CI/CD пайплайн

11 марта 2026 г.

Что мы имеем после прошлой лекции

Автоматизация через webhook работает:

- Git push → запуск тестов, автоматический деплой

Но:

- Изменения вебхук сервера требуют ручного вмешательства
- Нет простого доступа к логам скриптов, нет статуса деплоя

Индустриальные решения

В реальных проектах используют готовые платформы:

- **Jenkins** — самый популярный, self hosted
- **GitHub Actions** — встроен в GitHub, есть бесплатный режим
- **GitLab CI** — встроен в GitLab, есть self hosted режим
- Azure DevOps, TeamCity, CircleCI...

Общая идея: вместо своего webhook сервера используем платформу (отдельное приложение), которая позволяет выполнять скрипты по событиям в Git репозитории.

Преимущества и недостатки платформ

Плюсы:

- Надежность
- Масштабируемость
- Удобство
- Интеграции
- Переиспользуемые скрипты

Минусы:

- Дополнительная сложность
- Свой язык автоматизации
- Ограниченные возможности

Концепция Pipeline

У всех этих платформ есть общая концепция — **Pipeline**

Pipeline — последовательность шагов автоматизации, задаваемая в конфигурации системы.

Типичные шаги:

1. получить код из git репозитория;
2. собрать приложение, при необходимости;
3. запустить тесты;
4. задеплоить на сервер.

GitHub Actions

GitHub Actions реализует концепцию pipeline через описание процессов автоматизации в YAML файлах.

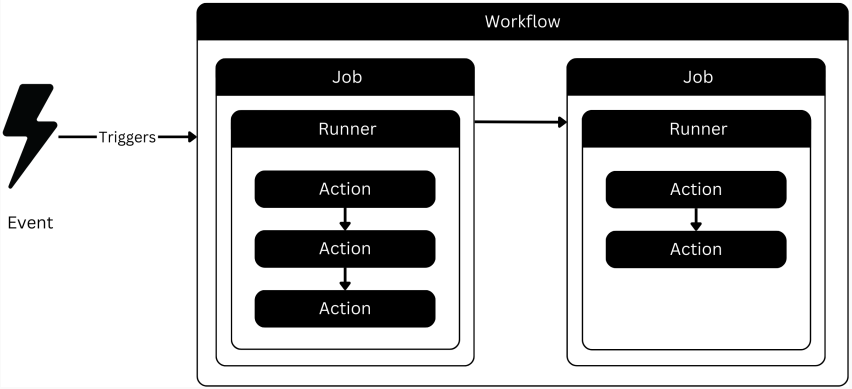
Базовые понятия:

- **Workflow** — описание pipeline в YAML файле;
- **Job** — группа связанных шагов (например, "тестирование");
- **Step** — shell скрипт или вызов action;
- **Action** — готовый набор шагов автоматизации (например, "протестируй dotnet приложение");
- **Runner** — сервер, выполняющий workflow.

Структура: Workflow содержит Jobs, Job содержит Steps

Триггеры: push, pull request, расписание, ручной запуск

GitHub Actions Components



Демонстрация: Переносим webhook на GitHub Actions

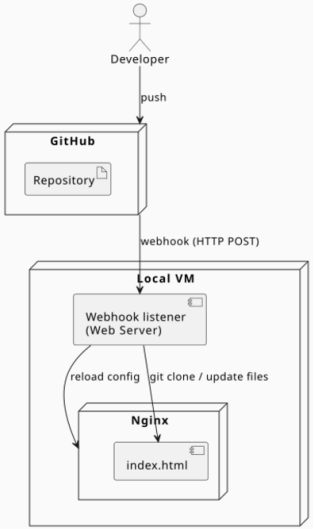
Задача: воспроизвести логику нашего webhook сервера в GitHub

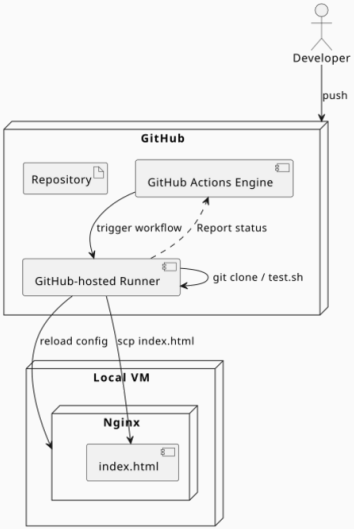
Что покажем:

- Как добавить workflow в GitHub репозиторий
- Описание workflow
- Работа с секретами в GitHub Actions

Демо-репозиторий:
https://github.com/prafdin/devops-demo-website/tree/cicd-demo

Webhook vs GitHub Actions pipeline





Что произойдет, если в основную ветку попадет некорректный код?

Последствия некорректного кода в основной ветке

- Сломается деплой;
- На прод попадет некорректный код;
- Другие разработчики получают нестабильное состояние проекта;

Основная ветка без защиты перестает быть надежным источником о состоянии проекта.

Quality gate для интеграции кода в основную ветку

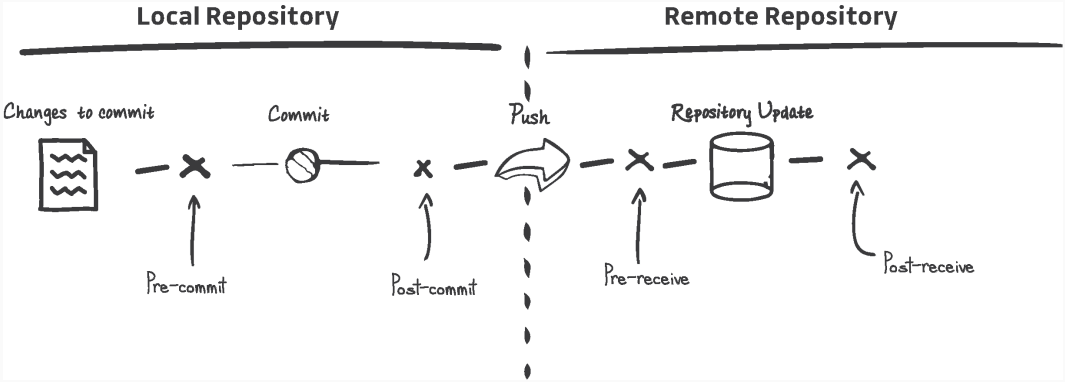
Цель: основная ветка должна отображать последнее исправное состояние проекта.

Quality gate — механизм проверки изменений перед их интеграцией.

Задача quality gate для кода:

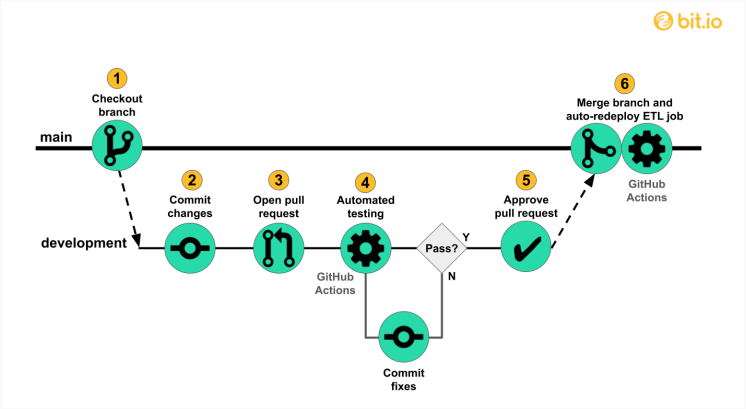
- проверять соответствие требованиям проекта: стайлчекер, линтер, тесты;
- не допускать интеграцию некорректных изменений в основную ветку.

Вариант QG: git hooks



https://blog.gitguardian.com/git-hooks...

Вариант QG: pull request checks



https://innerjoin.bit.io/making-a-simple...

Демонстрация: настраиваем проверки для интеграции кода

Задача: добавить проверки при интеграции кода в основную ветку репозитория

Что покажем:

- Новый workflow для проверки при интеграции
- Запрет работать с веткой напрямую

Демо-репозиторий:
https://github.com/prafdin/devops-demo-website/tree/cicd-pullrequest-demo

Наш pipeline — это CI/CD pipeline

Поздравляем! Мы только что создали CI/CD pipeline

CI/CD — собирательный термин для задач по автоматизации разработки:

- **CI** — Continuous Integration (непрерывная интеграция)
- **CD** — Continuous Deployment (непрерывное развертывание)

Что такое CI и CD

CI задачи это шлюзы:

- Линтинг
- Тесты
- Сканеры безопасности

Цель: убедиться что код соответствует требованиям

CD задачи это преобразования:

- Сборка приложения
- Публикация и обновление приложения

Цель: изменить состояние артефакта

Материалы по теме

- https://martinfowler.com/articles/continuousIntegration.html
- «Грокаем Continuous Delivery» Кристи Уилсон
- «GitHub Actions in Action» Michael Kaufmann
- «Continuous Deployment» Valentina Servile