**🚀 FINMATH - GUIA RÁPIDO (CHEAT SHEET)**

**Referência Rápida para Uso Diário**  
**Versão:** 1.1  
**Data:** 23/Outubro/2025

**⚡ COMANDOS MAIS USADOS**

**Início do Dia**

cd ~/workspace/fin-math

git status

git branch # Confirmar: sprint-X

find . -name "\*bak\*" -type f -delete # Limpar backups

pnpm -F @finmath/engine test # Verificar testes

**Durante Desenvolvimento**

# Testar frequentemente

pnpm -F @finmath/engine test:unit

pnpm -F @finmath/engine test:golden

# Commit frequente

git add .

git commit -m "feat(HXX): descrição"

# Verificar qualidade

pnpm -F @finmath/engine lint

pnpm -F @finmath/engine typecheck

**API Development**

# Build

pnpm -F @finmath/api build

# Dev Server (porta 3001)

pnpm -F @finmath/api dev

# Testar endpoint

curl -X POST http://localhost:3001/api/price \

-H "Content-Type: application/json" \

-d '{"pv": 10000, "rate": 0.02, "n": 12}'

# Swagger UI

open http://localhost:3001/api-docs

**📋 WORKFLOWS ESSENCIAIS**

**Iniciar Sprint**

git fetch origin

git pull origin main

git checkout -b sprint-X

find . -name "\*bak\*" -type f -delete

pnpm install

pnpm -F @finmath/engine test

**Implementar História**

# 1. Escrever código

# 2. Escrever testes

# 3. Rodar testes

pnpm -F @finmath/engine test

# 4. Commit (SE testes passarem)

git add .

git commit -m "feat(HXX): descrição

- Motor: ...

- Testes: ...

- Golden Files: ...

DoD: 5/5 critérios atendidos"

**Finalizar Sprint**

# 1. Limpar backups

find . -name "\*bak\*" -type f -delete

# 2. Validação COMPLETA

pnpm -F @finmath/engine typecheck

pnpm -F @finmath/engine lint

pnpm -F @finmath/engine test:unit

pnpm -F @finmath/engine test:property

pnpm -F @finmath/engine test:integration

pnpm -F @finmath/engine test:golden

pnpm -F @finmath/engine build

pnpm -F @finmath/api build

pnpm -F @finmath/api test:e2e

# 3. Merge e Push

git checkout main

git merge sprint-X --no-ff -m "chore: Merge Sprint X"

git push origin main

# 4. Deletar branch

git branch -d sprint-X

**❌ NUNCA FAZER**

❌ cp file.ts file.ts.bak # Backups físicos

❌ git push origin sprint-X # Push durante sprint

❌ git commit --no-verify # Bypass hooks

❌ git push --force # Force push

❌ const x = 10000 \* 0.025; # Float para cálculos

❌ git add PRICE\_001.json && git commit # Golden File sem revisão

❌ function calc(x: any): any # TypeScript any

❌ console.log('debug:', x) # console.log em prod

**✅ SEMPRE FAZER**

✅ find . -name "\*bak\*" -type f -delete # Limpar backups diariamente

✅ git add . && git commit # Commits frequentes

✅ pnpm test && git commit # Testar antes de commit

✅ import { Decimal } from 'decimal.js' # decimal.js para cálculos

✅ /\*\* @param {Decimal} pv \*/ # JSDoc em funções

✅ ./finalizar-sprint.sh # Validação completa antes de push

**📝 TEMPLATES DE COMMIT**

**Feature**

git commit -m "feat(H9): Implementa Price PMT

- Motor: calculatePMT com decimal.js

- Testes: 8 unitários + 3 propriedade + 5 golden files

- Cobertura: 87%

DoD: 5/5 critérios atendidos

Referências: Guia CET — SoT (§4)"

**Fix**

git commit -m "fix(H9): Corrige arredondamento em ajuste final

Problema: resíduo de 0.02 em alguns casos

Solução: aplicar round2 antes de comparação

Testes: adiciona PRICE\_006 (caso extremo)"

**Docs**

git commit -m "docs(H9): Adiciona JSDoc em calculatePMT

- Documenta parâmetros e retorno

- Adiciona exemplo de uso

- Atualiza OpenAPI"

**🔍 TROUBLESHOOTING RÁPIDO**

| **Problema** | **Solução** |
| --- | --- |
| pnpm install falha | pnpm store prune && rm -rf node\_modules && pnpm install |
| Testes falhando | pnpm -F @finmath/engine clean && pnpm -F @finmath/engine build && pnpm -F @finmath/engine test |
| TypeScript errors | rm -rf packages/\*/tsconfig.tsbuildinfo && pnpm -F @finmath/engine typecheck |
| Build falha | rm -rf packages/\*/dist && pnpm -F @finmath/engine build |
| Git conflicts | git merge --abort && git fetch origin && git reset --hard origin/main |
| Swagger não abre | lsof -i :3001 e kill -9 <PID> |
| Testes lentos | pnpm -F @finmath/engine test -- --maxWorkers=4 |

**📊 CHECKLIST DIÁRIO**

□ Limpei backups físicos?

□ Fiz commits locais das mudanças?

□ Testes passam localmente?

□ JSDoc atualizado nas funções novas?

□ OpenAPI atualizado (se novo endpoint)?

□ Golden Files identificados (se mudança)?

**📊 CHECKLIST DE HISTÓRIA**

Implementação:

□ Motor implementado

□ Testes unitários (≥3 casos)

□ Testes de propriedade

□ Golden File criado/atualizado

□ API implementada (se aplicável)

Documentação:

□ JSDoc em funções públicas

□ OpenAPI atualizado (se API)

□ ADR criado (se decisão estrutural)

Qualidade:

□ npm run test passa

□ npm run lint passa

□ npm run build passa

□ Golden Files passam

□ Cobertura mantida (≥80%)

Git:

□ Commit local feito

□ Mensagem convencional

□ Nenhum .bak commitado

**📊 CHECKLIST FINAL DE SPRINT**

Validação:

□ typecheck: PASSOU

□ lint: PASSOU

□ test:unit: PASSOU

□ test:property: PASSOU

□ test:integration: PASSOU

□ test:golden: PASSOU (X/X)

□ build: PASSOU

□ test:e2e: PASSOU

□ Swagger UI: ACESSÍVEL

Documentação:

□ README atualizado

□ OpenAPI sincronizado

□ ADRs criados (se necessário)

Golden Files:

□ Novos GFs revisados

□ motorVersion atualizado

□ Tolerâncias respeitadas

Limpeza:

□ Nenhum .bak

□ Nenhum console.log

□ Nenhum TODO/FIXME crítico

Git:

□ Merge na main feito

□ Mensagem descritiva

**🔗 LINKS IMPORTANTES**

* **GitHub**: https://github.com/PrinceOfEgypt1/fin-math
* **Swagger UI**: http://localhost:3001/api-docs
* **Guia Completo**: FINMATH\_WORKFLOW\_COMPLETO\_v1.1.md
* **Guia CET**: docs/cet-sot/guia-cet-v1.0.md
* **ADRs**: docs/adr/
* **Backlog**: docs/backlog/sprint-X.md

**📂 ESTRUTURA DE ARQUIVOS**

fin-math/

├── packages/

│ ├── engine/

│ │ ├── src/ # Código fonte do motor

│ │ ├── test/

│ │ │ ├── unit/ # Testes unitários

│ │ │ ├── property/ # Testes de propriedade

│ │ │ ├── integration/ # Testes de integração

│ │ │ └── golden/ # Golden files

│ │ └── package.json

│ └── api/

│ ├── src/ # Código fonte da API

│ ├── routes/ # Rotas da API

│ └── validators/ # Schemas Zod

├── docs/

│ ├── adr/ # Architecture Decision Records

│ ├── api/ # Documentação da API

│ ├── backlog/ # Backlogs das sprints

│ └── cet-sot/ # Guia CET Source of Truth

├── openapi-3.1\_finmath-v1.0.yaml

└── README.md

**💡 DICAS RÁPIDAS**

**Desenvolvimento**

* Commits frequentes (1 por funcionalidade)
* Testes antes de cada commit
* Limpar backups diariamente
* Documentar durante implementação

**Git**

* Branch sprint-X para cada sprint
* Push APENAS ao final da sprint
* Merge --no-ff para histórico claro
* Mensagens de commit descritivas

**Testes**

* Unitários: lógica isolada
* Propriedade: invariantes
* Golden Files: regressão
* E2E: fluxo completo

**Qualidade**

* Cobertura ≥ 80%
* Lint sem warnings
* TypeScript sem erros
* Build sem falhas

**🎯 FLUXO DE UMA SPRINT**

Início Durante Fim

↓ ↓ ↓

Pull → Implementar → Commit → Validação → Push

origin/main Histórias Local Completa GitHub

↓ ↓ ↓ ↓ ↓

Branch Testar Frequente Anti-Reg 1x Sprint

sprint-X Local Completa

**⏱️ ESTIMATIVAS DE TEMPO**

| **Atividade** | **Tempo** |
| --- | --- |
| Implementar história | 2-4 horas |
| Escrever testes | 1-2 horas |
| Criar golden files | 30min-1h |
| Documentar (JSDoc + OpenAPI) | 30min |
| Code review | 30min-1h |
| Validação anti-regressão | 10-15min |
| **Total por história** | **4-8 horas** |

**📞 AJUDA RÁPIDA**

**Problema com Git?**

git status # Ver estado

git log --oneline # Ver commits

git diff # Ver mudanças

**Problema com testes?**

pnpm -F @finmath/engine test -- --verbose # Ver detalhes

pnpm -F @finmath/engine test:unit -- --watch # Watch mode

**Problema com build?**

rm -rf packages/\*/dist # Limpar build

pnpm -F @finmath/engine build --verbose # Build verbose

**Problema com API?**

lsof -i :3001 # Ver o que está na porta

pkill -f "node.\*api" # Matar processos node

**🎓 CONVENÇÕES**

**Commits**

* feat(HXX): - Nova funcionalidade
* fix(HXX): - Correção de bug
* docs(HXX): - Documentação
* test(HXX): - Testes
* refactor(HXX): - Refatoração
* chore: - Manutenção (sem feat/fix)

**Branches**

* main - Produção
* sprint-X - Sprint em andamento
* hotfix-\* - Correções urgentes

**Arquivos**

* \*.test.ts - Testes
* \*.schema.ts - Validações Zod
* ADR-XXX-\*.md - Decisões arquiteturais
* PRICE\_001.json - Golden files

**📖 REFERÊNCIAS**

Para mais detalhes, consultar:

* **Guia Completo**: FINMATH\_WORKFLOW\_COMPLETO\_v1.1.md
* **Guia CET**: docs/cet-sot/guia-cet-v1.0.md
* **Contratos API**: docs/api/contratos.md
* **ADRs**: docs/adr/

**📌 Mantenha este guia sempre à mão para consultas rápidas!**

**Última atualização:** 23/Outubro/2025  
**Versão do documento completo:** 1.1