|  |  |
| --- | --- |
| ДИСЦИПЛИНА | Создание программного обеспечения |
| ИНСТИТУТ | Институт перспективных технологий и индустриального программирования |
| КАФЕДРА | Кафедра индустриального программирования |
| ВИД УЧЕБНОГО МАТЕРИАЛА | Лекция |
| ПРЕПОДАВАТЕЛЬ | Макиевский Станислав Евгеньевич |
| СЕМЕСТР | 3 семестр, 2024-2025 гг. |

# 

## Лекция №8

**Подведение итогов**

**Зачем нужна автоматизация тестирования?**

Автоматизация тестирования – это процесс замены ручного выполнения тестов на автоматическое выполнение с использованием специальных инструментов и скриптов. Автоматизация тестирования позволяет повысить эффективность и надежность тестирования, сократить время тестирования и улучшить качество продукта.

Преимущества автоматизации включают:

Быстроту и повторяемость: автотесты выполняются быстрее и могут быть запущены неограниченное количество раз.

Раннее обнаружение дефектов: автотесты позволяют выявлять проблемы на ранних этапах разработки.

Эффективное использование ресурсов: ручное тестирование может быть заменено автоматическим, освобождая время и ресурсы для других задач.

Увеличение покрытия тестирования: автоматизация позволяет протестировать большее количество сценариев, что улучшает качество продукта.

**Какие задачи можно автоматизировать?**

Автоматизация подходит для различных видов тестирования:

* Функциональное тестирование: проверка соответствия функциональности продукта требованиям.
* Регрессионное тестирование: проверка наличия новых дефектов после внесения изменений.
* Нагрузочное тестирование: оценка производительности и стабильности системы при больших нагрузках.
* Интеграционное тестирование: проверка взаимодействия компонентов системы.
* UI-тестирование: проверка пользовательского интерфейса на соответствие дизайну и функциональности.

**Выбор инструментов для автоматизации**

При выборе инструментов для автоматизации следует учитывать такие факторы, как:

* Язык программирования: выберите язык, на котором вы и ваша команда хорошо разбираетесь.
* Тип приложения: некоторые инструменты лучше подходят для веб-приложений, другие для мобильных.
* Уровень поддержки: убедитесь, что выбранный инструмент активно поддерживается и обновляется.
* Интеграция: интеграция с другими инструментами разработки и CI/CD платформой.

Популярные фреймворки и инструменты для автоматизации тестирования включают:

* Selenium WebDriver для веб-тестирования.
* Appium для тестирования мобильных приложений.
* JUnit и TestNG для тестирования на языке Java.
* Pytest и Unittest для тестирования на языке Python.
* Cucumber для поведенческого тестирования с использованием языка Gherkin.

**Преимущества и ограничения автоматизации**

Хотя автоматизация тестирования предоставляет множество преимуществ, она также имеет свои ограничения:

* Невозможность заменить полностью ручное тестирование: некоторые виды тестирования все еще лучше выполнять вручную.
* Значительные затраты на разработку и поддержку автотестов.
* Необходимость постоянного обновления автотестов после изменений в приложении.

Когда целесообразно автоматизировать тестирование, а когда лучше использовать ручное тестирование?

Автоматизация тестирования целесообразна в следующих случаях:

* При необходимости частого и повторяемого тестирования одного и того же функционала.
* При наличии стабильной функциональности, подверженной частым изменениям.
* При большом объеме данных для тестирования.

Однако автоматизация не всегда является лучшим решением, особенно в следующих случаях:

* При тестировании нового функционала или нестабильной версии приложения.
* При тестировании маленьких проектов с ограниченными ресурсами.
* При сложности написания автотестов для определенной функциональности.
* Критерии приемки
* Критерии приемки определяют, когда продукт считается готовым для выпуска.

Они включают такие аспекты, как:

* Прохождение всех автоматических тестов без ошибок.
* Задокументирование всех найденных ошибок и их исправление.
* Полное соответствие продукта требованиям.
* Метрики качества

Для оценки эффективности тестирования и качества продукта используются различные метрики. Некоторые из них включают:

* Количество выполненных автоматических тестов.
* Процент покрытия автоматизированных тестов кода приложения.
* Количество и прирост найденных и исправленных дефектов.
* Время, затраченное на выполнение автоматических тестов.
* Отчетность

Отчетность играет важную роль в автоматизации тестирования. Отчеты представляют результаты тестирования и содержат информацию о количестве пройденных и неудачных тестов, найденных дефектах и покрытии кода. Отчеты помогают команде разработки принимать взвешенные решения о качестве продукта и необходимости доработок.

Введение в автоматизацию тестирования предоставляет множество преимуществ, но требует тщательного планирования, выбора подходящих инструментов и правильного подхода к автоматизации. Автоматизация тестирования должна быть решением, которое соответствует потребностям конкретного проекта и команды разработки. Она позволяет повысить эффективность тестирования и обеспечить высокое качество продукта, что является ключевым фактором для успеха проектов в современном мире программной инженерии.

**Вопросы для самопроверки**

1. Что такое проект и что представляет собой проектная деятельность?
2. Какие требования должны соблюдаться при подготовке отчетности о проекте?
3. Что такое системы контроля версий и зачем они нужны?
4. Какие разновидности архитектур VCS существуют?
5. В чем отличия локальной системы контроля версий от централизованной и распределенной?
6. Какие команды используются в системе контроля версий Git: init, config, status, add, commit, push?
7. Какова роль Тимлида в организации и почему она важна для команды?
8. Как сформировать эффективную команду: процесс подбора специалистов и распределение ролей?
9. Как обеспечить открытую и эффективную коммуникацию в команде?
10. Как развивать команду: методы поощрения сотрудничества, создание атмосферы доверия и укрепление связей?
11. Что такое бизнес-цели и почему они важны для организации?
12. Какие типичные бизнес-цели могут быть у различных компаний?
13. Какие методологии используются при разработке карты бизнес-целей?
14. Почему важно выстраивать понятную и четкую структуру целей?
15. Какие методы визуализации данных существуют и как они влияют на понимание информации?
16. Что такое путевая карта пользователя и каким образом она помогает понять опыт пользователя?
17. Как создать путевую карту пользователя и какие преимущества она имеет?
18. В каких сферах применяются путевые карты пользователя и какие примеры реальных карт существуют?
19. Какие преимущества предоставляют карты пользовательских историй и какова их структура?
20. Как совместно использовать путевые карты и карты пользовательских историй для анализа и планирования проектов?
21. Что представляет собой BPMN и для чего он используется?
22. Как BPMN позволяет моделировать потоки данных и информации между задачами и событиями?
23. Что представляет собой серверная архитектура в разработке программного обеспечения?
24. Какая роль у сервера в клиент-серверной модели взаимодействия?
25. Почему важна хорошая архитектура на серверной стороне и как она влияет на производительность и масштабируемость приложения?
26. Что такое монолитные архитектуры и какова их структура?
27. Что такое микросервисные архитектуры и что делает их особенными в разработке?
28. Что такое серверы приложений и как они обрабатывают запросы?
29. В чем различия между различными фреймворками для разработки программного обеспечения?
30. Какие типы и разновидности проектов существуют?
31. Как осуществляется поддержка и обслуживание программного обеспечения?
32. Зачем нужны стандарты тестирования и как они повышают эффективность работы тестировщиков и разработчиков?
33. Какие популярные стандарты тестирования существуют и для чего они предназначены?
34. Какое значение имеет соблюдение стандартов в процессе тестирования и какие вызовы это может представить?
35. Какова роль тестировщика в команде разработки и как он взаимодействует с другими участниками команды?
36. Какое значение имеет коммуникация и коллаборация для повышения качества продукта?
37. Как решать конфликты и управлять ожиданиями в команде разработки?
38. Что такое тестовый сценарий и какова его структура и цель?
39. Какие техники разработки тестовых сценариев существуют и какие подходы применяются?
40. Какие практические примеры разработки тестовых сценариев для различных типов приложений существуют?
41. Что такое уровни иерархии в проекте и как они связаны с определением требований?
42. Какие инструменты используются для визуализации и документирования требований в проекте?
43. Каким образом происходит процесс управления изменениями в системе контроля версий Git?
44. Какие преимущества и недостатки имеют различные типы архитектур VCS (локальные, централизованные, распределенные)?
45. Какие ключевые принципы руководят командой разработчиков под руководством Тимлида?
46. Каким образом формируется и поддерживается атмосфера доверия в команде?
47. Какие методы анализа данных используются для выявления и формулирования бизнес-целей?
48. Какая роль дизайн-мышления в разработке карт бизнес-целей и пользовательских путевых карт?
49. Каким образом команда разработки может использовать анализ данных для оптимизации проектных решений?
50. Какая важность управления зависимостями в проекте и какие инструменты помогают в этом?
51. Что такое Continuous Integration (CI) и Continuous Delivery (CD) в контексте систем контроля версий?
52. Какие особенности сопровождения и поддержки монолитных приложений по сравнению с микросервисами?
53. Какая роль тестирования в различных этапах жизненного цикла проекта?
54. Какие методы тестирования используются для обеспечения качества программного обеспечения?
55. Каким образом тестировщики и разработчики могут сотрудничать для обнаружения и исправления ошибок?
56. Каковы основные преимущества и недостатки автоматизированного тестирования?
57. Что такое регрессионное тестирование и зачем оно используется в проекте?
58. Какие основные архитектурные шаблоны (паттерны) используются в разработке программного обеспечения?
59. Какие техники обеспечивают безопасность данных и защиту от несанкционированного доступа?
60. Что такое DevOps и как он способствует улучшению совместной работы между разработкой и операциями?
61. Какие методы управления рисками применяются в проекте и как они помогают снижать вероятность возникновения проблем?
62. Какие стратегии управления временем и ресурсами используются в проекте для достижения бизнес-целей?
63. Какие основные шаги включает в себя процесс разработки плана проекта?
64. Что такое инкрементальное и итеративное развитие в процессе разработки программного обеспечения?
65. Какие методы управления качеством используются для отслеживания выполнения требований проекта?
66. Что такое функциональное тестирование и как оно помогает проверить соответствие продукта требованиям?
67. Какие основные этапы включает процесс разработки плана тестирования?
68. Каким образом управление конфигурацией влияет на проектную деятельность?
69. Какие методы анализа производительности используются для оптимизации программного обеспечения?
70. Что такое контроль качества и как он взаимодействует с процессами разработки и тестирования?