`stage`

Use `stage` to define which stage a job runs in. Jobs in the same `stage` can execute in parallel (see **Additional details**).

If `stage` is not defined, the job uses the `test` stage by default.

Keyword type: Job keyword. You can use it only as part of a job.

Possible inputs: An array including any number of stage names. Stage names can be:

- The default stages.
- User-defined stages.

Example of `stage`:

```yaml
stages:

  - build

  - test

  - deploy



job1:

  stage: build

  script:

    - echo "This job compiles code."
```

```
job2:

  stage: test

  script:

    - echo "This job tests the compiled code. It runs when the build stage
completes."



job3:

  script:

    - echo "This job also runs in the test stage".
```

```
job4:
  stage: deploy
  script:
    - echo "This job deploys the code. It runs when the test stage completes."
```

Additional details:

- Jobs can run in parallel if they run on different runners.
- If you have only one runner, jobs can run in parallel if the runner's `concurrent` setting is greater than `1`.

`stage: .pre`

Introduced in GitLab 12.4.
Use the `.pre` stage to make a job run at the start of a pipeline. `.pre` is always the first stage in a pipeline. User-defined stages execute after `.pre`. You do not have to define `.pre` in `stages`.

If a pipeline contains only jobs in the `.pre` or `.post` stages, it does not run. There must be at least one other job in a different stage.

Keyword type: You can only use it with a job's `stage` keyword.

Example of `stage: .pre`:

```
stages:

  - build

  - test
```

```
job1:
  stage: build
  script:
    - echo "This job runs in the build stage."


first-job:
  stage: .pre
  script:
```

```
    - echo "This job runs in the .pre stage, before all other stages."


job2:
  stage: test
  script:
    - echo "This job runs in the test stage."
```

`stage: .post`

Introduced in GitLab 12.4.

Use the `.post` stage to make a job run at the end of a pipeline. `.post` is always the last stage in a pipeline. User-defined stages execute before `.post`. You do not have to define `.post` in `stages`.

If a pipeline contains only jobs in the `.pre` or `.post` stages, it does not run. There must be at least one other job in a different stage.

Keyword type: You can only use it with a job's `stage` keyword.

Example of `stage: .post`:

```
stages:

  - build

  - test
```

```
job1:
  stage: build
  script:
    - echo "This job runs in the build stage."


last-job:
  stage: .post
  script:
```

```
    - echo "This job runs in the .post stage, after all other stages."


job2:
  stage: test
  script:
    - echo "This job runs in the test stage."
```