Today's topics

Java
  Numbers
  Iteration
Upcoming
  More Java

Reading
  Great Ideas, Chapter 3

Numbers

- Have talked about and used numbers before
- Have built-in or “primitive types”
  - int for whole numbers
  - double for numbers that may include fractions
- One of simplest uses for integers is counting
  - Following example counts number of times button is pressed
    - Includes statement: numTotal = numTotal + 1;
      - Not an equality!
      - Evaluate right; copy into left

Counting Example

```java
public class Count extends java.applet.Applet
implement ActionListener
{
  TextField mQuery, mTotal;
  Button bCount;
  int noTotal = 0;
  public void init ()
  {
    mQuery = new TextField(80);
    mQuery.setText("Keep track of attendance by pushing the button.");
    bCount = new Button("Register");
    mTotal = new TextField(40);
    mTotal.setText("The total attendance is " + noTotal);
    bCount.addActionListener(this);
    add(mQuery); add(bCount); add(mTotal);
    noTotal = noTotal + 1;
    mTotal.setText("The total attendance is " + noTotal);
  }
}
```

Counting Example

```java
public void actionPerformed(ActionEvent event)
{
  Object cause = event.getSource();
  if (cause == bCount) // (could have omitted)
  {
    noTotal = noTotal + 1;
    mTotal.setText("The total attendance is " + noTotal + "");
  }
}
```

- Can extend this to multiple counts
Tallies

public class Tallies extends java.applet.Applet implements ActionListener
{
TextField mQuery, mAnsStu, mAnsFac, mAnsSta, mTotal;
Button bStudents, bFaculty, bStaff;
int noStu = 0, noFac = 0, noSta = 0, noTotal = 0;

public void init()
{
    mQuery = new TextField(80);
mQuery.setText("Keep track of attendance by pushing the buttons.");
bStudents = new Button("Students");
bFaculty = new Button("Faculty");
bStaff = new Button("Staff");
mAnsStu = new TextField(12);
mAnsFac = new TextField(12);
mAnsSta = new TextField(12);
mTotal = new TextField(80);
    bStudents.addActionListener(this);
bFaculty.addActionListener(this);
bStaff.addActionListener(this);
    add(mQuery); add(bStudents); add(bFaculty);
    add(bStaff); add(mTotal); add(mAnsStu); add(mAnsFac);
    add(mAnsSta);
mTotal.setText("The total attendance is " + noTotal + 
        " Subtotals are shown below.");
mAnsStu.setText(noStu + " students");
mAnsFac.setText(noFac + " faculty");
mAnsSta.setText(noSta + " staff");
}

Tallies (p2)

mAnsSta = new TextField(12);
mTotal = new TextField(80);
bStudents.add ActionListener(this);
bFaculty.add ActionListener(this);
bStaff.add ActionListener(this);
    add(mQuery); add(bStudents); add(bFaculty);
    add(bStaff); add(mTotal); add(mAnsStu); add(mAnsFac);
    add(mAnsSta);
mTotal.setText("The total attendance is " + noTotal + 
        " Subtotals are shown below.");
mAnsStu.setText(noStu + " students");
mAnsFac.setText(noFac + " faculty");
mAnsSta.setText(noSta + " staff");
}

Tallies (p3)

public void actionPerformed(ActionEvent event)
{
    Object cause = event.getSource();
    if (cause == bStudents)
    {
        noStu = noStu + 1;
nTotal = noTotal + 1;
mTotal.setText("The total attendance is " + 
            noTotal + " Subtotals are shown below.");
mAnsStu.setText(noStu + " students");
    }
    // similar blocks follow for faculty & staff

Tallies (p4)

if (cause == bFaculty)
{
    noFac = noFac + 1;
nTotal = noTotal + 1;
mTotal.setText("The total attendance is " + noTotal + 
            " Subtotals are shown below.");
mAnsFac.setText(noFac + " faculty");
}
if (cause == bStaff)
{
    noSta = noSta + 1;
nTotal = noTotal + 1;
mTotal.setText("The total attendance is " + noTotal + 
            " Subtotals are shown below.");
mAnsSta.setText(noSta + " staff");
}
**Numbers**

- Have classes similar to TextField to do I/O with them
  - Have used IntField methods before
  - Here is class declaration

```java
public class IntField extends TextField {
    public IntField();
    public IntField(int size);
    public void setInt(int number);
    public int getInt();
}
```

**Doubles**

- Similarly for doubles, we have

```java
public class DoubleField extends TextField {
    public DoubleField();
    public DoubleField(int size);
    public void setDouble(double num);
    public double getDouble();
}
```

**Calculations**

- Can use doubles and DoubleFields for real computations
- Here is an applet to calculate volume of a cylinder
  - Need to supply
    - radius
    - length
  - Use formula
    - \( \text{volume} = (\text{area-of-end}) \times \text{length} = \pi \times \text{radius} \times \text{radius} \times \text{length} \)

**Calculate volume cylinder**

```java
public class Numbers extends java.applet.Applet implements ActionListener {
    TextField instruct, result, mRadius, mLength;
    DoubleField gRadius, gLength;
    Button bCompute;
    double radius, length, cylVol, PI = 3.14159265;
    public void init () {
        instruct = new TextField(72);
        instruct.setText("Please enter radius and length below.");
        mRadius = new TextField(9);
        mRadius.setText("radius: ");
        mLength = new TextField(9);
        mLength.setText("length: ");
        gRadius = new DoubleField(10);
        ```
Calculate volume cylinder (p2)

gRadius = new DoubleField(10);
gLength = new DoubleField(10);
result = new TextField(72);
result.setText("The volume of the cylinder is: \( \pi \times r^2 \times h \)\);
bCompute = new Button("Compute");
bCompute.addActionListener(this);
add(instruct); add(mRadius); add(gRadius);
add(mLength); add(gLength); add(bCompute); add(result);

Calculate volume cylinder (p3)

public void actionPerformed(ActionEvent event) {
    Object cause = event.getSource();
    if (cause == bCompute) {
        length = gLength.getDouble();
        radius = gRadius.getDouble();
        cylVol = pi * radius * radius * length;
        result.setText("The volume of the cylinder is: \( \pi \times r^2 \times h \)\);
    }
}

Iteration

- Iteration -- also called repetition or looping
- Iteration by Button pushing
  - Often need to repeat a calculation with minor changes
  - Sometimes refine previous solution
  - Sometimes calculate successive values in a series
  - Can do this under control of a Button
- Applets have the concept of iteration built into their very nature
  - Waiting for a Button to be pressed implies a loop

Iteration by Button pushing example

public class ButCompound extends java.applet.Applet implements ActionListener {

    TextField mInstruct, mBalance;
    DoubleField gRate, gPrinc, gPay;
    Button bStart, bNextInstallment;
    double rate, princ, pay, balance;
    int months;

    public void init()
    {
        mInstruct = new TextField(80);
        mInstruct.setText("Enter principal, rate, payment, then press Start");
    }

Iteration by Button pushing example.2

gPrinc = new DoubleField(10);
gRate = new DoubleField(10);
gPay = new DoubleField(10);
mBalance = new TextField(80);
bStart = new Button("Start");
bNextInstallment = new Button("Next Installment");
bStart.addActionListener(this);
bNextInstallment.addActionListener(this);
add(mInstruct); add(gPrinc); add(gRate);
add(gPay); add(bStart); add(bNextInstallment);
add(mBalance);
}

Iteration by Button pushing example.3

public void actionPerformed(ActionEvent event)
{
    Object cause = event.getSource();
    if (cause == bStart)
    {
        princ = gPrinc.getDouble();
        rate = gRate.getDouble()/12;
        pay = gPay.getDouble();
        months = 0;
        balance = princ;
        mInstruct.setText("Press Next Installment for next Balance");
        mBalance.setText("Start with a balance of " + balance);
    }
}

Iteration by Button pushing example.4

if (cause == bNextInstallment)
{
    months = months + 1;
    balance = balance*(1.0 + rate) - pay;

    mBalance.setText("After " + months +
     " months at " + 100*rate*12 +
     "% and payments of " + pay +
     " the balance is " + balance);
}