The pipeline variables in GitLab

The **pipeline variables** in GitLab are a key element to make CI/CD (Continuous Integration/Continuous Deployment) pipelines more flexible, secure, and customizable. They allow developers to parameterize configurations, manage sensitive credentials, and create more generic and reusable pipelines. In this article, we will explore in detail what they are, how they work, the types of variables available, and some best practices for using them.

Pipeline variables are dynamic values that can be defined and used within `.gitlab-ci.yml` configuration files. These variables can be used to replace static strings, provide sensitive values like API tokens, configure pipeline behaviors, or pass information between different jobs.

In the `.gitlab-ci.yml` file:

```
variables:
  APP_ENV: production
  APP_VERSION: 1.0.0

build_job:
  script:
    - echo "Deploying version $APP_VERSION in
$APP_ENV environment"
```

When the pipeline runs, the command will output:

```
Deploying version 1.0.0 in production environment
```

GitLab offers several types of variables, each with specific uses and configuration methods.

Predefined Variables

These are built-in variables in GitLab and are automatically available for all pipelines. They contain useful information about the project, environment, and job.

Examples:

- `CI_PROJECT_NAME`: Project name.
- `CI_COMMIT_REF_NAME`: Branch or tag name.
- `CI_PIPELINE_ID`: Unique ID of the pipeline.

User-Defined Variables

These variables can be declared directly in the `.gitlab-ci.yml` file or at the project, group, or pipeline level.

Project-level configuration:

1. Go to **Settings > CI/CD > Variables**.
2. Create a new variable by specifying:
    - **Key**: Name of the variable.
    - **Value**: Assigned value.
    - **Scope**: Restricted access to certain pipelines or jobs.

Variables defined at a group level are inherited by all projects within the group.

Environment Variables

Allow specific values to be defined for a particular stage or job.

```
deploy:
  variables:
    APP_ENV: production
  script:
    - echo "Environment: $APP_ENV"
```

Protected Variables

Protected variables can only be used by jobs associated with protected branches or tags. This feature is useful for securely managing credentials or sensitive configurations.

Creation:

- Go to **Settings > CI/CD > Variables**.
- Select the "Protected" option.

Masked Variables

Masked variables hide their values in the pipeline execution logs, making them useful for protecting sensitive information like API keys.

Requirements:

- The value must contain only alphanumeric characters.
- It must not exceed 512 characters.

Secret Variables

Secret variables are ideal for managing sensitive data such as passwords or tokens. They are configurable through the GitLab interface and are not visible in the repository or `.gitlab-ci.yml` files.

Variables Passed from Manual Pipelines

Manual pipelines allow variables to be passed during the start.

```
test:
  script:
    - echo "Running in $MODE"
```

When starting a pipeline manually, you can specify the value of `MODE`.

Variable Priority

When variables with the same name are defined, GitLab assigns priority based on the following order (from highest to lowest):

1. Variables manually defined during pipeline execution.
2. Secret variables configured at the project or group level.
3. Variables defined in the `.gitlab-ci.yml` file.
4. GitLab predefined variables.

Best Practices

1. **Use protected and masked variables for sensitive credentials**: Avoid embedding sensitive information directly in the `.gitlab-ci.yml` file.

2. **Centralize variables at the group or project level**: To avoid redundancy and simplify maintenance.

3. **Document variables**: Maintain an updated list of variables used in the project, along with their purpose.

4. **Restrict access to variables**: Use the protection feature for sensitive branches/tags and mask values in logs.

5. **Make pipelines reusable**: Leverage variables to create generic pipelines adaptable to different scenarios.

Conclusion

GitLab pipeline variables are a powerful tool to make CI/CD processes flexible, secure, and scalable. Proper use requires a good understanding of their features and careful management, especially for sensitive data. By following the best practices outlined, you can enhance the efficiency and security of your pipelines while simplifying code and configuration management.