```
        x = rect.x;  y = rect.y;
        w = rect.w;  h = rect.h;
    }
    //--------------
    public void restore () {
        //restore the internal state of
        //the specified rectangle
        rect.x = x;  rect.y = y;
        rect.h = h;  rect.w = w;
    }
}
```

When we create an instance of the Memento class, we pass it the visRectangle instance that we want to save. It copies the size and position parameters and saves a copy of the instance of the visRectangle itself. Later, when we want to restore these parameters, the Memento knows which instance must be restored and can do it directly, as shown in the *restore* method.

The rest of the activity takes place in the Mediator class, where we save the previous state of the list of drawings as the Integer on the undo list.

```
    public void createRect(int x, int y) {
        unpick();    //make sure no rectangle is selected
        if (startRect)    {    //if the Rect button is depressed
            Integer count = new Integer(drawings.size());
            caretaker.addElement(count);   //save the previous size
            visRectangle v = new visRectangle(x, y);
            drawings.addElement(v);        //add a new element
            startRect = false;             //done with a rectangle
            rect.setSelected(false);       //unclick the button
            canvas.repaint();
        }else
            pickRect(x, y);    //if not pressed look for the Rect
    }
```

We also save the previous position of a rectangle before moving it into a Memento.

```
    public void rememberPosition() {
        if (rectSelected) {
            Memento m = new Memento (selectedRectangle);
            caretaker.addElement(m);
            repaint();
        }
    }
```

The *undo* method simply decides whether to reduce the drawing list by one or to invoke the *restore* method of a Memento.

```
public void undo() {
        if (undoList.size() > 0) {
```