3.3 Designing Data Types

Object Oriented Programming

Procedural programming. [verb-oriented]
- Tell the computer to do this.
- Tell the computer to do that.

Alan Kay’s philosophy. Software is a simulation of the real world.
- We know (approximately) how the real world works.
- Design software to model the real world.

Objected oriented programming (OOP). [noun-oriented]
- Programming paradigm based on data types.
- Identify things that are part of the problem domain or solution.
- Things in the world know things: instance variables.
- Things in the world do things: methods.

Alan Kay

Alan Kay. [Xerox PARC 1970s]
- Invented Smalltalk programming language.
- Conceived Dynabook portable computer.
- Ideas led to: laptop, modern GUI, OOP.

“ The computer revolution hasn’t started yet. ”
“ The best way to predict the future is to invent it. ”
“ If you don’t fail at least 90 per cent of the time, you’re not aiming high enough. ”

— Alan Kay

Encapsulation

Bond. What’s your escape route?
Saunders. Sorry old man. Section 26 paragraph 5, that information is on a need-to-know basis only. I’m sure you’ll understand.
Encapsulation

**Data type.** Set of values and operations on those values.
Ex. `int`, `String`, `Complex`, `Vector`, `Document`, `GuitarString`, `Tour`, ...

**Encapsulated data type.** Hide internal representation of data type.

Separate implementation from design specification.
- **Class** provides data representation and code for operations.
- **Client** uses data type as black box.
- **API** specifies contract between client and class.

Bottom line. You don’t need to know how a data type is implemented in order to use it.

Intuition

**Counter Data Type**

**Counter.** Data type to count electronic votes.

```java
public class Counter {
    public int count;
    public final String name;
    public Counter(String id) { name = id; }
    public void increment() { count++; }
    public int value() { return count; }
}
```

Legal Java client.

```java
Counter c = new Counter("Volusia County");
c.count = -16022;
```

Oops. Al Gore receives -16,022 votes in Volusia County, Florida.
Counter. **Encapsulated** data type to count electronic votes.

```java
public class Counter {
    private int count;
    private final String name;
    public Counter(String id) { name = id; }
    public void increment() { count++; }
    public int value() { return count; }
}
```

Does not compile.

Counter c = new Counter("Volusia County");
c.count = -16022;

**Benefit.** Can guarantee that each data type value remains in a consistent state.

---

**Encapsulation.**
- Keep data representation hidden with **private** access modifier.
- Expose API to clients using **public** access modifier.

```java
public class Complex {
    private final double re, im;
    public Complex(double re, double im) { … }
    public double abs() { … }
    public Complex plus(Complex b) { … }
    public Complex times(Complex b) { … }
    public String toString() { … }
}
```

e.g., to polar coordinates

**Advantage.** Can switch internal representation without changing client.

**Note.** All our data types are already encapsulated!

---

**Time Bombs**

**Internal representation changes.**
- [VIN numbers] We’ll run out by 2010.

---

**Lesson.** By exposing data representation to client, need to sift through millions of lines of code in client to update.

---

**Ask, Don’t Touch**

**Encapsulated data types.**
- Don’t touch data and do whatever you want.
- Instead, ask object to manipulate its data.

“Ask, don’t touch.”

---

**Lesson.** Limiting scope makes programs easier to maintain and understand.

"principle of least privilege"
Immutability

Immutable data type. Object’s value cannot change once constructed.

Immutability: Advantages and Disadvantages

**Immutable data type.** Object’s value cannot change once constructed.

**Advantages.**
- Avoid aliasing bugs.
- Makes program easier to debug.
- Limits scope of code that can change values.
- Pass objects around without worrying about modification.

**Disadvantage.** New object must be created for every value.

Final Access Modifier

**Final.** Declaring an instance variable to be `final` means that you can assign it a value only once, in initializer or constructor.

```java
public class Counter {
    private final String name;
    private int count;
    ...
}
```

**Advantages.**
- Helps enforce immutability.
- Prevents accidental changes.
- Makes program easier to debug.
- Documents that the value cannot not change.
Spatial Vectors

Vector Data Type

Set of values. Sequence of real numbers. [Cartesian coordinates]

API.

public class Vector
{
   private int N;
   private double[] coords;

   public Vector(double[] a)
   {
      N = a.length;
      coords = new double[N];
      for (int i = 0; i < N; i++)
         coords[i] = a[i];
   }

   public double dot(Vector b)
   {
      double sum = 0.0;
      for (int i = 0; i < N; i++)
         sum += (coords[i] * b.coords[i]);
      return sum;
   }

   public Vector plus(Vector b)
   {
      double[] c = new double[N];
      for (int i = 0; i < N; i++)
         c[i] = coords[i] + b.coords[i];
      return new Vector(c);
   }
}

Vector Data Type Applications

Relevance. A quintessential mathematical abstraction.

Applications.

- Statistics.
- Linear algebra.
- Clustering and similarity search.
- Force, velocity, acceleration, momentum, torque.
- ...

Vector Data Type: Implementation
This. The keyword `this` is a reference to the invoking object.

Ex. When you invoke `a.magnitude()`, this is an alias for `a`. 

```java
public Vector times(double t) {
    double[] c = new double[N];
    for (int i = 0; i < N; i++)
        c[i] = t * coords[i];
    return new Vector(c);
}

public double magnitude() {
    return Math.sqrt(this.dot(this));
}

public Vector direction() {
    return this.times(1.0 / this.magnitude());
}
...