5. Sadli A. Using the python library to create simple game animations. *International journal of management science and information technology*. 2022. № 2, vol. 2. P. 21–31. URL: https://doi.org/10.35870/ijmsit.v2i2.699 (date of application: 30.03.2023).

ТЕХНОЛОГІЧНІ ОСОБЛИВОСТІ ВИКОРИСТАННЯ ФРЕЙМВОРКУ GRADLE ДЛЯ ЗБИРАННЯ ПРОЄКТІВ

Галас Анатолій Віталійович
магістрант спеціальності 014.09 Середня освіта (Інформатика),
Тернопільський національний педагогічний університет імені Володимира Гнатюка,
galas_av@ fizmat.tnpu.edu.ua
Василенко Ярослав Пилипович
викладач кафедри інформатики та методики її навчання,
Тернопільський національний педагогічний університет імені Володимира Гнатюка,
yava@fizmat.tnpu.edu.ua

Технологічні особливості використання фреймворку Grandle для збирання проєктів є дуже актуальною темою серед інформаційних технологій сьогодення. Зараз на ринку праці велика кількість розробників програмного забезпечення шукає швидкі та ефективні способи збирання своїх проєктів.

Фреймворк Gradle – це потужний інструмент для автоматизації збірки, тестування та розгортання проєктів, що базуються на різних технологіях.

Основна перевага Gradle полягає в його гнучкості та можливостях налаштування, що дозволяє розробникам використовувати його для будь-яких проєктів, від найпростіших до найскладніших.

Завдяки підтримці різних мов програмування та технологій, включаючи Java, Kotlin, Groovy, Scala та інші, Gradle є універсальним інструментом для збирання проєктів будь-якої складності.

Використання Grandle дозволяє збирати проєкти на будь-якій платформі, що робить його дуже привабливим для розробників програмного забезпечення, які працюють з різними платформами.

Отже, дослідження технологічних особливостей використання фреймворку Gradle є актуальним для розробників, що працюють зі складними проєктами та бажають збільшити продуктивність та ефективність роботи.

Gradle – це інструмент збірки, що базується на принципах Apache Ant та Apache Maven. Він був створений для полегшення та автоматизації процесу збірки, тестування та розгортання програмного забезпечення [1].

Gradle є прикладом програмування на основі залежностей: ви визначаєте завдання та залежності між завданнями. Gradle гарантує, що ці завдання виконуються в порядку їх алежностей.

Скрипти збірки та плагіни налаштовують цей граф залежностей. Нижче подано граф завдань життєвого циклу (базовий цикл побудови проєкту, який доступний уже після встановлення Gradle у ваш проєкт) (рис. 1.) [3].

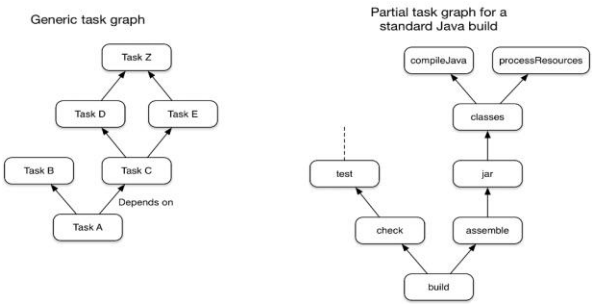

Рис. 1. Граф виконання завдань при побудові проєктів

У роботі [4] подано візуальне порівняння продуктивності Gradle та Maven (рис. 2).

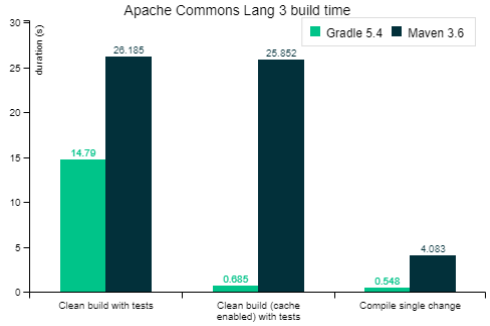

Рис. 2. Порівняння продуктивності Gradle та Maven Gradle

Gradle підтримує багато основних IDE, включаючи Android Studio, Eclipse, IntelliJ IDEA, Visual Studio 2021 та XCode. Можна також викликати Gradle через інтерфейс командного рядка у вашому терміналі або через сервер безперервної інтеграції [2].

```
dependencies {
    testImplementation 'junit:junit:4.13.2'
}

docker {
    name "${project.group}/${project.name}"
    files dockerDir // directory containing Dockerfile
}

checkstyle {
    toolVersion = '8.43'
    configFile = file('config/checkstyle/checkstyle.xml')
    ignoreFailures = true
    showViolations = true
    sourceSets = [sourceSets.main]
}

task runTests(type: Test) {
    useJUnit()
    testLogging {
        events "passed", "skipped", "failed"
    }
}
```

Рис. 3. Підключення до проєкту різноманітних інструментів розробки

Однією з основних переваг Gradle є його здатність до інкрементальної збірки. Це означає, що він збирає тільки ті частини проєкту, які зазнали змін, що зменшує час збірки та полегшує розробку.

Іншою важливою перевагою Gradle є підтримка різних мов програмування. Він підтримує Java, C++, Python, Ruby, Kotlin та багато інших мов програмування, що дозволяє розробникам працювати з будь-якими мовами програмування, які вони виберуть [2].

Крім того, Gradle має вбудовану систему кешування, що дозволяє зменшити час збірки проєкту, а також дозволяє зберегти результати попередніх збірок для використання у майбутніх збірках [2].

Одним із недоліків Gradle є складність налаштування. Він має багато параметрів та можливостей, що можуть призвести до складності налаштування. Однак, з розвитком спільноти Gradle, з'являється все більше матеріалів та документації для його використання.

Gradle дозволяє використовувати готові плагіни, які допомагають в роботі з різними інструментами, такими як JUnit для тестування, Checkstyle для перевірки якості коду, а також Docker для контейнеризації додатків (рис. 3). Крім того, Gradle підтримує різні типи залежностей, включаючи залежності Maven та Ivy, що дозволяє користувачам легко додавати залежності до своїх проєктів.

Для того, щоб підключити сторонні API використовується функція dependencies, яку ми бачимо у рядку:

testImpementaion 'junit:junit:4.13.2'

Це означає, що дане API буде використовуватись для тестування.

Іншим важливим аспектом Gradle є його підхід до скриптування. Gradle використовує мову Kotlin DSL або Groovy DSL для написання скриптів збирання. Це дає розробникам можливість налаштувати свої проєкти з використанням декларативного підходу, що дозволяє писати менше коду та зменшує кількість помилок [1].

Дослідження технологічних особливостей використання фреймворку Gradle для збирання проєктів має практичне значення для розробників програмного забезпечення, які шукають більш оптимальні і ефективні способи збірки своїх проєктів. Результати будуть корисними при плануванні процесу збирання проєкту, виборі підходящих плагінів та налаштуванні конфігурації Gradle для досягнення максимальної продуктивності та ефективності в процесі розробки програмного забезпечення.

Крім того, знання технологічних особливостей фреймворку Gradle може бути корисним при вивченні та викладанні курсів з програмування, де використовується цей фреймворк. Також результати дослідження можуть бути використані в наукових дослідженнях у галузі програмної інженерії.

Фреймворк Gradle є потужним інструментом для збирання проєктів, який має безліч переваг порівняно з іншими інструментами. Використання Gradle дозволяє розробникам легко налаштовувати процес збирання проєктів, встановлювати залежності, налаштовувати тестування та забезпечувати інші необхідні процеси. Фреймворк підтримує також розробку мультиплатформних.

Однак, використання Gradle може бути вимогливим до ресурсів і потребувати додаткових знань для ефективного використання. Крім того, він має дещо складну структуру файлів конфігурації, що може бути складною для початківців.

В цілому, використання фреймворку Gradle дозволяє підвищити продуктивність розробки та збірки проєктів, що робить його важливим інструментом для розробників програмного забезпечення. Розробники повинні мати достатні знання та ресурси для використання цього інструменту, щоб досягти максимальної користі від його функціоналу.

Список використаних джерел

1. Мушко Б., Доктер Х. Gradle у дії. Нью-Йорк. Manning Publications Co., 2014. С. 5–15.
2. Вступ до офіційної документації Gradle. URL: https://docs.gradle.org/current/userguide/userguide.html (дата зверенення: 01.04.2023).
3. Основи життєвого циклу Gradle. Офіційна документація Gradle. URL: https://docs.gradle.org/current/userguide/build_lifecycle.html (дата звернення: 01.04.2023).
4. Порівняння збірників проєктів на офіційному сайті Gradle у розділі Gradle vs Maven. URL: https://gradle.org/maven-vs-gradle (дата звернення: 01.04.2023).

ВИКОРИСТАННЯ ЦИФРОВОГО ІНСТРУМЕНТУ WOOCLAP В ОСВІТНЬОМУ ПРОЦЕСІ

Генсерук Галина Романівна
кандидат педагогічних наук, доцент кафедри інформатики та методики її навчання,
Тернопільський національний педагогічний університет імені Володимира Гнатюка,
genseruk@tnpu.edu.ua
Андрійчук Соломія Юріївна
студентка спеціальності 014 Середня освіта (Мова і література (англійська)),
Тернопільський національний педагогічний університет імені Володимира Гнатюка,
solomia2005y@gmail.com

Щоб протистояти спалаху коронавірусної хвороби 2019 (Covid-19), світова система освіти мала швидко адаптуватися. Постійний розвиток цифрових навчальних рішень був значно прискорений раптовим переходом від очного до дистанційного навчання. Станом на 1 квітня 2020 року 173 країни закрили свої школи та університети, і понад 1 мільярд студентів постраждали від цих рішень [3]. В Україні в березні 2020 року дистанційне навчання було запроваджено у всіх закладах освіти. Розробка стратегії дистанційного навчання вимагала спеціальних педагогічних підходів, включаючи дизайн курсу та методи зворотного зв’язку [2]. Залежно від навчального закладу, підтримка та заохочення для педагогів у розвитку дистанційного навчання була різною. Враховуючи гостроту пандемії навчальні заклади не мали часу, щоб належним чином забезпечити відповідне обладнання та навчити педагогів використовувати онлайн інструменти в освітньому процесі. Протягом цього періоду студенти також зіткнулися з кількома труднощами, такими як проблеми з підключенням до мережі Інтернет, відсутність або невідповідне обладнання та обмежений доступ до відповідного навчального середовища. Загалом, викладачам і студентам потрібно було налаштувати зворотній зв’язок. Незважаючи на всі ці труднощі, стратегія дистанційного навчання широко поширилися в практиці викладання, багато викладачів і студентів відчули цю нову форму навчання. Сьогодні педагоги мають більше часу для того щоб навчитися використовувати цифрові технології для навчання та спеціально переосмислити свої заняття для дистанційного навчання. Викладачі закладів вищої освіти визнають, що вони дізналися про дистанційне навчання під час цієї кризи більше, ніж за останні 10 років.

Сьогодні важливим є осмислення та можливість розробки нових методів навчання, які передбачають використання цифрових технологій. Серед багатьох