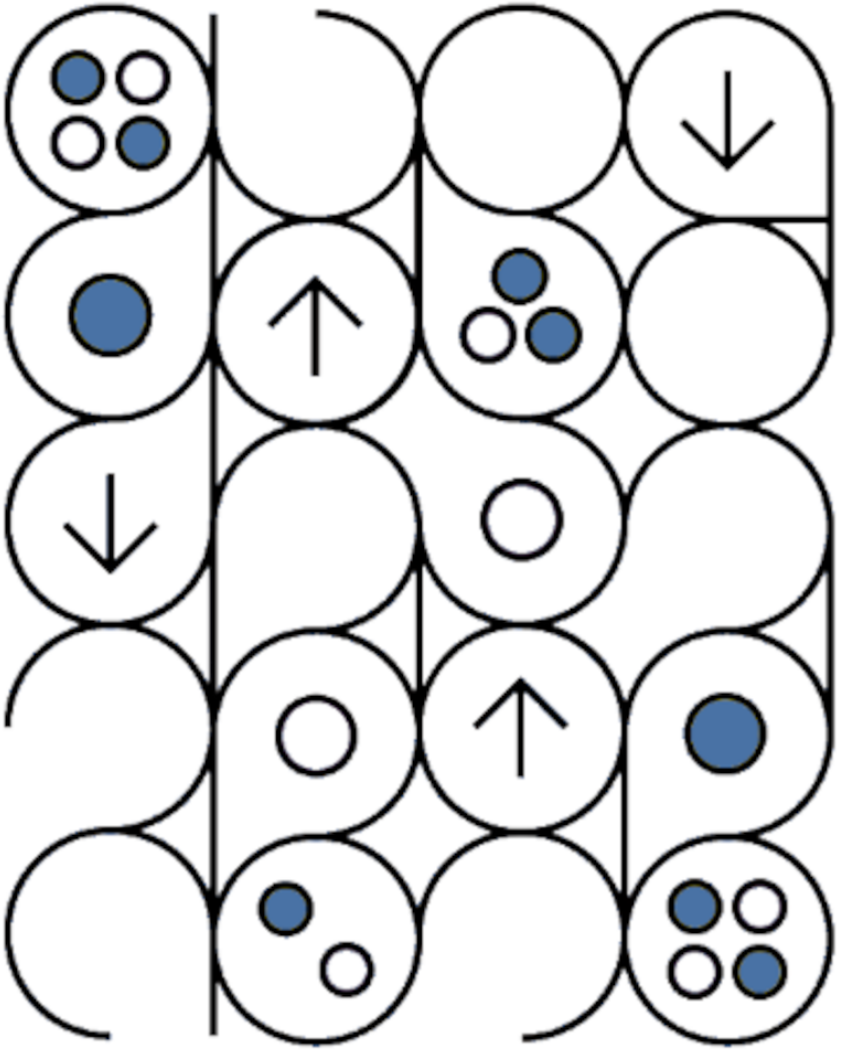


NextGraph
.org

Niko Bonnieure
Laurin Weger

European Commission
DG CNECT

NGI ASSURE

NextGraph
.org

NGI ZERO COMMONS

nlnet
FOUNDATION

A story about data

A story about data

Where is my data today ?

A story about data

Where is my data today ?

👎 **Big Tech**

Big Tech

Myriad of accounts

Big Tech

Myriad of accounts

No access to raw data

Big Tech

Myriad of accounts

No access to raw data

Plaintext in their cloud

Open Source Alternatives

Open Source Alternatives

Myriad of accounts

Open Source Alternatives

👎 Myriad of accounts

👎 **Incompatible APIs**

Open Source Alternatives

👎 Myriad of accounts

👎 Incompatible APIs

👎 **Plaintext in their server**

I want my data

I want my data
I want all my data

I want my data

I want all my data

I want all my data in one place

All my data in one place **control**

All my data in one place

control

ownership

All my data in one place

control

ownership

availability

control
ownership
availability
security & privacy

control
ownership
availability
security & privacy
share & collaborate*

I want a central place for all my data

With E2EE

Server = store & forward

I want a central place for all my data

With E2EE

Server = store & forward

Local First & CRDT

I want a central place for all my data

How many servers?

I want a central place for all my data

How many servers?

A federation!

I want a central place for all my data

How many servers?

A federation!

Synced in a pub/sub

I want a central place for all my data

Replication on all devices

I want a central place for all my data

Replication on all devices

And on brokers

I want a central place for all my data

Replication on all devices

And on brokers

Full decentralization

I want a completely decentralized central place for all my data

And that's

NG proto Specialized for E2EE and CRDT

NG proto

Specialized for E2EE and CRDT

Can sync any kind of CRDT

NG proto

Access control with Cryptographic capabilities

NG proto

DID decentralized identifiers

did:ng:

NG proto

upload/download binary files, with streaming, content addressing, and chunking

NG proto
Supported CRDTs

RDF
aka Linked Data
W3C standard
triples
SPARQL

RDF

Interoperability
Malleable software

RDF

Link and reference other documents.

It is a database

RDF

Automatic joins
no foreign key
all data joinable

RDF

Global IDs for each record, using URI. did:ng:...

RDF

A global database!

Tim Berners-Lee used to call it the Giant Global Graph

NextGraph engine :

Sync protocol

CRDT agnostic

Graph database

Encryption at rest

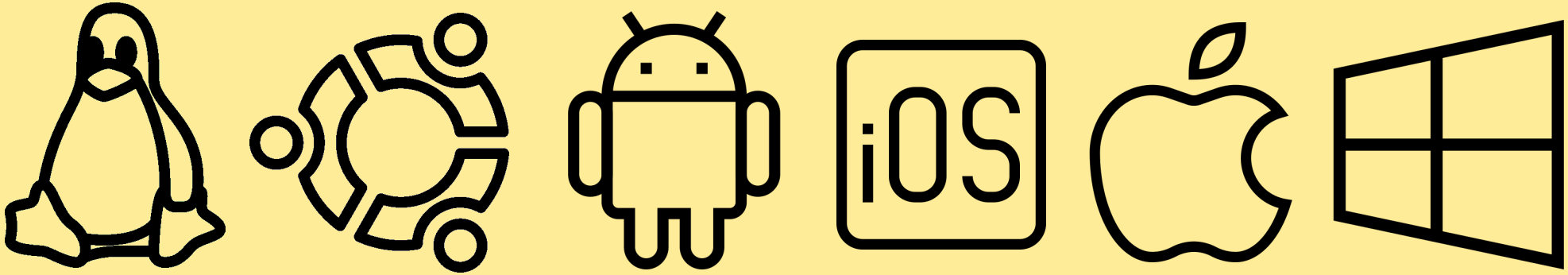
NextGraph SDK

Web (WASM)

Native (Tauri, webview)

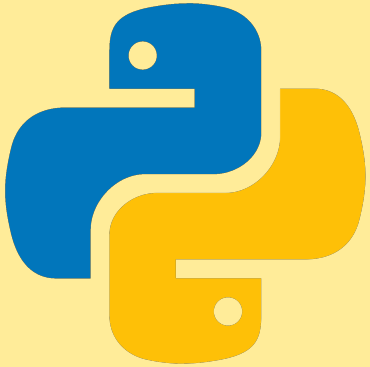
Rust, Python, Nodejs

NextGraph SDK



NextGraph SDK



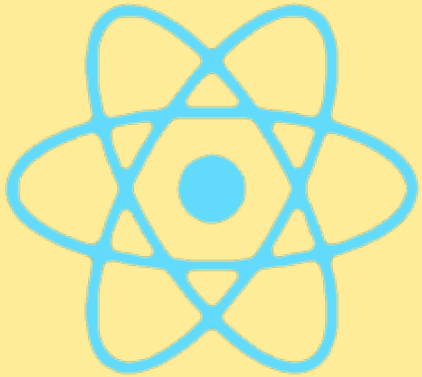



NextGraph SDK

Raw APIs: subscribe, update CRDT binary blobs, or RDF triples

ORM APIs: higher level. bind your POJOs to reactive frontend components

NextGraph SDK



React

SVELTE

Vue.js

NextGraph Framework

History

revision, signature, immutability, audit trail, Authenticity, time travel

Permissions

grant access: read, write, subscribe, pull
Create groups, share with others

NextGraph Framework

Search

full text search, compound indexes

SPARQL

Local queries on all documents, or a subset, or a store.

Reactive Queries

Incremental View Maintenance (IVM) (soon)

NextGraph Framework

Comments

on any document, within apps too

Chat

And group chat

Notifications

With OS integration

NextGraph Framework

Stores / Drive

Public, Protected, Private stores, and Groups

Wallet

User Identity manager, password, keys

App store

Install and publish apps, all decentralized

NextGraph Framework

Social Network

P2P web of trust private social network

Smart contracts

ACID transactions

with strong consistency. Paxos coordination

NextGraph Ecosystem

Web Apps

in third party mode

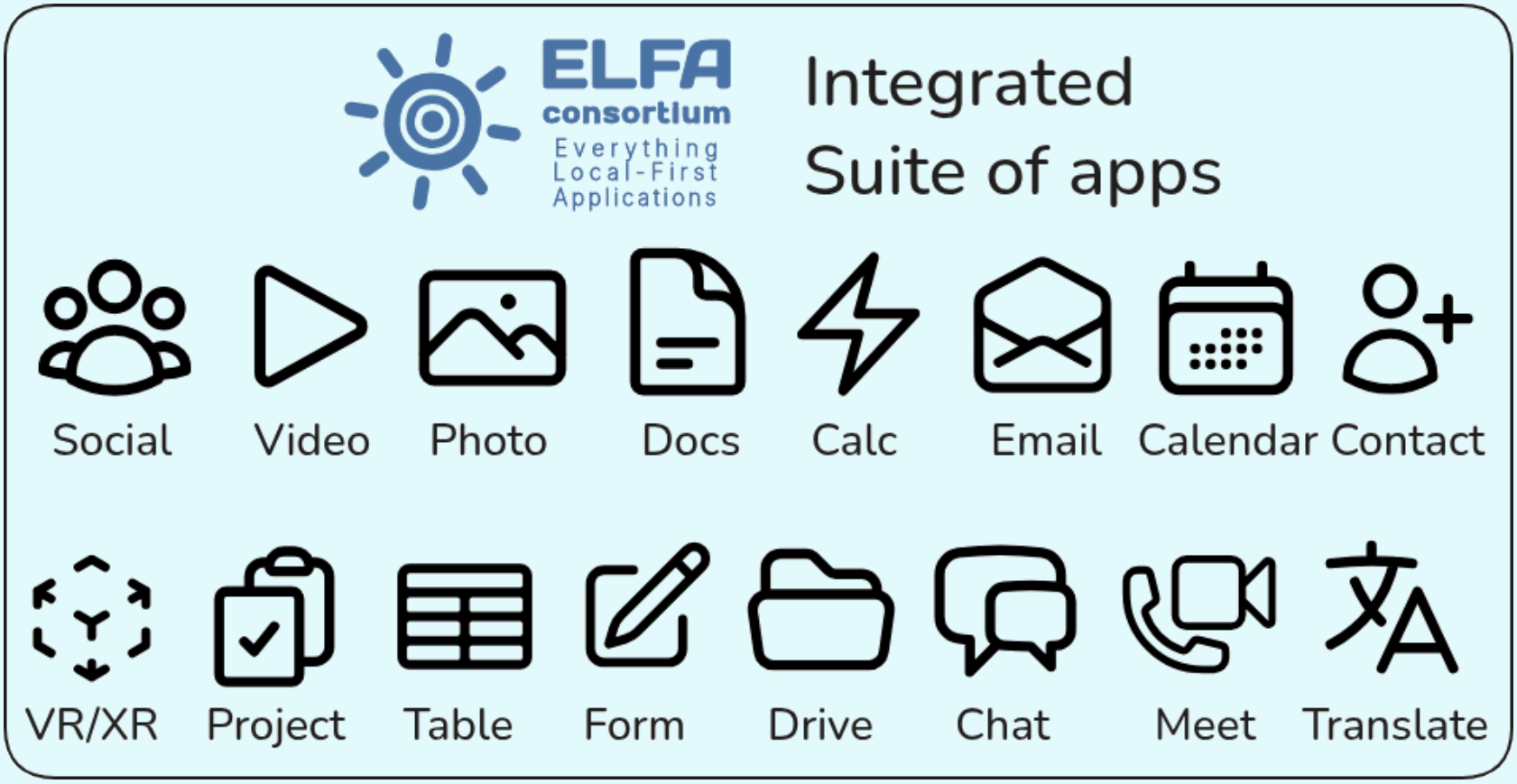
omyn.ai

Native Apps

Services

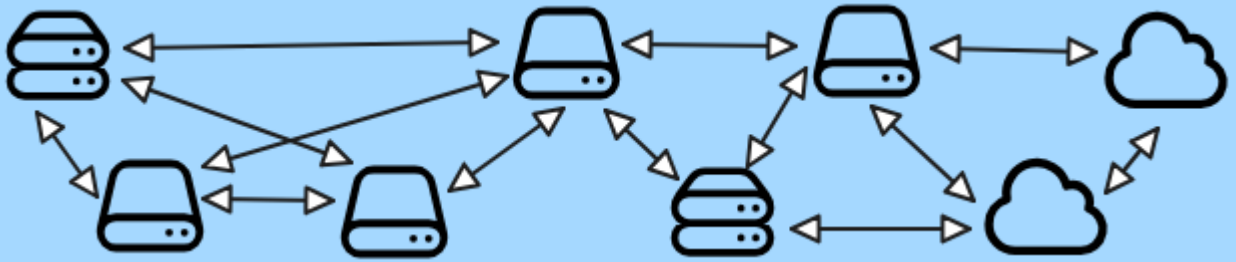
AtomicServer

Everything Local First Applications



NextGraph stack

OPEN PLATFORM : a federation of brokers



At home/office, On Premise, Self-hosted, community hosting, hosters, integrators, edge, cloud

NextGraph stack

NG PROTO : an E2EE encrypted sync protocol
NG ENGINE : local first database

NG OPEN PLATFORM : a federation of brokers

NextGraph stack

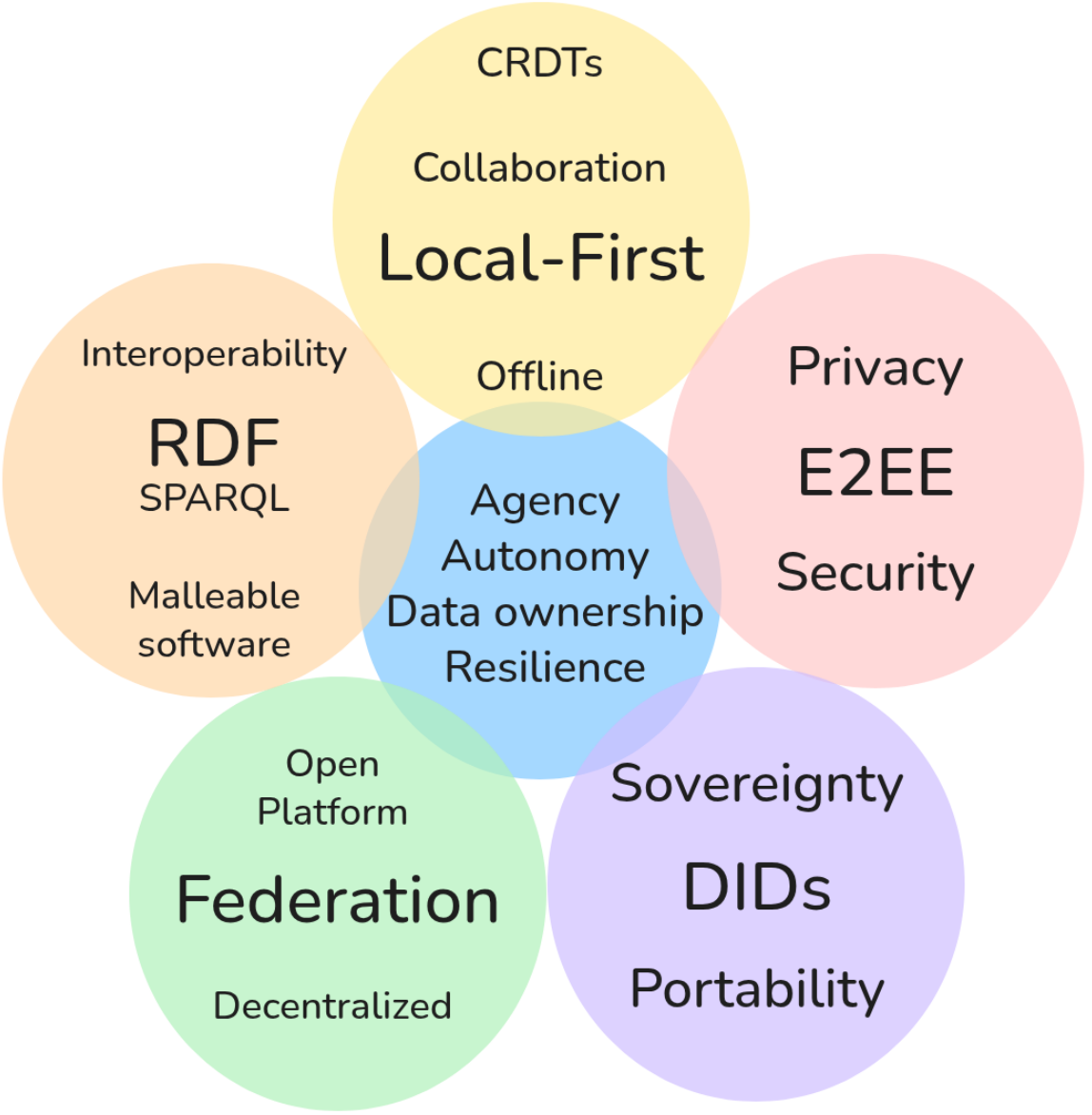
NG ECOSYSTEM : apps and services, ELFA productivity suite, social network, bridges, etc...

NG FRAMEWORK : common features & facilities

NG SDK : build web & native apps, services

NG PROTO & NG ENGINE : E2EE sync database

NG OPEN PLATFORM : a federation of brokers


CRDTs
Collaboration
Local-First
Offline
Interoperability
RDF
SPARQL
Malleable software
Agency
Autonomy
Data ownership
Resilience
Privacy
E2EE
Security
Open
Platform
Federation
Decentralized
Sovereignty
DIDs
Portability

Still in alpha. Next steps

performance
Tauri plugin
App store
framework
core protocol

The TypeScript ORM

The TypeScript ORM

Modify Data as a Regular TS Object → no CRDT-specific APIs

YJS: `ydoc.getMap().set("key", "value")`

NextGraph: `data.key = "value"`

The TypeScript ORM

Modify Data as a
Regular TS Object
→ no CRDT-specific APIs

Reactive, Live-Sync with your Frontend

Demo (RDF ORM)

Demo (YJS / Automerge)

Walk-through

→ Object is changed

```
expense.title = “FOSDEM Train Ticket”
```

Walk-through

→ Object is changed

```
expense.title = “FOSDEM Train Ticket”
```

→ Engine persists CRDT

Walk-through

→ Object is changed

```
expense.title = “FOSDEM Train Ticket”
```

→ Engine persists CRDT

→ **Syncs to other replicas**

Walk-through

→ Object is changed

```
expense.title = “FOSDEM Train Ticket”
```

→ Engine persists CRDT

→ Syncs to other replicas

→ **Engine receives change**

Walk-through

→ Object is changed

```
expense.title = “FOSDEM Train Ticket”
```

→ Engine persists CRDT

→ Syncs to other replicas

→ Engine receives change

→ TS object is updated

Walk-through

→ Object is changed

```
expense.title = “FOSDEM Train Ticket”
```

→ Engine persists CRDT

→ Syncs to other replicas

→ Engine receives change

→ TS object is updated

→ **Frontend re-renders**

Frontend Code (React)

```
function ExpenseInputs() {
    // For discrete (JSON) CRDTs
    const { doc } = useDiscrete(documentId);
    const expenses = doc.expenses;

    // Or for graph ORM across all documents
    const expenses = useShape(ExpenseShape, scope);

    return expenses.map((expense) => (
        <input
            value={expense.title}
            onChange={(event) => {
                // Modifying data triggers rerender.
                expense.title = event.target.value;
            }}
        ></input>
    ));
}
```

Schemas & Typing

For RDF Data

TS Type Definition

```ts
export interface Expense {
  /** The graph IRI.*/
  readonly "@graph": string;
  /** The subject IRI.*/
  readonly "@id": string;
  "@type": "http://example.org/Expense";
  /**
   * The name of the expense
   * Original string: http://example.org/title
   */
  title: string;
  totalPrice: number;
  dateOfPurchase: string;
  expenseCategory?: Set<ExpenseCategory>;
  paymentStatus:
    | "http://example.org/Paid"
    | "http://example.org/Pending"
    | "http://example.org/Overdue"
    | "http://example.org/Refunded"
}
```

```ts
export interface ExpenseCategory {
  readonly "@graph": string;
  readonly "@id": string;
  "@type": "http://example.org/ExpenseCategory";
  categoryName: string;
}
```

SHEX Schema Definition

```
PREFIX ex: <http://example.org/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

ex:ExpenseShape {
  a [ex:Expense] ;

  ex:title xsd:string
    // rdfs:comment "The name of the expense" ;

  ex:totalPrice xsd:float
    // rdfs:comment "The total price" ;

  ex:amount xsd:integer
    // rdfs:comment "The number of items bought" ;

  ex:dateOfPurchase xsd:date
    // rdfs:comment "The date of purchase" ;

  ex:expenseCategory @ex:ExpenseCategoryShape *
    // rdfs:comment "The use category of the product" ;

  ex:paymentStatus [ex:Paid ex:Pending ex:Overdue ex:Refunded] ;
    // rdfs:comment "Status of payment" ;
}
```

```
ex:ExpenseCategoryShape {
  a [ ex:ExpenseCategory ] ;

  ex:categoryName xsd:string
    // rdfs:comment "Name of expense category" ;

  ex:description xsd:string
    // rdfs:comment "Human-readable description of category" ;
}
```

Summary

Modify CRDTs in TS like any other object

Summary

Modify CRDTs in TS like any other object

ORM for RDF (and Automerge and YJS)

Summary

Modify CRDTs in TS like any other object

ORM for RDF (and Automerge and YJS)

Type Safety for RDF

Summary

Modify CRDTs in TS like any other object

ORM for RDF (and Automerge and YJS)

Type Safety for RDF

Have reactivity in React, Vue, Svelte

Summary

Modify CRDTs in TS like any other object

ORM for RDF (and Automerge and YJS)

Type Safety for RDF

Have reactivity in React, Vue, Svelte

Live-Sync with other users and devices

NextGraph.org

mastodon:

@nextgraph@fosstodon.org

+ forum, newsletter

Come say hi