TRINITY INTERNATIONAL COLLEGE

(Tribhuvan University Affiliated)

Lab Assignment 6: Advance Java Programming

Submitted By:

Name: Dipendra Shrestha

Program: **B. Sc. (CSIT)**

Roll No: 16

Semester: seventh (7th)

Date: 25th June 2020

Submitted to:

Aman Maharjan

KATHMANDU, NEPAL
2020

1. Write a suitable program illustrating the use of Java Beans. [2069]

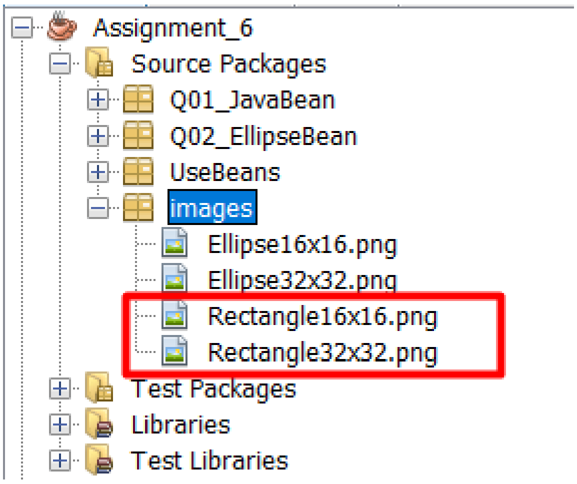
⇨ Program

```java
package Q01_JavaBean;

import java.awt.Graphics;
import javax.swing.JPanel;

public class Rectangle extends JPanel
{
    public Rectangle() { }
    private int length = 0;
    private int breadth = 0;
    public int getLength()
    {
        return length;
    }
    public void setLength(int length)
    {
        this.length = length;
        repaint();

    }
    public int getBreadth()
    {
        return breadth;
    }
    public void setBreadth(int breadth)
    {
        this.breadth = breadth;
        repaint();
    }

    @Override
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);

        if (length <= 0 || breadth<=0) return;
        int x = 50, y = 50;
        int l = length, b = breadth;

        g.drawRect(x, y, l, y);
    }
}
```

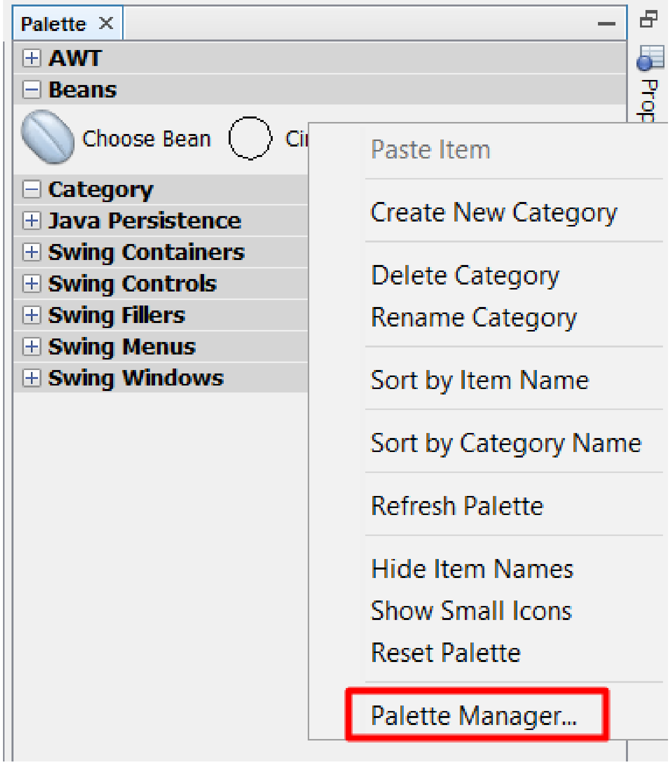
For Rectangle icon

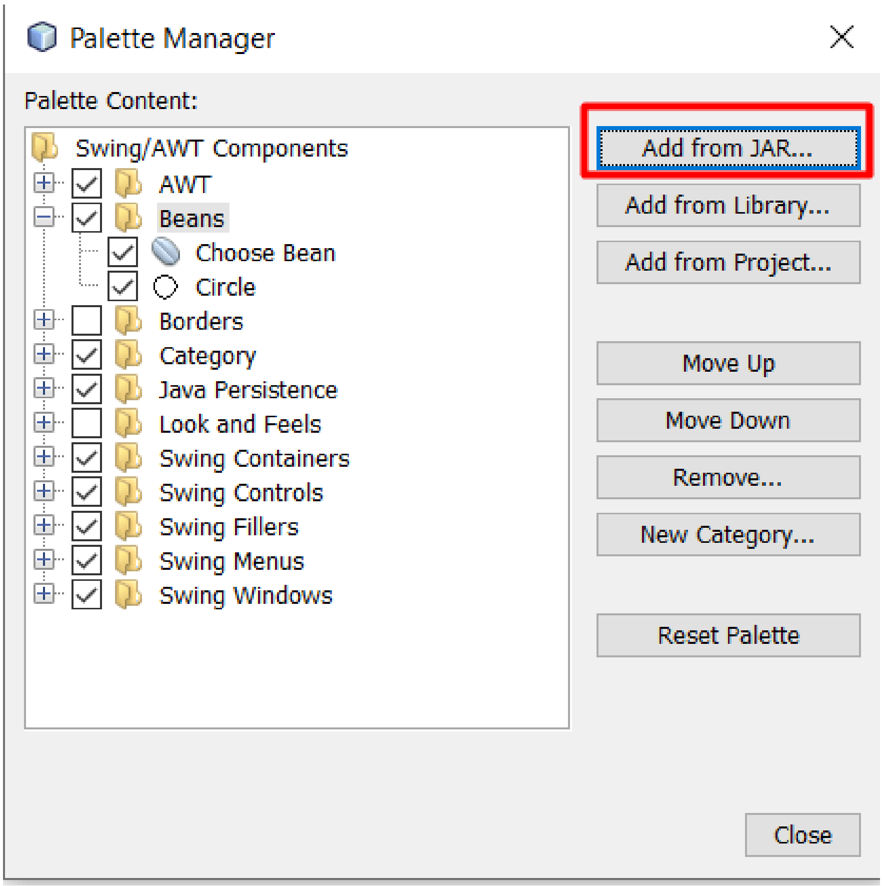
```
package Q01_JavaBean;

import java.awt.Image;
import java.beans.BeanInfo;
import java.beans.SimpleBeanInfo;

public class RectangleBeanInfo extends SimpleBeanInfo
{
        private final Image iconColor16;
        private final Image iconColor32;
        private final Image iconMono16;
        private final Image iconMono32;

        public RectangleBeanInfo()
        {
            iconColor16 = loadImage("/images/Rectangle16x16.png");
            iconColor32 = loadImage("/images/Rectangle32x32.png");
            iconMono16 = loadImage("/images/Rectangle16x16.png");
            iconMono32 = loadImage("/images/Rectangle32x32.png");
        }

        @Override
        public Image getIcon(int iconType)
         {
            switch (iconType)
            {
                case BeanInfo.ICON_COLOR_16x16:
                    return iconColor16;
                case BeanInfo.ICON_COLOR_32x32:
                    return iconColor32;
                case BeanInfo.ICON_MONO_16x16:
                    return iconMono16;
                case BeanInfo.ICON_MONO_32x32:
                    return iconMono32;
                default:
                    return null;
            }
        }
    }
```

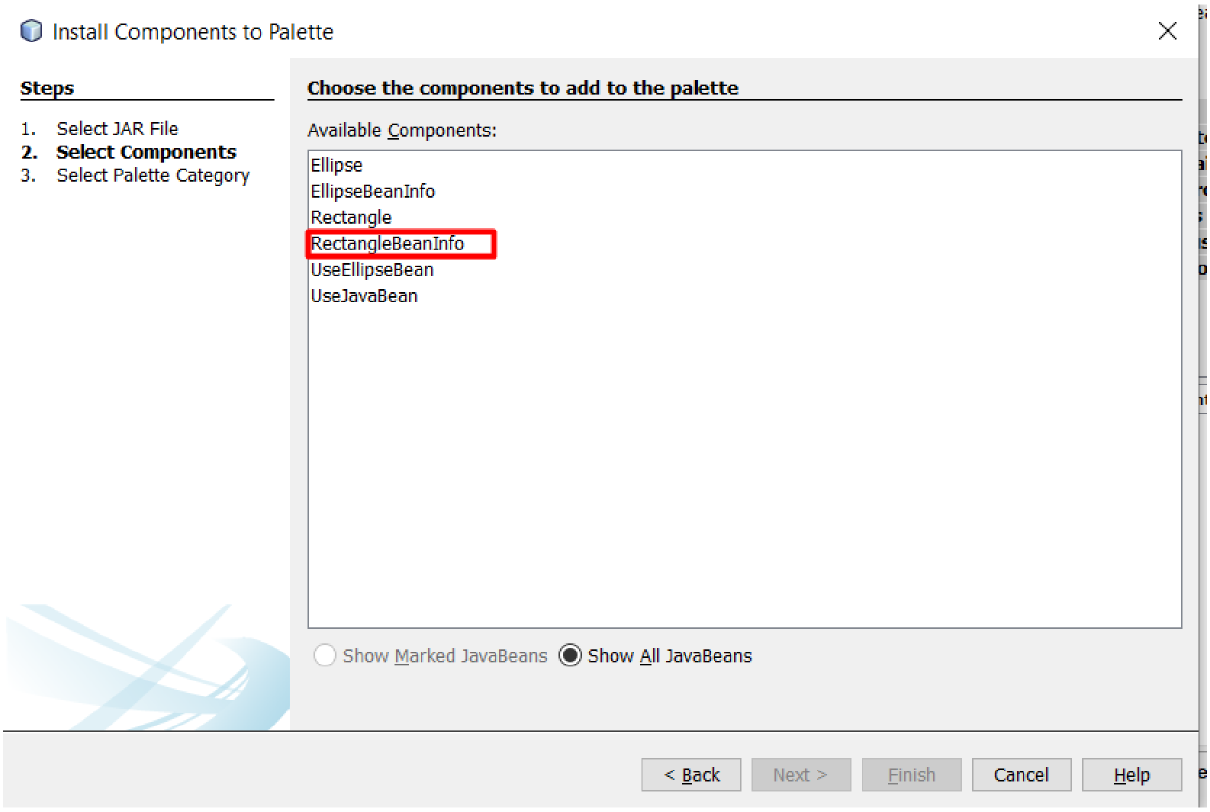
Icon images are kept inside images folder as:



Adding custom bean i.e. Rectangle to palette Manager







Install Components to Palette

Steps

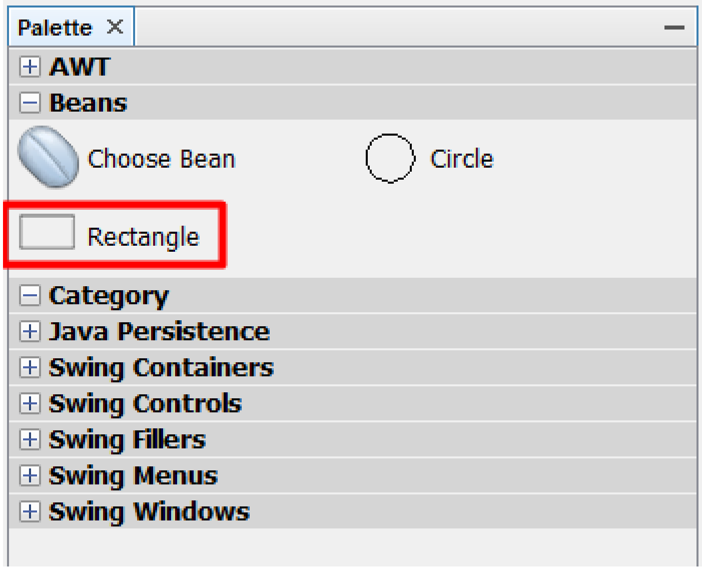
1. Select JAR File
2. **Select Components**
3. Select Palette Category

Choose the components to add to the palette

Available Components:

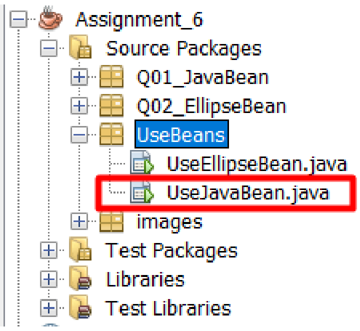
Ellipse
EllipseBeanInfo
Rectangle
RectangleBeanInfo
UseEllipseBean
UseJavaBean

○ Show Marked JavaBeans ● Show All JavaBeans

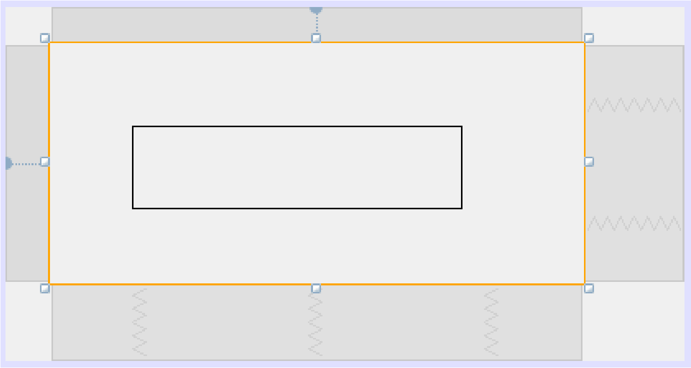


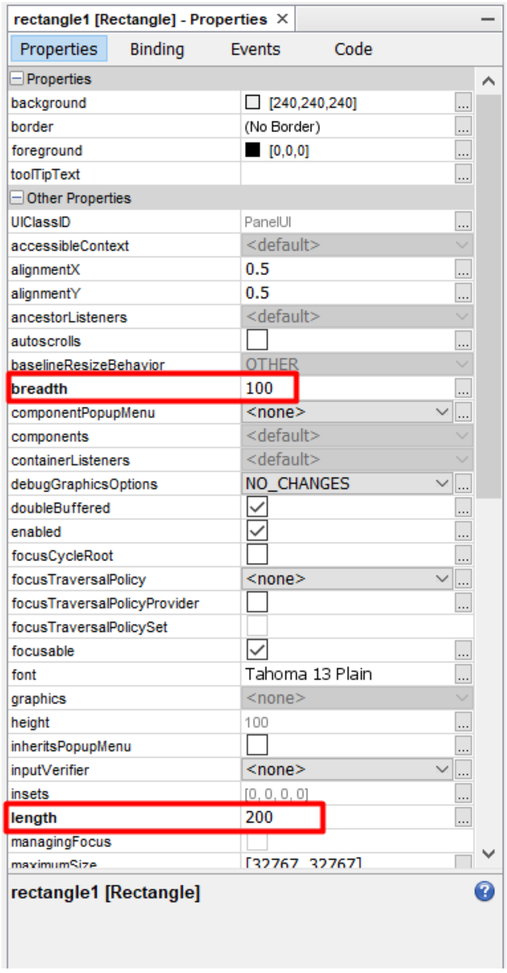
Output:

Now we use Rectangle bean from different package i.e. from ‘UseBeans’



- Assignment_6
  - Source Packages
    - Q01_JavaBean
    - Q02_EllipseBean
    - UseBeans
      - UseEllipseBean.java
      - UseJavaBean.java
    - images
  - Test Packages
  - Libraries
  - Test Libraries





rectangle1 [Rectangle] - Properties ×

Properties	
background	□ [240,240,240]
border	(No Border)
foreground	■ [0,0,0]
toolTipText	
Other Properties	
UIClassID	PanelUI
accessibleContext	<default>
alignmentX	0.5
alignmentY	0.5
ancestorListeners	<default>
autoscrolls	□
baselineResizeBehavior	OTHER
breadth	100
componentPopupMenu	<none>
components	<default>
containerListeners	<default>
debugGraphicsOptions	NO_CHANGES
doubleBuffered	☑
enabled	☑
focusCycleRoot	□
focusTraversalPolicy	<none>
focusTraversalPolicyProvider	□
focusTraversalPolicySet	□
focusable	☑
font	Tahoma 13 Plain
graphics	<none>
height	100
inheritsPopupMenu	□
inputVerifier	<none>
insets	[0, 0, 0, 0]
length	200
managingFocus	□
maximumSize	[32767, 32767]

rectangle1 [Rectangle]

2. Create a custom Java Bean named Ellipse which can be used to draw an ellipse.

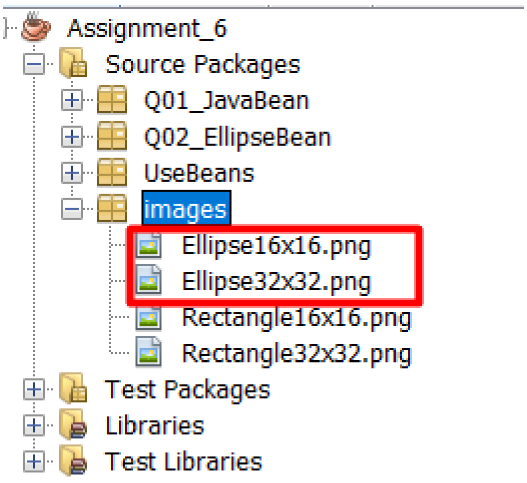
   ⇨ Program

```
package Q02_EllipseBean;

import java.awt.Graphics;
import javax.swing.JPanel;

public class Ellipse extends JPanel
{
    public Ellipse() { }

    private int majorAxis = 0;
    private int minorAxis = 0;

    public int getMinorAxis()
    {
        return minorAxis;
    }

    public void setMinorAxis(int minorAxis)
    {
        this.minorAxis = minorAxis;
        repaint();
    }

    public int getMajorAxis()
    {
        return majorAxis;
    }

    public void setMajorAxis(int majorAxis)
    {
        this.majorAxis = majorAxis;
        repaint();
    }

    @Override
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        if (minorAxis <= 0 || majorAxis<=0) return;

        int cx = getWidth() / 2, cy = getHeight() / 2;
        int x = cx - majorAxis, y = cy - minorAxis;
        int w = 2 * majorAxis, h = 2*minorAxis;
        g.drawOval(x, y, w, h);
    }
}
```

For Ellipse icon

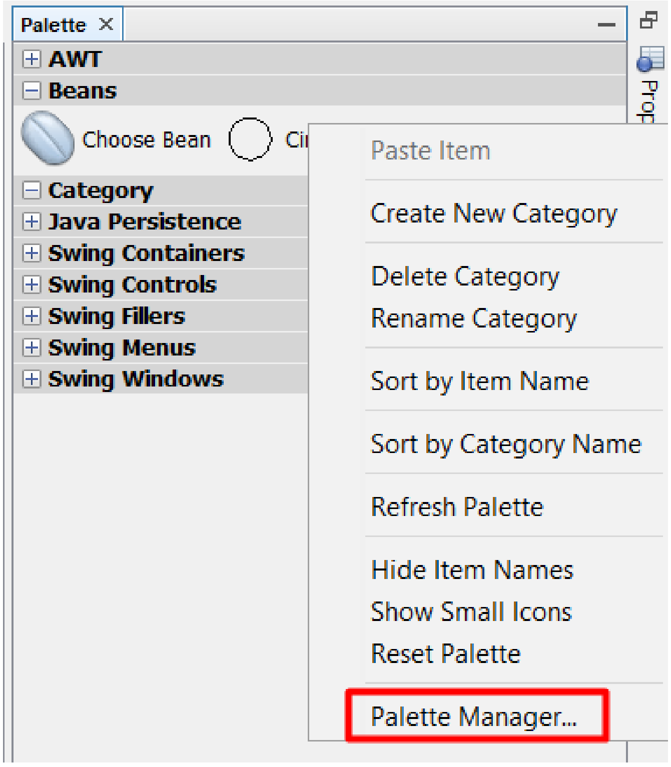
```
package Q02_EllipseBean;

import java.awt.Image;
import java.beans.BeanInfo;
import java.beans.SimpleBeanInfo;

public class EllipseBeanInfo extends SimpleBeanInfo
{
    private final Image iconColor16;
    private final Image iconColor32;
    private final Image iconMono16;
    private final Image iconMono32;

    public EllipseBeanInfo()
    {
        iconColor16 = loadImage("/images/Ellipse16x16.png");
        iconColor32 = loadImage("/images/Ellipse32x32.png");
        iconMono16 = loadImage("/images/Ellipse6x16.png");
        iconMono32 = loadImage("/images/Ellipse32x32.png");
    }

    @Override
    public Image getIcon(int iconType)
     {
        switch (iconType)
        {
            case BeanInfo.ICON_COLOR_16x16:
                return iconColor16;
            case BeanInfo.ICON_COLOR_32x32:
                return iconColor32;
            case BeanInfo.ICON_MONO_16x16:
                return iconMono16;
            case BeanInfo.ICON_MONO_32x32:
                return iconMono32;
            default:
                return null;
        }
    }
}
```

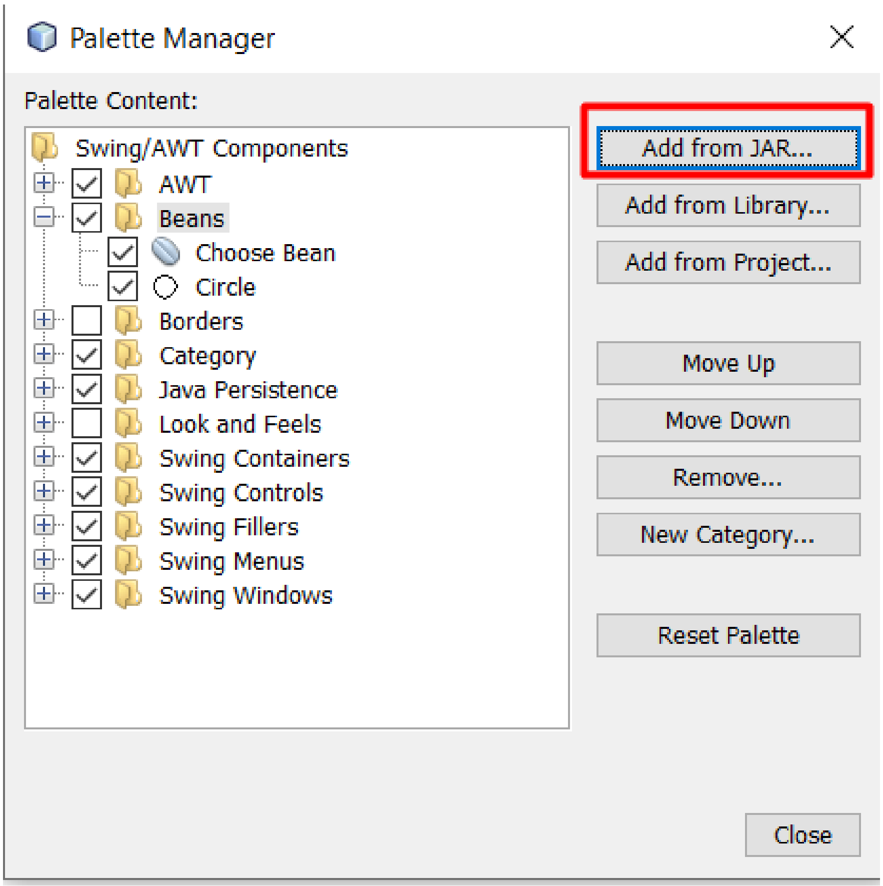
Icon images are kept inside images folder as:

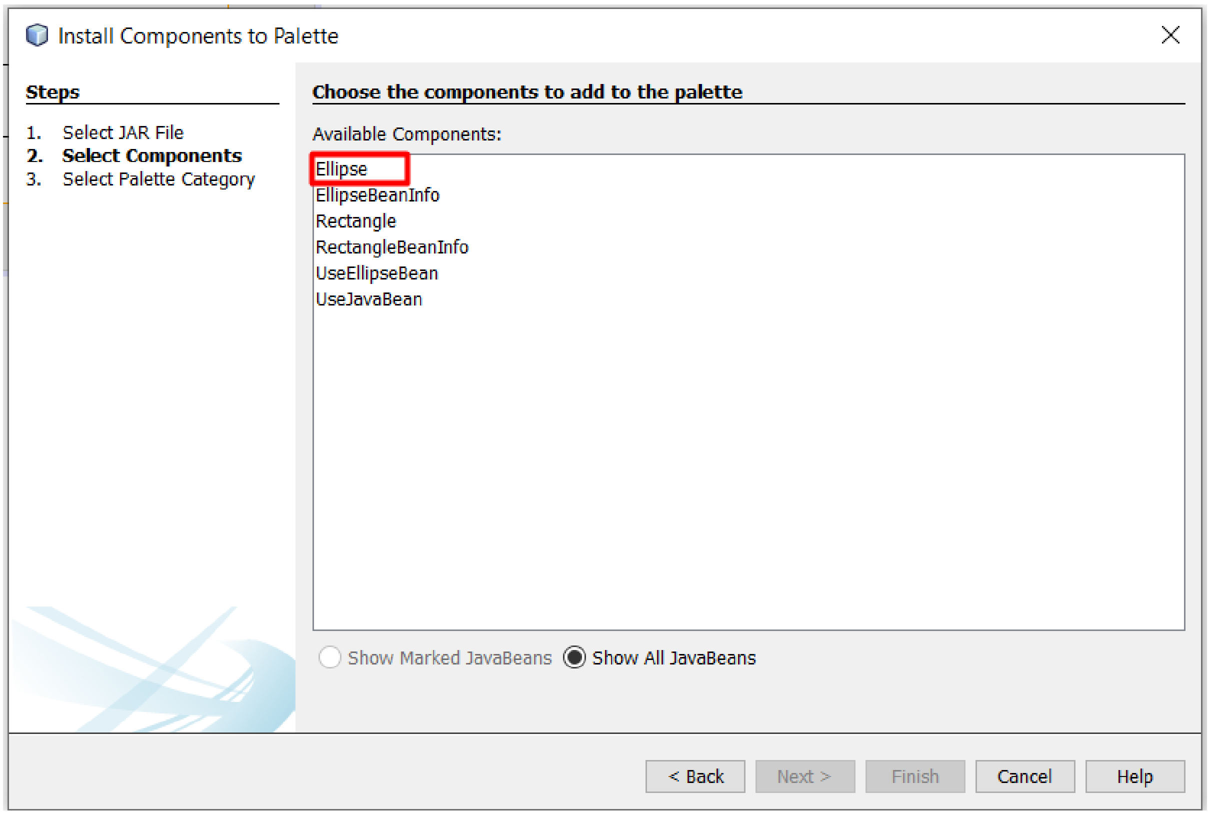


Assignment_6
- Source Packages
  - Q01_JavaBean
  - Q02_EllipseBean
  - UseBeans
  - images
    - Ellipse16x16.png
    - Ellipse32x32.png
    - Rectangle16x16.png
    - Rectangle32x32.png
- Test Packages
- Libraries
- Test Libraries

Adding custom bean i.e. Ellipse to palette Manager







Install Components to Palette

Steps

1. Select JAR File
2. **Select Components**
3. Select Palette Category

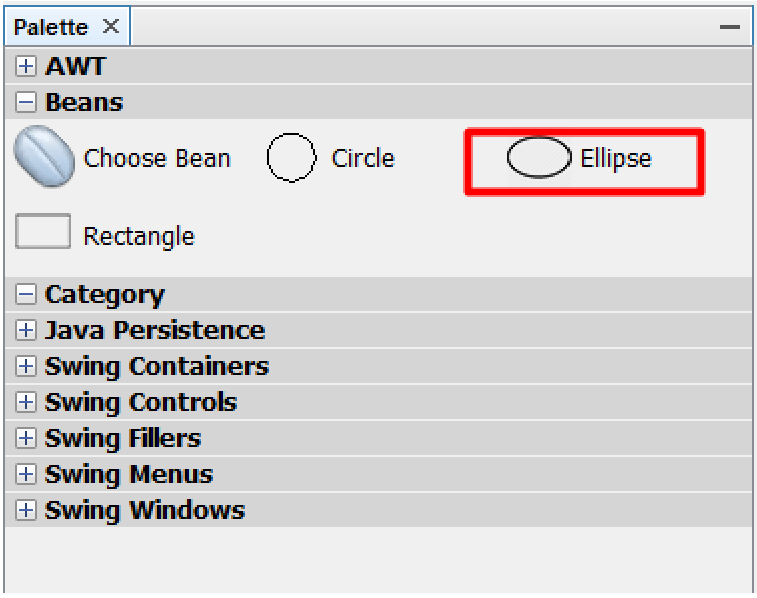
Choose the components to add to the palette

Available Components:

```
Ellipse
EllipseBeanInfo
Rectangle
RectangleBeanInfo
UseEllipseBean
UseJavaBean
```

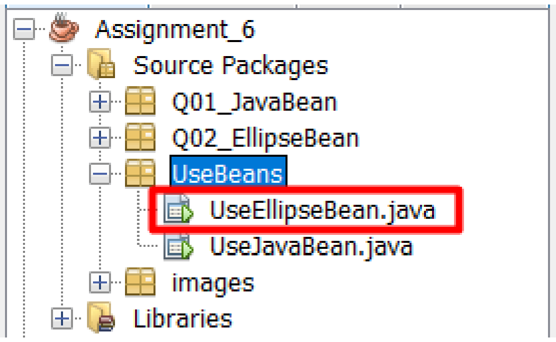
○ Show Marked JavaBeans ● Show All JavaBeans

Custom Ellipse bean is successfully added to the palette manager

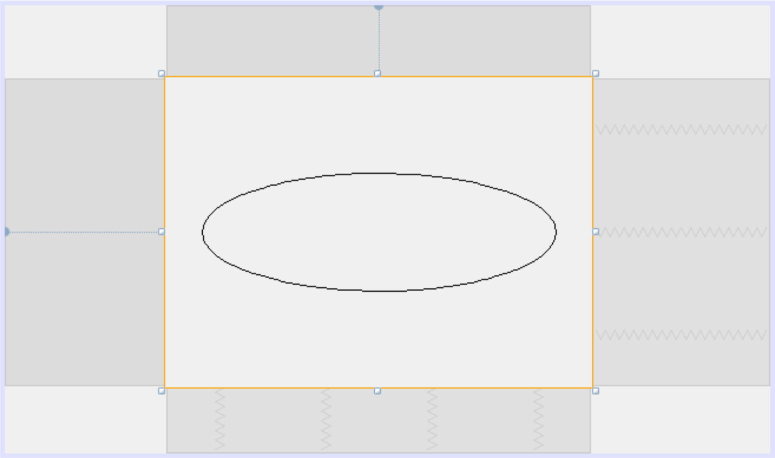


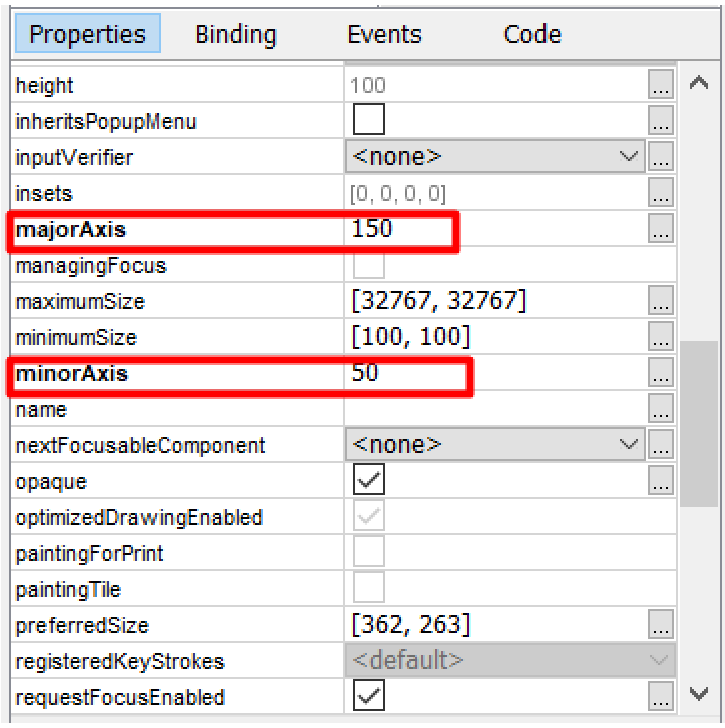
Output:

Now we use Ellipse bean from different package i.e. from 'UseBeans'



- Assignment_6
  - Source Packages
    - Q01_JavaBean
    - Q02_EllipseBean
    - UseBeans
      - UseEllipseBean.java
      - UseJavaBean.java
    - images
  - Libraries





height	100
inheritsPopupMenu	☐
inputVerifier	<none>
insets	[0, 0, 0, 0]
majorAxis	150
managingFocus	☐
maximumSize	[32767, 32767]
minimumSize	[100, 100]
minorAxis	50
name	
nextFocusableComponent	<none>
opaque	☑
optimizedDrawingEnabled	☑
paintingForPrint	☐
paintingTile	☐
preferredSize	[362, 263]
registeredKeyStrokes	<default>
requestFocusEnabled	☑