



TECNOLOGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE OAXACA

Carrera:

Ingeniería en Sistemas Computacionales

Materia:

Diseño e Implementación de software con patrones

Documentación

Docente:

Espinoza Pérez Jacob

Equipo:

Ordaz Pacheco Ruudvan

Santos Manuel Julia Marlenny

Vera Acevedo Héctor Aramís

Grupo:

7SB

Fecha de entrega:

24/03/2025

Documentación del Patrón Adapter en el Sistema de Ventas

Introducción

El patrón Adapter es un patrón de diseño estructural que permite que dos interfaces incompatibles trabajen juntas. En el código proporcionado, este patrón se utiliza para generar facturas en diferentes formatos (PDF y HTML) sin modificar la lógica principal de la aplicación. A continuación, se describe cómo se implementa el patrón Adapter en el código.

Cambios Realizados

Interfaz FacturaGenerator

La interfaz FacturaGenerator define el contrato que deben cumplir todos los adaptadores para generar facturas. Esta interfaz actúa como el **Target** en el patrón Adapter, ya que es la interfaz que el cliente (InterFacturacion) espera utilizar.

```
import java.util.List;
import modelo.DetalleVenta;

/**
 *
 * @author hecto
 */
public interface FacturaGenerator {
    void generarFactura(String nombreCliente, String cedulaCliente, String telefonoCliente, String direccionCliente,
        String fechaActual, String folioStr, List<DetalleVenta> listaProductos);
}
```

Clase VentaPDF

La clase VentaPDF es el **Adaptee**, es decir, la clase que tiene una interfaz incompatible con la que el cliente necesita. En este caso, VentaPDF tiene un método generarFacturaPDF que no coincide directamente con la interfaz FacturaGenerator.

```
public class VentaPDF {

    private String nombreCliente;
    private String cedulaCliente;
    private String telefonoCliente;
    private String direccionCliente;

    private String fechaActual = "";
    private String nombreArchivoPDFVenta = "";

    Venta venta = new Venta();
    int folioVenta = venta.obtenerYActualizarFolio();
    String folioStr = String.format("%03d", folioVenta); // Formato 001, 002, 003

    //metodo para obtener datos del cliente
    public void DatosCliente(int idCliente) {
        Connection cn = Conexion.conectar();
        String sql = "select * from tb_cliente where idCliente = '" + idCliente + "'";
        Statement st;
        try {

            st = cn.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                nombreCliente = rs.getString("nombre") + " " + rs.getString("apellido");
                cedulaCliente = rs.getString("cedula");
                telefonoCliente = rs.getString("telefono");
                direccionCliente = rs.getString("direccion");
            }
            cn.close();
        } catch (SQLException e) {
            System.out.println("Error al obtener datos del cliente: " + e);
        }
    }
}
```

```

public void generarFacturaPDF() {
    try {

        //cargar la fecha actual
        Date date = new Date();
        fechaActual = new SimpleDateFormat("yyyy/MM/dd").format(date);
        //cambiar el formato de la fecha de / a _
        String fechaNueva = "";
        for (int i = 0; i < fechaActual.length(); i++) {
            if (fechaActual.charAt(i) == '/') {
                fechaNueva = fechaActual.replace("/", "_");
            }
        }

        nombreArchivoPDFVenta = "Venta_" + nombreCliente + "_" + fechaNueva + ".pdf";

        FileOutputStream archivo;
        File file = new File("src/pdf/" + nombreArchivoPDFVenta);
        archivo = new FileOutputStream(file);

        Document doc = new Document();
        PdfWriter.getInstance(doc, archivo);
        doc.open();

        Image img = Image.getInstance("src/img/ventas.png");
        Paragraph fecha = new Paragraph();
        Font negrita = new Font(Font.FontFamily.TIMES_ROMAN, 12, Font.BOLD, BaseColor.BLUE);
        //fecha.add(Chunk.NEWLINE); //agregar nueva linea
        //fecha.add("Factura: 001" + "\nFecha: " + fechaActual + "\n\n");
        fecha.add(Chunk.NEWLINE); //agregar nueva linea
        fecha.add("Nota: " + folioStr + "\nFecha: " + fechaActual + "\n\n");

        PdfPTable Encabezado = new PdfPTable(4);
        Encabezado.setWidthPercentage(100);
        Encabezado.getDefaultCell().setBorder(0); //quitar el borde de la tabla
        //tamaño de las celdas
        float[] ColumnaEncabezado = new float[]{20f, 30f, 70f, 40f};
        Encabezado.setWidths(ColumnaEncabezado);
        Encabezado.setHorizontalAlignment(Element.ALIGN_LEFT);
        //agregar celdas
        Encabezado.addCell(img);
    }
}

```

```

String ruc = "Aun no hay";
String nombre = " Miscelanea Calicanto";
String telefono = "9547894569";
String direccion = " Av Montoya";
String razon = "Apoyando a la economia oaxaqueña";

Encabezado.addCell(""); //celda vacia
Encabezado.addCell("/*RUC: " + ruc + */\nNOMBRE: " + nombre + "\nTELEFONO: " + telefono + "\nDIRECCION: "
Encabezado.addCell(fecha);
doc.add(Encabezado);

//CUERPO
Paragraph cliente = new Paragraph();
cliente.add(Chunk.NEWLINE); //nueva linea
cliente.add("Datos del cliente: " + "\n\n");
doc.add(cliente);

//DATOS DEL CLIENTE
PdfPTable tablaCliente = new PdfPTable(4);
tablaCliente.setWidthPercentage(100);
tablaCliente.getDefaultCell().setBorder(0); //quitar bordes
//tamaño de las celdas
float[] ColumnaCliente = new float[]{25f, 45f, 30f, 40f};
tablaCliente.setWidths(ColumnaCliente);
tablaCliente.setHorizontalAlignment(Element.ALIGN_LEFT);
PdfPCell cliente1 = new PdfPCell(new Phrase("Cedula/RUC: ", negrita));
PdfPCell cliente2 = new PdfPCell(new Phrase("Nombre: ", negrita));
PdfPCell cliente3 = new PdfPCell(new Phrase("Telefono: ", negrita));
PdfPCell cliente4 = new PdfPCell(new Phrase("Direccion: ", negrita));
//quitar bordes
cliente1.setBorder(0);
cliente2.setBorder(0);
cliente3.setBorder(0);
cliente4.setBorder(0);
//agg celda a la tabla
tablaCliente.addCell(cliente1);
tablaCliente.addCell(cliente2);
tablaCliente.addCell(cliente3);
tablaCliente.addCell(cliente4);
tablaCliente.addCell(cedulaCliente);
tablaCliente.addCell(nombreCliente);
tablaCliente.addCell(telefonoCliente);
tablaCliente.addCell(direccionCliente);

```

```

        tablaProducto.addCell(cantidad);
        tablaProducto.addCell(producto);
        tablaProducto.addCell(precio);
        tablaProducto.addCell(total);
    }

    //agregar al documento
    doc.add(tablaProducto);

    //Total pagar
    Paragraph info = new Paragraph();
    info.add(Chunk.NEWLINE);
    info.add("Total a pagar: " + InterFacturacion.txt_total_pagar.getText());
    info.setAlignment(Element.ALIGN_RIGHT);
    doc.add(info);

    /* //Firma
    Paragraph firma = new Paragraph();
    firma.add(Chunk.NEWLINE);
    firma.add("Cancelacion y firma\n\n");
    firma.add("_____");
    firma.setAlignment(Element.ALIGN_CENTER);
    doc.add(firma);*/

    //Mensaje
    Paragraph mensaje = new Paragraph();
    mensaje.add(Chunk.NEWLINE);
    mensaje.add("¡Gracias por su compra!");
    mensaje.setAlignment(Element.ALIGN_CENTER);
    doc.add(mensaje);

    //cerrar el documento y el archivo
    doc.close();
    archivo.close();

    //abrir el documento al ser generado automaticamente
    Desktop.getDesktop().open(file);

} catch (DocumentException | IOException e) {
    System.out.println("Error en: " + e);
}
}

```

Clase VentaPDFAdapter (Adapter)

La clase VentaPDFAdapter es el **Adapter** que implementa la interfaz FacturaGenerator y adapta la clase VentaPDF para que pueda ser utilizada por el cliente. Esta clase actúa como un puente entre FacturaGenerator y VentaPDF.

```
public class VentaPDFAdapter implements FacturaGenerator {
    private VentaPDF ventaPDF;

    public VentaPDFAdapter(VentaPDF ventaPDF) {
        this.ventaPDF = ventaPDF;
    }

    private int obtenerIdClientePorCedula(String cedulaCliente) {
        int idCliente = -1;
        Connection cn = Conexion.conectar();
        String sql = "select idCliente from tb_cliente where cedula = '" + cedulaCliente + "'";
        Statement st;
        try {
            st = cn.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                idCliente = rs.getInt("idCliente"); // Obtener el ID del cliente
            }
            cn.close();
        } catch (SQLException e) {
            System.out.println("Error al obtener ID del cliente por cédula: " + e);
        }
        return idCliente;
    }

    @Override
    public void generarFactura(String nombreCliente, String cedulaCliente, String telefonoCliente, String direccionCliente,
        int idCliente = obtenerIdClientePorCedula(cedulaCliente);

    if (idCliente != -1) {
        ventaPDF.DatosCliente(idCliente); // Pasar el ID del cliente
        ventaPDF.generarFacturaPDF();
    } else {
        System.out.println("Error: No se encontró el cliente con cédula " + cedulaCliente);
    }
}
}
```

Clase VentaHTMLAdapter (Adapter)

La clase VentaHTMLAdapter es otro **Adapter** que implementa la interfaz FacturaGenerator para generar facturas en formato HTML. A diferencia de VentaPDFAdapter, esta clase no necesita adaptar una clase existente, ya que implementa directamente la lógica para generar el archivo HTML.

```
public class VentaHTMLAdapter implements FacturaGenerator {  
  
    @Override  
    public void generarFactura(String nombreCliente, String cedulaCliente, String telefonoCliente, String direccionCliente, String fechaActual, String folioStr)  
    {  
        // Crear el contenido HTML  
        StringBuilder contenidoHTML = new StringBuilder();  
        contenidoHTML.append("<!DOCTYPE html>\n");  
        contenidoHTML.append("<html lang='es'>\n");  
        contenidoHTML.append("<head>\n");  
        contenidoHTML.append("    <meta charset='UTF-8'>\n");  
        contenidoHTML.append("    <meta name='viewport' content='width=device-width, initial-scale=1.0'>\n");  
        contenidoHTML.append("    <title>Factura de Venta</title>\n");  
        contenidoHTML.append("    <style>\n");  
        contenidoHTML.append("        body { font-family: Arial, sans-serif; }\n");  
        contenidoHTML.append("        h1 { color: #333; }\n");  
        contenidoHTML.append("        table { width: 100%; border-collapse: collapse; margin-top: 20px; }\n");  
        contenidoHTML.append("        th, td { border: 1px solid #ddd; padding: 8px; text-align: left; }\n");  
        contenidoHTML.append("        th { background-color: #f2f2f2; }\n");  
        contenidoHTML.append("    </style>\n");  
        contenidoHTML.append("</head>\n");  
        contenidoHTML.append("<body>\n");  
        contenidoHTML.append("    <h1>Factura de Venta</h1>\n");  
        contenidoHTML.append("    <p><strong>Cliente:</strong> ").append(nombreCliente).append("</p>\n");  
        contenidoHTML.append("    <p><strong>Cédula:</strong> ").append(cedulaCliente).append("</p>\n");  
        contenidoHTML.append("    <p><strong>Teléfono:</strong> ").append(telefonoCliente).append("</p>\n");  
        contenidoHTML.append("    <p><strong>Dirección:</strong> ").append(direccionCliente).append("</p>\n");  
        contenidoHTML.append("    <p><strong>Fecha:</strong> ").append(fechaActual).append("</p>\n");  
        contenidoHTML.append("    <p><strong>Folio:</strong> ").append(folioStr).append("</p>\n");  
        contenidoHTML.append("    <table>\n");  
        contenidoHTML.append("        <thead>\n");  
        contenidoHTML.append("            <tr>\n");  
        contenidoHTML.append("                <th>Producto</th>\n");  
        contenidoHTML.append("                <th>Cantidad</th>\n");  
        contenidoHTML.append("                <th>Precio Unitario</th>\n");  
        contenidoHTML.append("                <th>Subtotal</th>\n");  
        contenidoHTML.append("            </tr>\n");  
        contenidoHTML.append("        </thead>\n");  
        contenidoHTML.append("    </table>\n");  
    }
```



```

// Agregar productos dinámicamente
double totalPagar = 0.0;
for (DetalleVenta producto : listaProductos) {
    contenidoHTML.append("        <tr>\n")
        .append("            <td>").append(producto.getNombre()).append("</td>\n")
        .append("            <td>").append(producto.getCantidad()).append("</td>\n")
        .append("            <td>$").append(producto.getPrecioUnitario()).append("</td>\n")
        .append("            <td>$").append(producto.getSubTotal()).append("</td>\n")
        .append("        </tr>\n");
    totalPagar += producto.getSubTotal();
}

contenidoHTML.append("    </tbody>\n")
    .append(" </table>\n")
    .append(" <p><strong>Total a Pagar:</strong> $").append(totalPagar).append("</p>\n")
    .append("</body>\n")
    .append("</html>");

// Guardar el contenido HTML en un archivo
String nombreArchivo = "Factura_" + nombreCliente.replace(" ", "_") + "_" + fechaActual.replace("/", "_") + ".html";
File archivo = new File("src/html/" + nombreArchivo);

try (BufferedWriter writer = new BufferedWriter(new FileWriter(archivo))) {
    writer.write(contenidoHTML.toString());
    System.out.println("Factura HTML generada: " + archivo.getAbsolutePath());

    // Abrir el archivo HTML en el navegador predeterminado
    java.awt.Desktop.getDesktop().open(archivo);
} catch (IOException e) {
    System.out.println("Error al generar la factura HTML: " + e.getMessage());
}
}

```

Clase InterFacturacion (Client)

La clase InterFacturacion es el **Cliente** que utiliza la interfaz FacturaGenerator para generar facturas en diferentes formatos. Esta clase no necesita conocer los detalles de implementación de los adaptadores, ya que trabaja directamente con la interfaz FacturaGenerator.

```
private void jButton_RegistrarVentaActionPerformed(java.awt.event.ActionEvent evt) {  
  
    CabeceraVenta cabeceraVenta = new CabeceraVenta();  
    DetalleVenta detalleVenta = new DetalleVenta();  
    Ctrl_RegistrarVenta controlVenta = new Ctrl_RegistrarVenta();  
  
    String fechaActual = "";  
    Date date = new Date();  
    fechaActual = new SimpleDateFormat("yyyy/MM/dd").format(date);  
  
    if (!jComboBox_cliente.getSelectedItem().equals("Seleccione cliente:")) {  
        if (listaProductos.size() > 0) {  
  
            //metodo para obtener el id del cliente  
            this.ObtenerIdCliente();  
            //registrar cabecera  
            cabeceraVenta.setIdCabeceraVenta(0);  
            cabeceraVenta.setIdCliente(idCliente);  
            cabeceraVenta.setValorPagar(Double.parseDouble(txt_total_pagar.getText()));  
            cabeceraVenta.setFechaVenta(fechaActual);  
            cabeceraVenta.setEstado(1);  
  
            if (controlVenta.guardar(cabeceraVenta)) {  
                JOptionPane.showMessageDialog(null, "¡Venta Registrada!");  
            }  
            // Obtener los datos del cliente  
            Ctrl_Cliente ctrlCliente = new Ctrl_Cliente();  
            Cliente cliente = ctrlCliente.obtenerClientePorId(idCliente);  
  
            // Obtener el formato seleccionado  
            String formato = cmbFormato.getSelectedItem().toString();  
  
            // Generar la factura de venta usando el Adapter correspondiente  
            if (formato.equals("PDF")) {  
                facturaGenerator = new VentaPDFAdapter(new VentaPDF());  
            } else if (formato.equals("HTML")) {  
                facturaGenerator = new VentaHTMLAdapter();  
            }  
  
            facturaGenerator.generarFactura(  
                cliente.getNombre() + " " + cliente.getApellido(), // Nombre del cliente  
                cliente.getCedula(), // Cédula del cliente  
                cliente.getTelefono(), // Teléfono del cliente  
                cliente.getDireccion(), // Dirección del cliente  
                fechaActual, // Fecha de la venta  
                "001", // Folio de la venta (puedes obtenerlo de tu lógica actual)  
                listaProductos // Lista de productos  
            );  
        }  
    }  
}
```

```

//guardar detalle
for (DetalleVenta elemento : listaProductos) {
    detalleVenta.setIdDetalleVenta(0);
    detalleVenta.setIdCabeceraVenta(0);
    detalleVenta.setIdProducto(elemento.getIdProducto());
    detalleVenta.setCantidad(elemento.getCantidad());
    detalleVenta.setPrecioUnitario(elemento.getPrecioUnitario());
    detalleVenta.setSubTotal(elemento.getSubTotal());
    detalleVenta.setDescuento(elemento.getDescuento());
    detalleVenta.setIva(elemento.getIva());
    detalleVenta.setTotalPagar(elemento.getTotalPagar());
    detalleVenta.setEstado(1);

    if (controlVenta.guardarDetalle(detalleVenta)) {
        //System.out.println("Detalle de Venta Registrado");

        txt_subtotal.setText("0.0");
        txt_iva.setText("0.0");
        txt_descuento.setText("0.0");
        txt_total_pagar.setText("0.0");
        txt_efectivo.setText("");
        txt_cambio.setText("0.0");
        auxIdDetalle = 1;

        this.CargarComboClientes();
        this.RestarStockProductos(elemento.getIdProducto(), elemento.getCantidad());

    } else {
        JOptionPane.showMessageDialog(null, "¡Error al guardar detalle de venta!");
    }
}
//vaciamos la lista
listaProductos.clear();
listaTablaProductos();

} else {
    JOptionPane.showMessageDialog(null, "¡Error al guardar cabecera de venta!");
}
} else {
    JOptionPane.showMessageDialog(null, "¡Seleccione un producto!");
}
} else {
    JOptionPane.showMessageDialog(null, "¡Seleccione un cliente!");
}
}

```

Ventajas del Patrón Adapter en este Código

- **Flexibilidad:** Permite añadir nuevos formatos de facturación (por ejemplo, XML o JSON) sin modificar el código del cliente (InterFacturacion).
- **Reutilización:** Permite reutilizar la clase VentaPDF sin necesidad de modificarla.
- **Desacoplamiento:** El cliente no necesita conocer los detalles de implementación de los adaptadores, lo que facilita el mantenimiento y la extensión del código.

Resultados de la implementación del patrón adapter.

Factura en formato PDF

Facturación

Ciente: Teresa Hernandez

Buscar

Seleccione producto:

Cantidad:

Añadir Productos

N	Nombre	Cantidad	P. Unitario	SubTotal	Descuento	Iva	Total Pagar	Accion
1	Cerveza	2	25.0	50.0	0.0	0.0	50.0	Eliminar
2	Sabritas Cl...	3	18.0	54.0	0.0	0.0	54.0	Eliminar
3	Tostadas ...	1	38.0	38.0	0.0	0.0	38.0	Eliminar

Formato de Reporte

PDF



Registrar Venta

Subtotal:

142.0

Descuento:

0.0

Iva:

0.0

Total a pagar:

142.0

Efectivo:

200

Cambio:

58.0

Calcular Cambio

Facturación

Ciente:

Teresa Hernandez

Buscar

Seleccione producto:

Cantidad:

Añadir Productos

N	Nombre	Cantidad	P. Unitario	SubTotal	Descuento	Iva	Total Pagar	Accion
1	Cerveza	2	25.0	50.0	0.0	0.0	50.0	Eliminar
2	Sabritas Cl...	3	18.0	54.0	0.0	0.0	54.0	Eliminar
3	Tostadas ...	1				0.0	38.0	Eliminar

Formato de Reporte

PDF



Subtotal:

142.0

Descuento:

0.0

Iva:

0.0

Total a pagar:

142.0

Efectivo:

200

Cambio:

58.0

Calcular Cambio

Message



¡Venta Registrada!

OK

1 / 1 | — 100% + | 🖨️ 🔗



NOMBRE: Miscelanea Calicanto
TELEFONO: 9547894569
DIRECCION: Av Montoya
RAZON SOCIAL: Apoyando a la
economia oaxaqueña

Nota: 017
Fecha: 2025/03/22

Datos del cliente:

Cedula/RUC:	Nombre:	Telefono:	Direccion:
9581732455	Teresa Hernandez	9581732455	Privada Texcoco

Cantidad:	Descripcion:	Precio Unitario:	Precio Total:
2	Cerveza	25.0	50.0
3	Sabritas Clasicas	18.0	54.0
1	Tostadas Mi Reyna	38.0	38.0

Total a pagar: 142.0

¡Gracias por su compra!

Factura en formato HTML

Factura de Venta

Cliente: Teresa Hernandez

Cédula: 9581732455

Teléfono: 9581732455

Dirección: Privada Texcoco

Fecha: 2025/03/22

Folio: 001

Producto	Cantidad	Precio Unitario	Subtotal
Cerveza	2	\$25.0	\$50.0
Sabritas Clasicas	3	\$18.0	\$54.0
Tostadas Mi Reyna	1	\$38.0	\$38.0

Total a Pagar: \$142.0

Conclusión

El patrón Adapter se utiliza en este código para generar facturas en diferentes formatos (PDF y HTML) de manera flexible y extensible. Gracias a este patrón, el cliente (InterFacturacion) puede trabajar con diferentes formatos de facturación sin necesidad de modificar su código, lo que facilita la adición de nuevos formatos en el futuro.