Build Your First DeAI Dapp 完整指南

环境配置步骤

Step 1：安装 AI 开发工具

选择一个：

- **Cursor**：VS Code fork，AI 原生编辑器
- **Claude Code**：命令行 AI 助手
- **Antigravity**：0G 官方推荐

Step 2：配置 AI Context： https://docs.0g.ai/ai-context

将 0G 文档加入项目上下文： https://docs.0g.ai/ai-context

```
# 创建 AI Context 文件
mkdir -p .claude
curl -o .claude/0g-context.md https://docs.0g.ai/ai-context
```

或在 Cursor 中添加：

```
// .cursor/settings.json
{
  "ai.contextFiles": [
    "https://docs.0g.ai/ai-context"
  ]
}
```

Step 3：配置 0G 开发环境：

```
# 创建 .env 文件
cat > .env << 'EOF'
# 0G Galileo 测试网配置
RPC_URL=https://evmrpc-testnet.0g.ai
CHAIN_ID=16602
INDEXER_RPC=https://indexer-storage-testnet-turbo.0g.ai
PRIVATE_KEY=你的私钥

# 可选：主网配置
```

```
# 可选：主网配置
# RPC_URL=https://evmrpc.0g.ai
# CHAIN_ID=16661
EOF
```

Step 4：获取测试代币

访问 https://faucet.0g.ai 获取测试网代币。

四、0G 模版快速启动

官方 Starter Kit 一览

模版	语言/框架	核心功能	适用场景
0g-storage-ts-starter-kit	TypeScript + Express	存储 API	Node.js 后端
0g-storage-web-starter-kit	React + Next.js	存储前端	Web DApp
0g-storage-go-starter-kit	Go + Gin	存储 API	高性能后端
0g-compute-ts-starter-kit	TypeScript + Express	AI 推理 + 支付	AI 应用

推荐：TypeScript Storage Starter Kit

```
# 克隆模版
git clone https://github.com/0gfoundation/0g-storage-ts-starter-kit
cd 0g-storage-ts-starter-kit

# 安装依赖
pnpm install

# 配置环境变量
cp .env.example .env
# 编辑 .env，填入你的 PRIVATE_KEY

# 运行测试
pnpm test
```

模版功能详解

1. Storage TypeScript Starter Kit

- Express REST API + Swagger 文档
- 文件上传/下载
- @0glabs/0g-ts-sdk v0.3.3 集成
- 支持 Testnet 和 Mainnet

2. Storage Web Starter Kit

- Next.js + React + TailwindCSS
- 钱包集成（MetaMask 等）
- 拖拽上传组件
- 费用估算与显示
- 模块化 Hooks：`useUpload`、`useDownload`、`useFees`

3. Compute TypeScript Starter Kit

- 7+ AI 模型支持（DeepSeek V3、GPT-OSS、Gemma、Whisper、Flux）
- 自动支付处理（微支付）
- TEE 可信执行环境验证
- 账本管理（存款、退款、资金转移）

五、0G 基础设施集成指南

网络配置

Galileo 测试网（推荐开发）

配置项	值
Chain ID	16602
RPC Endpoint	https://evmrpc-testnet.0g.ai
Storage 起始区块	326165
Faucet	https://faucet.0g.ai

Explorer	https://chainscan-galileo.0g.ai

Aristotle 主网

配置项	值
Chain ID	16661
RPC Endpoint	https://evmrpc.0g.ai
Storage 起始区块	2,387,557
Explorer	https://chainscan.0g.ai

5.1 0G Storage SDK

安装

```
# TypeScript
npm install @0glabs/0g-ts-sdk ethers

# Python
pip install 0g-storage-client

# Go
go get github.com/0gfoundation/0g-storage-client
```

文件上传示例

```
import { ZgFile, Indexer } from '@0glabs/0g-ts-sdk';
import { ethers } from 'ethers';

// 初始化
const provider = new ethers.JsonRpcProvider('https://evmrpc-testnet.0g.ai');
const signer = new ethers.Wallet(process.env.PRIVATE_KEY!, provider);
const indexer = new Indexer('https://indexer-storage-testnet-turbo.0g.ai');

// 创建文件并获取 Merkle 树
const file = await ZgFile.fromFilePath('/path/to/file');
const [tree, err] = await file.merkleTree();
```

```
const [tree, err] = await file.merkleTree();
console.log("Root Hash:", tree.rootHash());

// 上传文件
const [tx, uploadErr] = await indexer.upload(
  file,
  'https://evmrpc-testnet.0g.ai',
  signer
);
await file.close();
```

文件下载示例

```
const err = await indexer.download(
  '<root_hash>',
  '/output/path',
  true  // 验证数据
);
```

KV 存储示例

```
const [nodes, err] = await indexer.selectNodes(1);
const batcher = new Batcher(1, nodes, flowContract, evmRpc);
batcher.streamDataBuilder.set("0x...", key, value);
const [tx, batchErr] = await batcher.exec();
```

5.2 0G Storage CLI

安装

```
git clone https://github.com/0glabs/0g-storage-client.git
cd 0g-storage-client
go build
```

上传文件

```
./0g-storage-client upload \
  --url https://evmrpc-testnet.0g.ai \
  --key $PRIVATE_KEY \
  --indexer https://indexer-storage-testnet-turbo.0g.ai \
```

```
  --indexer https://indexer-storage-testnet-turbo.0g.ai \
  --file ./my-file.txt
```

下载文件

```
./0g-storage-client download \
  --indexer https://indexer-storage-testnet-turbo.0g.ai \
  --root <merkle-root-hash> \
  --file ./output.txt
```

5.3 0G Compute（AI 推理）

安装

```
pnpm add @0glabs/0g-serving-broker

# 全局安装 CLI
pnpm add @0glabs/0g-serving-broker -g
```

Web UI 快速启动

```
0g-compute-cli ui start-web
# 访问 http://localhost:3090
```

SDK 集成示例

```
import { createZGComputeNetworkBroker } from '@0glabs/0g-serving-broker';

// 初始化
const broker = await createZGComputeNetworkBroker(signer);

// 发现服务
const services = await broker.inference.listServices();

// 确认服务提供者
await broker.inference.acknowledgeProviderSigner(providerAddress);

// 获取请求头（用于 OpenAI 兼容 API）
const headers = await broker.inference.getRequestHeaders(
  providerAddress,
```

```
  serviceName,
  content
);

// 发送推理请求（OpenAI 兼容）
const response = await fetch(serviceUrl, {
  method: 'POST',
  headers: { ...headers, 'Content-Type': 'application/json' },
  body: JSON.stringify({
    messages: [{ role: 'user', content: 'Hello!' }]
  })
});

// 验证响应
const isValid = await broker.inference.processResponse(chatId);
```

5.4 0G DA（数据可用性层）

特性

- **50 Gbps** 吞吐量
- VRF 节点选择
- 支持 OP Stack 和 Arbitrum Nitro 集成

Rust SDK 安装

```
cargo add 0g-da-rust-sdk
```

六、0G Chain 智能合约部署

为什么选择 0G Chain?

性能优势：

- **2,500+ TPS**：高于以太坊主网
- **低费用**：仅为主网费用的一小部分
- **亚秒级确认**：1-2 秒区块时间

EVM 兼容性：

- 支持 Pectra & Cancun-Deneb
- 使用熟悉的工具：Hardhat、Foundry、Remix
- 现有以太坊代码无需修改即可运行

Step 1：准备智能合约

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.19;

contract MyToken {
    mapping(address => uint256) public balances;
    uint256 public totalSupply;

    constructor(uint256 _initialSupply) {
        totalSupply = _initialSupply;
        balances[msg.sender] = _initialSupply;
    }

    function transfer(address to, uint256 amount) public returns (bool) {
        require(balances[msg.sender] >= amount, "Insufficient balance");
        balances[msg.sender] -= amount;
        balances[to] += amount;
        return true;
    }
}
```

Step 2：编译合约

重要：使用 `--evm-version cancun` 确保兼容性。

Hardhat 配置：

```
// hardhat.config.js
module.exports = {
  solidity: {
    version: "0.8.19",
    settings: {
      evmVersion: "cancun",
      optimizer: {
        enabled: true
```

```
        enabled: true,
        runs: 200,
      },
    },
  },
  networks: {
    "0g-testnet": {
      url: "https://evmrpc-testnet.0g.ai",
      chainId: 16602,
      accounts: [process.env.PRIVATE_KEY]
    },
    "0g-mainnet": {
      url: "https://evmrpc.0g.ai",
      chainId: 16661,
      accounts: [process.env.PRIVATE_KEY]
    }
  }
};
```

Foundry 配置：

```
# foundry.toml
[profile.default]
evm_version = "cancun"

[rpc_endpoints]
0g_testnet = "https://evmrpc-testnet.0g.ai"
0g_mainnet = "https://evmrpc.0g.ai"
```

Step 3：部署合约

Hardhat 部署：

```
npx hardhat run scripts/deploy.js --network 0g-testnet
```

Foundry 部署：

```
forge create --rpc-url https://evmrpc-testnet.0g.ai \
  --private-key $PRIVATE_KEY \
  --evm-version cancun \
  src/MyToken.sol:MyToken \
  --constructor-args 1000000
```

Step 4：验证合约

Hardhat 验证：

```
npx hardhat verify DEPLOYED_CONTRACT_ADDRESS --network 0g-testnet
```

Foundry 验证：

```
forge verify-contract \
  --chain-id 16602 \
  --num-of-optimizations 200 \
  --verifier custom \
  --verifier-api-key "PLACEHOLDER" \
  --compiler-version v0.8.19 \
  <CONTRACT_ADDRESS> \
  src/MyToken.sol:MyToken \
  --verifier-url https://chainscan-galileo.0g.ai/open/api
```

故障排除

交易失败 "invalid opcode"?

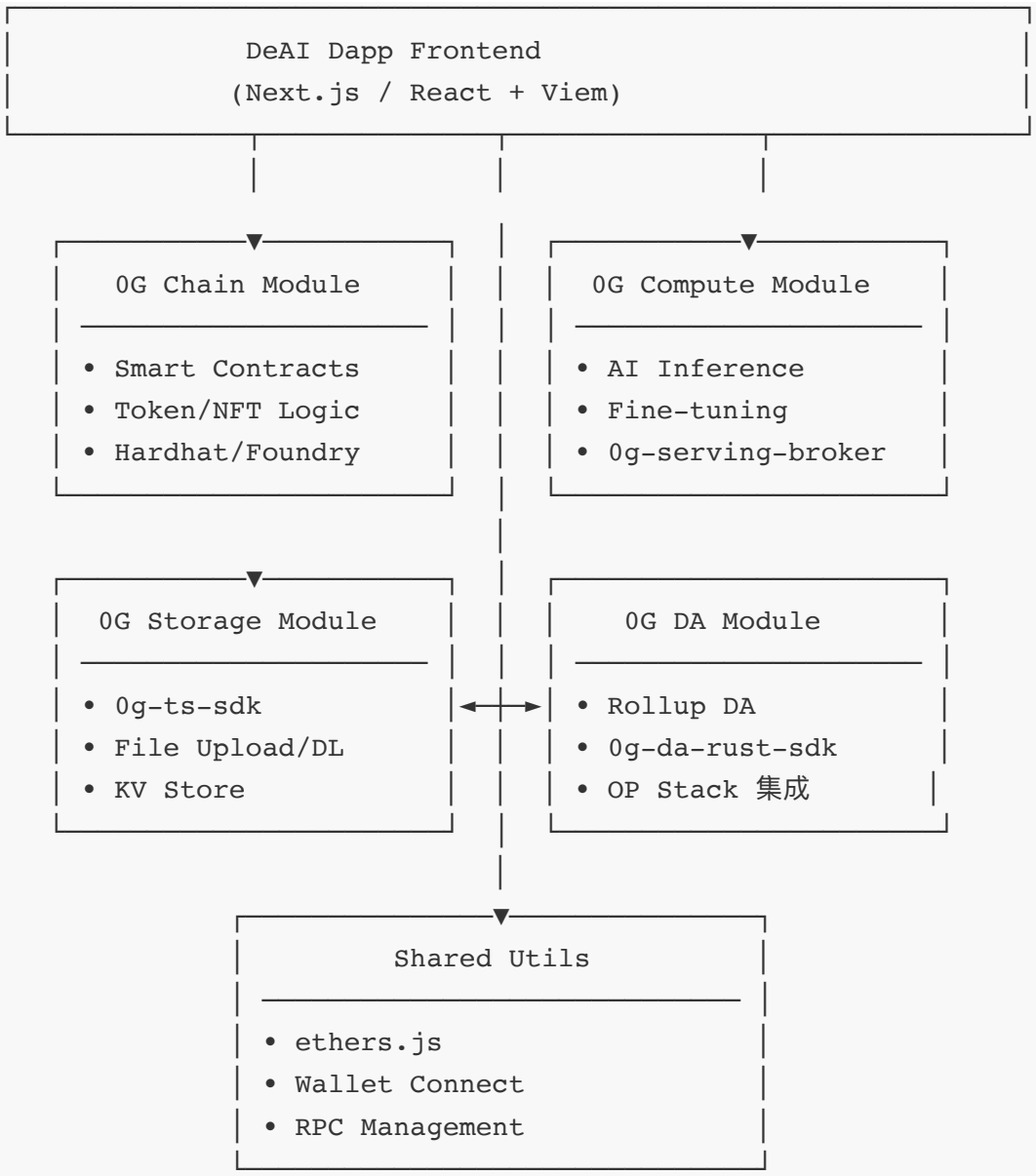
- 确保使用 `--evm-version cancun`
- 或降级 Solidity 版本（如从 0.8.26 到 0.8.19）

无法连接 RPC?

- 尝试 QuikNode 或 ThirdWeb 的替代端点

七、模块化 DeAI Dapp 架构

架构图



推荐项目结构

```
deai-dapp/
├── packages/
│   ├── chain/          # 智能合约 (Hardhat/Foundry)
│   │   ├── contracts/
│   │   ├── scripts/
│   │   └── hardhat.config.js
│   │
│   ├── storage/        # 0G Storage SDK 封装
│   │   ├── src/
│   │   │   ├── upload.ts
│   │   │   ├── download.ts
```

```
│   │   │   └── kv.ts
│   │   └── package.json
│   │
│   ├── compute/        # 0G Compute SDK 封装
│   │   ├── src/
│   │   │   ├── inference.ts
│   │   │   └── broker.ts
│   │   └── package.json
│   │
│   ├── da/             # 0G DA 集成 (可选)
│   │   └── ...
│   │
│   └── web/            # 前端应用
│       ├── app/
│       ├── components/
│       ├── hooks/
│       │   ├── useUpload.ts
│       │   ├── useDownload.ts
│       │   └── useFees.ts
│       └── package.json
│
├── .env.example
├── package.json
└── pnpm-workspace.yaml
```

各模块职责

模块	职责	主要依赖
chain	智能合约开发、部署、交互	Hardhat/Foundry、ethers
storage	文件上传/下载、KV 存储	@0glabs/0g-ts-sdk
compute	AI 推理请求、支付管理	@0glabs/0g-serving-broker
da	数据可用性、Rollup 集成	0g-da-rust-sdk
web	用户界面、钱包连接	Next.js、wagmi、viem

八、资源与参考

官方链接

资源	链接
官方文档	https://docs.0g.ai
Builder Hub	https://build.0g.ai
SDKs 列表	https://build.0g.ai/sdks
Faucet	https://faucet.0g.ai
Explorer (Testnet)	https://chainscan-galileo.0g.ai
Explorer (Mainnet)	https://chainscan.0g.ai

SDK & 工具

资源	链接
TypeScript SDK	https://github.com/0glabs/0g-ts-sdk
Go Storage Client	https://github.com/0glabs/0g-storage-client
Serving Broker	https://www.npmjs.com/package/@0glabs/0g-serving-broker
部署脚本	https://github.com/0glabs/0g-deployment-scripts

Starter Kits

模版	链接
Storage TS	https://github.com/0gfoundation/0g-storage-ts-starter-kit
Storage Web	https://github.com/0gfoundation/0g-storage-web-starter-kit
Storage Go	https://github.com/0gfoundation/0g-storage-go-starter-kit
Compute TS	https://github.com/0gfoundation/0g-compute-ts-starter-kit

0G 基础设施概览

组件	功能
0G Storage	去中心化数据存储，比 AWS 成本低 95%，支持即时检索
0G Compute	去中心化 AI 推理和训练，全球 GPU 网络，按使用付费
0G Chain	最快的模块化 AI L1 区块链，2,500+ TPS，EVM 兼容
0G DA	无限可扩展的数据可用性层，确保数据可访问和可验证
iNFTs	基于 ERC-7857 标准的智能 NFT，用于令牌化 AI Agent

快速检查清单

开始构建前，确保：

- ☐ 安装 Node.js 16+
- ☐ 安装 AI 开发工具（Cursor/Claude Code）
- ☐ 配置 0G Context 文件
- ☐ 创建 .env 并填入私钥
- ☐ 从 Faucet 获取测试代币
- ☐ 克隆合适的 Starter Kit
- ☐ 运行测试确认 SDK 连接正常

Happy Building! 🚀