Git Submodules

Reusing Others' Git Repositories

What are Git Submodules?

Git submodules allow you to **include one Git repository inside another** as a subdirectory.

- Your main project needs external libraries or components
- Instead of copying code, you link to another repository
- Changes in the external repo can be tracked and updated

Real-World Example

Scenario: Building a web application

- The `shared-components` folder contains a separate Git repository used across multiple projects.

```
my-webapp/
├── src/
├── docs/
├── shared-components/      ← Git submodule
│   ├── buttons/
│   └── navigation/
└── README.md
```

Why Use Submodules?

- **Code Reuse** - Share common libraries across projects
- **Version Control** - Pin to specific versions of dependencies
- **Separation** - Keep external code separate from your main project
- **Team Collaboration** - Everyone gets the same dependency versions

Common Use Cases:

- Shared UI component libraries
- Third-party libraries
- Documentation repositories
- Configuration files

Adding a Submodule

Basic Syntax:

```
git submodule add <repository-url> <path>
```

Example:

```
# Add a shared components repository
git submodule add
  https://github.com/company/shared-ui.git shared-components
```

This creates:

- A new folder with the external repository
- A `.gitmodules` file tracking submodule configuration

Cloning a Project with Submodules

Solution 1: Clone with submodules

```
git clone --recurse-submodules
  https://github.com/user/main-project.git
```

Solution 2: Initialize after cloning

```
git clone https://github.com/user/main-project.git
cd main-project
git submodule update --init --recursive
```

Updating Submodules

- pulling the shared-components
- Then commit to store the **current** version.

```
cd shared-components
git pull origin main
cd ..
git add shared-components
git commit -m "Update shared-components to latest version"
```

Update all submodules at once:

```
git submodule update --remote
```

Update and initialize in one command:

```
git submodule update --init --recursive --remote
```