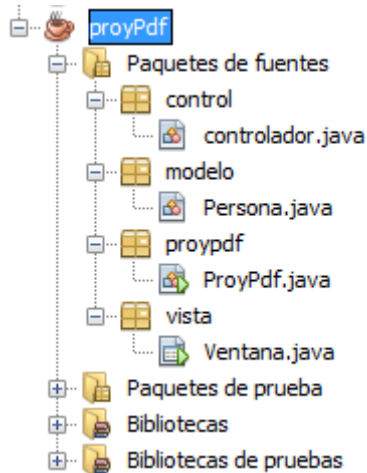


CREAR UN ARCHIVO PDF CON JAVA

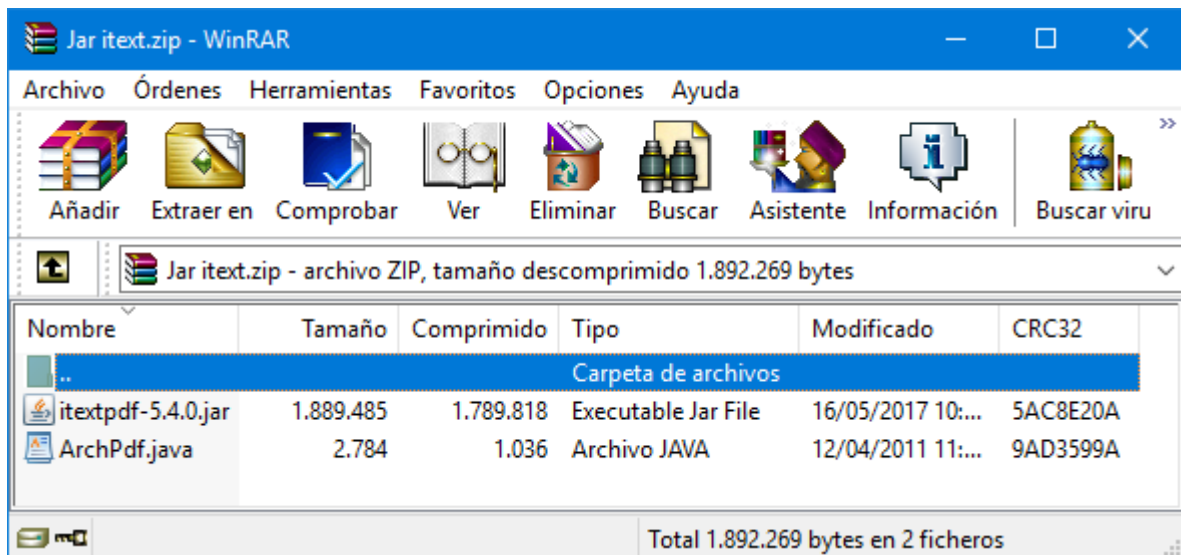
Existen varias formas de manejar y generar archivos pdf en java, en internet se pueden encontrar clases que pueden reutilizarse para tal fin. En este caso se utilizará iText- 5.4.11 que se puede descargar de la siguiente dirección:

<https://github.com/itext/itextpdf/releases/tag/5.5.11>

1. Crear un proyecto en Netbeans con la siguiente estructura:



2. Copie el contenido de la carpeta **jariText** al proyecto de la siguiente manera:

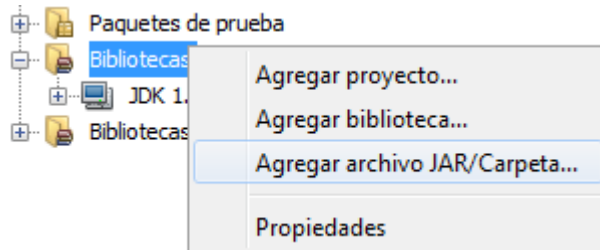


El archivo más importante es el **iTextpdf-5.4.0.jar** ya que sin este contiene las clases y funciones necesarias para generar el archivo pdf, ubíquelo en la raíz del proyecto.

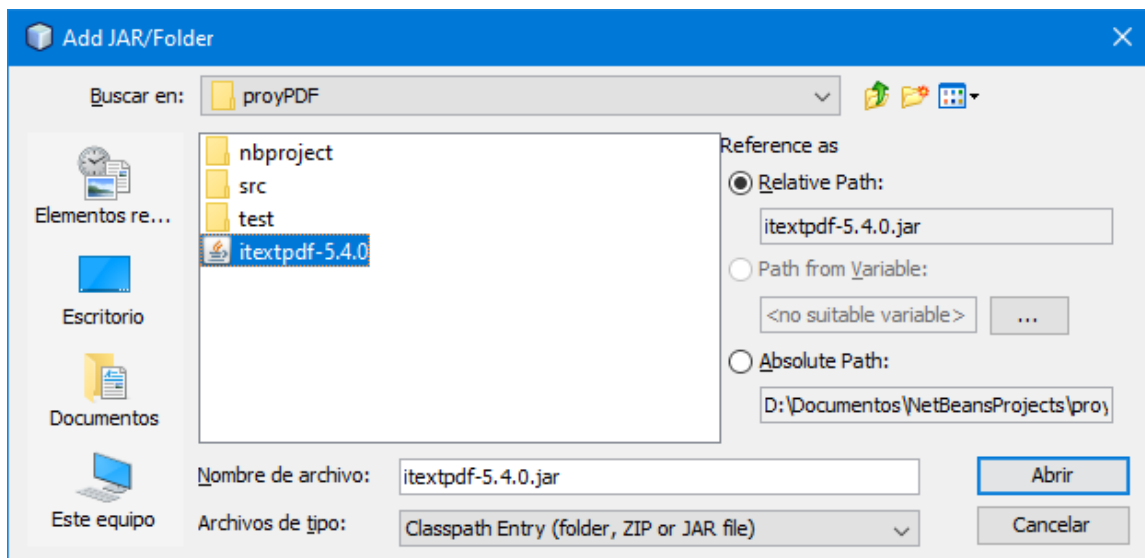
La clase **ArchPdf.java** colóquela en el paquete **modelo**.

3. Una vez copiado al archivo .jar es necesario agregarlo al proyecto en NetBeans IDE

, para ello se debe abrir el proyecto, luego se debe hacer clic derecho en la carpeta **Bibliotecas (Libraries)** y se seleccione la opción **Agregar archivo JAR/Carpeta (Add JAR/Folder)**.



4. En la ventana que aparece se debe ubicar el archivo .jar de iText que previamente debe estar guardado en el directorio del proyecto que se ha creado en Netbeans.



Luego de seleccionar el archivo **iText-5.0.6.jar** y luego hacer clic en el botón **Abrir**

De esta forma se agregarán las clases que contienen los métodos necesarios para generar y manipular los datos de un archivo pdf.

5. Cree una clase **Persona** con los atributos, Id, nombre y teléfono, agregue los constructores y métodos necesarios para usarla.

```

1  package logica;
2
3  public class Persona {
4      private String id, nom,tel;
5      private int edad;
6  + public Persona(String id, String nom, String tel, int edad)
12 + public Persona() {...6 lines }
18 + public String getId() {...3 lines }
21 + public void setId(String id) {...3 lines }
24 + public int getEdad() {...3 lines }
27 + public void setEdad(int edad) {...3 lines }
30 + public String getNom() {...3 lines }
33 + public void setNom(String nom) {...3 lines }
36 + public String getTel() {...3 lines }
39 + public void setTel(String tel) {...3 lines }
42 @Override
+ public String toString() {...6 lines }
49 }

```

6. Cree un formulario (Ventana) que contenga los siguientes elementos y asigne los nombres como sigue:

Control	Nombre	Descripción
TextField1	txtId	Campo de texto para ingresar identificación
TextField2	txtNom	Campo de texto para ingresar Nombre
TextField3	txtTel	Campo de texto para ingresar teléfono
Button1	btnReg	Registrar : guarda datos en el objeto persona
Button2	btnPdf	Crear Pdf: genera archivo pdf

El Diseño de interfaz quedará como se observa en la siguiente figura.

The image shows a Java Swing window titled "Generar PDF". Inside the window, there are three text input fields arranged vertically. The first field is labeled "Identificación:", the second is labeled "Nombre:", and the third is labeled "Teléfono:". Below these fields, there are two buttons: "Registrar" and "Generar Pdf". The window has a standard title bar with minimize, maximize, and close buttons.

7. En el archivo ArchPdf que se descargó se pueden observar los siguientes paquetes que están en el archivo .jar que se ha adicionado en la biblioteca.

```

3  import com.itextpdf.text.Document;
9  import com.itextpdf.text.DocumentException;
0  import com.itextpdf.text.Paragraph;
1  import com.itextpdf.text.pdf.PdfWriter;
2  //archivos
3  import java.io.File;
4  import java.io.FileNotFoundException;
5  import java.io.FileOutputStream;
5  import java.util.logging.Level;
7  import java.util.logging.Logger;
8  //librerías ajenas a itext
9  import javax.swing.JFileChooser;
0  import javax.swing.JOptionPane;
1  import javax.swing.filechooser.FileNameExtensionFilter;
2  import sun.tools.jar.Main;

```

El único atributo de la clase ArchPdf es un objeto de tipo archivo **File** con el cual se establece la ubicación donde se almacenará el archivo, iniciándolo con la siguiente instrucción:

File ruta_destino=null;

```

21  import javax.swing.filechooser.FileNameExtensionFilter;
22  import sun.tools.jar.Main;
23  /**...*/
27  public class ArchPdf {
28
29      private File ruta_destino=null;
30
31

```

Para obtener la ubicación de este archivo pdf se ha implementado el método **Colocar_Destino()** que utiliza la clase **JFileChooser** para crear el objeto **fileChooser** que genera un cuadro de dialogo de búsqueda de archivos, el cual visualizara el usuario para colocar el nombre y escoger la ruta del archivo pdf que se va a guardar, como se observa en la siguiente imagen.

```

public void Colocar_Destino(){
    FileNameExtensionFilter filter = new FileNameExtensionFilter("Archivo PDF","pdf","PDF");
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setFileFilter(filter);
    int result = fileChooser.showSaveDialog(null);
    if ( result == JFileChooser.APPROVE_OPTION ){
        this.ruta_destino = fileChooser.getSelectedFile().getAbsolutePath();
    }
}

```

8. El código del método **crear_PDF()** permite agregar varios datos en el archivo pdf, estos datos son cadenas de texto, en el ejemplo que se observa en la imagen corresponde a las variables t, p y m, se pueden ampliar o reducir estos parámetros, dado que solo se agregan con el método **add()** del objeto **mipdf**, por ejemplo:

mipdf.add() (new Paragraph(p+"\n"));

```
public void crear_PDF(String t, String p, String m){
//abre ventana de dialogo "guardar"
    Colocar_Destino();
    //si destino es diferente de null
    if(this.ruta_destino!=null){
        try {
            // se crea instancia del documento
            Document mipdf = new Document();
            // se establece una instancia a un documento pdf
            PdfWriter.getInstance(mipdf, new FileOutputStream(this.ruta_destino + ".pdf"));
            mipdf.open();// se abre el documento
            mipdf.addTitle(t); // se añade el titulo
            /*mipdf.addAuthor(a); // se añade el autor del documento
            mipdf.addSubject(s); //se añade el asunto del documento
            mipdf.addKeywords(k); //Se agregan palabras claves*/
            mipdf.add(new Paragraph(p+"\n"));
            mipdf.add(new Paragraph(m+"\n"));
            // se añade el contenido del PDF
            mipdf.close(); //se cierra el PDF&
            JOptionPane.showMessageDialog(null,"Documento PDF creado");
        } catch (DocumentException ex) {
            Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
        } catch (FileNotFoundException ex) {
            Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

También se pueden agregar nuevas características o metadatos al documento tales como el Título, el autor, un asunto y palabras clave, las cuales aparecen entre comentarios.

Modifique el código del método **crearPdf** tal como se observa en la imagen:

```
public void crear_PDF(Persona per){
    //abre ventana de dialogo "guardar"
    Colocar_Destino();
    //si destino es diferente de null
    if(this.ruta_destino!=null){
        try {
            // se crea instancia del documento
            Document mipdf = new Document();
            // se establece una instancia a un documento pdf
            PdfWriter.getInstance(mipdf, new FileOutputStream(this.ruta_destino + ".pdf"));
            mipdf.open(); // se abre el documento
            mipdf.addTitle("Datos Contacto"); // se añade el titulo
            /*mipdf.addAuthor(a); // se añade el autor del documento
            mipdf.addSubject(s); //se añade el asunto del documento
            mipdf.addKeywords(k); //Se agregan palabras claves*/
            mipdf.add(new Paragraph("Datos Persona " + per.toString()+"\n"));
            mipdf.close(); //se cierra el PDFs
            JOptionPane.showMessageDialog(null,"Documento PDF creado");
        } catch (DocumentException ex) {
            JOptionPane.showMessageDialog(null,"Error creando Documentoc");
        } catch (FileNotFoundException ex) {
            JOptionPane.showMessageDialog(null,"Error creando Documentoc");
        }
    }
}
```

Observe que se está recibiendo el objeto **Persona** completo, con el cual se podrán obtener los datos que se ingresaron desde el formulario utilizando el método **toString**, aunque también puede hacer uso de los métodos **get** de cada atributo si lo requiere.

- En la clase **Controlador**, cree los objetos de las clases **Persona**, **Ventana** y **ArchPdf**, genere los constructores y adicione los Listener como se observa en el siguiente código.

```
package control;
import Logica.ArchPdf;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JOptionPane;
import modelo.Persona;
import vista.Ventana;
public class Controlador implements ActionListener{
    Ventana frmV;
    Persona objP;
    ArchPdf aPdf;
    public Controlador(Persona objP, Ventana objV, ArchPdf aPdf) {...7 lines }
    public Controlador() {
        this.objP= null;
        this.frmV = new Ventana();
        this.aPdf= new ArchPdf();
        frmV.getBtnReg().addActionListener(this);
        frmV.getBtnPdf().addActionListener(this);
    }
}
```

Agregue el siguiente código en el método **actionPerformed()**, para crear el objeto persona cuando se haga clic en el botón **Registrar** y para generar el archivo pdf cuando se haga clic en el botón **Crear Pdf**.

```
@Override
public void actionPerformed(ActionEvent e) {
    if( e.getSource() == frmV.getBtnReg() ) {
        Persona contacto= new Persona();
        contacto.setNom(frmV.getTxtNom().getText());
        contacto.setTel(frmV.getTxtTel().getText());
        contacto.setId(frmV.getTxtId().getText());
        objP= contacto;
        JOptionPane.showMessageDialog(frmV,
            "Se han registrado los siguientes datos "+ objP.toString());
    }
    if(e.getSource() == frmV.getBtnPdf() ) {
        if(objP != null){
            aPdf.crear_PDF(objP);
        }
        else
        { JOptionPane.showMessageDialog(frmV,"No se ha registrado persona...");
        }
    }
}
```

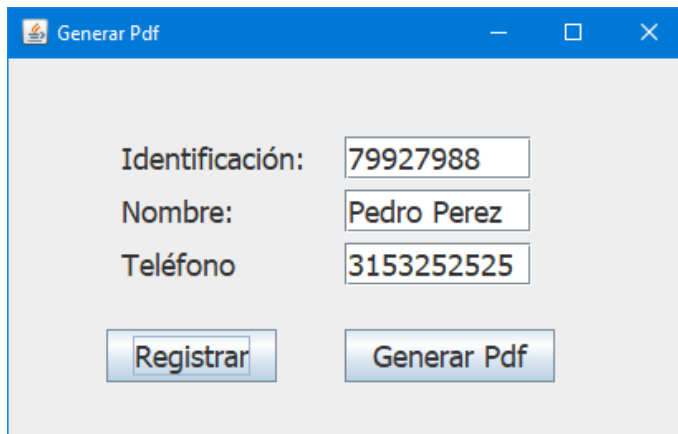
Finalmente cree el método **iniciar** como se observa en la siguiente imagen.

```
public void iniciar(){
    frmV.setTitle("Generar Pdf");
    frmV.setVisible(true);
}
```

- En el archivo principal del proyecto cree el objeto controlador **objC** y adicione la instrucción que ejecuta el método **iniciar()**.

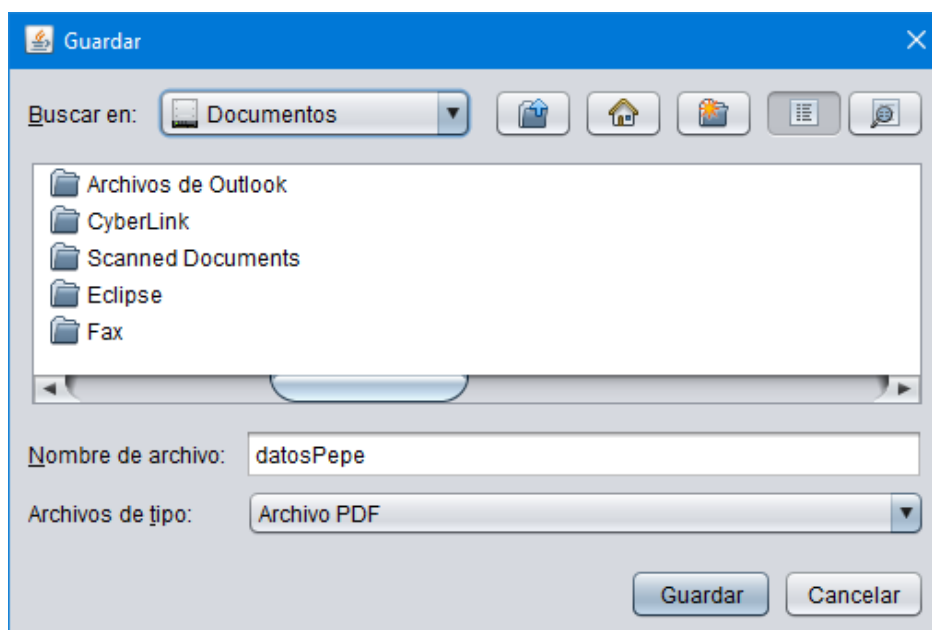
```
public class ProyPdf {
    /**...3 lines */
    public static void main(String[] args) {
        Controlador objC= new Controlador();
        objC.iniciar();
    }
}
```

11. Ejecute el formulario e ingrese datos por ejemplo como se muestra en la imagen.

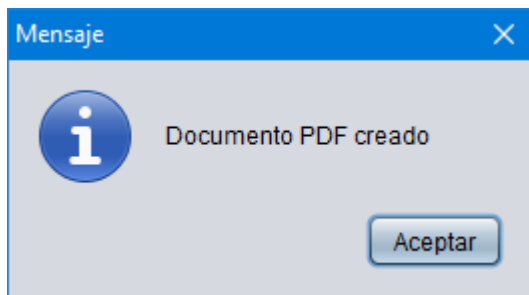


A screenshot of a Java Swing window titled "Generar Pdf". The window has a light gray background and a blue title bar. It contains three text input fields with labels to their left: "Identificación:" with the value "79927988", "Nombre:" with the value "Pedro Perez", and "Teléfono" with the value "3153252525". Below these fields are two buttons: "Registrar" and "Generar Pdf".

Al realizar esta acción se abrirá el cuadro de dialogo Guardar archivo en el cual se selecciona la ruta y se escribe el nombre del archivo pdf.



La aplicación genera el siguiente mensaje indicando que el archivo ya se ha creado



Para terminar se busca el archivo en la ruta especificada y se abre para visualizar su contenido, aunque también se puede agregar el código que realice dicha acción en java.

