Universal Manifest v0.5 — Cookbook

Worked Examples · Informative

v0.5.0. Published as tag v0.5.0 of UniversalManifest/universal-manifest, with the same wire files as release candidate 1 (v0.5.0-rc.1). It is a mechanism release, not a completed implementation or conformance qualification. Immutable versions in UniversalManifest/universal-manifest are normative; main is the working version. Earlier versions were reviewed in the OMA3 Technical Working Group, and publication asserts no OMA3 ratification.

8 November 2026

This document: /spec/v0.5/cookbook/

Document set: Universal Manifest v0.5.0 · What changed in v0.5.0

Artifacts: JSON Schema · JSON-LD context

History: https://universalmanifest.net/spec/ (version index and changelog)

Author: Grig Bilham — inventor, architect, and author of the Universal Manifest.

Copyright © 2026 SumSet Tech LLC and the Universal Manifest contributors. This specification is available under the Apache License, Version 2.0. It is free and remains free. Earlier published revisions retain their original licence notices.

---

v0.5.0. Published as tag `v0.5.0` of `UniversalManifest/universal-manifest`, with the same wire files as release candidate 1 (`v0.5.0-rc.1`). Wire version `0.5`; namespace `https://universalmanifest.net/ns/v0.5`; reading routes `/spec/v0.5/`. It is a mechanism release, not a completed implementation or conformance qualification. Feature-specific PREVIEW and conditional requirements remain in force.

This Cookbook collects every worked example from the Universal Manifest v0.5 document set in one place, organized by scenario and **tagged by the conformance tier it exercises**. It is **informative**: examples illustrate the normative documents but do not themselves carry requirements. Where an example and a normative document differ, the normative document governs. All cryptographic values are illustrative placeholders. The normative documents keep at most one anchoring example per concept inline; everything else lives here.

Where each example is specified: Base · EXT-T1 binding · EXT-T2 crypto · EXT-T3 ceremony · EXT-OPT optional features. New readers should start with the Primer.

Universal Manifest v0.5 — Cookbook

Examples are informative candidate shapes, not freshly verified v0.5 signatures or conformance vectors. Changing a version or context changes signing bytes: produce a new signature for every new v0.5 document; never relabel an existing signed v0.4 document. Any abbreviated signature/proof material below remains illustrative.

The examples build up in a breadth→depth progression: the minimal envelope first, then each structural member, then signing and encryption, then the lifecycle output (receipts), then the higher-assurance tiers, then a complete end-to-end manifest-and-receipt pair. The **Tier** tag on each example is the assurance level it demonstrates.

1. The minimal envelope

Example 1 — Minimal Universal Manifest payload — *Tier 0*

A manifest carrying only the anchor, identity, and lifespan members plus a signature. This is the smallest conformant manifest. Specified in Base §1.1.

```
{
  "@context": ["https://universalmanifest.net/ns/v0.5"],
  "@id": "urn:uuid:123e4567-e89b-12d3-a456-426614174000",
  "@type": ["um:Manifest"],
  "manifestVersion": "0.5",
  "subject": "did:example:123",
  "issuedAt": "2026-03-01T00:00:00Z",
  "expiresAt": "2026-03-02T00:00:00Z",
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkExample#keys-1",
    "publicKeySpkiB64": "MCowBQYDK2VwAyEA...",
    "created": "2026-03-01T00:00:00Z",
    "value": "base64url-encoded-signature-bytes"
  }
}
```

2. Structural members

These show the payload arrays an evaluator discovers inside the envelope: claims, consents, devices, the session actor, and the signature object.

Example 2 — Base claim object — *Tier 0 → Tier 1*

Two claims: a self-asserted membership claim (Tier 0) and a cross-DID binding claim offered for Tier-1 attester evaluation. Specified in Base §1.4.3.

```
{
  "claims": [
    {
      "@type": "membership.organization",
      "issuer": "did:web:example-org.com",
      "subject": "did:key:z6MkAlice",
      "issuedAt": "2026-03-01T00:00:00Z",
      "expiresAt": "2027-03-01T00:00:00Z"
    },
    {
      "@type": "identity.crossDidBinding",
      "issuer": "did:web:verify.example",
      "boundDids": ["did:key:z6MkAlice", "did:plc:alice-bsky"],
      "attester": "did:web:verify.example",
      "attestationMethod": "AT Protocol handle resolution",
      "attestedAt": "2026-03-15T10:30:00Z",
      "claimProof": "https://verify.example/attestations/abc123"
    }
  ]
}
```

Example 3 — Consent entry — *Tier 0*

Two consent entries governing two facets, with distinct scope, purpose, validity windows, and a condition. Specified in Base §1.4.4.

```
{
  "consents": [
    {
      "@id": "urn:uuid:consent-public-profile-001",
      "@type": "um:Consent",
      "facetRef": "urn:uuid:facet-public-profile-001",
      "scope": ["read", "display", "cache"],
      "purpose": "session-personalization",
      "grantedAt": "2026-04-01T09:00:00Z",
      "expiresAt": "2026-04-01T18:00:00Z"
    },
    {
      "@id": "urn:uuid:consent-health-data-002",
      "@type": "um:Consent",
      "facetRef": "urn:uuid:facet-health-data-002",
      "scope": ["read"],
      "purpose": "age-verification",
      "grantor": "did:key:z6MkAlice",
      "grantedAt": "2026-04-01T09:00:00Z",
      "expiresAt": "2026-04-01T12:00:00Z",
      "conditions": ["no-third-party-sharing"]
    }
  ]
}
```

Example 4 — Derived-variant sensor consent with purpose binding — *Tier 0* — PREVIEW

A spatial-computing consent that grants only region-of-interest-derived gaze data, bound to a DPV purpose. The sensor key is a consent-`scope` operation token (O9.1); no dedicated `sensorConsent` member exists. Specified in EXT-OPT §O9.1.

```
{
  "consents": [
    {
```

```
      "@id": "urn:uuid:consent-gaze-roi-001",
      "@type": "um:Consent",
      "facetRef": "urn:uuid:facet-eye-roi-001",
      "scope": ["sensor.eye.gaze.roi-derived"],
      "purpose": "dpv:UserInterfacePersonalisation",
      "grantedAt": "2026-06-09T09:00:00Z",
      "expiresAt": "2026-06-09T18:00:00Z"
    }
  ]
}
```

Example 5 — Device entry with both components — *Tier 0* — PREVIEW

An XR headset entry carrying both a manufacturer-signed `deviceAttestation` and a session-scoped `deviceCapability`. Specified in EXT-OPT §O9.2.

```
{
  "devices": [
    {
      "@id": "urn:uuid:device-headset-1",
      "@type": "um:Device",
      "deviceAttestation": {
        "deviceClass": "xr-headset",
        "modelHash": "sha256:9f2c...",
        "manufacturer": "did:web:headset-vendor.example",
        "attestedAt": "2026-06-01T00:00:00Z",
        "attestation": { "@type": "VerifiableCredential" }
      },
      "deviceCapability": {
        "sensors": ["eye-tracking", "hand-tracking", "depth"],
        "openxrExtensions": ["XR_EXT_eye_gaze_interaction"],
        "positioningTier": "6dof",
        "privacyModes": ["roi-derived-only"],
        "sessionSigningKey": "z6MkpTHR8VNsBxYAAWHut2Geadd9jSwuBV8xRoAnwWsdvktH"
      }
    }
  ]
}
```

Example 6 — actorState with an agent executor — *Tier 0* — PREVIEW

A session operated by a delegated agent on the principal's behalf, referencing the delegation pointer that authorizes it. Specified in EXT-OPT §O9.3.

```
{
  "subject": "did:key:z6MkAlice",
  "actorState": {
    "principal": "did:key:z6MkAlice",
    "executor": {
      "type": "agent",
      "delegateId": "did:key:z6MkAgentBot",
      "delegationRef": "urn:uuid:pointer-delegation-1",
      "lastVerifiedAt": "2026-06-09T09:00:00Z"
    }
  }
}
```

3. Signing and encryption

Example 7 — Signature Profile A object — *Tier 0*

The baseline JCS + Ed25519 signature object. Specified in Base §1.6.2.

```
{
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkAlice#keys-1",
    "publicKeySpkiB64": "MCowBQYDK2VwAyEA...",
    "created": "2026-04-01T10:00:00Z",
    "value": "base64url-encoded-signature-bytes"
  }
}
```

Example 8 — Plaintext facet with all fields — *Tier 0*

A facet with a display name, an authoritative `ref`, and an embedded entity. Specified in Base §2.1.

```
{
  "@id": "urn:uuid:facet-public-profile",
  "@type": "um:Facet",
  "name": "publicProfile",
  "ref": "https://example.com/profiles/alice",
  "entity": {
    "@id": "urn:uuid:entity-alice-profile",
    "@type": ["um:Entity", "um:IdentityProfile"],
    "displayName": "Alice Example",
    "avatarUrl": "https://example.com/avatars/alice.png"
  }
}
```

Example 9 — Plaintext entity — *Tier 0*

The minimal entity: `@id`, `@type`, and profile-extensible fields. Specified in Base §2.2.

```
{
  "@id": "urn:uuid:entity-alice-profile",
  "@type": ["um:Entity", "um:IdentityProfile"],
  "displayName": "Alice Example",
  "avatarUrl": "https://example.com/avatars/alice.png",
  "preferredLanguage": "en"
}
```

Example 10 — Facet with JWE inline encryption — *Tier 0*

A sealed-entry facet: the payload is a JWE readable only by the named recipient; every other evaluator records it as present-but-sealed. Specified in Base §2.3.

```
{
  "@id": "urn:uuid:facet-medical-records",
  "@type": "um:Facet",
  "name": "privateHealth",
```

```
  "encryptionProfile": "jwe-inline-v1",
  "entity": {
    "protected": "eyJhbGciOiJFQ0RILUVTK0EyNTZLVyIsImVuYyI6IkEyNTZHQ00ifQ",
    "recipients": [
      {
        "header": { "kid": "did:example:clinic#key-agree-1" },
        "encrypted_key": "base64url-encrypted-key-for-k1"
      }
    ],
    "iv": "base64url-iv-1",
    "ciphertext": "base64url-ciphertext-1",
    "tag": "base64url-tag-1"
  }
}
```

4. Receipts (the lifecycle output)

Example 11 — Structured receipt — *Tier 0*

A receipt for a partially-accepted manifest: one facet processed, one sealed (no decryption key). Specified in Base §3.3.1.

```
{
  "@type": "um:Receipt",
  "manifestId": "urn:uuid:123e4567-e89b-12d3-a456-426614174000",
  "outcome": "accepted-partial",
  "signatureCheck": "valid",
  "freshnessCheck": "fresh",
  "revocationStatus": "unchecked",
  "facetStatuses": [
    { "facetId": "urn:uuid:facet-public-profile", "name": "publicProfile", "sta
    { "facetId": "urn:uuid:facet-private-health", "name": "privateHealth", "sta
  ],
  "processedAt": "2026-05-19T12:00:00Z"
}
```

Example 12 — Chained receipt manifest with typed events — *Tier 0* — PREVIEW

A receipt promoted to a first-class manifest class: hash-chained (`seq`/`prevHash`), carrying typed events, signed with a session-scoped key. Specified in EXT-OPT §O3.

```
{
  "@context": ["https://universalmanifest.net/ns/v0.5"],
  "@id": "urn:uuid:receipt-0002",
  "@type": ["um:Manifest", "um:Receipt"],
  "manifestVersion": "0.5",
  "subject": "did:key:z6MkVenueEdge",
  "issuedAt": "2026-06-09T12:00:05Z",
  "expiresAt": "2026-06-10T12:00:05Z",
  "manifestId": "urn:uuid:123e4567-e89b-12d3-a456-426614174000",
  "outcome": "accepted",
  "signatureCheck": "valid",
  "freshnessCheck": "fresh",
  "facetStatuses": [
    { "facetId": "urn:uuid:facet-public-profile", "status": "processed" }
  ],
  "chainId": "urn:uuid:chain-7c1f",
  "seq": 2,
  "prevHash": "uEiD9f2c...",
  "events": [
    { "eventType": "manifest-verified", "at": "2026-06-09T12:00:05Z" },
    { "eventType": "facet-processed", "at": "2026-06-09T12:00:05Z", "subjectRef
  ],
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkSessionEphemeral#keys-1",
    "value": "base64url-encoded-signature-bytes"
  }
}
```

Example 13 — statusRef response — *Tier 0* (revocation-aware)

The JSON a status endpoint returns when an evaluator resolves `signature.statusRef`. Specified

in EXT-OPT §O2.2.

```
{
  "manifestId": "urn:uuid:123e4567-e89b-12d3-a456-426614174000",
  "status": "active",
  "updatedAt": "2026-05-22T10:00:00Z",
  "cursor": "v3",
  "nextCheck": "PT1H"
}
```

5. Sessions

Example 14 — Bilateral session object — *Tier 0* — PREVIEW

A two-party session in the `manifests-exchanged` state, carrying the shared `exchangeId` both parties echo in their receipts. Specified in EXT-OPT §O4.1.

```
{
  "@type": "um:BilateralSession",
  "sessionId": "urn:uuid:session-abc-123",
  "exchangeId": "urn:uuid:exchange-def-456",
  "participants": [
    { "did": "did:key:z6MkAlice", "role": "initiator" },
    { "did": "did:key:z6MkVenue", "role": "responder" }
  ],
  "initiatedAt": "2026-05-22T10:00:00Z",
  "expiresAt": "2026-05-22T10:05:00Z",
  "state": "manifests-exchanged"
}
```

6. Trust tiers (climbing the ladder)

Example 15 — Cross-DID binding claim — *Tier 1*

An attester-asserted binding of two DIDs, evaluated against the evaluator's attester trust list. Specified in Base §6.4.4.

```
{
  "@type": "identity.crossDidBinding",
  "issuer": "did:web:verify.example",
  "boundDids": ["did:key:z6MkAlice", "did:plc:alice-bsky"],
  "attester": "did:web:verify.example",
  "attestationMethod": "AT Protocol handle resolution",
  "attestedAt": "2026-03-15T10:30:00Z",
  "expiresAt": "2026-06-15T10:30:00Z"
}
```

Example 16 — Manifest with requiredTrustTier — *Tier 1 floor, Tier 2 on a claim*

A manifest-level floor of Tier 1, with one claim raising its own floor to Tier 2. Specified in Base §6.4.5.

```
{
  "requiredTrustTier": 1,
  "claims": [
    {
      "@type": "personhood.worldId.verification",
      "issuer": "did:web:worldcoin.org",
      "requiredTrustTier": 2
    }
  ]
}
```

Example 17 — Tier 2 cross-DID binding with linked-secret proof — *Tier 2* — PREVIEW

A cryptographic cross-DID binding via a BBS+ linked-secret proof (Profile 2A), carried in `bindingProof`. Specified in EXT-T2 §T2.1.1.

```
{
  "@type": "identity.crossDidBinding",
  "issuer": "did:key:z6MkAlice",
  "boundDids": ["did:key:z6MkAlice", "did:pkh:eip155:1:0xabc"],
  "requiredTrustTier": 2,
  "bindingProof": {
    "type": "ZkLinkedSecretProof",
    "cryptosuite": "bbs-2023",
    "proofPurpose": "authentication",
    "proofValue": "u2V0BhV...",
    "publicInputs": {
      "commitmentA": "z3tC...",
      "commitmentB": "z7dF..."
    }
  }
}
```

Example 18 — Tier 3 ceremony proof — *Tier 3* — PREVIEW

A multi-party 3-of-5 threshold attestation, carried in `ceremonyProof`. Specified in EXT-T3 §T3.1.

```
{
  "@type": "identity.crossDidBinding",
  "issuer": "did:key:z6MkAlice",
  "boundDids": ["did:key:z6MkAlice", "did:pkh:eip155:1:0xabc"],
  "requiredTrustTier": 3,
  "ceremonyProof": {
    "type": "ThresholdAttestationProof",
    "threshold": "3-of-5",
    "attesters": [
      "did:web:notary-a.example",
      "did:web:notary-b.example",
      "did:web:notary-c.example"
    ],
```

```
    "ceremonyId": "urn:uuid:ceremony-2026-05-30-abc",
    "aggregateProof": "z..."
  }
}
```

7. Delegation and migration

Example 19 — Agent delegation pointer — *Tier 0*

A delegation granting an AI agent two bounded scopes for a one-hour window. Specified in Base §6.5.1.

```
{
  "@type": "um:agentDelegation",
  "delegateType": "ai-agent",
  "delegateId": "did:key:z6MkAgentBot",
  "delegatedBy": "did:key:z6MkAlice",
  "delegatedAt": "2026-04-01T10:00:00Z",
  "expiresAt": "2026-04-01T11:00:00Z",
  "scope": ["spatial.session", "social.messaging"]
}
```

Example 20 — Unbound-claims warning — *Tier 0* (v0.3 compatibility)

The warning a v0.5 evaluator emits when it processes claims that lack holder binding — they are accepted only at Tier 0. Specified in EXT-T1 §T1.6.4.

```
{
  "warnings": [
    {
      "code": "um:reason:trust:unbound-claims",
      "message": "Claims urn:claim:poh and urn:claim:avatar-control lack holder
    }
  ]
}
```

Example 21 — Dual-signature manifest — *Tier 0* (post-quantum migration) — PREVIEW

A manifest carrying both a classical Ed25519 signature and a post-quantum `postQuantumSignature` during the migration window. Specified in EXT-T2 §T2.3.3.

```
{
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkAlice#keys-1",
    "value": "base64url-ed25519-signature"
  },
  "postQuantumSignature": {
    "algorithm": "ML-DSA-65",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkAlice#keys-pq-1",
    "value": "base64url-ml-dsa-signature"
  }
}
```

8. End-to-end: a complete manifest and its receipt

This pair shows one complete Universal Manifest carrying a facet, a claim, a consent, and a pointer — signed under Signature Profile A and answering a verifier challenge with a presentation proof — followed by the receipt a conformant evaluator produces. The pair doubles as a conformance fixture. Specified in the Base front matter; relocated here from the monolith's Appendix C.

Example 22 — Complete signed manifest — *Tier 1*

```
{
  "@context": ["https://universalmanifest.net/ns/v0.5"],
  "@id": "urn:uuid:6f5b2c40-9d1e-4a8e-b2c1-0a1b2c3d4e5f",
  "@type": ["um:Manifest"],
  "manifestVersion": "0.5",
  "subject": "did:key:z6MkAlice",
  "issuedAt": "2026-06-09T12:00:00Z",
```

```
  "expiresAt": "2026-06-09T20:00:00Z",
  "facets": [
    {
      "@id": "urn:uuid:facet-public-profile",
      "@type": "um:Facet",
      "name": "Public profile",
      "entity": {
        "@id": "urn:uuid:entity-alice-profile",
        "@type": ["um:Entity", "ProfileCard"],
        "displayName": "Alice"
      }
    }
  ],
  "claims": [
    {
      "@id": "urn:uuid:claim-over-18",
      "@type": "ageOver",
      "issuer": "did:web:issuer.example",
      "value": 18,
      "requiredTrustTier": 1,
      "claimProof": { "@type": "VerifiablePresentation", "proofValue": "eyJ...k
      "holderBinding": { "mode": "sd-jwt-kb", "cnfThumbprint": "NzbLsXh8..." }
    }
  ],
  "consents": [
    {
      "@id": "urn:uuid:consent-public-profile-001",
      "@type": "um:Consent",
      "facetRef": "urn:uuid:facet-public-profile",
      "scope": ["read", "display"],
      "purpose": "session-personalization",
      "grantedAt": "2026-06-09T12:00:00Z",
      "expiresAt": "2026-06-09T20:00:00Z"
    }
  ],
  "pointers": [
    {
      "@id": "urn:uuid:pointer-avatar-1",
      "@type": "mediaRef",
      "target": "https://cdn.example/alice/avatar.glb",
      "createdAt": "2026-06-09T12:00:00Z"
    }
  ],
```

```
  "presentationProof": {
    "proofType": "sd-jwt-kb",
    "challenge": "nonce-venue-8f2a",
    "audience": "did:key:z6MkVenueEdge",
    "created": "2026-06-09T12:00:00Z",
    "proofValue": "base64url-kb-jwt-proof"
  },
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkAlice#keys-1",
    "publicKeySpkiB64": "MCowBQYDK2VwAyEA...",
    "created": "2026-06-09T12:00:00Z",
    "value": "base64url-encoded-signature-bytes"
  }
}
```

Example 23 — Receipt produced by a conformant evaluation of Example 22 — *Tier 1*

```
{
  "@context": ["https://universalmanifest.net/ns/v0.5"],
  "@id": "urn:uuid:receipt-7a2e",
  "@type": ["um:Manifest", "um:Receipt"],
  "manifestVersion": "0.5",
  "subject": "did:key:z6MkVenueEdge",
  "issuedAt": "2026-06-09T12:00:01Z",
  "expiresAt": "2026-06-10T12:00:01Z",
  "manifestId": "urn:uuid:6f5b2c40-9d1e-4a8e-b2c1-0a1b2c3d4e5f",
  "outcome": "accepted",
  "signatureCheck": "valid",
  "freshnessCheck": "fresh",
  "keyRefResolution": "resolved",
  "revocationStatus": "unchecked",
  "effectiveTrustTier": 1,
  "holderBindingStatus": "verified",
  "presentationProofStatus": "verified",
  "facetStatuses": [
    { "facetId": "urn:uuid:facet-public-profile", "status": "processed", "name"
  ],
```

```
  "consentStatuses": [
    { "facetId": "urn:uuid:facet-public-profile", "consentRef": "urn:uuid:conse
  ],
  "claimStatuses": [
    { "claimRef": "urn:uuid:claim-over-18", "status": "verified", "tier": 1 }
  ],
  "processedAt": "2026-06-09T12:00:01Z",
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkVenueEdge#keys-1",
    "created": "2026-06-09T12:00:01Z",
    "value": "base64url-encoded-signature-bytes"
  }
}
```


The receipt records a Tier 1 outcome: the claim carried a verified `holderBinding`, the manifest signature verified against a resolved `keyRef`, the single facet was processed under a valid consent whose `scope` and `purpose` covered the evaluator's use, and the verifier-issued challenge was answered by a valid `presentationProof` (so `presentationProofStatus` is `"verified"`). The manifest carries no `signature.statusRef`, so revocation status is recorded as `"unchecked"`.

9. Locked-tier portable unlock

Example 24 — Locked personal-data facet: hardware-bound-preferred unlock gating an AI delegate — *Tier 1* — PREVIEW

A locked health facet conforming to the **locked-tier portable-unlock profile** (`um:profile:trust:locked-tier-portable-unlock` — EXT-T1 §T1.3.3). The facet carries both the locked liveness floor (`requiredLiveness: {minFreshness: "live", userVerified: true}`) and the new assurance-class floor (`requiredAssuranceClass: "hardware-bound"`). An unlock-window consent opens it for a single time-boxed session — with session-policy fields (EXT-OPT §O4.5) — so a delegated agent can read it within the window; the floors are never lowered by

the window, and a below-floor gesture is refused with the receipt facet status `"assuranceInsufficient"`.


```
{
  "@context": ["https://universalmanifest.net/ns/v0.5"],
  "@id": "urn:uuid:9c4a17e0-7b21-4e2a-9c33-0d2e4f6a8b10",
  "@type": ["um:Manifest"],
  "manifestVersion": "0.5",
  "subject": "did:key:z6MkAlice",
  "issuedAt": "2026-06-19T09:00:00Z",
  "expiresAt": "2026-06-19T21:00:00Z",
  "facets": [
    {
      "@id": "urn:uuid:facet-health-records",
      "@type": "um:Facet",
      "name": "Health records (locked)",
      "requiredLiveness": { "minFreshness": "live", "userVerified": true },
      "requiredAssuranceClass": "hardware-bound"
    }
  ],
  "consents": [
    {
      "@id": "urn:uuid:consent-unlock-health-001",
      "@type": "um:Consent",
      "facetRef": "urn:uuid:facet-health-records",
      "scope": ["read", "unlock.window"],
      "purpose": "Open the locked health facet for an authenticated agent sessi
      "grantedAt": "2026-06-19T09:00:00Z",
      "expiresAt": "2026-06-19T09:30:00Z",
      "unlockWindowFacets": ["urn:uuid:facet-health-records"],
      "idleLifetime": "PT10M",
      "absoluteLifetime": "PT30M",
      "lockOnSleep": true
    }
  ],
  "livenessAttestation": {
    "proofType": "webauthn-uv",
    "attestedAt": "2026-06-19T09:00:00Z",
    "validUntil": "2026-06-19T09:01:00Z",
    "userVerified": true,
    "proofValue": "base64url-webauthn-uv-assertion"
  },
```

```
  "signature": {
    "algorithm": "Ed25519",
    "canonicalization": "JCS-RFC8785",
    "keyRef": "did:key:z6MkAlice#keys-1",
    "publicKeySpkiB64": "MCowBQYDK2VwAyEA...",
    "created": "2026-06-19T09:00:00Z",
    "value": "base64url-encoded-signature-bytes"
  }
}
```

A conformant evaluator that opens the window with a **device-bound** (hardware-bound) authenticator records, on the facet status, `"processed"` plus an `assuranceStatus` echo of `{ "assertedClass": "hardware-bound", "met": true }` (Base §3.3.1.1) and a `session-unlocked` event (EXT-OPT §O3.2). The **same** manifest presented with only a software-class unlock is refused: the facet status is `"assuranceInsufficient"` (content never released), with a `session-unlock-refused` event carrying `um:reason:trust:assurance-insufficient`. The assurance floor and the liveness floor compose; the unlock window lowers neither.

✧ ✧ ✧

Informative companion to the Base specification. Examples illustrate; the normative documents bind.