Публикация на тему

Gitlab CI/CD, деплой приложений

В статье рассмотрим как осуществляется деплой приложений в репозитории Gitlab с помощью CI/CD

Анотация

CI/CD — это непрерывный метод разработки программного обеспечения, при котором вы непрерывно создаете, тестируете, развертываете и отслеживаете итеративные изменения кода. Этот итеративный процесс помогает снизить вероятность разработки нового кода на основе предыдущих версий с ошибками или сбоями. GitLab CI/CD может выявлять ошибки на ранних этапах цикла разработки и гарантировать, что код, развернутый в производстве, соответствует вашим установленным стандартам кода.

CI/CD is a continuous method of software development, where you continuously build, test, deploy, and monitor iterative code changes.

This iterative process helps reduce the chance that you develop new code based on buggy or failed previous versions. GitLab CI/CD can catch bugs early in the development cycle, and help ensure that the code deployed to production complies with your established code standards.

Автор

Михалькевич Александр Викторович

Публикация

Наименование Gitlab CI/CD, деплой приложений

Автор А.В.Михалькевич

Специальность В статье рассмотрим как осуществляется деплой приложений в репозитории Gitlab с помощью CI/CD,

Анотация

CI/CD — это непрерывный метод разработки программного обеспечения, при котором вы непрерывно создаете, тестируете, развертываете и отслеживаете итеративные изменения кода. Этот итеративный процесс помогает снизить вероятность разработки нового кода на основе предыдущих версий с ошибками или сбоями. GitLab CI/CD может выявлять ошибки на ранних этапах цикла разработки и гарантировать, что код, развернутый в производстве, соответствует вашим установленным стандартам кода.

Anotation in English

CI/CD is a continuous method of software development, where you continuously build, test,

deploy, and monitor iterative code changes.

This iterative process helps reduce the chance that you develop new code based on buggy or failed previous versions. GitLab CI/CD can catch bugs early in the development cycle, and help ensure that the code deployed to production complies with your established code standards.

Ключевые слова Деплой, развертывание, веб-приложения, CI/CD, deploy, gitlab

Количество символов 6287

Содержание

Введение

Чтобы научиться пользоваться Gitlab CI/CD необходимо разобраться с синтаксисом файла .gitlab-ci.yml и помнить про 4 этапа: 1) создание файла 2) определение заданий для раннера в этом файле 3) создание и запуск раннера на сервере, 4) настройки раннера и использование переменных сервера в заданиях.

1 Создать файл.gitlab-ci.yml

Чтобы использовать GitLab CI/CD, вы начинаете с файла .gitlab-ci.yml в корне вашего проекта.

Этот файл определяет этапы, задания и скрипты, которые будут выполняться во время вашего конвейера CI/CD. Это файл YAML с собственным синтаксисом. В этом файле вы определяете переменные, зависимости между заданиями и указываете, когда и как каждое задание должно выполняться. Вы можете назвать этот файл как угодно, но .gitlab-ci.yml — наиболее распространенное имя, и документация продукта называет его файлом .gitlab-ci.yml или файлом конфигурации CI/CD.

Для получения дополнительной информации см.:

Tutorial: Create your first `.gitlab-ci.yml` file
The CI/CD YAML syntax reference,
Basics of CI blog

2 Создать runner

Runners — это агенты, которые запускают ваши задания. Эти агенты могут работать на физических машинах или виртуальных экземплярах. В файле .gitlab-ci.yml вы можете указать

образ контейнера, который вы хотите использовать при запуске задания. Runner загружает образ, клонирует ваш проект и запускает задание локально или в контейнере.

Для получения дополнительной информации см.:

Create a runner on your local machine
More information about runners

3 Определить pipeline

Pipelines — это то, что вы определяете в файле .gitlab-ci.yml, и то, что происходит, когда runner считывает файл. Pipeline состоит из заданий (jop) и этапов (stage): Этапы определяют порядок выполнения. Типичными этапами могут быть сборка (build), тестирование (test) и развертывание (deploy). Задания определяют задачи, которые должны быть выполнены на каждом этапе. Например, задание может компилировать или тестировать код. Pipelines могут запускаться различными событиями, такими как фиксации или слияния, или могут быть запущены по расписанию.

Для получения дополнительной информации см.:

Pipeline editor,
Visualize your pipeline
Pipelines

4 Использовать CI/CD переменные в заданиях

Переменные GitLab CI/CD — это пары «ключ-значение», которые вы используете для хранения и передачи параметров конфигурации и конфиденциальной информации, например паролей или ключей API, в задания в конвейере.

Используйте переменные CI/CD для настройки заданий, делая значения, определенные в другом месте, доступными для заданий. Вы можете жестко закодировать переменные CI/CD в файле .gitlab-ci.yml, задать их в настройках проекта или сгенерировать их динамически. Вы можете определить их для проекта, группы или экземпляра.

Существует два типа переменных: пользовательские переменные и предопределенные.

Пользовательские переменные определяются пользователем. Создавайте и управляйте ими в пользовательском интерфейсе GitLab, API или в файлах конфигурации. **Предопределенные** переменные автоматически устанавливаются GitLab и предоставляют информацию о текущем задании, конвейере и среде.

Для получения дополнительной информации см.:

CI/CD variables
Dynamically generated predefined variables

Заключение

Список использованных источников

1. [url] **Документация Gitlab** https://docs.gitlab.com

Приложения