OASIS TOKEN2049 CHEATSHEET

www.oasisprotocol.org

Demo starter project
github.com/oasisprotocol/demo-starter

Example projects
playground.oasis.io

Testnet Faucet
faucet.testnet.oasis.io

Oasis Docs docs.oasis.io

```
docs.oasis.io/dapp/sapphire                  # Sapphire Chain ID, RPCs
docs.oasis.io/dapp/sapphire/quickstart       # Quickstart with Hardhat
docs.oasis.io/dapp/sapphire/browser          # MetaMask browser integration
docs.oasis.io/dapp/sapphire/guide            # Complete guide
docs.oasis.io/dapp/sapphire/gasless          # Gasless txes, account abstraction
docs.oasis.io/dapp/sapphire/authentication # View-call authentication
api.docs.oasis.io/sol/sapphire-contracts   # Solidity API Docs
docs.oasis.io/dapp/opl                       # OPL: Cross-chain Toolkit
```

Oasis Wallet
wallet.oasis.io

Block Explorer
explorer.oasis.io

NPM packages

```
npm i -D @oasisprotocol/sapphire-contracts # Sapphire solidity contracts
npm i -D @oasisprotocol/sapphire-hardhat   # Hardhat integration
npm i -D @oasisprotocol/sapphire-paratime  # MetaMask, Ethers wrapper
```

sapphire-localnet Docker

```
docker run -it -p8545-8546:8545-8546 ghcr.io/oasisprotocol/sapphire-localnet
# to fund the wallet: -to 0xYOUR_ADDRESS
# or -test-mnemonic for standard Hardhat node addresses
```

@oasisprotocol/sapphire-contract cheatsheet

```
import "@oasisprotocol/sapphire-contracts/contracts/Sapphire.sol";

// Random Number Generator: Favorite number + On-chain key generation
bytes memory seed = Sapphire.randomBytes(32, "");
favoriteNumber = uint256(keccak256(abi.encodePacked(msg.sender, seed))) % 100;
Sapphire.SigningAlg alg = Sapphire.SigningAlg.Secp256k1PrehashedKeccak256;
(pk, sk) = Sapphire.generateSigningKeyPair(alg, seed); // Public/Secret key

// On-chain encryption/decryption
bytes memory encrypted = Sapphire.encrypt(sk, nonce, ”plain text”, “context”);
bytes memory decrypted = Sapphire.decrypt(sk, nonce, encrypted, ”context”);

// On-chain signing/verification
bytes memory digest = abi.encodePacked(keccak256("signed message"));
bytes memory signature = Sapphire.sign(alg, sk, digest, "context");
require( Sapphire.verify(alg, pk, digest, "context", signature) );

// On-chain TX Signing
import {EIP155Signer} from "@oasisprotocol/sapphire-contracts/contracts/EIP155Signer.sol";
bytes memory gaslessTx = EIP155Signer.sign(addr, sk,
  EIP155Signer.EthTx({ nonce: nonce, gasPrice: 100_000_000_000, gasLimit:250_000,
    to: address(this), value: 50_000_000_000_000_000_000, chainId: block.chainid,
    data: abi.encodeCall(this.myPayableFunc, abi.encode(”param1”, “param2”)),
}));
```

Oasis CLI
github.com/oasisprotocol/cli

Discord #dev-central
oasis.io/discord