Guide des Commandes

Cargo & Clippy

Référence complète pour Rust

Framework Rusti - 2025

■ Commandes Cargo Essentielles

Compilation et Build

```
cargo build
```

Compiler le projet en mode debug

```
cargo build --release
```

Compiler en mode release (optimisé)

```
cargo build --all-features
```

Compiler avec toutes les features

```
cargo build --no-default-features
```

Compiler sans les features par défaut

Vérification et Tests

```
cargo check
```

Vérifier que le code compile (plus rapide)

```
cargo check --all-targets
```

Vérifier toutes les targets

```
cargo test
```

Lancer les tests

```
cargo test -- --nocapture
```

Tests avec sortie détaillée

```
cargo test --lib
```

Tests unitaires uniquement

```
cargo test --doc
```

Tests de documentation

Exécution

```
cargo run
```

Exécuter le projet

```
cargo run -- arg1 arg2
```

Exécuter avec arguments

```
cargo run --release
```

Exécuter en mode release

```
cargo run --example example_name
```

Exécuter un exemple

Documentation

```
cargo doc
```

Générer la documentation

```
cargo doc --open
```

Générer et ouvrir dans le navigateur

```
cargo doc --no-deps
```

Générer sans les dépendances

Formatage

```
cargo fmt
```

Formater tout le code

```
cargo fmt --all
```

Formater tous les packages

```
cargo fmt -- --check
```

Vérifier sans modifier

■ Commandes Clippy Essentielles

Vérifications de Base

```
cargo clippy
```

Lancer clippy

```
cargo clippy --all-targets
```

Clippy sur toutes les targets

```
cargo clippy --all-features
```

Clippy avec toutes les features

```
cargo clippy -- -D warnings
```

Traiter les warnings comme erreurs

```
cargo clippy --all-targets --all-features -- -D warnings
```

Clippy complet (CI/CD)

Niveaux de Lint

```
cargo clippy -- -A clippy::lint_name
```

Autoriser un warning

```
cargo clippy -- -D clippy::lint_name
```

Refuser (erreur)

```
cargo clippy -- -F clippy::lint_name
```

Forcer un warning

```
cargo clippy -- -D warnings
```

Interdire tous warnings

Catégories de Lints

```
cargo clippy -- -W clippy::style
```

Lints de style

```
cargo clippy -- -W clippy::complexity
```

Lints de complexité

```
cargo clippy -- -W clippy::perf
```

Lints de performance

```
cargo clippy -- -W clippy::correctness
```

Lints de correction

```
cargo clippy -- -W clippy::pedantic
```

Tous les lints pedantic

Fix Automatique

```
cargo clippy --fix
```

Corriger automatiquement

```
cargo clippy --fix --allow-dirty
```

Fix avec modifications non committées

```
cargo clippy --fix --all-targets
```

Fix sur toutes les targets

■ Workflows Recommandés

Avant chaque commit

```
cargo fmt --all cargo clippy --all-targets --all-features -- -D warnings cargo test --all-features
```

Avant publication sur crates.io

```
cargo clean cargo fmt --all cargo clippy --all-targets --all-features -- -D warnings cargo test --all-features cargo doc --no-deps cargo package --list cargo publish --dry-run
```

Pipeline CI/CD

```
cargo fmt -- --check cargo clippy --all-targets --all-features -- -D warnings cargo test --all-features cargo doc --no-deps cargo build --release
```

■ Tableau Récapitulatif

Commande	Description
`cargo build`	Compiler le projet
`cargo run`	Compiler et exécuter
`cargo test`	Lancer les tests
`cargo check`	Vérifier la compilation
`cargo fmt`	Formater le code
`cargo clippy`	Lint le code
`cargo clean`	Nettoyer les artefacts
`cargo doc`	Générer la documentation
`cargo update`	Mettre à jour les dépendances
`cargo publish`	Publier sur crates.io