Pipeline Architecture

Globals | Includes | Before/After | Extends

Global Defaults

default	image | services | before _script | after_ script | cache
variables	Cannot be specified under default
stages	Cannot be specified under default

Job values always override global defaults.

Include

```
include:
   - remote: 'https :// git -
lab.co m/a wes ome -pr oje ct/ -
raw /ma ste r/.b ef ore -sc -
rip t-t emp lat e.yml'
   - local: '/temp lat es/.af -
ter -sc rip t-t emp lat e.yml'
   - template: Auto-D evO ps.g -
it lab -ci.yml
   - project: 'my-gr oup /my -
pr oject'
       ref: master
        file: '/temp lat -
es/.gi tla b-c i-t emp lat -
e.yml'
```

extension: .yml | .yaml

Before and After Scripts

```
default:
    bef ore _sc ript:
       - global before script
job:
    bef ore _sc ript:
       - execute this instead of
global version
    script:
       - my command
    aft er_ script:
       - execute this after my
script
```

Extends

```
.only-important:
    only:
       - master
       - stable
    tags:
       - production
.in-do cker:
    tags:
       - docker
    image: alpine
rspec:
    ext ends:
       - .only- imp ortant
       - .in-docker
    script:
       - rake rspec
spinach:
    ext ends: .in-docker
    script: rake spinach
```

Jobs Management

Stages | Parameters | Environments

Stages

```
stages:
   - .pre
   - build
   - test
   - deploy
   - .post
```

.pre and .post stages are guaranteed to be the first (.pre) or last (.post) stage in a pipeline

Disabling Jobs by Hiding Them

```
.hidden_job:
    script:
       - run test
```

temporarily ‘disable’ a job by prepending a dot (.)

Variables

```
variables:
    ENV IRO NMENT: " sta gin g"
    DB_URL: " pos tgr es: //p -
ost gre s@p ost gres/db
build:
    script: mvn build
    var iables:
        ENV IRO NMENT: " pro -
duc tio n"
```

Environment

```
review_app:
    stage: deploy
    script: make deploy-app
    env iro nment:
        name: review
        on_ stop: stop_r evi -
ew_app
stop_r evi ew_app:
    stage: deploy
    var iables:
        GIT _ST RATEGY: none
    script: make delete-app
    when: manual
    env iro nment:
        name: review
        action: stop
deploy as review app:
```


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Environment (cont)

```
>  stage: deploy
 script: make deploy
 environment:
  name: review/$CI_COMMIT_REF_NAME
  url: https://$CI_ENVIRONMENT_SLUG.e-
xample.com/
```

Pages

```
pages:
    stage: deploy
    script:
       - mkdir .public
       - cp -r * .public
       - mv .public public
    art ifacts:
        paths:
           - public
    only:
       - master
```

Pages is a special job that is used to upload static content to GitLab that can be used to serve your website

Dependencies

```
build:osx:
    stage: build
    script: make build:osx
    art ifacts:
        paths:
           - binaries/
build: linux:
    stage: build
    script: make build: linux
```

Dependencies (cont)

```
>  artifacts:
   paths:
    - binaries/
test:osx:
 stage: test
 script: make test:osx
 dependencies:
  - build:osx
test:linux:
 stage: test
 script: make test:linux
 dependencies:
  - build:linux
deploy:
 stage: deploy
 script: make deploy
```

By default, all artifacts from all previous stages are passed to the current job, but you can use the dependencies parameter to define a limited list of jobs (or no jobs) to fetch artifacts from.

Needs

```
linux:build:
    stage: build
mac:build:
    stage: build
linux: rspec:
    stage: test
    needs: ["li nux :bu ild "]
linux: rub ocop:
    stage: test
    needs: ["li nux :bu ild "]
```

Needs (cont)

```
> mac:rspec:
 stage: test
 needs: ["mac:build"]
mac:rubocop:
 stage: test
 needs: ["mac:build"]
production:
 stage: deploy
```

The needs: keyword enables executing jobs out-of-order, allowing you to implement a directed acyclic graph. This lets you run some jobs without waiting for other ones, disregarding stage ordering so you can have multiple stages running concurrently.

Tags

```
job:
    tags:
       - ruby
       - postgres
osx job:
    stage:
       - build
    tags:
       - osx
    script:
       - echo " Hello, $USER! "
```

tags is used to select specific Runners from the list of all Runners that are allowed to run this project. During the registration of a Runner, you can specify the Runner's tags, for example ruby, postgres, windows, osx.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Trigger

```
staging:
    stage: deploy
    tri gger: my/dep loyment
stagin g-b ranch:
    stage: deploy
    tri gger:
        pro ject: my/dep loyment
        branch: stable
```

trigger allows you to define downstream pipeline trigger. When a job created from trigger definition is started by GitLab, a downstream pipeline gets created.

Parallel

```
test:
    par allel: 3
    script:
       - bundle
       - bundle exec rspec_ -
booster --job $CI_NO DE_ IND -
EX/ $CI _NO DE_ TOTAL
```

parallel allows you to configure how many instances of a job to run in parallel. This value has to be greater than or equal to two (2) and less than or equal to 50.

Flow Control

Rules | Retries

rules Evaluation

```
docker build:
    script: docker build -t my-
ima ge: $SLUG .
    rules:
       - changes:
           - Dockerfile
            when: manual
       - if: '$VAR == " string
value"'
            when: manual
       - when: on_success
docker build:
    script: docker build -t my-
ima ge: $SLUG .
    rules:
       - if: '$VAR == " string
value"'
            cha nges:
           - Dockerfile
           - docker /sc ripts/*
            when: manual
```

To conjoin if and changes clauses with an AND, use them in the same rule.

Job Retries

```
test:
    script: rspec
    retry:
       max: 2
        when:
           - runner _sy ste -
m_f ailure
           - stuck_ or_ tim -
eou t_f ailure
```

```
when: always | unknow n_f ailure
 | script _fa ilure | api_fa ilu
re | stuck_ or_ tim eou t_f ailu
re | runner _sy ste m_f ailure |
  missin g_d epe nde ncy _fa ilu
re | runner _un sup ported
```

Interruptible

```
stages:
   - stage1
   - stage2
step-1:
    stage: stage1
    script:
       - echo "Can be cancel ed"

step-2:
    stage: stage2
    script:
       - echo "Can not be
cancel ed"
    int err upt ible: false
```

This value will only be used if the **automatic cancellation of redundant pipelines** feature is enabled.

Protecting Manual Jobs

```
deploy_prod:
    stage: deploy
    script:
       - echo " Deploy to
production server "
    env iro nment:
        name: production
       urh:ttps: //e xam -
ple.com
    when: manual
    only:
       - master
    all ow_ fai lure: false
```

In the protected environments settings, select the environment and add the users, roles or groups that are authorized to trigger the manual job to the Allowed to Deploy list

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Artifact Management

Artifacts | Docker | Cache

Artifacts

```
job:
    art ifacts:
        name: " $CI _JO B_N -
AME "
        paths:
           - binaries/
           - .config
        unt racked: true
        when: on_failure
        exp ire_in: 1 week
code_q uality:
    stage: test
    script: codequ ality /code
    art ifacts:
        rep orts:
            cod equ ality: gl-
cod e-q ual ity -re por t.json
    cov erage: '/Code coverage:
\d+\.\d+/'
```

```
untracked: true | false when: on_
success | on_failure | always |
manual
```

Docker Image

```
image:
    name: super/ sql :ex per -
imental
    ent ryp oint: [""]
```

Docker Service

```
services:
   - name: postgr es:9.4
        alias: db
        ent ryp oint: ["do cke -
r-e ntr ypo int.sh "]
        com mand: ["po stg -
res "]
```

Cache

```
build:
    script: mvn test
    cache:
       key: build
        unt racked: true
        paths:
           - binaries/
        policy: pull
```

```
policy : pull | push | pull-push
untracked : true | false
```


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com