

Lab Class Visualization

- Part 1: Assignments in InfoVis: *10 Pt.*
(Patrick Riehmann)
- Part 2: Assignments in SciVis: *10 Pt.*
(Carl Matthes)
- Part 3: Final Project: *30 Pt.*

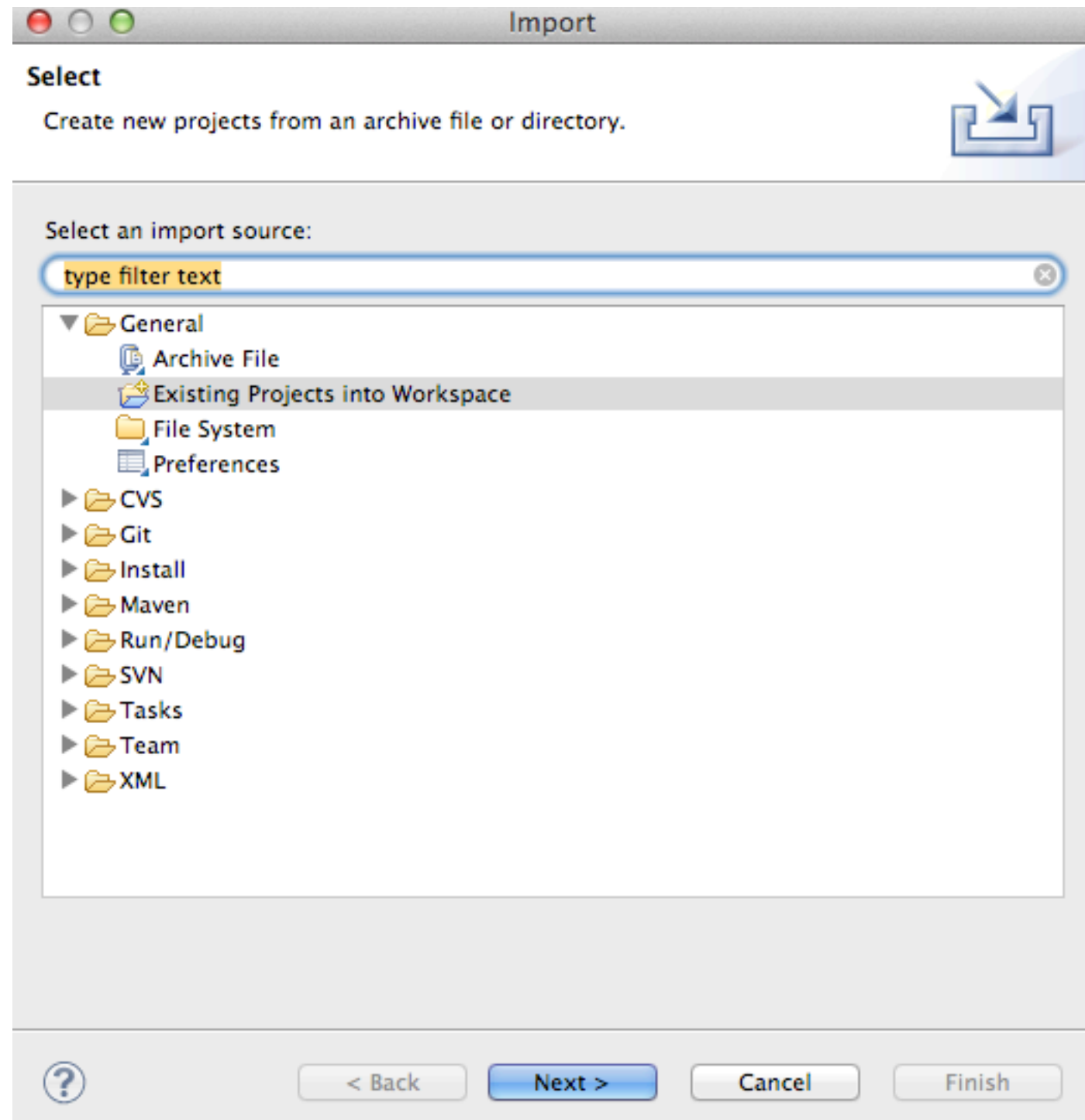
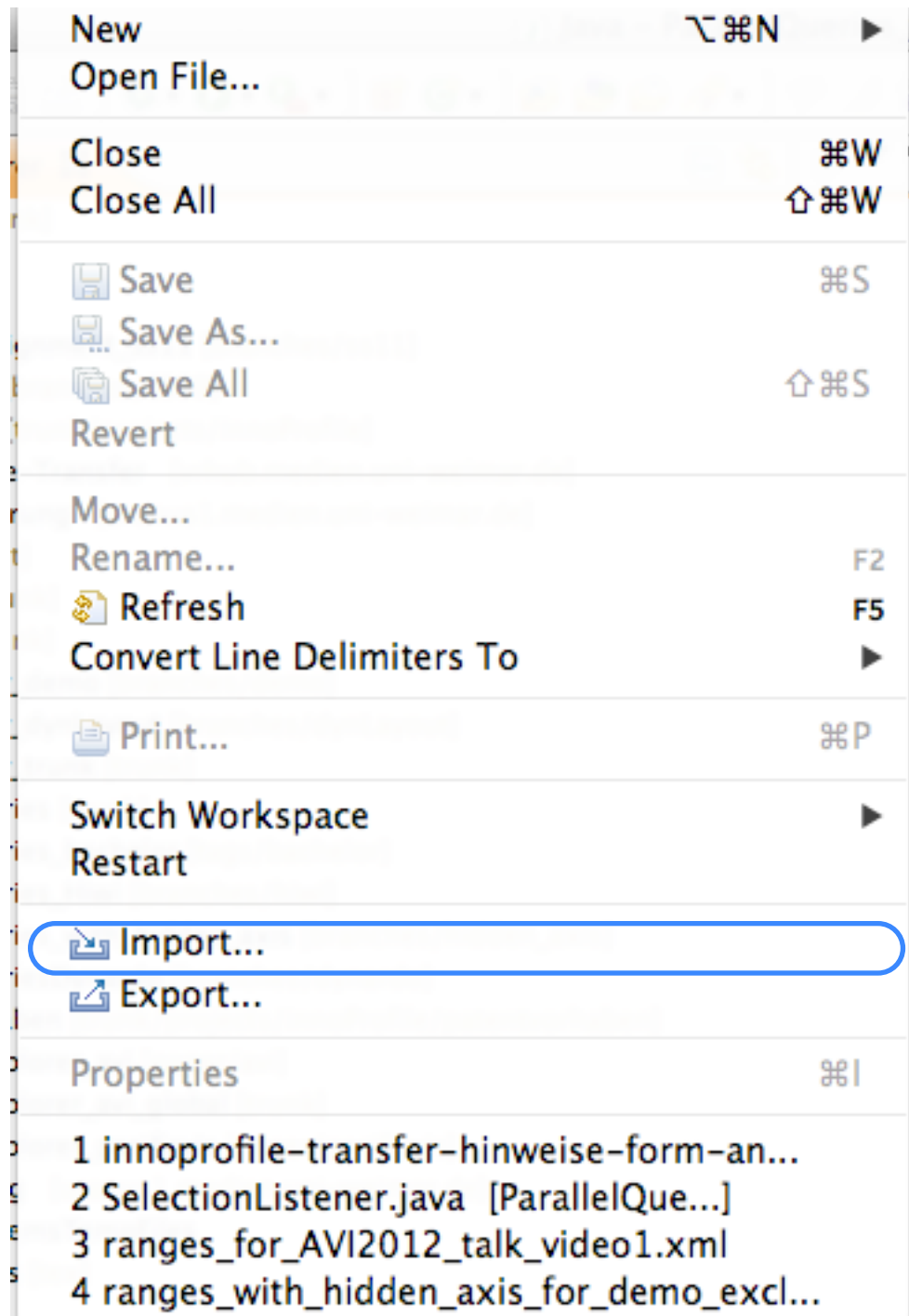
Final Project Vis/InfoVis

- Topic: Up to you (Either InfoVis or SciVis)
- Expenditure of time: ~40h/Student
- Requirements:
 - ▶ Autonomous implementation
(Groups of max. two students will be accepted)
 - ▶ Unique and fresh kind of visualization
 - ▶ At least two complex interaction techniques
- Due to **September, ____th 2019 (12 pm CET)**
 - ▶ Send your code to
 - patrick.riehmann[at]uni-weimar.de (InfoVis Topics)
 - carl-feofan.matthes[at]uni-weimar.de (SciVis Topics)
- Project presentation on **September, ____th 2019 (10 am CET)**

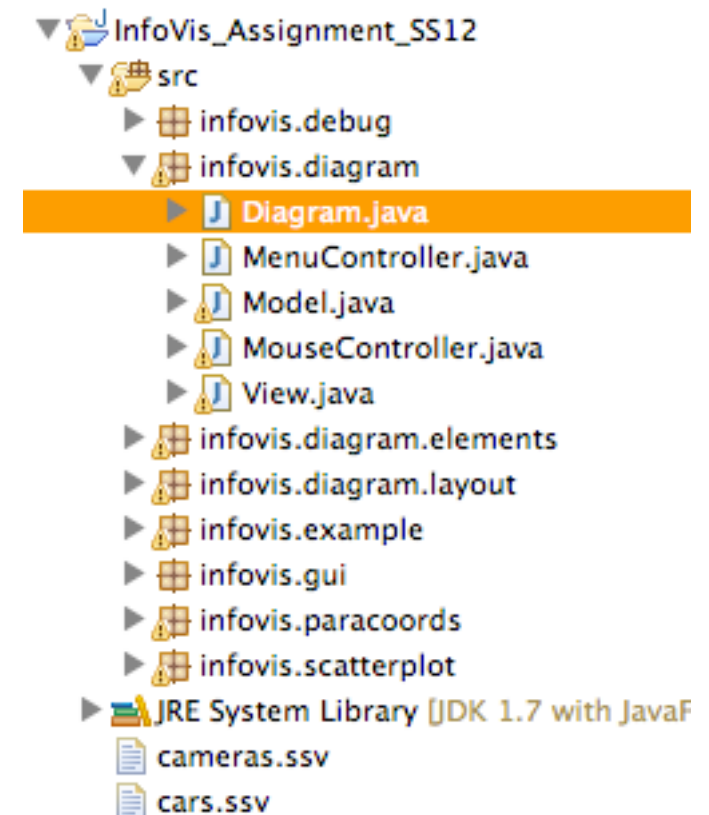
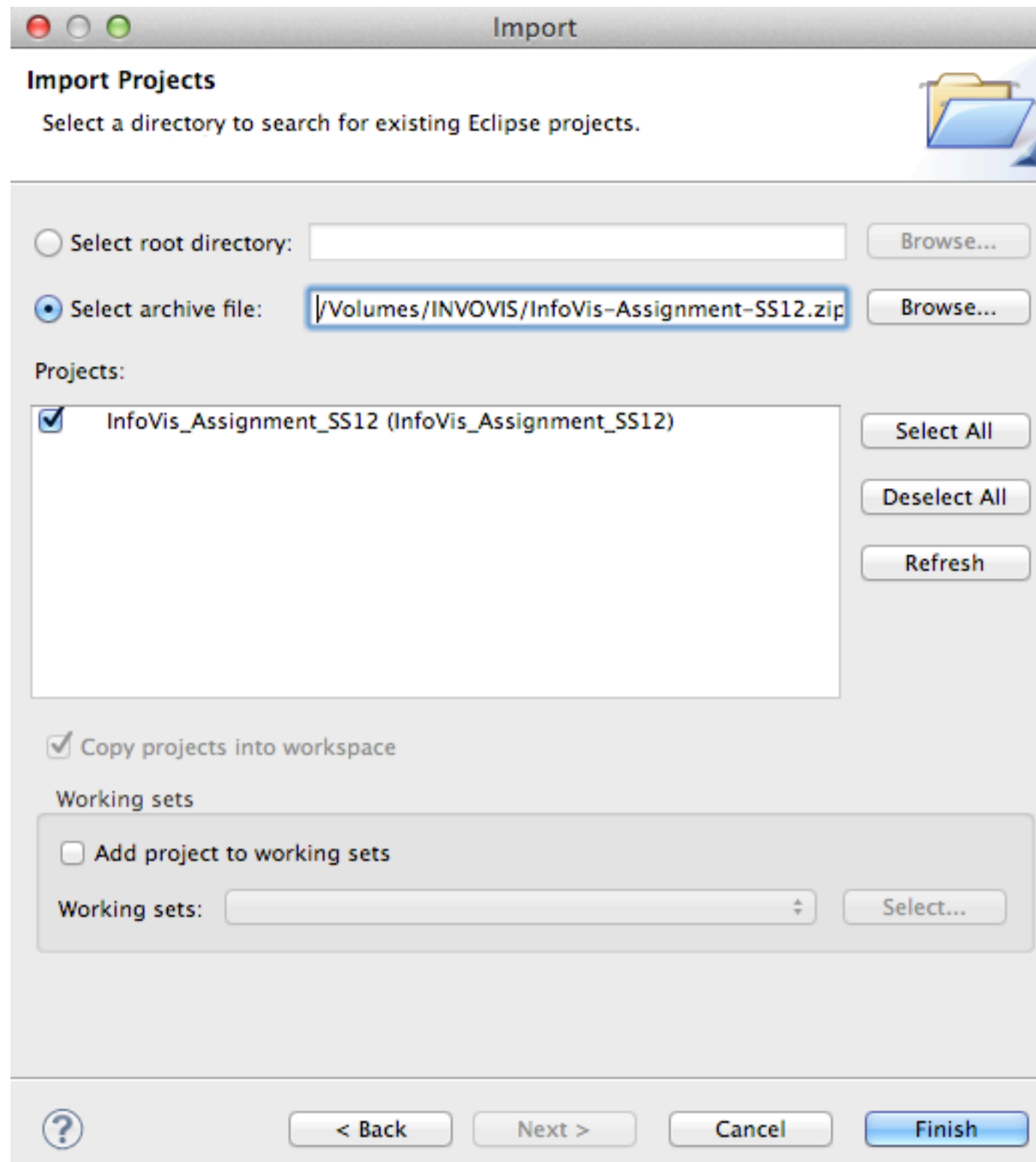
InfoVis-Assignments

- 4 Assignments with overall 10 points
- Requirements
 - ▶ JDK 6
 - ▶ Eclipse
- Due to **May, 14th (Early Birds) and May, 21th 2019**

Import Project



Import Project

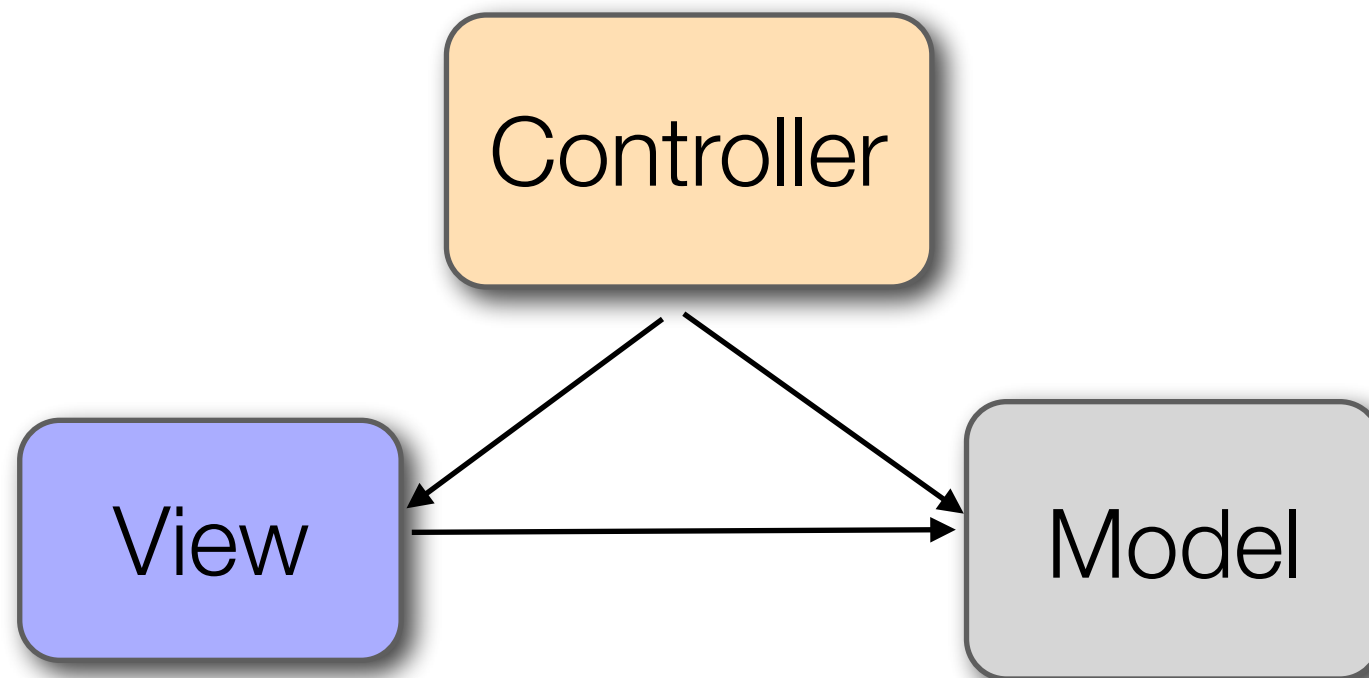


Drawing 2D in Java

- Java2D / Swing
 - ▶ Backend:
 - OpenGL (OSX, Linux - Treiber!!)
 - “-Dsun.java2d.opengl=True”
 - DirectX (Windows, generally enabled since 1.6)
 - ▶ No canvas class in Swing
 - Create a subclass of *JPanel* class
 - Override *paint(Graphics g)*

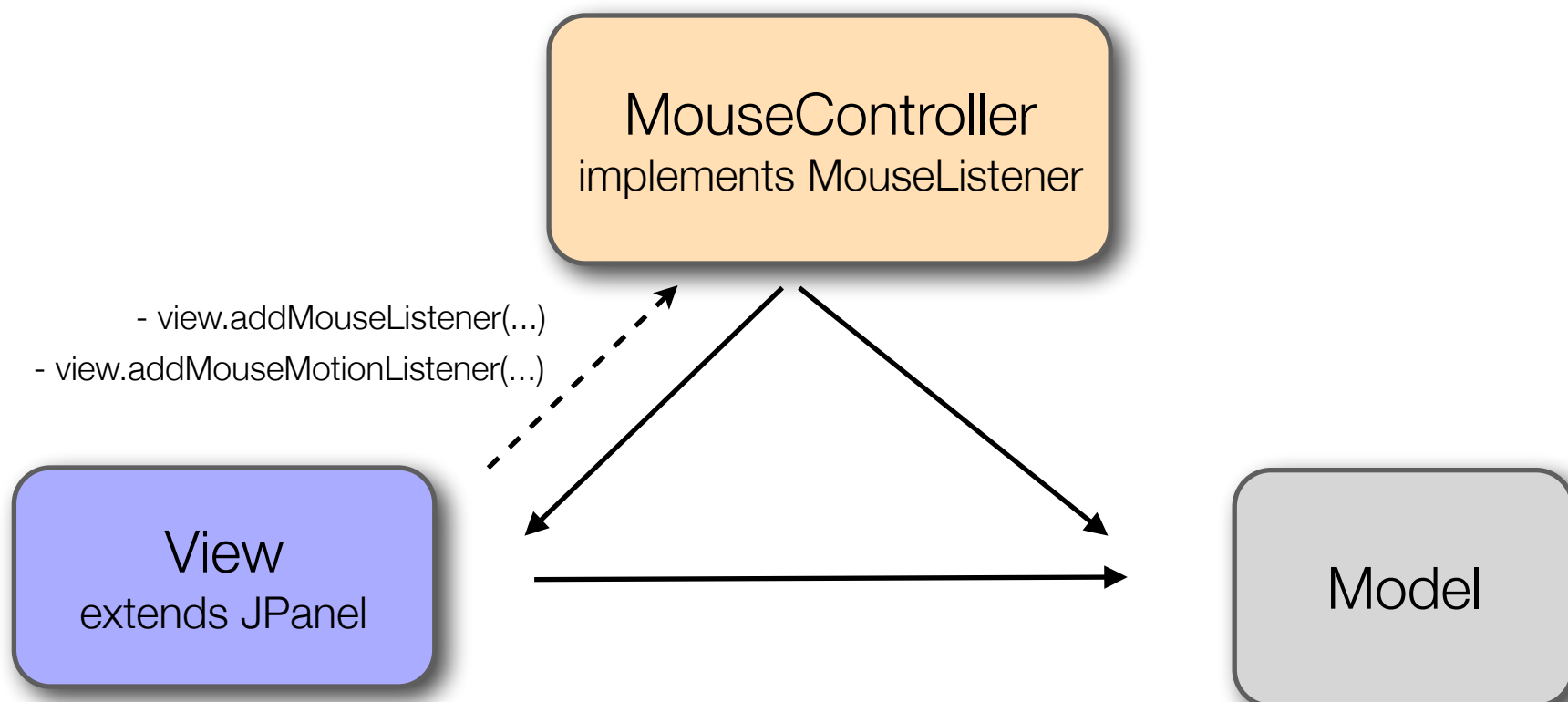
Example

- Model-View-Controller



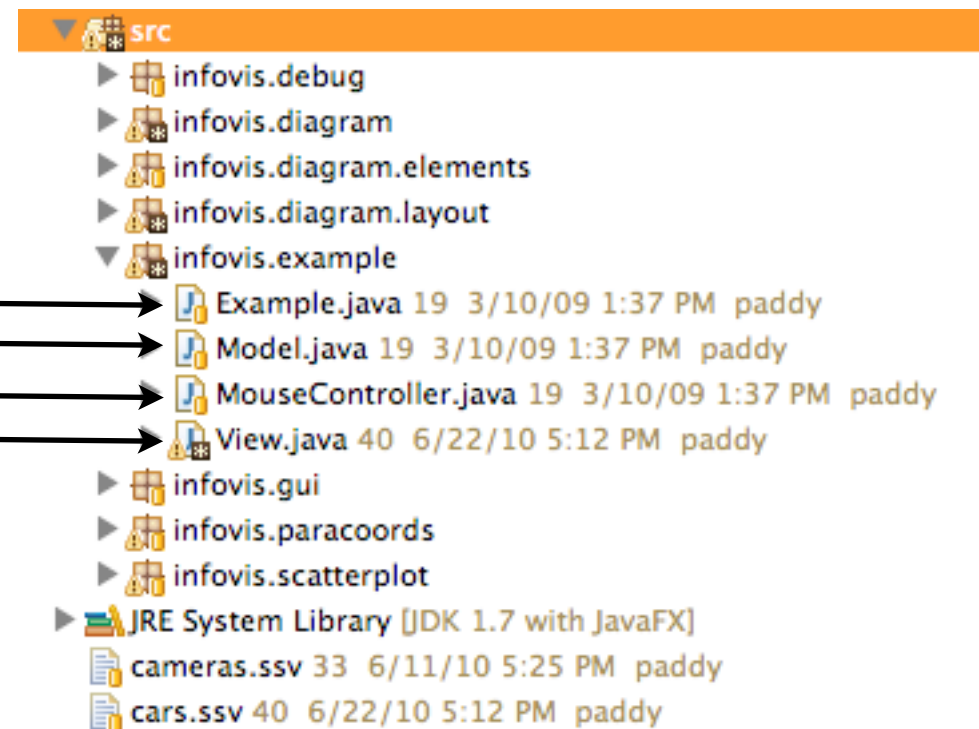
Add Listener to View

- Model-View-Controller



Example Package

Start class of example application
Model (Stub)
Implement Listeners
Override *paint()* method



Drawing Example

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

    public void paint(Graphics g){
        Graphics2D g2D = (Graphics2D) g; // cast explicitly
        for(Iterator i = model.iterator();i.hasNext();){
            ...
        }
    }
}
```

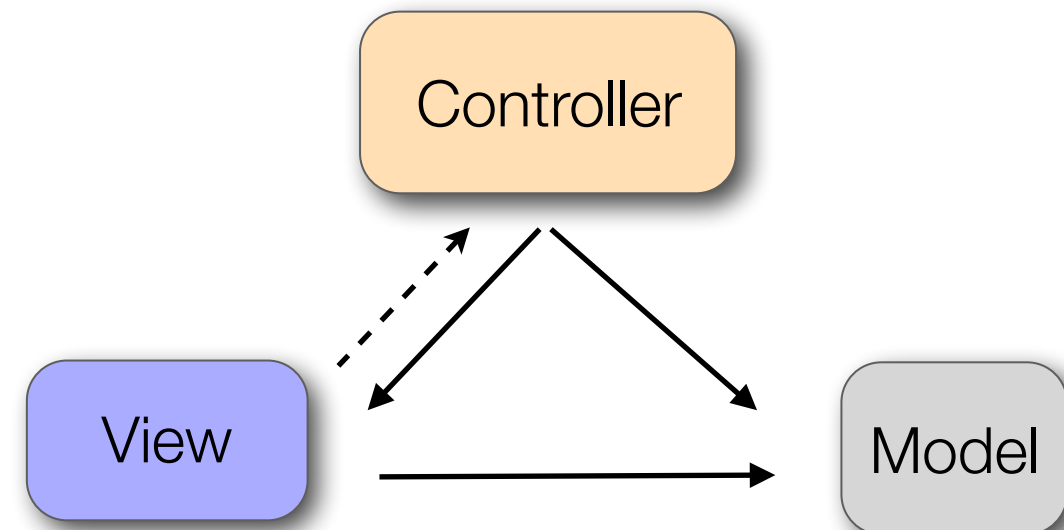
```
public class MouseController implements MouseListener, MouseMotionListener {
    private View view;
    private Model model;

    public void mouseClicked(MouseEvent e) {
    }

    public void mousePressed(MouseEvent e) {
        int x = e.getX();
        int y = e.getY();
        ...
    }

    public void mouseReleased(MouseEvent e) {
    }

    public void mouseDragged(MouseEvent e) {
    }
}
```



Drawing Example

- Graphics Context

- ▶ Graphics2D extends Graphics

- g2D.setStroke(...);
 - g2D.setColor(Color);
 - g2D.draw(shape);
 - g2D.fill(shape);
 - g2D.translate(...);
 - g2D.rotate(...);
 - g2D.scale(...);

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

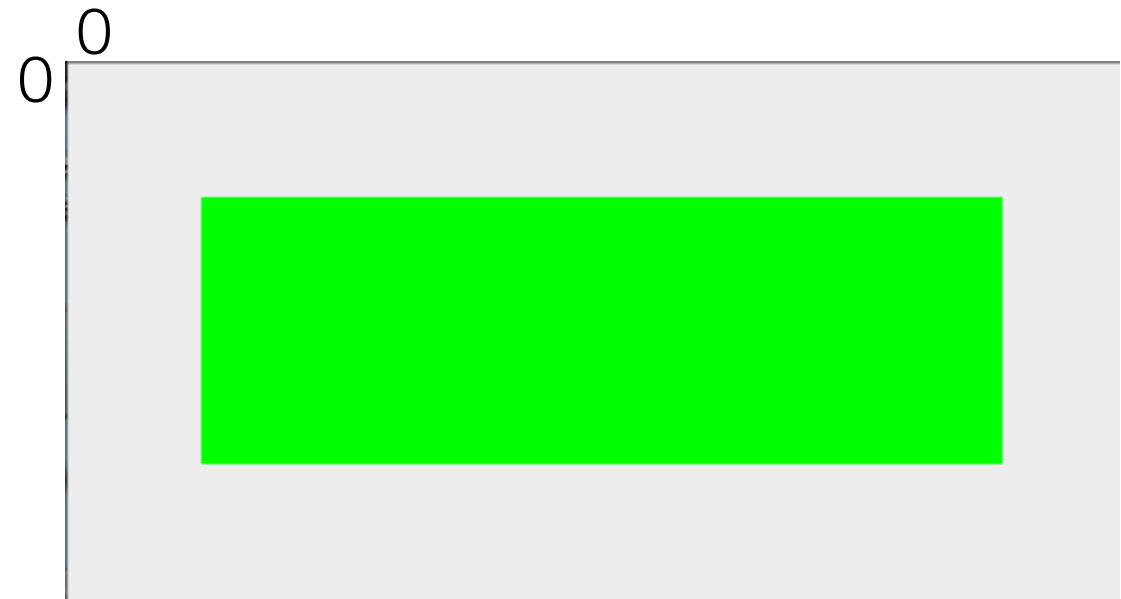
    public void paint(Graphics g){
        Graphics2D g2D = (Graphics2D) g; // cast explicitly
        for(Iterator i = model.iterator();i.hasNext();){
            ...
        }
    }
}
```

```
...
Rectangle2D rect = new Rectangle2D.Double(1,1,300,100);
g2D.setColor(Color.RED);
g2D.fill(rect);
g2D.setColor(Color.BLACK);
g2D.draw(rect);
...
```



Z-Order Example

```
//Back To Front  
  
...  
Rectangle2D rect =  
    new Rectangle2D.Double(150,150,300,100);  
g2D.setColor(Color.RED);  
g2D.fill(rect);  
  
g2D.setColor(Color.BLUE);  
g2D.fill(rect);  
  
g2D.setColor(Color.GREEN);  
g2D.fill(rect);  
...
```

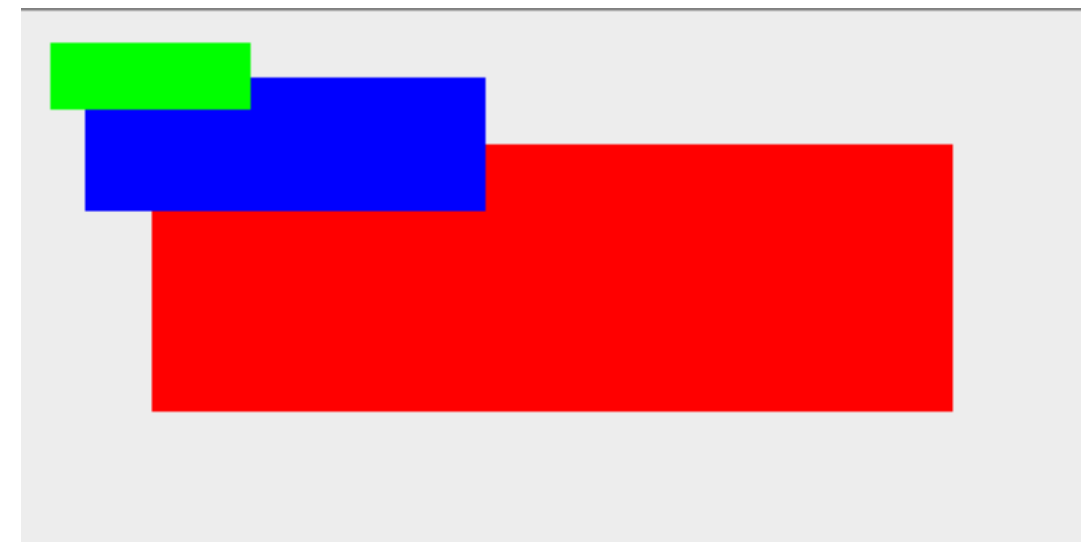


Scale

```
...
Rectangle2D rect =
    new Rectangle2D.Double(50,50,300,100);
g2D.setColor(Color.RED);
g2D.fill(rect);
g2D.scale(0.5, 0.5);
g2D.setColor(Color.GREEN);
g2D.fill(rect);
...
```

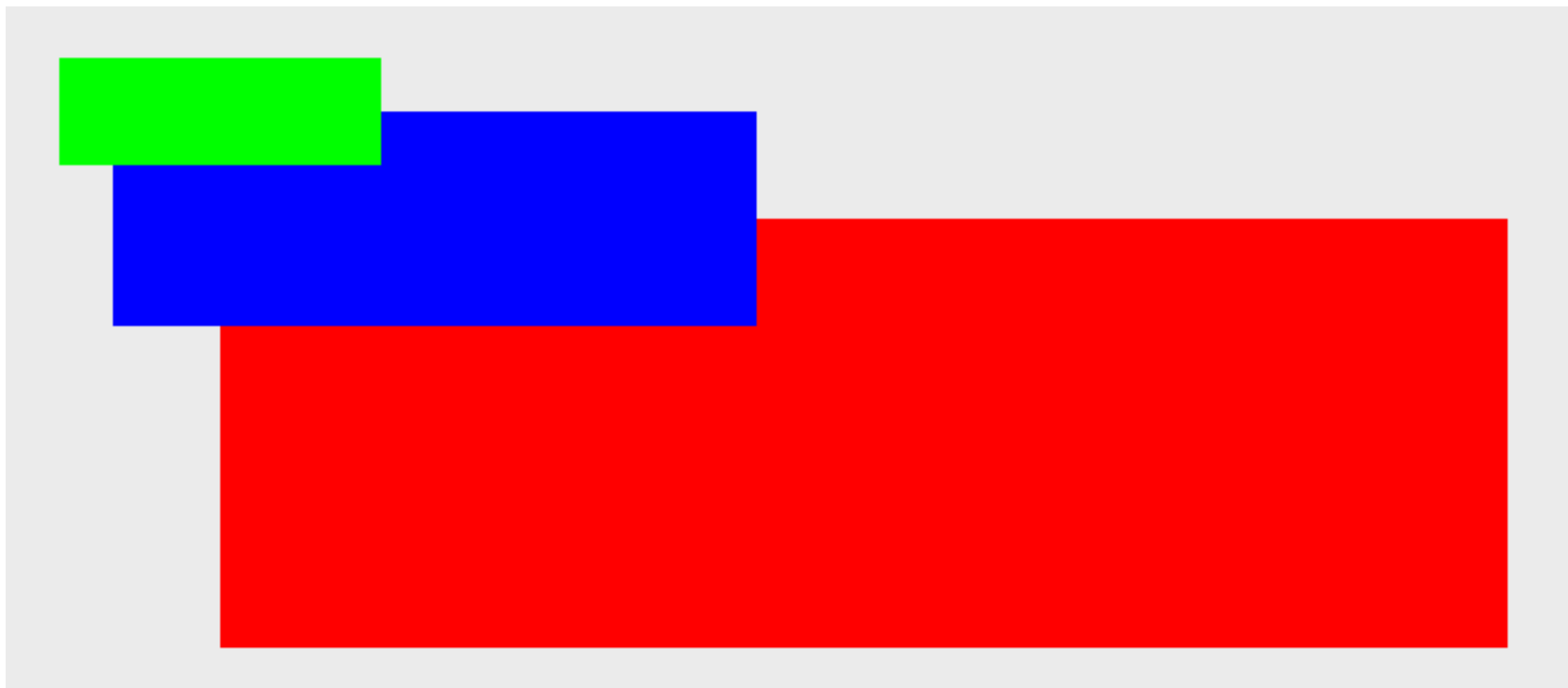


```
...
Rectangle2D rect =
    new Rectangle2D.Double(50,50,300,100);
g2D.setColor(Color.RED);
g2D.fill(rect);
g2D.scale(0.5, 0.5);
g2D.setColor(Color.BLUE);
g2D.fill(rect);
g2D.scale(0.5, 0.5);
g2D.setColor(Color.GREEN);
g2D.fill(rect);
...
```



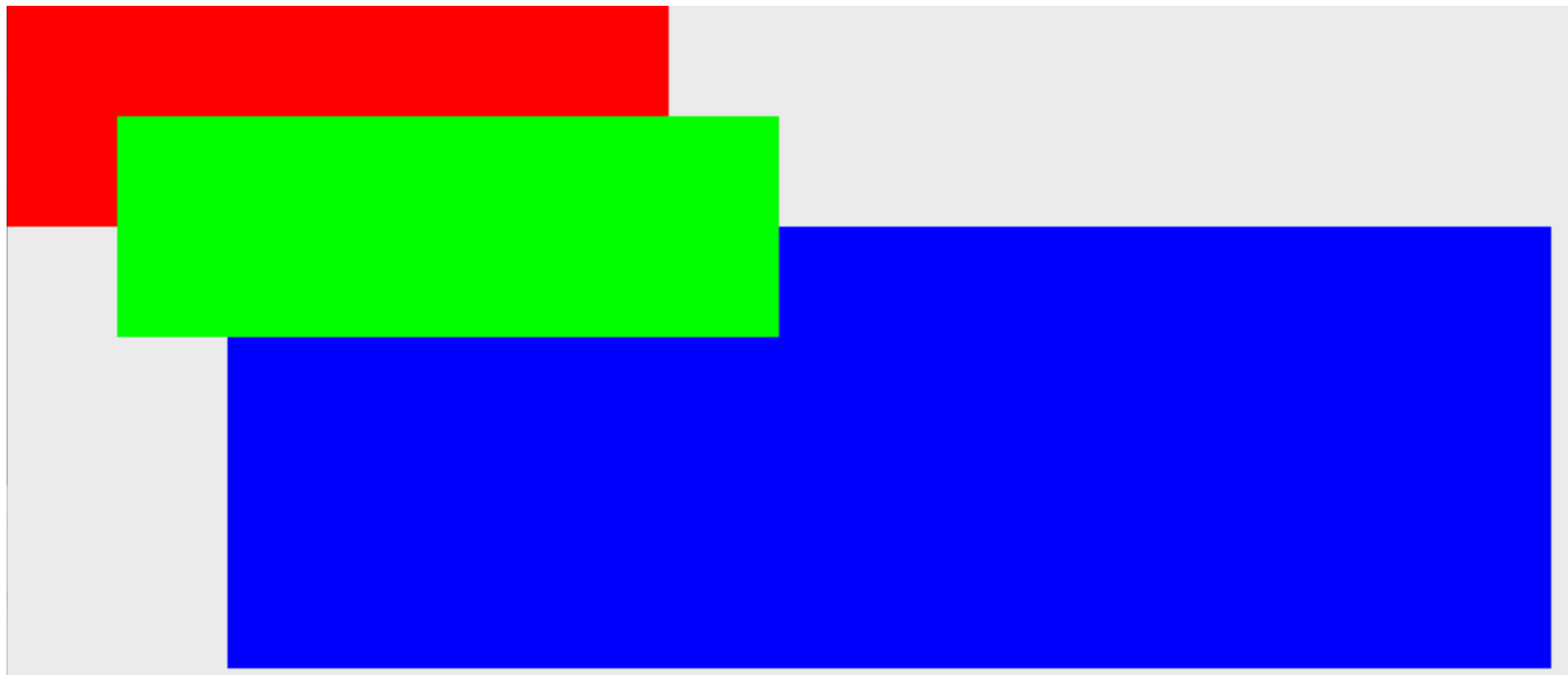
Scale

```
Rectangle2D rect =  
    new Rectangle2D.Double(50,50,300,100);  
  
g2D.scale(2, 2);  
g2D.setColor(Color.RED);  
g2D.fill(rect);  
g2D.scale(0.5, 0.5);  
g2D.setColor(Color.BLUE);  
g2D.fill(rect);  
g2D.scale(0.5, 0.5);  
g2D.setColor(Color.GREEN);  
g2D.fill(rect);
```



Scale and Transform

```
Rectangle2D rect =  
    new Rectangle2D.Double(50,50,300,100);  
  
g2D.translate(-50, -50);  
g2D.setColor(Color.RED);  
g2D.fill(rect);  
g2D.scale(2, 2);  
g2D.translate(25,25);  
g2D.setColor(Color.BLUE);  
g2D.fill(rect);  
g2D.scale(0.5, 0.5);  
g2D.setColor(Color.GREEN);  
g2D.fill(rect);
```



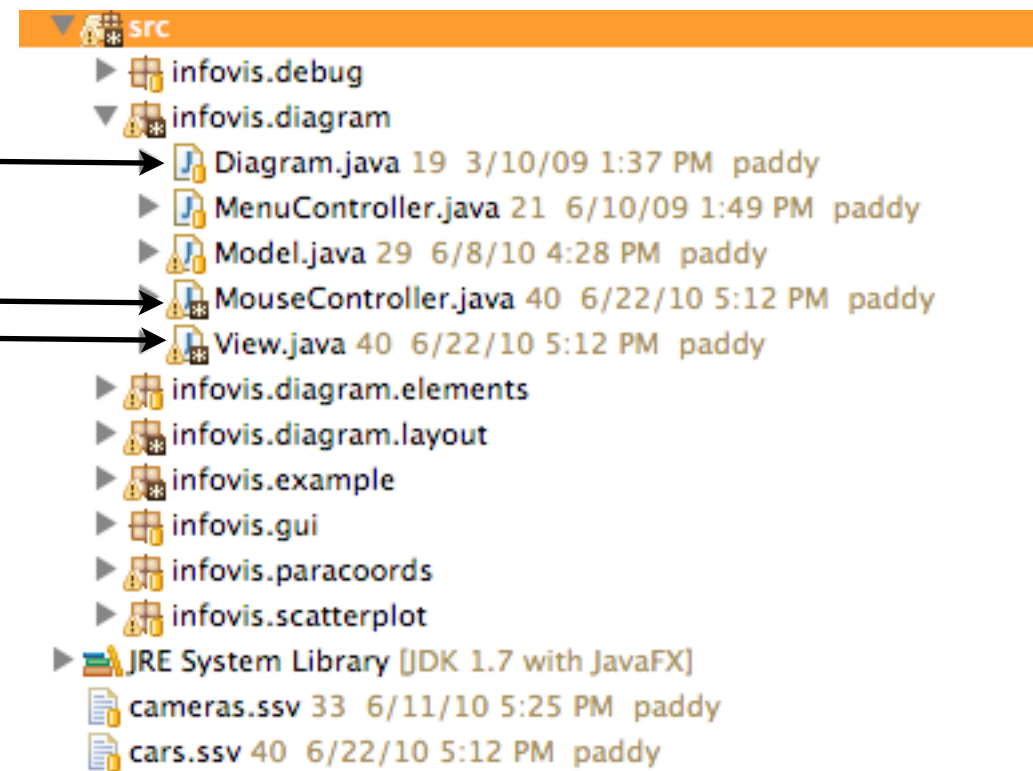
Assignment 1 - Detail And Overview

1. Extend the *paint()* method of *infovis.diagram.View* to draw an overview frame on top of the diagram that show a smaller version of the diagram. Use the *scale* member to zoom in and out within the main view. The overview frame remains with its size and shows always the entire diagram.
[1 Point]
2. Implement a marker rectangle, which highlights the current viewable area of the main view within the overview frame. Extend the *infovis.diagram.MouseController* class for navigating the viewable area of the main view by moving the marker rectangle. Use the *translateX* and *translateY* member variables.
[[1 Point]
3. Create a overview window that is arbitrarily placeable.
[optional, 1 Point]

Start class of diagram application

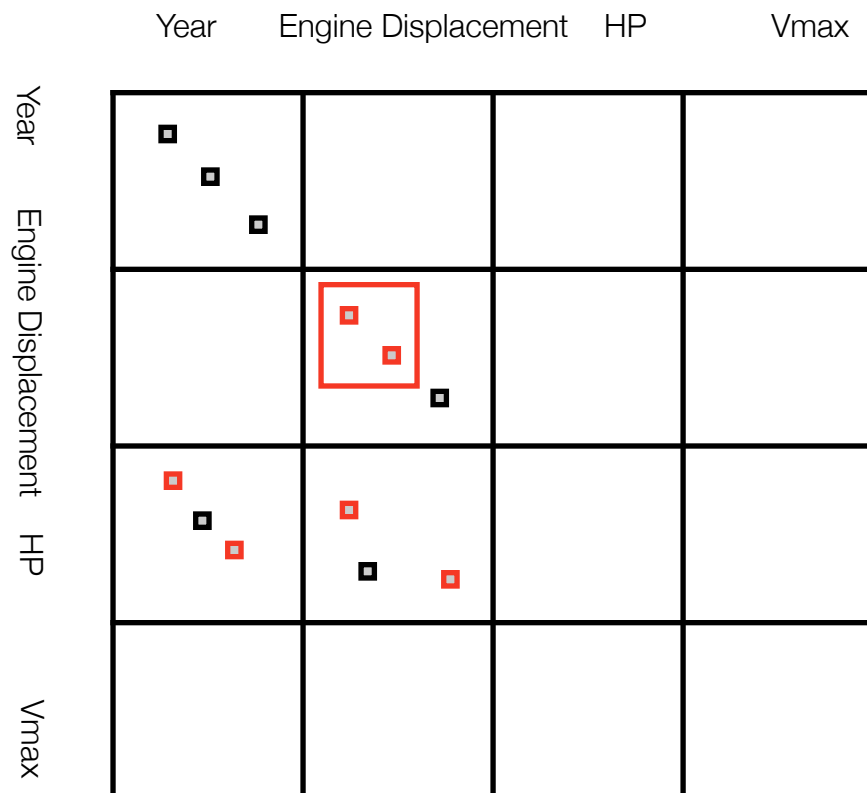
Implement Listeners

Override *paint()* method



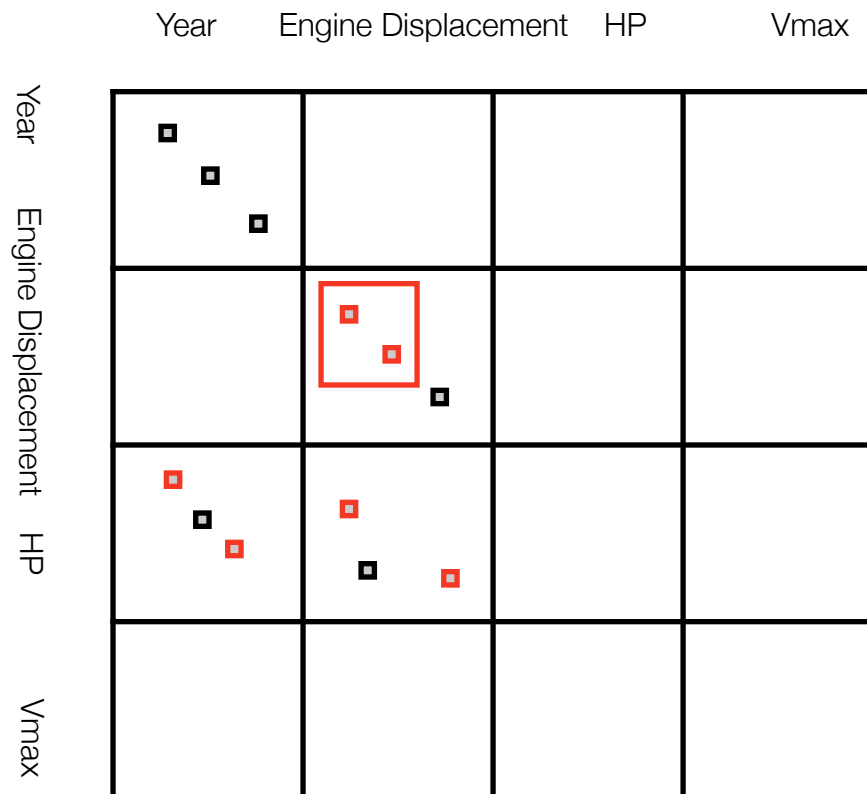
Assignment 2 - Brushing And Linking

1. Realize a scatterplot matrix for depicting multi-dimensional data. Use the `infovis.scatterplot`. Data class, which contains a multidimensional record. Override the `paint()` -method of the `infovis.scatterplot.View` class.
[1 Point]
2. Implement the “Brushing And Linking”- technique, to mark points in a single scatter plot for being highlighted in all plots. Override the methods of the class `infovis.scatterplot.MouseController`.
[1 Point]



```
public class Model{  
  
    private ArrayList<Data> data= new ArrayList<Data>();  
    private ArrayList<Range> ranges = new ArrayList<Range>(); //dim  
    private ArrayList<String> labels = new ArrayList<String>(); //dim  
    private int dim = 4; //dim=7  
  
    public Iterator<Data> iteratorDate(){  
        return list.iterator();  
    }  
    public ArrayList<Range> iteratorRanges() {  
        return ranges.iterator();  
    }  
    public Iterator<String> iteratorLabels() {  
        return labels.iterator();  
    }  
}
```

Assignment 2 - Brushing And Linking



```
public class Model{

    private ArrayList<Data> data= new ArrayList<Data>();
    private ArrayList<Range> ranges = new ArrayList<Range>(); //dim
    private ArrayList<String> labels = new ArrayList<String>();//dim
    private int dim = 4; //dim = 7

    public Iterator<Data> iteratorDate(){
        return list.iterator();
    }
    public ArrayList<Range> iteratorRanges() {
        return ranges.iterator();
    }
    public Iterator<String> iteratorLabels() {
        return labels.iterator();
    }
}
```

```
public class Data{

    private double [] values; //dim
    private Color color;
    private String label;

    public int getLength(){
        return values.length;
    }
    public double getValue(int i){
        return values[i];
    }
}
```

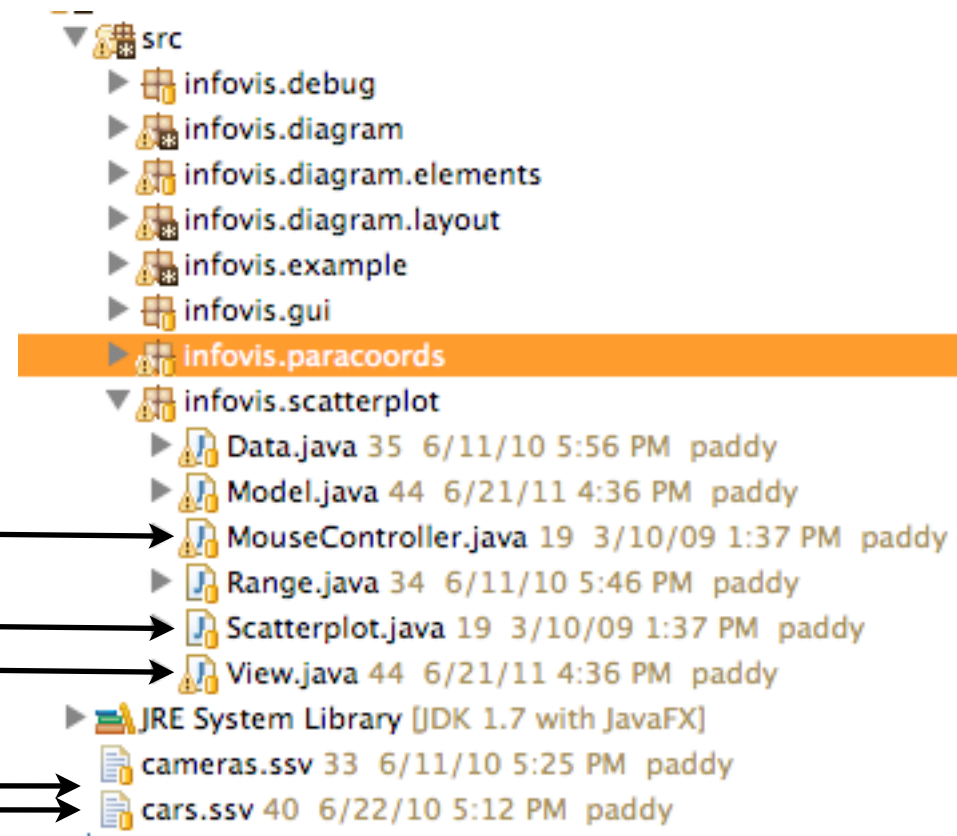
```
public class Range{

    private double min;
    private double max;

    ...
}
```

Remarks Assignment 2

Implement Listeners
Start class of scatterplot application
Override *paint()* method
Test the plot with both datasets
(see *Model.importValues()*)



Assignment 3- Parallel Coordinates

1. Use the data set if assignment 2 for drawing a parallel coordinates display. Override the *paint()* -method of the *infovis.paracoords.View* class.
[2 Points]
2. Implement a marking technique for highlighting paths. Override methods of the class *infovis.paracoords.MouseController*.
[1 Point]
3. Create axes that are moveable in horizontal direction.
[optional, 1 Point]

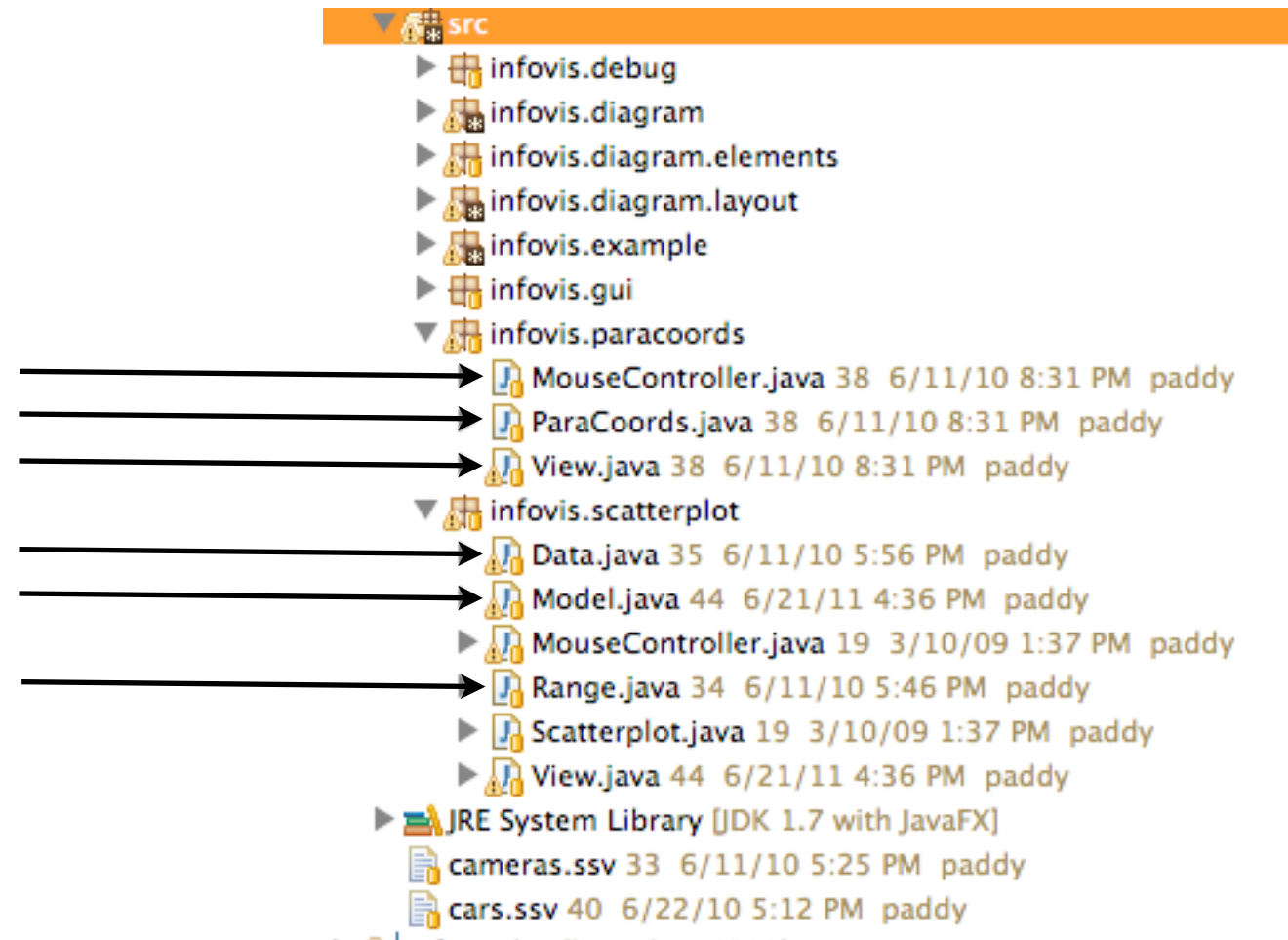
Remarks Assignment 3

Implement listeners

Start class of example application

Override *paint()* method

Use classes from previous assignment



Assignment 4 - Focus And Context

1. Override the method *transform()* of the class *infovis.diagramm.layout.Fisheye* to provide the geometry transformation for a “Graphical Fisheye View of Graphs”. Please read in preparation the Paper of Sarkar and Brown (till functions F1 und F2). Consider the preservation of the width height ratio of nodes.

[2 Points]

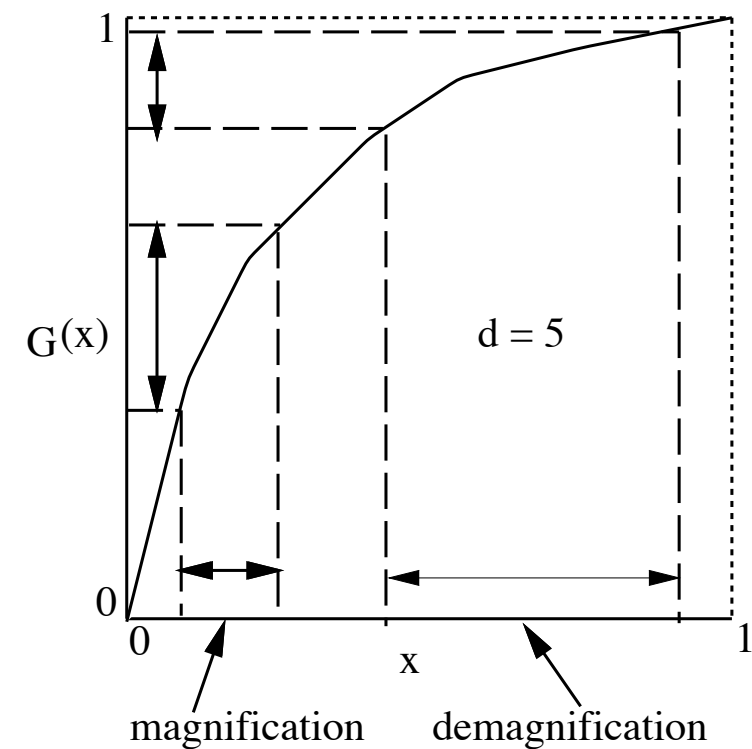
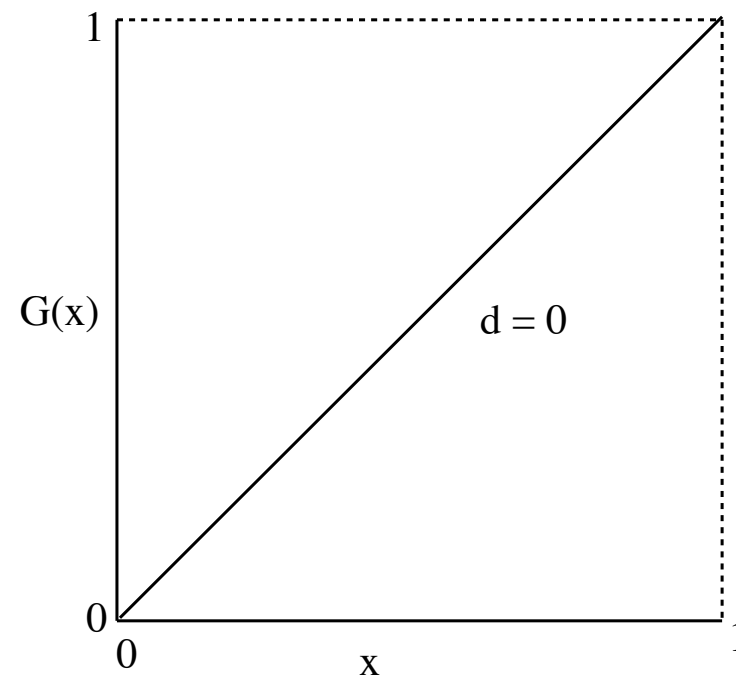
- Graphical Fisheye Views of Graphs (1992):
[\(accessible via Google Search\)](#)
- Graphical Fisheye Views (1993):
[\(accessible via Google Search\)](#)

2. Extend the technique for using the current mouse coordinates as focus point while moving the pointer.

[1 Point]

Remarks Assignment 4 - F1

$$G(x) = \frac{(d+1)x}{dx+1}$$



$$P_{fish_x} = P_{focus_x} \pm G\left(\frac{D_{norm_x}}{D_{max_x}}\right) D_{max_x}$$

$$P_{fish_y} = P_{focus_y} \pm G\left(\frac{D_{norm_y}}{D_{max_y}}\right) D_{max_y}$$

$$D_{max_x} = \begin{cases} P_{boundary_x} - P_{focus_x}, & P_{norm_x} > P_{focus_x} \\ 0 - P_{focus_x}, & P_{norm_x} < P_{focus_x} \end{cases}$$

$$D_{norm_x} = P_{norm_x} - P_{focus_x}$$

Remarks Assignment 4 - F2

$$G(x) = \frac{(d+1)x}{dx+1}$$

$$P_{fish_x} = P_{focus_x} \pm G\left(\frac{D_{norm_x}}{D_{max_x}}\right) D_{max_x}$$

$$P_{fish_y} = P_{focus_y} \pm G\left(\frac{D_{norm_y}}{D_{max_y}}\right) D_{max_y}$$

$$D_{max_x} = \begin{cases} P_{boundary_x} - P_{focus_x}, & P_{norm_x} > P_{focus_x} \\ 0 - P_{focus_x}, & P_{norm_x} < P_{focus_x} \end{cases}$$

$$D_{norm_x} = P_{norm_x} - P_{focus_x}$$

$$Q_{norm} = P_{norm} \pm S_{norm}/2$$

$$Q_{fish} = F1(Q_{norm})$$

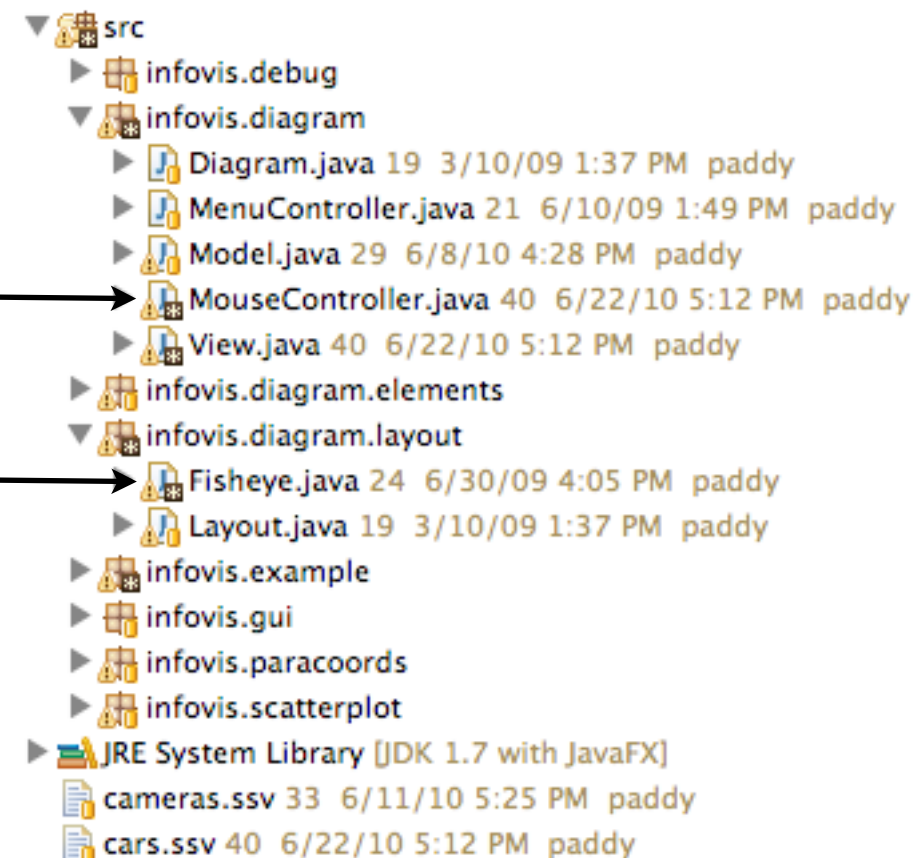
$$S_{geom} = 2\min(|Q_{fish_x} - P_{fish_x}|, |Q_{fish_y} - P_{fish_y}|) \quad ! \text{ Breite / Höhe beachten !}$$

$$S_{fish} = S_{geom} \quad \text{keine API}$$

Remarks Assignment 4

Change Listeners

Implement methods
transform(...)
setMouseCoords(...)



InfoVis Toolkits

- Protovis
- Prefuse
- d3
- JavaScript InfoVis Toolkit: <http://thejit.org/>
- PhiloGL: <http://senchalabs.github.com/philogl/>
- VTK
- [Nodebox](#) / [Nodebox2](#)
- ...

Data Sources

- Panama Papers <https://offshoreleaks.icij.org/pages/database>
- VAST Challenge 2016: <http://vacommunity.org/VAST+Challenge+2016>
- DATA.GOV: <http://explore.data.gov/catalog/raw/>
- API Leipzig: <http://www.apileipzig.de/wiki/show/Was-ist-die-API-LEIPZIG>
- Open Platform: <http://www.guardian.co.uk/open-platform>
- Developer Network: <http://developer.nytimes.com/>
- London Datastore: <http://data.london.gov.uk/>
- Offene Daten Berlin: <http://daten.berlin.de/>
-