Проектирование программного обеспечения (практика)

Практика 1: Введение, задача про CLI

Юрий Литвинов
yurii.litvinov@gmail.com

15.01.2020г

Формальности

- В конце курса оценка
  - Домашние работы
  - Дедлайны (-50% баллов за пропуск)
  - Работа в аудитории
  - Индивидуальные и групповые задачи (с возможностью перераспределить баллы)
  - Связанные цепочки заданий
- Условия, материалы и сдача домашек через http://hwproj.me/courses/51
  - Надо записаться на курс
  - Ссылка на пуллреквест в собственный репозиторий на GitHub
  - Задач будет несколько, так что выкладывать лучше так, чтобы их все было можно смерджить

Критерии оценивания

- Меньше 60% заданий – 0
- От 60% до 100% — линейная шкала от 2 до 10 баллов
  - то есть, ровно 60% заданий — 2 балла
  - 80% заданий — 6 баллов
  - 100% заданий — 10 баллов
- Задача с сегодняшей пары оценивается в 20 баллов, остальные (пока что) в 10
- Оценки за работу на паре с небольшим весом (ближайшая такая пара через две недели)
  - Оценка входит в процент выполненных заданий!
- В итоговой оценке практика учитывается с весом 0.4, экзамен — 0.6
- Округление арифметическое
- Две оценки — в конце этого и следующего модулей
  - Итоговая — оценка за 3-й модуль с весом 0.6 и оценка за 4-й модуль с весом 0.4

Краткое содержание курса по практике

- Снова лекции
  - Про архитектурную документацию
  - Про UML и другие языки проектирования
  - Про реализацию паттернов
  - Про антипаттерны
  - Различные примеры архитектур
- Небольшие задачи прямо на паре
  - Проектировать разные приложения
  - Технические вещи, типа рисования диаграмм
- Большие задачи на дом

Что ожидается от решений

- Работоспособность и соответствие требованиям условия
- Наличие архитектурной документации
  - Комментарии к каждому классу, интерфейсу и public-методу
  - Краткое описание деталей реализации в README
- Следование стайлгайдам и правилам здравого смысла
- Наличие юнит-тестов
- Применение индустриальных практик: логирование, Continuous Integration, обработка исключений

Ещё комментарии

- Овердизайн и активное использование знаний с лекций приветствуются
- Обоснованность принятых решений важнее, чем техника кодирования
- Некоторые требования могут показаться ненужными — это нормально
  - Мы учимся не написанию кода, а инструментам и техникам проектирования
- Комментарии вида “у вас неправильная архитектура” будут очень редки, как ни странно

Задача про CLI

Реализовать простой интерпретатор командной строки, поддерживающий команды:

- **cat [FILE]** — вывести на экран содержимое файла
- **echo** — вывести на экран свой аргумент (или аргументы)
- **wc [FILE]** — вывести количество строк, слов и байт в файле
- **pwd** — распечатать текущую директорию
- **exit** — выйти из интерпретатора

Задача про CLI (продолжение)

- Должны поддерживаться одинарные и двойные кавычки (full and weak quoting)
- Окружение (команды вида “имя=значение”), оператор $
- Вызов внешней программы через Process (или его аналоги)
  - если введено что-то, чего интерпретатор не знает
- Пайплайны (оператор “|”)

Примеры

```
>echo "Hello, world!"
Hello, world!
> FILE=example.txt
> cat $FILE
Some example text
> cat example.txt | wc
1 3 18
> echo 123 | wc
1 1 3
> x=exit
> $x
```

Нефункциональные требования

- Легко добавлять новые команды (расширяемость)
- Наличие возможности реализовать что-то новое из того, что умеют другие шеллы (сопровождаемость)
- Адекватное покрытие юнит-тестами (качество, сопровождаемость)
- Комментарии в коде (сопровождаемость)
- Архитектурное описание, как умеете (сопровождаемость)
- Стайлгайд (сопровождаемость)

Как сдавать

- Сделать для этого курса репозиторий
- Выложить решение в отдельную ветку
- Сделать пуллреквест к себе в мастер
- Записаться на курс http://hwproj.me/courses/51
- Приложить там ссылку на пуллреквест к решению
- Смерджить пуллреквест, когда задача зачтена
- **Дедлайн 10:00 05.02.2020**

Что делать сейчас

Первые фазы жизненного цикла

- Выполнить анализ и определить подходы к решению
- Выявить подводные камни и способы их преодоления
- Декомпозировать задачу на подсистемы, классы и методы
- Нарисовать диаграмму классов
- Словами описать принцип работы и основные принятые решения
- К концу пары все должны понимать, что и как надо кодить

Соображения

- Проектирование сверху вниз
  - Определитесь с общей структурой системы
  - Определитесь с компонентами, их ответственностью и связями между ними
  - Только после этого переходите к проектированию компонентов
    - По такой же схеме
  - Возможно, придётся возвращаться на уровень выше
- Опасайтесь архитектурной жадности, надо вовремя остановиться