

SDK Quickstart

Platform & Architecture · Published by UnyKorn LLC (Wyoming), technology provider · September 2026

DEMONSTRATION DOCUMENT — NOT AN OFFERING, NOT A BOUND CONTRACT. All entities, identifiers, reserves, serials, vaults, policies, counterparties and figures in this document are illustrative and synthetic. UnyKorn LLC is not a broker-dealer, exchange, alternative trading system, investment adviser, bank, trust company, qualified custodian, money transmitter, insurer or issuer, and does not hold client assets or funds. No entity described here is registered with the SEC, FINRA, the FCA or any other regulator. No refinery, vault operator, insurer, broker, auditor or qualified person named or implied is a counterparty to UnyKorn LLC. Nothing here is an offer to sell or a solicitation of an offer to buy any security, or investment, legal, tax or accounting advice.

DGX CLIENT SDK QUICKSTART GUIDE

Official Multi-Language Integration Libraries: TypeScript, Python, Rust & cURL

1. TypeScript / JavaScript Integration

Install via npm / yarn:

```
npm install @dgx/l1-sdk
```

Code Example:

```
import { DGXClient, OrderSide } from "@dgx/l1-sdk";

async function main() {
  // 1. Connect to institutional L1 RPC endpoint
  const client = new DGXClient({
    rpcUrl: "https://rpc.dgx.unykorn.org",
    apiKey: process.env.DGX_API_KEY
  });

  // 2. Query Allocated Bar Passport
  const bar = await client.getBarPassport("VAL-CH-994820");
  console.log(`Bar: ${bar.serial} | Purity: ${bar.fineness} | Enclave: ${bar.vaultEnclave}`);
  console.log(`Specie Policy: ${bar.insurancePolicyId} (Insured: ${!bar.isEncumbered})`);

  // 3. Submit an Atomic Delivery-versus-Payment (DvP) Order
  const dvpReceipt = await client.submitDvpOrder({
    side: OrderSide.BUY,
    pair: "DIGAU.PHYS/USD",
    quantityOz: 50.0,
    limitPriceUsd: 2650.50,
    settlementCurrency: "USDC"
  });

  console.log(`Atomic DvP Cleared: #${dvpReceipt.orderId} at Block Height #${dvpReceipt.blockHeight}`);
}

main().catch(console.error);
```

2. Python Integration

Install via pip:

```
pip install dgx-sdk
```

Code Example:

```

import os
from dgx_sdk import DGXClient, OrderSide

# Initialize institutional connection
client = DGXClient(
    rpc_url="https://rpc.dgx.unykorn.org",
    api_key=os.getenv("DGX_API_KEY")
)

# 1. Fetch Chain Telemetry & Quantum Randomness Beacon
status = client.get_chain_status()
print(f"DGX Block Height: #{status.block_height} | Validator Stake: {status.active_stake_oz} oz")

qrng = client.query_qrng_beacon(round_id=status.block_height)
print(f"QRNG Beacon: {qrng.entropy_seed[:18]}... (Hardware Vacuum Fluctuations)")

# 2. Inspect In-Ground Concession Streaming Health
concession = client.get_concession_passport("NI43-101-EUREKA-NV")
print(f"Mine: {concession.name} | P&P Reserves: {concession.proven_reserves_oz:,.0f} oz")
print(f"Coverage Ratio: {concession.coverage_ratio}x | Cash Margin: ${concession.net_margin_oz:,.2f}/oz")

# 3. Execute Atomic DvP Settlement
receipt = client.submit_dvp_order(
    side=OrderSide.BUY,
    pair="DIGAU.PHYS/USD",
    quantity_oz=100.0,
    price_usd=2650.50
)
print(f"Cleared Order: {receipt.order_id} | Settled: ${receipt.total_cash_usd:,.2f}")

```

3. Rust Integration

Add to your `Cargo.toml`:

```

[dependencies]
dgx-core = { git = "https://github.com/FTHTrading/DGX.git", package = "dgx-core" }
dgx-quantum = { git = "https://github.com/FTHTrading/DGX.git", package = "dgx-quantum" }
tokio = { version = "1.38", features = ["full"] }

```

Code Example:

```

use dgx_core::LbmaGoldBar;
use dgx_quantum::DilithiumKeypair;

#[tokio::main]
async fn main() -> Result<(), Box<dyn std::error::Error>> {
    // 1. Generate Post-Quantum Dilithium-3 Keypair
    let keypair = DilithiumKeypair::generate()?;
    println!("Dilithium Public Key (bytes): {}", keypair.public_key.len());

    // 2. Sign institutional custody transaction
    let payload = b"ALLOCATE_BAR_VAL_CH_994820_TO_SPV_TRUST";
    let signature = keypair.sign(payload)?;
    println!("Post-Quantum Signature Verified: {}", signature.verify(payload,
&keypair.public_key)?);

    Ok(())
}

```

4. Direct cURL / Shell Integration

```

# Query Allocated Bullion Passport
curl -s -X POST https://rpc.dgx.unykorn.org -H "Content-Type: application/json" -H
"Authorization: Bearer INSTITUTIONAL_API_KEY" -d '{
    "jsonrpc": "2.0",
    "id": 1,
    "method": "dgx_getBarPassport",
    "params": { "barSerial": "VAL-CH-994820" }
}' | jq .

```