修改本例，增加一个新的concrete的Builder

Code：

import java.io.FileWriter;  
import java.io.IOException;  
  
// 抽象建造者  
abstract class Builder {  
 public abstract void makeTitle(String title);  
 public abstract void makeString(String str);  
 public abstract void makeItems(String[] items);  
 public abstract Object getResult();  
}  
  
// 具体建造者：普通文本  
class TextBuilder extends Builder {  
 private StringBuilder buffer = new StringBuilder();  
  
 @Override  
 public void makeTitle(String title) {  
 buffer.append("==========\n");  
 buffer.append("[" + title + "]\n");  
 buffer.append("\n");  
 }  
  
 @Override  
 public void makeString(String str) {  
 buffer.append("> " + str + "\n");  
 buffer.append("\n");  
 }  
  
 @Override  
 public void makeItems(String[] items) {  
 for (String item : items) {  
 buffer.append(" - " + item + "\n");  
 }  
 buffer.append("\n");  
 }  
  
 @Override  
 public Object getResult() {  
 buffer.append("==========\n");  
 return buffer.toString();  
 }  
}  
  
// 具体建造者：HTML  
class HTMLBuilder extends Builder {  
 private String filename;  
 private FileWriter writer;  
  
 @Override  
 public void makeTitle(String title) {  
 filename = title + ".html";  
 try {  
 writer = new FileWriter(filename);  
 writer.write("<html><head><title>" + title + "</title></head><body>\n");  
 writer.write("<h1>" + title + "</h1>\n");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 @Override  
 public void makeString(String str) {  
 try {  
 writer.write("<p>" + str + "</p>\n");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 @Override  
 public void makeItems(String[] items) {  
 try {  
 writer.write("<ul>\n");  
 for (String item : items) {  
 writer.write("<li>" + item + "</li>\n");  
 }  
 writer.write("</ul>\n");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 @Override  
 public Object getResult() {  
 try {  
 if (writer != null) {  
 writer.write("</body></html>\n");  
 writer.close();  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 return filename;  
 }  
}  
  
// 新增具体建造者：Markdown  
class MarkdownBuilder extends Builder {  
 private StringBuilder buffer = new StringBuilder();  
  
 @Override  
 public void makeTitle(String title) {  
 buffer.append("# " + title + "\n\n");  
 }  
  
 @Override  
 public void makeString(String str) {  
 buffer.append(str + "\n\n");  
 }  
  
 @Override  
 public void makeItems(String[] items) {  
 buffer.append("- " + String.*join*("\n- ", items) + "\n\n");  
 }  
  
 @Override  
 public Object getResult() {  
 return buffer.toString();  
 }  
}  
  
// 指挥者  
class Director {  
 private Builder builder;  
  
 public Director(Builder builder) {  
 this.builder = builder;  
 }  
  
 public Object construct() {  
 builder.makeTitle("Greeting");  
 builder.makeString("从早上到白天结束");  
 builder.makeItems(new String[]{"早上好", "白天好"});  
 builder.makeString("到了晚上");  
 builder.makeItems(new String[]{"晚安。", "祝你有个好梦。"});  
 return builder.getResult();  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 if (args.length != 1) {  
 *usage*();  
 System.*exit*(0);  
 }  
  
 String format = args[0];  
 Object result;  
  
 switch (format) {  
 case "plain":  
 Director plainDirector = new Director(new TextBuilder());  
 result = plainDirector.construct();  
 System.*out*.println("普通文本格式：\n" + result);  
 break;  
 case "html":  
 Director htmlDirector = new Director(new HTMLBuilder());  
 result = htmlDirector.construct();  
 System.*out*.println("HTML格式：\n已生成 " + result + ".");  
 break;  
 case "markdown":  
 Director markdownDirector = new Director(new MarkdownBuilder());  
 result = markdownDirector.construct();  
 System.*out*.println("Markdown格式：\n" + result);  
 break;  
 default:  
 *usage*();  
 System.*exit*(0);  
 }  
 }  
  
 private static void usage() {  
 System.*out*.println("Usage: java Main [plain/html/markdown]");  
 System.*out*.println(" plain - 生成普通文本格式");  
 System.*out*.println(" html - 生成HTML格式");  
 System.*out*.println(" markdown- 生成Markdown格式");  
 }  
}

运行结果：

