* **Runtime Polymorphism/Dynamic Binding:**
  + This polymorphism happens during runtime and hence the name. This works on the principle that object of an interface and be assigned instance of all the classes that implement it.
  + Thus different methods from different classes act as the method definition. And thus the definition of a certain function changes during the runtime.
  + When an instance invokes a method it is called static binding as it takes place during compilation.
* **Chaining Model:**
* Graphics:
  + Jdk1.0
    - java.awt has a package Graphics.
    - Graphics is a abstract class which need not be extended.
  + Jdk 2.0
    - Graphic2D extended class Graphics.
    - Device Context
      * Screen.
      * Printer
    - Container Context
      * JFrame
      * JPanel.
      * Jcomponent.
      * All the above 3 have their own g, g is object of the class Graphics.
* Indirect Graphics:
  + Can be done without threads.
  + This is for parametric programing.
  + Donot mix GUI and graphics at the same time. Incase needed add graphics to a Jpanel first and then add the panel to the frame.
  + Methods in JFrame: paint(), update(), repaint().
  + When changing the parameter we call repaint() methods which calls update(). Update() clears the container and calls paint(). Now as paint is overloaded the paint defined in user defined class is called.
  + Repaint() method should be overloaded it has to be inherited.
  + For graphics upper left corner is the origin unlike maths functions.
  + Thus care should be taken while working with cos and sin. It should be multiplied by -1.
  + On execution the setVisible method calls the paint method of Jframe. As paint is overridden by us our paint is called.

Public void paint(Graphic g)

{

super.paint(g);

g.drawLine(100,100,300,300);

g.drawString(“Hello World”, 310,310);

g.drawRect(x,y,width,height);

g.fillRect(x,y,width,height);

Color clr;

Clr = new Color(\_\_,\_\_\_,\_\_\_);

jbtnClick.setForeground(Color.blue);

jbtnClicl.setBackground(Color.yellow);

g.setColor(Color.blue);

g.drawRect();

g.setColor(Color.red);

g.fillRect(\_\_,\_\_ ,\_\_ ,\_\_ )

g.setColor(Color.red);

g.drawOval(x,y,width,height) //width=height draws circle.

// width != height draws eclipse.

g.setColor(Color.blue);

g.fillOval(\_,\_,\_,\_);

g.drawArc();

g.fill Arc();

}

* **Java Beans:**
  + These are classes with their instance data ‘xxx’ and corresponding getXxx() and setXxx() methods.
  + JTextField:
    - Document doc;