

Java Programming

Arthur Hoskey, Ph.D.
Farmingdale State College
Computer Systems Department

- Go over important dates.

Important Dates

- Classes, objects, methods and instance variables.
- Classes:
 - Declare a class
 - Create an object.
- Declare methods of a class to implement the class's behaviors.
- Declare instance variables of a class to implement the class's attributes.
- Object Methods:
How to call an object's method to make that method perform its task.
- Difference:
Instance variables Vs local variables of a method.
- Constructor: How to use a constructor to ensure that an object's data is initialized when the object is created.
- Difference:
Primitive Vs reference types.

Objectives

- What is an object?
- Objects are just things in the real world.
- For example, people, animals, plants, cars, planes, buildings, computers and so on.
- Telephones, houses, traffic lights, microwave ovens, and water coolers are a few more examples of objects.

Object-Oriented Programming

- Objects all have attributes and exhibit behaviors.
- Attributes: size, shape, color, weight
- Behaviors:
 - A ball bounces, inflates and deflates
 - A baby cries, sleeps, crawls
 - A car accelerates, brakes, turns

Object-Oriented Programming

- Object-oriented design models software in terms similar to those that people use to describe real-world objects.
- Think of the problem domain and look for the things that are objects.

Object-Oriented Programming

- When constructing a building what is the first thing that an architect will do?
- An architect creates a blueprint.
- What is a blueprint?
- A blueprint is a detailed description of what the building will look like after it is built.
- Similarly, a class definition is a blueprint of what an object will look like in memory when it is created.

Classes

- How would we go about modeling a car object?
- First create the blueprint.
- The blueprint in object-oriented programming is the class definition.

Classes

Here are some of the attributes and behaviors of a car.

Color Key

Class Name

Attributes

Behaviors

Car

Year

Color

Speed

Accelerate

Decelerate

What are the data types of the attributes?

Classes

Color Key

Class Name

Attributes

Behaviors

Data types for attributes



Car

Year – int

Color – String

Speed – int

Accelerate

Decelerate

Classes

Color Key

Class Name

Attributes

Behaviors

Car

Year – int

Color – String

Speed – int

Accelerate

Decelerate

What effect will the behaviors accelerate and decelerate have on the object?

Classes

Behaviors change the state of the object. Accelerate and Decelerate will change the speed.

Color Key

Class Name

Attributes

Behaviors

Car

Year:int

Color:String

Speed:int

Accelerate – **Increases speed**

Decelerate – **Decreases speed**

Classes

Color Key

Class Name

Attributes

Behaviors

Car

Year:int

Color:String

Speed:int

Accelerate – Increases speed

Decelerate – Decreases speed

How do we model attributes and behaviors in a program?

Classes

Model in a Program
Attributes → Variables
Behaviors → Methods

Color Key

Class Name

Variables

Methods

Car

int Year;

String Color;

int Speed;

void Accelerate() { // method code... }

Void Decelerate() { // method code... }

Classes

- Write a class definition to model our car in a program.
- Create a class called "Car" that we can use to model a car.
- Attributes are represented by variables.
- Behaviors are represented by methods.

Classes

```
public class Car
{
    // Attributes
    // Fill in the attributes here

    // Behaviors
    // Fill in the behaviors here
}
```

- Everything that is part of the class will be between the braces.


Sample Class Definition


```
public class Car  
{
```

```
    // Attributes
```

```
    private int year;  
    private int speed;  
    private String color;
```

Attributes should be
declared as member
variables of the class



```
    // Behaviors
```

```
    Fill in the behaviors here
```

```
}
```

Sample Class Definition

```
public class Car
```

```
{
```


```
    // Attributes
```

```
    private int year;
```

```
    private int speed;
```

```
    private String color;
```

Behaviors should be
defined as member
methods of the class



```
    // Behaviors
```

```
    public void Accelerate() { Code to accelerate }
```

```
    public void Decelerate() { Code to decelerate }
```


```
}
```

Sample Class Definition

```
public class Car
{
    // Attributes
    private int year;
    private int speed;
    private String color;

    // Behaviors
    public void Accelerate() {
        speed = speed + 10;
    }
    public void Decelerate() {
        speed = speed - 10;
    }
}
```

Accelerate will increase the speed by adding to the speed member variable. This changes the state of the object.

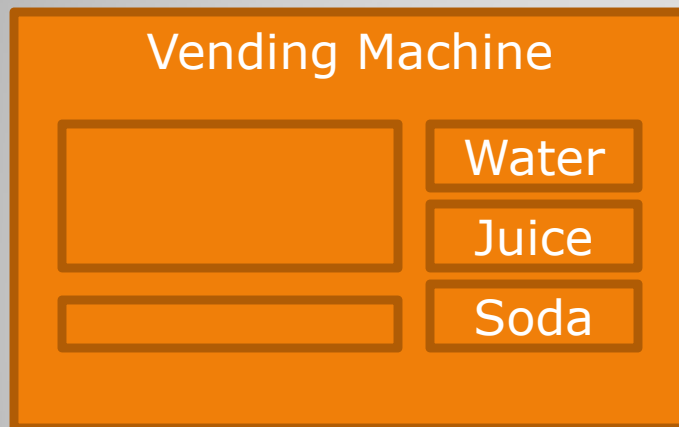


Sample Class Definition

- What does the ***private*** keyword that precedes each class variable name mean?

Access Modifiers

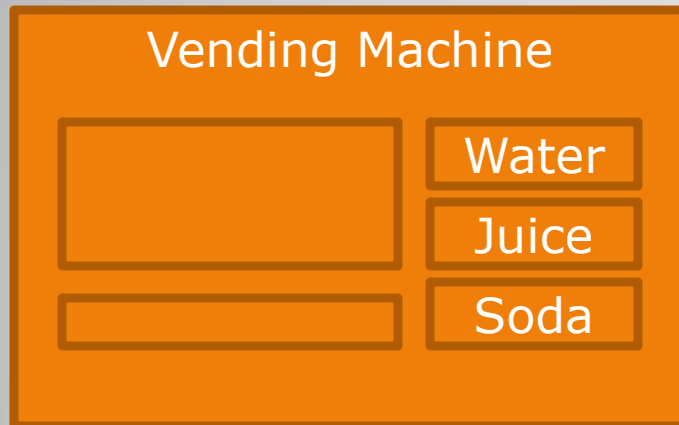
- We cannot see the inner workings of an object from the outside
- For example:
 - We do not see the engine of a car from the outside
 - We do not see the inner mechanisms of a vending machine from the outside



The inside mechanisms of the vending machine are not visible from the outside

Access Modifiers

- What we can see from the outside is ***public***
- The things on the inside that we cannot see are ***private***
- Are the buttons “Water”, “Juice”, and “Soda” public or private?
- What do the buttons represent?

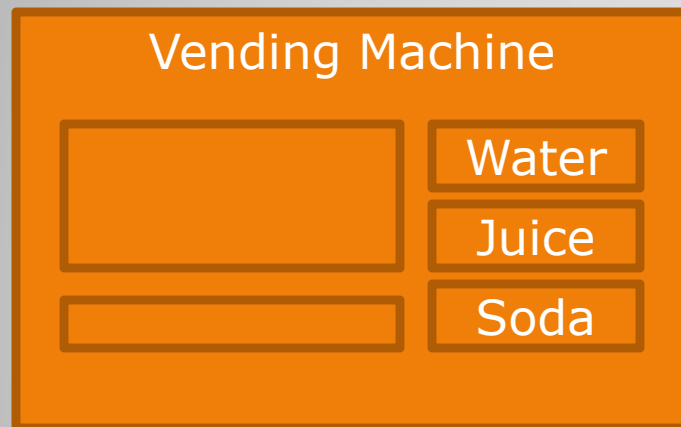


The inside mechanisms of the vending machine are not visible from the outside

Access Modifiers

- What we can see from the outside is **public**
- The things on the inside that we cannot see are **private**
- The “Water”, “Juice”, and “Soda” buttons represent public behaviors

Behaviors



Buy bottle of water

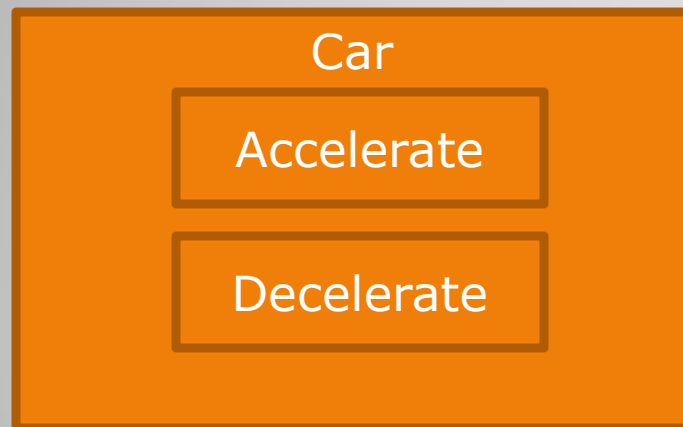
Buy bottle of juice

Buy bottle of soda

Access Modifiers

- The “Accelerate” button would be the gas pedal of a car
- The “Decelerate” button would be the brake of a car
- Note: We cannot see the private member variables “speed”, “year”, and “color”

Behaviors



Accelerate – Make the car go faster

Decelerate – Make the car go slower

Access Modifiers

- Private member
 - Only visible from inside the object
 - Cannot be seen from "outside"
- Public members
 - Visible from the outside
- Now look at the class definition again:
 - Look for the public members
 - Look for the private members

Access Modifiers

```
public class Car  
{
```

```
    // Attributes
```

```
    private int year;
```

```
    private int speed;
```

```
    private String color;
```

Member variables are generally
declared private



```
    // Behaviors
```

```
    public void Accelerate() {  
        speed = speed + 10;
```


```
    }
```

```
    public void Decelerate() {  
        speed = speed - 10;
```

```
    }
```

```
}
```

Member methods are generally
declared public



Public and Private Members

- **private** keyword
 - **Used for most instance variables.**
 - private variables and methods are accessible only to methods of the class in which they are declared.
 - Declaring instance variables private is known as "**data hiding**".
- **public** keyword
 - **Used for most methods.**
 - Public methods are accessible outside the class.

Access Modifiers

- If the private members cannot be seen from the outside then how do we change them?

Access Modifiers

- **Use get/set methods to change private member variables.**
- private instance variables
 - Cannot be accessed directly by clients of the object.
 - Use **set** methods to **change the value.**
 - Use **get** methods to **retrieve the value.**

Get and Set Methods

```
public class Car  
{
```

```
    // Attributes  
    private int year;  
    private int speed;  
    private String color;
```

```
    // Behaviors
```

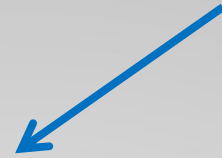
```
    public int GetYear() { return year; }  
    public int GetSpeed() { return speed; }  
    public String GetColor() { return color; }
```

```
    public void SetYear(int newYear) { year = newYear; }  
    public void SetSpeed(int newSpeed) { speed = newSpeed; }  
    public void SetColor(String newColor) { color = newColor; }
```

```
    // Accelerate and Decelerate not shown
```

```
}
```

Get/Set methods of
the Car class



Get and Set Methods

Real World Example of Public vs Private

- Think of how a supermarket is setup.
- The supermarket can be divided into two main areas:
 - Main Floor
 - Stock Room
- Which area do customers or "clients" have access to?

Public Vs Private

- Which area do customers or "clients" have access to?

***ANSWER:
MAIN FLOOR***

Public Vs Private

- If a customer wants something that is in the stock room how can they get it?

Public Vs Private

- If a customer wants something that is in the stock room how can they get it?

ANSWER:

Ask a worker to get it for you. They have access to the stock room.

Public Vs Private

- Now think in terms of a Java class.
- What are the **public** areas of a Java class more similar to:
Main floor or Stock room?
- What are the **private** areas of a Java class more similar to:
Main floor or Stock room?

Public Vs Private

- Now think in terms of a Java class.
- What are the **public** areas of a Java class more similar to:

ANSWER:

Main floor

- What are the **private** areas of a Java class more similar to:

ANSWER:

Stock room

Public Vs Private

- If a "client" of a Java class wants to get something from the private area of a class how can she get it?

Public Vs Private

- If a "client" of a Java class wants to **get** something from the private area of a class how can she get it?

ANSWER:

Use a get method.

Any method defined on a class has access to all member variables and member methods of that class (this includes the private variables and private methods).

Public Vs Private

- **Local Variables** - Declared in the body of method. Can only be used within that method.
- **Instance Variables** - Declared in a class declaration but not in a method. Each object of the class has a separate instance of the variable.

```
public class MyClass {  
    int x;  
  
    public void myMethod {  
        int y;  
        y = 10;  
        x = 20;  
    }  
  
    public void otherMethod() {  
        x = 20;  
  
        y = 30;  
    }  
}
```

x is an instance variable (accessible from all methods of the class)

y is a local variable for myMethod (only accessible from inside myMethod)

x can be used in both of these places because member methods have access to all member variables

Y is local to myMethod so it CANNOT be used here (y is out of scope)

Scope of variables

- Take Attendance!

Attendance