Методы и средства сборки и развертывания ПО, 25

ФИО преподавателя: Смирнов Михаил Вячеславович

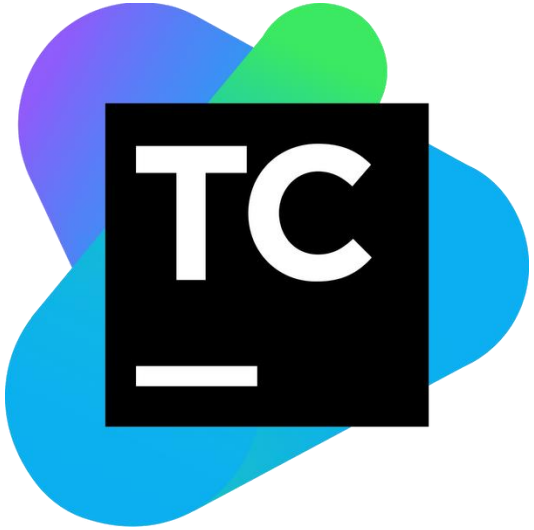
e-mail: smirnovmgupi@gmail.com

Лекция 12

Инструменты создания и запуска конвейеров CI/CD. Инструмент CI/CD Jenkins

Обзор инструментов CI/CD

Jenkins



Инструменты CI/CD можно разделить на категории:

Серверные (Self-hosted): Устанавливаются на подготовленную инфраструктуру (Jenkins, GitLab CI, TeamCity).

Облачные (SaaS): Предоставляются как услуга (GitHub Actions, GitLab.com, CircleCI, Travis CI). Быстрый старт, масштабируемость, меньше администрирования.

Встроенные в платформы: Часть экосистемы (GitHub Actions, GitLab CI).

Знакомство с Jenkins

Je

Jenkins - это open-source сервер автоматизации развертывания. Он написан на Java и используется для реализации CI/CD-конвейера. «Форк» проекта Hudson (2005). Стал де-факто стандартом CI/CD.

Ключевые особенности:

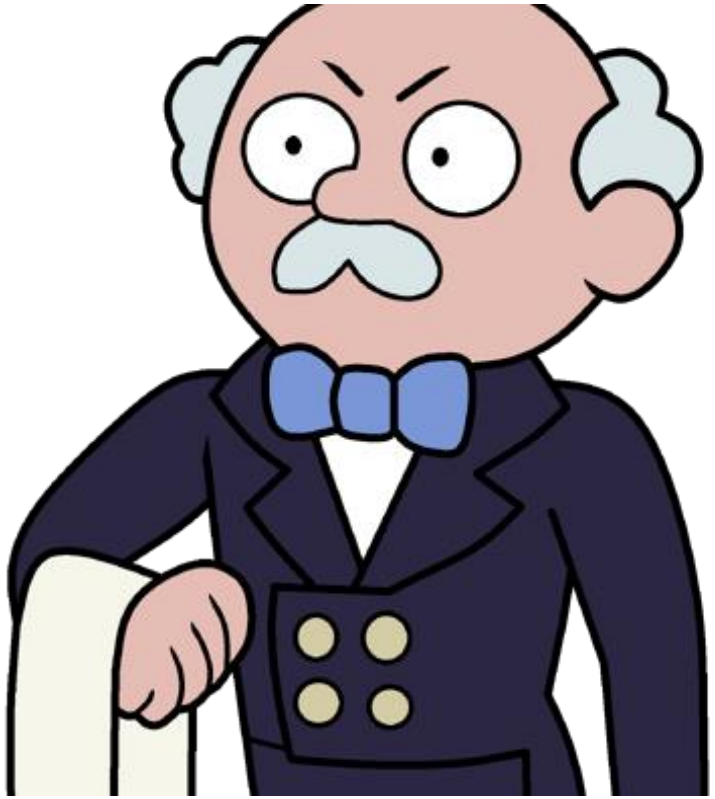
Гибкость. Огромное количество плагинов (1500+).

Расширяемость. Легко интегрируется с любыми инструментами DevOps.

Распределенность: Поддержка агентов (нод) для распределения нагрузки.

Бесплатный и с открытым исходным кодом.

Архитектура Jenkins



Master (Контроллер):

Управляет конвейерами, планирует задачи.

Предоставляет web-интерфейс.

Хранит конфигурации и результаты сборок.

Не должен выполнять тяжелые задачи сборки.

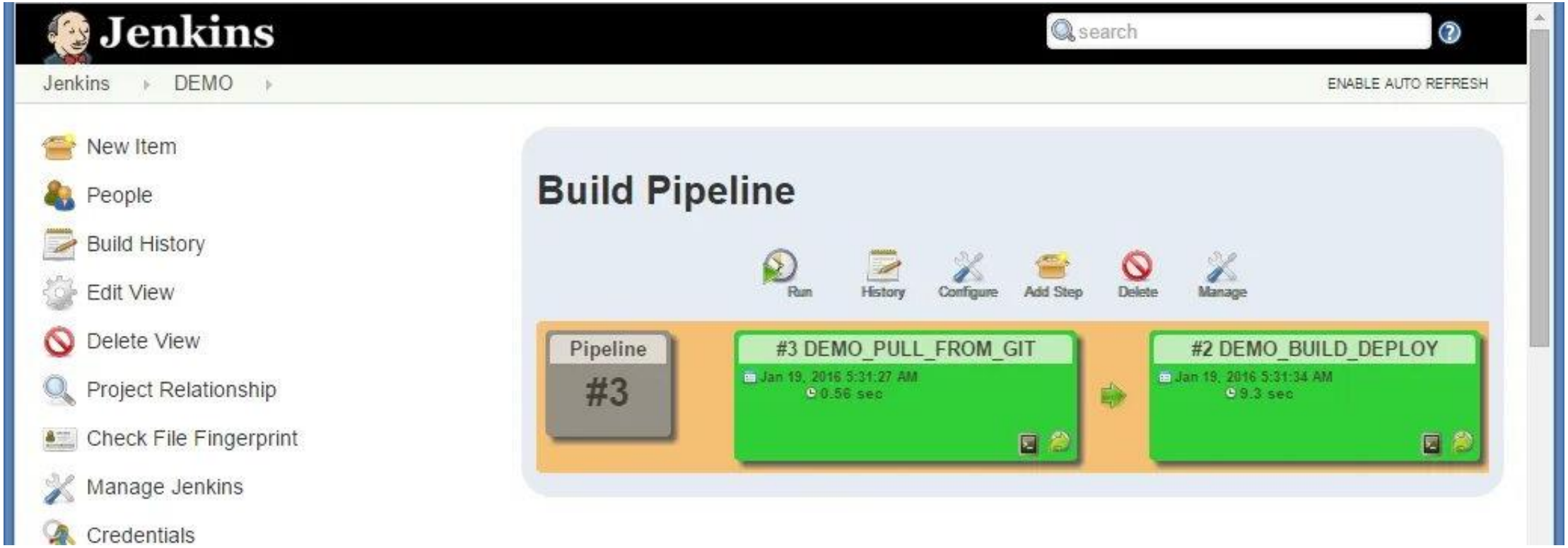
Agent (Нод):

Исполнительная единица. Получает задания от Master.

Выполняет сборку, тесты, развертывание.

Может работать на разных ОС и с разным окружением (Docker, Windows, Linux).

Веб-интерфейс Jenkins



Основные понятия: Job, Build, Workspace

Job (Задача). Конфигурация, которая описывает *что* и *как* делать. Это может быть сборка, тест и т.д.

Build (Сборка). Один экземпляр Задачи. Имеет номер, статус (успех/неудача), логи, артефакты.

Workspace (Рабочее пространство):. Каталог на компьютере-ноде, где выполняется конкретная сборка.

Процесс: Jenkins запускает Сборку для определенной Задачи в нужном Рабочем пространстве.

Типы Задач в Jenkins

Freestyle project. Классический, гибкий тип. Настройка через веб-форму. Подходит для простых задач.

Pipeline. Стандартный тип Задач Jenkins. Процесс описывается кодом (Jenkinsfile).

Multibranch Pipeline. Автоматически создает конвейер для каждой ветки в репозитории.

Folder. Для организации задач в группы.

Multi-configuration project. Для запуска сборок на разных платформах/конфигурациях.

Jenkins как код (Jenkinsfile)

Jenkinsfile - это текстовый файл, который содержит определение конвейера. Хранится вместе с кодом в репозитории.

Существует два стиля управления Jenkinsfile:

Декларативный. Простой, четко структурированный синтаксис.

Скриптовый. Основан на Groovy, дает полную гибкость и контроль для сложной логики.

```
pipeline { agent any stages {
stage('Hello World') { steps {
script { println('Hello, World')
} } } } }
```

Пример декларативного конвейера в Groovy

```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                git
'https://github.com/your-repo.git' //
Получить код
            }
        }
        stage('Build') {
            steps {
                sh 'mvn clean compile'
//Собрать проект (Maven)
            }
        }
        stage('Test') {
            steps {
                sh 'mvn test' //
Запустить тесты
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying to
staging...' }
            }
    }
}
```

Плагины в Jenkins

Плагины - это дополнения, которые расширяют функциональность Jenkins.

Управление исходным кодом. Git, GitHub, Bitbucket.

Сборка и тестирование. Maven, Gradle, JUnit, pytest.

Контейнеризация. Docker, Kubernetes.

Уведомления. Email, Slack, Telegram.

Визуализация. Blue Ocean, Dashboard View.

Manage Jenkins -> Plugins.

Использование Jenkins

✅ **Открытость и сообщество:** Бесплатный, огромная база знаний, плагины на все случаи.
✅ **Гибкость:** Можно автоматизировать почти что угодно.
✅ **Зрелость и надежность:** Проверен годами использования в тысячах компаний.
✅ **Кроссплатформенность:** Работает везде, где есть Java.
✅ **"Infrastructure as Code":** Конвейеры в Jenkinsfile обеспечивают воспроизводимость.

❌ **Сложность администрирования:** Требует обслуживания, обновлений, мониторинга.
❌ **Устаревший интерфейс:** Базовый UI воспринимается как старомодный.
❌ **"Состояние сервера":** Конфигурация Master'а может быть хрупкой. Необходимо резервное копирование.
❌ **Сложность для новичков:** Обилие опций и плагинов может ошеломить.
❌ **Перформанс:** Может "проседать" при большом количестве задач и плагинов.

Jenkins и альтернативы

Критерий	Jenkins	GitLab CI	GitHub Actions	CircleCI
Тип	Self-hosted	Встроенный / SaaS	SaaS	SaaS
Конфигурация	Jenkinsfile (Groovy)	.gitlab-ci.yml	Workflow (YAML)	config.yml
Гибкость	Очень высокая	Высокая	Средняя/Высокая	Средняя
Админи-ие	Требуется	Минимальное	Нет	Нет
Стоимость	Бесплатно (инфраструктура)	Зависит от тарифа	Бесплатно для публичных репо	Зависит от использования
Интеграция	Через плагины	Отличная с GitLab	Идеальная с GitHub	Хорошая с GitHub/Bitbucket

Спасибо за внимание!