



# Устройство Gitlab CI

## Построение процесса непрерывной интеграции

# План

- Gitlab CI
- Концепции Gitlab CI
- CI для обычного приложения
- CI для контейнеризованного приложения

# Continuous Integration

- Непрерывная интеграция – это процесс разработки ПО с частыми автоматизированными сборками и автоматическим тестированием
- Необходимый процесс в поставке ПО
- Интеграция большого количества инструментов

# Gitlab CI

- Развивается с 2011 года
- SaaS и приватные инсталляции
- Community Edition
- Enterprise Edition
  - LDAP / Kerberos
  - Remote repository mirroring
  - Push rules
  - Support
  - HA

# Не только CI

- Репозитории
- Issue-треккер
- Wiki
- Gitlab Pages
- Снимпеты
- Хранение артефактов и Docker registry
- Встроенный мониторинг и метрики

# Под капотом

- Ruby on Rails, Go
- Postgres, Redis
- Sidekiq, Nginx, Prometheus

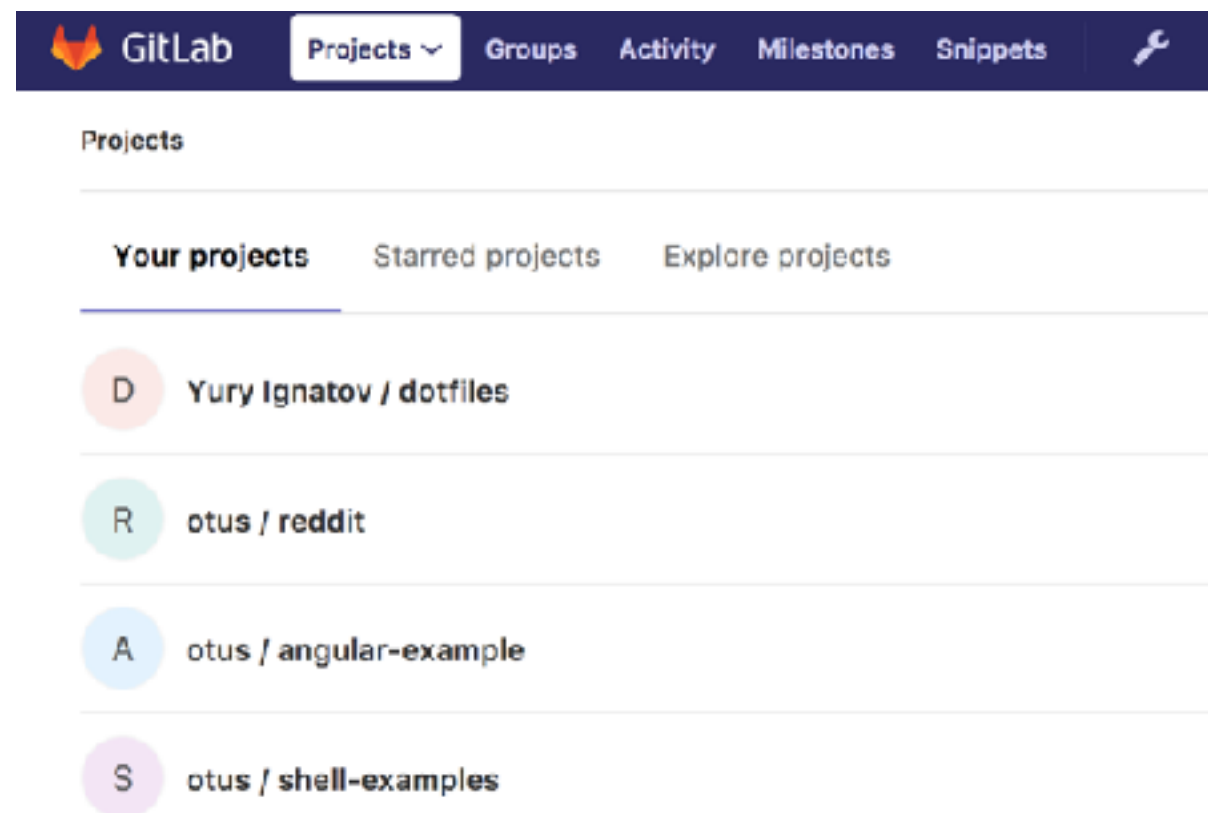
# Концепции Gitlab CI

- group
- project
- pipeline
- job
- scheduler
- runner
- executor
- Это не все и про каждый возможного говорить очень долго :)



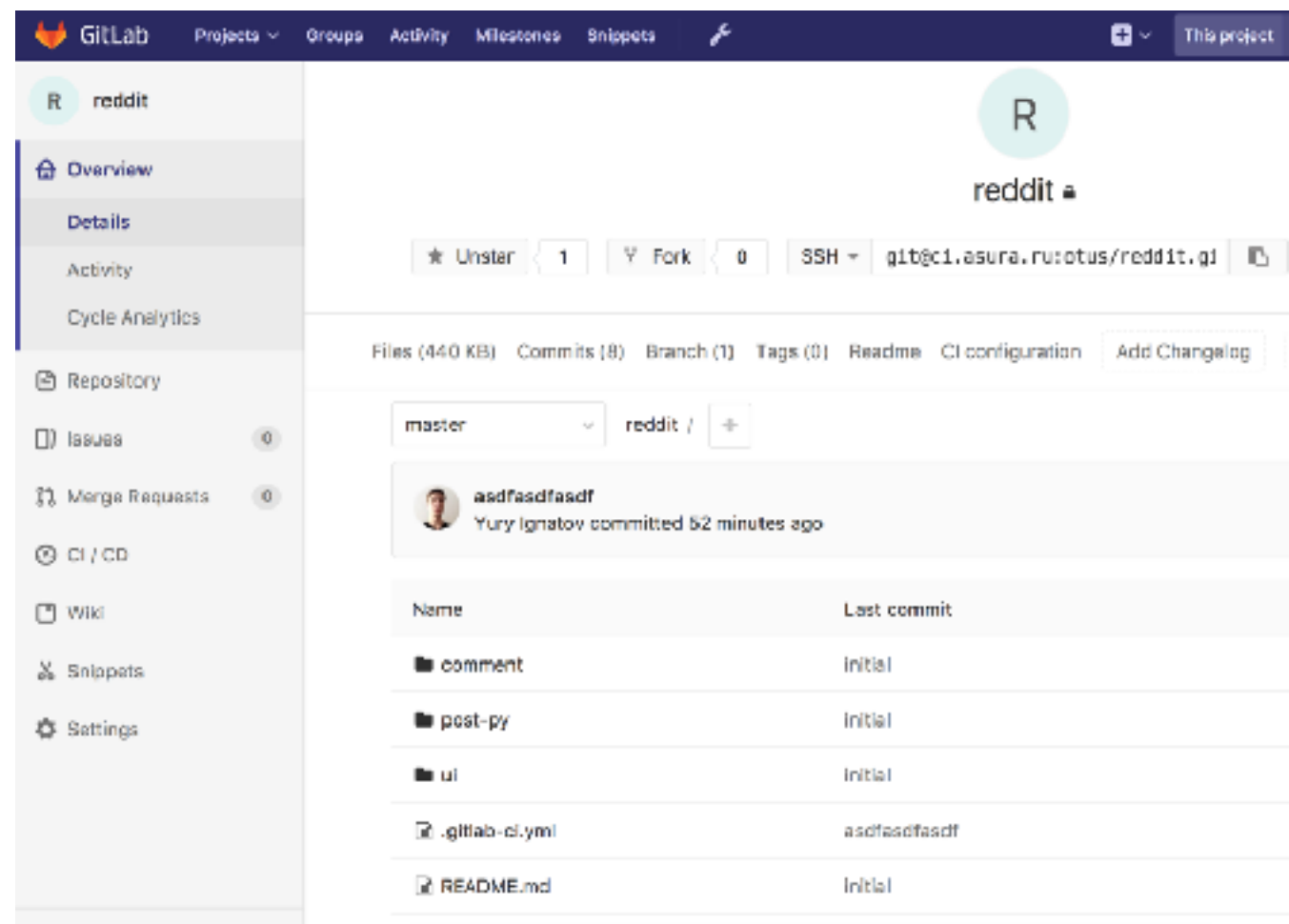
# Группы

- Иерархия проектов
  - Для отделов
  - Для организаций
  - Для продуктов
- Контроль доступа



# Проекты

- Репозиторий с кодом
- Issue-треккер
- Wiki
- CI/CD
- Настройки



# Pipelines

- Пайплайн определяется на проект
- Состоит из задач (jobs)
- Задачи поделены на этапы (stages)
- Запускается по коммиту или по расписанию
- Триггеры

# Jobs

- Минимальная выполняемая сущность
- Запускается на Runners
- Каждый Job выполняется независимо
- vars, only/except, when, allow\_failure, retry
- Могут порождать артефакты

# Runners

- Процесс, выполняющий Jobs
- Shared - обслуживает все проекты / Specific - выделенный для проекта
- Executors
- Теги

The screenshot displays the GitLab web interface for a project named 'reddit'. The left sidebar shows the navigation menu with 'CI / CD' selected. The main content area is divided into two sections: 'Specific Runners' and 'Shared Runners'.

**Specific Runners**

**How to setup a specific Runner for a new project**

1. Install a Runner compatible with GitLab CI (checkout the [GitLab Runner section](#) for information on how to install it).
2. Specify the following URL during the Runner setup: <http://ci.asura.ru/>
3. Use the following registration token during setup: **48E2d4BxtqnyHkVuErEd**
4. Start the Runner!

**Runners activated for this project**

A single runner is shown with ID **326823b4** and a lock icon. A 'Remove Runner' button is next to it. Below it, the 'Best Runner' is listed with tags: **highmem java8 linux ubuntu xenial**.

**Available specific runners**

A runner with ID **2ae0c72e** is listed with the tag **dockerz**. An 'Enable for this project' button is next to it. Below it, the available tags are: **docker linux ubuntu xenial**.

**Shared Runners**

GitLab Shared Runners execute code of different projects on the same Runner unless you configure GitLab Runner Autoscale with MaxBuilds 1 (which it is on GitLab.com).

**Enable shared Runners for this project**

**Available shared Runners : 1**

A single shared runner is shown with ID **48788594** and the tag **default**. Below it, the available tags are: **docker linux ubuntu xenial**.

# Artifacts

- Результат работы билда нужно сохранить и сделать доступным для команды
- Обеспечение жизненного цикла артефакта
- HTTP API для загрузки
- Docker registry

# .gitlab-ci.yml

- Ветки в git могут иметь свой пайплайн
- Этапы
  - Jobs в одном этапе выполняются параллельно
  - Следующий этап запускается, когда все Job прошлого завершились
- Переменные и секретные переменные
- Cache

# Пример проекта

Для простоты используем shell-скрипты имитирующие шаги

The screenshot shows the GitLab interface for a project named 'shell-examples'. The left sidebar contains navigation links: Overview, Details, Activity, Cycle Analytics, Repository, Issues, Merge Requests, CI / CD, Wiki, Snippets, and Settings. The main content area displays the project name, a star/fork button, and the SSH URL: `git@ci.asura.ru:otus/shell-exa`. Below this, there are buttons for 'Add Changelog', 'Add License', and 'Add Contribution guide'. A commit history table is shown with the following data:

Name	Last commit	Last Update
<code>.gitlab-ci.yml</code>	add gitlabci	2 minutes ago
<code>README.md</code>	add README	55 minutes ago
<code>build.sh</code>	add build script	9 minutes ago





# Пример проекта

## Содержание файлов








 **build.sh** 187 Bytes 

```
1  #!/bin/bash
2
3  echo "Starting application build"
4
5  BUILD_DURATION=$(( $RANDOM % 10 ) + 1 )s
6  echo "Build will take $BUILD_DURATION"
7  sleep $BUILD_DURATION
8
9  echo "Finising application build"
```


 **.gitlab-ci.yml** 80 Bytes 


```
1  before_script:
2    - echo "Before script"
3
4  build_job:
5    script:
6      - ./build.sh
7
```

otus > shell-examples > Pipelines

All 1 Pending 0 Running 0 Finished 1 Branches Tags				
Status	Pipeline	Commit	Stages	
 passed	#1 by  latest	P master  3d83daaa  add gitlabci		 00:00:11  4 minutes ago

otus > shell-examples > Jobs > #1

 passed

Job #1 triggered 7 minutes ago by  Yury Ignatov

Retry

```
Running with gitlab-runner 10.0.2 (a9a76a50)
  on default (48788594)
Using Shell executor...
Running on 811179723cfc...
Cloning repository...
Cloning into '/home/gitlab-runner/builds/48788594/0/otus/shell-examples'...
Checking out 3d83daaa as master...
Skipping Git submodules setup
$ echo "Before script"
Before script
$ ./build.sh
Starting application build
Build will take 10s
Finising application build
Job succeeded
```

# Пример проекта

Pipeline состоит из нескольких стадий

Jobs в одной стадии выполняются параллельно

Stages выполняются последовательно



.gitlab-ci.yml 286 Bytes

```
1 stages:
2   - build
3   - test
4   - deploy
5
6 build_job:
7   stage: build
8   script:
9     - ./build.sh
10
11 test_unit_job:
12   stage: test
13   script:
14     - ./test-unit.sh
15
16 test_integration_job:
17   stage: test
18   script:
19     - ./test-integration.sh
20
21 deploy_job:
22   stage: deploy
23   script:
24     - ./deploy.sh
25
```

# Менее искусственный пример

Приложение на JS фреймворке Angular

The screenshot shows the GitLab interface for a repository named 'angular-example'. The left sidebar contains navigation links: Overview, Details, Activity, Cycle Analytics, Repository, Issues (0), Merge Requests (0), CI / CD, Wiki, Snippets, and Settings. The main content area displays the repository's file structure and commit history. At the top, there are buttons for Star (0), Fork (0), and SSH key. Below this, there are buttons for adding a changelog, license, contribution guide, and setting up CI. The commit history table shows a single commit from '@angular/cli' with the message 'chore: initial commit from @angular/cli' and a commit hash of '269dea84'. The file list includes 'a2e', 'src', '.angular-cli.json', '.editorconfig', '.gitignore', 'README.md', 'karma.conf.js', and 'package.json', all committed 2 minutes ago.

Name	Last commit	Last Update
a2e	chore: initial commit from @angular/cli	2 minutes ago
src	chore: initial commit from @angular/cli	2 minutes ago
.angular-cli.json	chore: initial commit from @angular/cli	2 minutes ago
.editorconfig	chore: initial commit from @angular/cli	2 minutes ago
.gitignore	chore: initial commit from @angular/cli	2 minutes ago
README.md	chore: initial commit from @angular/cli	2 minutes ago
karma.conf.js	chore: initial commit from @angular/cli	2 minutes ago
package.json	chore: initial commit from @angular/cli	2 minutes ago

# Менее искусственный пример

Определим pipeline для приложения

```
.gitlab-ci.yml 354 Bytes
1  image: node:latest
2
3  stages:
4    - build
5    - test
6
7  cache:
8    paths:
9      - node_modules/
10
11 build_job:
12   stage: build
13   script:
14     - npm install
15     - ./node_modules/@angular/cli/bin/ng build
16
17 test_unit_job:
18   stage: test
19   cache:
20     policy: pull
21   script:
22     - ./node_modules/@angular/cli/bin/ng test -
```

Используем Docker executor

Кэшируем временные и не очень временные файлы

Тесты не модифицируют кэш

# Менее искусственный пример

Определим pipeline для приложения

Pipeline Jobs 2

Build Test

✓ build\_job

✓ test\_unit\_job

passed Job #36 triggered 3 minutes ago by Yury Ignatov

Retry

```
Running with gitlab-runner 10.0.2 (a9a75a50)
on dockerz (2ae0c72e)
Using Docker executor with image node:latest ...
Using docker image sha256:6182d4fd366f29cda26b6dae76b301073c358d317bf0d673db61b7c26d5192ac for predefined container...
Pulling docker image node:latest ...
Using docker image node:latest ID=sha256:badd967af535567f92c04665ccbf4ccf63f58960e38a43f611b1b19ca3a713e7 for build container...
Running on runner-2ae0c72e-project-3-concurrent-0 via 811179723cfc...
Fetching changes...
Removing node_modules/
HEAD is now at d5d717c afdasfsaf
From http://ci.asura.ru/otus/angular-example
d5d717c..1af3772 master -> origin/master
Checking out 1af37721 as master...
Skipping Git submodules setup
Checking cache for default...
Successfully extracted cache
$ npm install
npm info it worked if it ends with ok
npm info using npm@5.4.2
npm info using node@v8.7.0
npm info lifecycle angular-example@0.0.0~preinstall: angular-example@0.0.0
npm http fetch GET 200 https://registry.npmjs.org/fsevents/-/fsevents-1.1.2.tgz 584ms
npm info lifecycle abbrev@1.1.0~preinstall: abbrev@1.1.0
```

# Менее искусственный пример

Результаты сборки нужно сохранить

```
.gitlab-ci.yml 427 Bytes
1  image: node:latest
2
3  stages:
4    - build
5    - test
6
7  cache:
8    paths:
9      - node_modules/
10
11 build_job:
12   stage: build
13   script:
14     - npm install
15     - ./node_modules/@angular/cli/bin/ng build
16   artifacts:
17     name: "${CI_PROJECT_NAME}_${CI_COMMIT_SHA}"
18     paths:
19       - dist/
20     expire_in: 1 week
21
22 test_unit_job:
23   stage: test
24   cache:
25     policy: pull
26   script:
27     - echo 'Tests was successfull, trust me'
28   dependencies: []
29
30
```

Определение артефакта

obus > angular-example > Jobs > #36 > Artifacts

passed Job #36 in pipeline #16 for 1af37721 from master by @yignatov 9 minutes ago

Artifacts / dist

Download artifacts archive

Name	Size
..	
favicon.ico	5.3 KB
index.html	817 Bytes
inline.bundle.js	5.6 KB
inline.bundle.js.map	5.7 KB
main.bundle.js	5.9 KB
main.bundle.js.map	3.8 KB
polyfills.bundle.js	200 KB

DevOps консультанты

# Проект с использованием Docker

The screenshot shows the GitLab interface for a repository named 'reddit'. The left sidebar contains navigation links: Overview, Details, Activity, Cycle Analytics, Repository, Issues (0), Merge Requests (0), CI / CD, Wiki, Snippets, and Settings. The main content area displays the repository details, including the SSH URL, a list of files (comment, post.py, ui, README.md), and a commit history table. The commit history table has columns for Name, Last commit, and Last Update. The commit is by 'initial' and was made 'less than a minute ago'. The README.md file is also visible.

GitLab Projects Groups Activity Milestones Snippets

reddit

Unstar 1 Fork 0 SSH git@ci.asura.ru:otus/reddit.git

Files (246 KB) Commit (1) Branch (1) Tags (0) Readme Add Changelog Add License Add Contribution guide Set up CI

master reddit /

History Find file

initial  
Yury Ignatov committed less than a minute ago 6d656b9d

Name	Last commit	Last Update
comment	initial	less than a minute ago
post.py	initial	less than a minute ago
ui	initial	less than a minute ago
README.md	initial	less than a minute ago

README.md

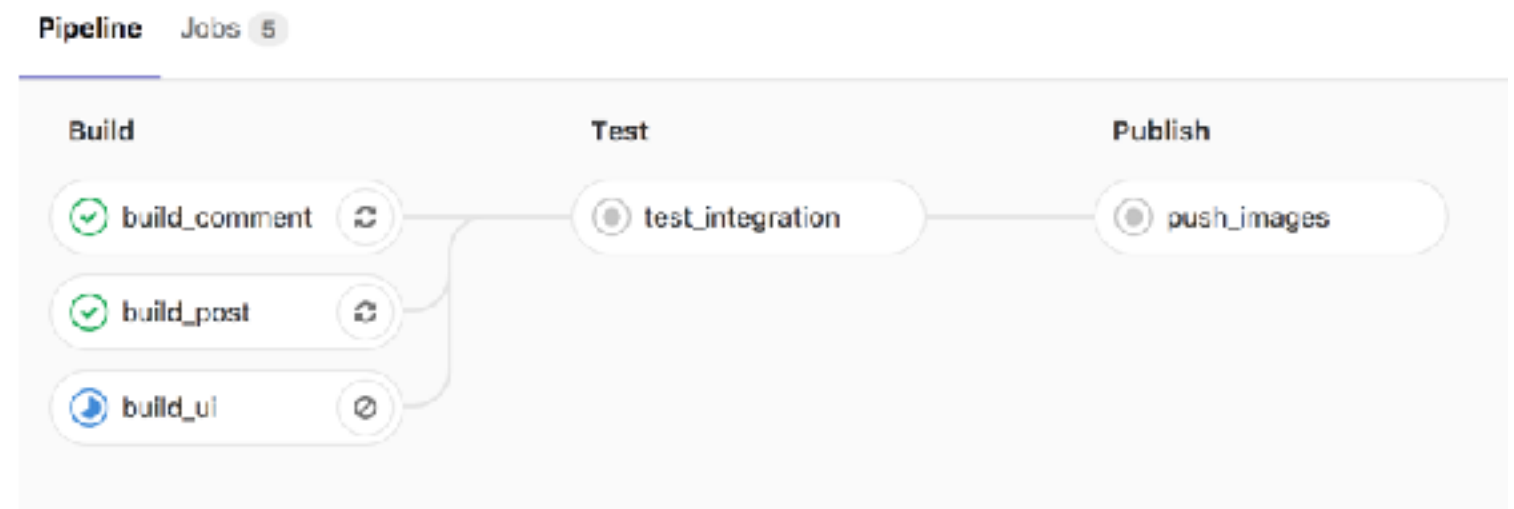
Micorservices



# Проект с использованием Docker

## Опишем Pipeline

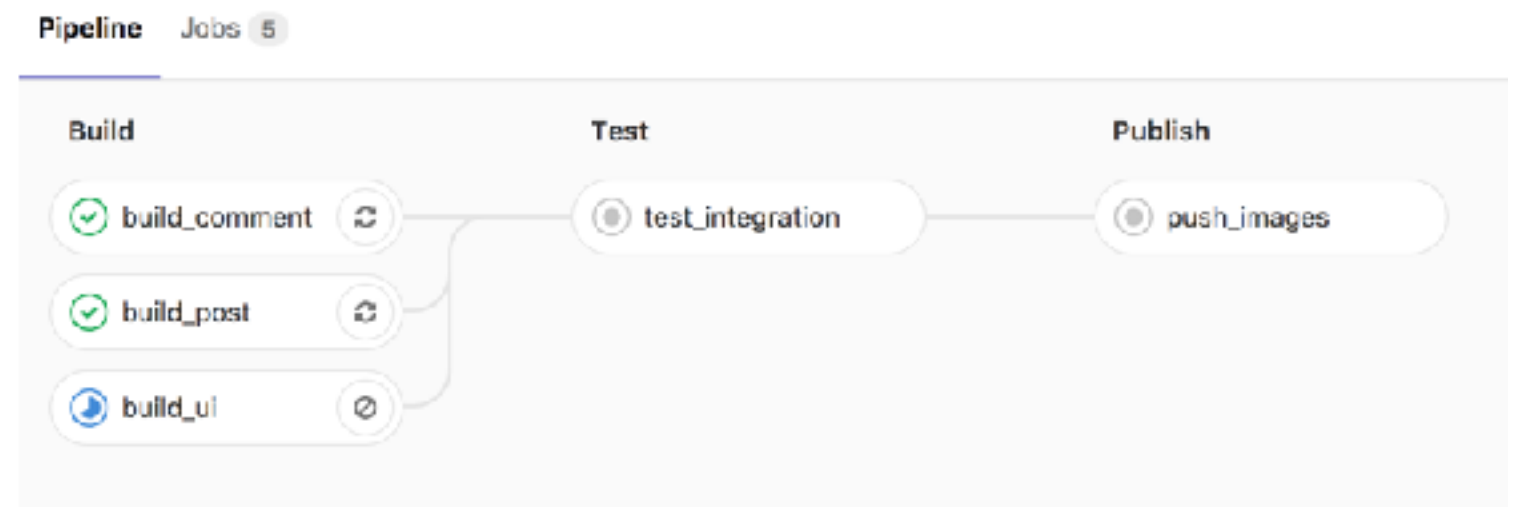
```
.gitlab-ci.yml 470 Bytes
1  image: docker:latest
2
3  stages:
4    - build
5    - test
6    - publish
7
8  build_post:
9    stage: build
10   script:
11     - docker build -t post-py:latest post-py
12
13  build_comments:
14    stage: build
15    script:
16     - docker build -t comment:latest comment
17
18  build_ui:
19    stage: build
20    script:
21     - docker build -t ui:latest ui
22
23  test_integration:
24    stage: test
25    script:
26     - test -d / && echo 'I successfully tested something!'
27
28  push_images:
29    stage: publish
30    script:
31     - docker push post-py
```



# Проект с использованием Docker

## Опишем Pipeline

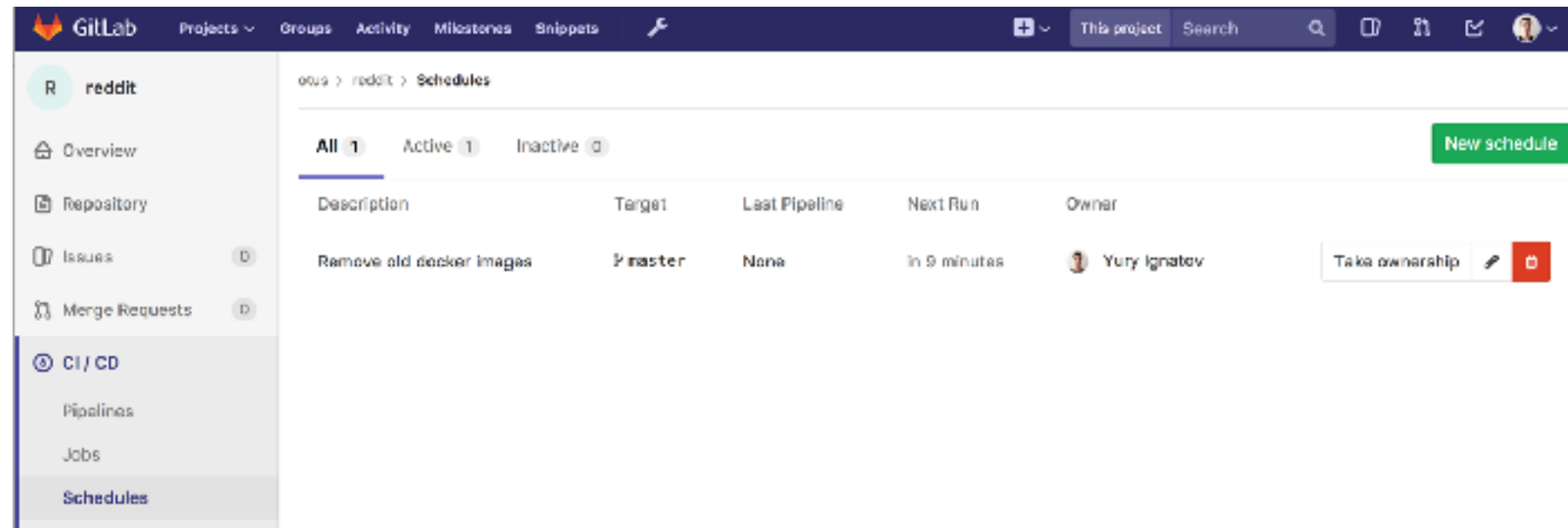
```
.gitlab-ci.yml 470 Bytes
1  image: docker:latest
2
3  stages:
4    - build
5    - test
6    - publish
7
8  build_post:
9    stage: build
10   script:
11     - docker build -t post-py:latest post-py
12
13  build_comments:
14    stage: build
15    script:
16     - docker build -t comment:latest comment
17
18  build_ui:
19    stage: build
20    script:
21     - docker build -t ui:latest ui
22
23  test_integration:
24    stage: test
25    script:
26     - test -d / && echo 'I successfully tested something!'
27
28  push_images:
29    stage: publish
30    script:
31     - docker push post-py
```



Через сколько рабочих дней  
кончится место на сервере?

# Проект с использованием Docker

## Определим Schedule для Pipeline

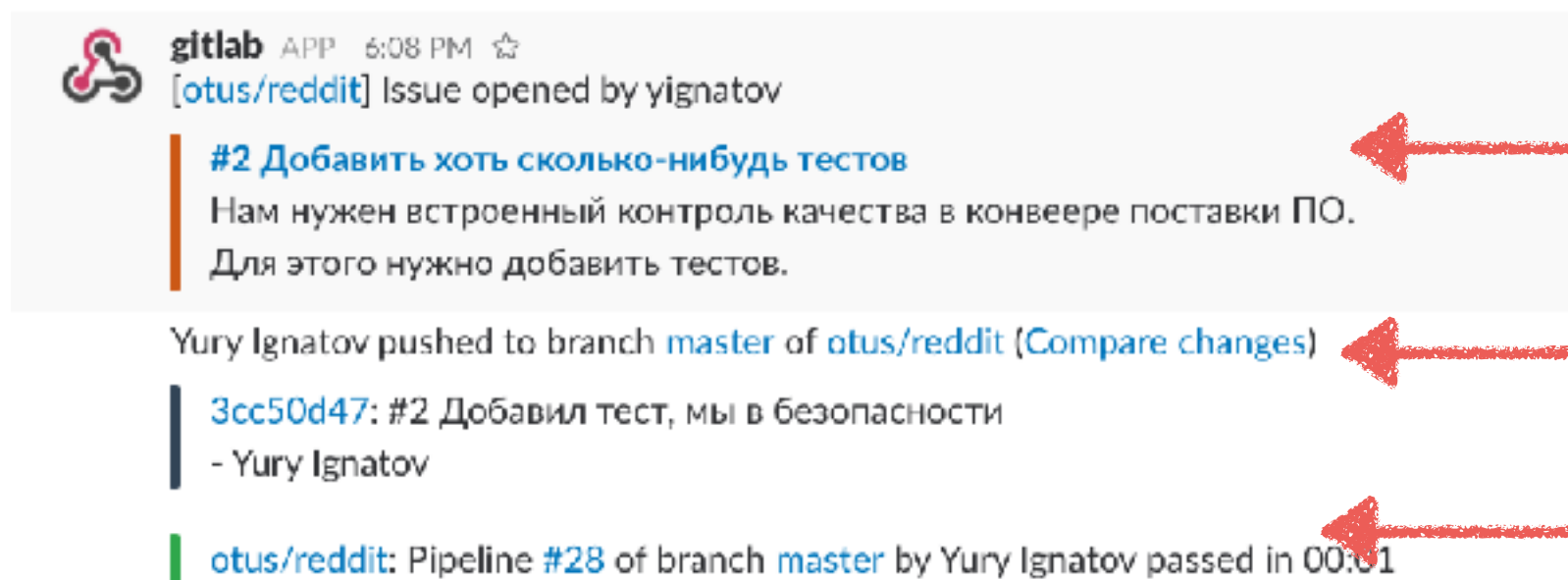


И новый Job в `.gitlab-ci.yml` Не делайте так ;)

```
43  
44  remove_old_images:  
45    script:  
46      - docker images | grep "weeks ago" | awk '{print $3}' | xargs docker rmi  
47    only:  
48      - schedules
```

# Интеграции

- Хорошие интеграции повышают прозрачность
- Много всего из коробки
- Легко интегрируется по HTTP API
- Подходит для ChatOps



The screenshot shows a GitLab interface. At the top, it says 'gitlab APP 6:08 PM ☆' and '[otus/reddit] Issue opened by yignatov'. Below this is a new issue titled '#2 Добавить хоть сколько-нибудь тестов' with the description 'Нам нужен встроенный контроль качества в конвейере поставки ПО. Для этого нужно добавить тестов.' Below the issue, there is a commit message 'Yury Ignatov pushed to branch master of otus/reddit (Compare changes)' followed by a commit hash '3cc50d47: #2 Добавил тест, мы в безопасности' and the author 'Yury Ignatov'. At the bottom, there is a pipeline status 'otus/reddit: Pipeline #28 of branch master by Yury Ignatov passed in 00:01'.

создали задачу

выполнили работу

получили фидбек

# Спасибо!

- Ждем ваших вопросов
- На прошлом занятии мы передохнули от ДЗ, в этот раз придется поработать :)