CI/CD на основе GitLab

Создание и настройка проектов. Тюнинг GitLab Runner

otus.ru

Проверить, идет ли запись

Меня хорошо видно & слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Создание и настройка проектов. Тюнинг GitLab Runner

Эрик Арайс

преподаватель курса администрирования Linux и DevOps
7 лет опыта работы системным администратором Linux
DevOps-инженер в компании “Сигма”

telegram: @arais_erick

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в учебной группе
#канал группы

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

Карта курса

СТАРТ

CI/CD - системы, подходы и workflow

Gitlab CI

Безопасность

Проектная работа

ФИНИШ

Маршрут вебинара

- Вводная часть
- Группы, подгруппы и проекты в Gitlab
- Создание и настройка проектов в Gitlab
- Настройка и тюнинг GitLab Runner
- Рефлексия

Цели вебинара

К концу занятия вы сможете

1. Создать проект, группу и подгруппу в Gitlab

2. Ориентироваться в основной части интерфейса Gitlab

3. Выполнить базовую настройку Gitlab Runner

4. Запустить простой пайплайн CI/CD

Смысл

Это нужно уметь чтобы:

1. Понимать, как организована командная работа посредством Gitlab

2. Уверенно ориентироваться в самом часто используемом функционале Gitlab

3. Предлагать командам более эффективную организацию совместной работы за счет создания оптимальной структуры групп, подгрупп и проектов

4. Приобрести навык развертывания и настройки Gitlab Runner, который востребован у современного разработчика или devops-инженера

Создание и настройка проектов. Тюнинг GitLab Runner

Краткое напоминание

Для успешного прохождения этого занятия ранее нужно было сделать:

1. Создать и запустить **Managed Service for Gitlab** в облаке Yandex.Cloud
2. Создать и запустить отдельную **Virtual Machine Ubuntu** для Gitlab Runner там же
   (в той же виртуальной подсети и зоне, чтобы они видели друг друга)
   a. На VM развернуть Docker (для работы Gitlab Runner с Docker Executor)
      по этой инструкции, плюс дополнительная настройка
   b. На VM развернуть Gitlab Runner по этой инструкции

Вопросы для проверки

Получилось ли у вас развернуть Gitlab и Gitlab Runner?

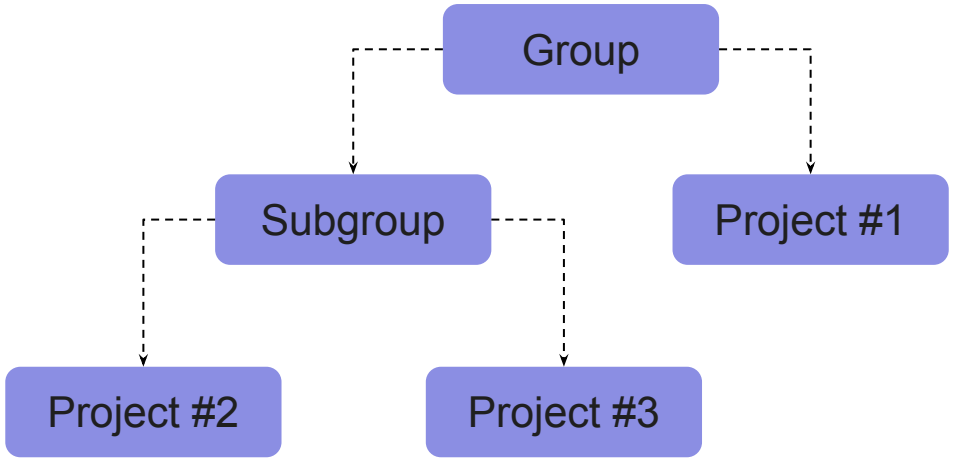
Напишите в чат одну цифру с ответом:

0. - еще не сделал / не было времени
1. - начал делать / в процессе
2. - частично сделал / возникли сложности
3. - да, задеплоил и то и другое, все готово
4. - уже запустил CI/CD пайплайн и он успешно выполнился!

Создание и настройка проектов

Основные сущности проектов в Gitlab

1. Namespace
2. Group
3. Subgroup
4. Project



Пространства имен

Пространства имен (namespaces) используются для организации и группировки проектов. Типы namespaces:

1. **Пользовательские пространства имен (User namespaces):**
   - Пользователь GitLab имеет свое личное пространство имен
   - URL вида: gitlab.com/username/project-name
1. **Групповые пространства имен (Group namespaces):**
   - Для организации проектов, связанных с группой или командой
   - URL вида: gitlab.com/group-name/project-name
1. **Подгрупповые пространства имен (Subgroup namespaces):**
   - Подгруппы внутри групп для дополнительной иерархической организации (необязательный элемент)
   - URLвида: gitlab.com/parent-group/subgroup/project-name

Группы

Инструмент для организации проектов и управления доступом. Основные функции:

1. **Упорядочивание**: организация проектов в иерархию
2. **Управление доступом**: права доступа для всех проектов в группе и назначение ролей пользователям на уровне группы
3. **Совместное использование ресурсов**: Общие CI/CD переменные, общие Docker-реестры и пакеты
4. **Совместная работа**: Групповые доски задач (issue boards) для управления задачами по нескольким проектам, групповые вики для общей документации.
5. **Отчетность и аналитика**: Групповые панели мониторинга для отслеживания активности в проектах, статистика и метрики по группе
6. **Безопасность**: Централизованное управление настройками безопасности
7. **Администрирование**: Единое управление большим числом проектов через группы
8. **Интеграции**: Настройка интеграций на уровне группы, которые автоматически применяются ко всем проектам

Подгруппы

Инструмент для организации гибкой информационной структуры.
Функции подгрупп:

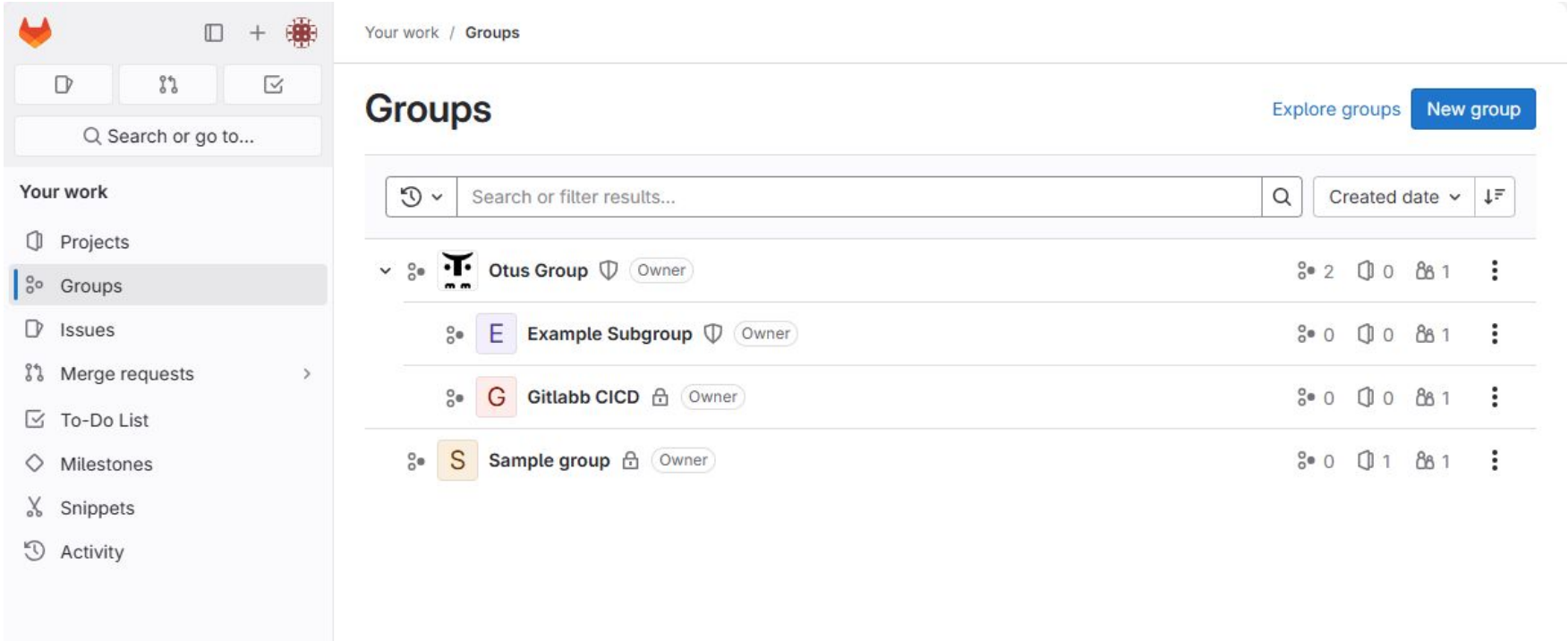
1. Дополнительная иерархическая организация проектов
2. Детальное разделение доступа на уровне подгруппы
3. Улучшенная навигация и поиск по проектам
4. Изоляция проектов
5. Наследование настроек от родительских групп
6. Гибкость в управлении ресурсами, назначение квот и лимитов для подгрупп
7. Отдельные рабочие пространства для разных команд или проектов
8. Улучшенная аналитика на уровне подгруппы
9. Упрощение миграции проектов между группами

Проекты

Проекты -- это основные структурные элементы Gitlab, включающие комплексную среду для управления жизненным циклом разработки, организации кода и совместной работы:

1. Хранение кода в репозитории Git
2. Управление задачами (Issues)
3. Слияние кода (Merge Requests)
4. CI/CD пайплайны
5. Вики для документации проекта
6. Snippets для совместного использования фрагментов кода
7. Безопасность
8. Мониторинг и аналитика
9. Управление доступом
10. Интеграции с внешними инструментами
11. Артефакты и пакеты
12. Статичные сайты (Gitlab Pages)
13. Планирование спринтов
14. Контейнеризация Docker
15. Совместная работа

Пример создания групп и подгрупп



Уровни доступа в GitLab

- **Instance administrator** — доступен только для отдельных инсталляций, может всё.
- **Owner** — владелец группы проектов, может всё кроме чисто технических штук типа включения-отключения фич и интеграции с другими сервисами.
- **Maintainer** — может всё кроме некоторых действий в отношении всего проекта типа изменения его названия или степени видимости, а также деструктивных действий типа удаления задач
- **Developer** — может то же что и Maintainer кроме некоторых администраторских и деструктивных функций внутри проекта типа настройки защищенности ветвей и редактирования комментариев.

Уровни доступа в GitLab

- **Reporter** — может редактировать задачи, но не может вносить изменения в репозиторий.
- **Guest** — доступ только на чтение issue кроме конфиденциальных, может создавать новые задачи.


group
settings
general
Export group
Change group URL
Transfer group
Remove group
access tokens
Add a group access token
integrations
Active integrations
Add an integration
projects
New project
Repository Settings
Deploy tokens
New deploy token
Default branch
Initial default branch name
CI/CD Settings
General pipelines
Variables
Runners
Auto DevOps
applications
Add new application
Package and registry settings
Duplicate packages
Package forwarding
Dependency Proxy
packages and registiries
Package Registry
Dependency Proxy
ci/cd
runners
all
group
project
Register a group runner
group information
Activity
All
Push events
Merge events
Issue events
Comments
Wiki
Team
Labels
New labels
Group members
Invite a group
Invite members
Remove member / Leave group
issues
list
New project
Select project to create issue
board
Create list
Edit board
milesones
New milestone
merge request
open
merged
closed
all
Select project to create merge request

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Практика

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Тюнинг GitLab Runner

Как работает Gitlab Runner

Gitlab Runner предназначен для выполнения заданий, запущенных в пайплайне Gitlab CI/CD. Задания описываются в файле .gitlab-ci.yml

Как работает Gitlab Runner:

1. Runner запрашивает новые задания из экземпляра Gitlab Server
2. Runner компилирует и отправляет задачу в Executor
3. Executor клонирует исходные файлы, загружает (если нужно) требуемые заданию артефакты из Gitlab Registry и выполняет задачу
4. Executor возвращает результат работы задания обратно в Runner
5. Runner обновляет выходные данные и статус задания в Gitlab Server

Типы Runners

1. **Shared runner (Instance)**
   Где найти: Admin Area > CI/CD > Runners
2. **Group runner**
   Где найти: CI/CD > Runners
3. **Project runner (Specific)**
   Где найти: Settings > CI/CD

Типы Gitlab Runner Executors

- SSH
- Shell
- Parallels
- VirtualBox
- Docker
- Docker Autoscaler
- Docker Machine (auto-scaling)
- Kubernetes
- Instance
- Custom

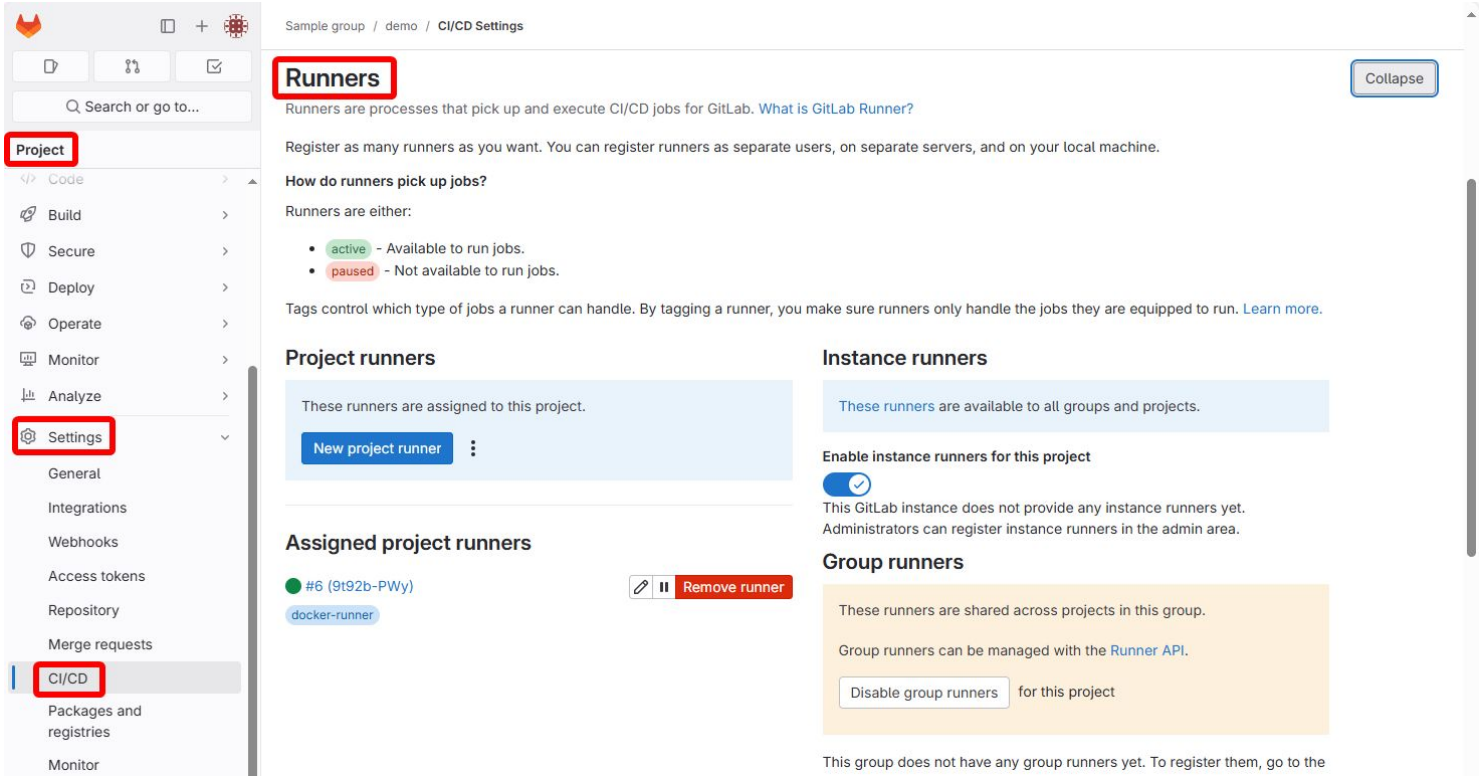
Docker Executor -- рекомендуемый вариант для большинства случаев (или если не знаете, что выбрать)

Вопросы для проверки

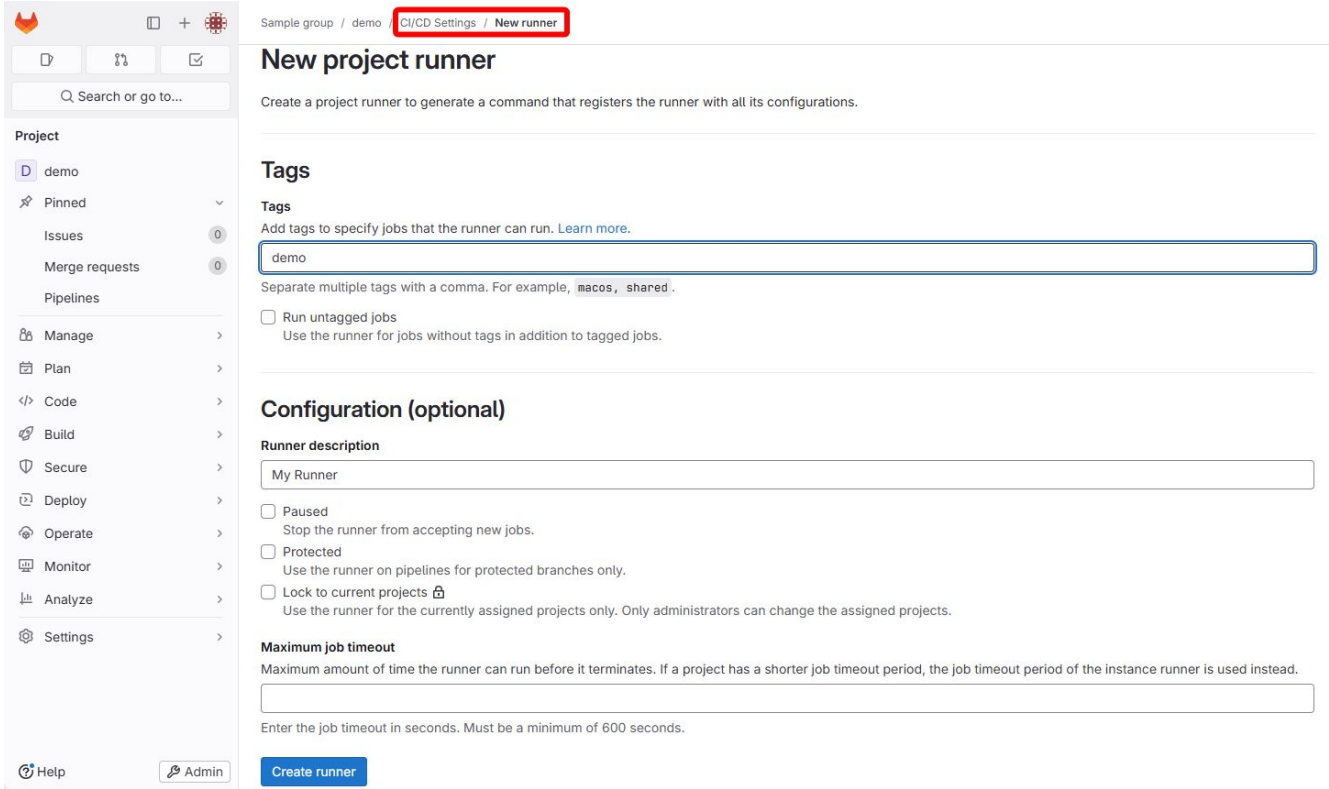
Понимаете ли вы, как работает Gitlab Runner?
А чем отличается Runner от Executor?
Напишите в чат одну цифру с ответом:

0. - не уверен или не понимаю до конца
1. - в целом понятно, но остались неясные моменты
2. - да, все понятно
3. - вчера деплоил раннеры в кубер, элементарно

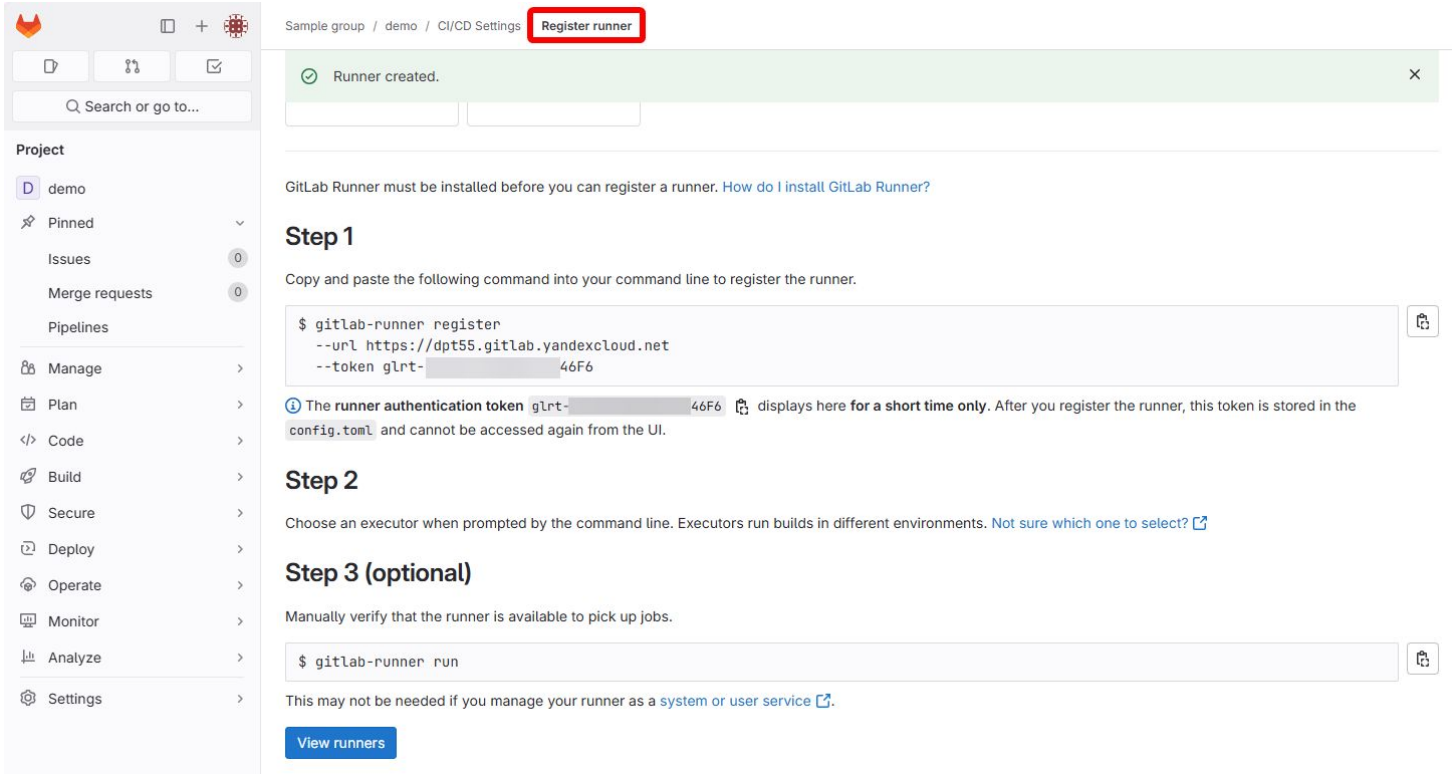
1/3 Создаем Project Runner



2/3 Создаем Project Runner



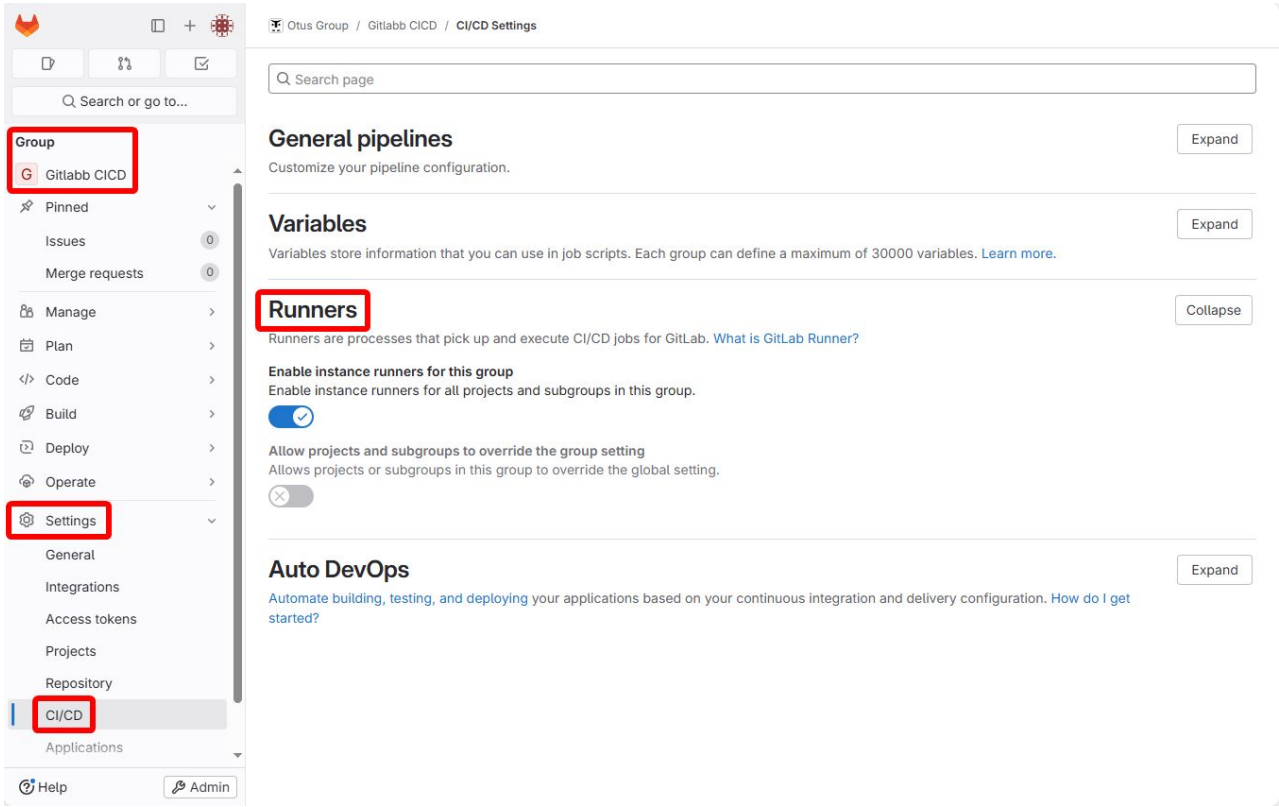
3/3 Создаем Project Runner



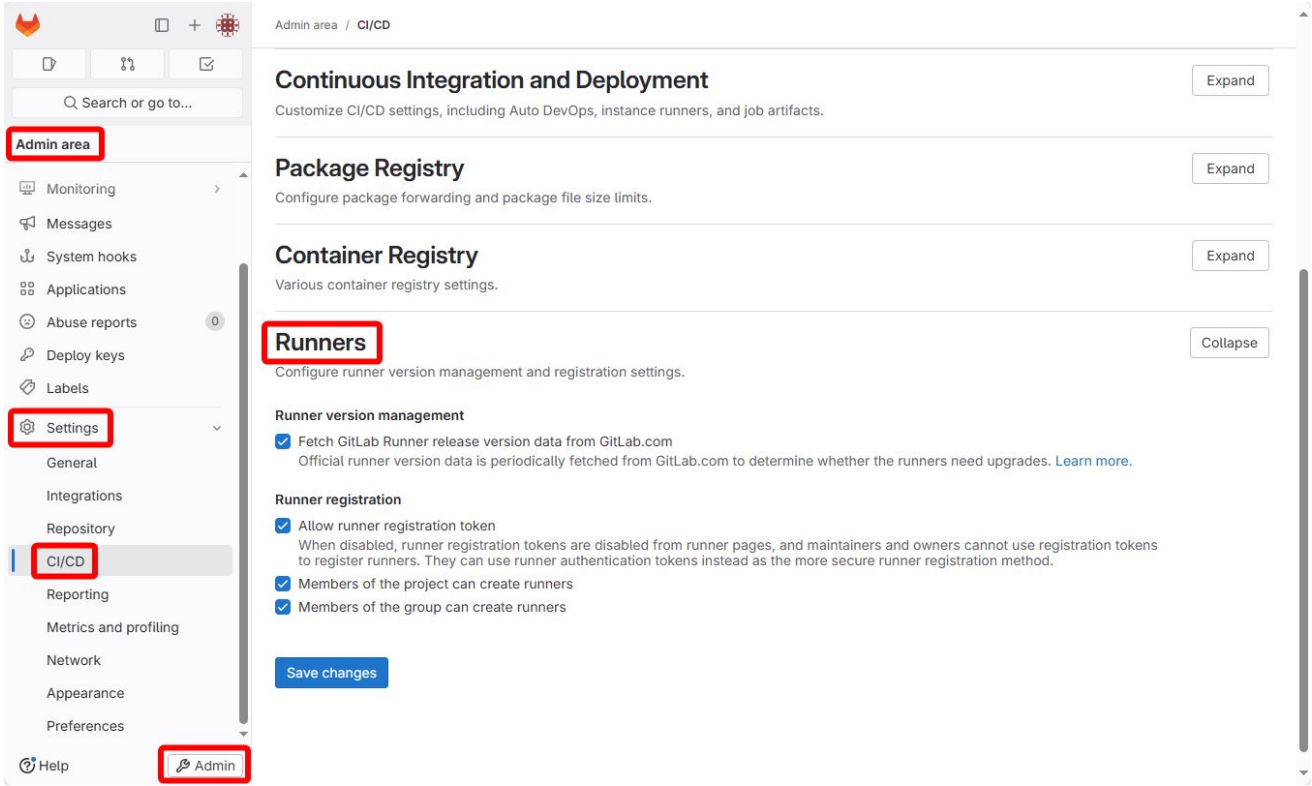
Регистрируем Runner

1. Токен показывается **один раз**. Сохраните команду с токеном, если необходимо
2. Откройте консоль VM с Ubuntu, где вы ранее установили Runner, и отправьте предложенную команду регистрации с URL вашего инстанса Gitlab и токеном
3. На вопрос установщика о выборе Executor введите “`docker`”, затем укажите любой подходящий Docker образ (можно предлагаемый по умолчанию)
   NB: Не забудьте предварительно установить Docker на VM.
4. Проверьте, что Runner появился: `gitlab-runner list`
   (добавьте `sudo` перед командой, если Runner был установлен с админ правами)
5. Перезапустите демон, если Runner был установлен как служба:
   `sudo systemctl restart gitlab-runner`
6. Вернитесь в Gitlab и проверьте статус Runner

Настройка Group runner



Настройка Shared Runner



Создаем простой пайплайн

1. Создайте файл .gitlab-ci.yml в корневой папке репозитория вашего проекта Gitlab
2. Добавьте в него следующее содержимое:

```
echo job:
  script:
   - echo "Hello world!"
```

1. Сохраните и отправьте коммит с изменениями

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Практика

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Подведём итоги занятия

Теперь вы сможете:

Отправьте в чат номера достигнутых целей

1. Создать проект, группу и подгруппу в Gitlab

2. Ориентироваться в основной части интерфейса Gitlab

3. Выполнить базовую настройку Gitlab Runner

4. Запустить простой пайплайн CI/CD

Домашнее задание

К этому уроку на платформе есть ДЗ, в котором вам нужно:

1. Создать и настроить проект
2. Привязать к нему GitLab Runner
3. Запустить простой пайплайн из одной задачи
4. Проверить успешность выполнения задачи в пайплайне

Полезные ссылки

1. Установка Gitlab Runner
2. Зарезервированные слова, которые нельзя использовать при именовании проектов и групп
3. Что не забыть сделать после установки Docker на Ubuntu (для запуска Gitlab Runner с Docker Executor)

Заполните, пожалуйста, опрос о занятии

по ссылке в чате

СКЛАДЧИК

ЭТОТ КОНТЕНТ КУПЛЕН НА САЙТЕ SKLADCHIK.ORG

SKL.TO

ТЫ ЕЩЕ НЕ В КЛУБЕ?
 присоединяйся и начинай экономить!

ЧТО ТАКОЕ КЛУБ «СКЛАДЧИК»?

Платформа
Это платформа, где каждый день тысячи людей собираются вместе, чтобы общими усилиями находить, приобретать и изучать курсы интересные именно им.

Сообщество
Это сообщество из 500 000 людей, открывших для себя выгодный способ быть в тренде самых актуальных и получать ценные знания по минимальной цене.

Библиотека
Это крупнейшая библиотека инфопродуктов в которой можно найти практически любой курс или тренинг продававшийся за последние 10 лет, а также уникальные авторские инфопродукты, которые не получиться найти больше нигде.