Open source data on Ceramic
Why the future of the web is self-certifying
Regulation to force data interoperability

Company A

API 1

Company B

Integration
Regulation to force data interoperability

Company A

API 1

Company B

Integration

API 2

Re-integration
Self-Certifying protocols provides an alternative

1. Cryptographic user keys
2. Content-addressed data

Image credit: Web3 is Self-Certifying – Jay Graber
Self-Certifying protocols provide an alternative

1. Cryptographic user keys
2. Content-addressed data
Ceramic is a network of self-certifying data objects

**Verifiable history:** each object is represented as a stream of events, where each event is signed and includes a pointer to the previous event.

**Location agnostic:** each object is accessed globally by StreamID, and anyone can make them available on their Ceramic node.

**Open data:** an object in Ceramic can link to any other object creating a global information graph.

**Shared network effects:** objects are owned and controlled by users, and thus doesn’t create app silos.
Self-Certifying Objects in Ceramic

1. Stream of signed updates
   - Linked events
   - Any IPLD dag

2. Immutable StreamID
   - Created from hash of genesis event
   - Global namespace

3. Signed by DID
   - Cross-network, key-agnostic, future-proof
   - Key rotation & revocation built-in

4. Events are processed by local node
   - Validate signature and StreamType
   - Update & track state
   - Share with network via libp2p

5. Updates anchored onto a blockchain
   - Provides proof-of-existence
   - Enables secure key rotation

6. Configurable state transition logic
   - StreamTypes define custom logic for how to process events
Identity on Ceramic

**DIDs**: a w3c standard for Decentralized Identifiers

**PKH DID**: Makes any account on any blockchain into a Decentralized Identifier

**3ID DID**: Ceramic native identities that aggregate accounts across different blockchains

**NFT DID**: all NFTs can be used to facilitate write access to data objects, or be used as identities

**Other DID methods**: Ceramic can easily be extended to support various types of DIDs
How do we use this to build apps?
DataModels – templates for user data

https://github.com/ceramicstudio/datamodels/
### Human centric data using Self.ID

#### Data Model #1

<table>
<thead>
<tr>
<th>DID</th>
<th>kjz123...</th>
<th>kjz456...</th>
<th>kjz789...</th>
</tr>
</thead>
<tbody>
<tr>
<td>did:pkh:eip155:1:0xab12...</td>
<td>Alice</td>
<td>{ key1: “nft”, key2: “wizard” }</td>
<td>Y2IwaGVydGV4dGZld8Ko...</td>
</tr>
<tr>
<td>did:3:kjzv4r3ujm6...</td>
<td>Bob</td>
<td>{ key1: “foo”, key2: “bar” }</td>
<td>ZmV3cWZhc2RmZXdmE5...</td>
</tr>
<tr>
<td>did:nft:eip155:1:0xcd34...</td>
<td>Carol</td>
<td>{ key1: “defi”, key2: “ape” }</td>
<td>w7ZvcDg0ZzM5N2hzcw43...</td>
</tr>
</tbody>
</table>

#### Data Model #2

#### Data Model #3
## Human centric data using Self.ID

### Data Model #1

<table>
<thead>
<tr>
<th>DID</th>
<th>kjz123...</th>
<th>kjz456...</th>
<th>kjz789...</th>
</tr>
</thead>
<tbody>
<tr>
<td>did:pkh:eip155:1:0xab12...</td>
<td>Alice</td>
<td>{ key1: “nft”, key2: “wizard” }</td>
<td>Y2lwaGVydGV4dGZld8Ko...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>did:3:kjzv4r3ujm6...</th>
<th>Bob</th>
<th>{ key1: “foo”, key2: “bar” }</th>
<th>ZmV3cWZhc2RmZXdmE5...</th>
</tr>
</thead>
</table>

| did:key:z6Mkp... | Carol    | { key1: “defi”, key2: “ape” } | w7ZvcDg0ZzM5N2hzcw43... |

### Data Model #2

### Data Model #3
Human centric data using Self.ID

<table>
<thead>
<tr>
<th>DID</th>
<th>Name</th>
<th>Description</th>
<th>Data schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>kjz123...</td>
<td>Alice</td>
<td>{ key1: &quot;nft&quot;, key2: &quot;wizard&quot; }</td>
<td>Y2lwaGVydGV4dDZld8Ko...</td>
</tr>
<tr>
<td>did:pkh:eip155:1:0xab12...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>did:3:kjzv4r3ujm6...</td>
<td>Bob</td>
<td>{ key1: &quot;foo&quot;, key2: &quot;bar&quot; }</td>
<td>ZmV3cWZhC2RmZXdmE5...</td>
</tr>
<tr>
<td>did:key:z6Mkp...</td>
<td>Carol</td>
<td>{ key1: &quot;defi&quot;, key2: &quot;ape&quot; }</td>
<td>w7ZvoDg0ZzM5N2hzow43...</td>
</tr>
</tbody>
</table>
Human data commons
New Docsite: A new documentation site for developers to learn about Ceramic!

Go-IPFS: Support for DagJOSE (IETF standard for signing & encryption) released in IPFS 0.11

Object Capabilities: Secure session keys for dapps, using Sign-in with Ethereum + CACAO

TipSync: Using the libp2p DHT to query streams in Ceramic. Will enable greater scalability and resilience of the network

Glaze CLI: A comprehensive CLI for interacting with Ceramic and DataModels
Research topics

**Stream privacy:** Easy to encrypt content, hard to encrypt metadata while maintaining public verifiability

**CRDTs:** In a multiple writer scenario conflicts in the even log will occur, CRDTs in combination with IPLD can solve this

**Recursive ZKPs:** By proving each state transition of a stream with a recursive ZKP we can verify the integrity of the entire event log with just one proof (excluding key rotations)

**Validator network:** Users should be able to pay validators to keep their data available in the network
Introduction to the Ceramic Protocol

Joel Thorstensson, @joelthorst

Documentation: https://developers.ceramic.network

Self.ID: https://self.id

Chat with us: https://chat.ceramic.network