1.00/1.001/1.002
Introduction to Computers and Engineering Problem Solving

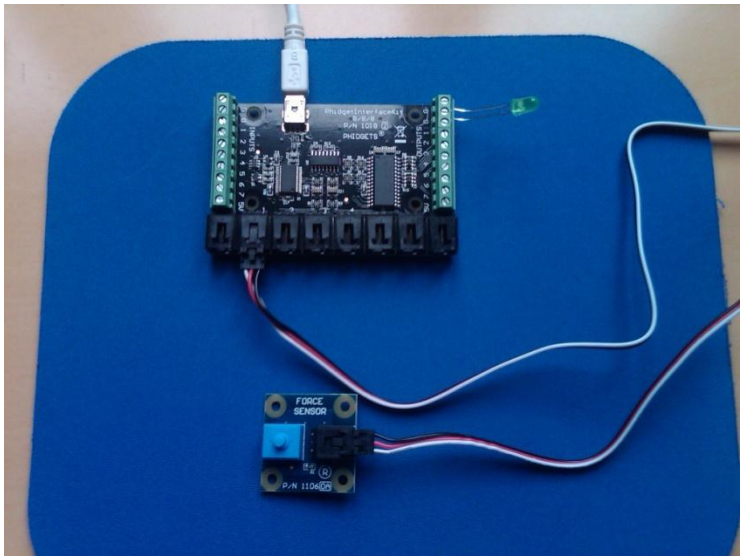
Recitation 8

Phidget Setup, Model-View-Controller, 2D API, Affine Transformations

April 9 & 10 2012

Phidget Interface Kit

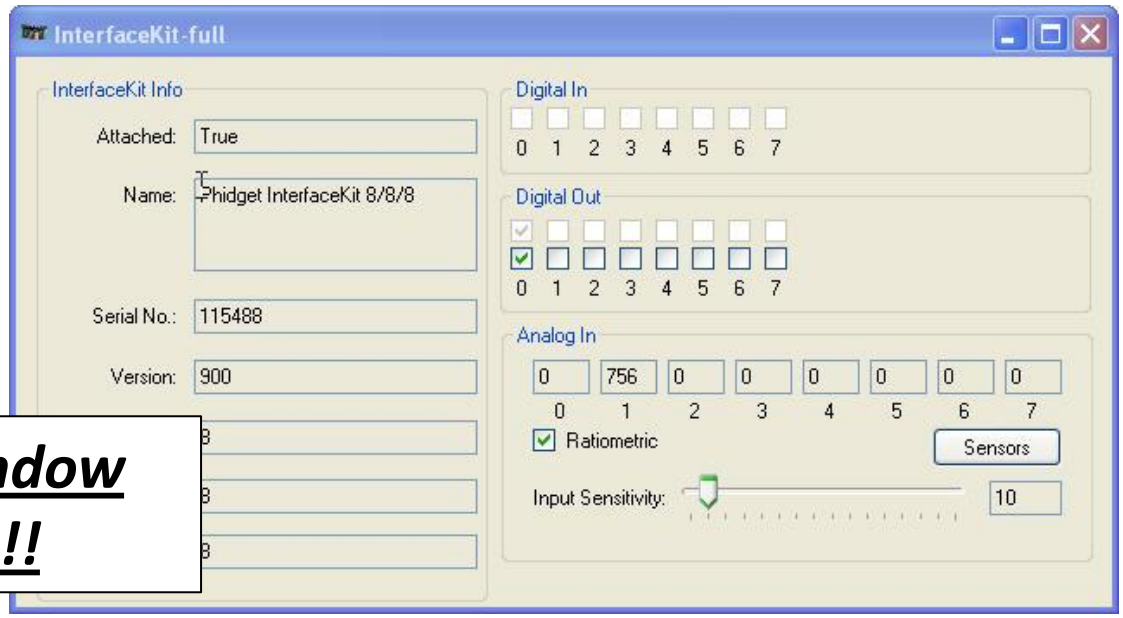
1. Download the Phidgets software for your OS from www.phidgets.com/drivers.php.
2. Install it. Choose 32 bit or 64 bit version to match your OS and the version of Java you installed
3. Download the phidget.jar file from www.phidgets.com/programming_resources.php
4. Unzip it to someplace where you can find it again.
5. Open your Phidget kit and find:
    1. USB cable to connect the interface board to your computer
    2. Interface board (1018)
    3. Force sensor (1106) with its cable
    4. A green LED
6. Connect them as in the image
    - USB from laptop to interface board
    - Force sensor to Analog In1
    - LED wired between GND and Digital Out 0
    - Short wire in GND, long in Digital Out 0
    - Use the screwdriver for the LED



Phidget Interface Kit, 2

- If you have installed the Phidget software, you should see a Phidget icon in your taskbar:

- Click it. It should bring up the Phidget test application
  - If it brings up the Phidget control panel, click the General tab and then double click the Phidget interface kit device to bring up the test app
- Press the Phidget force sensor button. Watch the reading change.
- Click Digital Out box 0
- LED should light up



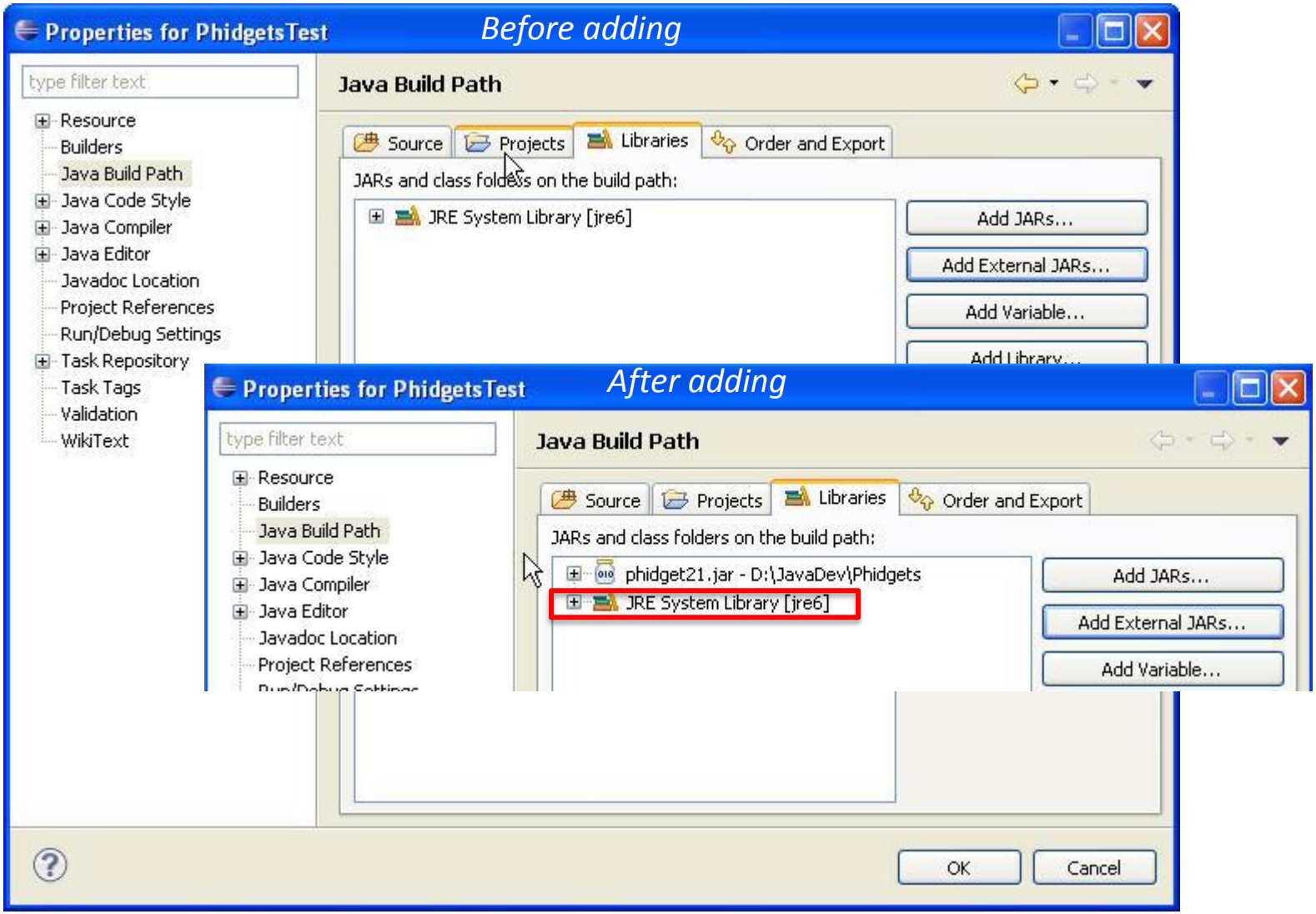
You must close this window when using Eclipse!!!

Courtesy of Phidgets. Used with permission.

Phidgets and Java

- Download PressureController.java and compile it in a new project
  - You get errors because Eclipse can't find the Phidget.jar file, the library that tells Java how to communicate with Phidgets
- Open the Java Properties/Java Build Path popup by right clicking on the project
- Click "Add External Jars…" and navigate to where you unzipped the phidget21.jar file
- Select it and click Open, and then OK
  - Next slide shows before and after shots
  - Errors will disappear from java files
- Run PressureController

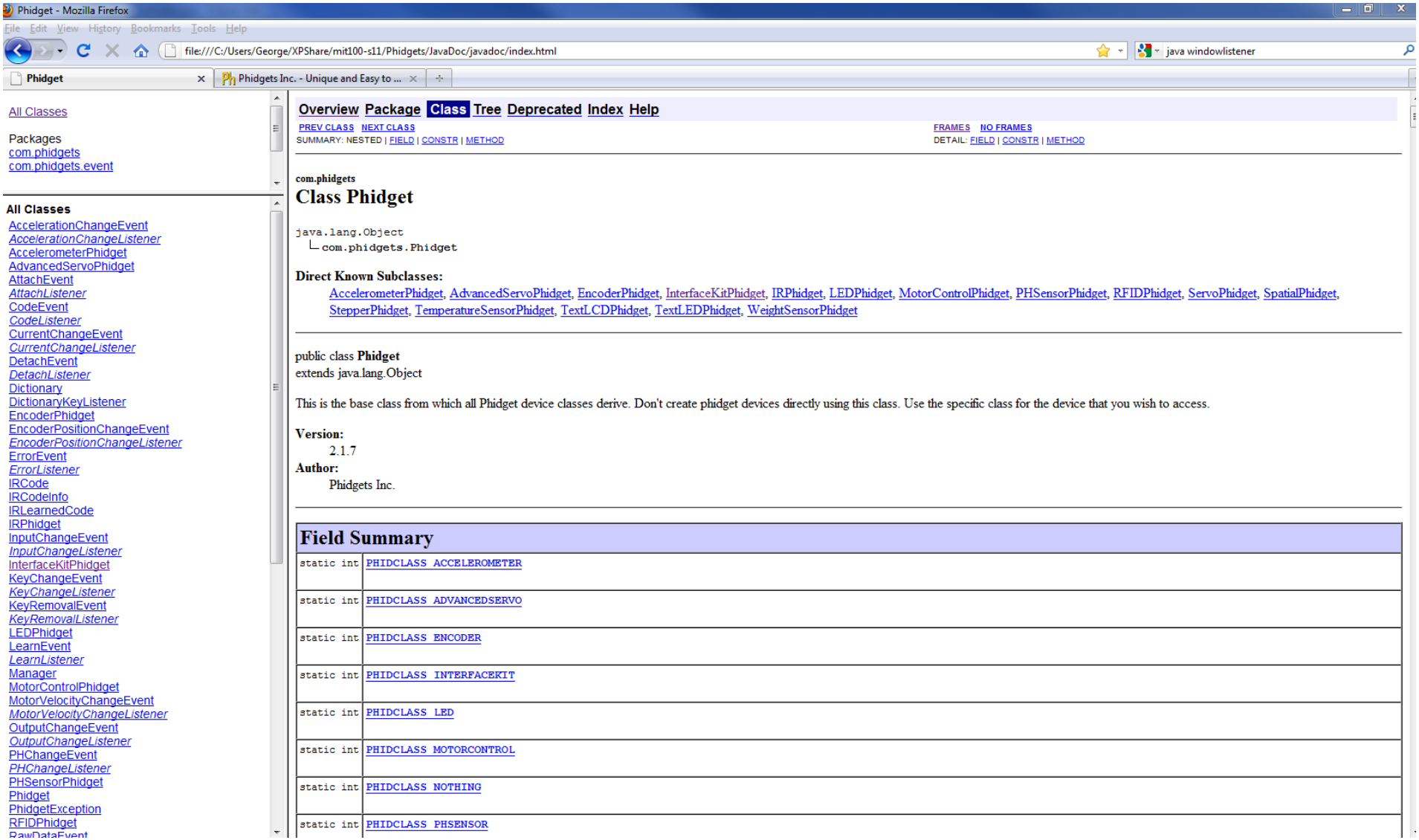
Phidgets and Java



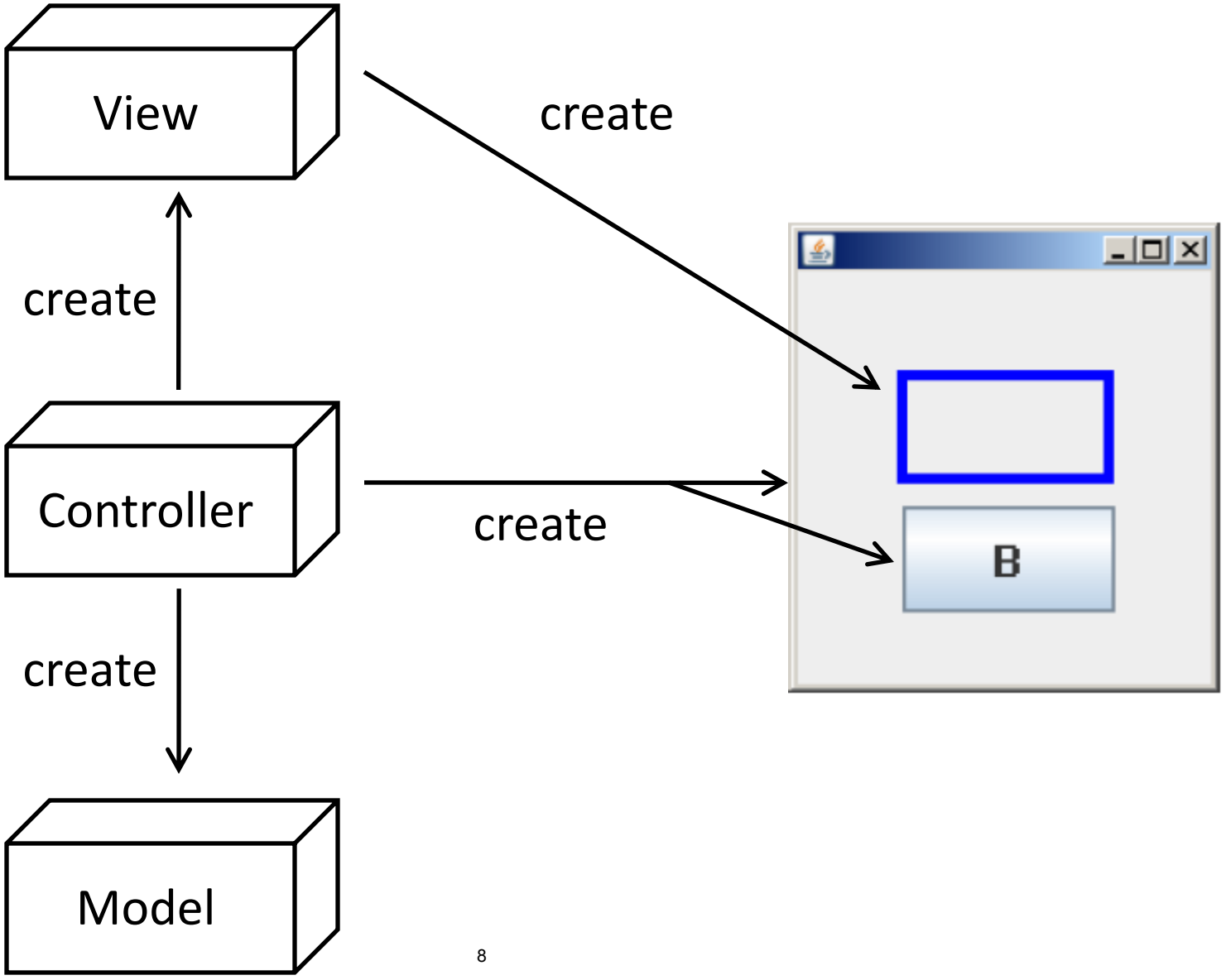
Phidget21.jar

- Jar file is a Java archive
  - Zip format of compiled (byte code) Java classes
- By placing it in your project, you can use all its classes
- See its documentation (API Reference) for a list of classes and methods. Download from
  - phidgets.com/programming_resources.php
  - Unzip
  - Bookmark it in your browser
- Also look at Java *Getting Started Guide*

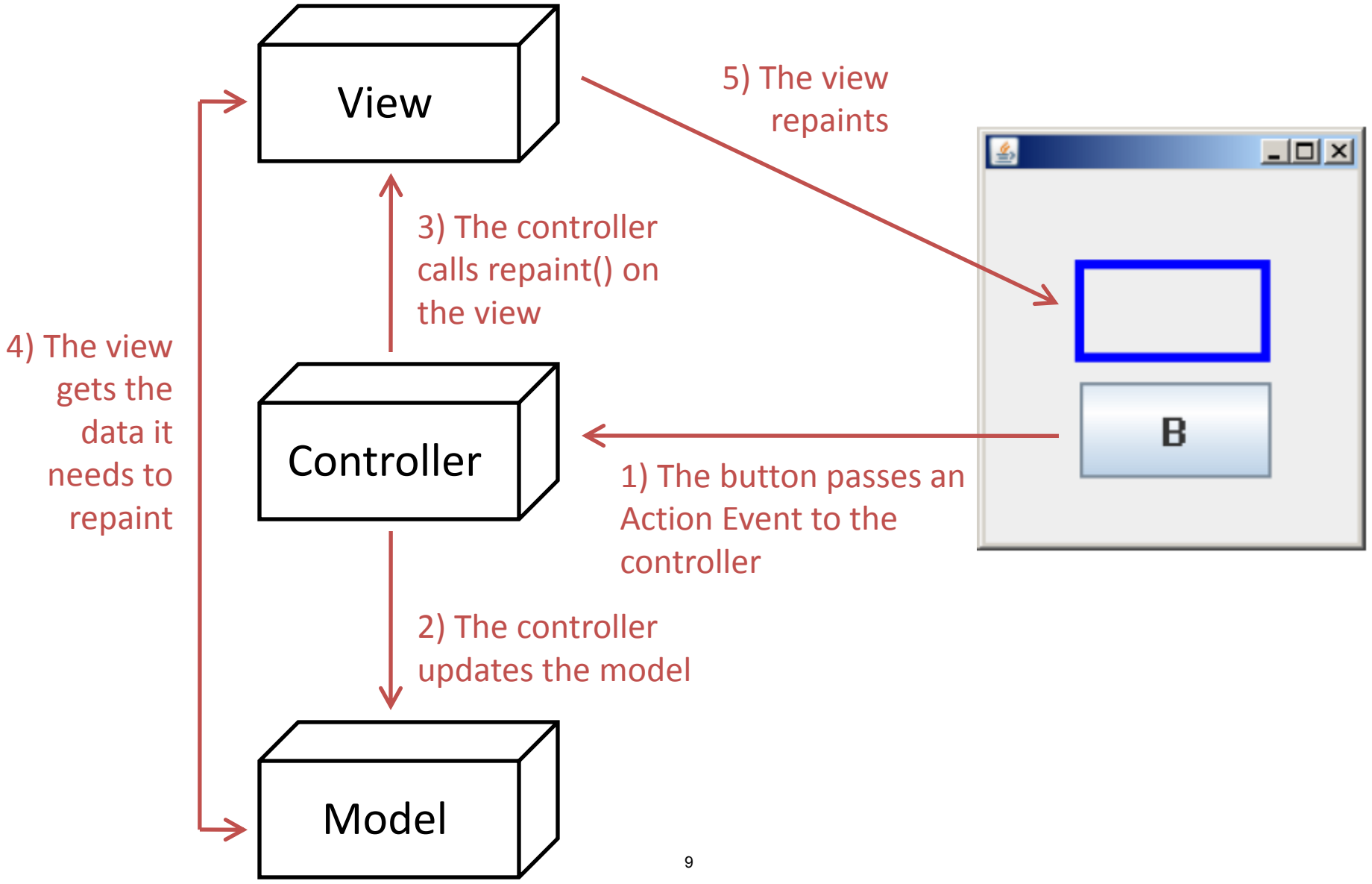
Phidgets Javadoc



Model-View-Controller Construction



Model-View-Controller Operation



Model-View-Controller

```
public class Model
{
    // data members

    public Model(...){...}

    //get/set methods for the
    //View/Controller respectively
}
```

```
public class View extends JPanel
{
    private Model model;

    public View(Model m) {
        model = mo;
    }

    public void paintComponent(...)
    {
        // drawing statements using
        // data from model
    }
}
```

```
public class Controller extends JFrame
{
    private Model model;
    private View view;
    private JButton b;

    public Controller() {
        model = new Model(...);
        view = new View(m);
        getContentPane.add(v)

        b = new JButton("...");
        b.addActionListener(...);
        getContentPane.add(b)

        // other JComponents and
        // anonymous inner classes
        // that modify the Model/View
    }

    public static void main(...){
        Controller c =new Controller();
    }
}
```

Model-View-Controller Exercise

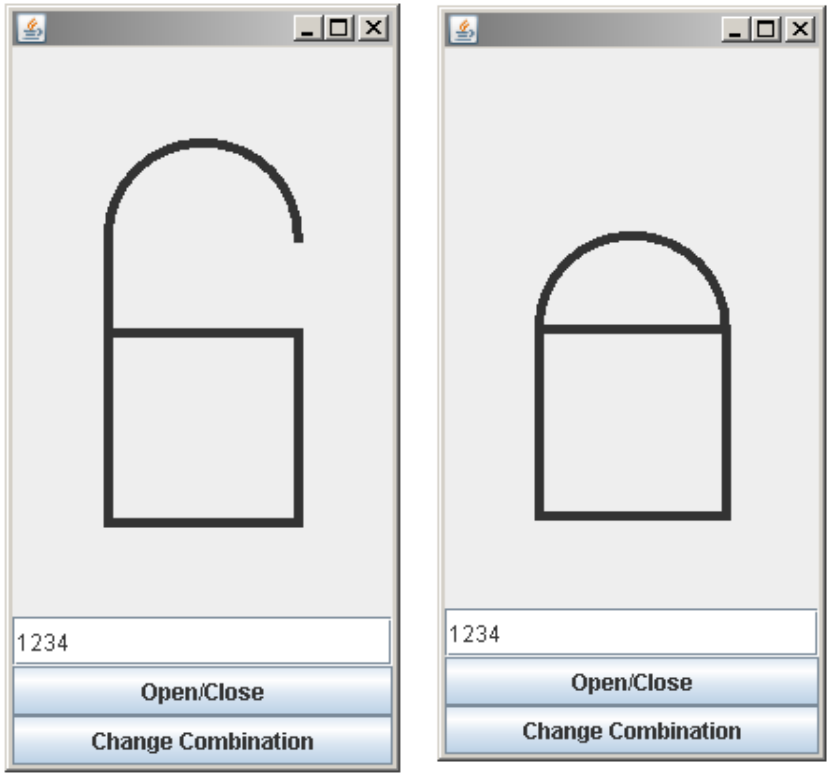
We will write another version of the combination lock program from recitation 7 using, this time, a model-view-controller implementation.

Open/Close button

- if the lock is opened, close it.
- if the lock is closed, open it if the digits match the combination

Change Combination button:

- when clicked while lock is opened, set the combination to the current digits



LockModel

Create a model class called **`LockModel`** for the lock. A new lock is initially opened.

- Data members to keep track of the state of the lock?
- Methods to interact with controller and view?

LockView (1)

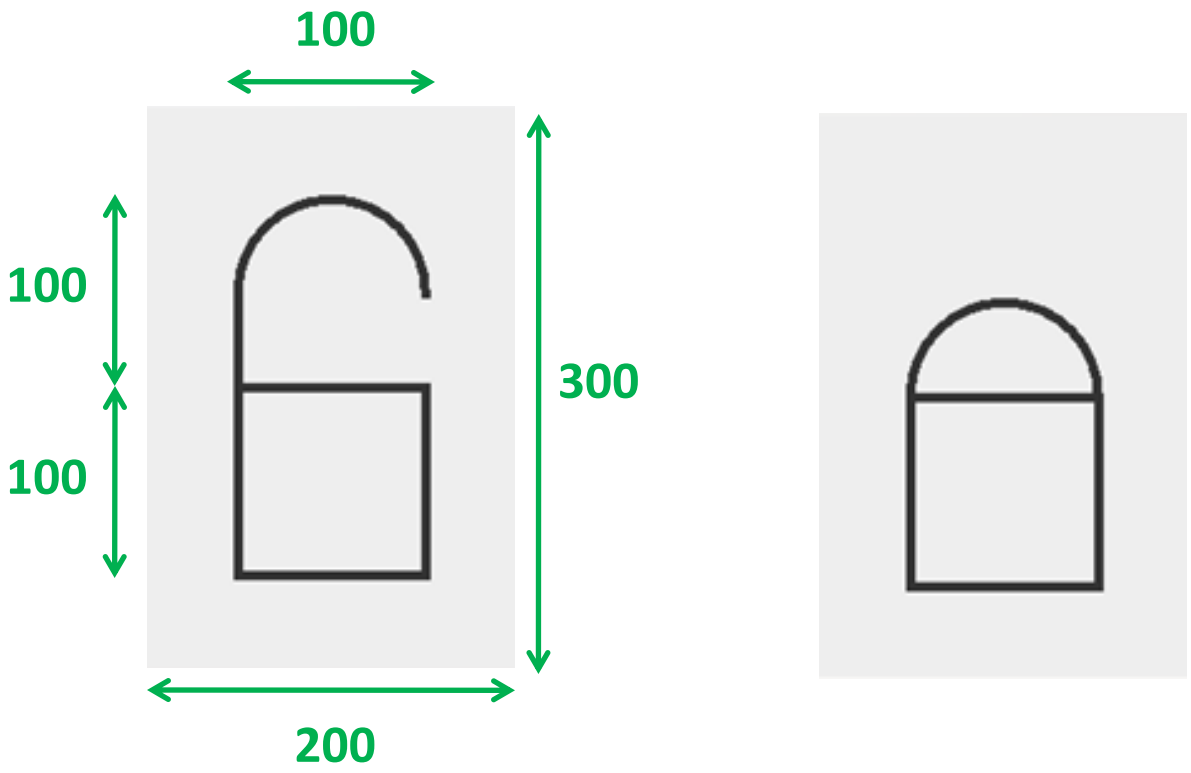
Create a view class that extends JPanel.
Add the appropriate data members and write the constructor.

- How does the view get data from the model when repainting?
- Do not implement paintComponent yet.

LockView 2D Drawing Exercise

Write the **`paintComponent(Graphics g)`** method to draw the lock opened or closed, depending on the state of the model. Use the following shapes:

- Line2D.Double(xA, yA, xB, yB)
- Rectangle2D.Double(xCorner, yCorner, height, width)
- Arc2D.Double(xCorner, yCorner, height, width, startAngle, endAngle, 0)



LockController (1)

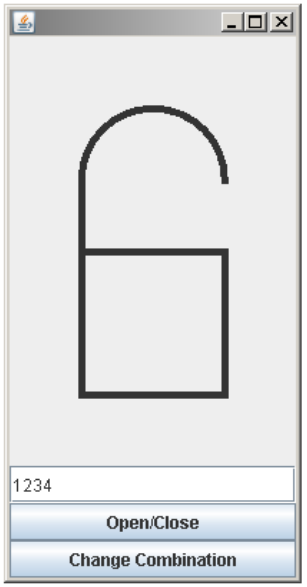
Create a LockController class. Add the data members and the main() method, which should only create an instance of the controller and display it.

- We'll add the constructor later.

LockController (2)

Write the constructor so that the frame looks as follows.

- We'll add the code to handle events later.



LockController (3)

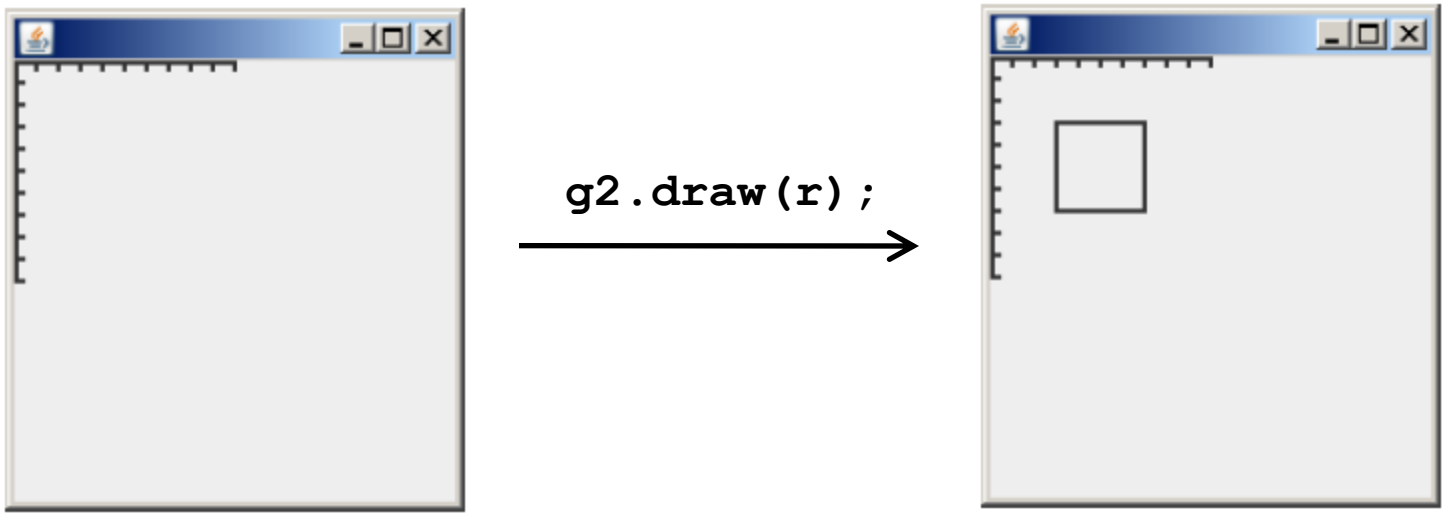
Complete your program by attaching an anonymous inner class to each button.
When a button is pressed, you need to:
- retrieve the user input
- update the model
- tell the view to repaint

Affine Transformations

Shapes are drawn using reference axes
Applying a 2D transformation is equivalent to transforming **those axes**.
The shapes drawn afterwards are drawn in the transformed reference axes.

The default axes are in the top left hand corner, y pointing down.

```
Rectangle2D r = new Rectangle2D.Double(30,30,40,40);
```



Default axes

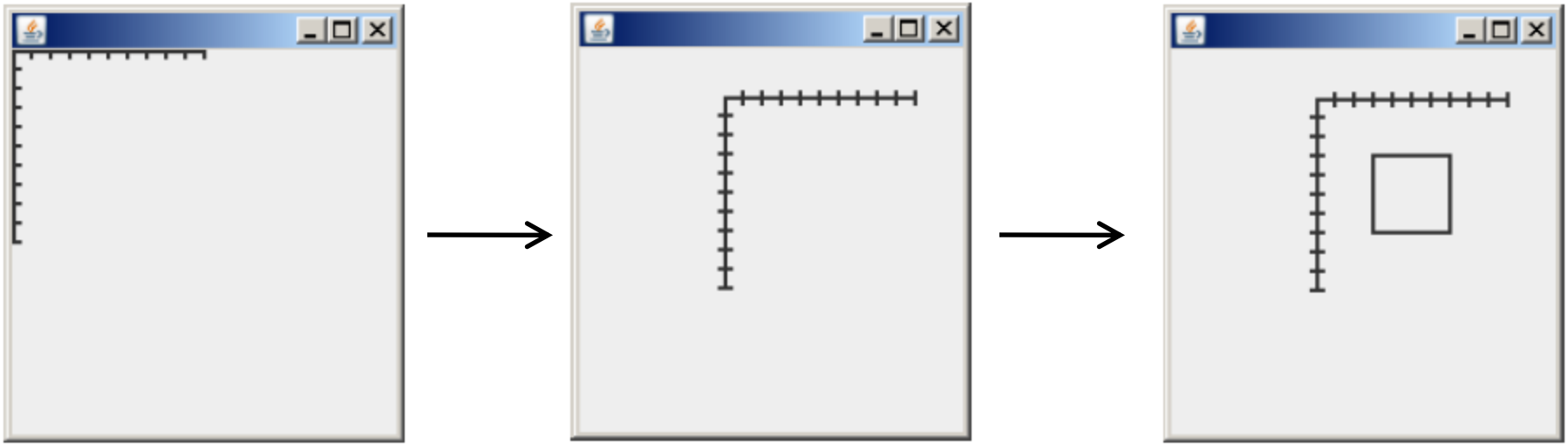
Affine Transformations

```
Rectangle2D r = new Rectangle2D.Double(30,30,40,40);
```

```
AffineTransform t
= new AffineTransform();
t.translate(75,25);
g2.transform(t);
```

```
g2.draw(r);
```

Default axes

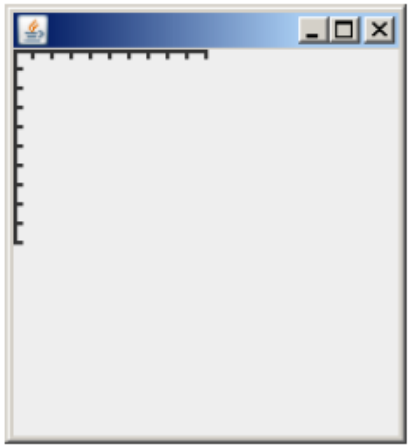


Combined Transformations

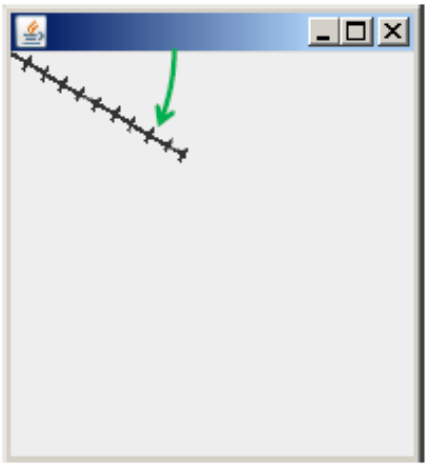
When multiple transformations are combined (a single AffineTransform object is used), the transformations are applied in reverse of the order that they were called.

How does the following AffineTransform change the axes?

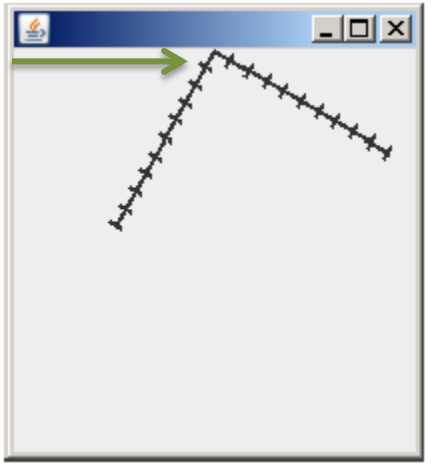
```
   AffineTransform t = new AffineTransform();
2) t.translate(100,0);
1) t.rotate(Math.toRadians(30));
   g2.transform(t);
```



Default axes



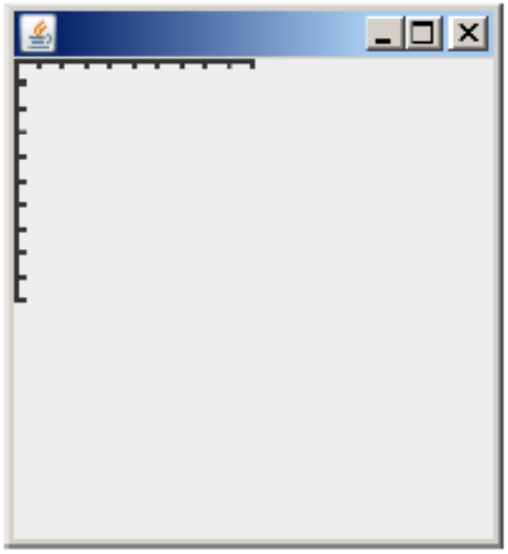
1)



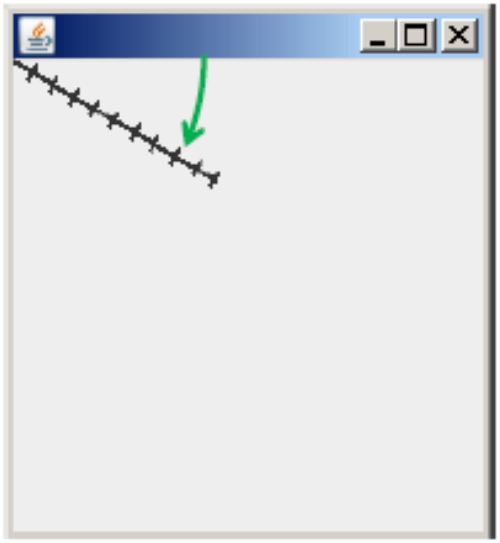
2)

Successive Transformations

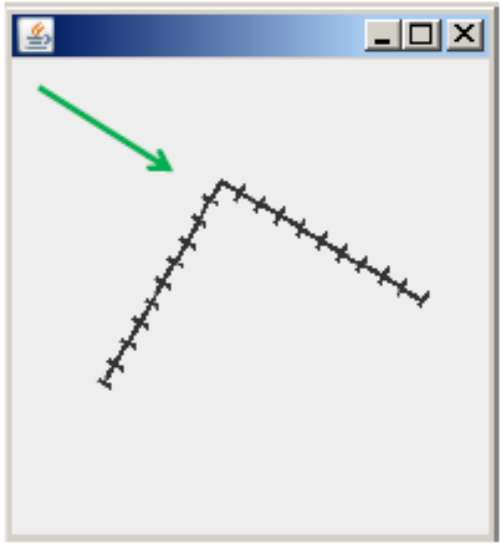
When two transformations are applied separateley (g2.transform() is called twice), the second transformation is defined in the axis system resulting from the first transformation.



Default axes

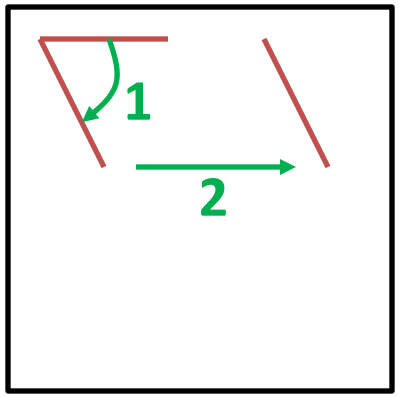


```
AffineTransform t
= new AffineTransform();
t.rotate(Math.toRadians(30));
g2.transform(t);
```

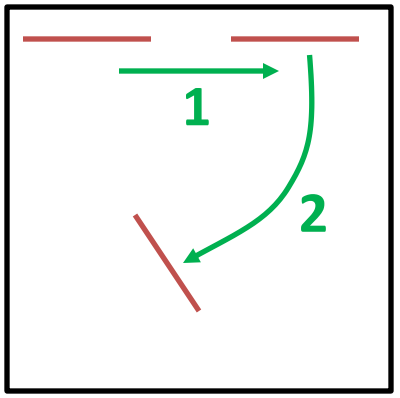


```
AffineTransform s
= new AffineTransform();
s.translate(100,0);
g2.transform(s);
```

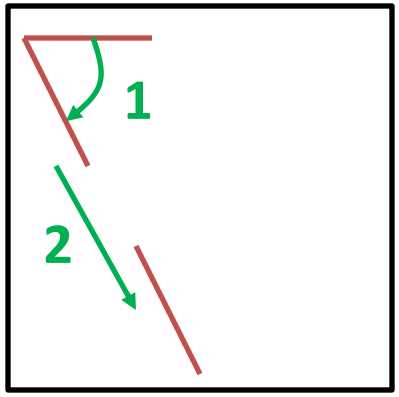
Compare



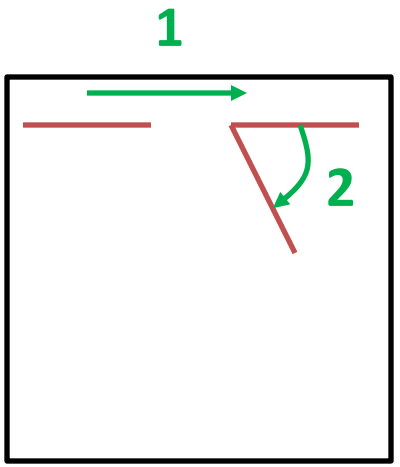

```
AffineTransform t = new AffineTransform();
t.translate(100, 0);
t.rotate(Math.toRadians(60));
g2.transform(t);
```




```
AffineTransform t = new AffineTransform();
t.rotate(Math.toRadians(60));
t.translate(100, 0);
g2.transform(t);
```




```
AffineTransform t = new AffineTransform();
t.rotate(Math.toRadians(60));
g2.transform(t);
t = new AffineTransform();
t.translate(100, 0);
g2.transform(t);
```




```
AffineTransform t = new AffineTransform();
t.translate(100, 0);
g2.transform(t);
t = new AffineTransform();
t.rotate(Math.toRadians(60));
g2.transform(t);
```

Combined vs. Successive Transformations

What does the set of axes look like after the following transformations are applied?

```
AffineTransform t = new AffineTransform();
t.rotate(Math.toRadians(45));
g2.transform(t);
t = new AffineTransform();
t.scale(2,1);
g2.transform(t);
t = new AffineTransform();
t.translate(50,0);
g2.transform(t);
```

Remember, Quiz 2 Review

Date: Wed. April 11

Time: 7pm – 9pm

1.00 / 1.001 / 1.002 Introduction to Computers and Engineering Problem Solving

Spring 2012