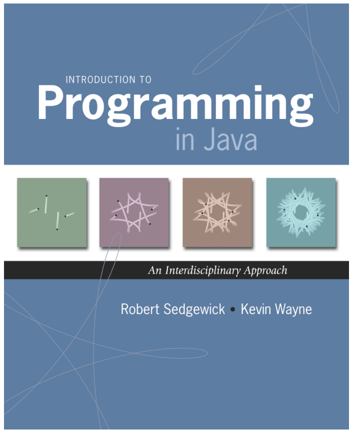
3.3 Designing Data Types



INTRODUCTION TO

Programming

in Java

An Interdisciplinary Approach

Robert Sedgewick · Kevin Wayne

Introduction to Programming in Java: An Interdisciplinary Approach · Robert Sedgewick and Kevin Wayne · Copyright © 2008 · March 29, 2009 1:04 PM

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.

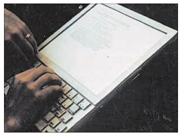
Objected oriented programming (OOP). [noun-oriented]

- 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 percent of the time, you're not aiming high enough."

Alan Kay
2003 Turing Award

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.

Intuition

Client

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

Implementation
- cathode ray tube
- electron gun
- Sony Wega 36XBR250
- 241 pounds

client needs to know
how to use API

implementation needs to know
what API to implement

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

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

Can substitute better implementation without changing the client.

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; }
}
```

Legal Java client.

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

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

Counter Data Type

Counter. Encapsulated data type to count electronic votes.

```
public class Counter {
   private int count;
   public 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.

Changing Internal Representation

Encapsulation.

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

```
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.

- [Y2K] Two digit years: January 1, 2000.
- [Y2038] 32-bit seconds since 1970: January 19, 2038.
- [VIN numbers] We'll run out by 2010.

HI-TECH SUPPLIES
DIGITAL-COMPATIBLE & YEAR 2000-COMPLIANT SYSTEMS IN STOCK NOW

www.cartoonstock.com/directory/m/millenium_time-bomb.asp

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."

Adele Goldberg
Former president of ACM
Co-developed Smalltalk

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

"principle of least privilege"

Immutability

Immutability

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

mutable	*immutable*
`Picture`	`Charge`
`Histogram`	`Color`
`Turtle`	`Stopwatch`
`StockAccount`	`Complex`
`Counter`	`String`
Java arrays	primitive types

Immutability: Advantages and Disadvantages

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

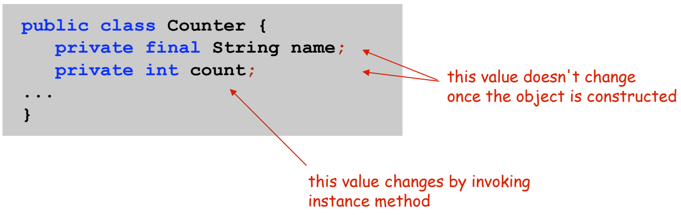
Advantages.

- 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.



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

this value doesn't change once the object is constructed

this value changes by invoking instance method

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		
	`Vector(double[] a)`	*create a vector with the given Cartesian coordinates*
`Vector`	`plus(Vector b)`	*sum of this vector and* `b`
`Vector`	`minus(Vector b)`	*difference of this vector and* `b`
`Vector`	`times(double t)`	*scalar product of this vector and* `t`
`double`	`dot(Vector b)`	*dot product of this vector and* `b`
`double`	`magnitude()`	*magnitude of this vector*
`Vector`	`direction()`	*unit vector with same direction as this vector*

$$x = (0, 3, 4, 0), \quad y = (0, -3, 1, -4)$$

$$x + y = (0, 0, 5, -4)$$

$$3x = (0, 9, 12, 0)$$

$$x \times y = (0 \times 0) + (3 \times -3) + (4 \times 1) + (0 \times -4) = -5$$

$$|\, x \,| = (0^2 + 3^2 + 4^2 + 0^2)^{1/2} = 5$$

$$\vec{x} = x \,/\, |\, x \,| = (0, 0.6, 0.8, 0)$$

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

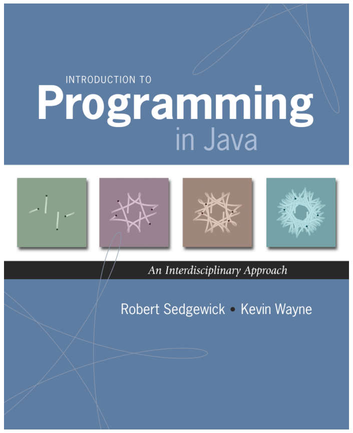
```
public class Vector {
   private int N;
   private double[] coords;                         instance variables

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

   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);                         methods
   }
```

Vector Data Type: Implementation

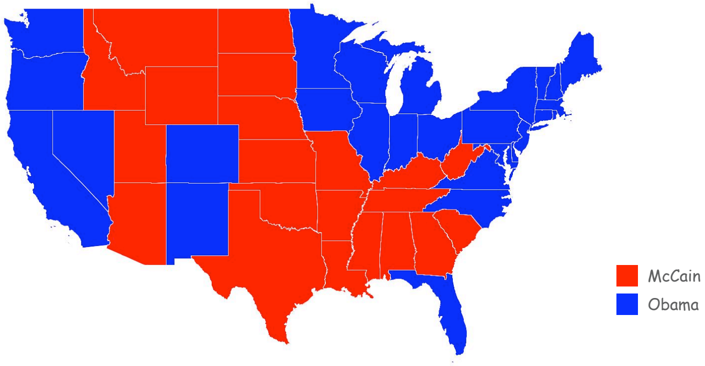
```
public Vector times(double t) {
   double[] c = new double[N];
   for (int i = 0; i < N; i++)
      c[i] = t * coords[i];
   return new Vector(c);
}

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

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

This. The keyword `this` is a reference to the invoking object.
Ex. When you invoke `a.magnitude()`, `this` is an alias for `a`.

3.5 Modular Programming

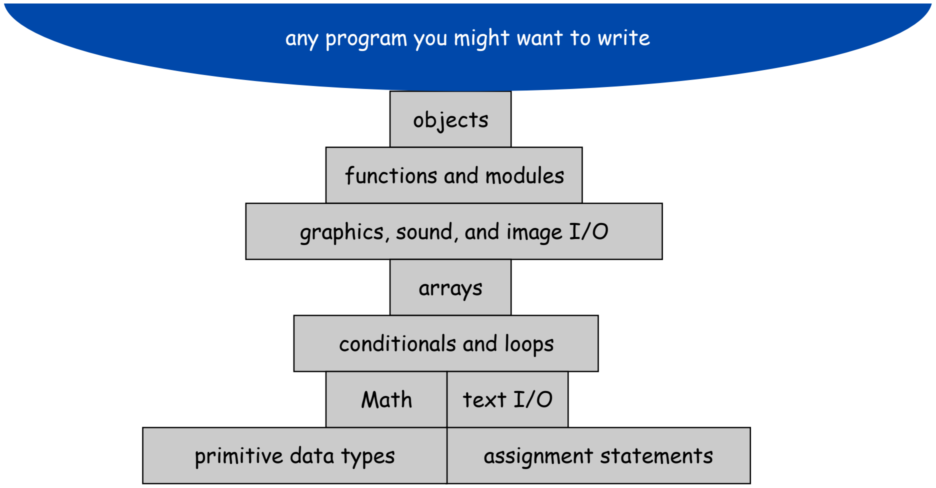


Case Study: Red States, Blue States



2008 Presidential election

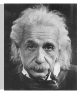
A Foundation for Programming



Data Visualization

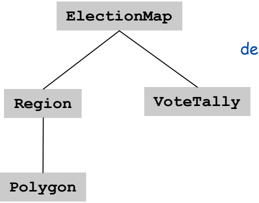
Challenge. Visualize election results.

> *" If I can't picture it, I can't understand it. "*
> *— Albert Einstein*



Basic approach.

- Gather data from web sources, save in local file.
- Build modular program that reads files, draws maps.



dependency graph

Modular Programming

Modular programming.

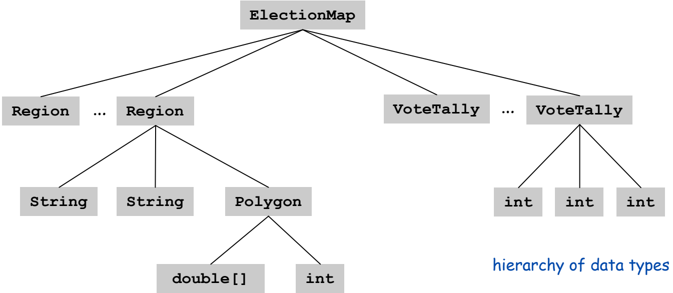
- Model problem by decomposing into components.
- Develop data type for each component.

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.



hierarchy of data types

Data Sources

Boundary Data: States within the Continental US

Geometric data. [US census bureau]

- `www.census.gov/tiger/boundary`
- `NJ.txt` has boundaries of every county in New Jersey.
- `USA.txt` that has boundary of every state.

format useful for programmers

Election results. [David Leip]

- `http://uselectionatlas.org/RESULTS`
- Interactive and graphical.
- Need to screen-scrape to get data.

format useful for browsers and end-users
(need to parse to extract raw data)

Emerging standard.

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

Boundary Data: States within the Continental US

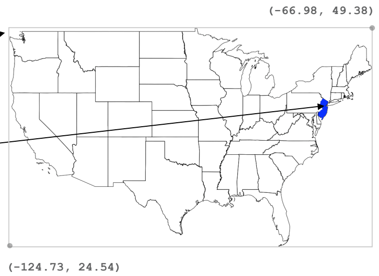
USA data file. State names and boundary points.

```
% more USA.txt
-124.731216 24.544102 -66.980385 49.384365
104
Alabama
USA
498
 -88.200027   34.995548
 -88.202919   35.007942
…
New Jersey
USA
368
 -74.695305   41.357330
 -74.461754   41.250000
 -74.366302   41.202801
…
 -74.721313   41.347294
…
```

number of regions

bounding box

368 points
(longitude, latitude)



Boundary Data: Counties within a State

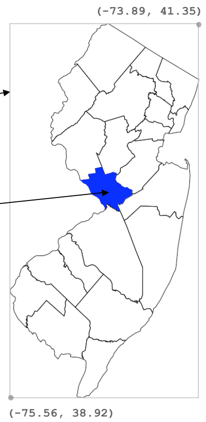
State data files. County names and boundary points.

```
% more NJ.txt
-75.560143 38.928589 -73.894402 41.35733
21
Atlantic
NJ
127
 -74.877563   39.608414
 -74.736694   39.729721
…
Mercer
NJ
88
 -74.748825   40.424248
 -74.722702   40.375301
 -74.674507   40.384399
…
 -74.808403   40.415401
…
```

number of regions

bounding box

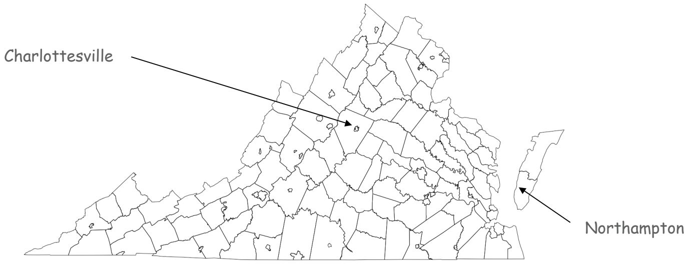
88 points
(longitude, latitude)



Pitfalls: Pieces and Holes

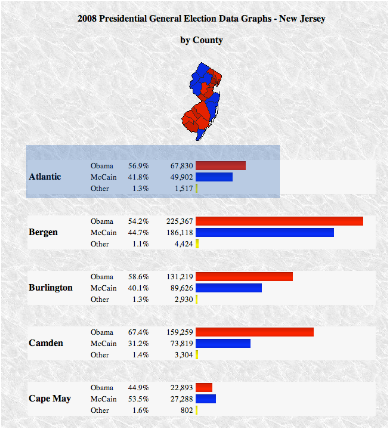
Pieces. A state can be comprised of several disjoint polygons.

Holes. A county can be entirely inside another county.



Screen Scraping the Election Returns

Screen scrape. Download html from web and parse.



2008 Presidential General Election Data Graphs - New Jersey

by County

Atlantic	Obama	56.9%	67,830
	McCain	41.8%	49,902
	Other	1.3%	1,517
Bergen	Obama	54.2%	225,367
	McCain	44.7%	186,118
	Other	1.1%	4,424
Burlington	Obama	58.6%	131,219
	McCain	40.1%	89,626
	Other	1.3%	2,930
Camden	Obama	67.4%	159,259
	McCain	31.2%	73,819
	Other	1.4%	3,304
Cape May	Obama	44.9%	22,893
	McCain	53.5%	27,288
	Other	1.6%	802

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

```
<div>
<br /><b>2008 Presidential General Election Data Graphs - New Jersey<br /><br />by
County</b><br /><br /><img src="img.php?
year=2008&amp;st=NJ&amp;type=map&amp;off=0&amp;fips=34&amp;elect=0" alt="Map" /><br
/><br /><div class="info"><table cellpadding="2"><tr><td style="width:100px"
rowspan="3"><b>Atlantic</b></td><td class="cnd">Obama</td><td
class="per">56.9&#37;</td><td class="dat">67,830</td><td class="bar"><div
class="bardem" style="width:28.2%">&nbsp;</div></td></tr><tr><td>McCain</td><td
class="per">41.8&#37;</td><td class="dat">49,902</td><td><div class="barrep"
style="width:20.8%">&nbsp;</div></td></tr><tr><td>Other</td><td
class="per">1.3&#37;</td><td class="dat">1,517</td><td><div class="baroth"
style="width:1.0%">&nbsp;</div></td></tr></table><br /><br /><table
cellpadding="2"><tr><td style="width:100px" rowspan="3"><b>Bergen</b></td><td
class="cnd">Obama</td><td class="per">54.2&#37;</td><td class="dat">225,367</td><td
class="bar"><div class="bardem"
style="width:93.8%">&nbsp;</div></td></tr><tr><td>McCain</td><td
class="per">44.7&#37;</td><td class="dat">186,118</td><td><div class="barrep"
style="width:77.5%">&nbsp;</div></td></tr><tr><td>Other</td><td
class="per">1.1&#37;</td><td class="dat">4,424</td><td><div class="baroth"
style="width:1.8%">&nbsp;</div></td></tr></table><br /><br /><table
cellpadding="2"><tr><td style="width:100px" rowspan="3"><b>Burlington</b></td><td
class="cnd">Obama</td><td class="per">58.6&#37;</td><td class="dat">131,219</td><td
class="bar"><div class="bardem"
style="width:54.6%">&nbsp;</div></td></tr><tr><td>McCain</td><td
class="per">40.1&#37;</td><td class="dat">89,626</td><td><div class="barrep"
style="width:37.3%">&nbsp;</div></td></tr><tr><td>Other</td><td
class="per">1.3&#37;</td><td class="dat">2,930</td><td><div class="baroth"
style="width:1.2%">&nbsp;</div></td></tr></table><br /><br /><table
cellpadding="2"><tr><td style="width:100px" rowspan="3"><b>Camden</b></td><td
class="cnd">Obama</td><td class="per">67.4&#37;</td><td class="dat">159,259</td><td
class="bar"><div class="bardem"

…
```

`http://uselectionatlas.org/RESULTS/datagraph.php?year=2008&fips=34`

Election Scraper (sketch)

```
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


  // save results to file
  file.println(county + "," + bush + "," + kerry + "," + nader + ",");
}
```

extract text between <b> and </b> tags, that occurs after `width:100px`

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.

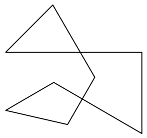
- Unreported results.
- Third-party candidates.
- Changes in county boundaries.

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

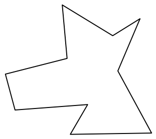
Polygons and Regions

Polygon Data Type

Polygon. Closed, planar path with straight line segments.
Simple polygon. No crossing lines.



polygon
(8 points)



simple polygon
(10 points)

simple polygon
(368 points)

Polygon Data Type: Java Implementation

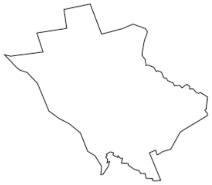
```
public class Polygon {
   private final int N;           // number of boundary points
   private final double[] x, y;  // the points (x[i], y[i])

   // read from input stream
   public Polygon(In in) {
       N = in.readInt();
       x = new double[N];
       y = new double[N];
       for (int i = 0; i < N; i++) {
          x[i] = in.readDouble();
          y[i] = in.readDouble();
       }
   }

   public void fill() { StdDraw.filledPolygon(x, y); }

   public boolean contains(double x0, double y0)    { … }
   public String toString()                          { … }
}
```

see COS 226

easy

Region Data Type

Region. Represents a state or county.



Mercer, NJ
88 point polygon

New Jersey, USA
368 point polygon

Region Data Type: Java Implementation

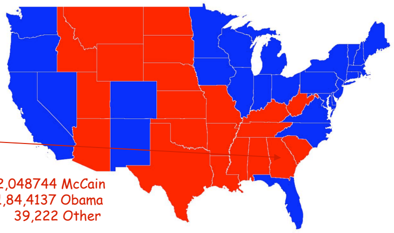
```
public class Region {
   private final String  name;  // name of region
   private final String  usps;  // postal abbreviation
   private final Polygon poly;  // polygonal boundary

   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

Election Returns: By State

Screen-scraping results. Number of votes for Bush, Kerry, Nader by region.

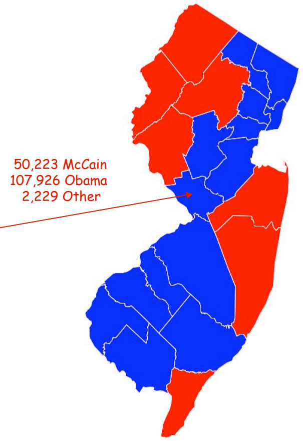
```
% more USA2008.txt
Alabama,1266546,813479,19773,
Alaska,193841,123594,8762,
Arizona,1230111,1034707,39020,
Arkansas,638017,422310,26290,
California,5011781,8274473,289260,
Colorado,1073584,1288568,39197,
Connecticut,629428,997772,19592,
Delaware,152374,255459,4579,
District of Columbia,17367,245800,2686,
Florida,4045624,4282074,82621,
Georgia,2048744,1844137,39222,
Hawaii,120566,325871,7131,
Idaho,403012,236440,17978,
Illinois,2031527,3419673,71851,
…
Virginia,1725005,1959532,38723,
Washington,1229216,1750848,68820,
West Virginia,398061,304127,12550,
Wisconsin,1262393,1677211,43813,
Wyoming,164958,82868,6832,
```



Election Returns: By County

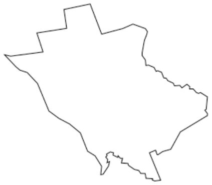
Screen-scraping results. Number of votes for Bush, Kerry, Nader by region.

```
% more NJ2008.txt
Atlantic,49902,67830,1517,
Bergen,186118,225367,4424,
Burlington,89626,131219,2930,
Camden,73819,159259,3304,
Cape May,27288,22893,802,
Cumberland,22360,34919,915,
Essex,74063,240306,2181,
Gloucester,60315,77267,1848,
Hudson,55360,154140,2116,
Hunterdon,39092,29776,1147,
Mercer,50223,107926,2229,
Middlesex,123695,193812,4283,
Monmouth,160433,148737,4244,
Morris,132331,112275,2913,
Ocean,160677,110189,4111,
Passaic,72552,113257,1904,
Salem,14816,16044,672,
Somerset,70085,79321,1672,
Sussex,44184,28840,1393,
Union,78768,141417,2241,
Warren,27500,20628,980,
```



Vote Tally Data Type

VoteTally. Represents the election returns for one region.



Mercer, NJ

New Jersey, USA

election returns

50,223 McCain
107,926 Obama
2,229 Other

1,613,207 McCain
2,215,422 Obama
47,826 Other

Vote Tally Data Type: Java Implementation

```
public class VoteTally {
   private final int rep, dem, ind;

   public VoteTally(String name, String usps, int year) {
      In in = new In(usps + year + ".txt");
      String input = in.readAll();
      int i0 = input.indexOf(name);
      int i1 = input.indexOf(",", i0+1);
      int i2 = input.indexOf(",", i1+1);
      int i3 = input.indexOf(",", i2+1);
      int i4 = input.indexOf(",", i3+1);
      rep = Integer.parseInt(input.substring(i1+1, i2));
      dem = Integer.parseInt(input.substring(i2+1, i3));
      ind = Integer.parseInt(input.substring(i3+1, i4));
   }

   public Color getColor() {
      if (rep > dem) return StdDraw.RED;
      if (dem > rep) return StdDraw.BLUE;
      return StdDraw.BLACK;
   }
}
```

```
   % more NJ2008.txt
   …
   Mercer,50223,107926,2229,
   …
i0       i1     i2      i3    i4
```

Election Map

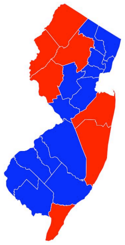
Election Map Data Type

ElectionMap. Represents the election map for a given election.

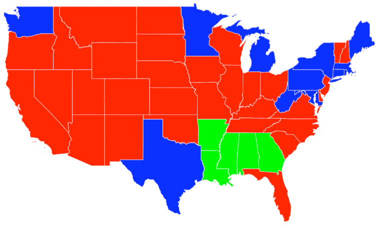
client

```
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 NJ 2008
```



```
% java ElectionMap USA 1968
```



Election Map Data Type: Java Implementation

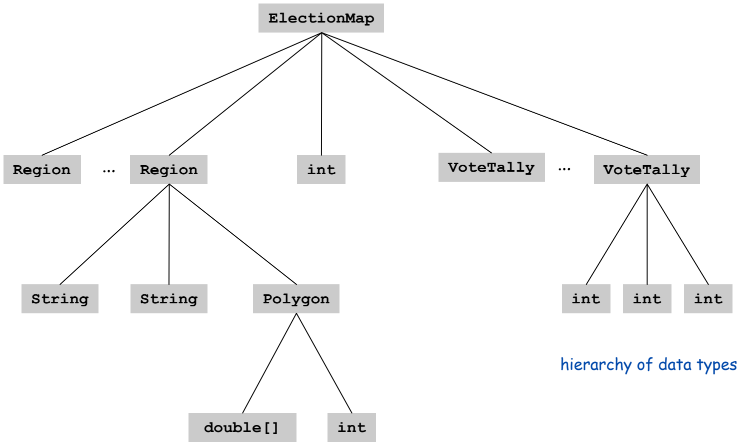
```
public class ElectionMap {
   private final int N;
   private final Region[] regions;
   private final VoteTally[] votes;

   public ElectionMap(String name, int year) {
      In in = new In(name + ".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();
      }
   }
}
```

use polygon, region, and vote tally data types to build map

draw map

Modular Programming

Modular program. Collection of data types.

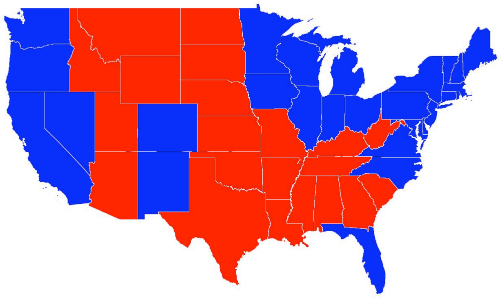


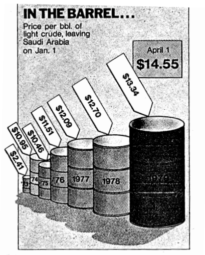
hierarchy of data types

Data Visualization

Visual Display of Quantitative Information

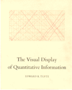
Red states, blue states. Creates a misleading and polarizing picture.

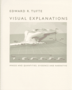


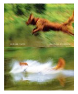


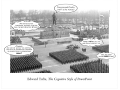
Time, April 9, 1979, p. 57.

Edward Tufte. Create charts with high data density that tell the truth.









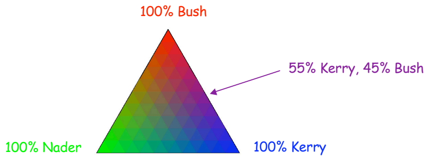
Purple America

Idea. [Robert J. Vanderbei] Assign color based on number of votes.

http://www.princeton.edu/~rvdb/JAVA/election2004

- a_1 = Bush votes.
- a_2 = Nader votes.
- a_3 = Kerry votes.

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

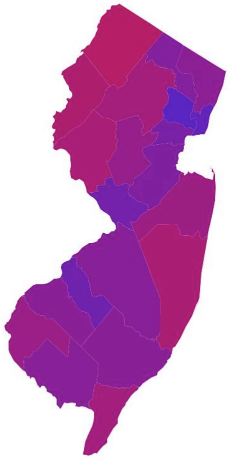


Implementation. Change one method!

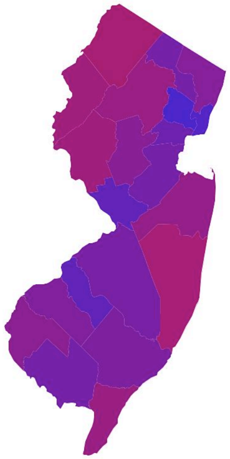
```
public Color getColor() {                                    VoteTally.java
   int tot = dem + rep + ind;
   return new Color((float) rep/tot, (float) ind/tot, (float) dem/tot);
}
```

Purple New Jersey

```
% java ElectionMap NJ 2004
```

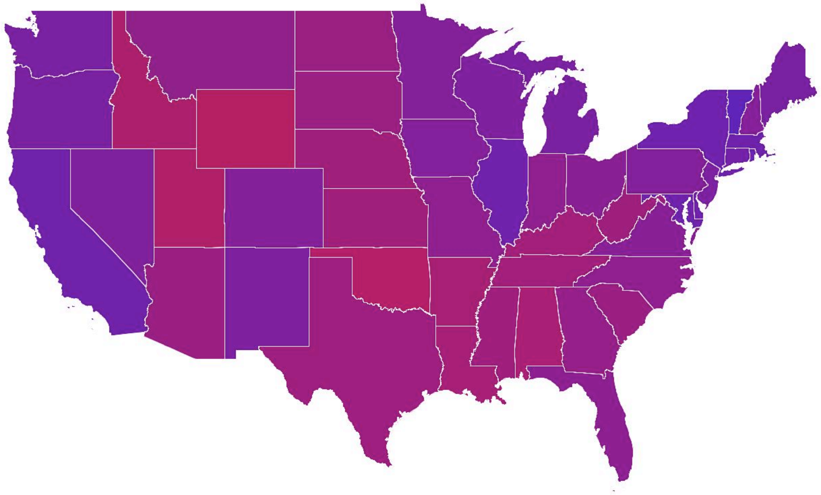


```
% java ElectionMap NJ 2008
```



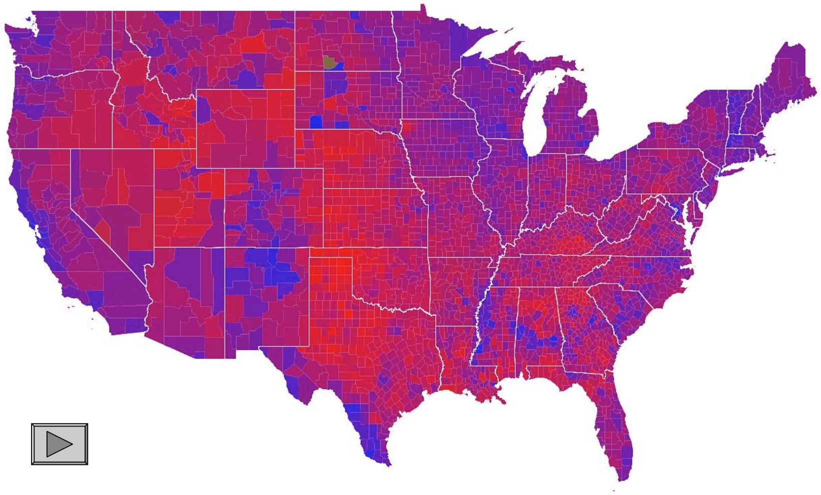
Purple America

```
% java ElectionMap USA 2008
```



Purple America

```
% java ElectionMap USA-county 2008
```

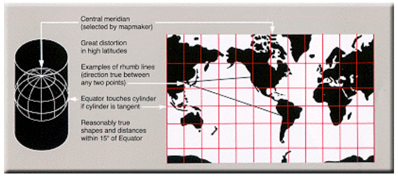


Data Visualization: Design Issues

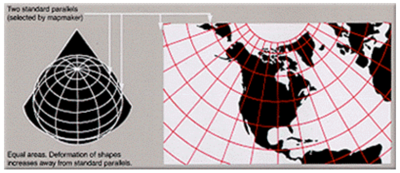
Remark. Humans perceive red more strongly than blue.

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

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



Mercator projection

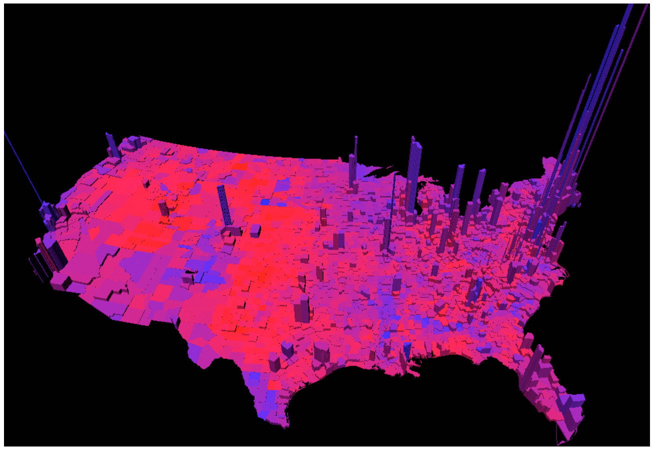


Two standard parallels
(selected by mapmaker)
Equal areas. Deformation of shapes
increases away from standard parallels.

Albers projection

3D Visualization

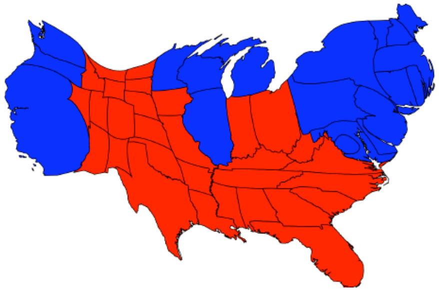
3D visualization. Volume proportional to votes; azimuthal projection.



Robert J. Vanderbei
www.princeton.edu/~rvdb/JAVA/election2004

Cartograms

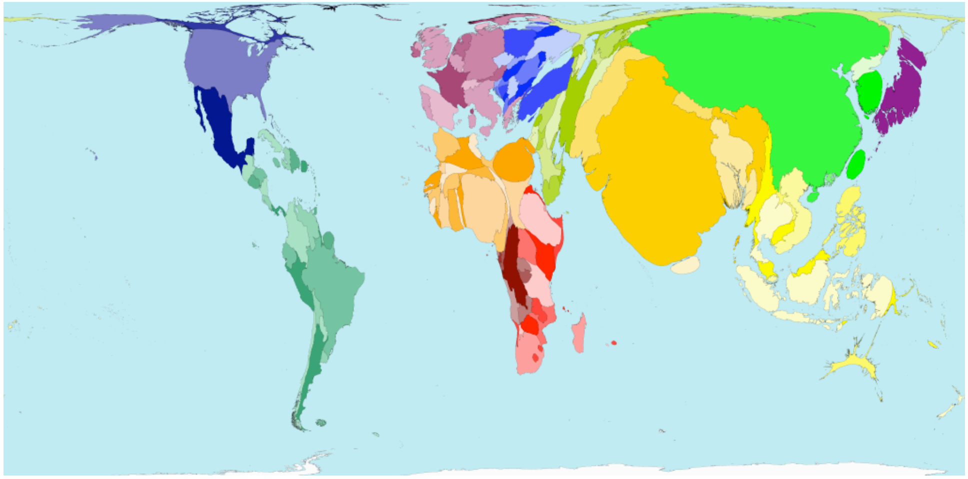
Cartogram. Area of state proportional to number of electoral votes.

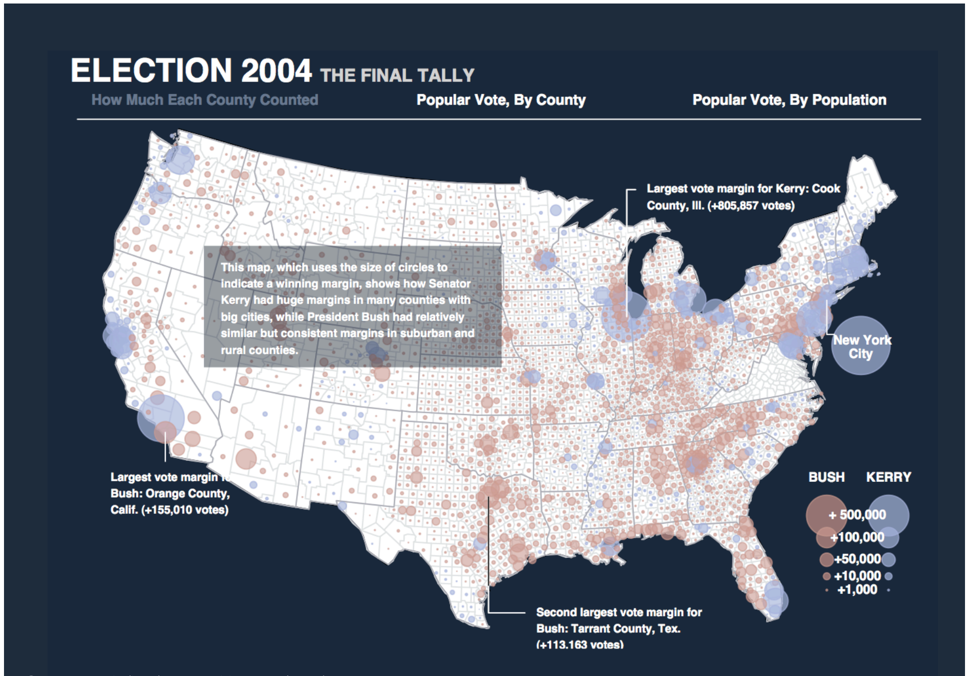


Michael Gastner, Cosma Shalizi, and Mark Newman
`www-personal.umich.edu/~mejn/election`

Cartograms

Cartogram. Area of country proportional to population.




ELECTION 2004 THE FINAL TALLY
How Much Each County Counted
Popular Vote, By County
Popular Vote, By Population
Largest vote margin for Kerry: Cook County, Ill. (+805,857 votes)
New York City
Largest vote margin for Bush: Orange County, Calif. (+155,010 votes)
Second largest vote margin for Bush: Tarrant County, Tex. (+113.163 votes)
BUSH
KERRY
+ 500,000
+100,000
+50,000
+10,000
+1,000

This map, which uses the size of circles to indicate a winning margin, shows how Senator Kerry had huge margins in many counties with big cities, while President Bush had relatively similar but consistent margins in suburban and rural counties.

Eugene Pyatigorsky (based on New York Times visualization)
`www.ocf.berkeley.edu/~gene/media/docs/Election.pdf`

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)