プログラム設計

http://bit.ly/design4d

UMLからコードの生成

後期 第3週

2019/10/14

今回は

UML作成したクラス図から、プログラムのスケルトンコードを生成して、プログラムを完成させるまでの流れを学びます。

【課題の準備】

演習室で作業する前に、以下のコマンドを
入れるだけで準備が完了する

```
$   mygitclone4d   「自分のGitHubユーザ名」
$   cd prog4d-(ユーザ名)
$   ./myconf
```

※本体をシャットダウンするまでは、
上記「mygitclone」と「myconf」の設定は有効です

【課題の準備】

以下の流れで、課題のプログラムを作るためのフォルダを準備しましょう。

1. 端末を起動して、以下のコマンドを実行して後期第3週のフォルダを作る

```
$ cd prog4d-(ユーザ名)        (←既に移動しているなら不要)
$ mkdir week203
$ cd week203
```

※課題で作るファイル名は各自で決めて構いません。

① クラス図を作る

「Todo（リマインダ）を扱う」プログラムのクラス図（以下の3つのクラス）を作っていきます。

- **「日付」を表すDate**
- **「1つのTodo」を表すTodo**
- **「複数のTodoをリストで持つ」TodoList**

クラスDate

【属性】

- int型の「year」（年）
- int型の「month」（月）
- int型「day」（日）

【操作】

- コンストラクタ ・・・ 1970年1月1日にセットする
- 各属性のsetterとgetter
- void show() ・・・ 各属性を標準出力へ出力する

クラスTodo

【属性】

- String型の「name」（Todoの名前）
- int型の「priority」（優先順位）

【操作】

- コンストラクタ ･･･ nameを"undefined"、priorityを0とし、「期限日」のインスタンスを作る
- 各属性のsetterとgetter（「期限日」も含む）
- void show() ･･･ Todoのname, priorityを表示し、「期限日」のメソッドshowを呼び出す

【関係】

- 「期限日」を表すため、クラスDateを集約する

クラスTodoList

【属性】

- int型の「count」（リストの要素数）

【操作】

- コンストラクタ ・・・ countを0とし、「Todo」の集約を表す配列を作成する
- Todoの配列に対するsetterとgetter
- void show() ・・・ Todoの一覧を表示する、つまりTodoの配列に繰り返しshowを呼び出す
- void addTodo(Todo t) ・・・ Todoをリストに追加する、count番目にTodoを追加するが、配列の上限に達した場合はcountを0にしてから追加する、追加後はcountを1増やす

【関係】

- 「Todoのリスト」を表すため、クラスTodoを集約する
  多重度は0～3個の範囲内

② スケルトンコードを生成する

次の手順でJavaのスケルトンコードを生成してみましょう。

「ツール」メニュー
→ 「Java」
→ 「スケルトンコードの作成」
→ 保存先フォルダを選ぶ

③ コードを完成させる

全てのクラスのメソッド内の処理を、先程のクラスの説明を参考に書き足します。また、出力の様子は後で示す実行結果を参考にして下さい。

Eclipseを使ってコードを完成させる場合は、Javaプロジェクトを作成したあとに、Javaコードをそのプロジェクトにインポートします。

④ プログラムを実行する

コードが完成したら、mainを持ったファイル2_03_main.java（講義資料と同じところからダウンロードできる）も含め、全てのファイルをコンパイルします。

（複数のJavaファイルに分かれている場合は、「javac」コマンドで全てのJavaファイルをまとめて一緒にコンパイルできます。）

④ プログラムを実行する

[実行結果]

```
  2019-10-31                                    (← d1.show() の出力)
  (1) Prepare Halloween  2019-10-31             (← t1.show() の出力)
  (1) Prepare Halloween  2019-10-31             (← l1.show() の出力)
  (0) undefined  1970-1-1
```

【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
$ git commit -m “課題3-1提出”
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザ名)

今回の小テスト

第2週までの内容で小テストを実施します。

小テストについて

小テストの注意点

- **他人の力は借りずに、自分だけでプログラムを作成する。（つまり定期試験と同様）**

- **プログラムの提出はGitHubを使用する。**

小テストについて

小テスト中に参照できるもの

□ **教科書, 参考書, 配付資料**

□ **自分のホームディレクトリ（ホームフォルダ）以下に保存されているファイル**

□ **小テストでは紙媒体のものは参照可能**

□ **上記以外の情報を参照することは不正行為とする**

例：USBで接続された機器に保存されているファイルの参照
Webブラウザ、ネットワークを介した情報の参照
自分のPCを使用する、など