Relatórios em PDF

Prof. Dr. João Paulo Lemos Escola

Tópicos da aula

- Nesta aula vamos aprender a criar uma classe para gerar nossos relatórios em formato PDF;
- A biblioteca iText permite criar arquivos PDF;
- Trata-se de um recurso bem simples e poderoso para geração de relatórios para impressão;

Baixar a biblioteca iText

A14ex01.jar

- Neste exemplo, criamos um arquivo “doc.pdf”, com a palavra “hello world”, na raiz do diretório do projeto:

```
public static void main(String[] args) {
    // cria um novo documento iText
    Document document = new Document();
    try{
        // configura o local e nome do arquivo pdf
        PdfWriter.getInstance(document, new FileOutputStream("doc.pdf"));
        // abre o arquivo pdf
        document.open();
        // insere conteúdo no arquivo
        document.add(new Paragraph("hello world"));
    }
    catch(Exception e){
        e.printStackTrace();
    }
    // fecha o documento
    document.close();
}
```

Exibindo o arquivo PDF

- Com o código abaixo, é possível exibir um arquivo qualquer (doc, gif, jpg, pdf etc);
- Inclua esse código ao gerar o PDF para que seja exibido na tela automaticamente:

```
// instancia o objeto do pdf gerado
File meuPdf = new File("doc.pdf");
// mostra o pdf
Desktop.getDesktop().open(meuPdf);
```

Formatando o texto

- A classe Font, permite configurar o tipo da letra, tamanho, cor e espessura (negrito, itálico);
- A classe Chunk cria um objeto com o texto já formatado com o objeto Font:

```
// configura o local e nome do arquivo pdf
PdfWriter.getInstance(document, new FileOutputStream("doc.pdf"));
// abre o arquivo pdf
document.open();

// tipos de fontes
Font helvetica =  new Font(FontFamily.HELVETICA);
Font helveticaN =  new Font(FontFamily.HELVETICA,Font.BOLD);
Font helvetica20N =  new Font(FontFamily.HELVETICA, 20, Font.BOLD);
Font helvetica20NAzul =  new Font(FontFamily.HELVETICA, 20, Font.NORMAL, BaseColor.BLUE);

// chunk é a junção do texto com a fonte
Chunk textoAzul = new Chunk("texto formatado", helvetica20N);

// cria um parágrafo com o texto formatado
Paragraph p = new Paragraph(textoAzul);
// insere conteúdo no arquivo
document.add(p);
// fecha o documento
document.close();
// instancia o objeto do pdf gerado
File meuPdf = new File("doc.pdf");
// mostra o pdf
Desktop.getDesktop().open(meuPdf);
```

Classe PDF

- Vamos criar uma classe Pdf no pacote util para facilitar o processo de geração de relatórios:

```
public class Pdf {

    // cria um novo documento iText
    private Document document = new Document();
    // nome utilizado para gerar o pdf
    private String arquivoPdf;
    // configuração da fonte do título do documento
    private Font fonteTitulo = new Font(Font.FontFamily.COURIER, 20, Font.BOLD, BaseColor.RED);
    // cria uma linha separadora (opcional)
    DottedLineSeparator linha = new DottedLineSeparator();


    public Pdf(String titulo){
        // nome do arquivo a salvar
        arquivoPdf = titulo+new SimpleDateFormat("d-M-Y").format(new Date())+".pdf";
        // ajusta a distância do parágrafo para a linha
        linha.setOffset(-5);

        // adiciona o título
        Chunk cTitulo = new Chunk(titulo,fonteTitulo);

        try{
            // configura o local e nome do arquivo pdf
            PdfWriter.getInstance(document, new FileOutputStream(arquivoPdf));
            // inicializa o pdf
            document.open();
            // adiciona o chunk
            document.add(cTitulo);
            // adiciona a linha separadora
            addLinha();
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
```

Classe PDF (cont.)

- Os métodos abaixo devem ser incluídos também na nossa classe Pdf:

```
public void add(String s){
    try{
        document.add(new Paragraph(s));
    }
    catch(Exception e){
        e.printStackTrace();
    }
}

public void addLinha(){
    try{
        document.add(linha);
        document.add(new Paragraph(" "));
    }
    catch(Exception e){
        e.printStackTrace();
    }
}

public void exibir(){
    try{
        // finaliza o pdf
        document.close();
        // instancia o objeto do pdf gerado
        File meuPdf = new File(arquivoPdf);
        // mostra o pdf
        Desktop.getDesktop().open(meuPdf);
    }
    catch(Exception e){
        e.printStackTrace();
    }
}
```

Utilizando nossa classe PDF

```
private void mnuRelatorioPessoasActionPerformed(java.awt.event.ActionEvent evt) {
    PessoaController.listarPdf("");// listar todos
}
```

```
public abstract class PessoaController {

    public static void listarPdf(String s){
        // cria o objeto de acesso ao bd
        PessoaDao dao = new PessoaDao();
        // cria o objeto pdf
        Pdf pdf = new Pdf("Listagem de Pessoas");
        // busca a lista
        List<Pessoa> pessoas = dao.listar(s);
        // percorre a lista
        for (Pessoa p : pessoas){
            pdf.add(p.getNome());// adiciona o nome no pdf
            pdf.add(p.getCpf());// adiciona o cpf no pdf
            pdf.addLinha();// adiciona uma régua no pdf
        }
        // mostra o pdf
        pdf.exibir();
    }
}
```

A14ex02.jar

- Crie um programa que gere um relatório de Pessoas a partir dos dados existentes no banco de dados.

PDF em formato de tabela

- Podemos também criar tabelas em documentos PDF:

```
public static void main(String[] args) {
    // cria um novo documento iText
    Document document = new Document();
    try{
        // configura o local e nome do arquivo pdf
        PdfWriter.getInstance(document, new FileOutputStream("doc.pdf"));
        // abre o arquivo pdf
        document.open();

        // cria uma tabela com 3 colunas
        PdfPTable t = new PdfPTable(3);

        // adiciona células na tabela
        t.addCell("c1");
        t.addCell("c2");
        t.addCell("c3");
        t.addCell("c4");
        t.addCell("c5");
        t.addCell("c6");

        // adiciona a tabela no pdf
        document.add(t);

        // fecha o documento
        document.close();
        // instancia o objeto do pdf gerado
        File meuPdf = new File("doc.pdf");
        // mostra o pdf
        Desktop.getDesktop().open(meuPdf);
    }
    catch(Exception e){
        e.printStackTrace();
    }
}
```

Classe Pdf com tabela

- Trabalhar com tabelas pode deixar os relatórios melhor organizados;
- Vamos aprimorar nossa classe Pdf para que utilize tabelas ao invés de parágrafos, como fazia anteriormente.

addColunasTabela

- Método que recebe um vetor de String com os nomes das colunas da tabela:

```
public void addColunasTabela(String[] colunas) {
    try {
        // cria uma tabela com 3 colunas
        PdfPTable t = new PdfPTable(colunas.length);
        // percorre o vetor e inclui cada valor como uma célula da tabela
        for (String s: colunas){
            // adiciona o título da coluna
            Chunk cTitulo = new Chunk(s,fonteColunas);
            t.addCell(new Paragraph(cTitulo));
        }
        // adiciona a tabela no pdf
        document.add(t);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

addLinhaTabela

- Método que recebe um vetor com os dados de cada linha da tabela:

```
public void addLinhaTabela(String[] linha){
    try {
        // cria uma tabela com 3 colunas
        PdfPTable t = new PdfPTable(linha.length);

        // percorre o vetor e inclui cada valor como uma célula da tabela
        for (String s: linha)
            t.addCell(s);

        // adiciona a tabela no pdf
        document.add(t);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Na classe controller:

```
public static void listarPdfTabela(String s){
    // cria o objeto de acesso ao bd
    PessoaDao dao = new PessoaDao();
    // cria o objeto pdf
    Pdf pdf = new Pdf("Listagem de Pessoas");
    // busca a lista
    List<Pessoa> pessoas = dao.listar(s);
    // envia o vetor com os nomes das colunas da tabela
    pdf.addColunasTabela(new String[]{"Nome", "CPF"});
    // percorre a lista
    for (Pessoa p : pessoas){
        // cria um vetor com os dados da linha atual
        String[] linha = new String[]{
            p.getNome(),p.getCpf()
        };
        // adiciona a linha atual na tabela
        pdf.addLinhaTabela(linha);
    }
    // mostra o pdf
    pdf.exibir();
}
```

A14ex03.jar

- Implementar um programa que utilize o recurso de tabelas para gerar o relatório de Pessoas em pdf.

O que aprendemos?

- Biblioteca iText;