Comp 401
Graph vs. Dag vs. Tree Object Structures

Instructor: Prasun Dewan

Prerequisite

- Composite Object Shapes

Cartesian Plane Alternatives

```
public interface CartesianPlane {
  public int getAxesLength();
  public void setAxesLength(int anAxesLength);
  public Line getXAxis();
  public Line getYAxis();
  public StringShape getXLabel();
  public StringShape getYLabel();
}
```

```
public interface DAGCartesianPlane {
 public LineWithObjectProperty getXAxis();
 public LineWithObjectProperty getYAxis();
 public Point getXAxisLocation();
 public Point getYAxisLocation();
 public int getAxesLength();
 public void setAxesLength(int anAxesLength);
 public StringShape getXLabel();
 public StringShape getYLabel();
}
```

DAG Cartesian Plane

```
public class ADAGCartesianPlane implements DAGCartesianPlane {
  int originX, originY;
  int axesLength;
  LineWithObjectProperty xAxis, yAxis;
  Point xAxisLocation, yAxisLocation;
  StringShape xLabel;
  StringShape yLabel;
  public ADAGCartesianPlane(int theAxesLength,
                            int theOriginX, int theOriginY) {
    axesLength = theAxesLength;
    originX = theOriginX;
    originY = theOriginY;
    xAxisLocation = toXAxisLocation();
    yAxisLocation = toYAxisLocation();
    xAxis = new ALineWithObjectProperty
                (xAxisLocation, theAxesLength, 0);
    yAxis = new ALineWithObjectProperty
                (yAxisLocation, 0, theAxesLength);
    xLabel = new AStringShape("X", toXLabelX(), toXLabelY());
    yLabel = new AStringShape("Y", toYLabelX(), toYLabelY());
}
```

Address copied, not object

DAG Cartesian Plane

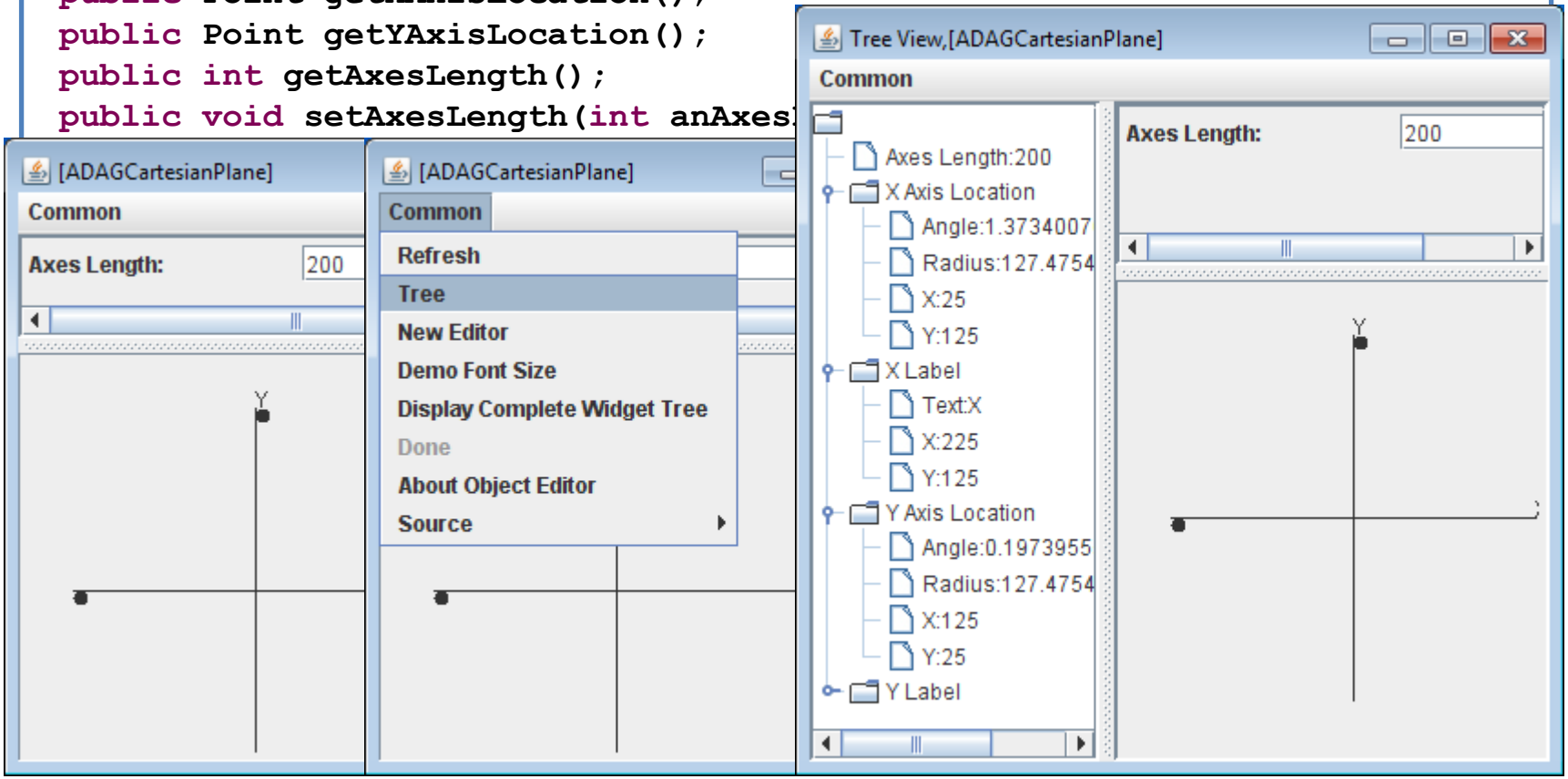
```
public LineWithObjectProperty getXAxis() {return xAxis;
public LineWithObjectProperty getYAxis() {return yAxis;}
public Point getXAxisLocation() {return xAxisLocation;}
public Point getYAxisLocation() {return yAxisLocation;}
public void setAxesLength(int anAxesLength) {
  axesLength = anAxesLength;
  xAxis.setWidth(axesLength);
  yAxis.setHeight(axesLength);
  xAxisLocation = toXAxisLocation();
  yAxisLocation = toYAxisLocation();
  xAxis.setLocation(xAxisLocation );
  yAxis.setLocation(yAxisLocation );
  xLabel.setX(toXLabelX());
  xLabel.setY(toXLabelY());
  yLabel.setX(toYLabelX());
  yLabel.setY(toYLabelY());
}
Point toXAxisLocation() {
  return new ACartesianPoint(toXAxisX(), toXAxisY());
}
Point toYAxisLocation() {
   return new ACartesianPoint(toYAxisX(), toYAxisY());
}
```

Address copied, not object

Displaying New Cartesian Plane

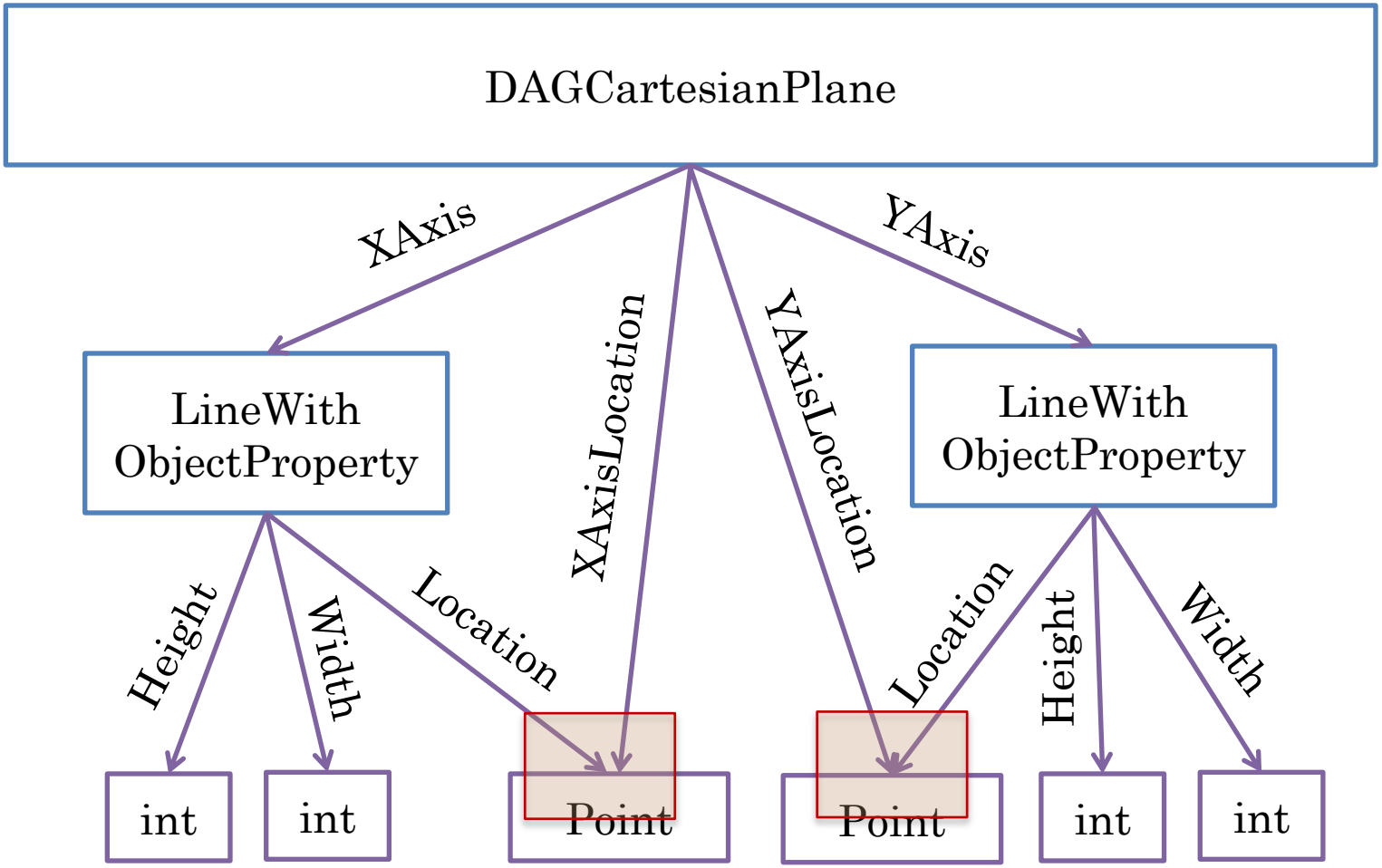
```
public interface DAGCartesianPlane {
 public LineWithObjectProperty get
 public LineWithObjectProperty get
 public Point getXAxisLocation();
 public Point getYAxisLocation();
 public int getAxesLength();
 public void setAxesLength(int anAxes
```

X and Y axes not shown in Tree View



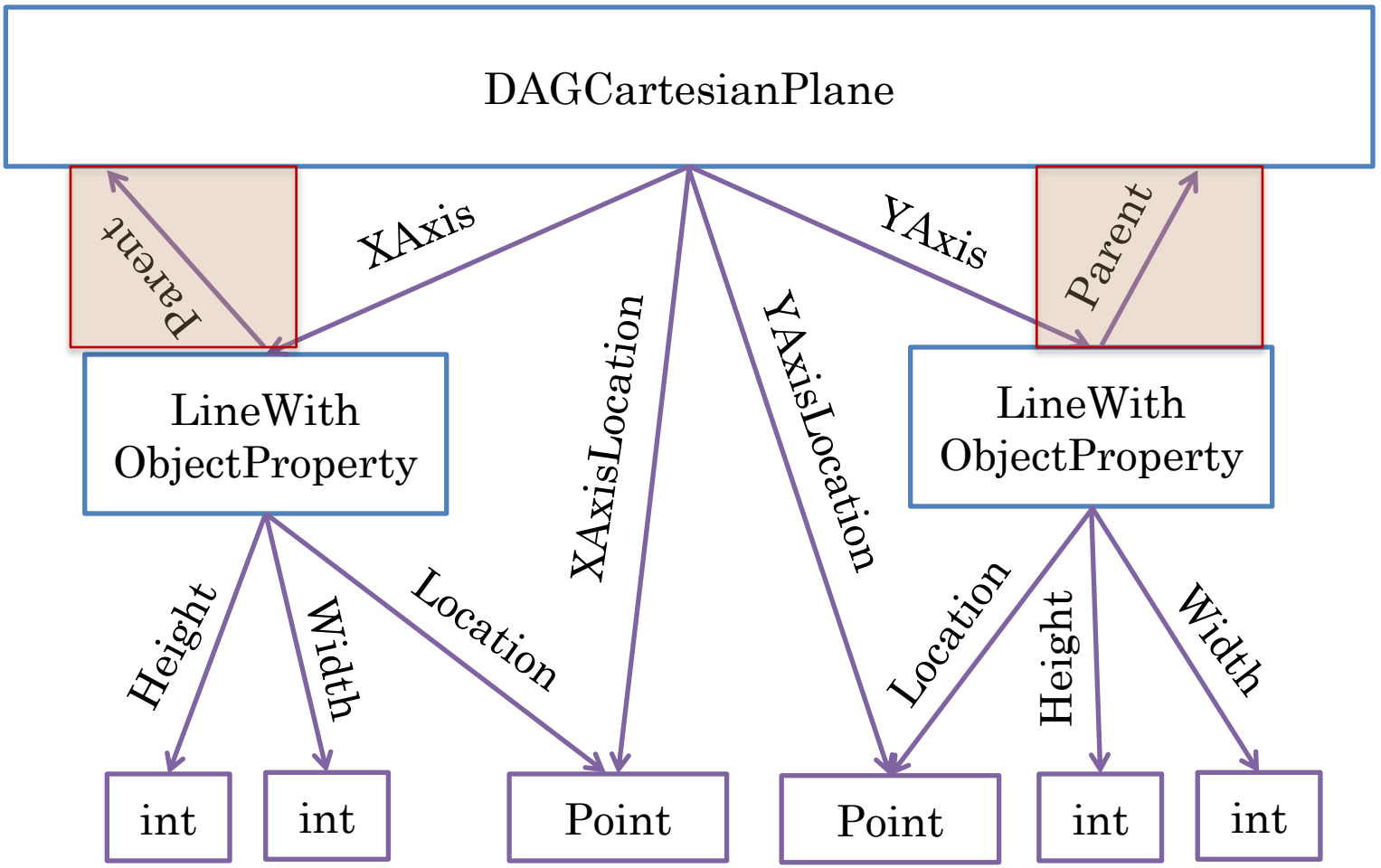

```
E*** Object:lectures.graphics.ACartesianPoint@10cafa1displayed multiple times with following references:
 [root.YAxis.location, root.YAxisLocation]

E*** Object:lectures.graphics.ACartesianPoint@15e00b7displayed multiple times with following references:
 [root.XAxisLocation, root.XAxis.location]
```

(Part of) DAG Logical Structure



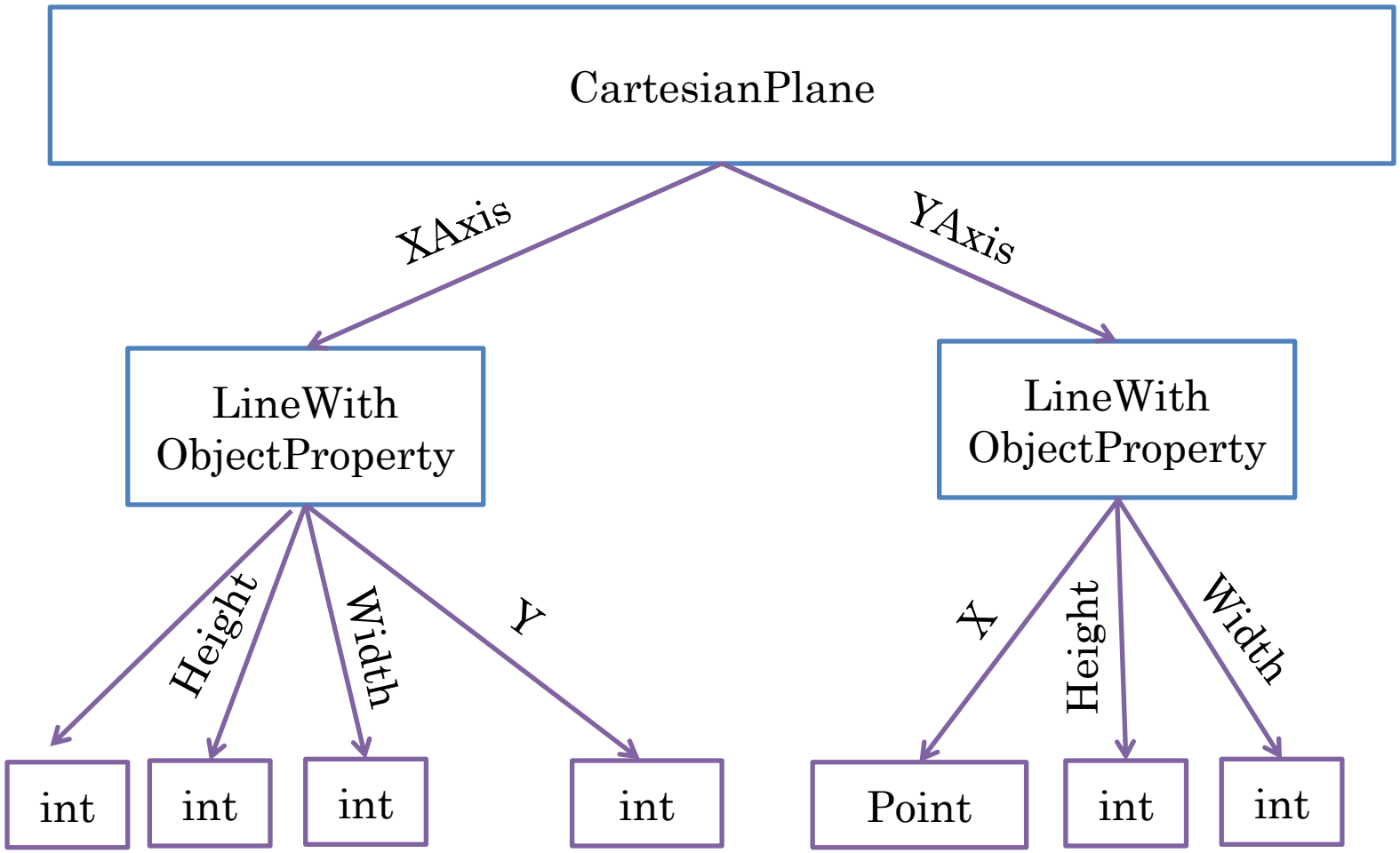
Can have multiple Paths to an Object (Node) but no cycles

Graph



Returning back to a node: cycle

Tree



Single Path to an Object (Node)

Types of Structures

- Tree
  - Each node has a single incoming edge
  - Cannot have multiple paths to a node
  - Each internal node roots a **subtree** in which no node has an edge to a node in any other subtree
- Directed Acyclic Graph (DAG)
  - Can have multiple incoming edges to a node
  - But cannot return back to a node when one follows edges
- Graph
  - A node can have multiple incoming edges to a node and thus can have multiple paths to a node
  - Can have cycles

Object Editor and Structures

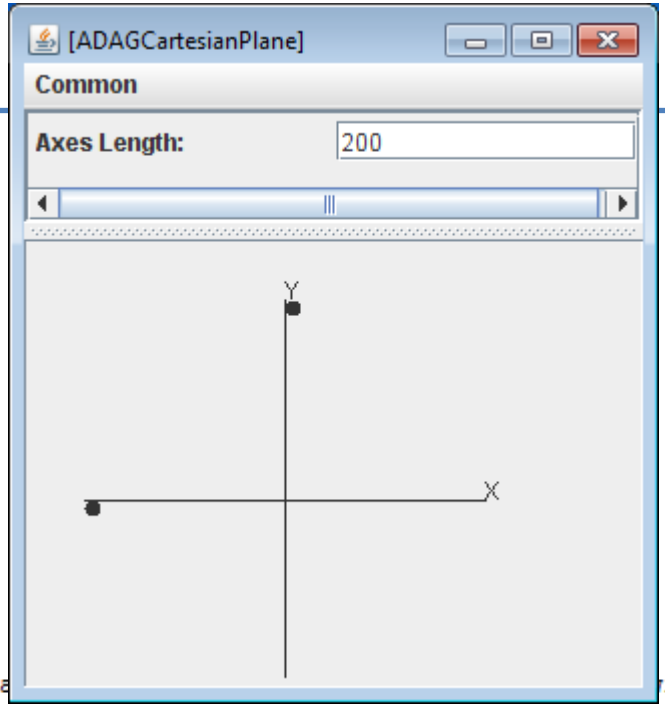
- ObjectEditor does not display certain non-tree logical structures
  - How to display them textually?
  - Performance and implementation reasons.
- Neither does Swing or AWT
- What if we have a non tree logical structure?

Two Problems

```
public interface DAGCartesianPlane {
 public LineWithObjectProperty getXAxis();
 public LineWithObjectProperty getYAxis();
 public Point getXAxisLocation();
 public Point getYAxisLocation();
 public int getAxesLength();
 public void setAxesLength(int anAxesLength);
 public StringShape getXLabel();
 public StringShape getYLabel();
}
```

Cannot see X and Y axis

See X and Y points




```
E*** Object:lectures.graphics.ACartesianPoint@10ca                    ying references:
 [root.YAxis.location, root.YAxisLocation]

E*** Object:lectures.graphics.ACartesianPoint@15e00b7displayed multiple times with following references:
 [root.XAxisLocation, root.XAxis.location]
```

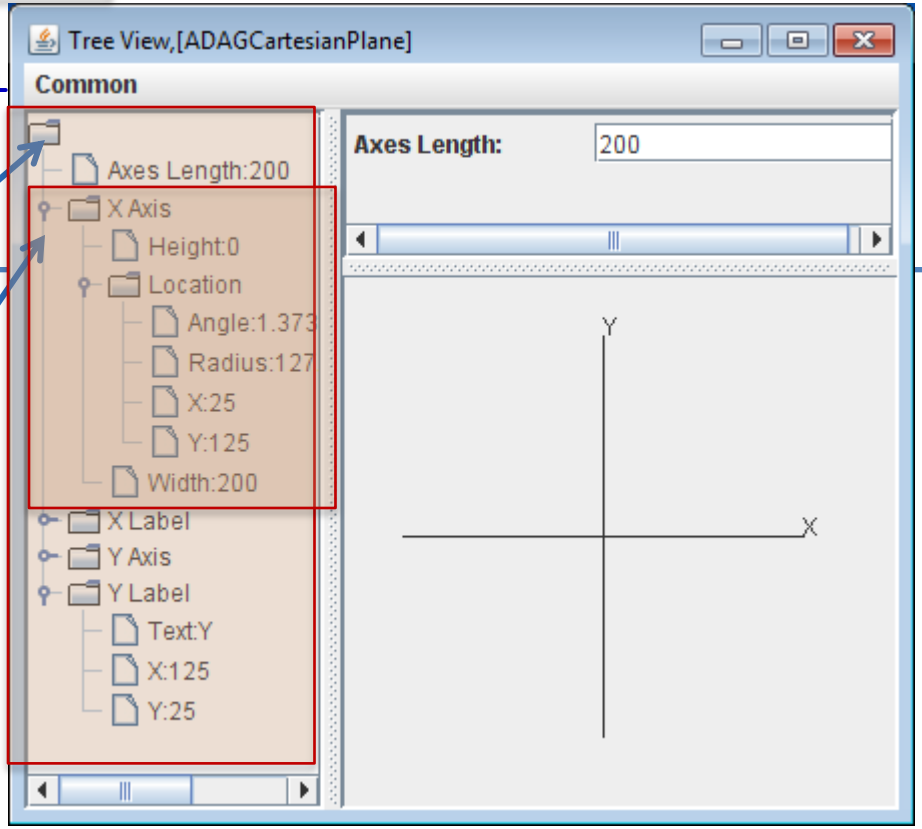
Making Display Structure a Tree

```
public class ADAGCartesianPlane implements DAGCartesianPlane {
    @Visible(false)
    public Point getXAxisLocation(
      return xAxisLocation;
    }
    @Visible(false)
    public Point
      return yAxi
    }
    …
}
```

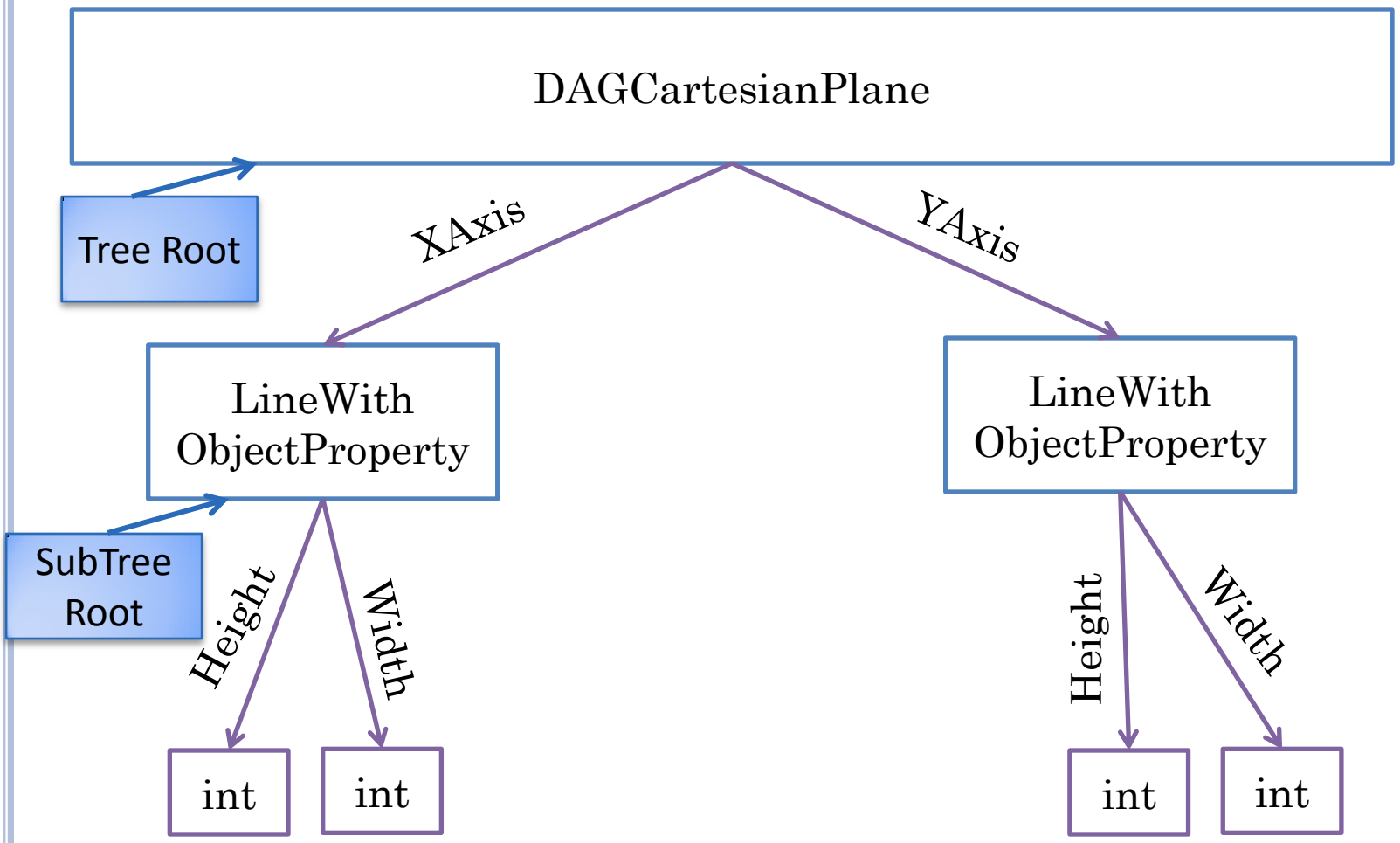
To remove P from the displayed structure add annotation @Visible(false) to its getter

Tree Root

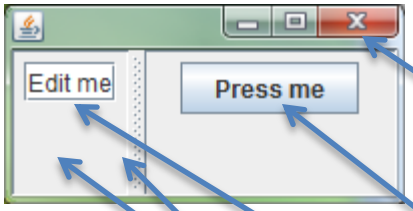
SubTree Root



(Part of) Tree Logical Display Structure



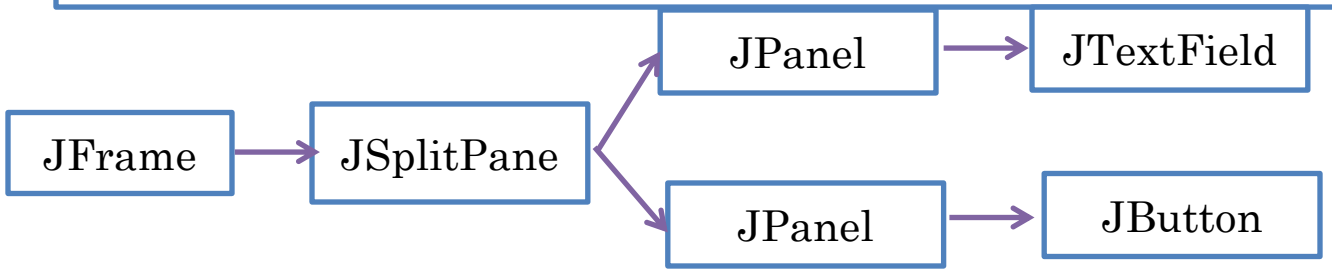
User Interface Structures

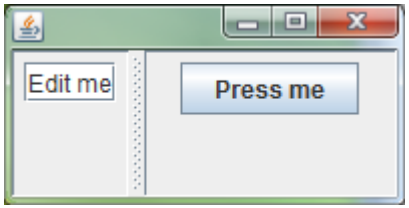


- User interfaces are also structures; they consist of objects called windows or widgets or (UI) components
- Root objects:
  - JFrame, Frame(top level window)
- Leaf level objects:
  - JButton
  - JTextField,
- Composite internal nodes:
  - JPanel – contains other UI components
  - JSplitPane – divides parent composite into two units with adjustable boundary

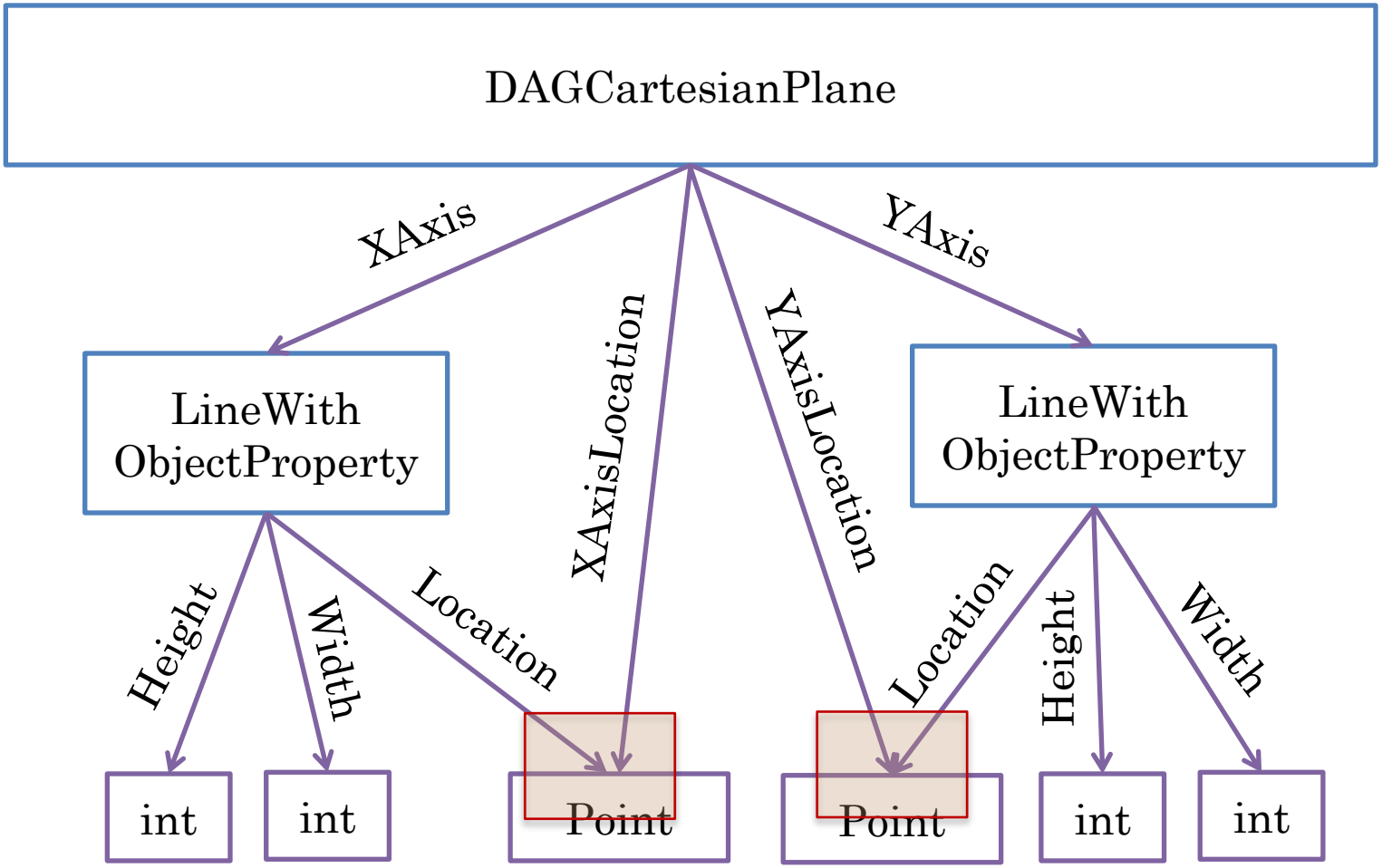
Window Tree Creator

```
public static JFrame createTree {
  JFrame frame = new JFrame();
  JSplitPane splitPane = new JSplitPane();
  frame.add(splitPane);
  JPanel leftPanel = new JPanel();
  JPanel rightPanel = new JPanel();
  splitPane.setLeftComponent(leftPanel);
  splitPane.setRightComponent(rightPanel);
  JTextField textField = new JTextField("Edit me");
  leftPanel.add(textField);
  JButton button = new JButton ("Press me");
  rightPanel.add(button);
  frame.setSize(200, 100);
  frame.setVisible(true);
  return frame;
}
```



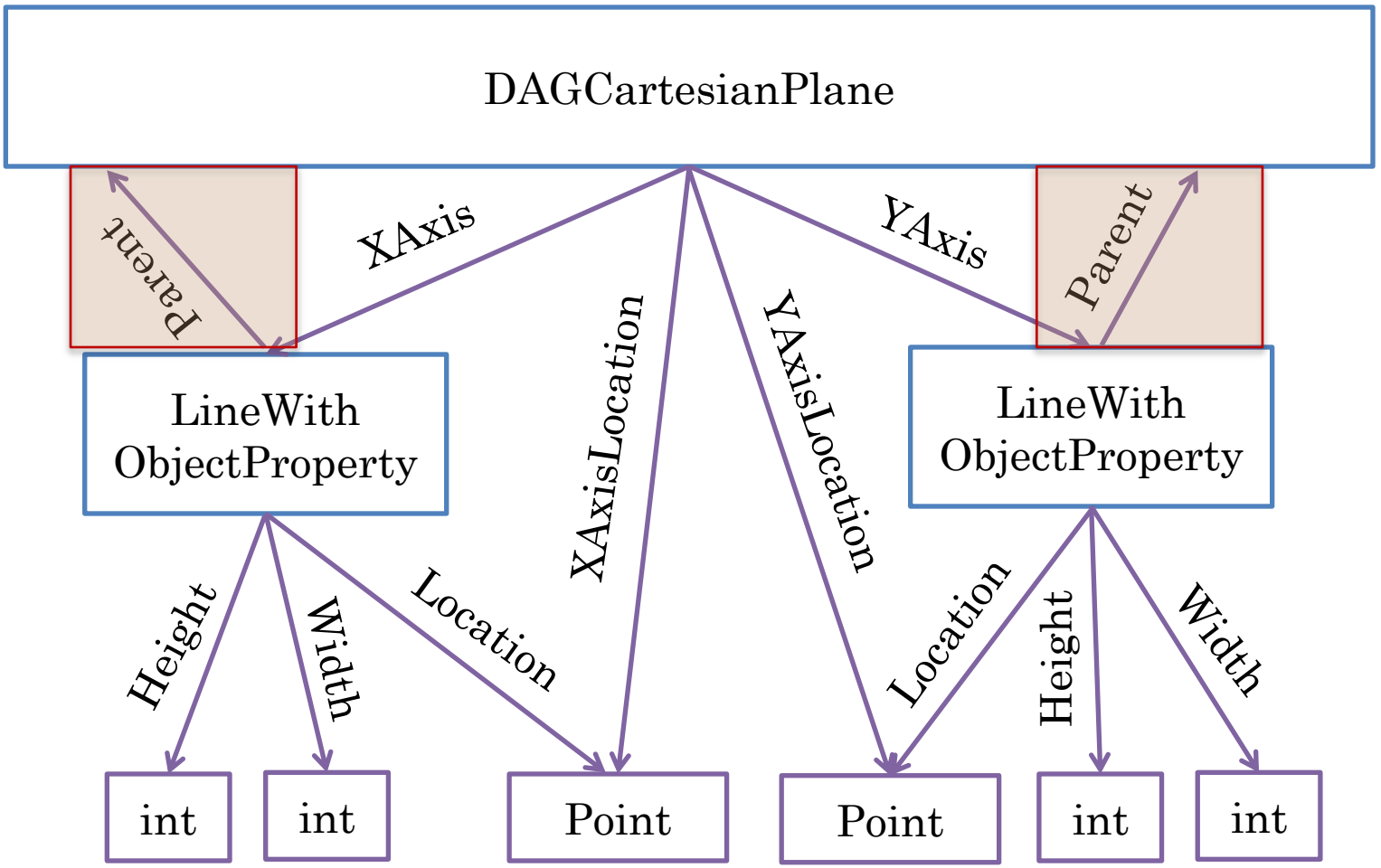


(Part of) DAG Logical Structure (Review)



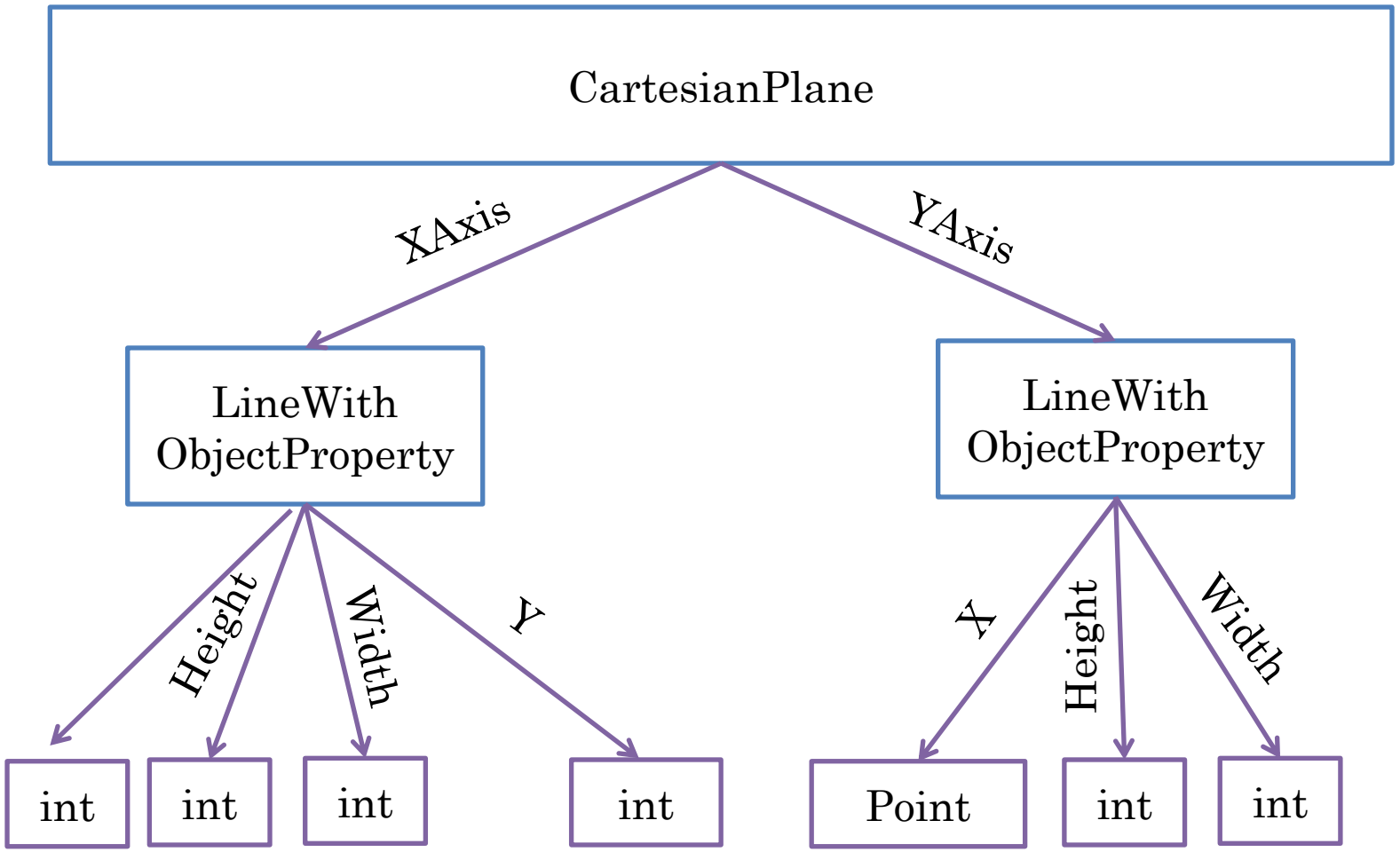
Can have multiple Paths to an Object (Node) but no cycles

Graph (Review)



Returning back to a node: cycle

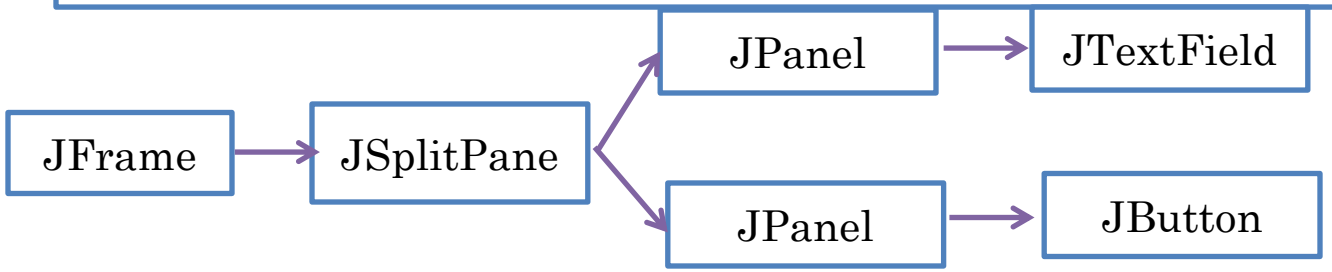
Tree (Review)

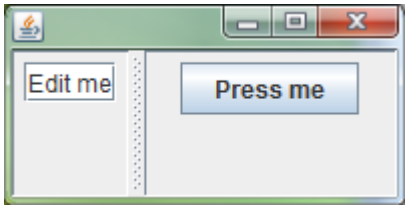


Single Path to an Object (Node)

Window Tree Creator

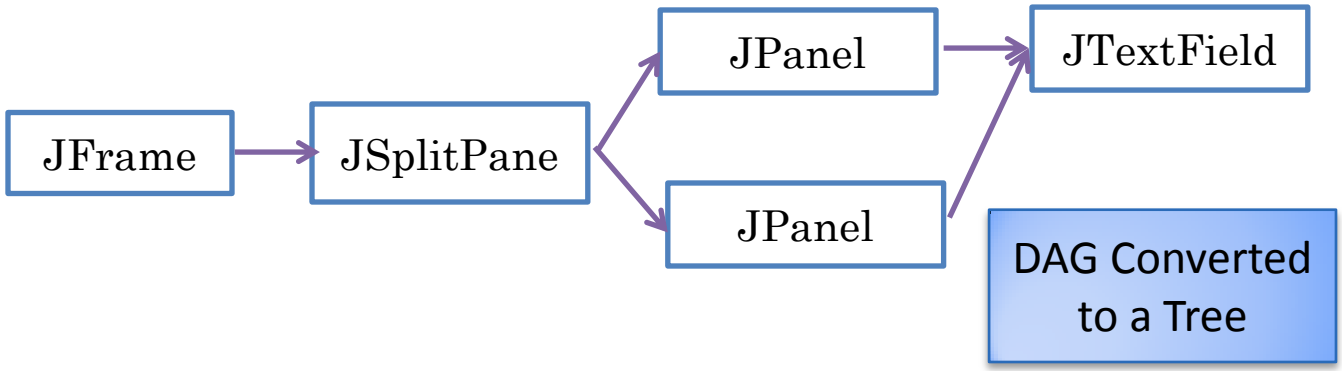
```
public static JFrame createTree {
  JFrame frame = new JFrame();
  JSplitPane splitPane = new JSplitPane();
  frame.add(splitPane);
  JPanel leftPanel = new JPanel();
  JPanel rightPanel = new JPanel();
  splitPane.setLeftComponent(leftPanel);
  splitPane.setRightComponent(rightPanel);
  JTextField textField = new JTextField("Edit me");
  leftPanel.add(textField);
  JButton button = new JButton ("Press me");
  rightPanel.add(button);
  frame.setSize(200, 100);
  frame.setVisible(true);
  return frame;
}
```

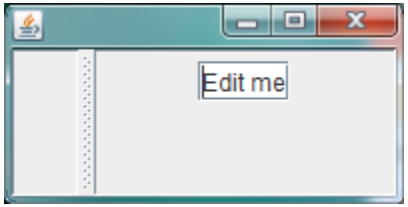




Window DAGCreator

```
  public static JFrame createTree {
    JFrame frame = new JFrame();
    JSplitPane splitPane = new JSplitPane();
    frame.add(splitPane);
    JPanel leftPanel = new JPanel();
    JPanel rightPanel = new JPanel();
    splitPane.setLeftComponent(leftPanel);
    splitPane.setRightComponent(rightPanel);
    JTextField textField = new JTextField("Edit me");
    leftPanel.add(textField);
    rightPanel.add(textField);
    frame.setSize(200, 100);
    frame.setVisible(true);
    return frame;
}
```

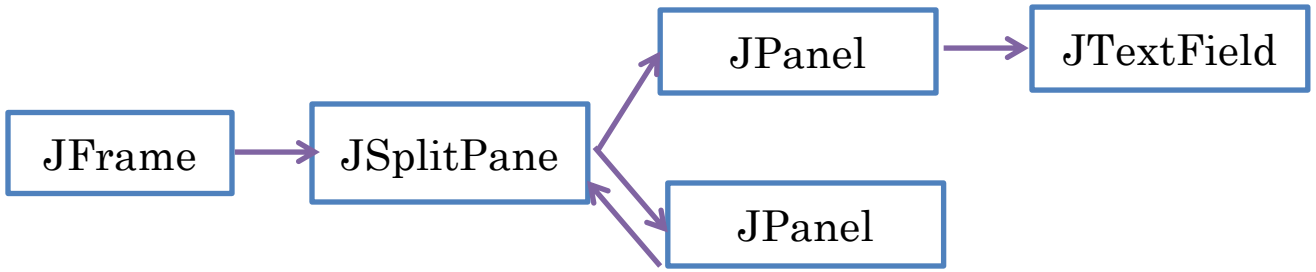




Window (Cyclic) Graph Creator

```
Exception in thread "main" java.lang.IllegalArgumentException: adding container's parent to its
        at java.awt.Container.checkAddToSelf(Unknown Source)
        at java.awt.Container.addImpl(Unknown Source)
        at java.awt.Container.add(Unknown Source)
        at lectures.composite.TreeVsDAG_Objects_Windows.WindowGraphCreator.main(WindowGraphCrea
```

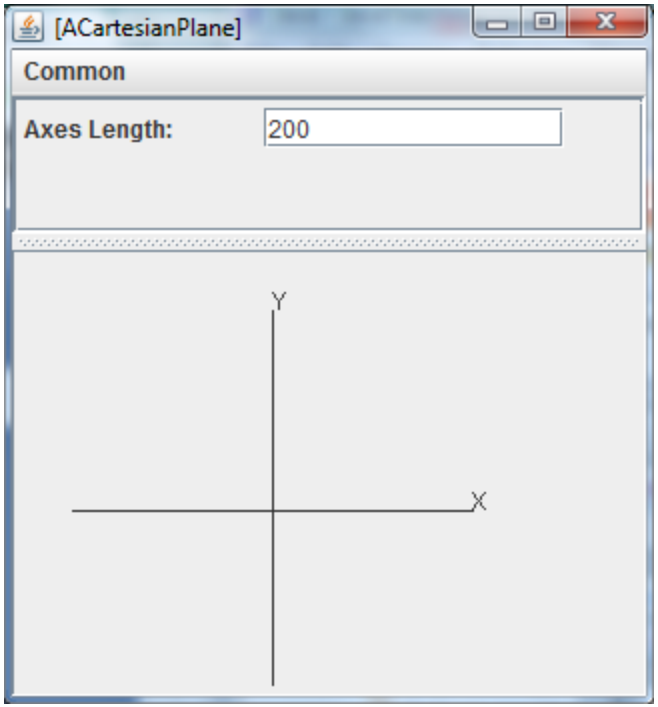
```
public static JFrame createTree {
    splitPane.setLeftComponent(leftPanel);
    splitPane.setRightComponent(rightPanel);
    JTextField textField = new JTextField("Edit me");
    leftPanel.add(textField);
    rightPanel.add(splitPane);
    frame.setSize(200, 100);
    frame.setVisible(true);
    return frame;
}
```



Window vs. Other Structures

- Both are object structures that can be trees, DAGs and arbitrary Graphs
- Window structures describe components of the user interface
- Non window object structures we have seen are mapped to window structures by ObjectEditor
- Some window structures are Beans
  - JSplitPane has fixed number of left and right component of with getters and setters
- Some are collections
  - JPanel has dynamic number of components added by add()

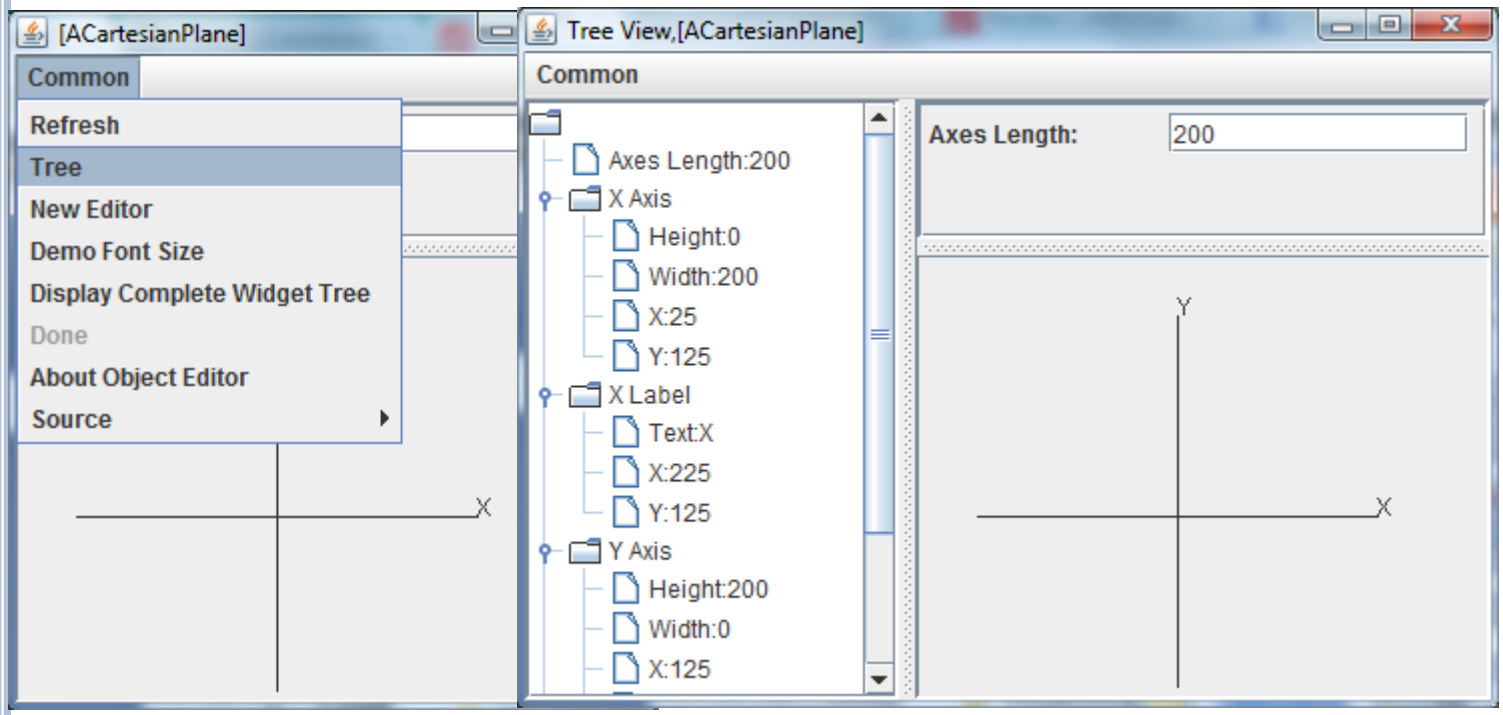
Window vs. Other Structures



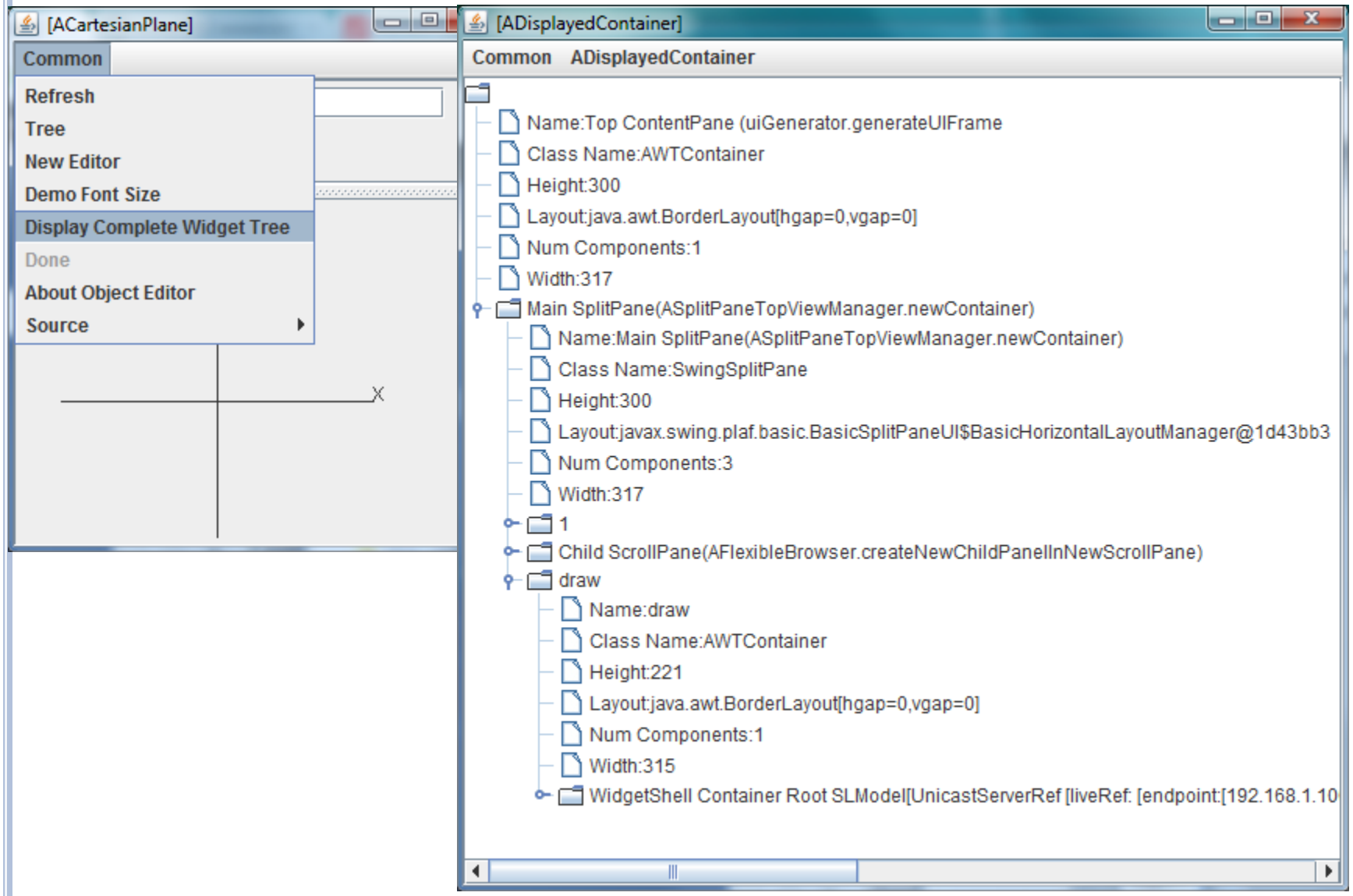
ACartesianPlane Structure?

JFrame Structure

ACartesianPlane Structure



Window Structures



Object Editor

Automatically maps logical structure to window structure

Rest are extra

User Interface Structures

- User interfaces are also structures; they consist of objects called windows or widgets or (UI) components
- Root objects:
  - JFrame, Frame(top level window)
- Leaf level objects:
  - JTextField, JButton, JSlider
- Composite internal nodes:
  - JPanel – contains other UI components
  - JSplitPane – divides parent composite into two units with adjustable boundary