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.

Counter Data Type

Counter. Data type to count electronic votes.

```
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; }
}
```

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

Intuition

Client

API
- volume
- change channel
- adjust picture
- decode NTSC signal

Implementation
- gas plasma monitor
- Samsung FPT-6374
- wall mountable
- 4 inches deep

client needs to know how to use API
implementation needs to know what API to implement

Oops. Al Gore receives -16,022 votes in Volusia County, Florida.

Implementation and client need to agree on API ahead of time.
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.

```java
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.

---

**Time Bombs**

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

---

**Ask, Don’t Touch**

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

---

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

---

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

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.

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

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

API.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public class Vector(double[] a)</td>
<td>create a vector with the given Cartesian coordinates</td>
</tr>
<tr>
<td>Vector plus(Vector b)</td>
<td>sum of this vector and b</td>
</tr>
<tr>
<td>Vector minus(Vector b)</td>
<td>difference of this vector and b</td>
</tr>
<tr>
<td>Vector times(double t)</td>
<td>scalar product of this vector and t</td>
</tr>
<tr>
<td>double dot(Vector b)</td>
<td>dot product of this vector and b</td>
</tr>
<tr>
<td>double magnitude()</td>
<td>magnitude of this vector</td>
</tr>
<tr>
<td>Vector direction()</td>
<td>unit vector with same direction as this vector</td>
</tr>
</tbody>
</table>

Example:

\[
x = (0, 3, 4, 0), \quad y = (0, -3, 1, -4)
\]
\[
x + y = (0, 0, 5, -4)
\]
\[
3x = (0, 9, 12, 0)
\]
\[
x \cdot y = (0 \cdot 0) + (3 \cdot -3) + (4 \cdot 1) + (0 \cdot -4) = -5
\]
\[
|x| = \sqrt{0^2 + 3^2 + 4^2 + 0^2} = 5
\]
\[
x = x / |x| = (0, 0.6, 0.8, 0)
\]

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

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);
    }

    constructor
    instance variables
    methods
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());
}
```

Data Visualization

Challenge. Visualize election results.

“If I can’t picture it, I can’t understand it.”
— Albert Einstein

Modular Programming

Modular programming.
- Model problem by decomposing into components.
- Develop data type for each component.

Polygon. Geometric primitive.
Region. Name, postal abbreviation, polygonal boundary.
Vote tally. Number of votes for each candidate.
Election map. Regions and corresponding vote tallies for a given election.
Data Sources

Boundary Data: States within the Continental US

USA data file. State names and boundary points.

Boundary Data: Counties within a State

State data files. County names and boundary points.

Geometric data. [US census bureau]
- www.census.gov/tiger/boundary
- NJ.txt has boundaries of every county in New Jersey.
- USA.txt has boundary of every state.

Election results. [David Leip]
- http://uselectionatlas.org/RESULTS
- Interactive and graphical.
- Need to screen-scrape to get data.

Emerging standard.
- Publish data in text form on the web (like geometric data).
- Write programs to produce visuals (like we’re doing!)
- Mashups.

Format useful for programmers
Format useful for browsers and end-users (need to parse to extract raw data)
Pitfalls: Pieces and Holes

Pieces. A state can be comprised of several disjoint polygons.

Holes. A county can be entirely inside another county.

Screen Scraping the Election Returns

Screen scrape. Download html from web and parse.


county name is text between <b> and </b> tags
that occurs after width:100px

Election Scraper (sketch)

```java
int year = 2008; // election year
String usps = "NJ"; // United States postal code for New Jersey
int fips = 34; // FIPS code for New Jersey

String url = "http://uselectionatlas.org/RESULTS/datagraph.php";
In in = new In(url + "?year=" + year + "&fips=" + fips);
Out file = new Out(usps + year + ".txt");
String input = in.readAll();

while (true) {
    // screen scrape county name
    int p = input.indexOf("width:100px", p);
    if (p == -1) break;
    int from = input.indexOf("<b>", p);
    int to = input.indexOf("</b>", from);
    String county = input.substring(from + 3, to);

    // screen scrape vote totals for each candidate
    String mccain = input.substring(p + 100, p + 100 + 20);
    String obama = input.substring(p + 100 + 20, p + 100 + 40);
    String other = input.substring(p + 100 + 40, p + 100 + 60);

    // save results to file
    file.println(county + "," + mccain + "," + obama + "," + other + ",");
}
```

More Pitfalls

Data sources have different conventions.
- FIPS codes: NJ vs. 34.
- County names: LaSalle vs. La Salle, Kings County vs. Brooklyn.

Plenty of other minor annoyances.
- Unreported results.
- Third-party candidates.
- Changes in county boundaries.

Bottom line. Need to clean up data (but write a program to do it!)
Polygons and Regions

**Polygon Data Type**

**Polygon.** Closed, planar path with straight line segments.

**Simple polygon.** No crossing lines.

**Region Data Type**

**Region.** Represents a state or county.
public class Region {
    private final String name; // name of region
    private final String usps; // postal abbreviation
    private final Polygon poly; // polygonal boundary

    public Region(String name, String usps, Polygon poly) {
        this.name = name;
        this.usps = usps;
        this.poly = poly;
    }

    public void draw() { poly.fill(); }

    public boolean contains(double x0, double y0) {
        return poly.contains(x0, y0);
    }

    public String toString() { … }
}

Election Returns

Screen-scraping results. Votes for McCain, Obama, Other by region.

% more USA2008.txt
Alabama,1265546,813479,19773,
Alaska,193841,123594,8762,
Arizona,1230111,1034707,39020,
Arkansas,638017,422310,26290,
California,5011781,8274473,289260,
Colorado,1073584,1288568,39197,
Connecticut,629428,997772,19592,
Delaware,152374,255459,4579,
District of Columbia,17367,245800,2686,
Florida,4045624,4282074,82621,
Georgia,2048744,1844137,39222,
Hawaii,120566,325871,7131,
Idaho,403012,236440,17978,
Illinois,123695,193812,4223,
Indiana,1229216,170848,4828,
Iowa,78768,141417,2241,
Kansas,398061,304127,12550,
Kentucky,1262393,1677211,43813,
Louisiana,408744,1844137,39222,
Maine,120566,325871,7131,
Maryland,78768,141417,2241,
Massachusetts,1262393,1677211,43813,
Michigan,1262393,1677211,43813,
Minnesota,1262393,1677211,43813,
Mississippi,1262393,1677211,43813,
Missouri,1262393,1677211,43813,
Montana,1262393,1677211,43813,
Nebraska,1262393,1677211,43813,
Nevada,1262393,1677211,43813,
New Hampshire,1262393,1677211,43813,
New Jersey,1262393,1677211,43813,
New Mexico,1262393,1677211,43813,
New York,1262393,1677211,43813,
North Carolina,1262393,1677211,43813,
North Dakota,1262393,1677211,43813,
Ohio,1262393,1677211,43813,
Oklahoma,1262393,1677211,43813,
Oregon,1262393,1677211,43813,
Pennsylvania,1262393,1677211,43813,
Rhode Island,1262393,1677211,43813,
South Carolina,1262393,1677211,43813,
South Dakota,1262393,1677211,43813,
Tennessee,1262393,1677211,43813,
Texas,1262393,1677211,43813,
Utah,1262393,1677211,43813,
Vermont,1262393,1677211,43813,
Virginia,1262393,1677211,43813,
Washington,1262393,1677211,43813,
West Virginia,1262393,1677211,43813,
Wisconsin,1262393,1677211,43813,
Wyoming,1262393,1677211,43813,
**Vote Tally Data Type**

*VoteTally.* Represents the election returns for one region.

New Jersey, USA

Mercer, NJ

<table>
<thead>
<tr>
<th>McCain</th>
<th>Obama</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,613,207</td>
<td>2,215,422</td>
<td>47,826</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>McCain</th>
<th>Obama</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>50,223</td>
<td>107,926</td>
<td>2,229</td>
</tr>
</tbody>
</table>

**Election Map Data Type**

*ElectionMap.* Represents the election map for a given election.

```
public static void main(String[] args) {
    String name = args[0];
    int year = Integer.parseInt(args[1]);
    ElectionMap election = new ElectionMap(name, year);
    election.show();
}
```

```
% java ElectionMap USA 1968
% java ElectionMap NJ 2008

```
Election Map Data Type: Java Implementation

```
public class ElectionMap {
    private final int N;
    private final Region[] regions;
    private final VoteTally[] votes;

    public ElectionMap(String name, int year) {
        In in = new In(name + "\t.txt");
        // read in bounding box and rescale coordinates
        N = in.readInt();
        regions = new Region[N];
        votes = new VoteTally[N];
        for (int i = 0; i < N; i++) {
            String name = in.readLine();
            String usps = in.readLine();
            Polygon poly = new Polygon(in);
            regions[i] = new Region(name, usps, poly);
            votes[i] = new VoteTally(name, usps, year);
        }
    }

    public void show() {
        for (int i = 0; i < N; i++) {
            StdDraw.setPenColor(votes[i].getColor());
            regions[i].draw();
        }
    }
}
```

Modular Programming

Modular program. Collection of data types.

Data Visualization

Visual Display of Quantitative Information

Red states, blue states. Creates a misleading and polarizing picture.

Edward Tufte. Create charts with high data density that tell the truth.
Idea. [Robert J. Vanderbei] Assign color based on number of votes.
- $a_1 =$ McCain votes.
- $a_2 =$ Other votes.
- $a_3 =$ Obama votes.

$$(R, G, B) = \left( \frac{a_1}{a_1 + a_2 + a_3}, \frac{a_2}{a_1 + a_2 + a_3}, \frac{a_3}{a_1 + a_2 + a_3} \right)$$

Implementation. Change one method!

```java
public Color getColor() {
    int tot = dem + rep + ind;
    return new Color((float) rep/tot, (float) ind/tot, (float) dem/tot);
}
```

% java ElectionMap NJ 2004
% java ElectionMap NJ 2008

% java ElectionMap USA 2008
% java ElectionMap USA-county 2008
Remark. Humans perceive red more strongly than blue.

Remark. Amount of color should be proportional to number of votes, not geographic boundary.

Remark. Project latitude + longitude coordinates to 2d plane.

3D visualization. Volume proportional to votes; azimuthal projection.

Cartogram. Area of state proportional to number of electoral votes.
Cartograms

*Cartogram*. Area of country proportional to population.

Summary

**Modular programming**
- Break a large program into smaller independent components.
- Develop a *data type* for each component.
- Ex: Polygon, Region, VoteTally, ElectionMap, In, Out.

Ex 1. Build large software project.
- Software architect specifies API.
- Each programmer implements one module.
- Debug and test each piece independently. [unit testing]

Ex 2. Build reusable libraries.
- Language designer extends language with new data types.
- Programmers share extensive libraries.
- Ex: In, Out, Draw, Polygon, …

Data visualization. You can do it! (worthwhile to learn from Tufte)