Bridge Example [Shalloway and Trott, 2002]

Java code

```
class Client {
    public static void main(String[] args) {
      Shape s;
      Drawing d;

      d = new V1Drawing();
      s = new Rectangle(d,1,1,2,2);
      s.draw();

      d = new V2Drawing();
      s = new Circle(d,2,2,3);
      s.draw();
    }
}

abstract class Shape {
    public abstract void draw();

    protected void drawLine(double x1, double y1, double x2, double y2) {
      d.drawLine(x1,y1,x2,y2);
    }
    protected void drawCircle(double x, double y, double r) {
      d.drawCircle(x,y,r);
    }
    protected Shape(Drawing dr) { d = dr; }

    private Drawing d;
}

class Rectangle extends Shape {
    public void draw() {
      drawLine(p1x,p1y,p2x,p1y);
      drawLine(p1x,p1y,p1x,p2y);
      drawLine(p2x,p2y,p2x,p1y);
      drawLine(p2x,p2y,p1x,p2y);
    }
    public Rectangle(Drawing dr, double x1, double y1,
                     double x2, double y2) {
      super(dr); // invokes the superclass constructor
      p1x = x1; p1y = y1;
      p2x = x2; p2y = y2;
    }
    // (p1x,p1y) is the lower left corner
    // (p2x,p2y) is the upper right corner
    private double p1x, p1y, p2x, p2y;
}

class Circle extends Shape {
    public void draw() { drawCircle(x,y,r); }
    public Circle(Drawing dr, double cx, double cy, double rd) {
      super(dr); x = cx; y = cy; r = rd;
    }
```



```
    private double x, y, r;
}
abstract class Drawing {
    public abstract void drawLine(double x1, double y1, double x2, double y2);
    public abstract void drawCircle(double x, double y, double r);
    // "final" means that the values of d1 and d2 cannot be changed
    // after the initializations shown below
    public static final D1 d1 = new D1();
    public static final D2 d2 = new D2();
}

class V1Drawing extends Drawing {
    public void drawLine(double x1, double y1, double x2, double y2) {
      d1.draw_a_line(x1,y1,x2,y2);
    }
    public void drawCircle(double x, double y, double r) {
      d1.draw_a_circle(x,y,r);
    }
}

class V2Drawing extends Drawing {
    public void drawLine(double x1, double y1, double x2, double y2) {
      d2.drawln(x1,x2,y1,y2);
    }
    public void drawCircle(double x, double y, double r) {
      d2.drawcr(x,y,r);
    }
}

class D1 {
    public void draw_a_line(double x1, double y1, double x2, double y2) {
      // some implementation
    }
    public void draw_a_circle(double x, double y, double r) {
      // some implementation
    }
}


class D2 {
    public void drawln(double x1, double x2, double y1, double y2) {
      // some implementation
    }
    public void drawcr(double x, double y, double r) {
      // some implementation
    }
}
```

C++ code

```
void main() {
  Shape *s;
  Drawing *d;

  d = new V1Drawing;
  s = new Rectangle(d,1,1,2,2);
  s->draw();
```

```
  d = new V2Drawing;
  s = new Circle(d,2,2,3);
  s->draw();
}
class Shape {
 public:
   virtual void draw() = 0;
 protected:
   void drawLine(double x1, double y1, double x2, double y2);
   void drawCircle(double x, double y, double r);
   Shape(Drawing *dp);
 private:
   Drawing *d;
};
void Shape::drawLine(double x1, double y1, double x2, double y2)
      { d->drawLine(x1,y1,x2,y2); }
void Shape::drawCircle(double x, double y, double r)
      { d->drawCircle(x,y,r); }
Shape::Shape(Drawing *dp) { d = dp; }



class Rectangle : public Shape {
 public:
   virtual void draw();
   Rectangle(Drawing *dr, double x1, double y1, double x2, double y2);
 private:
   double p1x, p1y, p2x, p2y;
};
void Rectangle::draw() {
  drawLine(p1x,p1y,p1x,p2y);
  drawLine(p1x,p1y,p2x,p1y);
  drawLine(p2x,p2y,p1x,p2y);
  drawLine(p2x,p2y,p2x,p1y);
}
Rectangle::Rectangle(Drawing *dr, double x1, double y1,
                     double x2, double y2) : Shape(dr)
      { p1x = x1; p1y = y1; p2x = x2; p2y = y2; }


class Circle : public Shape {
 public:
   virtual void draw();
   Circle(Drawing *dr, double cx, double cy, double rd);
 private:
   double x, y, r;
};
void Circle::draw() { drawCircle(x,y,r); }
Circle::Circle(Drawing *dr, double cx, double cy, double rd) : Shape(dr)
      { x = cx; y = cy; r = rd; }



class Drawing {
 public:
  virtual void drawLine(double x1, double y1, double x2, double y2) = 0;
```



```
  virtual void drawCircle(double x, double y, double r) = 0;
  static D1 *d1;
  static D2 *d2;
};

D1 *Drawing::d1 = new D1;
D2 *Drawing::d2 = new D2;
class V1Drawing : public Drawing {
 public:
  virtual void drawLine(double x1, double y1, double x2, double y2);
  virtual void drawCircle(double x, double y, double r);
};
void V1Drawing::drawLine(double x1, double y1, double x2, double y2)
      { d1->draw_a_line(x1,y1,x2,y2); }
void V1Drawing::drawCircle(double x, double y, double r)
      { d1->draw_a_circle(x,y,r); }


class V2Drawing : public Drawing {
 public:
  virtual void drawLine(double x1, double y1, double x2, double y2);
  virtual void drawCircle(double x, double y, double r);
};
void V2Drawing::drawLine(double x1, double y1, double x2, double y2) {
      { d2->drawln(x1,x2,y1,y2); }
void V2Drawing::drawCircle(double x, double y, double r) { d2->drawcr(x,y,r); }


class D1 {
 public:
  void draw_a_line(double x1, double y1, double x2, double y2);
  void draw_a_circle(double x, double y, double r);
};
void D1::draw_a_line(double x1, double y1, double x2, double y2) {
  // some implementation
}
void D1::draw_a_circle(double x, double y, double r) {
  // some implementation
}



class D2 {
 public:
  void drawln(double x1, double x2, double y1, double y2);
  void drawcr(double x, double y, double r);
};
void D2::drawln(double x1, double x2, double y1, double y2) {
  // some implementation
}
void D2::drawcr(double x, double y, double r) {
  // some implementation
}
```