¿Cómo se puede mandar la información de una ventana a la impresora?

A continuación se presenta un ejemplo de uso de la impresora. El ejemplo consiste en añadir una opción de menú al `JFrameContador` (`jMenuItemImprimir`) cuya acción asociada (`jMenuItemImprimir_actionPerformed`) consiste en abrir el diálogo de impresión e imprimir la ventana del contador.

Para realizar esta función utilizamos la clase `java.awt.print.PrinterJob` que controla la impresión y la interfaz `Printable` (contenida en el mismo paquete) Cualquier componente que deseemos imprimir tiene que implementar la interfaz `Printable`., en cuya definición se encuentra un único método:

```
int print(Graphics graphics, PageFormat pageFormat, int pageIndex)
```

Por tanto la clase `JFrameContador`, que es lo que deseamos imprimir, debe implementar la interfaz `Printable`, y en consecuencia implementar el método `print` que se define en dicha interfaz.

Para comenzar la impresión se debe obtener un objeto de la clase `PrinterJob` para que controle la impresión, a través del método static `getPrinterJob`. Entre los métodos que proporciona la clase `PrinterJob` encontramos:

`void` **`cancel`**`()`	Cancela un trabajo de impresión que está en marcha
`FormatPage` **`defaultPage`**`()`	Crea una nueva instancia de PageFormat y establece la orientación y tamaño predeterminados.
`PageFormat` **`pageDialog`**`(PageFormat page)`	Muestra un diálogo que permite modificar la instancia PageFormat.
`void` **`print`**`()`	Imprime un conjunto de páginas
`boolean` **`printDialog`**`()`	Presenta un diálogo para cambiar las propiedades del trabajo de impresión.
`void` **`setPrintable`**`(Printable painter, PageFormat format)`	Formatea las páginas que se van a imprimir.

El código que debemos añadir a la clase JFrameContador es el que se presenta a continuación:

```
…
import java.awt.print.*;
…
public class JFrameContador extends JFrame implements Printable
…
void jMenuItemImprimir_actionPerformed(ActionEvent e){
    try{
      mandarImprimir();
    }catch (PrinterException prte){
      JOptionPane.showMessageDialog(null, "Error de impresión.");
    }
  }
```

```
  //Métodos relacionados con la impresión--------------------------------------

  public void mandarImprimir() throws PrinterException {
    // Obtener el objeto PrinterJob
    PrinterJob job = PrinterJob.getPrinterJob();
    // Obtener el formato de pagina. Se abre un dialogo con el formato por
    // defecto
    PageFormat format = job.pageDialog(job.defaultPage());
    // Decimos al PrinterJob qué queremos imprimir (this = vtnaContador)
    job.setPrintable(this, format);
    // Abrimos el dialogo de impresión y le pedimos al usuario que lo confirme
    if (job.printDialog())
      //iniciamos el proceso de impresion (print es un método de PrinterJob)
      job.print();
  }

  /**
   * Método del interfaz Printable que será invocado por el método print
   * de la clase PrinterJob, preguntando al objeto que se quiere imprimir
   * cómo hacerlo.
   *
   * El objeto gráfico que se le pasa como parámetro representa al objeto
   * PrinterJob.
   */
  public int print(Graphics g, PageFormat format, int pagenum) {
    // El PrinterJob imprimira páginas hasta que se le avise que hemos llegado
    // al final con el valor NO_SUCH_PAGE
    if (pagenum > 0)
      return Printable.NO_SUCH_PAGE;

    // El objeto gráfico lo convertimos en un objeto gráfico de dos dimensiones
    Graphics2D g2 = (Graphics2D) g;

    // Establecemos los márgenes (superior e izdo) de impresión que se
    // especifican en el objeto PageFormat
    g2.translate(format.getImageableX(), format.getImageableY());

    // Pedimos al componente que se dibuje él mismo en al impresora
    this.paint(g2);

    //Con esta constante le decimos al PrinterJob que hemos impreso la página
    return Printable.PAGE_EXISTS;
  }
}
```