSpreadsheet.setHeight(1.77);
        bmiSpreadsheet.setWeight(75);
        System.out.println(bmiSpreadsheet.getBMI());
    }
}
```

- Interface as type
- Same variable assigned instances of two different classes
public class BMISpreadsheetUser {
    public static void main(String[] args) {
        BMISpreadsheet bmiSpreadsheet = new ABMISpreadsheet(1.77, 75);
        System.out.println(bmiSpreadsheet.obtainBMI());
        bmiSpreadsheet = new AnotherBMISpreadsheet();
        bmiSpreadsheet.setHeight(1.77);
        bmiSpreadsheet.setWeight(75);
        System.out.println(bmiSpreadsheet.getBMI());
    }
}
**INTERFACE METHODS CONSIDERED IN TYPE CHECKING**

```java
public class BMISpreadsheetUser {
    public static void main(String[] args) {
        BMISpreadsheet bmiSpreadsheet = new ABMISpreadsheet(1.77, 75);
        System.out.println(bmiSpreadsheet.getBMI());
        bmiSpreadsheet = new AnotherBMISpreadsheet();
        bmiSpreadsheet.setHeight(1.77);
        bmiSpreadsheet.setWeight(75);
        System.out.println(bmiSpreadsheet.getBMI());
    }
}
```

Not defined in interface so illegal, even though defined in class.
public class ABMISpreadsheetAndCalculator implements ??? {
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = calculateBMI(height, weight);
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = calculateBMI(height, weight);
    }
    public double getBMI() {
        return bmi;
    }
    public double calculateBMI(double height, double weight) {
        return weight/(height*height);
    }
}
public class ABMISpreadsheetAndCalculator implements BMISpreadsheet, BMICalculator{
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = calculateBMI(height, weight);
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = calculateBMI(height, weight);
    }
    public double getBMI() {
        return bmi;
    }
    public double calculateBMI(double height, double weight) {
        return weight/(height*height);
    }
}
public interface BMICalculator {
    public double calculateBMI(double height, double weight);
}
public class ABMICalculatorWithInterface implements BMICalculator {
    public double calculateBMI(double height, double weight) {
        return weight/(height*height);
    }
}
public static void main (String[] args) {
    BMICalculator bmiCalculator = new ABMISpreadsheetAndCalculator();
    BMISpreadsheet bmiSpreadsheet = new ABMISpreadsheetAndCalculator();
    double bmi = bmiCalculator.calculateBMI(1.77, 75);
    bmi = bmiSpreadsheet.getBMI();
    // bmi = bmiCalculator.getBMI();
    // bmi = bmiSpreadsheet.calculateBMI(1.77, 75);
}

public static void main (String[] args) {
    BMICalculator[] bmiCalculators = {
        new ABMISpreadsheetAndCalculator(),
        new ABMICalculatorWithInterface()};
}

CAR ANALOGY

- A car is characterized by
  - Its make
  - License plate
  - Registration

- Licensing authority groups car by the registration or license plate
CANNOT INstantiate SPECIFICATION

- Cannot order a car from a specification
  - Must order from factory
  - A car defined by Corvette specification ordered from factory implementing the specification

- Cannot instantiate interface
  - Must instantiate class
  - BMISpreadsheet instance created by instantiating class implementing interface
INTERFACE AS A SYNTACTIC SPECIFICATION

```java
public class ABMISpreadsheet implements BMISpreadsheet{
    double height;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
    }
    double weight;
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
    }
    public double getBMI() {
        return weight/(height*height);
    }
}
```
public class ABMISpreadsheet implements BMISpreadsheet{
    double height;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
    }
    double weight;
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
    }
    public double getBMI() {
        return 13450;
    }
}
INTERFACE REQUIRED

Define interfaces for
  • All classes (that are instantiated)
  • Some are not
  • Each public method of a class should be in some interface it implements
### Impact of Differences in the Two Classes

```java
public class AnotherBMISpreadsheet implements BMISpreadsheet {
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = weight/(height*height);
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = weight/(height*height);
    }
    public double getBMI() {
        return bmi;
    }
}
```

```java
public class ABMISpreadsheet implements BMISpreadsheet {
    double height;
    public double getHeight() {
        return height;
    }
    public void setHeight(double newHeight) {
        height = newHeight;
    }
    double weight;
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
    }
    public double getBMI() {
        return weight/(height*height);
    }
}
```
ABMISpreadsheet vs. AnotherBMISpreadsheet

- AnotherBMISpreadsheet harder to program
  - Computation of bmi done in multiple setters.
  - These setters have “side effects”
- AnotherBMISpreadsheet does eager evaluation
  - Value computed may never be used.
- ABMISpreadsheet uses less space (variables)
- Getter methods of AnotherBMISpreadsheet are faster
- Setter methods of ABMISpreadsheet are faster
- Usually getter methods are called more often that setter methods
  - e.g. when ObjectEditor refresh command is executed
- Typically AnotherBMISpreadsheet will be faster, overall
TIME-SPACE TRADEOFF

Space Miser

Time Miser
TIME-SPACE TRADEOFF

Space Miser

Time Miser
**VARIABLE POINTING TO ANOTHERBMISPREADSHEET INSTANCE**

BMISpreadsheet bmiSpreadsheet = new ABMISpreadsheet(1.77, 75);
System.out.println(bmiSpreadsheet.getBMI());
bmiSpreadsheet = new AnotherBMISpreadsheet();

Because object variable stores a pointer, it can point to variable size chunks of memory
# RELATING INTERFACE AND CLASS NAMES

**Class Name:**

- `<Qualifier><Interface>`
  - ABMISpreadsheet
  - ASpaceEfficientBMISpreadsheet
  - SpaceEfficientBMISpreadsheet

- `<Interface><Qualifier> Impl`
  - BMISpreadsheetImpl
  - BMISpreadsheetSpaceEfficientImpl

**Interface Name:**

- `<ClassName>Interface`
  - ABMISpreadsheetInterface

Assumes only one implementation of interface will be created.