Comp 110/401
Least Privilege

Instructor: Prasun Dewan

Prerequisite

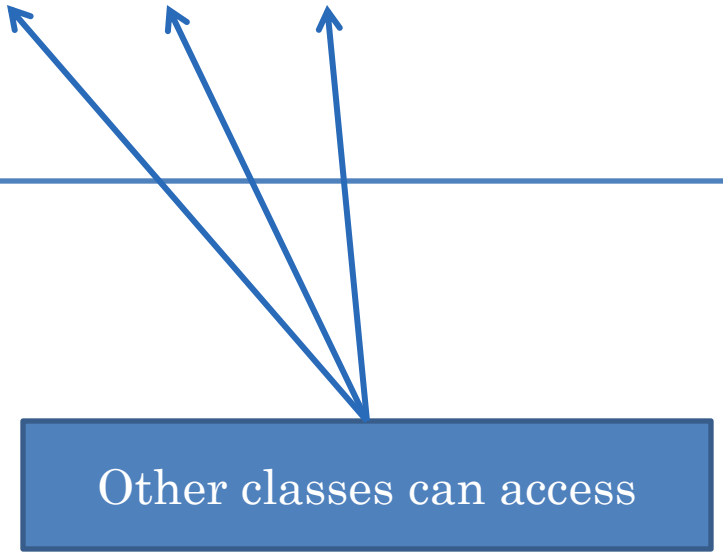
- Interfaces

Non-public Instance Variables

```
public class ABMISpreadsheet implements BMISpreadsheet {
    double height, weight, bmi;
    …
}
```

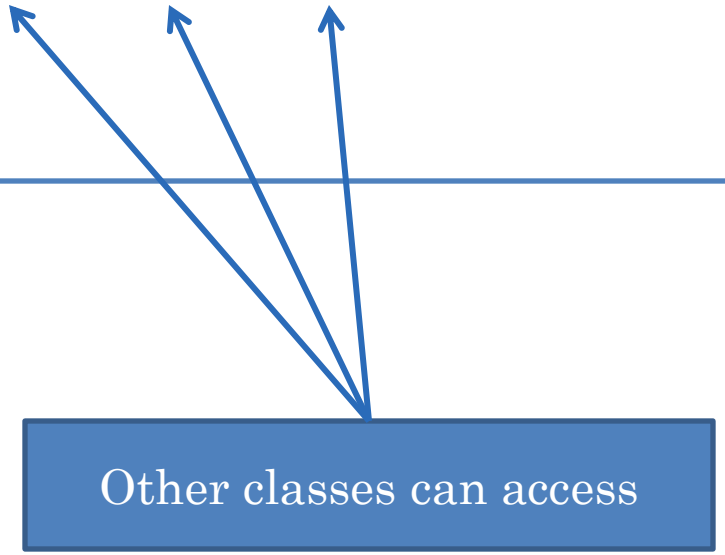
Making Instance Variables Public

```
public class ABMISpreadsheet implements BMISpreadsheet {
    public double height, weight, bmi;
    …
}
```



Hard to Change

```
public class ABMISpreadsheet implements BMISpreadsheet {
    public double height, weight;
    …
}
```



Consistency Constraints Violated

```
public class ABMISpreadsheetWithPublicVariables {
    public double height, weight, bmi;
    …
}
```

```
bmiSpreadsheet = new ABMISpreadsheetWithPublicVariables ();
bmiSpreadsheet.weight = 75;
bmiSpreadsheet.height = 1.77;
bmiSpreadsheet.bmi = 1.2;
```

Preconditions Violated

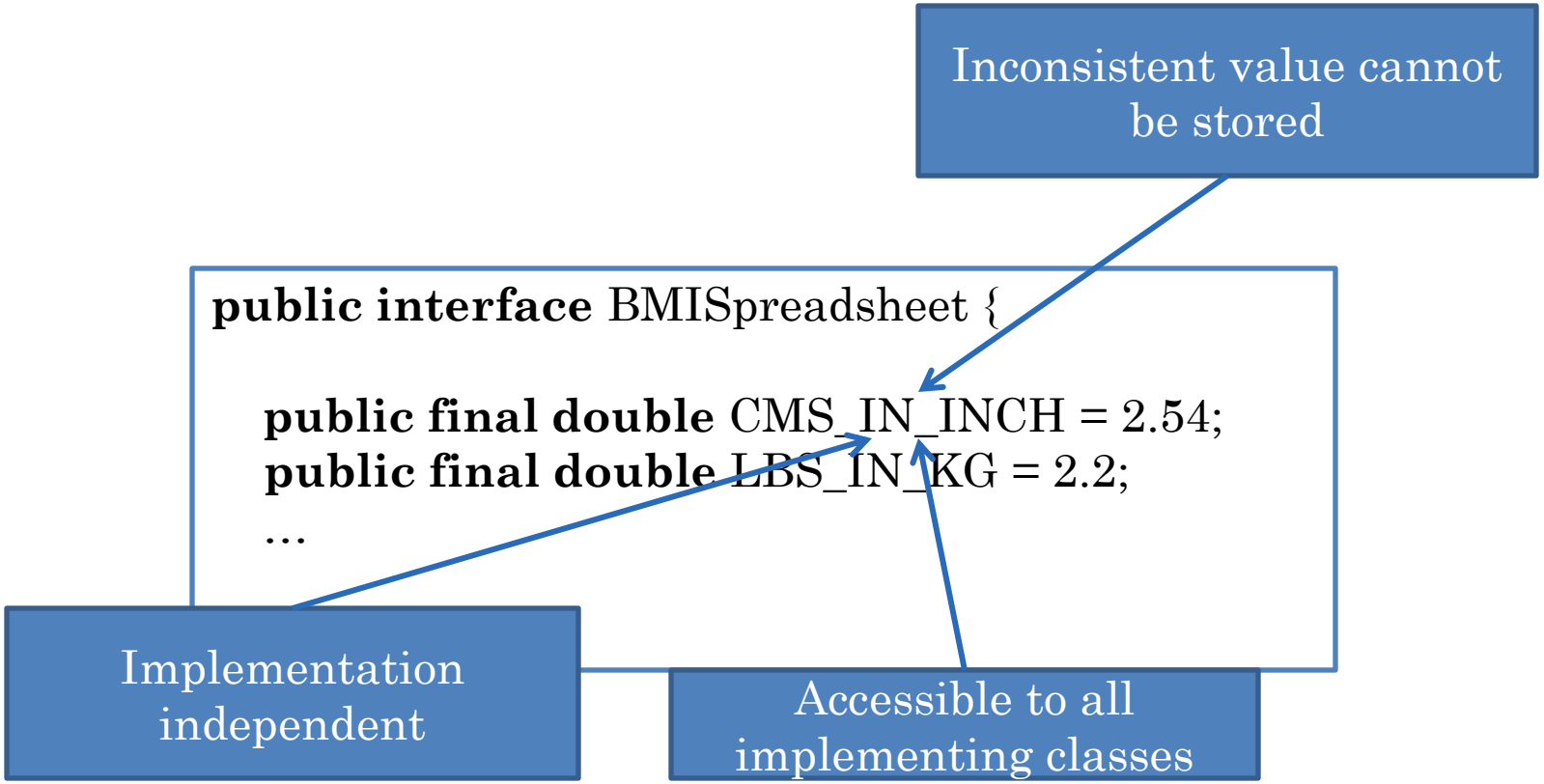
```
public class ABMISpreadsheet implements BMISpreadsheet {
    public double height, weight, bmi;
    …
}
```

More on this later

Encapsulation Principle

- Do not make instance variables public
  - Expose them through public methods

Constants typically should be public



Principle

- Declare implementation-independent named constants in interfaces
  - implementing classes can access them

Improving the Style

```
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;
    }
}
```

Code repetition

Assuming ABMICalculator does not exist

Re-using Code

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

Changing Re-used Code Once for LB, INCH Spreadsheet

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    …
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = calculateBMI();
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double nev
        weight = newWeight;
        bmi = calculateBMI();
    }

    double calculateBMI() {
        return (weight/2.2)/(height * 2.54/100*height*2.54/100);
    }
    …
}
```

Changed units to lb and inches

Have to change a single method

Should calculateBMI() be in interface?

Changing Re-used Code Once for LB, INCH Spreadsheet (Review)

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    …
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = calculateBMI();
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double nev
        weight = newWeight;
        bmi = calculateBMI();
    }

    double calculateBMI() {
        return (weight/2.2)/(height * 2.54/100*height*2.54/100);
    }
    …
}
```

Changed units to lb and inches

Have to change a single method

Should calculateBMI() be in interface?

Only Public Methods in Interface

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    public double getHeight() {
        return height;
    }
    public void setHeight(double nev
        height = newHeight;
        bmi = calculateBMI();
    }
    …
    double calculateBMI() () {
        return (weight/2.2)/(height * 2.54/1
    }
    …
}
```

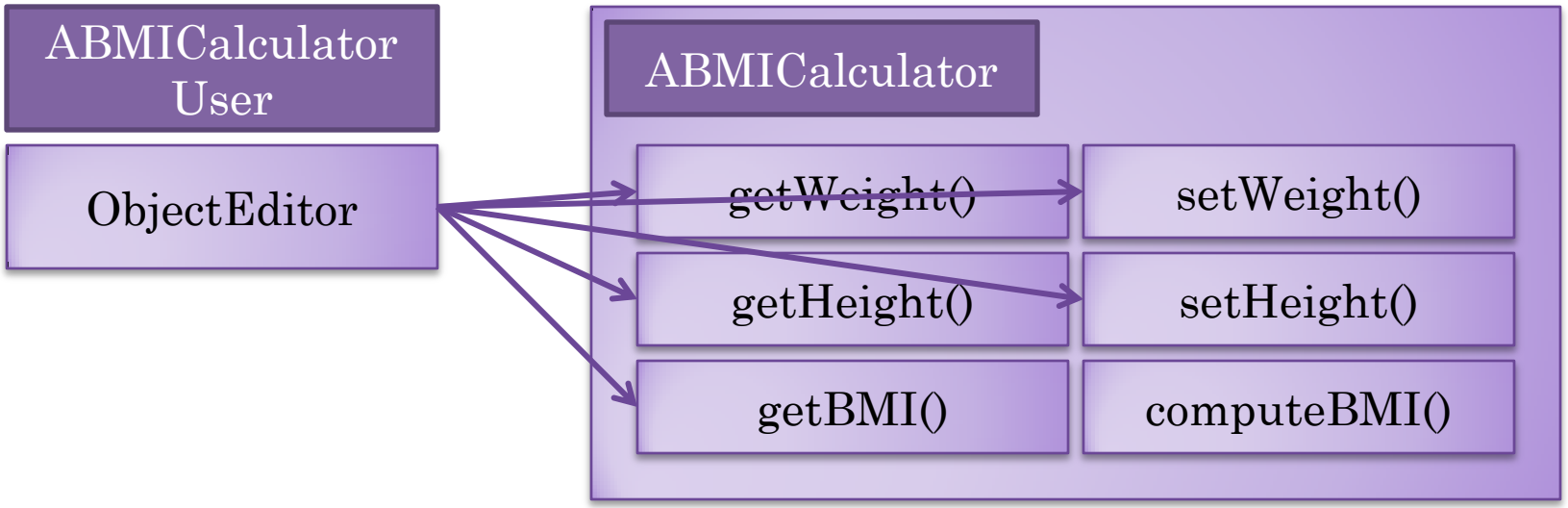
BMISpreadsheet.java

```
public interface BMISpreadsheet {

    public double getWeight();
    public void setWeight(double newVal)

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

    public double getBMI();
```

Not in interface

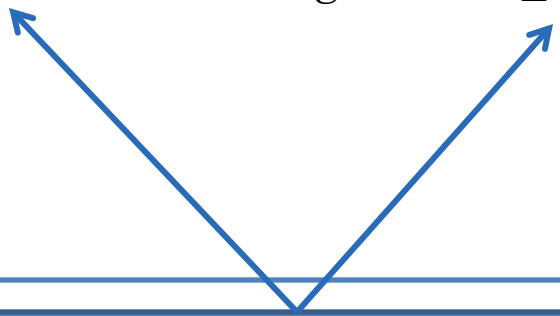
Principle of Least Privilege

- Do not give a user of some code more rights than it needs
  - Code is easier to change
  - Need to learn less to use code
  - Less likelihood of accidental or malicious damage to program
- Like hiding engine details from car driver



More Code Repetition

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    …
    final double LBS_IN_KG = 2.2;
    final double CMS_IN_INCH = 2.54;
    …
    double calculateBMI() {
        return (weight/LBS_IN_KG) /
            (height*CMS_IN_INCH/100*height*CMS_IN_INCH/100);
    }
    …
}
```



Within same method and has the same value

Removing Code Repetition

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    …
    final double LBS_IN_KG = 2.2;
    final double CMS_IN_INCH = 2.54;
    …
    double calculateBMI() {
        double heightInMeters = height*CMS_IN_INCH/100;
        return (weight/LBS_IN_KG) /
            (heightInMeters*heightInMeters);
    }
    …
}
```

Local vs. Global Variable

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    double heightInMeters;
    …
    final double LBS_IN_KG = 2.2;
    final double CMS_IN_INCH = 2.54;
    …
    double calculateBMI() {
        heightInMeters = height*CMS_IN_INCH/100;
        return (weight/LBS_IN_KG) /
            (heightInMeters*heightInMeters);
    }
    …
}
```

Local vs. Global Variable

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    double heightInMeters = height*CMS_IN_INCH/100;
    …
    final double LBS_IN_KG = 2.2;
    final double CMS_IN_INCH = 2.54;
    …
    public void setHeight(double newHeight) {
        heightInMeters = newHeight;
        bmi = calculateBMI();
    }
    …
    double calculateBMI() {
        return (weight/LBS_IN_KG) /
            (heightInMeters*heightInMeters);
    }
    …
}
```

Violating least privilege

SCOPE

height scope

```
public class AnotherBMISpreadsheet implements BMISpreadsheet{
    double height, weight, bmi;
    …
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = calculateBMI();
    }
```

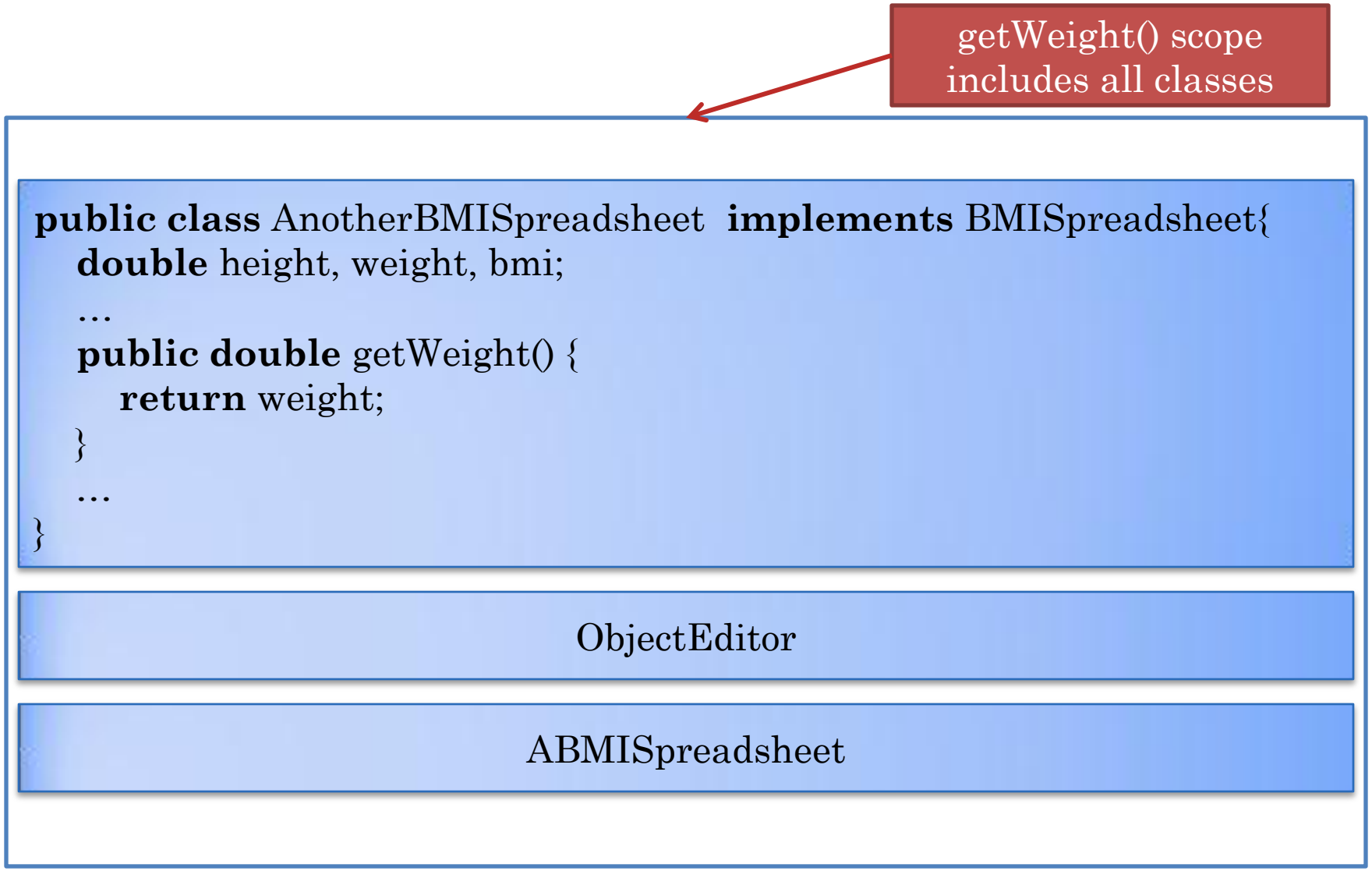
Not a scope

```
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = weight/(height*height);
    }
    …
```

heightInMeters scope

```
    double calculateBMI () {
        double heightInMetres = height*CMS_IN_INCH/100;
        return (weight/LBS_IN_KG) / (heightInMetres*heightInMetres);
    }
    …
}
```

Scope of Public Items



Scope Modifiers

- **public**: accessible in all classes.
- **protected**: accessible in all subclasses of its class and all classes in its package.
  - Will see this later.
  - Many of the variables/methods in lecture code have protected access even though PPT slides do not show it
- **default**: accessible in all classes in its package.
- **private**: accessible only in its class.

Will use default access for non public variables as we do not know the full context for the code right now

Some purists of least privilege insist on private access

Identifier Scope

- Region of code where the identifier is visible
- Arbitrary scopes not possible
- Least Privilege => Make scope as small as possible

Following Least Privilege

```
public class AnotherBMISpreadsheet  implements BMISpreadsheet{
    double height, weight, bmi;
    …
    public void setHeight(double newHeight) {
        height = newHeight;
        bmi = calculateBMI();
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double newWeight) {
        weight = newWeight;
        bmi = weight/(height*height);
    }
    …
    double calculateBMI() () {
        double heightInMetres = height*CMS_IN_INCH/100;
        return (weight/LBS_IN_KG) / (heightInMetres*heightInMetres);
    }
    …
}
```

heightInMeters scope

Naming of Variables in Different Scopes

```
public class ABMISpreadsheet  {
  double height;
  double weight;
  public ABMISpreadsheet(
    double theInitialHeight, double theInitialWeight) {
    setHeight(theInitialHeight);
    setWeight(theInitialWeight);
  }
  public void setWeight(double newWeight) {
    weight = newWeight;
  }
  public void setHeight(double newHeight) {
    height = newHeight;
  }
  …

}
```

Same Variable Name in Nested Scopes

```
public class ABMISpreadsheet  {
  double height;
  double weight;
  public ABMISpreadsheet(
    double height, double weight) {
    setHeight(height);
    setWeight(weight);
  }
  public void setWeight(double weight) {
    weight = weight;
  }
  public void setHeight(double height) {
    height = height;
  }
  …

}
```

Local, not global instance variable

Eclipse uses fonts and colors to indicate scope

Disambiguation with this (Standard Convention)

```
public class ABMISpreadsheet  {
  double height;
  double weight;
  public ABMISpreadsheet(
    double height, double weight) {
    setHeight(height);
    setWeight(weight);
  }
  public void setWeight(double weight) {
    this.weight = weight;
  }
  public void setHeight(double height) {
    this.height = height;
  }
  …

}
```

Local, not global instance variable

Eclipse features based on this convention

Can forget to put this

Using Different Names

```
public class ABMISpreadsheet  {
  double height;
  double weight;
  public ABMISpreadsheet(
    double theHeight, double theInitialWeight) {
    setHeight(theHeight);
    setWeight(theInitialWeight);
  }
  public void setWeight(double aWeight) {
    weight = weight;
  }
  public void setHeight(double newVal) {
    height = newVal;
  }
  …

}
```

Must sometimes fight with Eclipse

Examples use multiple conventions for local variables