did:phoenix

Method Specification

Version: *1.0*
Status: *Implementors*
Published: *2026-05-16*
Latest version: *https://phoenixkey.me/did-method/v1*
Authors: *GreenSun Tech Corporation — contact@greensun.tech — https://greensun.tech*
Repository: *https://github.com/PhoenixKeyDID*
Conforms to: *W3C Decentralized Identifiers (DIDs) v1.0*

Abstract

The did:phoenix DID method anchors decentralized identifiers on the **Cardano blockchain** (Layer 1) using the PhoenixKey protocol. It supports ten entity types — human, organization, device, machine, asset, bot, AI agent, digital service, context, and virtual character — with hardware-backed key security, tiered recovery without seed phrases, and optional cross-chain identity linking.

All identities created under this method are self-sovereign: no central registry controls them. Authority is enforced by on-chain smart contracts (Plutus V3) and device hardware security modules.

1. Method Name

The method name that identifies this DID method is: **phoenix**

A DID that uses this method MUST begin with the prefix did:phoenix:. The prefix is case-insensitive for matching; canonical form is lowercase.

2. Method-Specific Identifier

```
did-phoenix       = "did:phoenix:" slot-component ":" hash-component
slot-component    = 1*( ALPHA / DIGIT )   ; base32-encoded Cardano slot number, no padding
hash-component    = 64HEXDIG              ; 32-byte hash, lowercase hex
```

Construction:

```
random_256 ← SecureRandom(256)
hash = Hash( encode(entity_type) || (owner_did ?? "root") || encode(slot) || random_256 )
did  = "did:phoenix:" || base32_nopad(slot) || ":" || lower_hex(hash)
```

Example:

did:phoenix:aaaaaaq:3d7f9a1b2c4e5678901234567890123456789012345678901234567890abcdef1234567890abcdef1234

The slot number in the identifier enables temporal ordering and auditing without a central registry. The 256-bit random nonce makes DIDs unpredictable before creation.

3. DID Document

A did:phoenix DID Document is derived from the on-chain TAAD UTxO datum at resolution time. All documents MUST include:

```
{
 "@context": [
  "https://www.w3.org/ns/did/v1",
  "https://w3id.org/security/suites/ed25519-2020/v1",
  "https://phoenixkey.me/context/v1"
 ]
}
```

PersonDID Example

```
{
 "@context": [
  "https://www.w3.org/ns/did/v1",
  "https://w3id.org/security/suites/ed25519-2020/v1",
  "https://phoenixkey.me/context/v1"
 ],
 "id": "did:phoenix:SLOT:HASH",
 "didType": "Person",
 "controller": "did:phoenix:SLOT:HASH",
 "verificationMethod": [
  {
   "id": "did:phoenix:SLOT:HASH#hw-key-current",
   "type": "Ed25519VerificationKey2020",
   "controller": "did:phoenix:SLOT:HASH",
   "publicKeyMultibase": "z...",
   "securityLevel": "BiometricHardware"
  },
  {
   "id": "did:phoenix:SLOT:HASH#controller-key-current",
   "type": "Ed25519VerificationKey2020",
   "controller": "did:phoenix:SLOT:HASH",
   "publicKeyMultibase": "z..."
  }
 ],
```

```
  "authentication":       ["did:phoenix:SLOT:HASH#hw-key-current"],
  "assertionMethod":      ["did:phoenix:SLOT:HASH#hw-key-current"],
  "capabilityInvocation": ["did:phoenix:SLOT:HASH#controller-key-current"],
  "capabilityDelegation": ["did:phoenix:SLOT:HASH#controller-key-current"],
  "service": [
    {
      "id": "did:phoenix:SLOT:HASH#taad",
      "type": "PhoenixKeyTAAD",
      "serviceEndpoint": {
        "network": "cardano:mainnet",
        "utxoTxHash": "...",
        "utxoIndex": 0
      }
    }
  ]
}
```

Non-human DIDs (OrgDID, DeviceDID, etc.) use the same structure with the owner's DID as controller and method-appropriate verification methods. Full examples: https://phoenixkey.me/did-method/v1/examples

4. CRUD Operations

4.1 Create

1. Generate a hardware-bound Ed25519 key pair inside the device Secure Enclave and derive a recoverable controller key from on-device key material.
2. Compute the DID using the construction formula in §2.
3. Submit a Cardano transaction creating a TAAD UTxO at Script(TAAD_Validator) with a datum containing the DID, controller key hash, hardware public key, and initial state Active.
4. DID is live after 3 block confirmations on Cardano mainnet.

Error conditions:

Error	Cause
duplicateController	Controller key already controls an active TAAD UTxO
insufficientFunds	Wallet has insufficient ADA for TAAD UTxO minimum ADA
invalidOwner	Specified owner DID is not active

4.2 Read (Resolve)

1. Parse did:phoenix:SLOT:HASH to extract slot (base32 decode) and hash (hex decode).
2. Query Cardano for the UTxO at Script(TAAD_Validator) whose datum matches the DID.
3. **MUST** traverse the full ancestor ownership chain — if any ancestor is revoked or suspended, return "deactivated": true.
4. Construct and return the DID Document from the datum fields.

Resolution Metadata:

```
{
 "contentType": "application/did+ld+json",
 "resolvedAt": "<ISO8601 timestamp>",
 "cardanoSlot": 12345678,
 "cardanoBlockHash": "abc123...",
 "taadSequence": 3
}
```

Error conditions:

Error	Cause
notFound	No TAAD UTxO found for this DID
deactivated	DID or an ancestor is revoked or suspended
invalidDID	Malformed DID string
networkError	Cardano node unreachable

4.3 Update

Key rotation: Initiated via the TAAD tiered recovery protocol. A new hardware key is committed on-chain after a mandatory waiting period. The DID string does not change; verification methods update to reflect the new keys.

Self-initiated rotation (device in hand) takes effect **immediately**. Guardian-assisted recovery (device lost) requires a waiting period during which the legitimate owner may cancel.

Other updates (guardian config, service endpoints): require a signature from the current hardware key. Recorded on-chain by spending and reissuing the TAAD UTxO with an incremented sequence number.

4.4 Deactivate

Submit a Cardano transaction setting revoked_slot in the TAAD datum. Requires current hardware key signature. Deactivation is **permanent and irreversible**. All descendant DIDs are implicitly deactivated — resolvers MUST check ancestry on every resolution.

5. DID URL Syntax

did-phoenix-url = did-phoenix ["/" path] ["?" query] ["#" fragment]

Standard fragment identifiers:

Fragment	Resolves to
#hw-key-current	Current hardware signing key
#controller-key-current	Current controller key
#multisig	OrgDID threshold signature method
#taad	TAAD UTxO service endpoint

Query parameters:

Parameter	Description
versionId=<seq>	Resolve at a specific sequence number
versionTime=<slot>	Resolve as of a specific Cardano slot

6. Resolver Requirements

A conformant did:phoenix resolver MUST:

1. Connect to a Cardano node or indexer service to read UTxO state. Recommended: self-hosted cardano-node with Ogmios and Kupo, or a compatible Cardano SPO relay endpoint.
2. Traverse the full ancestor chain for every resolution. Resolvers MUST NOT cache ancestry status without a slot-bounded TTL.
3. Return deactivated: true if the DID or any ancestor is revoked or suspended.
4. Verify linkedDIDProofs by resolving the external DID using a conformant resolver for that DID method.
5. Return resolution metadata including the Cardano slot at which the TAAD UTxO was read.
6. Support the DIF Universal Resolver driver interface (SHOULD).

7. Security Considerations

Hardware-bound keys. The hardware signing key is generated in and permanently bound to a device Secure Enclave (Apple SEP, Android StrongBox, or ARM TrustZone). It cannot be exported. Every signing operation requires biometric authentication.

Replay prevention. Every TAAD state transition increments a monotonic sequence number enforced by the on-chain validator. Transactions with seq ≤ on-chain seq are rejected unconditionally.

Guardian suspension. Guardians may suspend a DID, but cancel and recovery operations bypass the suspension check — preventing permanent lockout of the legitimate owner.

Ancestor deactivation. Resolvers MUST walk the full ownership ancestry. A child DID cannot be considered active if any ancestor is deactivated.

Recovery integrity. Key rotation initiated while the owner holds their device takes effect immediately. Rotation initiated after device loss enters a mandatory waiting period during which the legitimate owner may cancel a fraudulent attempt.

8. Privacy Considerations

No PII on-chain. The DID and TAAD datum contain no personally identifiable information. Biometric data is never stored on-chain.

Cross-chain links are user-initiated. alsoKnownAs entries constitute a public, permanent assertion of identity equivalence. Users should understand this before adding links.

Activity timeline. TAAD UTxO spending history is publicly visible on Cardano. This is inherent to any blockchain-anchored DID method.

Resolver logging. Resolver operators may log resolution requests. Users requiring query privacy should operate their own resolver.

---

9. Reference Implementation

Component	URL
Full method specification	https://phoenixkey.me/did-method/v1
TAAD Validator (Plutus V3)	https://github.com/PhoenixKeyDID/phoenixkey-validator
Resolver (Node.js)	https://github.com/PhoenixKeyDID/did-phoenix-resolver
Universal Resolver Driver	https://github.com/PhoenixKeyDID/uni-resolver-driver-did-phoenix
JSON-LD Context	https://phoenixkey.me/context/v1
PhoenixKey Whitepaper	https://phoenixkey.me/whitepaper
