EB-API-003 · DEVELOPER SERIES

SDK & API Documentation

The complete interface reference for Earth Blockchain: BaaS authentication and quotas, the standard and earth_ JSON-RPC namespaces with request/response schemas, WebSocket streams, the Anchoring, Asset-Issuance, and Event/Webhook services, client SDKs for TypeScript, Python, Go, and Java, and light-client finality verification.

JSON-RPC · WEBSOCKET · BAAS SERVICES · CLIENT SDKS · FINALITY PROOFS

Version 1.0 · **Status** Public Release · **Date** July 2026

Canonical source Earth Blockchain Whitepaper v2.0

Audience integration engineers · SDK users · platform teams

EB-API-003 · DEVELOPER SERIES · VERSION 1.0 · JULY 2026

SDK & API Documentation

INTERFACE REFERENCE · JSON-RPC · BAAS SERVICES · CLIENT SDKS

About This Reference

This document specifies every public interface of Earth Blockchain: the node JSON-RPC surface (standard EVM namespaces plus the `earth_` extensions defined normatively in EB-SPEC-001 §11), WebSocket subscriptions, the BaaS service APIs (Anchoring, Asset Issuance, Events & Webhooks, Indexing), the official client SDKs, and the light-client verification flow for consuming finality proofs. Conceptual guidance lives in EB-DEV-002; this reference is schema-level.

Conventions. Endpoints use canonical names (`https://rpc.earth.ooo` , `https://api.earth.ooo` for BaaS, `wss://ws.earth.ooo`); live values ship with genesis artifacts at earth.ooo. Quantities are decimal strings in gr (1 EARTH = 10^{18} gr) unless a field is hex-quantified per Ethereum JSON-RPC convention. All examples are JSON-RPC 2.0.

Table of Contents

1. Access Models and Authentication

PUBLIC RPC · BAAS PROJECTS · KEYS AND QUOTAS

Surface	Base URL (canonical)	Auth
Node JSON-RPC	https://rpc.earth.ooo · https://rpc.testnet.earth.ooo	None (public) or project key when served via BaaS.
BaaS REST services	https://api.earth.ooo/v1	`Authorization: Bearer <PROJECT_KEY>`; keys scoped per project and environment.
WebSocket	wss://ws.earth.ooo	Project key as query parameter or header on upgrade.

BaaS projects carry metered quotas and optional EARTH-settled billing at contracted rates (Whitepaper §7.2, §7.5). Keys MUST be treated as secrets: server-side only, rotated on personnel change, and never embedded in client bundles — browser dApps should call your backend or use public RPC for reads.

2. Standard JSON-RPC Namespaces

ETH_ · NET_ · WEB3_ — UNCHANGED SEMANTICS

All standard methods behave exactly as on the pinned EVM baseline (EB-SPEC-001 §6.1): `eth_chainId`, `eth_blockNumber`, `eth_getBalance`, `eth_call`, `eth_estimateGas`, `eth_feeHistory`, `eth_sendRawTransaction`, `eth_getTransactionReceipt`, `eth_getLogs`, and companions. Two Earth-specific notes:

- **Block tags.** The `"finalized"` and `"safe"` tags both resolve to the highest quorum-certified block — on Earth Blockchain they are the same thing. `"latest"` is the current head (typically one slot ahead of finalized).
- **Header fields.** Block objects include the PoSA fields `proposer`, `epoch`, `validatorSetHash`, and `quorumCert` (EB-SPEC-001 §4.1); decoders MUST tolerate the extra fields.

3. The earth_ Namespace: Schemas

POSA DATA · REQUEST/RESPONSE DEFINITIONS

3.1 earth_getFinalizedHead

```
→ {"jsonrpc":"2.0","id":1,"method":"earth_getFinalizedHead","params":[]}
← {"result":{"hash":"0x…","number":"0x1a2b3c","epoch":"0x5e",
             "timestamp":"0x66a1f2b0"}}
```

3.2 earth_getQuorumCert

```
→ {"method":"earth_getQuorumCert","params":["0xBLOCK_HASH"]}
← {"result":{
    "payload":"0x…",            // FP = H(blockHash‖number‖epoch‖validatorSetHash)
    "aggregateSignature":"0x…", // BLS12-381, 96 bytes
    "participation":"0x1ffb",   // n-bit bitmap over the epoch's ordered set
    "signerCount":16,
    "epoch":"0x5e"}}
```

3.3 earth_getValidatorSet

```
→ {"method":"earth_getValidatorSet","params":["0x5e"]}   // epoch (or "latest")
← {"result":{"epoch":"0x5e","validatorSetHash":"0x…","validators":[
    {"operator":"0x…","blsKey":"0x…","stakeGr":"81200000000000000000000",
     "reputation":"0.9341","tier":"enterprise","greenAttested":true}, … ]}}
```

3.4 earth_getEpochInfo

```
→ {"method":"earth_getEpochInfo","params":["0x5e"]}
← {"result":{"boundaryBlock":"0x1a2b00","seed":"0x…",
    "emissionGr":"…","baseFeesBurnedGr":"…","rewardPoolGr":"…",
    "beta":"0.30","activeSetSize":21}}
```

3.5 earth_getReputation · earth_getSlashingEvents

```
→ {"method":"earth_getReputation","params":["0xOPERATOR"]}
← {"result":{"R":"0.9341","U":"0.998","A":"0.994","F":"0.021",
             "window":"30 epochs","evidenceRefs":["0x…"]}}

→ {"method":"earth_getSlashingEvents","params":[{"fromEpoch":"0x50","toEpoch":"0x5e"}]}
← {"result":[{"id":"S1","operator":"0x…","alpha":"0.02",
              "penaltyGr":"…","evidenceHash":"0x…","epoch":"0x57"}]}
```

Field-by-field types: hashes B32 hex; keys BLS 48-byte compressed hex; reputation values fixed-point decimal strings with 4 places; monetary fields decimal gr strings. Error semantics in §8.

4. WebSocket Streams

SUBSCRIPTIONS OVER WSS://WS.EARTH.OOO

Subscription	Payload per Message
newHeads	Standard header stream (includes PoSA fields), at head cadence (~3 s).
finalizedHeads	Header stream emitted only on quorum certification — the recommended cursor for applications and indexers (EB-DEV-002 §5, §8).
logs (filter)	Standard log objects; combine with finalizedHeads or use BaaS webhooks for reorg-free delivery.
earth_epochs	One message per epoch boundary: the §3.4 object — settlement, burn, and set-rotation telemetry.

```
→ {"id":1,"method":"eth_subscribe","params":["finalizedHeads"]}
← {"method":"eth_subscription","params":{"subscription":"0xab…",
    "result":{"number":"0x1a2b3d","hash":"0x…","quorumCert":{…}}}}
```

5. BaaS Service APIs

ANCHORING · ASSET ISSUANCE · EVENTS & WEBHOOKS · INDEXING

5.1 Anchoring API

One-call notarization of a content hash — the primitive beneath EarthDoc, EarthESG, and EarthSCM (Whitepaper §7.5). The service submits, pays, and returns a finality receipt.

```
POST /v1/anchors
{ "hash":"0xSHA256_OF_CONTENT", "schema":"earthdoc/credential-v1",
  "tags":["invoice","2026-Q3"] }
← 201 { "anchorId":"anc_9f2…", "txHash":"0x…", "block":"0x1a2b3d",
        "finalized":true, "quorumCertRef":"0x…", "anchoredAt":"2026-07-12T09:14:03Z" }

GET /v1/anchors/verify?hash=0xSHA256_OF_CONTENT
← 200 { "exists":true, "anchorId":"anc_9f2…", "issuer":"did:earth:0x…",
        "revoked":false }
```

5.2 Asset Issuance API

Template-driven deployment of EBS-standard assets from the audited registry implementations (Whitepaper §7.5; EB-SPEC-001 §8.5):

```
POST /v1/assets
{ "standard":"EBS-1155", "template":"carbon-credit-v1",
  "params":{ "name":"Project Vintage 2026", "uri":"ipfs://…",
             "roles":{ "admin":"0x…", "minter":"0x…" } } }
← 201 { "assetId":"ast_51c…", "address":"0x…", "implementation":"EBS-1155@1.2.0",
        "verified":true }
```

5.3 Events & Webhooks

```
POST /v1/webhooks
{ "url":"https://erp.example.com/hooks/earth", "secret":"whsec_…",
  "filters":[{ "address":"0xASSET", "topics":["Transfer(address,address,uint256)"] }],
  "delivery":"finalized" }                      // only certified events leave BaaS

Delivery (HTTP POST, signed):
Headers: X-Earth-Signature: hmac-sha256=…  ·  X-Earth-Block: 0x1a2b3d
Body:    { "event":{…decoded log…}, "quorumCertRef":"0x…", "attempt":1 }
```

Receivers MUST verify the HMAC and SHOULD verify `quorumCertRef` for high-value actions (§7). Delivery retries with exponential backoff for 24 h; endpoints must be idempotent on `(txHash, logIndex)`.

5.4 Indexing API

```
GET /v1/index/transfers?asset=0x…&from=2026-07-01&cursor=…
GET /v1/index/holders?asset=0x…&at=finalized
GET /v1/index/exports?type=compliance-csv&range=2026-Q2   // async job → signed URL
```

6. Client SDKs

TYPESCRIPT · PYTHON · GO · JAVA

Official SDKs wrap the RPC and BaaS surfaces with typed models, finality helpers, and EBS bindings (Whitepaper §7.6). Package identifiers are published at earth.ooo/developers; the API shape is uniform:

```
// TypeScript
import { EarthClient } from '@earth-blockchain/sdk'
const c = new EarthClient({ rpc: RPC_URL, baasKey: process.env.EARTH_KEY })
const rcpt = await c.tx.sendAndFinalize({ to, value })      // resolves on quorum cert
const cred = await c.anchoring.anchor({ hash, schema })
const set  = await c.consensus.validatorSet('latest')
```

```
# Python
from earth_sdk import EarthClient
c = EarthClient(rpc=RPC_URL, baas_key=os.environ["EARTH_KEY"])
receipt = c.tx.send_and_finalize(to=addr, value_earth="1.0")
proof   = c.consensus.quorum_cert(receipt.block_hash)
assert c.consensus.verify(proof)                    # BLS + set-hash check
```

Go (`earthsdk`) and Java (`ooo.earth.sdk`) expose the same modules: `tx`, `consensus`, `anchoring`, `assets`, `events`, `ebs`. All SDKs implement §7 verification natively.

7. Light-Client Finality Verification

TRUST THE CERTIFICATE, NOT THE ENDPOINT

Any party can verify finality without trusting an RPC provider, using only genesis artifacts:

1. **Bootstrap.** Load the genesis validator set and its `validatorSetHash` from published artifacts.
2. **Follow set transitions.** For each epoch boundary, verify the new set commitment is certified by the previous set (EB-SPEC-001 §12); cache the current set.
3. **Verify a block.** Recompute `FP = H(blockHash‖number‖epoch‖validatorSetHash)`; check the aggregate BLS signature of `earth_getQuorumCert` against the bitmap-selected keys; require signerCount ≥ ⌈2n/3⌉.

Fig. 1 — Independent Finality Verification

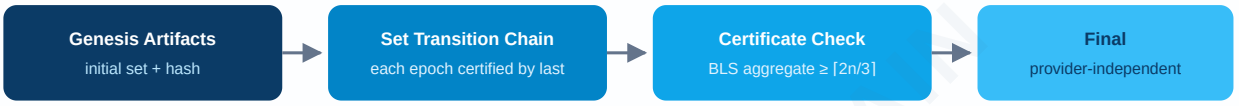

Fig. 1. The light-client flow implemented by all official SDKs — the basis for bridge verifiers and auditor tooling (EB-SPEC-001 §10, §12).

8. Errors, Rate Limits, and Versioning

UNIFORM FAILURE SEMANTICS

Code	Meaning	Client Action
-32000 / 400	Invalid params or execution revert	Fix request; decode revert data.
-32005 / 429	Quota or rate limit exceeded	Backoff per `Retry-After`; upgrade project tier.
401 / 403	Missing or unauthorized project key	Rotate key; check environment scoping.
409 (BaaS)	Idempotency conflict (duplicate anchor/issuance)	Treat as success; fetch the existing resource.
503 + retry hint	Upstream finality lag or maintenance window	Retry after hint; alarms only if sustained.

Versioning. REST paths are versioned (`/v1`); breaking changes ship as new versions with ≥ 90 days overlap. RPC changes follow protocol upgrades, which are always announced through the 48 h EarthDAO Timelock (EB-SPEC-001 §9) — subscribe to the Timelock queue as your earliest-warning channel.

References

SUITE CROSS-REFERENCES

[1] Earth Blockchain Whitepaper v2.0 — BaaS platform (§7.5), utility settlement (§7.2, §11).

[2] EB-SPEC-001 Technical Protocol Specification — normative RPC list (§11), certificates (§5.5), system contracts (§8).

[3] EB-DEV-002 Developer Documentation — conceptual guidance and finality patterns.

[4] Ethereum JSON-RPC and EIP-1559/155 conventions (eips.ethereum.org) — informative baseline for §2.

EB-API-003 v1.0 (July 2026). Endpoint names are canonical conventions; live values, package identifiers, and full machine-readable schemas (OpenAPI/JSON) are published at earth.ooo/developers. Examples are illustrative. This document does not constitute financial, investment, tax, or legal advice.