Еще одно объяснение ООП (слабая связность, четкие функции)

Java Core (/quests/QUEST_JAVA_CORE)
1 уровень (/quests/lectures/?quest=QUEST_JAVA_CORE&level=1), 3 лекция (/quests/lectures/questcore.level01.lecture03)

ОТКРЫТА

— Привет, Амиго! Хотела тебе рассказать еще об одном преимуществе использования ООП. Видишь ли – программы больше напоминают не строения, а животных. Их не строят, их выращивают. Разработка — это постоянные изменения. В строительстве ты можешь иметь хороший план и четко ему следовать. В случае с разработкой программ – это не так.

Очень часто что-то нельзя сделать тем способом, который ты себе наметил, и приходится многое переделывать. Еще чаще меняются требования заказчика.

— А если заказчик проекта дал очень точную его спецификацию?

— Тогда взгляни на ситуацию во времени. Успех продукта приведет к тому, что заказчик захочет выпустить его новую версию, а затем еще и еще. И, конечно, нужно будет всего лишь добавить «небольшие изменения» в уже существующий продукт. Поэтому разработка продукта – это последовательность постоянных изменений. Только масштаб времени разный. Каждая новая версия может выходить раз в неделю, раз в месяц или раз в полгода.

— И какой вывод можно сделать из всего этого?

— Внутреннюю структуру продукта нужно поддерживать в таком состоянии, которое позволит внести значительные (и не очень) изменения с минимальными переделками.

— И как такое сделать?

— Мы уже говорили, что программа состоит из объектов, которые взаимодействуют между собой. Давай нанесем на доску все объекты нашей программы, обозначив их жирными точками. И проведем от каждого объекта (точки) стрелочки ко всем объектам (точкам), с которыми он взаимодействуют.

Теперь мы будем объединять объекты (точки) в группы. Точки должны быть объединены в группу, если связи между ними гораздо интенсивнее, чем с остальными точками. Если большинство стрелочек от точки идет к точкам ее же группы, тогда разбиение на группы произошло правильно. Точки внутри одной группы мы будем называть сильно связанными, а точки из разных групп – слабо связанными.

Это называется «принцип слабой связности». Программа разбивается на несколько частей, часто слоев, логика которых сильно завязана на их внутреннее устройство и очень слабо на другие слои/части. Обычно взаимодействие слоев очень регламентировано. Один слой может обращаться ко второму и использовать только небольшую часть его классов.

— Тот же принцип «разделения на отделы» только в большем масштабе?

— Именно. Это приводит к тому, что мы можем реорганизовать отдел, повысить его эффективность, нанять в него еще больше людей, но если мы не изменим протокол взаимодействия других отделов с нашим, то все сделанные изменения останутся локальными. Никому не придется переучиваться. Не придется переделывать всю систему. Каждый отдел может заниматься такой внутренней оптимизацией, если общие механизмы взаимодействия выбраны удачно.

— Выбраны удачно. А что будет, если они выбраны неудачно?

— Тогда «запас изменений» быстро иссякнет и придется переделывать всю систему. Такое приходится делать время от времени. Нельзя предугадать, что будет в будущем, но можно свести количество таких переделок к минимуму.

— Хорошо. Про пользу такого разделения я понял, а ООП тут причем?

— Выбор структуры отделов и способа их взаимодействия – это «принцип Абстракции». В программировании он используется для определения, на какие части лучше разбить программу, и как эти части должны взаимодействовать. Данный принцип также можно применять к разделению полученных частей, пока мы не разобьем программу на отдельные классы.

— А сокрытие внутренней структуры этих частей, и жёсткие ограничения на взаимодействие с другими частями – это Инкапсуляция, да?

— Именно. Инкапсуляция + Абстракция – это краеугольные камни ООП. Хорошая программа обязана следовать этим двум принципам. В дальнейшем мы рассмотрим остальные принципы и поймем, какие преимущества они дают.

— Очень интересно. Жду с нетерпением.

‹ (/quests/lectures/questcore.level01.lecture02)

×11 › (/quests/lectures/questcore.level01.lecture04)

Комментарии (16) популярные новые старые

Никита

Введите текст комментария

Hello world! 35 уровень 5 декабря 2017, 08:48

— Привет, Амиго! Хотела тебе рассказать еще об одном преимуществе использования ООП. Видишь ли – программы больше напоминают не строения, а животных. Их не строят, их выращивают. Разработка — это постоянные изменения.
Очень плохой пример как по мне, кто читал "Совершенный код" поймут)

Ответить 0

Atom 16 уровень 20 октября 2017, 18:16

Хотите много денег просто так? не бывает такого. Сначала вложите (инвестируйте) в себя, а потом средства многократно вернуться. Иначе зачем пришли сюда?

Ответить +8

Владимир 11 уровень, Днепр 25 сентября 2017, 20:38

Эх...джавараш. Жаль, а так все хорошо начиналось, но я не сойду с пути программиста - пора браться за "Библию".

Ответить +1

Степан Карсаков 21 уровень, Минск 17 октября 2017, 13:47

Что, не потянул?

Ответить 0

Владимир 11 уровень, Днепр 29 октября 2017, 21:34

Потянул, технологии учу на ютубе и практикую с помощью курсовых работ по java на фрилансе - практика на деле и деньги платят,правда маленькие ,в основном делаю уклон на английский

Ответить 0

Владимир 11 уровень, Днепр 19 сентября 2017, 23:40

...одни философы смотрю :)

Ответить 0

Сергей Черник 33 уровень 3 сентября 2017, 13:55

По книге "Совершенный код"?

Ответить 0

Эрик Айткулов 34 уровень, Санкт-Петербург 23 июня 2017, 01:09

мнение не претендует на академичность:
В основе декомпозиции лежит принцип абстракции. В этом смысле абстракция являеется принципом. А то, что Вячеслав имеет в виду что принципы ООП это Инкапсуляция, Наследование и Полиморфизм. То да, абстракция туда не входит. В этой лекции абстракция рассматривается в сочетании с Инкапсуляцией. Абстракция лежит в основе Наследования (абстрактные классы) и Полиморфизма (интерфейсы). Это важный инструмент ООП. И с этой точки зрения Абстракцию можно рассматривать как принцип. (не в терминологии ООП, а в общей терминологии русского языка). Можно сказать, что из принципа Абстракции выделили два принципа - Наследование и Полиморфизм... как то так...
Собственно о принципах ООП:
продублирую две ссылки от Max Miheev в комментах следующей лекции
https://habrahabr.ru/post/87119/
https://habrahabr.ru/post/87205/
если кому проще/интереснее/комфортнее видео: //4.47 - 21...
https://www.youtube.com/watch?v=cY9nFsx4q0I&index=2&list=PLmqFxxywkatStbd9hdzVOS1h

Ответить +25

P0huber 14 уровень 25 августа 2017, 14:11

плюсанул ваш ответ за ссылки, спасибо. Но как бы то нибыло, мое уверенное имхо, на собеседовании особенно на английском не стоит совершенно упоминать, что Абстракция = принцип ОПП, чтобы не ставить себя в неловкую ситуации перед интервьювером...

Ответить +1

Вячеслав Перфилкин 14 уровень, Москва 30 мая 2017, 12:45

Абстракция - это не принцип ООП! То что в этой статье называется "Абстракцией", на самом деле является обычной декомпозицией. А абстракция суть программная реализация инкапсуляции. Авторы не владеют терминологией!

Ответить 0

Denis Beck 37 уровень, Екатеринбург 4 июня 2017, 17:21

даже не вдаваясь в дебри терминологии, только во фразе "абстракция суть программная реализация" уже видно противоречие. По-моему, "абстракция" и "реализация" - это, мягко говоря, совершенно разные понятия, если не антонимы.

Ответить +1

Vitya 33 уровень, Нижний Новгород 15 июня 2017, 16:20

Абстракция, как я понял, не только принцип ООП, а вообще программирования и не только. А в ООП основная идея в том, что бы описать назначения объектов отдельно от их реализации. И абстракцией будет набор значимых характеристик объекта.

Ответить 0

Эрик Айткулов 34 уровень, Санкт-Петербург 22 июня 2017, 23:48

жкошь пацанчик, давай еще!! нам важно твое мнение!
"А абстракция суть программная реализация инкапсуляции."
окончания нигде не перепутал? или может слова не в том порядке поставил?
http://ilyabirman.ru/meanwhile/all/are/
ежели, вы, сударь, технарь, извольте выражать мысль терминологией, коей владеете.
Суть коммента - либо троллинг, либо понт...
http://tolkslovar.ru/s14329.html
Не понял что вы хотели сказать фразой выше, но правильнее будет:
Наследование и Полиморфизм суть Абстракции.

Ответить +4

Roman Bogdanov 20 уровень, Днепр 22 сентября 2017, 05:51

"абстракция суть программная реализация инкапсуляции" - читай про себя : " абстракция ПО СУТИ - программная реализация инкапсуляции".

Ответить 0

Лолилоп 13 уровень, Минск 30 апреля 2017, 18:27

Придется идти решать задачки =)

Ответить 0

Кирилл Нигматуллин 32 уровень 16 апреля 2017, 18:14

Радость и веселье

Ответить 0