BDD

№ уроку: 35 **Курс:** Manual QA

Засоби навчання: Браузер, Microsoft Office

Огляд, мета та призначення уроку

Метою даного уроку є поглиблене вивчення підходу BDD. Основних особливостей, переваг та недоліків підходу.

Вивчивши матеріал даного заняття, учень зможе:

- Грамотно застосовувати BDD підхід у роботі.
- Відрізняти правильно побудований процес BDD від звичайних процесів з використанням інструментів BDD.
- Писати Gherkin сценарії у різних стилях.

Зміст уроку

- BDD – що це таке, як утворився, чому популярний
- BDD тестування та BDD проект
- Gherkin сценарії
- Інструменти для BDD тестування в Java/C#/JS
- Звіти про тестування BDD інструментів

Резюме

- Головна думка BDD – постійне спілкування між розробниками, тестувальниками та бізнес-аналітиками. Таке спілкування не тільки створює спільне бачення проекту, а й дозволяє складати приклади поведінки системи.
- На основі прикладів поведінки розробники пишуть код програми, а тестувальники можуть писати автотести або тестувати вручну.
- Таким чином, приклади поведінки (сценарії) є живою документацією проекту. При оновленні вимог оновляться сценарії. Як наслідок, автоматичні тести більше не проходитимуть.
- *DD (*щось* Driven Development) — розробка, яка ґрунтується на чомусь.
- TDD (Test Driven Development) — розробка на основі тестів.
- **TDD**
    - TDD добре підходить для юніт-тестування, тобто. для перевірки роботи окремих модулів самих собою. BDD - для інтеграційного (тобто для перевірки, як окремі модулі працюють один з одним) та e2e (тобто для перевірки всієї системи повністю) тестування.
    - TDD: тести відразу реалізуються в коді, для BDD найчастіше описуються кроки мовою, зрозумілою всім, а не тільки розробникам.
    - TDD: юніт-тести пишуть самі розробники. BDD вимагає поєднання зусиль різних членів команди. Зазвичай тест-кейси (кроки) описуються ручним тестувальником або аналітиком та втілюються в код тестувальником-автоматизатором. У нашій команді ми (фронтенедери) описуємо кроки разом із тестувальниками, а код тестів пише фронтенд-команда.

      TDD перевіряє роботу функцій, BDD - користувальницькі сценарії.
- **BDD** (Behavior Driven Development) — Розробка на основі поведінки.

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- BDD є розширенням TDD-підходу. Тим не менш, вони призначені для різних цілей і їх реалізації використовуються різні інструменти. У різних командах ці поняття можуть інтерпретувати по-різному і часто виникає плутанина між ними.
- На відміну від TDD, цей підхід будується на написанні декількох сценаріїв користувача, під які складаються тести. BDD дозволяє "передбачити", як поведеться користувач, використовуючи продукт відповідно до вимог, які записані в технічній документації. Порядок проходження тестів схожий на TDD. Єдина відмінність – перед проходженням окремих тестів формулюється ряд попередніх умов (сценаріїв), за яких вони повинні бути пройдені.
- Вимоги для BDD зазвичай складає група експертів і не у вигляді програмного коду, а словесно – мовою, зрозумілою всім учасникам проекту.
- Підхід також ефективний в end-to-end-тестуванні (E2E) і дає програмістам уявлення про те, як функціонує вся система, що ними розробляється. Виходить, незважаючи на те, що BDD є розширенням TDD-методології, вони мають різне призначення і реалізуються різним набором інструментів.
- Методології BDD- та TDD-тестування допомагають досягти взаєморозуміння між замовником продукту та всіма учасниками, задіяними в його реалізації.
  Чітке дотримання прописаних заздалегідь специфікацій дозволяє уникнути підводного каміння у вигляді необумовлених сценаріїв або розрізненого трактування функціоналу продукту різними фахівцями.
  Вагомою перевагою таких підходів у тестуванні є відсутність невалідних (недостовірних) сценаріїв. Кожен учасник проекту ще на стадії проектування може побачити функції, що не реалізуються, і розповісти про це колегам. Так можна виключити навіть саму можливість створення неефективного та марного коду.
- Ще одна перевага – наявність технічної документації. TDD та BDD припускають чітко структуровану документацію, яка допомагає швидше адаптуватися новачкам, які зайшли до проекту, вже на етапі виробництва.
- Розробка більше не залежить від людського фактора: щоб зрозуміти, що відбувалося у проекті з моменту його створення, достатньо відкрити потрібний файл.
- Рекомендації щодо складання тестів:
  - у процесі створення та розробки нових сценаріїв для BDD-тестування повинні брати участь усі члени команди та замовник;
  - описуйте бажану поведінку у сценаріях, керуючись вже існуючими специфікаціями поведінки (behavioral specifications);
  - не намагайтеся перевірити в одному тесті відразу кілька сценаріїв чи функцій – це перевантажує його;
  - стежте за тим, щоб модульне тестування завжди виконувалося швидко (тут стандарт - мілісекунди);
  - юніт-тести повинні бути самостійними і не мати зовнішніх залежностей від баз даних, файлових систем;
  - тести призначені не тільки для перевірки коду, а й для ведення технічної документації;
  - void-методи, які не повертають жодних даних, теж треба тестувати.

Закріплення матеріалу

- У чому переваги використання BDD підходів для автоматизації тестування?
- Яким чином можна писати сценарії із застосуванням DDT підходу?

Самостійна діяльність учня

Завдання 1
Створіть вимогу у стилі BDD для пральної машини.

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Напишіть 4 Gherkin сценарію для покриття цієї вимоги. Використовуйте різні стилі та параметризуйте хоча б один тест.

Завдання 2

Використовуючи TDD підхід, напишіть електронний лист начальнику на тему "Відгук про роботу колеги". Ваше завдання спочатку написати "тести" – критерії, яким має задовольняти лист. Потім написати першу версію листа, яка задовольнятиме лише їм. Потім доповнити список тестів ще декількома і переписати. Продовжувати доки лист не буде завершено.

Завдання 3

Контекст:

Це симуляція справжнього проекту, на якому існує автоматизація тестування у стилі BDD. Рішення автоматизації написане на Java + Maven. BDD інструмент - Cucumber.

Ваш QA процес передбачає наступне розподілення ролей:

- AQA відповідальні за розвиток рішення для автоматизації
- MQA проводять ручне тестування та пишуть Gherkin сценаріїв у рішенні для автоматизації

Задача:

- Ви отримали Jira task згідно з яким вам потрібно додати декілька сценаріїв у рішення для автоматизації тестування. Ви оцінили що одного чи двох .feature файлів з декількома сценаріями буде достатньо. Результат виконання задачі – відкритий pull request у main гілку

Правила роботи з репозиторієм

- Основна гілка main захищена, ви не можете пушити в неї напряму
- Працюйте у своїх гілках. Шаблон для назви "task1/{your_name}"
- Як завершили зміни, створіть пул реквест у основну гілку на GitHub

Правила пул реквестів

- Білд на гілці повинен бути зеленим
- Для завершення пул реквесту необхідно отримати апрув від викладача

Процес білду програми передбачає валідацію та проходження тестів. В нашому випадку проходження тестів вимкнуте, та залишив лише валідацію сценаріїв. Це означає що якщо ви використовуєте відсутній крок то білд буде червоний

Для роботи зі сценаріями ви можете використовувати будь-які інструменти редагування коду та роботи з Git. На приклад, IDE IntelliJ IDEA може забезпечити обидві вимоги

Посилання на репозиторій - https://github.com/BreslavetsOleksandr/CBS_QA_BDD.git

Рекомендовані ресурси

Введення у BDD
http://agilerussia.ru/practices/introducing-bdd/

Знайомство з Behavior Driven Development
https://www.ibm.com/developerworks/ru/library/j-cq09187/index.html

Переваги та недоліки BDD підходу написання тестів у вільній формі
https://automated-testing.info/t/kakovy-preimushhestva-i-nedostatki-bdd-podhoda-napisaniya-testov/18592

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com