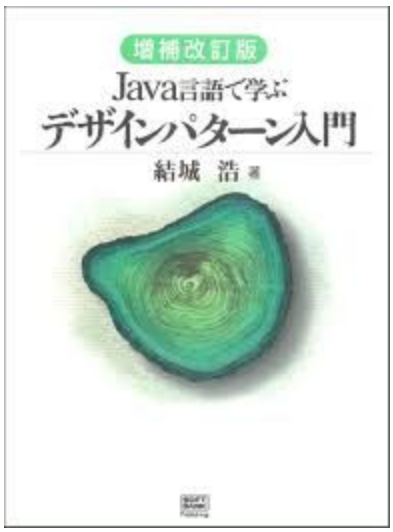
Java言語で学ぶデザインパターン入門

Builder
複雑なインスタンスを組み立てる



Builderパターン

構造をもったインスタンスをくみ上げていく。

サンプルプログラム

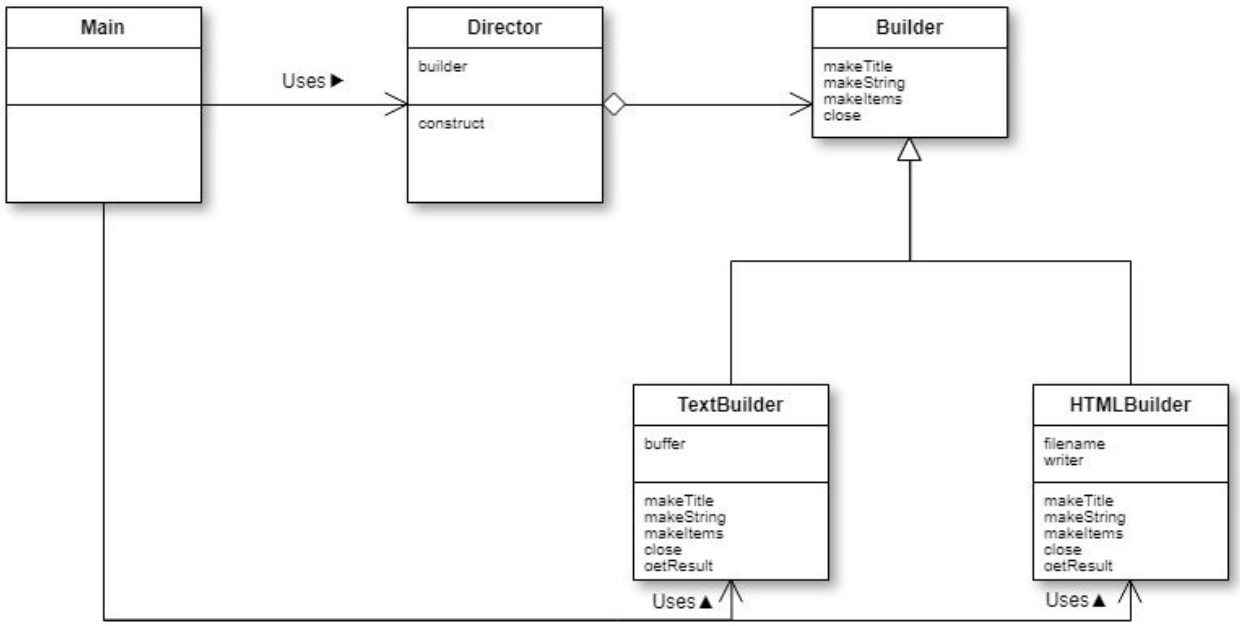
文章を作成するプログラムを作る。

- タイトルを１つ含む
- 文字列をいくつか含む
- 箇条書きの項目をいくつか含む

クラス一覧

名前	解説
Builder	文書を構成するためのメソッドを定めた抽象クラス。
Director	１つの文書を作るクラス
TextBuilder	プレーンテキスト（普通の文字列）を使って文書を作るクラス
HTMLBuilder	HTMLファイルを使って文書を作るクラス
Main	動作テスト用のクラス

クラス図



Builderクラス

Builderクラスは文書を作るメソッドたちを宣言している抽象クラス。

```
public abstract class Builder {
	public abstract void makeTitle(String title);
	public abstract void makeString(String string);
	public abstract void makeItems(String[] items);
	public abstract void close();
}
```

Directorクラス

```
public class Director {
	private Builder builder;
	public Director(Builder builder) {
		this.builder = builder;
	}
	public void construct() {
		builder.makeTitle("Greeting");
		builder.makeString("朝から昼にかけて");
		builder.makeItems(new String[] {
				"おはようございます。",
				"こんにちは。"
		});
		builder.makeString("夜に");
		builder.makeItems(new String[] {
				"こんばんは。",
				"おやすみなさい。",
				"さようなら。"
		});
		builder.close();
	}
}
```

TextBuilderクラス

```
public class TextBuilder extends Builder{
	private StringBuffer buffer = new StringBuffer();
	public void makeTitle(String title) {
		buffer.append("==========================================\n");
		buffer.append(" 『" + title + "』 \n");
		buffer.append("\n");
	}

	public void makeString(String string) {
		buffer.append('■' + string + "\n");
		buffer.append("\n");
	}
	public void makeItems(String[] items) {
		for(int i=0; i<items.length;i++) {
			buffer.append(" ・" + items[i] + "\n");
		}
		buffer.append("\n");
	}
	public void close() {
		buffer.append("==========================================\n");
	}
	public String getResult() {
		return buffer.toString();
	}
}
```

HTMLBuilderクラス

```
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;

public class HTMLBuilder extends Builder {
```

```
	private String filename;
	private PrintWriter writer;

	public void makeTitle(String title) {
		filename = title + ".html";
		try {
			writer = new PrintWriter(new FileWriter(filename));
		} catch (IOException e) {
			e.printStackTrace();
		}
		writer.println("<html><head><title>" + title +
"</title></head><body>");
		writer.println("<hl>" + title + "</hl>");
	}

	public void makeString(String string) {
		writer.println("<p>" + string + "</p>");
	}

	public void makeItems(String[] items) {
		writer.println("<ul>");
		for (int i = 0; i < items.length; i++) {
			writer.println("<li>" + items[i] + "</li>");
		}
		writer.println("</ul>");
	}

	public void close() {
		writer.println("</body></html>");
		writer.close();
	}

	public String getResult() {
		return filename;
	}
}
```

Mainクラス

```
public class Main {
	public static void main(String[] args) {
		if(args.length != 1) {
			usage();
			System.exit(0);
		}
		if(args[0].equals("plain")) {
			TextBuilder textBuilder = new TextBuilder();
			Director director = new Director(textBuilder);
			director.construct();
			String result = textBuilder.getResult();
			System.out.println(result);
		} else if(args[0].equals("html")) {
			HTMLBuilder htmlBuilder = new HTMLBuilder();
			Director director = new Director(htmlBuilder);
			director.construct();
			String filename = htmlBuilder.getResult();
			System.out.println(filename + "が作成されました。");
		} else {
			usage();
			System.exit(0);
		}
	}
	public static void usage() {
		System.out.println("Usage: java Main plain プレーンテキストで文書
作成");
		System.out.println("Usage: java Main html HTMLファイルで文書作成
");
	}
}
```