Dependencies

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

To use this feature, define dependencies in context of the job and pass a list of all previous jobs from which the artifacts should be downloaded. You can only define jobs from stages that are executed before the current one. An error will be shown if you define jobs from the current stage or next ones. Defining an empty array will skip downloading any artifacts for that job. The status of the previous job is not considered when using dependencies, so if it failed or it is a manual job that was not run, no error occurs.

In the following example, we define two jobs with artifacts, build:osx and build:linux. When the test:osx is executed, the artifacts from build:osx will be downloaded and extracted in the context of the build. The same happens for test:linux and artifacts from build:linux.

The job deploy will download artifacts from all previous jobs because of the stage precedence:

```
build:osx:
  stage: build
  script: make build:osx
  artifacts:
    paths:
      - binaries/

build:linux:
  stage: build
  script: make build:linux
  artifacts:
```

```yaml
    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
```

When a dependent job will fail

Introduced in GitLab 10.3.

If the artifacts of the job that is set as a dependency have been expired or erased, then the dependent job will fail.