3. Build and release pipelines

└ 3.7 Maintain and optimize pipelines

└ 3.7.4 Migrate Classic to YAML

---

1. What are the key differences between Classic and YAML pipelines?
2. Why should you migrate from Classic to YAML pipelines?
3. How do you export a Classic pipeline to YAML?
4. Which features in Classic pipelines require manual translation when migrating?
5. How can task groups be converted for YAML pipelines?
6. What tools assist in converting Classic to YAML?
7. What common issues arise when migrating triggers?
8. How do you structure multi-stage YAML pipelines?
9. What are best practices when migrating pipelines incrementally?
10. How do you validate that a migrated YAML pipeline matches original behavior?

---

1. What are the key differences between Classic and YAML pipelines?

- Classic pipelines use a GUI for configuration
- YAML pipelines are defined as code, versioned with source, and support advanced features like templates and multi-stage workflows.

---

2. Why should you migrate from Classic to YAML pipelines?

- YAML enables pipeline-as-code,
- better version control,
- reuse with templates,
- portability,
- and full CI/CD traceability.

---

3. How do you export a Classic pipeline to YAML?

Go to the *Classic pipeline → Edit → View YAML (only available for simple jobs) → Copy and manually refine*. Complex flows require manual recreation.

---

4. Which features in Classic pipelines require manual translation when migrating?

Multi-agent jobs, deployment groups, task groups, approvals, and environment strategies often require re-implementation using YAML stages and environments.

---

5. How can task groups be converted for YAML pipelines?

Recreate task group logic as YAML templates, then reference with `- template: path/to/template.yml` in pipelines.

---

6. What tools assist in converting Classic to YAML?

There's no full auto-converter; use "View YAML" for individual jobs, then rely on docs, pipeline schema references, and manual refactoring.

---

7. What common issues arise when migrating triggers?

Classic pipelines may use UI-based schedules or branch filters that must be manually defined in YAML using trigger and schedules.

8. How do you structure multi-stage YAML pipelines?
Use the stages keyword with each stage defining jobs and steps, enabling CI, CD, approvals, and environment separation in one file.

---

9. What are best practices when migrating pipelines incrementally?

- Start with CI stages, validate output, then add CD stages.
- Keep Classic as fallback until YAML proves stable.
- Use templates for reuse.

---

11. **How do you validate that a migrated YAML pipeline matches original behavior?**
    - Compare build logs, test results, deployment targets, and approval flows side-by-side.
    - Use test branches to validate before cutting over.