Objects: Packaging Computation

Chapter 4

For Next Time

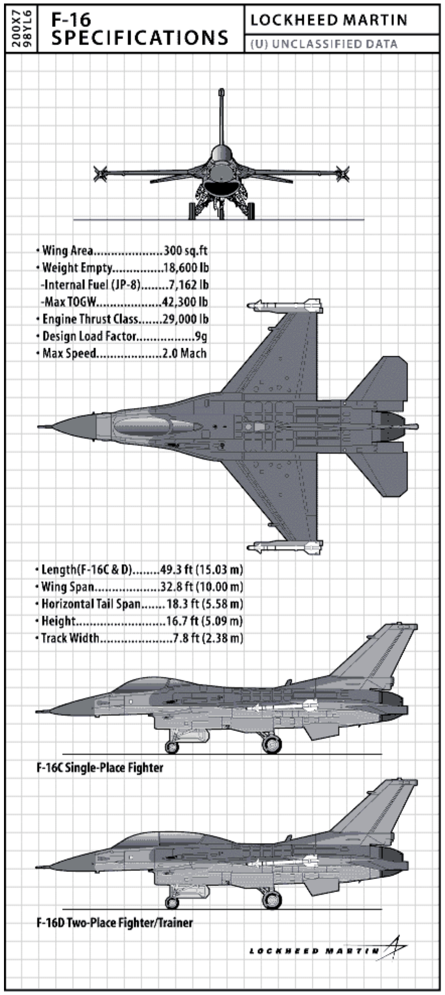
- Read Chapter 4

Objects

- Java is an object-oriented programming language
  - Most modern languages are OO
- Objects form the basis for computation
- Objects correspond more directly to real world objects; primitive types do not
- An object has a programmer-defined type
  - The primitive types are fixed

Class

- A *class* is like a *blueprint* that defines the structure and capabilities of an object
  - Other descriptive analogies include: *pattern* and *template*
  - We say an object is an *instance* of a class



F-16 SPECIFICATIONS

LOCKHEED MARTIN

(U) UNCLASSIFIED DATA

- Wing Area....................300 sq.ft
- Weight Empty...............18,600 lb
  - -Internal Fuel (JP-8)........7,162 lb
  - -Max TOGW..................42,300 lb
- Engine Thrust Class.......29,000 lb
- Design Load Factor.................9g
- Max Speed..................2.0 Mach

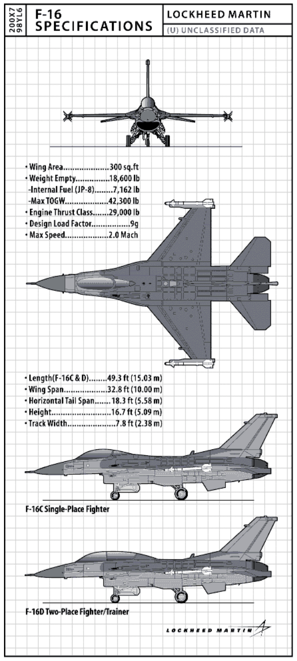
- Length(F-16C & D)........49.3 ft (15.03 m)
- Wing Span..................32.8 ft (10.00 m)
- Horizontal Tail Span....... 18.3 ft (5.58 m)
- Height.........................16.7 ft (5.09 m)
- Track Width....................7.8 ft (2.38 m)

F-16C Single-Place Fighter

F-16D Two-Place Fighter/Trainer

Class

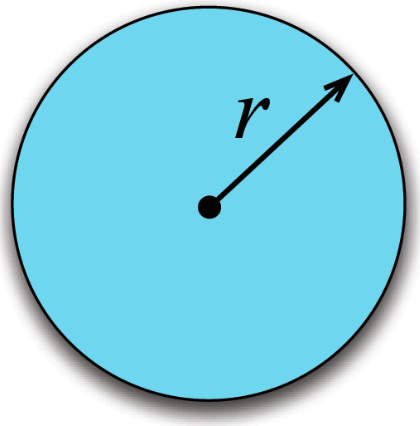
- The class definition must come first before any objects of that class can be created
- In this chapter we limit our objects to be a way to perform computations
  - They are like fancy, custom calculators
- Later we will see how objects can do much more


200X7
98YL6
F-16
SPECIFICATIONS
LOCKHEED MARTIN
(U) UNCLASSIFIED DATA
• Wing Area....................300 sq.ft
• Weight Empty..............18,600 lb
-Internal Fuel (JP-8)........7,162 lb
-Max TOGW..................42,300 lb
• Engine Thrust Class.......29,000 lb
• Design Load Factor................9g
• Max Speed...................2.0 Mach
• Length(F-16C & D)........49.3 ft (15.03 m)
• Wing Span...................32.8 ft (10.00 m)
• Horizontal Tail Span....... 18.3 ft (5.58 m)
• Height..........................16.7 ft (5.09 m)
• Track Width.....................7.8 ft (2.38 m)
F-16C Single-Place Fighter
F-16D Two-Place Fighter/Trainer
LOCKHEED MARTIN

Examples

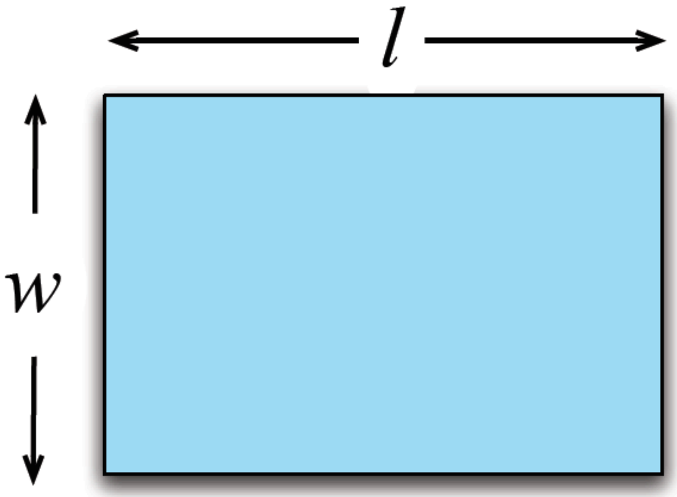
- Let’s define some objects in DrJava and see what they can do
  - `CircleCalculator`
  - `RectangleCalculator`

Shape Formulas



$$C = 2\pi r$$

$$A = \pi r^2$$



$$P = 2l + 2w$$

$$A = lw$$

Attributes

```
private final double PI = 3.14159;
```

- An object can store data
- A piece of an object’s data is called an *attribute*
- Terms used more commonly in place of attribute:
  - instance variable
  - field

Operations

```
public double circumference(double radius) {
    return 2 * PI * radius;
}
```

- Objects can exhibit *behavior*
- Object behavior is defined in *operations*
- In Java, the term *method* is used more often in place of operation
- A method is a named collection of statements that perform some useful task

Clients

- The developer of a class designs the class so others can use it to make objects
- The parties that use the class by creating and using objects of that class are called *clients* of that class
  - Clients can be people (like us in DrJava’s Interactions Pane)
  - Clients can be other other software (by far the most typical clients)

`public` vs. `private`

- Parts of the class (fields and methods) that the client needs to know about to make full use of the class should be made `public`
- Parts of the class that client does not need to know about to make full use of the class and its objects should be made `private`
- In general, attributes are usually `private` and most methods are `public`
  - We’ll refine these guidelines as time goes on

Comments

- Single-line comments

```
//  This is a brief note
```

- Block comments

```
/*  This is a longer remark that covers
    several lines.  */
```

- Documentation comments

```
/**  These are used for embedding
     documentation in code.  */
```

DrJava hint: Highlight a block of code and press Ctrl /
What about Shift Ctrl / ?

Method Definition vs. Method Invocation

- A method has exactly one definition
  - The class author writes the method definition
  - The definition appears in the class
- A method may have many invocations
  - Clients invoke (call) the methods
  - The call can be from anywhere, from outside the class or from within the class

Local Variables

- A variable declared within a method is local to that method
- It is distinct from all other variables declared elsewhere
  - Even if it has the same name
- Local variables for a method are like a temporary “scratch sheet” that the method uses to do its task
- Local variables are re-created each time the method is called

Method Parameters

- Information is passed to methods via parameter(s)
- The types of parameters must be declared in the method definition
- Parameters specified in the method definition are called *formal* parameters
  - The client passes *actual* parameters
  - Actual parameters are copied to the formal parameters when the method is called
- Parameters are like local variables

Method Results

- Methods can compute a result and return that result to the client
- The `return` statement indicates what will be returned
- A method can return a value of any valid Java type
- A method with a return type of `void` does not return a result to the client
  - In a `void` method a return statement may not attempt to return a value

Next . . .

Boolean Expressions and
Conditional Execution