Manual QA

Регресійне тестування

План уроку

- Що таке регресія?
- Мета регресійного тестування
- Підходи у регресійному тестуванні
- Коли проводити регресійне тестування?
- Автоматизувати чи не автоматизувати?

Що було минулого разу

План уроку

- ЧЕК-ЛИСТ – це тестовий документ, який описує, які функції мають бути перевірені. Організований у вигляді списку завдань, функцій, які необхідно протестувати
- TEST CASE – це тестовий артефакт, що описує сукупність кроків, конкретних умов та параметрів, необхідних для перевірки реалізації тестованої функції або її частини.
- ТЕСТ-СЬЮТ – це набір тестів-кейсів, які об'єднані за загальним сенсом.
- СЦЕНАРІЙ ТЕСТУВАННЯ — послідовність дій над продуктом, які пов'язані єдиним обмеженим бізнес-процесом використання, і відповідних їм перевірок коректності поведінки продукту під час цих дій.

Регресійне тестування

Що це таке?

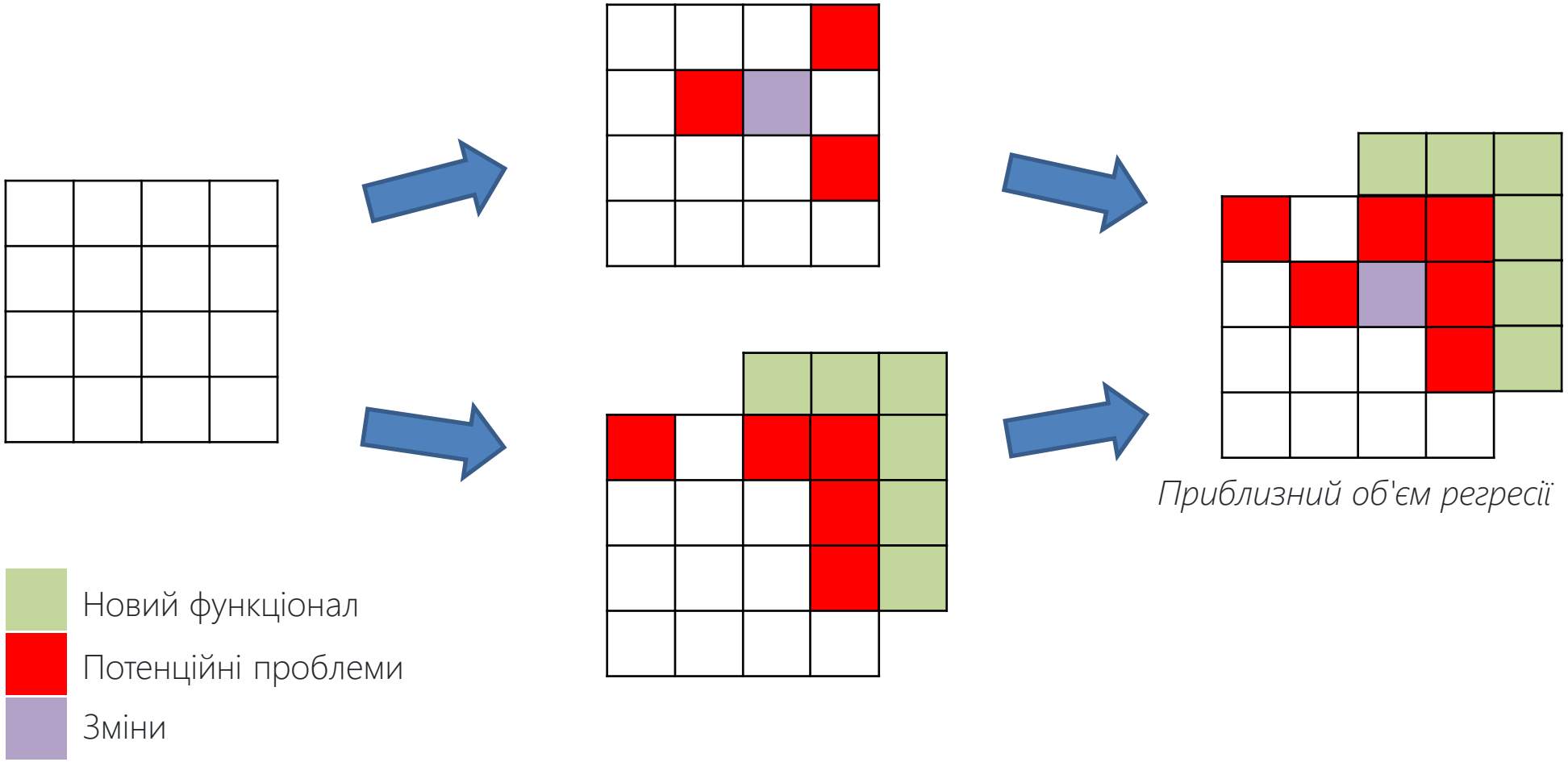
Регресійне тестування – збірна назва для всіх видів тестування програмного забезпечення, спрямованих на виявлення помилок у вже протестованих ділянках вихідного коду.

Такі помилки – коли після внесення змін до програми перестає працювати те, що мало продовжувати працювати, – називають регресійними помилками.

Визначення

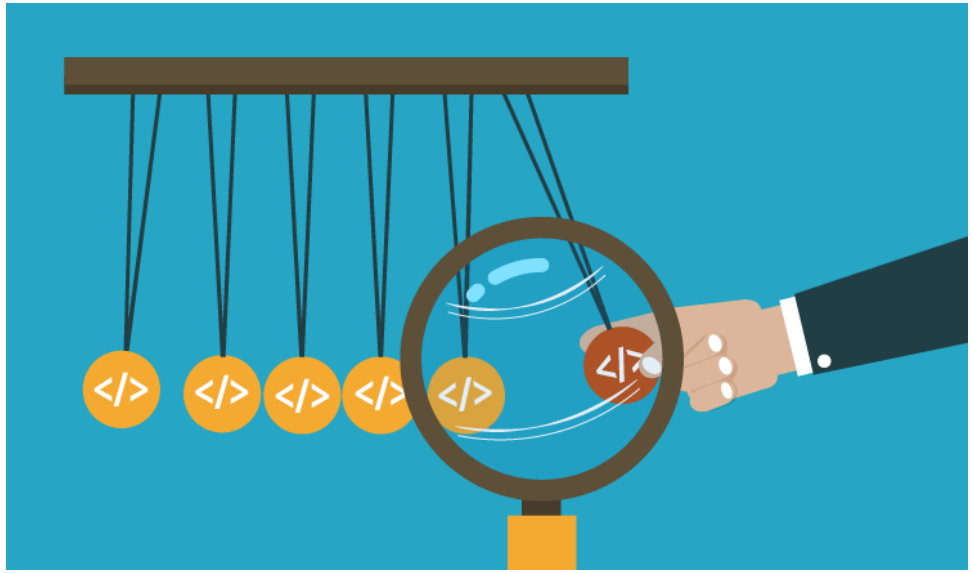
Регресійне тестування – один із видів тестування, пов'язаного із змінами, при якому перевіряється застосунок після внесених змін у коді (новий функціонал та/або виправлення дефекту) на предмет, чи не вплинули вони на інші модулі ПЗ та чи не викликали нові баги.

Необхідність регресії



Приблизний об'єм регресії

Цілі регресії



Головна мета:
забезпечення якості системи після змін в одній її частині.

Додаткові цілі:

- перевірка нового функціоналу у складі вже сформованої робочої системи.
- забезпечення якості ПЗ перед релізом.

Планування регресії

Що нам потрібно знати для проведення цього виду тестування?

Що тестувати?

Чим тестувати?

Коли тестувати?

Підходи у виборі тестів

Аналіз впливу (Impact analysis)	Визначити функціонал або сукупність функцій, області (affected areas), які зазнали змін
Аналіз ризиків (Risk analysis)	Знайти модулі або області, де можуть виникнути ризики/дефекти
Вибір критичних сценаріїв (Critical users scenarious)	Виділити найбільш важливі/використовувані сценарії для користувача

Аналіз впливу (Impact analysis)

1. Визначте компоненти/області для нового функціоналу.
2. Проаналізуйте, які компоненти взаємодіють із новим функціоналом.
3. Проаналізуйте, яка загальна поведінка застосунку може бути схильна до впливу – продуктивність, UI, мобільна версія.
4. Поспілкуйтеся з розробником про неявні залежності функціоналу в коді.

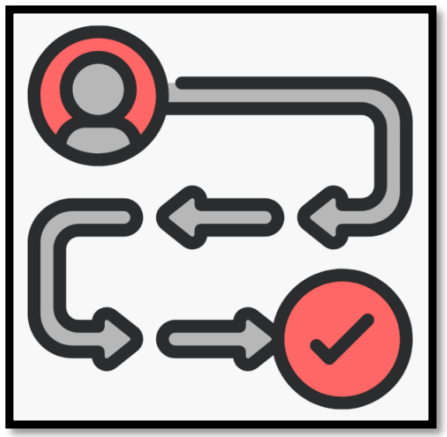
Аналіз ризиків (Risk analysis)

Який функціонал матиме більше дефектів?

1. Розроблений поспіхом, новим розробником, недосвідченим розробником.
2. Не покритий модульними тестами.
3. Функції зі складною бізнес-логікою або технологічним стеком.
4. Функціонал, у якому зазвичай знаходимо багато дефектів.

Такі області мають бути охоплені регресійним тестуванням!

Вибір критичних сценаріїв (Critical users scenarios)



- Розробляйте end2end тести, щоб покрити основні сценарії використання застосунку.
- Сфокусуйтеся на позитивних сценаріях.
- Проведіть рев'ю сценаріїв користувачами / BA.
- Отримайте статистику щодо того, які компоненти використовуються більше, якщо ви можете це зробити.

Об’єм регресії

- Як правило, у команді QA є готовий список тестів для обсягу регресії, який постійно оновлюється.
- Набір тестів для регресії може включати як функціональні, так і нефункціональні тести.
- Поширеною практикою є розподіл функціоналу між членами команди, які раніше не працювали з даними функціоналом для уникнення ефекту пестициду.
- Також, як і в регулярному тестуванні, важливим є розставлення пріоритетів при регресійному тестуванні. Це скоротить набір перевірок.

Склад тестів для регресії

- Безпосередньо саме регресійне тестування – повторне виконання всіх тестів, написаних і проведених раніше. Вони виконуються за вже існуючими тест-кейсами незалежно від того, чи були в ході їх проходження знайдені баги, чи ні.

- Тест-кейси після ретестінгу. Проводяться для перевірки виправлення виявленого та відкритого раніше багу.

- Тестування в новому білді вже виправлених багів у старих білдах. Це виконується для того, щоб перевірити, чи оновлення білда не відновило старих дефектів.

Коли проводити регресійне тестування

- Після кожної відправки кода (commit) розробником?
- Після кожної розгортання (deployment) на якомусь з середовищ?
- В кінці спринта?
- Перед релізом?

Типова стратегія регресійного тестування

Тріггер	Вид тестів	Час виконання	Час на аналіз звітів
Після кожного комміту	Компонентні тести (Unit tests)	~ 1 хв	~ 1 хв
Після кожного розгортання на середовищі	Автоматизований димове тестування	0.5 - 1 год	~5 - 10 хв
Після завершення всіх сторів в спрінті	Цільова регресія	3 - 4 години мануального тестування	
Кожної ночі / 1-2 рази на спрінт	Автоматизована регресія	1 - 6 годин	20 - 60 хв
Перед релізом	Повна регресія	Дні/Тижні/Місяці .. мануальних тестувальників	

Regression vs Smoke

Регресія	Смоук
Регресійне тестування використовується для перевірки глибокої працездатності системи.	Димове тестування використовується для поверхневої оцінки працездатності системи.
Регресійне тестування займає багато часу та дає великий розгорнутий звіт про якість.	Димовий тест виконується швидко, щоб підтвердити, прийняти чи відхилити збірку.
Виконується як вручну, так і засобами автоматизації.	

Як організувати регресійну бібліотеку

1. Не всі нові тести слід додавати до бібліотеки регресії.

2. Структура важлива:
   - за компонентами продукту;
   - за рівнем тестування: Smoke, Acceptance, Regression
   - наскрізні сценарії (End-to-end scenarios).

3. Пріоритети тестування повинні бути визначені, щоб у всієї команди було загальне розуміння.
4. Повинна бути визначена процедура підтримки бібліотеки регресії: тести повинні оновлюватися та видалятися.
5. Матриця простежуваності до автоматизованих тестів повинна бути складена.

Автоматизація регресійного тестування

1 тест:

- 5-10 хвилин для виконання вручну;
- 4-8 годин на автоматизацію + підтримка.

Щонайменше 24 рази нам потрібно запустити тест, щоб окупити автоматизацію.

- При тестуванні 1 раз на 2-тижневий спринт гроші повертаємо через рік.
- Якщо тест запускати один раз на 3-місячний реліз, цього може і не статися.

Автоматизація регресійного тестування

Що автоматизувати:

- димове тестування;
- тести, які ви повторюєте кожні 1-2 тижні;
- бекенд-тести (набагато довше запускати вручну);
- однакові дії з користувацьким інтерфейсом, тести керовані даними;

Що не можна автоматизувати:

- всі регресії для проектів із релізами не щомісяця;
- тести, що проводяться кожні 3-6 місяців.

Що ми сьогодні вивчили

Regression testing

- Регресійне тестування – один із видів тестування, пов'язаного зі змінами, при якому перевіряється застосунок після внесених змін у коді (новий функціонал та/або виправлення дефекту) на предмет, чи не вплинули вони на інші модулі та чи не викликали нові баги.

- Основні цілі регресії:
  - перевірка нового функціоналу у складі вже сформованої робочої системи.
  - забезпечення якості ПЗ перед релізом.

- Підходи регресійного тестування:
  - із застосуванням компонентного підходу;
  - із застосуванням сортування кейсів за пріоритетом;
  - за допомогою автоматизованого підходу.

Підсумки

Що одне, найголовніше, ви дізнались сьогодні?

Дякую за увагу! До нових зустрічей!

Інформаційний відеосервіс для розробників програмного забезпечення

TestProvider Pearson CyberBionic systematics Microsoft Partner Windows Azure EBA