Comp 401
Animation Loops

Instructor: Prasun Dewan

Prerequisite

- Loops Advanced

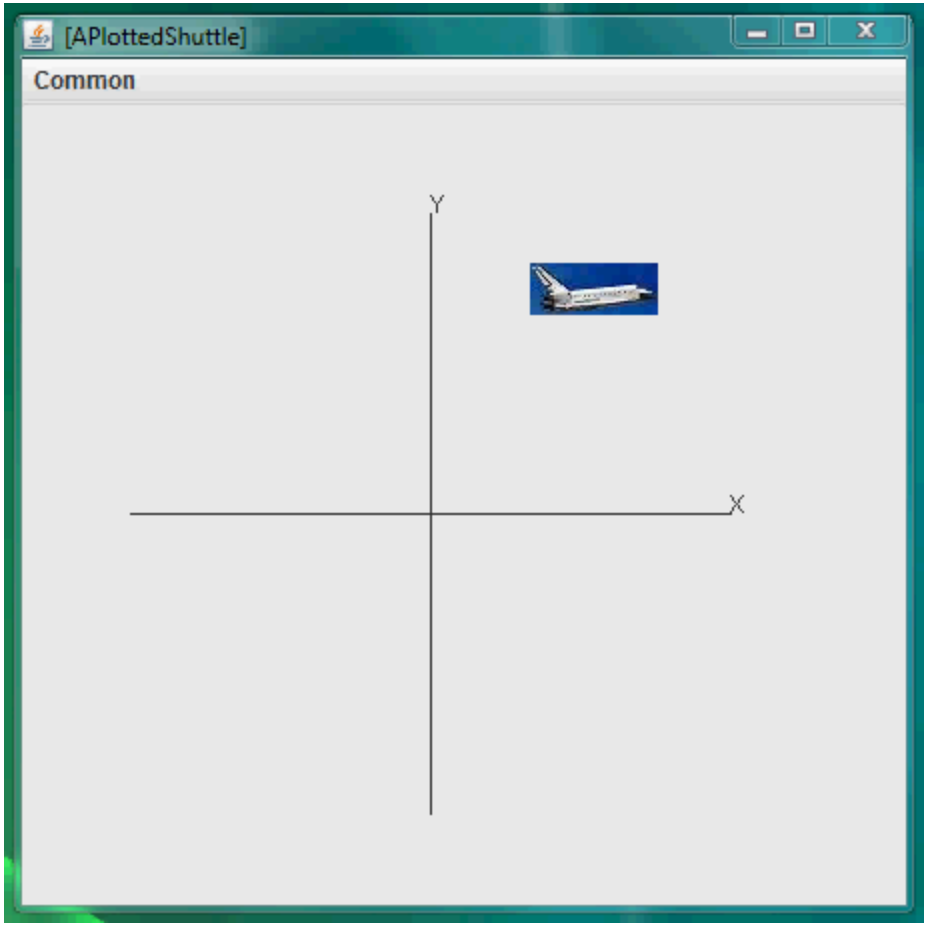
.

Topics

- Animation

.

Shuttle Animation



AnimateFromOrigin Semantics

- Basic Idea: Should animate from origin to current location
- Should move to origin
- Should animate first in Y direction to current Y
- Next should animate in X direction to current X

Animating to a Destination Location

1. Should move some distance (< total_distance) in straight line to destination
2. Go back to 1 if location not reached

Animating to a Destination Location

1. Should move a constant distance (< total_distance) in straight line to destination
2. Should pause for some time
3. Go back to 1 if location not reached

Pausing Program*

```
void sleep (int pauseTime) {
   int numberOfAssignments = pauseTime; //ASSIGNMENT_TIME;

    for (int i = 0; i < numberOfAssignments; i++) {
            int dummy = 0; // nonsense assignment
}
```

- ASSIGNMENT_TIME different on different computers!
- Unnecessarily using the CPU (Busy Waiting)
- Need the OS to suspend to put to sleep program for pause time

Pausing Program: ThreadSupport.Sleep()

```
public void sleep(int pauseTime) {
        try {
            // OS suspends program for pauseTime
            Thread.sleep(pauseTime);
        } catch (Exception e) {
            // program may be forcibly interrupted while sleeping
            e.printStackTrace();
         };
}
```

Animation in Y Direction

```
  protected void animateYFromOrigin(PlottedShuttle shuttle,
                        int animationStep, int animationPauseTime,
                        int startY, int endY) {
    // make sure we don't go past final Y position
    while (startY < endY) {
      ThreadSupport.sleep(animationPauseTime);
      startY += animationStep;
      shuttle.setShuttleY(startY);
    }
   // move to destination Y position
   shuttle.setShuttleY(endY);
}
```

AnimateFromOrigin

```
  public void animateFromOrigin(PlottedShuttle shuttle,
                int animationStep, int animationPauseTime) {
    //System.out.println ("Current thread:" + ());
    int originalX = shuttle.getShuttleX();
    int originalY = shuttle.getShuttleY();
    int curX = 0;
    int curY = 0;
    shuttle.setShuttleX(curX);
    shuttle.setShuttleY(curY);
    animateYFromOrigin(shuttle, animationStep,
                animationPauseTime, curY, originalY);
    animateXFromOrigin(shuttle, animationStep,
                        animationPauseTime, curX, originalX);
}
```

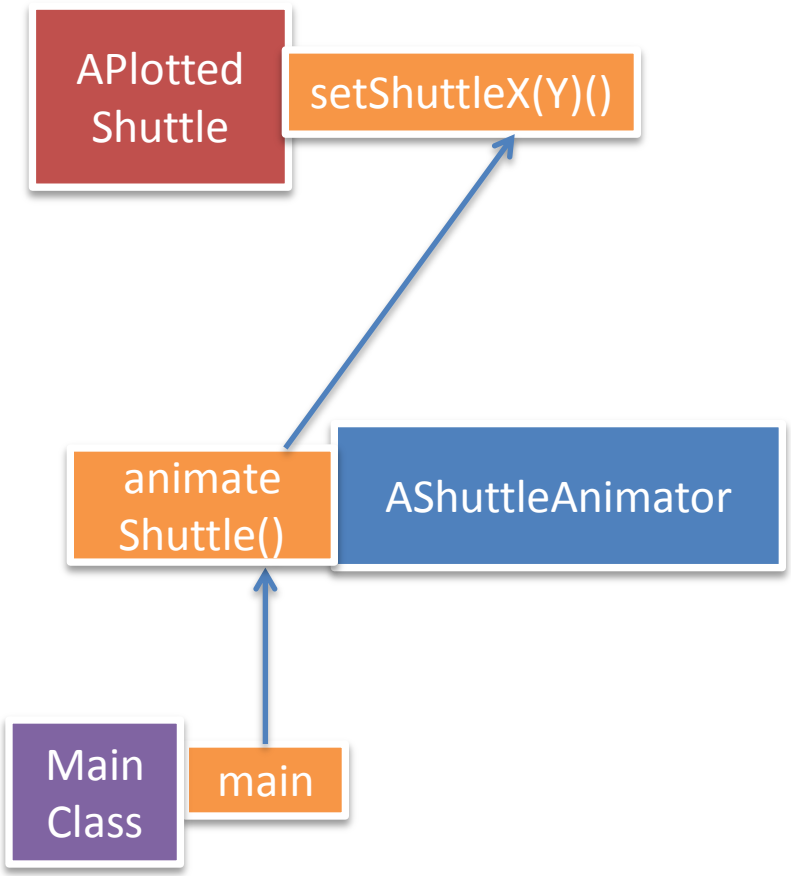
Main Method

```
public static void main(String[] args) {
  PlottedShuttle shuttle = new APlottedShuttle(50, 100);
  OEFrame oeFrame = ObjectEditor.edit(shuttle);
  oeFrame.hideMainPanel();
  oeFrame.setSize (450, 450);
  ShuttleAnimator shuttleAnimator = new AShuttleAnimator();
  shuttleAnimator.animateFromOrigin(aShuttle, 5, 100, anOEFrame);
}
```

Call animation code from main for now

Need to understand threads to trigger animation from the user interface

Basic Architecture



No Refresh

```
protected void animateYFromOrigin(PlottedShuttle shuttle,
                    int animationStep, int animationPauseTime,
                    int startY, int endY) {
   // make sure we don't go past final Y position
   while (startY < endY) {
     ThreadSupport.sleep(animationPauseTime);
     startY += animationStep;
     shuttle.setShuttleY(startY);
   }
  // move to destination Y position
  shuttle.setShuttleY(endY);
}
```

Shuttle being displayed by some GUI code (e.g. ObjectEditor)

Even if the method was called by it and it does automatic refresh

GUI needs to be told when shuttle coordinates change

Automatic refresh done when the method returns, not at indeterminate intermediate points

Manual Full Refresh

```
  protected void animateYFromOrigin(PlottedShuttle shuttle,
                      int animationStep, int animationPauseTime,
                      int startY, int endY, OEFrame frame) {
    // make sure we don't go past final Y position
    while (startY < endY) {
      ThreadSupport.sleep(animationPauseTime);
      startY += animationStep;
      shuttle.setShuttleY(startY);
      frame.refresh();
    }
   // move to destination Y position
   shuttle.setShuttleY(endY);
}
```

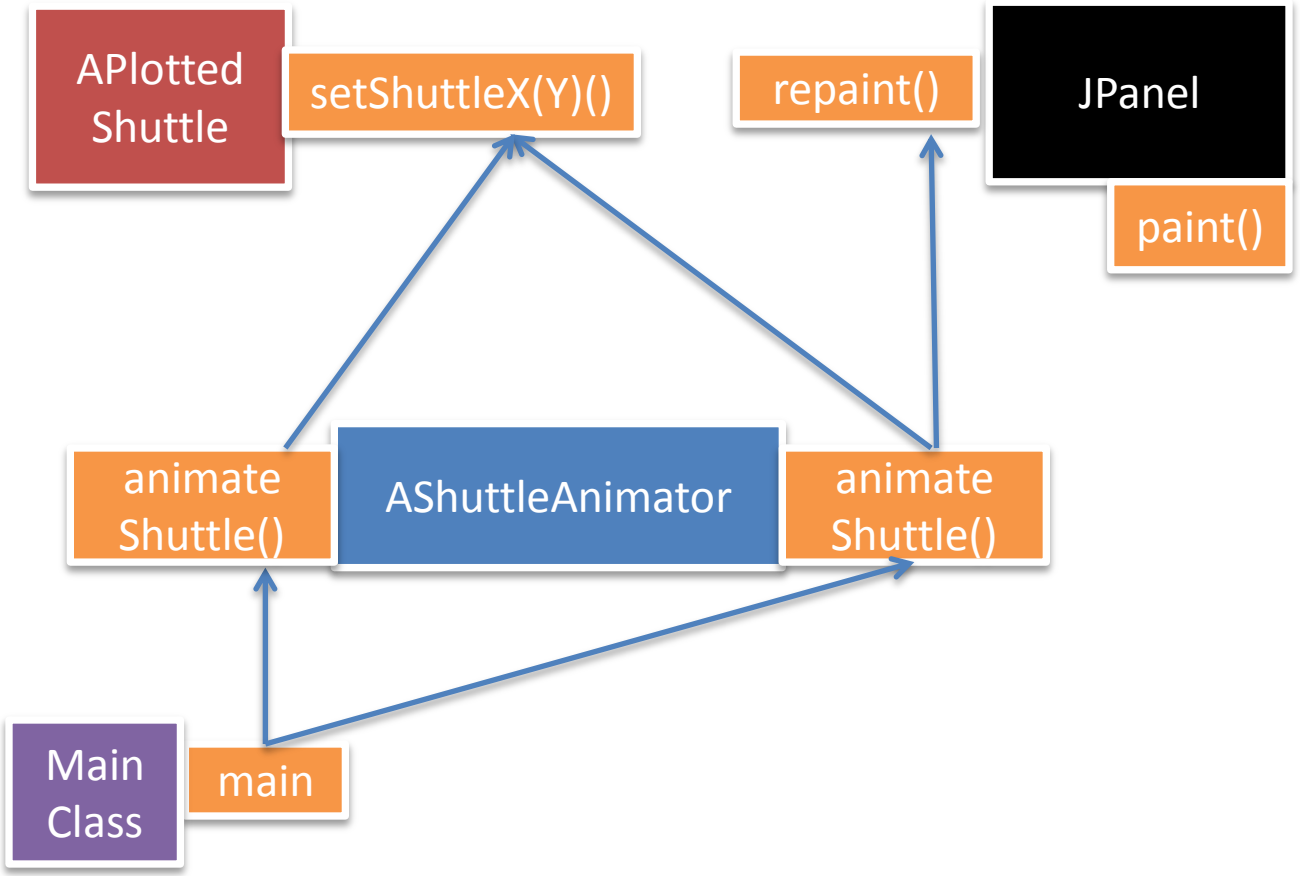
Manual Full Refresh

```
public class ShuttleAnimationDriver {
  public static void demoShuttleAnimation(
                          ShuttleAnimator aShuttleAnimator,
                          PlottedShuttle aShuttle,
                          OEFrame anOEFrame) {
    aShuttleAnimator.animateFromOrigin(aShuttle, 5, 100);
    aShuttleAnimator.animateFromOrigin(aShuttle, 5, 100, anOEFrame);
  }
  public static void main(String[] args) {
    PlottedShuttle shuttle = new APlottedShuttle(50, 100);
    OEFrame oeFrame = ObjectEditor.edit(shuttle);
    oeFrame.hideMainPanel();
    oeFrame.setSize (450, 450);
    ShuttleAnimator shuttleAnimator = new AShuttleAnimator();
    demoShuttleAnimation(shuttleAnimator, shuttle, oeFrame);
  }
}
```

Both animateFromOrigins take the same amount of time

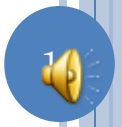
The refreshing one changes the display, the other one does not

Refreshing Architecture



Problem with Full Refresh

- The entire UI is refreshed, not just the part that changes
- Multiple GUIs (views) may be displaying the same object
- Should create animated objects as observables

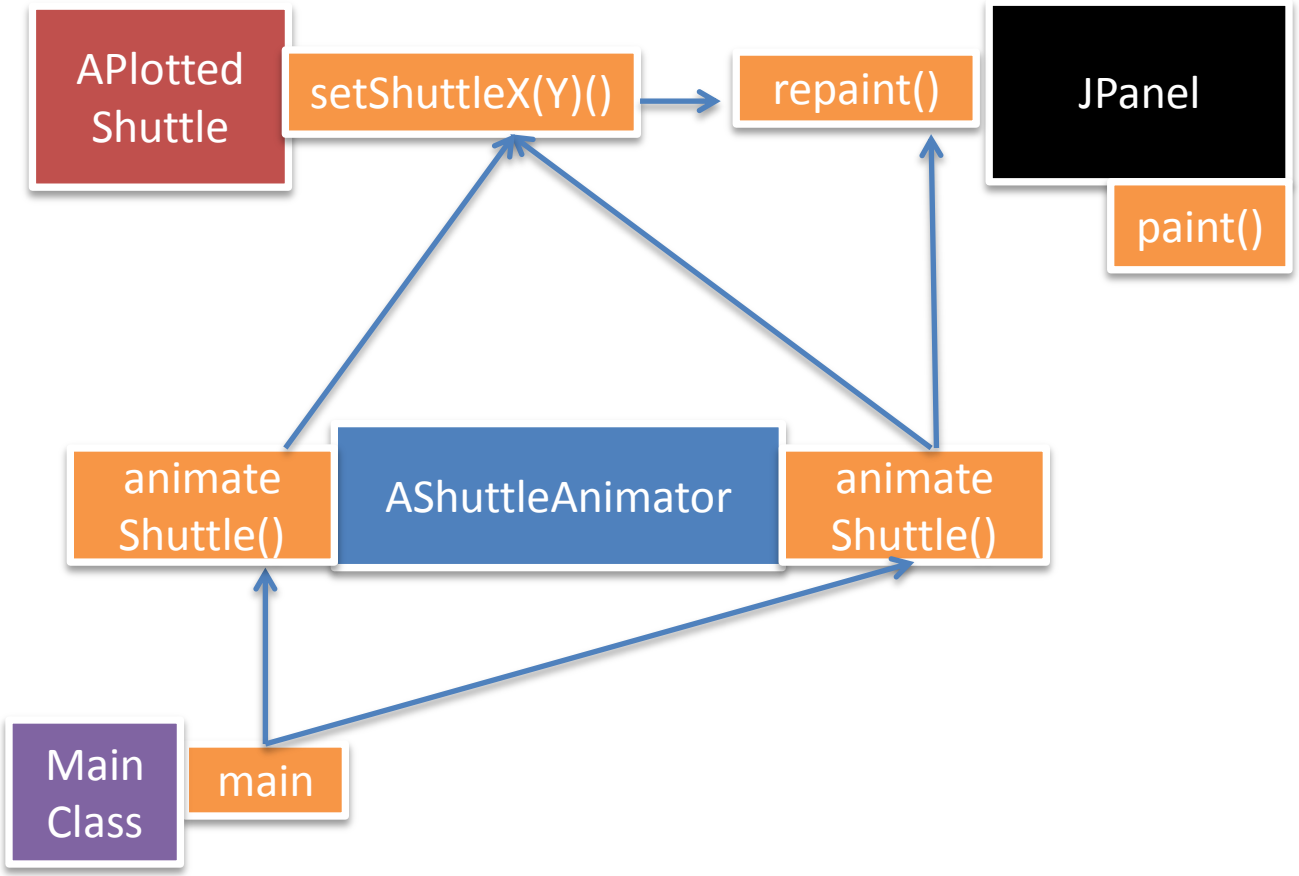


Using MVC

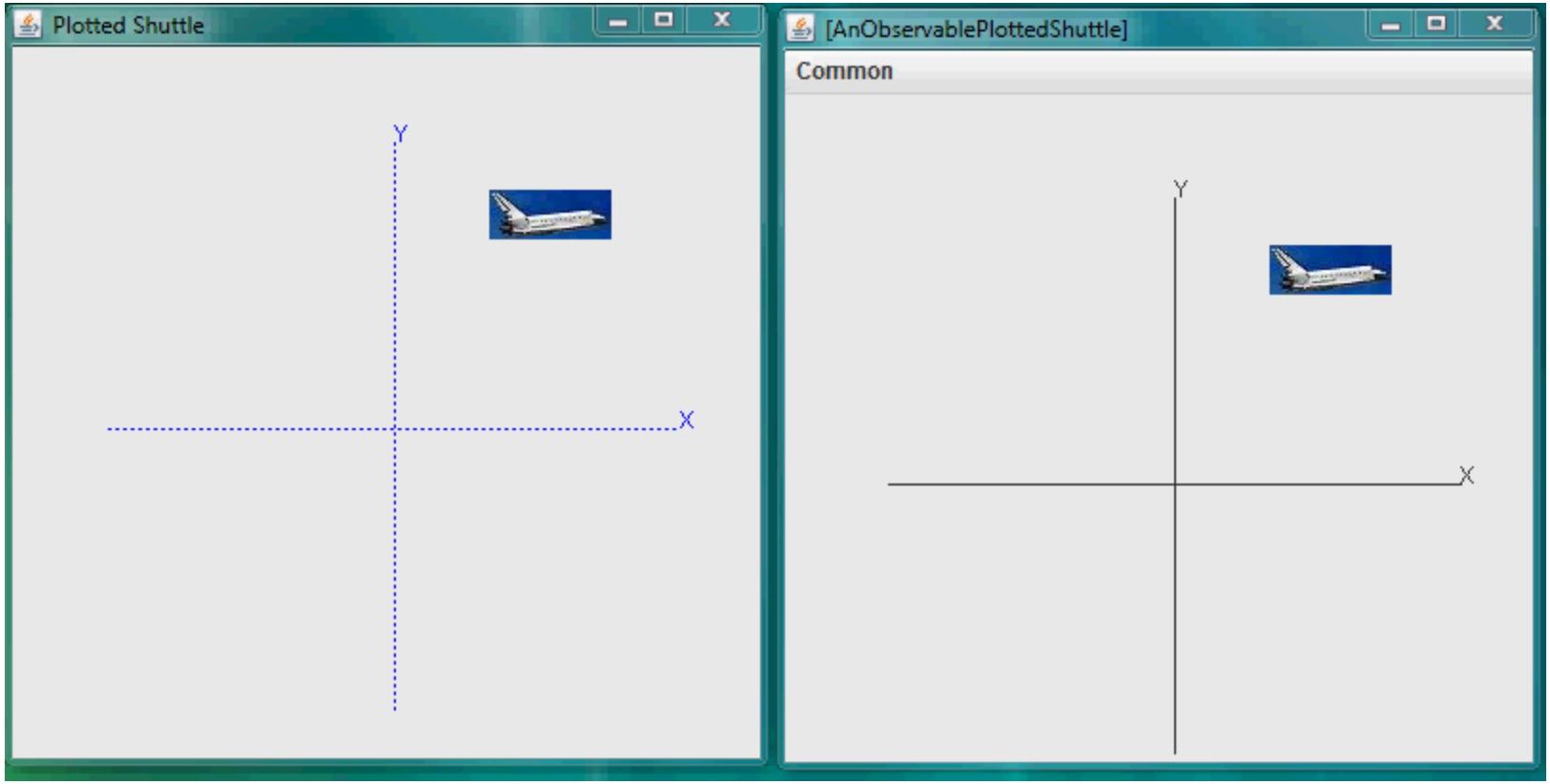
```
  public static void demoShuttleAnimation(
                        ShuttleAnimator aShuttleAnimator,
                        PlottedShuttle aShuttle,
                        OEFrame anOEFrame) {
    aShuttleAnimator.animateFromOrigin(aShuttle, 5, 100);
    aShuttleAnimator.animateFromOrigin(aShuttle, 5, 100, anOEFrame);
```

```
public class ObservableShuttleAnimationDriver extends
ShuttleAnimationDriver {
public static void main(String[] args) {
  ObservablePlottedShuttle shuttle =
        new AnObservablePlottedShuttle(50, 100);
  OEFrame oeFrame = ObjectEditor.edit(shuttle);
  oeFrame.hideMainPanel();
  oeFrame.setSize (400, 400);
  oeFrame.setLocation(400, 0);
  BeanView view = new APlottedShuttleJPanelAndView(shuttle);
  JFrame frame = new JFrame("Plotted Shuttle");
  frame.add((Component) view);
  frame.setSize(400, 400);
  frame.setVisible(true);
  ShuttleAnimator shuttleAnimator = new AShuttleAnimator();
  demoShuttleAnimation(shuttleAnimator, shuttle, oeFrame);
}
```

MVC Architecture



Animation and MVC



Main vs. Interactive Animation

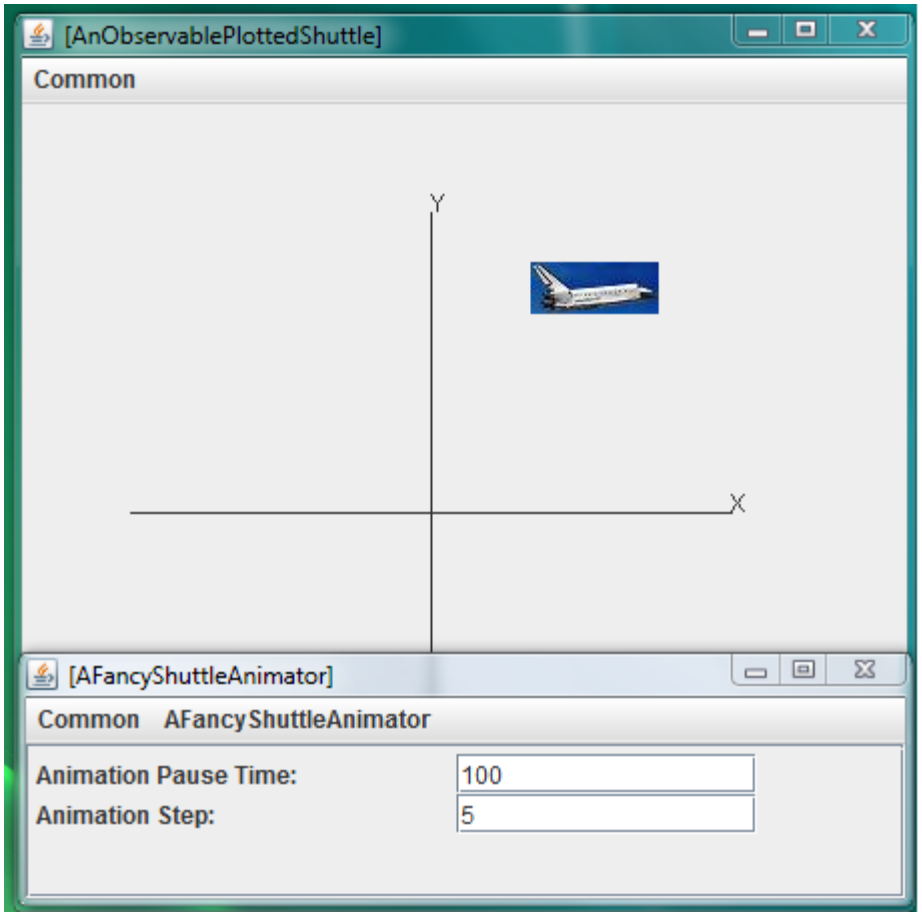
```
public static void main(String[] args) {
  PlottedShuttle shuttle = new APlottedShuttle(50, 100);
  OEFrame oeFrame = ObjectEditor.edit(shuttle);
  oeFrame.hideMainPanel();
  oeFrame.setSize (450, 450);
  ShuttleAnimator shuttleAnimator = new AShuttleAnimator();
  shuttleAnimator.animateFromOrigin(aShuttle, 5, 100);
}
```

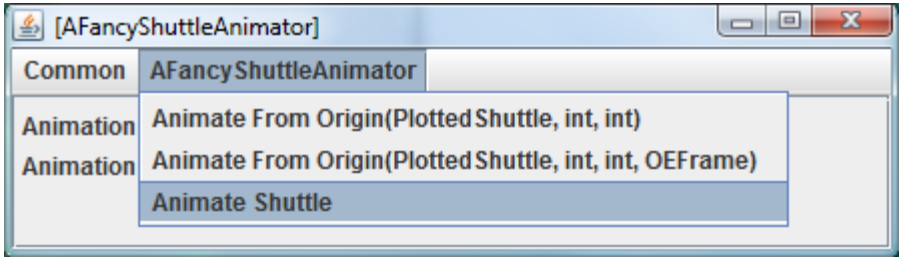
```
public static void main (String[] args) {
  PlottedShuttle shuttle = new APlottedShuttle(50, 100);
  OEFrame oeFrame = ObjectEditor.edit(shuttle);
  oeFrame.hideMainPanel();
  oeFrame.setSize (450, 450);
  FancyShuttleAnimator shuttleAnimator = new AFancyShuttleAnimator();
  ObjectEditor.edit(shuttleAnimator);
}
```

Fancy Animator

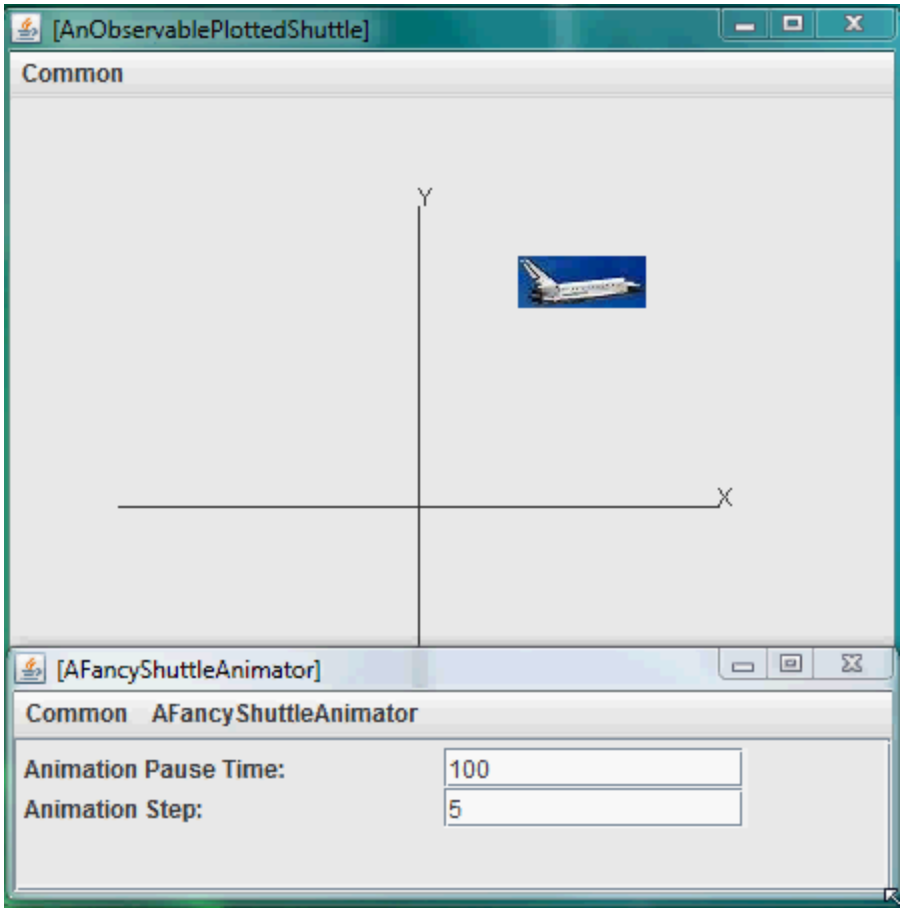
```
public class AFancyShuttleAnimator extends AShuttleAnimator
                                   implements FancyShuttleAnimator {
  int animationStep = 5;
  int animationPauseTime = 100;
  PlottedShuttle shuttle;
  public AFancyShuttleAnimator(PlottedShuttle theShuttle) {
    shuttle = theShuttle;
  }
  public int getAnimationStep() {
    return animationStep;
  }
  public void setAnimationStep(int animationStep) {
    this.animationStep = animationStep;
  }
  public int getAnimationPauseTime() {
    return animationPauseTime;
  }
  public void setAnimationPauseTime(int animationPauseTime) {
    this.animationPauseTime = animationPauseTime;
  }
  public void animateShuttle() {
    animateFromOrigin(shuttle, animationStep, animationPauseTime);
  }
}
```

GUI


[AnObservablePlottedShuttle]
Common
Y
X
[AFancyShuttleAnimator]
Common
AFancyShuttleAnimator
Animation Pause Time:
100
Animation Step:
5


[AFancyShuttleAnimator]
Common
AFancyShuttleAnimator
Animation
Animation
Animate From Origin(PlottedShuttle, int, int)
Animate From Origin(PlottedShuttle, int, int, OEFrame)
Animate Shuttle

Video


[AnObservablePlottedShuttle]
Common
Y
X
[AFancyShuttleAnimator]
Common
AFancyShuttleAnimator
Animation Pause Time:
100
Animation Step:
5

UI vs Main Thread

Problem is that the user interface both executes loop and does the redraws the screen

Redrawing of screen done after loop is over

In the main case, the man thread or activity, executes loop

The UI thread or activity redraws the screen

Asking ObjectEditor to Interactive Call in Separate Thread

```
@SeparateThread(true)
 public void animateShuttle() {
   animateFromOrigin(shuttle, animationStep, animationPauseTime);
 }
```