## Scope: Local Variables

- We have discussed the scope of local variables (ones defined in methods) already
  - A variable’s scope extends from the line it is declared until the end of the block it is contained in
  - A formal parameter’s scope is the entire method
  - A variable declared in the for loop control has a scope that extends to the end of the for loop
- This means that variables can’t be used before they are declared, or after the end of their scope

## Scope: Instance and Class Variables

- The scope rules for instance and class variables are different
- The scope of instance and class variables is the entire class
  - regardless of where they are declared
- They can be referred to from any method in the class
- This means that you can refer to an instance or class variable before you declare it

## Example: Clock class

class Clock {
    private int hour;
    private int minute;
    private int second;
    public Clock( int h, int m, int s ) {
        hour = h % HOURSPERDAY;
        minute = m % MINUTESPERHOUR;
        second = s % SECONDSPERMINUTE;
    }
    public Clock() { ... }
    public void add( Clock c ) { ... }
    public void tick() { ... }
    public String toString() { ... }
    public static final int HOURSPERDAY = 24;
    public static final int MINUTESPERHOUR = 60;
    public static final int SECONDSPERMINUTE = 60;
}

## Hidden Variables

- Local variables hide or eclipse instance and class variables with the same name
- References to that name will refer to the local variable
Hidden Class Variables

• Access hidden class variables using the class name

```java
//Scope2.java: class versus local scope
class Scope2 {
    public static void main(String[] args) {
        int x = 2;
        System.out.println("local x = " + x);
        System.out.println("class variable x = " + Scope2.x);
    }
    static int x = 1;
}
```

Hidden Instance Variables

• Last time one of the sample question asked us to write a class Point

```java
class Point {
    private int x;
    private int y;
    public Point( int x1, int y1 ) {
        x = x1;
        y = y1;
    }
    public String toString() {
        return ( x + "", " + y ");
    }
}
```

Hidden Instance Variables

• Notice that we named the Constructor's parameters x1 and y1 instead of the more intuitive x and y

```java
class Point {
    private int x;
    private int y;
    public Point( int x, int y ) {
        x = x;     // which x is which?
        y = y;     // ?
    }
    ...
}
```

The Keyword `this`

• Java provides the keyword `this` to allow us to refer to an object's hidden instance variables — this is a reference to the implicit object

```java
class Point {
    private int x;
    private int y;
    public Point( int x, int y ) {
        this.x = x;
        this.y = y;
    }
    ...
}
```
Arrays of Objects

• Just as we can have arrays of primitive types, we can also have arrays of objects

• Recall that
  – When we declare an array we have to use new to create the storage for the array
  – When we create an object we have to use new to create the storage for the object

• So, when we create an array of objects we have to use new twice; once for the array and once for the objects.

    Point[] pArray = new Point[5];

The array

for (int i = 0; i < pArray.length; i++) {
    pArray[i] = new Point(i, i+1);
}

A Deck of Cards

• The book has an example where a deck of cards is represented as a 52 element array.
• Each element is an object of type Card

    //Deck.java - a deck of playing cards
    class Deck {
        Card[] deck;
        Deck() {
            deck = new Card[52];
            for (int i = 0; i < deck.length; i++) {
                deck[i] = new Card(new Suit(i / 13 + 1),
                                    new Pips(i % 13 + 1));
            }
        }
    }
Shuffle the Deck

void shuffle() {
for (int i = 0; i < deck.length; i++)
    int k = (int)(Math.random() * 52);
    Card t = deck[i];
    deck[i] = deck[k];
    deck[k] = t;
}

• The book has an error
• The highlighted line is a correction

Deck.toString()

public String toString() {
    String t = "";
    for (int i = 0; i < 52; i++)
        if ( (i + 1) % 5 == 0)
            t = t + deck[i] + "\n";
        else
            t = t + deck[i];
    return t;
}

• The book has an error
• The highlighted line is a correction

Testing the Deck

//CardTest.java testing the shuffling method.
public class CardTest {
    public static void main(String argv[])
        Deck deck = new Deck();
        System.out.println("\nNew Shuffle\n" + deck);
        deck.shuffle();
        System.out.println("\nNew Shuffle\n" + deck);
        deck.shuffle();
        System.out.println("\nNew Shuffle\n" + deck);
}

Using classes with HW9

• You might choose to represent the yard as a two dimensional array of Cells
  --Cell contains information about whether that yard location is exposed, contains scat, etc.

Cell[][] yard = new Cell[SIZE][SIZE];
for (int i=0; i < yard.length; i++ ) {
    for (int j=0; j < yard[i].length; j++ ) {
        yard[i][j] = new Cell();
    }
}