Разница между **Ansible** и **GitLab** заключается в их назначении. Они решают разные задачи в рамках DevOps, но могут и часто работают вместе.

**Ansible: Инструмент управления конфигурациями**

* **Назначение:** Ansible — это **инструмент для автоматизации, управления конфигурациями и развёртывания приложений**. Его основная цель — выполнять команды на удалённых серверах.
* **Что он делает:**
  + **Управление конфигурацией (Configuration Management):** Установка программного обеспечения, настройка служб, управление файлами на множестве серверов одновременно.
  + **Развёртывание приложений (Application Deployment):** Копирование кода, запуск скриптов сборки, перезапуск сервисов на серверах.
  + **Автоматизация (Orchestration):** Выполнение последовательных, сложных задач, которые требуют координации между разными серверами (например, сначала обновить базу данных, потом обновить приложение).
* **Ключевые особенности:**
  + **Декларативный подход:** Вы описываете желаемое состояние системы ("установить Nginx и убедиться, что он работает"), а Ansible сам находит шаги, чтобы достичь этого состояния.
  + **Без агентов (Agentless):** Работает через SSH, не требует установки специального программного обеспечения на управляемых серверах.
  + Использует YAML-файлы (**Playbooks**) для описания задач.

**Пример использования Ansible:** Вы хотите развернуть своё веб-приложение на 10 серверах. Вместо того чтобы вручную заходить на каждый сервер, вы пишете Ansible-Playbook, который:

1. Устанавливает веб-сервер Nginx.
2. Клонирует ваш код из репозитория.
3. Настраивает Nginx для работы с вашим приложением.
4. Перезапускает Nginx. Затем вы запускаете этот Playbook один раз, и он выполняет все эти шаги на всех 10 серверах.

**GitLab: Платформа для жизненного цикла разработки (DevOps Platform)**

* **Назначение:** GitLab — это **единая, интегрированная платформа для всего жизненного цикла разработки ПО (DevOps)**. Она объединяет в себе управление репозиторием Git, CI/CD, отслеживание задач и многое другое.
* **Что он делает:**
  + **Управление репозиторием Git:** Хранение исходного кода.
  + **CI/CD (Continuous Integration/Continuous Deployment):** Автоматическая сборка, тестирование и развёртывание кода при каждом коммите.
  + **Управление проектами:** Доски задач, отслеживание ошибок.
  + **Управление безопасностью (DevSecOps):** Сканирование кода на уязвимости.
* **Ключевые особенности:**
  + **Все в одном:** Объединяет множество инструментов в одном интерфейсе.
  + **CI/CD Pipeline:** Использует файл .gitlab-ci.yml для описания этапов сборки и развёртывания.

**Пример использования GitLab:** Вы работаете над своим проектом. Вы делаете коммит и пушите код в репозиторий GitLab.

1. GitLab видит новый коммит и запускает **CI/CD Pipeline**.
2. Pipeline (описанный в .gitlab-ci.yml) запускает тесты и собирает ваше приложение.
3. После успешной сборки, Pipeline может вызвать **Ansible-Playbook** для развёртывания этого нового приложения на ваших серверах.

**Совместная работа**

GitLab и Ansible не конкурируют, а **дополняют друг друга**.

* **GitLab** — это **дирижёр**, который автоматизирует весь процесс разработки. Он решает **когда** и **что** делать.
* **Ansible** — это **исполнитель**, который автоматизирует конкретные задачи на серверах. Он решает **как** это делать.

Типичный сценарий: GitLab CI/CD запускает Pipeline, который, в свою очередь, вызывает Ansible для выполнения задач развёртывания.