**🧩 What Is Trunk-Based Development?**

**➤ Key Idea:**

All developers **commit to a single branch**—typically called main or trunk—frequently (often multiple times per day).

**🔧 How It Works:**

* One long-lived branch: main
* Developers use **short-lived feature branches** (optional), merged quickly (within hours or a day)
* Frequent integration and testing
* Often used with **CI/CD**, **feature flags**, and **automated testing**

**✅ Pros:**

* Fast integration → fewer merge conflicts
* Promotes CI/CD and fast releases
* Simple branching structure
* Easy to automate and test

**❌ Cons:**

* Requires high discipline and testing
* Not ideal for teams without robust CI/CD
* Risky without feature flags (incomplete code might go live)

**🧭 What Is Git Flow?**

**➤ Key Idea:**

Git Flow defines **multiple long-lived branches** for structured development, release, and hotfix workflows.

**🏗 Branch Types:**

* main: production-ready code
* develop: integration of all features
* feature/\*: feature branches
* release/\*: pre-production testing
* hotfix/\*: urgent production fixes

**✅ Pros:**

* Clear structure and separation of work
* Useful for traditional release cycles
* Good for large teams with slower, staged releases

**❌ Cons:**

* More complex branching (can be hard to manage)
* Slower integration (can lead to big merge conflicts)
* Less suitable for continuous delivery

**⚔️ Trunk-Based vs Git Flow: Side-by-Side**

| **Feature** | **Trunk-Based Development** | **Git Flow** |
| --- | --- | --- |
| **Branching Complexity** | Simple (1 main branch) | Complex (5+ branches) |
| **Merge Frequency** | Very frequent (daily/hourly) | Less frequent |
| **Release Cycle** | Continuous Delivery | Scheduled Releases |
| **Best For** | CI/CD, DevOps, Agile teams | Structured, slower delivery teams |
| **Merge Conflicts** | Minimal (short branches) | Higher (long-lived branches) |
| **Feature Flags Needed?** | Yes (usually) | Not necessarily |
| **Tooling Required** | Strong CI/CD, testing tools | Basic Git tools enough |

**🎯 When to Use What?**

| **Use Case** | **Recommended Strategy** |
| --- | --- |
| You want fast, frequent releases | ✅ **Trunk-Based** |
| You have strong CI/CD pipelines | ✅ **Trunk-Based** |
| You have large enterprise app with staged QA/release | ✅ **Git Flow** |
| Your team is new to Git or prefers structured workflow | ✅ **Git Flow** |

**✅ Recommendation**

If you're doing **modern DevOps**, **agile** development, and have a good **CI/CD setup**—go for **Trunk-Based Development**.

If you're in a **regulated environment**, do **manual testing**, or have **staged releases**, then **Git Flow** may fit better.