

CI/CD. Часть 3

Добро пожаловать на урок по CI/CD! Сегодня мы погрузимся в мир непрерывной интеграции и развертывания, изучая работу Jenkins агентов. Узнаем, как они функционируют, как установить их и запустить сборку.

N by Nikita Bezmen

Nikita Bezmen

cd-linux.club





План урока



Описание работы Jenkins агентов

Разберемся, как функционируют агенты и зачем они нужны



Установка

Пошаговая инструкция по установке на разных платформах



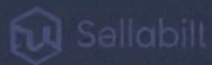
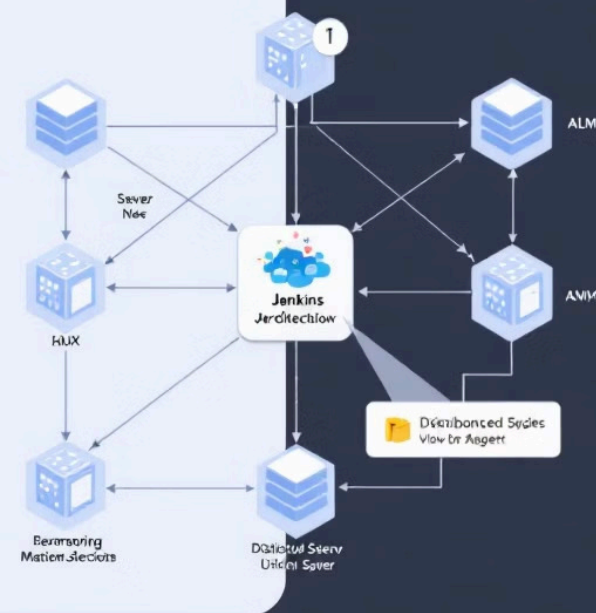
Запуск сборки

Практические примеры запуска сборки через агенты

Jenkins Agent

Architecture Diagrams

Agent



Описание работы Jenkins агентов

Что такое Jenkins?

Популярная система CI/CD для автоматизации разработки ПО

Что такое агенты?

Служебные процессы, выполняющие сборку и развертывание на удаленных компьютерах

Как они работают?

Взаимодействуют с мастер-узлом, который координирует их работу

Ключевые компоненты системы

Мастер-узел

- Центральный сервер Jenkins
- Управляет настройками
- Координирует работу агентов
- Хранит конфигурации

Агенты

- Выполняют задания
- Могут быть на разных ОС
- Позволяют распределить нагрузку
- Масштабируют систему

Распределение нагрузки



Множество агентов

Работают параллельно на разных машинах



Распределение задач

По доступным ресурсам и меткам



Оптимизация ресурсов

Эффективное использование вычислительной мощности



Сокращение времени

Быстрое выполнение CI/CD процессов



Настройка распределения нагрузки

Управление узлами и облаками

Используйте плагин для динамического добавления и удаления агентов.

Настройка агента в Groovy

Создание скрипта для программной конфигурации узлов.

Указание параметров

Имя, количество исполнителей, метки, режим использования.

Пример кода настройки агента

```
// Настройка агента в Groovy скрипте
import hudson.slaves.DumbSlave
import hudson.util.Secret

def jenkins = Jenkins.getInstance()
def agentName = "agent-2"
def numExecutors = 4
def remoteFS = "/var/jenkins/workspace"
def labels = "windows"
def usageMode = Node.Mode.NORMAL

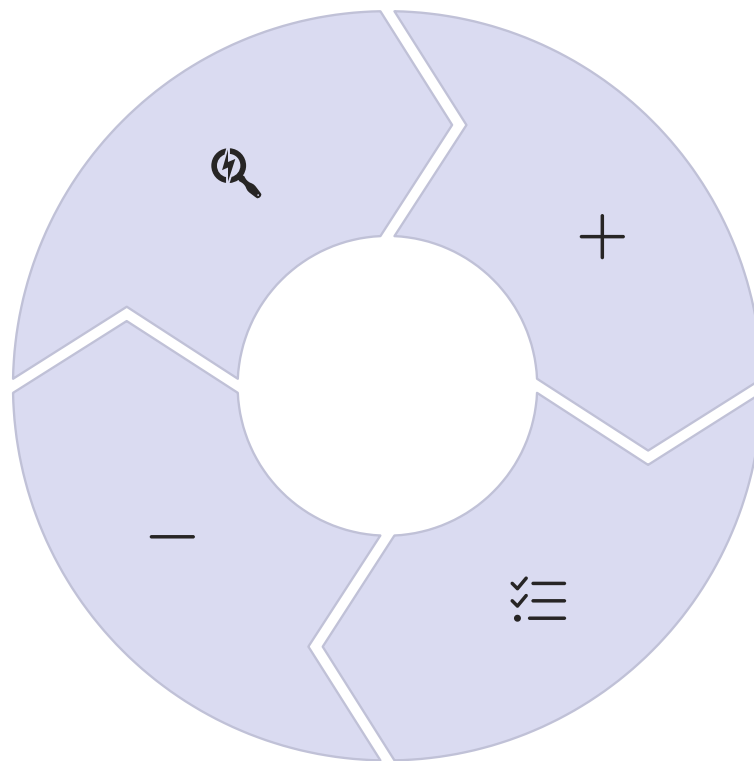
def agent = new DumbSlave(
    agentName,
    "Agent Description",
    remoteFS,
    numExecutors,
    usageMode,
    labels,
    "host",
    "username",
    Secret.fromString("password"),
    "ssh-credentials-id"
)

jenkins.addNode(agent)
```

Автоматическое масштабирование агентов

Мониторинг нагрузки
Отслеживание активных заданий и очередей

Удаление агентов
Освобождение ресурсов при низкой нагрузке



Добавление агентов
Автоматическое создание при высокой нагрузке

Распределение задач
Назначение заданий новым агентам

Автономность агентов



Работа без мастер-узла

Агенты могут продолжать выполнение заданий даже при недоступности мастера



Локальное хранение

Сохранение конфигурации и состояния выполнения задач



Синхронизация

Обновление данных при восстановлении связи с мастером



Настройка автономной работы агентов

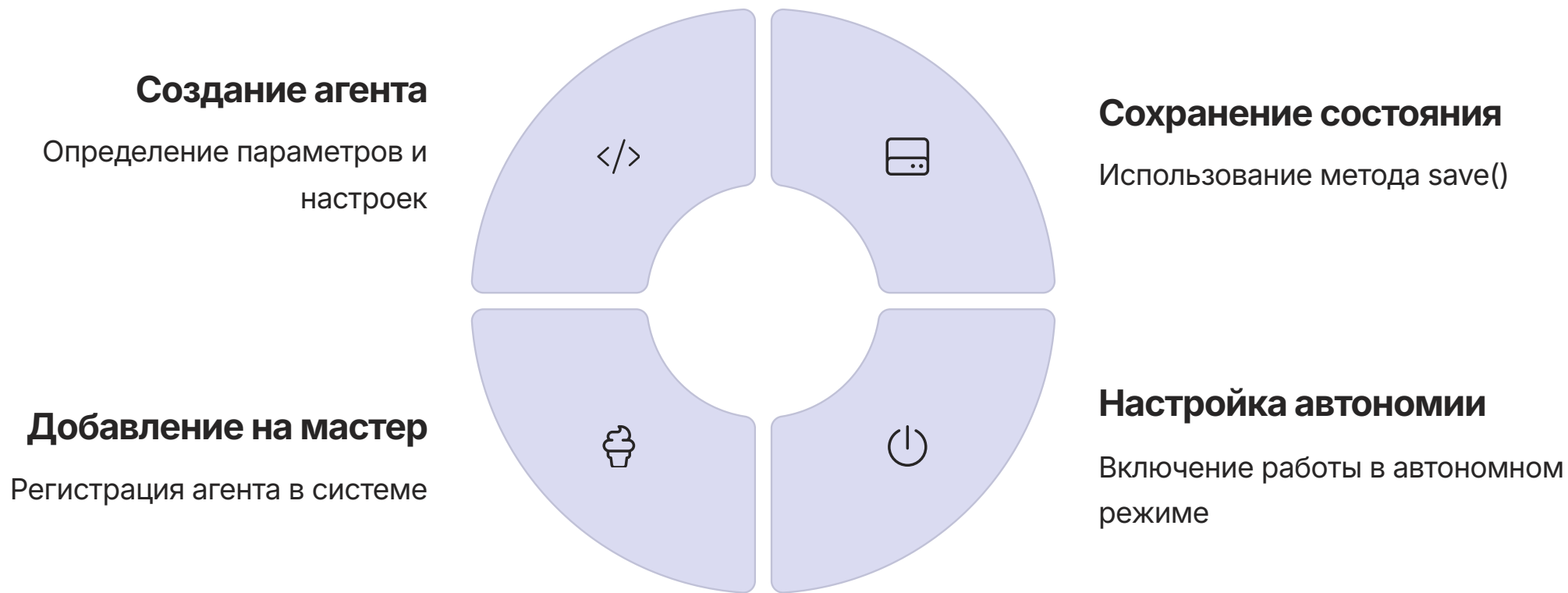
```
// Настройка агента в режиме "Автономия" в Groovy скрипте
import hudson.slaves.DumbSlave
```

```
def jenkins = Jenkins.getInstance()
def agentName = "agent-1"
def numExecutors = 2
def remoteFS = "/var/jenkins/workspace"
def labels = "linux"
def usageMode = Node.Mode.EXCLUSIVE
def offlineAutonomy = true
```

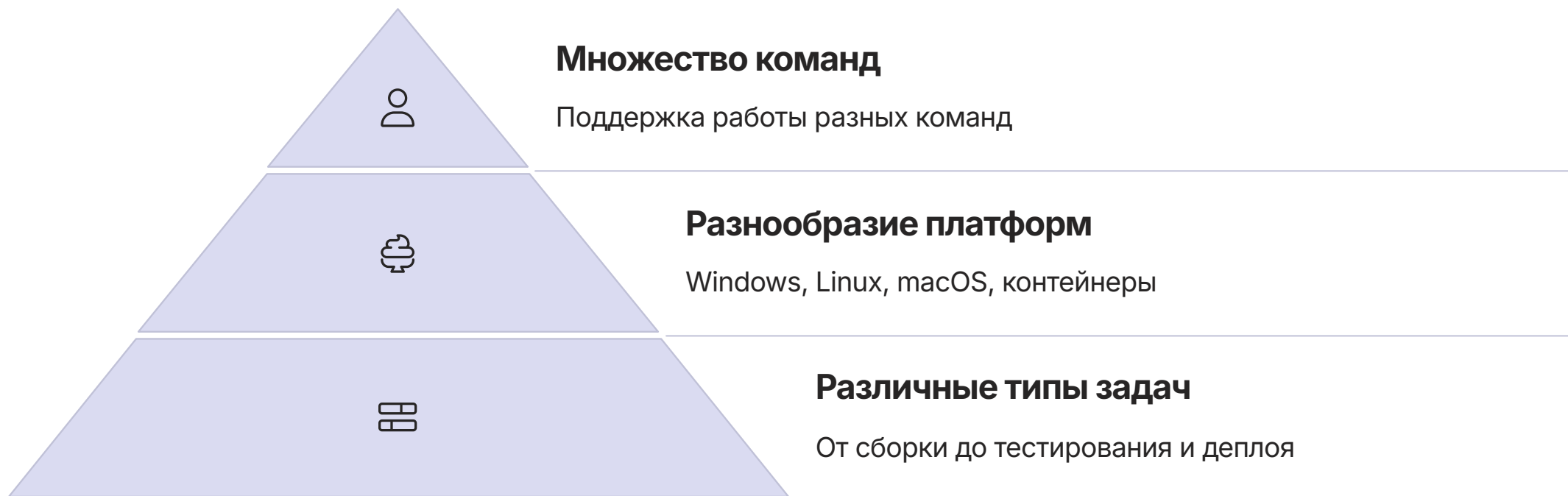
```
def agent = new DumbSlave(
    agentName,
    "Agent Description",
    remoteFS,
    numExecutors,
    usageMode,
    labels,
    "ssh-credentials-id"
)
```

```
agent.setWorksWhenOffline(offlineAutonomy)
jenkins.addNode(agent)
```

Сохранение конфигурации агента



Масштабируемость агентов Jenkins



Настройка агента для Linux



Подготовка скрипта

Создание Groovy-скрипта для настройки



Указание параметров Linux

Задание путей и учетных данных



Назначение метки linux

Для направления соответствующих задач

Настройка агента для Windows

// Пример настройки агента для Windows-сервера в Groovy скрипте

```
import hudson.slaves.DumbSlave
```

```
def jenkins = Jenkins.getInstance()
```

```
def agentName = "windows-agent"
```

```
def numExecutors = 4
```

```
def remoteFS = "C:\\jenkins\\workspace"
```

```
def labels = "windows"
```

```
def usageMode = Node.Mode.NORMAL
```





```
def agent = new DumbSlave(  
    agentName,  
    "Windows Agent Description",  
    remoteFS,  
    numExecutors,  
    usageMode,  
    labels,  
    "host",  
    "username",  
    Secret.fromString("password"),  
    "windows-credentials-id"  
)
```

```
jenkins.addNode(agent)
```





Гибкость агентов Jenkins

-  **Разнообразие задач**
Агенты могут выполнять сборку, тестирование, деплой и другие операции
-  **Разные среды**
Работа с различными языками программирования и фреймворками
-  **Настраиваемые параметры**
Гибкая конфигурация под конкретные нужды проекта
-  **Метки и фильтры**
Точное управление назначением заданий на агенты

Nikita Bezmen cd-linux.club

Настройка агента для сборки и тестирования

Код настройки

```
// Пример настройки агента
// для сборки и тестирования
import hudson.slaves.DumbSlave

def jenkins = Jenkins.getInstance()
def agentName = "build-test-agent"
def numExecutors = 2
def remoteFS = "/var/jenkins/workspace"
def labels = "build test"
def usageMode = Node.Mode.NORMAL

def agent = new DumbSlave(
    agentName,
    "Build and Test Agent Description",
    remoteFS,
    numExecutors,
    usageMode,
    labels,
    "host",
    "username",
    Secret.fromString("password"),
    "ssh-credentials-id"
)

jenkins.addNode(agent)
```

Особенности настройки

- Метки "build test" для идентификации
- Режим NORMAL для разделения ресурсов
- 2 исполнителя для параллельных задач
- Стандартное рабочее пространство

Такой агент автоматически получает задания сборки и тестирования, помеченные соответствующими метками в пайплайне.



Настройка агента для деплоя



Создание агента

Определение параметров для deployment-агента



Назначение метки deployment

Для фильтрации соответствующих задач



Настройка эксклюзивного режима

Выполнение только задач деплоя для безопасности



Отдельное рабочее пространство

Изоляция файлов развертывания от сборки

Безопасность агентов Jenkins

Аутентификация

- Защищенные учетные данные
- SSH-ключи
- Токены доступа

Авторизация

- Разграничение прав
- Ролевой доступ
- Правила выполнения

Шифрование

- Защищенные соединения
- Шифрование хранимых данных
- Секретные переменные

Настройка агента с аутентификацией



Создание защищенного агента

Определение базовых параметров агента



Настройка учетных данных

Использование безопасного хранилища учетных данных



Применение настроек безопасности

Наследование реалма безопасности от мастера



Настройка авторизации

Применение стратегии авторизации от мастера



Настройка агента с шифрованием данных

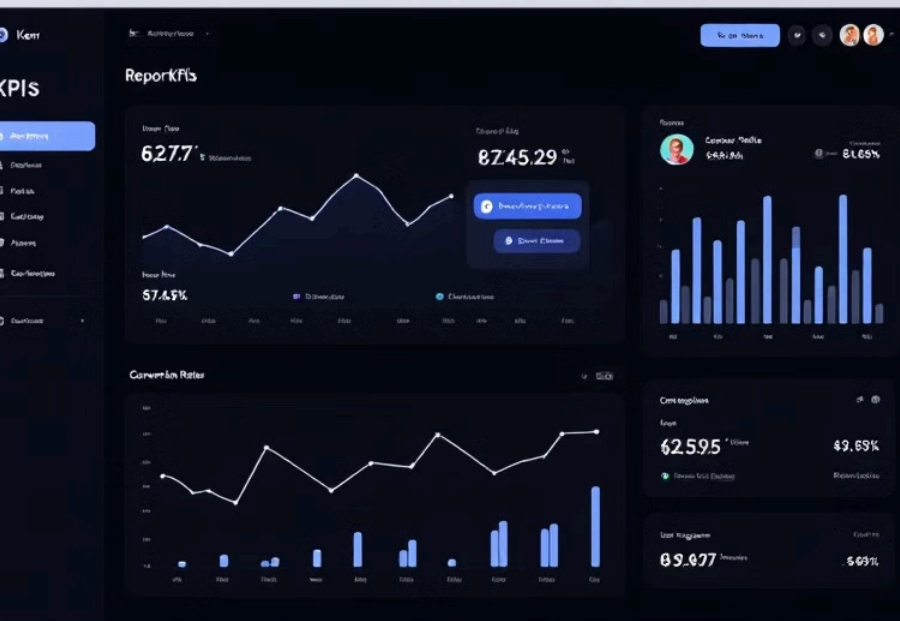
```
// Пример настройки агента с шифрованием данных в Groovy скрипте
import hudson.slaves.DumbSlave
```

```
def jenkins = Jenkins.getInstance()
def agentName = "encrypted-agent"
def numExecutors = 1
def remoteFS = "/var/jenkins/encrypted-agent"
def labels = "encrypted"
def usageMode = Node.Mode.EXCLUSIVE
```

```
def agent = new DumbSlave(
    agentName,
    "Encrypted Agent Description",
    remoteFS,
    numExecutors,
    usageMode,
    labels,
    "host",
    "username",
    Secret.fromString("password"),
    "ssh-credentials-id"
)
```

```
agent.setSecretKey(jenkins.getSecretKey())
jenkins.addNode(agent)
```

Nikita Bezmen cd-linux.club



Репортинг в агентах Jenkins

100%

Покрытие отчетами

Все аспекты работы агентов
документируются

24/7

Мониторинг

Непрерывный контроль
состояния агентов

5+

Типы отчетов

От статуса задач до
тестирования и покрытия кода

Nikita Bezmen cd-linux.club

Отчеты о выполнении в консоли

Базовый отчет в консоли

```
// Пример генерации отчета  
// о статусе выполнения задачи  
// в консоли на агенте Jenkins
```

```
node('agent-label') {  
  stage('Build') {  
    // Ваш код сборки  
    // и развертывания  
  }  
}
```

Преимущества консольных отчетов

- Мгновенная обратная связь
- Подробные логи выполнения
- Диагностика проблем в реальном времени
- Отслеживание каждого шага сборки
- Интеграция с системой уведомлений

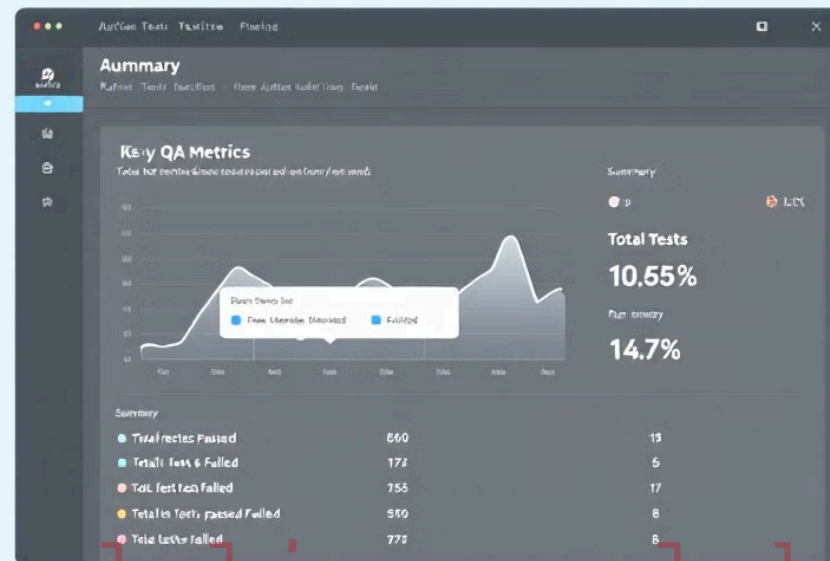
Отчеты о тестировании с JUnit

// Пример генерации отчета о тестировании
// с использованием плагина JUnit на агенте Jenkins

```
node('agent-label') {  
    stage('Build') {  
        // Ваш код сборки и развертывания  
    }  
}
```

```
stage('Test') {  
    // Ваш код тестирования  
}
```

```
post {  
    always {  
        junit 'target/surefire-reports/*.xml'  
    }  
}
```



Плагин Cobertura позволяет генерировать подробные отчеты о покрытии кода тестами. Метрики включают покрытие строк, ветвлений, классов и методов, что помогает выявить протестированные участки кода.



Гибридные окружения для агентов



Облачные ресурсы

AWS, Azure, Google Cloud для динамического масштабирования



Контейнеры

Docker, Kubernetes для изолированных сред сборки



Физические серверы

Выделенное оборудование для требовательных сборок



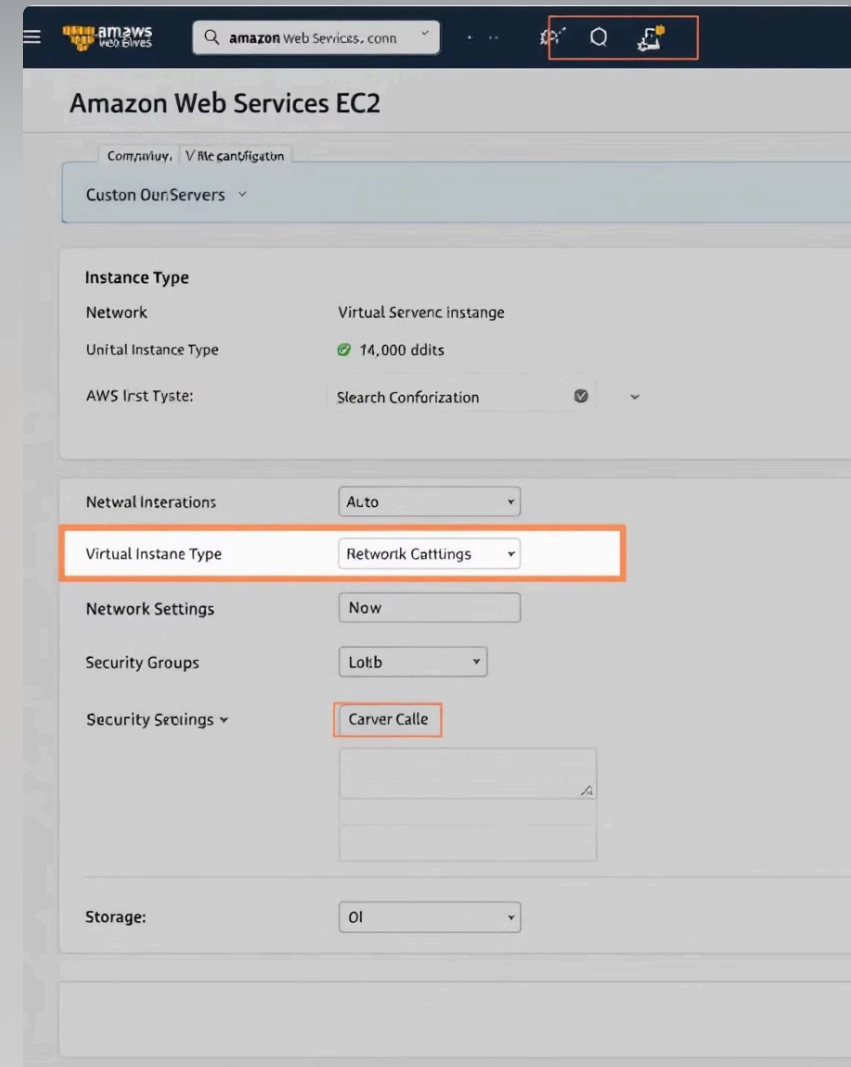
Виртуальные машины

VMware, VirtualBox для эмуляции разных окружений

Настройка агента в AWS EC2

```
// Пример настройки агента Jenkins для работы с облачными ресурсами Amazon EC2
def label = 'my-ec2-agent' // метка агента
def amild = 'ami-0c55b159cb4ffe1c3' // ID образа Amazon Machine Image
def instanceType = 't2.small' // тип инстанса EC2
def sshKey = 'my-ec2-ssh-key' // имя SSH-ключа
def region = 'us-west-2' // регион EC2

// Создание экземпляра агента EC2
ec2NodeTemplate(
    label: label,
    amild: amild,
    instanceType: instanceType,
    sshKey: sshKey,
    region: region
){
    // Конфигурация агента
    // Ваш код настройки агента, например,
    // установка нужных пакетов, настройка окружения и т.д.
}
```



Настройка агента в Docker

Код настройки

```
// Пример настройки агента Jenkins
// для работы с контейнерами Docker
def label = 'my-docker-agent'
def dockerImage = 'jenkins/jnlp-slave'

// Создание контейнера Docker с агентом Jenkins
dockerNode(
    label: label,
    dockerImage: dockerImage
){
    // Конфигурация агента
    // Ваш код настройки агента, например,
    // установка нужных пакетов,
    // настройка окружения и т.д.
}
```

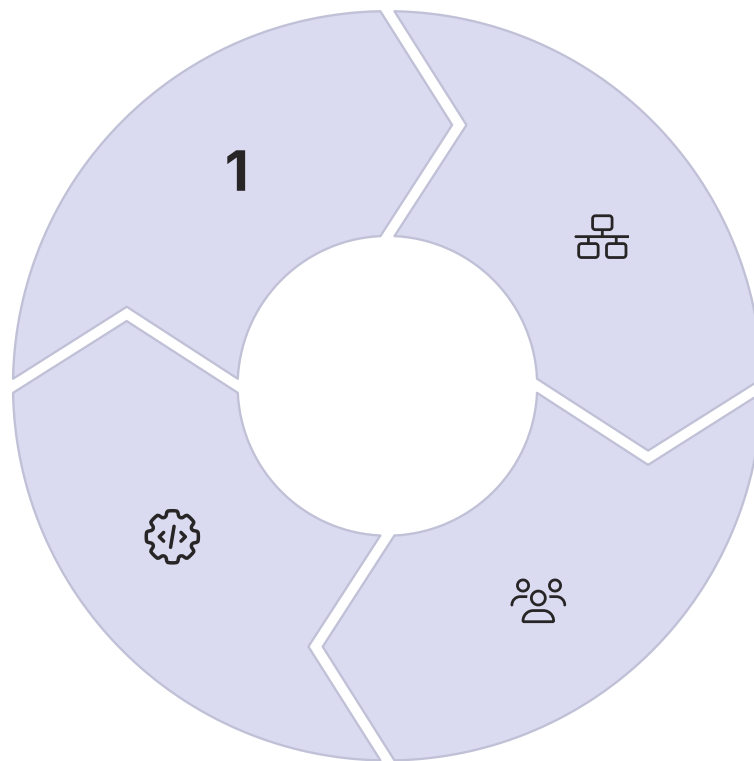
Преимущества Docker агентов

- Быстрое создание и удаление
- Изоляция окружения сборки
- Воспроизводимость процесса
- Простая масштабируемость
- Поддержка различных технологий
- Экономия ресурсов

Настройка агента на физическом сервере

Подготовка сервера
Установка ОС и необходимого ПО

Конфигурация агента
Установка и настройка Jenkins
агента



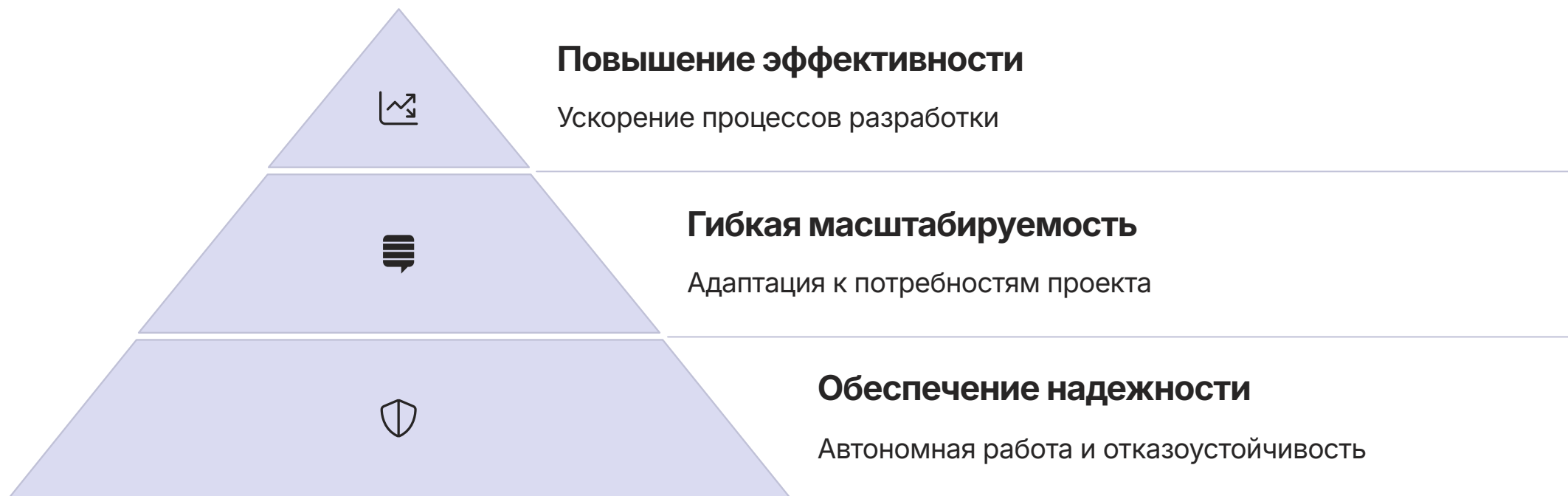
Настройка сети

Обеспечение доступа к мастер-узлу

Создание пользователя

С правами для запуска агента

Значение агентов в инфраструктуре CI/CD



Установка



Загрузка

Получение установочных файлов с официального сайта



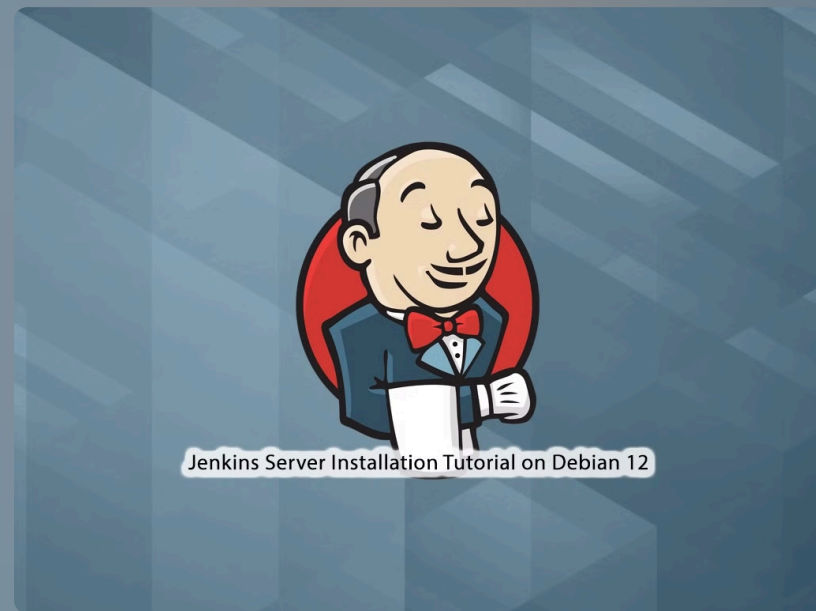
Настройка

Конфигурирование базовых параметров



Запуск

Активация агента и подключение к мастеру



Подготовка к установке



Проверка требований

Соответствие оборудования и ПО



Права администратора

Необходимы для корректной установки



Установка Java

Jenkins агенты требуют Java для работы

Установка на Linux

1 Обновление системы

Выполните обновление пакетов командой `sudo apt update`

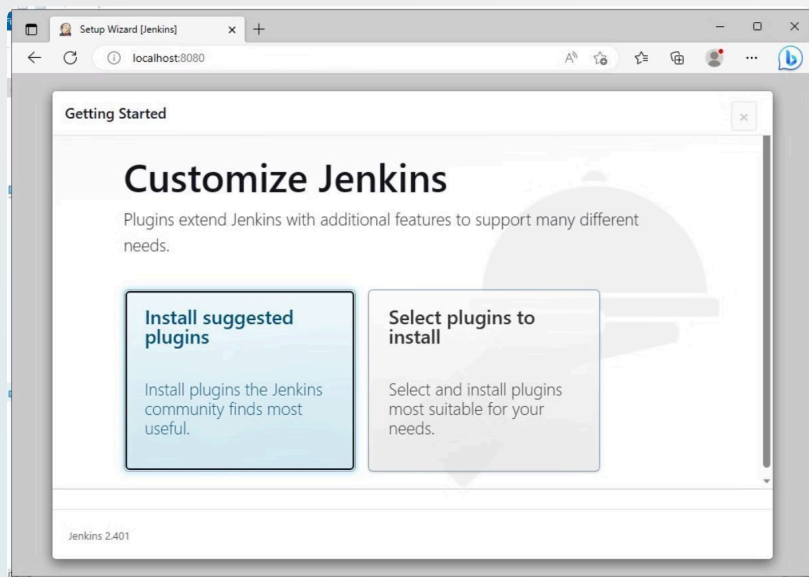
2 Установка Java

Установите JDK командой `sudo apt install default-jdk`

3 Загрузка и запуск Jenkins

Скачайте WAR-файл и запустите агент командой `java -jar jenkins.war`

Установка на Windows



Скачивание установщика

Загрузите MSI файл с официального сайта Jenkins



Запуск мастера установки

Дважды кликните по скачанному файлу



Выбор компонентов

Отметьте нужные опции в установщике



Запуск службы

Активация Windows-службы Jenkins

Установка в Docker

Установка Docker

На Ubuntu:

```
sudo apt update  
sudo apt install docker.io
```

Запуск контейнера

Команда для запуска:

```
docker run -d \  
  --name jenkins-agent \  
  -p 8080:8080 \  
  -p 50000:50000 \  
  jenkins/jenkins
```

Настройка взаимодействия

Мастер-сервер

Центральный узел управления

1



URL подключения

Адрес для связи агента с мастером

Параметры агента

Количество исполнителей, метки и режим



Ключ агента

Уникальный идентификатор для аутентификации

Глобальная настройка агента

Управление узлами

Перейдите в раздел "Управление Jenkins" → "Управление узлами"

Создание нового агента

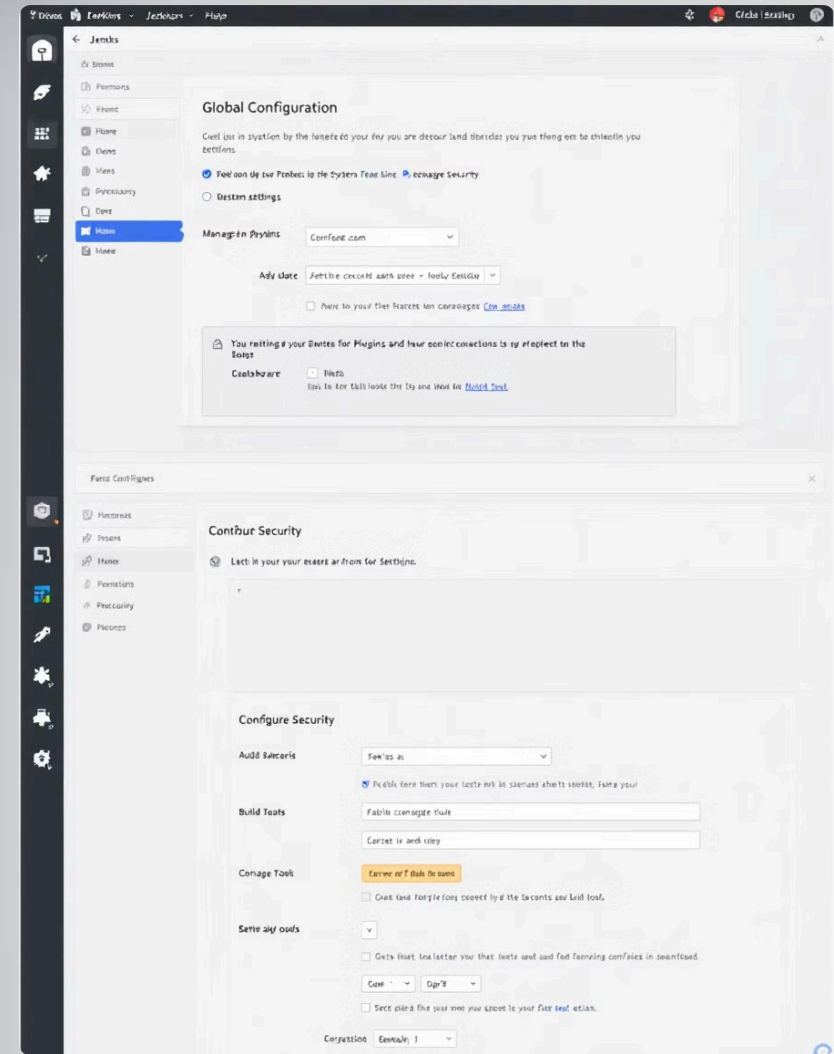
Нажмите "Новый узел" и заполните основные параметры

Настройка соединения

Укажите способ подключения и учетные данные

Определение окружения

Настройте рабочую директорию и инструменты



Пример интерфейса настройки агента

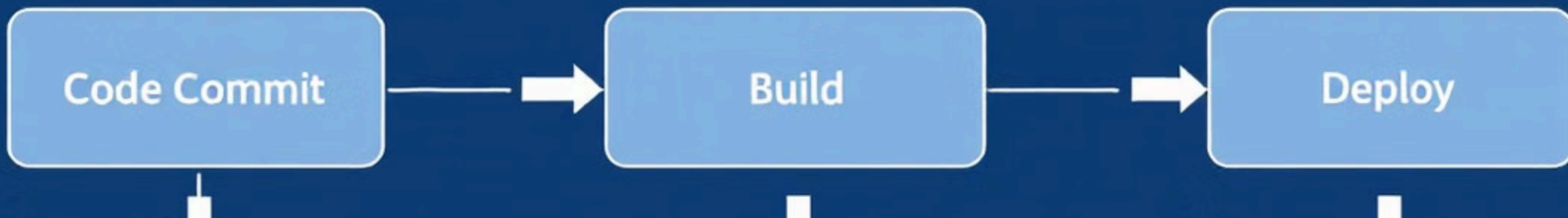
- 1** **Имя агента**
Уникальный идентификатор в системе
- 2** **Описание**
Дополнительная информация о назначении агента
- 3** **Количество исполнителей**
Сколько задач может выполняться параллельно
- 4** **Корневая директория**
Путь к рабочему пространству агента

```
// Задание глобальных настроек агента
jenkins {
  agent {
    // Указание URL сервера Jenkins, на котором будет работать агент
    label 'my-agent'
    // Установка ключа агента для аутентификации
    remote {
      url 'http://<агент>:<порт>/jenkins'
      credentials 'jenkins-agent-credentials'
    }
    // Другие настройки агента (если необходимо)
    // ...
  }
}

// Создание учетных данных для аутентификации агента
credentials {
  // Указание учетных данных агента (например, логин/пароль)
  usernamePassword {
    // Имя учетных данных
    id 'jenkins-agent-credentials'
    // Логин и пароль агента
    username '<логин-агента>'
    password '<пароль-агента>'
    // Описание учетных данных (необязательно)
    description 'Credentials for Jenkins agent'
  }
}
```

Системные настройки агента

```
// Задание системных свойств агента
jenkins {
  agent {
    // Указание URL сервера Jenkins, на котором будет работать агент
    label 'my-agent'
    // Установка ключа агента для аутентификации с использованием системных свойств
    properties([
      // Указание системных свойств для агента
      [
        $class: 'hudson.plugins.sshslaves.SSHLauncher$AuthorizedKeyProperty',
        authorizedKeys: [
          [
            $class: 'hudson.plugins.sshslaves.TrustedKey',
            key: '<ключ-агента>'
          ]
        ]
      ]
    ])
    // Другие настройки агента (если необходимо)
    // ...
  }
}
```



Запуск сборки



Ручной запуск

Инициирование процесса через интерфейс Jenkins



Автоматический запуск

По расписанию или при изменении в репозитории



Через API

Программный запуск с помощью REST API

Что такое запуск сборки



Процесс сборки

Запуск процессов компиляции, тестирования, упаковки и деплоя



Источник кода

Скачивание исходников из репозитория



Проверка качества

Выполнение автоматизированных тестов и инспекций



Подготовка к выпуску

Создание артефактов для последующего деплоя

Запуск сборки через pipeline

```
// Определение pipeline
pipeline {
  agent {
    // Указание метки агента, на котором будет выполняться сборка
    label 'my-agent'
  }
  stages {
    stage('Build') {
      steps {
        // Шаг сборки - можно использовать различные команды сборки
        // например, Maven, Gradle, или другие
        sh 'mvn clean install' // Пример команды сборки с использованием Maven
      }
    }
  }
}
```

Запуск сборки с плагинами

Пример использования плагина Maven

```
// Определение pipeline
pipeline {
  agent {
    // Указание метки агента
    label 'my-agent'
  }
  stages {
    stage('Build') {
      steps {
        // Использование плагина для запуска сборки
        // В данном примере используется плагин "Pipeline: Maven"
        steps([
          $class: 'hudson.tasks.Maven',
          mavenName: 'Maven',
          goals: 'clean install'
        ])
      }
    }
  }
}
```

Преимущества плагинов

- Расширенный функционал
- Интеграция с инструментами
- Поддержка различных технологий
- Больше возможностей настройки
- Профессиональные отчеты
- Повышенная надежность

Проверь свои знания

Ответьте на следующие вопросы, чтобы закрепить изученный материал по агентам Jenkins:

- 1 Что такое агенты Jenkins и какова их основная функция?
- 2 Какие преимущества даёт распределение нагрузки на агентах?
- 3 Что означает автономность агентов Jenkins?
- 4 Какими способами можно обеспечить безопасность агентов?
- 5 Какие существуют способы установки агента на Linux?
- 6 Чем отличается установка на Windows от установки на Linux?
- 7 Какие основные параметры нужно указать при настройке агента?
- 8 Что такое гибридные окружения для агентов?
- 9 Как запустить сборку на выбранном агенте через pipeline?
- 10 Какие преимущества дает использование плагинов при сборке?