

OPNet Development Quick Start Guide

How to Build on Bitcoin L1 with Claude Code

Prerequisites

- Node.js v18+
- A Claude account (Pro \$20/mo or Max \$100–200/mo recommended)
- Basic TypeScript knowledge

Step 1: Install Claude Code

```
npm install -g @anthropic-ai/claude-code
```

Step 2: Install the Official OPNet Skill

```
npx skills add https://github.com/btc-vision/opnet-skills --skill  
opnet-development
```

This is the official OPNet development skill maintained by the core team. It teaches Claude all the contract patterns, package versions, security rules, frontend/backend conventions, and OPNet-specific gotchas.

Browse skills: <https://skills.sh/btc-vision/opnet-skills/opnet-development>

Step 3: Create Your Project

```
mkdir my-project && cd my-project
```

Step 4: Add the CLAUDE.md Cheat Sheet

Download from the MOTO Whales Chat pinned messages and save as `CLAUDE.md` in your project root. This gives Claude instant knowledge of:

- Correct package versions (never guesses)
- AssemblyScript contract patterns
- SafeMath, storage pointers, SHA256 selectors
- Frontend stack (Vite + React + OPWallet)
- Backend rules (hyper-express only)
- Library restrictions (what to use, what to avoid)

Step 5: Reference Code

Clone working OPNet projects to study and copy patterns:

```
git clone https://github.com/bclplainview/opsea-marketplace
git clone https://github.com/bclplainview/opswitch-v2
git clone https://github.com/bclplainview/opnet-analyzer
git clone https://github.com/bclplainview/opnet-audit-kit
git clone https://github.com/bclplainview/opspec
git clone https://github.com/bclplainview/op-scribe
git clone https://github.com/bclplainview/ordinals-bridge-plugin
```

Step 6: Start Building

```
claude
```

Tell Claude what you want. With the OPNet skill + CLAUDE.md loaded, it already knows all OPNet conventions.

Example prompts:

- "Build me an OP20 token called MyToken with 1 billion supply"
- "Create a staking contract where users deposit OP20 tokens and earn rewards"
- "Build a Vite + React frontend with OPWallet integration"
- "Audit this contract for security issues"

Step 7: Compile Your Contract

```
npm install
npx asc src/index.ts --config asconfig.json
```

Output: `build/MyContract.wasm` + `abis/MyContract.abi.json`

Step 8: Analyze Your Contract

```
npx skills add bclplainview/opnet-analyzer
```

Or clone and run directly:

```
git clone https://github.com/bclplainview/opnet-analyzer
node opnet-analyzer/dist/cli.js src/MyContract.ts
```

Step 9: Get It Reviewed

Bring your compiled contract back to the **MOTO Whales Chat** for a free audit from Bob.

Key Resources

- OPNet Docs: <https://docs.opnet.org>
- Skills Directory: <https://skills.sh>
- OPNet Skill: <https://skills.sh/btc-vision/opnet-skills/opnet-development>

- **Reference Repos:** <https://github.com/bc1plainview>
- **OPNet GitHub:** <https://github.com/btc-vision>

RPC Endpoints

- **Regtest:** <https://regtest.opnet.org>
- **Mainnet:** <https://mainnet.opnet.org>

Package Versions (Current)

```
@btc-vision/btc-runtime: 1.11.0-rc.4
@btc-vision/transaction: 1.8.0-beta.10
@btc-vision/bitcoin: 7.0.0-alpha.11
@btc-vision/walletconnect: 1.1.0-beta.10
opnet: 1.8.1-beta.13
assemblyscript: 0.27.36
```