PdWriter Class

The PdWriter class (com.baseprogramming.pdwriter.PdWriter) is a class that demonstrates how to use the Apache project PDFBox. More so, it demonstrates how PDFBox can be extended to provide a more user-friendly interface to write content to PDF--without having to worry about breaking up a large chunk of text such that it fits in a page.

This class attempts to emulate a basic word processor approach, where text is written in paragraphs(PdParagraph class), and each paragraph has settings such as font and spacing.

The PdWriter class has two constructors

```
  public PdWriter(PDDocument document, Margin margin)
```

And

```
    public PdWriter(PageMetadata meta, PDDocument document)
```

The Margin class stores margin information (Top, Left, Bottom, and Right). The margins are stored as a PdUnit. The concept of a PdUnit (as with all other code in this project) is an experimental concept; its goal is to provide a client with a wide range of options for units of measures. Currently, the available units of measure are: PdInch, PdMillimeters,PdPica, PdPixels, and Points. All units of measure convert the given value to points--the standard unit of measure in graphic systems; the PdPoints class merely echos the value given.

The PageMetadata class has basic page information(Margin and PDRectangle), and has methods to compute page boundaries. The default PDRectangle is PDRectangle.LETTER

To get started with the PdWriter class, create create an instance:

```
  Margin margin= new Margin(0.75f, 0.2f, 0.5f, 0.25f);
  PdWriter writer= new PdWriter(pdDoc, margin);
```

Then create one (or more) PdParagraph objects:

```
  PdParagraph heading=writer.createParagraph();
  heading.setFont(PDType1Font.TIMES_BOLD);
  heading.setFontSize(24);
  heading.setAboveSpacing(new PdInch(0.75f));
  heading.setBelowSpacing(new PdInch(0.75f));
```

```
PdParagraph body = writer.createParagraph();
body.setFirstLineIndent(new PdInch(0.3f));
body.setBelowSpacing(new PdInch(0.17f));

PdParagraph code=writer.createParagraph();
code.setFont(PDType1Font.COURIER);
code.setBeforeTextIndent(new PdInch(0.5f));
code.setAboveSpacing(new PdInch(0.3f));
code.setBelowSpacing(new PdInch(0.3f));
```

Write text (paragraphs, by calling the method PdWriter.write(PdParagraph,String):

```
writer.write(body,"Write text (paragraphs, by calling the method
 PdWriter.write(PdParagraph,String):");
```

Note that where is a write(String) method, that creates its own PdParagraph instance with the default values.

It is also possible to write a numbered or bullet list--use the class PdList, which extends PdParagraph. Use the factory method numberedList(PageMetadata) to create a numbered list paragraph style:

```
PdList list= PdList.numeredList(writer.getMeta());
```

Or

```
PdList list= PdList.bulletList(writer.getMeta());
```

to create a bullet list. The line of code:

```
writer.write(list, "Java","C++","Python");
```

generates the following numbered list:

1. Java
2. C++
3. Python

A List of String can also be passed

To create a bullet list, simple use the other factory method:

```
list=PdList.bulletList(writer.getMeta());
```

and call the PdWriter write(PdList,...) method as with the numbered list:

*Java
*C++
*Python