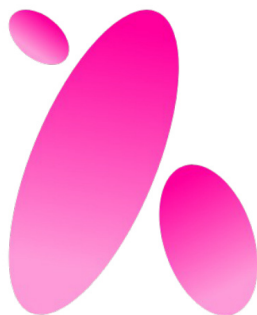


Interoperable Local-First in The Age of AI

Noel De Martin · 2026-10-02 · English

Edited transcript of the presentation and discussion on 2 October 2026. Repetitions and clear recognition errors have been corrected while preserving the order, examples, qualifications and questions. Product status and plans reflect the talk date. Timestamps refer to the edited video.



Interoperable Local-First in the age of AI

noeldemartin.com/local-first-2026

Interoperable Local-First in the age of AI · Frame 00:00:00

Hello, my name is Noel



- Developer & Entrepreneur.
- Working with the Solid Protocol.
- Working in the Open.

noeldemartin.com | noeldemartin.social | [@NoelDeMartin](https://twitter.com/NoelDeMartin)

noeldemartin.com/local-first-2026

Noel and working in the open · Frame 00:00:05

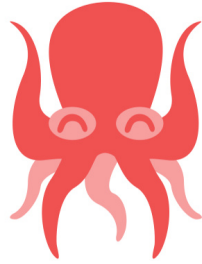
00:00:00 · Noel De Martin

Hello, everyone. My name is Noel. I'm a developer and entrepreneur, and I have been working for a while on the Solid protocol. I will talk more about this later. I work in the open, so everything I talk about today and most of my Solid work is open source. You can go to my GitHub and see how everything works under the hood. I also keep a development journal where I explain the things I'm working on. So if you're interested, you will be able to dig deeper into most of the things I talk about today.

My Apps



Focus



Media Kraken



Umai

Find them at noeldemartin.com/projects

noeldemartin.com/local-first-2026

Focus, MediaKraken and Umai · Frame 00:00:37

00:00:35 · Noel De Martin

I started by making apps. This is what got me into this journey with Solid and local-first. I have a few applications that you can use if you want. So far, these three are the main ones, or the ones that are more stable. If you're interested in learning more, I also have other projects and things you can explore on my website. But making applications is my main thing. The reason I started making applications is that I wasn't very happy with the software I used.

Three ideals: universality, sovereignty and interoperability

My 3 Software Ideals

- Universality
 - The Web (any device)
 - Local-first (any location)

noeldemartin.com/local-first-2026

Software ideals: universality · Frame 00:01:50

00:01:09 · Noel De Martin

After thinking more about it, I came up with some ideas about how I want my software to work. There are three of them. The first is universality. This means that I want to use my software on any device. This is why all of my applications are web applications: the web is already universal. It's one of its main principles, and you can already use a website on any device. So that covers this part.

00:01:41 · Noel De Martin

I also wanted to be able to use my apps in any location. It was frustrating when I was on a train or the subway, without an internet connection, and my applications stopped working—even if I needed to do something that didn't require an internet connection. Thinking about that is how I came across the local-first movement, and I really liked it. So all my applications are also local-first. It's one of the things that fits my ideals for apps.

My 3 Software Ideals

- Universality
- Sovereignty
 - Longevity
 - Privacy & Data Ownership
 - Untangle Network Effects

noeldemartin.com/local-first-2026

Software ideals: sovereignty · Frame 00:03:17

00:02:13 · Noel De Martin

The second ideal is sovereignty. This is also what first got me into the Solid protocol; you will learn more about it later. What do I mean by sovereignty? One example is longevity. I want to be able to use my applications and software for a long time. The main reason I started making my own apps was that I was frustrated with third-party applications that stopped working. Some were even paid apps, but because someone acquired the company or something happened, they started degrading.

00:02:50 · Noel De Martin

Even though I liked the application and was a paying customer, they started changing or removing things, and I couldn't keep using it. This is another of my ideals for software: you should be able to use it for as long as you want. There is also privacy and data ownership. I want to be in control of what happens to my data, and I want to really own it. Finally, I would also like to untangle network effects.

00:03:21 · Noel De Martin

This is something I would like, but I'm not working on it very much because my applications are not social applications. When it comes to social software, most of the time I have to keep using the same ones other people use. I have a Twitter, or X, account, a Bluesky account and a Mastodon account. But I cannot decide to use only my own social network, because nobody is going to use it. Ideally, software shouldn't depend on network effects.

My 3 Software Ideals

- Universality
- Sovereignty
- Interoperability
 - Malleable Software
 - Interoperable Serendipity

noeldemartin.com/local-first-2026

Software ideals: interoperability · Frame 00:04:19

00:03:51 · Noel De Martin

People should be able to choose their own software and interoperate with others. That brings me to the third ideal: interoperability. You may have seen the term malleable software, even if you haven't heard about interoperability. A term I came up with, and the way I like to describe what I want from interoperability, is interoperable serendipity. I define it like this: interoperable serendipity happens when apps built by different developers work together without explicit coordination.

Interoperable Serendipity happens when apps built by different developers work together without explicit coordination”

Apps built by different developers work together without explicit coordination · Frame 00:04:42

00:04:30 · Noel De Martin

What does that mean? Suppose I'm using a task manager, and another developer or company makes another task manager for the same use case. Both are task managers; technically, they do the same thing. It would be nice if those two worked together out of the box. Some applications are interoperable, but that is because both developers are working on the same stack, or perhaps using the same protocol.

Interoperable Serendipity

“Interoperable Serendipity happens when apps built by different developers **Als** work together without explicit coordination”

noeldemartin.com/local-first-2026

AI and interoperable serendipity · Frame 00:06:11

00:05:01 · Noel De Martin

Most of the time, they are doing that explicitly. My end goal—the way I would like software to work—is software that works together out of the box, without anyone having to think about interoperability. That was my original description of the term, but nowadays I think it is also important to say that we want interoperable serendipity with AI. A lot of people are now making their own applications using things like Lovable, v0 or other AIs that make apps.

00:05:40 · Noel De Martin

It would be nice if these applications were interoperable out of the box. If I vibe-code a task manager and you vibe-code a task manager, it would be nice if both were interoperable. Even if I vibe-code two task managers using the same vibe-coding tool, most of the time they won't be interoperable anyway. For me, this is something we should solve. It would be nice if software worked like this: interoperable out of the box.



noeldemartin.com/solid-world-2025

More on interoperable serendipity · Frame 00:06:23

00:06:15 · Noel De Martin

This idea guides a lot of my work. If you want to learn more about how it can work at a technical level and some of the implications, you can watch this presentation I gave a while ago, where I go into more detail. To give you an initial explanation, or an overview of how this works, I'm going to talk about the Solid protocol. This is the technology I decided to use to solve interoperability.

Solid: separating applications from data

The Solid Protocol



noeldemartin.com/local-first-2026

The Solid Protocol · Frame 00:06:40

What is Solid?

- A protocol for decentralized storage.
 - “Decouple apps from data”

noeldemartin.com/local-first-2026

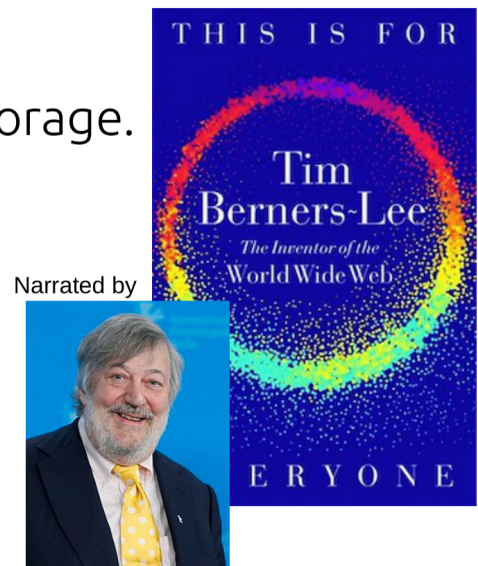
Decoupling apps from data · Frame 00:07:00

00:06:47 · Noel De Martin

In case you don't know what Solid is: Solid is a protocol for decentralized storage. Its main idea, or what it's trying to achieve, is to decouple apps from data, which is another way of talking about interoperability. It was created by Tim Berners-Lee, the creator of the web, and it is an evolution of the web. It isn't a completely different protocol he created out of nowhere. It's an evolution of his idea of how the web should work, expanding the original idea of the web to encompass storage and data ownership as well.

What is Solid?

- A protocol for decentralized storage.
- Created by Tim Berners-Lee.
...as an evolution of the Web.



noeldemartin.com/local-first-2026

Solid and the evolution of the Web · Frame 00:07:58

00:07:27 · Noel De Martin

If you're interested in the history of the web and Solid, I recommend this book by Tim Berners-Lee. It came out, I think, a couple of years ago—I'm not sure now, but recently. He explains how he came up with the web, the things he has been thinking about and the goals he has. If you don't like reading books, or prefer listening to podcasts, I also recommend the audiobook, which is narrated by Stephen Fry. I think it's a great book and audiobook, so please check it out if you're interested in the topic.

What is Solid?

- A protocol for decentralized storage.
- Created by Tim Berners-Lee.
- Learn more:
 - solidproject.org
 - [Why Solid?](#)

noeldemartin.com/local-first-2026

Learn more about Solid · Frame 00:08:13

00:08:03 · Noel De Martin

You can also learn more about Solid on the official website, solidproject.org. Some years ago, I wrote a blog post called [Why Solid](#), explaining why I chose it for my applications. I discuss the pros and cons and what you can do with Solid. Now I'll explain some basic concepts so you can understand how it works technically. The main idea of Solid is Solid Pods.



Solid basic concepts · Frame 00:08:28

Solid Basic Concepts

- PODs → Personal Object Datastore

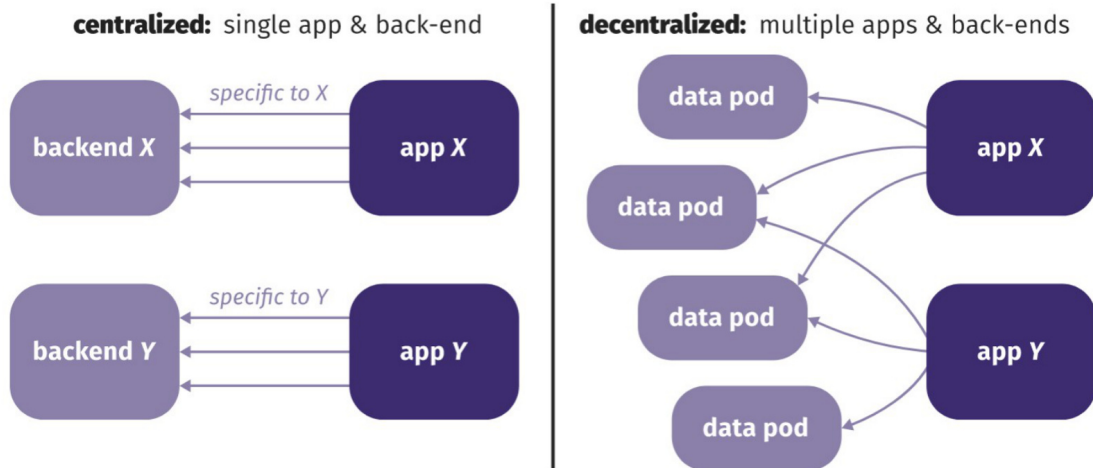


noeldemartin.com/local-first-2026

PODs: personal data stores · Frame 00:08:35

A Solid Pod is a Personal Object Datastore. Think of Dropbox, Nextcloud or other cloud storage: the basic idea is that you have your own data storage. But it isn't just files; it is semantic data, data that software can understand, rather than only binary blocks. Usually, we follow the familiar paradigm where you use an application and the data is stored in that application's backend.

Solid Basic Concepts



Ref: [Ruben Verborgh @ FOSDEM 2019](#)

noeldemartin.com/local-first-2026

Separating applications and data · Frame 00:09:59

For example, if you upload a picture to Facebook and another picture to Twitter or X, each application has its own image storage. With Solid, the idea is that your things live in pods, and different applications can access the same data. Suppose you have a social-network application. If you upload images through it, and it is built using the Solid protocol, another application—an image viewer or image manager—will be able to show and manage all the pictures you uploaded through the social-network application.

Solid Basic Concepts

- PODs → Personal Object Datastore
- RDF → Resource Definition Framework
 - Data Graph + Custom “Ontologies” or “Vocabularies”
 - e.g schema.org
 - Semantic Web

noeldemartin.com/local-first-2026

RDF, graphs and vocabularies · Frame 00:10:44

00:09:51 · Noel De Martin

That's the idea: you own the data, and applications are just ways of working with it. Data and applications don't have to be coupled together. This is the main idea of Solid. Technically, it works using something called RDF, the Resource Definition Framework. Essentially, it's a data graph with custom ontologies or vocabularies. You may have seen schema.org, which is used for SEO, but the applications of RDF are much wider than SEO.

Editorial note: The speaker says “Resource Definition Framework,” which also appears in the slides. The standard expansion of RDF is Resource Description Framework. The transcript preserves the spoken wording; this note supplies the terminology correction. W3C RDF 1.1 Concepts and Abstract Syntax

00:10:28 · Noel De Martin

schema.org has become more popular because of that use case. The basic idea is that you define different types of data using these vocabularies or ontologies, and that is how applications know how to use the data. This is also called the semantic web; you may have seen the term before. As I mentioned, it is built on the web, rather than being a new protocol that appeared out of nowhere. It works with the basic HTTP verbs you already know: GET, POST and so on.

Solid Basic Concepts

- PODs → Personal Object Datastore
- RDF → Resource Definition Framework
- Built on top of the Web (GET, POST, etc.)
- Permissions, Authentication, etc.

noeldemartin.com/local-first-2026

Solid is built on the Web · Frame 00:11:02



noeldemartin.com/fosdem

Getting started with Solid · Frame 00:11:18

00:11:00 · Noel De Martin

It adds some extra functionality on top of the web: permissions, authentication and the other things you need to make an application. If you want to dig into how Solid works, I also gave this presentation some years ago, going into the details from a developer's perspective. You can watch it if you're interested in the technical details. I started working on Solid a few years ago, before all these LLMs and AIs became so popular.

What changes in the age of AI?

The Age of AI

- Generative AI
- Vibe Coding
- AI Agents
- ... but where are all the apps!?
- ... do we even want apps!?



noeldemartin.com/local-first-2026

The Age of AI: what apps do we want? · Frame 00:13:10

00:11:38 · Noel De Martin

Today, though, we are in a different paradigm: the age of AI. Everybody is using AI for everything. We have generative AI, vibe coding and AI agents. The way software works is changing, and in theory software is easier to make than ever before. However, when I look at the actual outputs of AI, or what is happening in the industry, personally I don't see a lot of new applications that I'm using.

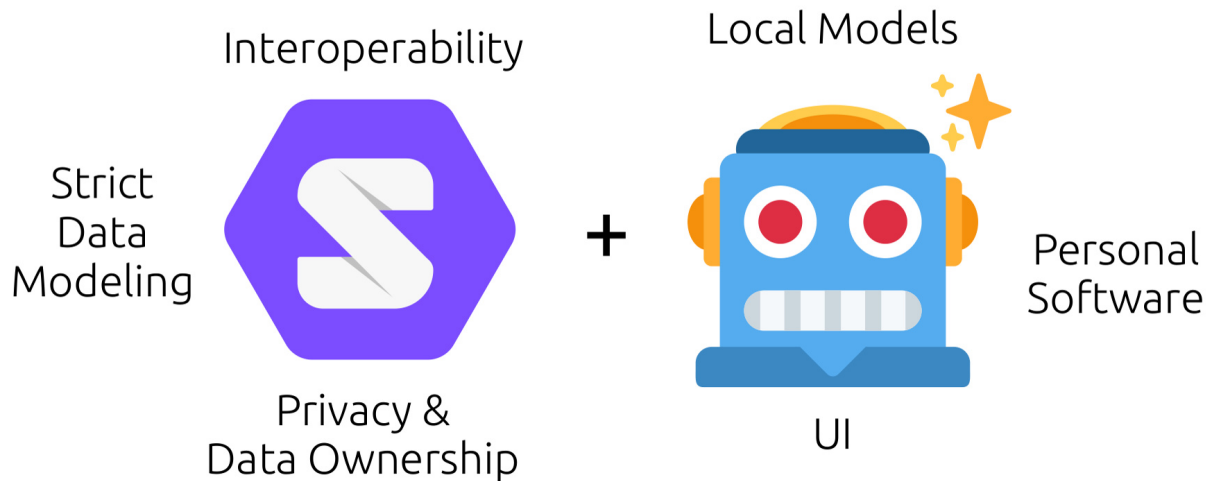
00:12:10 · Noel De Martin

I see people talking about new applications and saying they made one in five minutes, but personally, I can't say I'm finding more applications I like than before. From my point of view, the situation is very similar to what we had before AI. I think most people would agree with that. You may disagree, but I think that's the general opinion. What some people tell me, though—and I can agree with them—is that maybe we don't even want applications.

00:12:44 · Noel De Martin

Maybe the reason we don't have new apps is that, with AI, you don't need apps. You can use AI to achieve your purposes or carry out your tasks, so what is the point of having applications? I agree to a certain extent, but personally, I still like having applications. That's why I'm still working on them. I do want them to follow a different paradigm, though: the three ideals I mentioned at the beginning.

Solid to the rescue?



noeldemartin.com/local-first-2026

Bringing Solid and AI together · Frame 00:14:40

00:13:15 · Noel De Martin

In my opinion, the way to achieve that—and where AI would be more useful—is to have two layers. The first layer uses Solid, or whichever protocol you choose, to give you real data ownership and privacy. It also gives you strict data modeling. Even though AIs are getting better every day, they can still hallucinate. If you give an AI a strict data model, and you know what you actually have and what data you want to use, it is easier for the AI to build things on top of it.

00:13:53 · Noel De Martin

As I mentioned, it would also be nice if everything we built with AI were interoperable out of the box. In my opinion, this is what Solid brings, and why it is still relevant today—more relevant than ever, I would say. We also have AI, of course, so we can use its new functionality and capabilities to augment this basic idea. If we are using Solid, for example, we can use local models to work on our data.

00:14:26 · Noel De Martin

We can also use AI to build UIs on top of a defined data structure and schema. This is something AI can be good at. If you give it a data schema and it already reuses all the authentication and capabilities, it won't hallucinate as much, and the results will be better. We have also been talking about the idea of personal software. A lot of people say that, with vibe coding and AI, they are now building their own software.

Ànima: combining Solid and AI



Inspired by “Charlie, the AI that works for you”

👉 See w3.org/DesignIssues/Charlie.html

noeldemartin.com/local-first-2026

Ànima and the inspiration of Charlie · Frame 00:16:01

00:14:58 · Noel De Martin

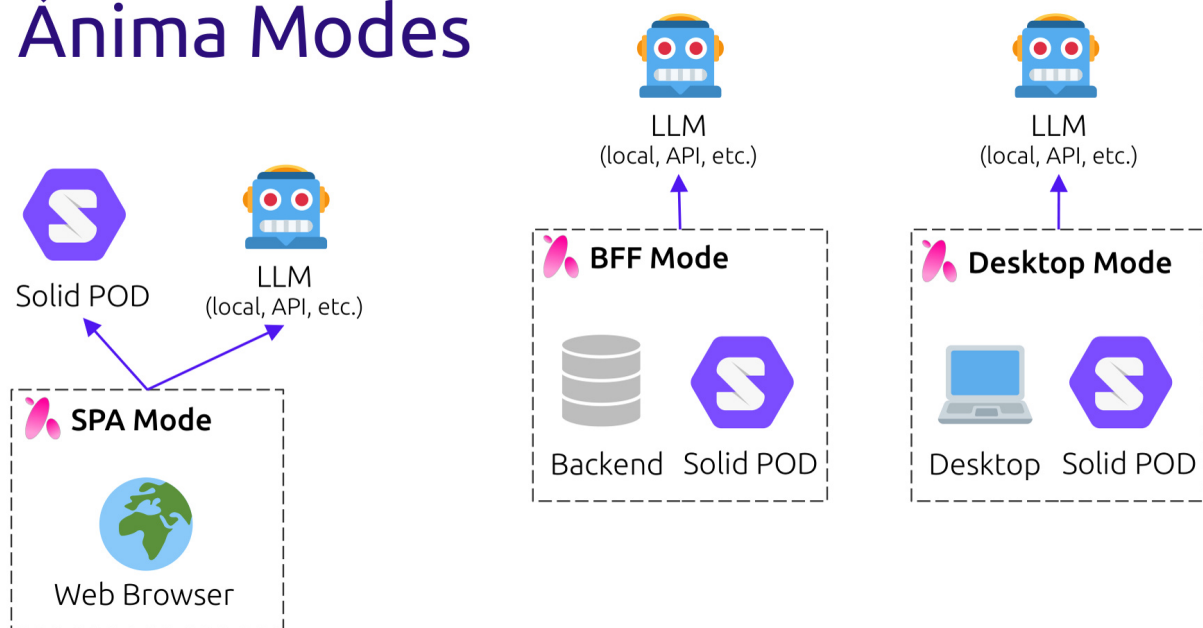
I think that's a great idea and a great use case for Solid. If everything is built with Solid, or with something that follows these ideals, it is much better, in my opinion, because all the data from your vibe-coded applications is reusable. You don't have to start from scratch every time you vibe-code an app or start working on something new. You can build up your data and keep working with it. Ànima is something I've been working on for a few months. It's my answer to how we can combine Solid and AI to achieve what I've been talking about.

00:15:40 · Noel De Martin

It was heavily inspired by another idea from Tim Berners-Lee, called Charlie. In an essay he wrote a few years ago, he describes an AI that works for you—not one owned by a third-party company and serving that company's interests, but an AI that works for you and follows your interests. In my opinion, local models and Solid fit this use case exactly. So how does Ànima work today?

Three modes: SPA, BFF and desktop

Ànima Modes



noeldemartin.com/local-first-2026

Three ways to run Ànima · Frame 00:18:03

00:16:14 · Noel De Martin

As I mentioned on the previous slide, it is still a work in progress, but this is already working. My idea is to have three modes. The first is SPA mode: single-page application. This is what is deployed at the URL right now. If you visit anima.noeldemartin.com, you can use it. It is Ànima running in a browser, with no backend, so you can even run it offline. When you log in with a Solid Pod, all the chats and data you generate in the application will be stored in that pod.

00:16:53 · Noel De Martin

You can connect it to third-party LLMs. That could be a local AI running on your laptop, or a provider such as Anthropic or Gemini—whichever model you want. This is an easy way to start using it and see how it works. But there are other modes I think are more relevant in practice, or that I would use myself. I call the second one BFF, meaning backend for frontend. The basic idea is that Ànima runs on a server.

00:17:26 · Noel De Martin

Since we have a server in this mode, we can take advantage of it and provide a Solid Pod out of the box. One of the main issues with Solid is that people don't have a Solid account, or don't know where to create one. We can use Ànima to have our own self-hosted pod, integrated with the AI and everything I've been talking about. It can also connect to third-party LLMs. The idea is the same, except the pod is integrated with the backend.

00:18:01 · Noel De Martin

Finally, the mode I've been working on, which I think is very promising, is desktop mode. You download Ànima as a binary application and run it on your desktop. With the previous mode, backend for frontend, you need a server or something self-hosted. Most people aren't going to do that, or don't know how. In desktop mode, you only need a computer. You install the desktop application, and whenever the computer is running, you have Ànima working with a Solid Pod wherever you decide.

What works today, and an experimental roadmap

Ànima “Roadmap”

-  Model agnostic (Vercel's AI SDK)
-  Read Solid POD
-  Built-in Solid POD (BFF/Desktop Modes)
-  External reachability (Desktop Mode)
-  Write to Solid POD
-  Create and serve Apps/Views
-  Agentic Loop
-  Scheduled and background Jobs
-  3rd-party chat integration (Telegram, WeChat, etc.)
-  Plugins / Tools

noeldemartin.com/local-first-2026

The Ànima roadmap · Frame 00:21:14

00:18:41 · Noel De Martin

These are the three modes I have right now, and they are already working. Everything is quite experimental. I haven't tested it a lot, but it should work; at least the things I've tried work. What is the roadmap for Ànima? I put roadmap in quotes, because I'm not sure I will do everything here, but these are my current ideas for the project. What already works is that it is model-agnostic. It uses Vercel's AI SDK, so it can support any model that SDK supports.

00:19:15 · Noel De Martin

That means most models. It can also read a Solid Pod: you connect with Solid and it interacts with the pod. And it can provide a built-in Solid Pod in BFF and desktop modes. These are the things that already work today. One thing I'm working on, or thinking about, is external reachability. Desktop mode is probably more interesting to many people, because not everybody has a server or wants to self-host something.

00:19:49 · Noel De Martin

Installing a desktop application is easy, but the problem is that it won't be reachable from outside your computer. I'm thinking about using something like Tailscale or HTTP tunnels, so you can expose the Ànima running on your laptop to the internet if you want. Also, right now it only reads from the pod. Implementing write capabilities would be easy, but before doing that, I want to think it through, because the idea of Solid is that you use it for all your information.

00:20:22 · Noel De Martin

You have to be careful before giving an AI write access to your Solid Pod. I definitely want to do it, but I need to think about how to do it properly. I would also like to give it the ability to create and serve applications, or what I call views. You don't always want a new application; sometimes you just want a small view of data you already have. With Solid and AI, I think we can achieve something like that.

00:20:55 · Noel De Martin

This is more in line with people saying they don't want apps; maybe what they want is views. You don't always want to interact with your data through a chatbot. So that is the idea: give Ànima the ability to create applications or views. Finally, there are some things I may do someday, but I'm not sure, and I make no promises. One is to build an agentic loop. Right now, you talk to Ànima, it gives you a reply, and that's it.

00:21:29 · Noel De Martin

For many tasks, though, a loop is useful: you give it a goal, and it keeps going until it thinks the goal is complete. Another possibility is scheduling background jobs, which is useful for AI and could become more powerful when everything is integrated with Solid. There is also third-party chat integration. Right now, the only way to talk to Ànima is through the website's chat interface, but it would be nice to talk to it through different chat applications as well.

Demo: Focus, a local pod and Ànima

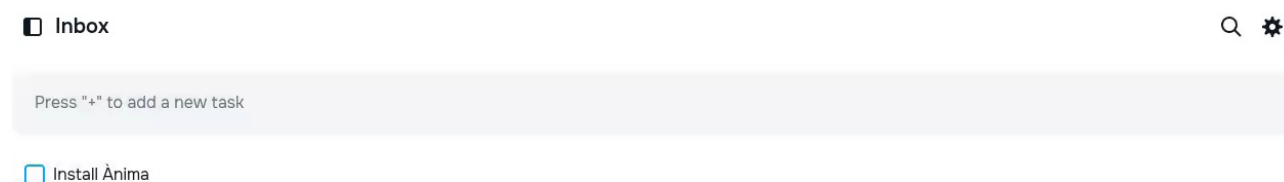
DEMO

noeldemartin.com/local-first-2026

Live demonstration · Frame 00:22:16

00:22:03 · Noel De Martin

Finally, plugins and tools could let people extend the functionality for their own needs. That's the basic idea of what I want to do with Ànima. I've talked a lot about how it works, so let me show you a demo of it in practice. Here is my application, Focus, a task manager. Over here, I have Ànima running. I've installed Ànima's desktop mode on this computer, so I now have a pod running locally.



A local task in Focus · Frame 00:23:10

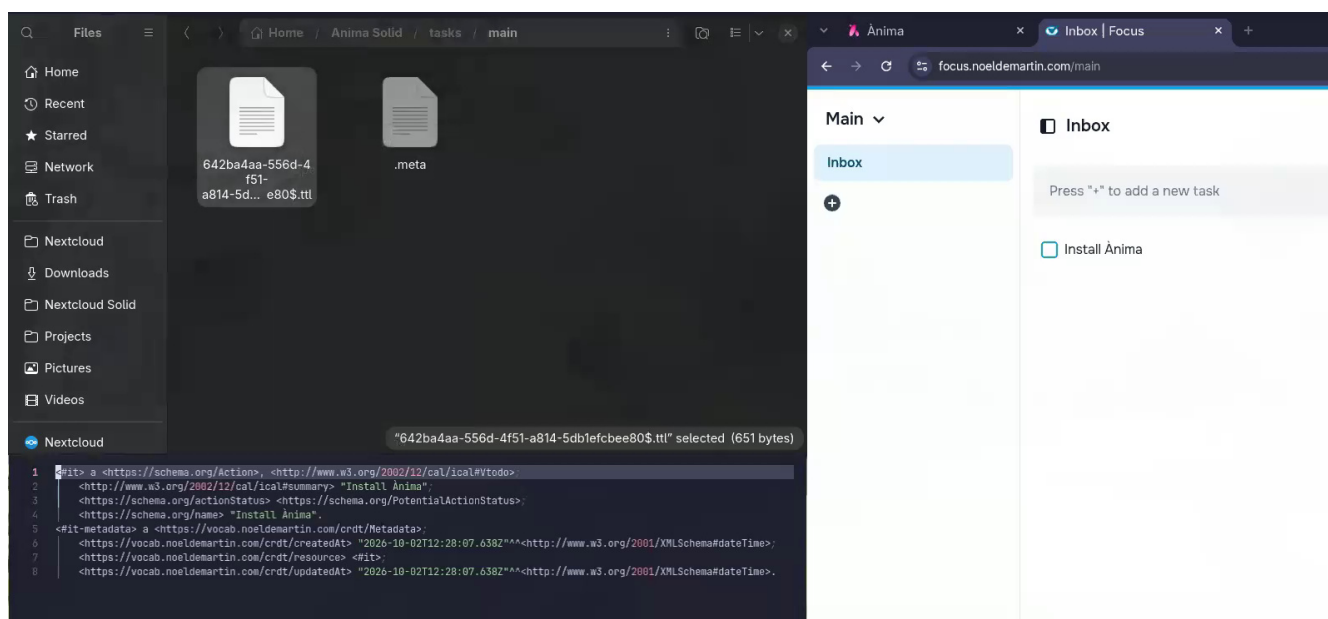
00:22:36 · Noel De Martin

Let's start using Focus, my task manager. Let's add 'Install Ànima.' As you can see, I rehearsed this demo before. We've created a to-do in our task manager. This is local-first, and you may have noticed that I didn't log in or anything. All the data is local. With all my applications, you can use them for as long as you want without creating an account, and all the data will be stored on your device.

00:23:09 · Noel De Martin

At some point, though, you will probably want to synchronize data between devices, or store it securely in the cloud so you don't lose it. Whenever you're ready, you can connect an account by connecting to a Solid Pod. This works with any Solid Pod, not only Ànima or one of my pods; it follows the Solid protocol. We're going to log in to the pod running here on this computer, the one I said I had installed. We copy the URL of the local Ànima instance.

Login, synchronization and readable Turtle files



Synchronizing tasks with local Solid files · Frame 00:24:30

00:23:43 · Noel De Martin

This redirects back to Ànima and asks you to log in and grant permissions. I'm going to log in and allow it. It asks whether you want to back up your data; yes. Now it has synchronized and stored the data in the Solid Pod. Let me show you what that looks like. This is a folder on my computer. When I installed Ànima, I configured which folder to use for storing the data. This is all the data in my pod. Right now, there are only a few things: my profile, some settings and some tasks.

00:24:26 · Noel De Martin

This is what was just created. If we look at the contents, you can see how Solid works. Think of it as similar to Obsidian or applications like it, where you have a collection of Markdown files that are the real source of data, and you can edit the Markdown yourself. With Solid, it's the same idea. Here, the data is in a Turtle file, which is an RDF format. It is human-readable—or at least readable for developers.

00:24:59 · Noel De Martin

It's easy to open the file and see its contents. For example, if we complete the task, it will synchronize, and this file will be updated. As you can see, it has updated: the task is complete. It has also added more data because it uses CRDTs. Since my applications are local-first, every time you make a change, it creates the CRDT data. So now we have this in the pod.

Ask AI about your tasks

00:25:29 · Noel De Martin

We've connected the pod to the application. Let's see what Ànima can do. As I mentioned, it can work with different models. I have a couple of locally running models here, but they aren't very good because my computer doesn't have much RAM. I'm going to do the example with Sonnet from Anthropic. You could use any local or third-party model. Let's ask Ànima something about the pod: 'Which tasks do I currently have open?'

Hello <http://localhost:1191/pod/noeldemartin/profile/card#me>, how can I help you today?

10/2/2026, 2:31:33 PM

Which tasks do I have currently open?

10/2/2026, 2:31:33 PM · Anthropic / Sonnet 5

➤ Read type index

...

Ànima reads personal data · Frame 00:26:18

00:26:03 · Noel De Martin

I think I have to log in again. 'Which tasks do I have?' Yes, the session had probably timed out because I had left this open for a while. You send the message, and Ànima is already connected to your pod, so it can read your data. As you can see, it says: you have the 'Install Ànima' task, but it is completed, so there are no open tasks. If I go here and add a new task—let's say I want to learn Chinese—it synchronizes, so it is now on the pod. As you can see, it has created a new file.

I checked your tasks, and it looks like you only have one task recorded — "Install Ànima" — but it's already marked as **completed** (finished on October 2, 2026).

You currently have **no open tasks**. 🤖

10/2/2026, 2:32:14 PM

Can you take another look, do we have any new tasks?

10/2/2026, 2:32:14 PM · Anthropic / Sonnet 5

➤ Listed files from <http://localhost:1191/pod/noeldemartin/tasks/main/>

There is a new one! Let me check its details.

➤ Read <http://localhost:1191/pod/noeldemartin/tasks/main/cc093f82-db0e-4b51-a5ae-1727607155c2>

Yes, you've got a new one! 🤖

"**Learn Chinese**" — this task is currently **open** (not yet started/completed), created on October 2, 2026.

So to sum up, your open tasks now are:

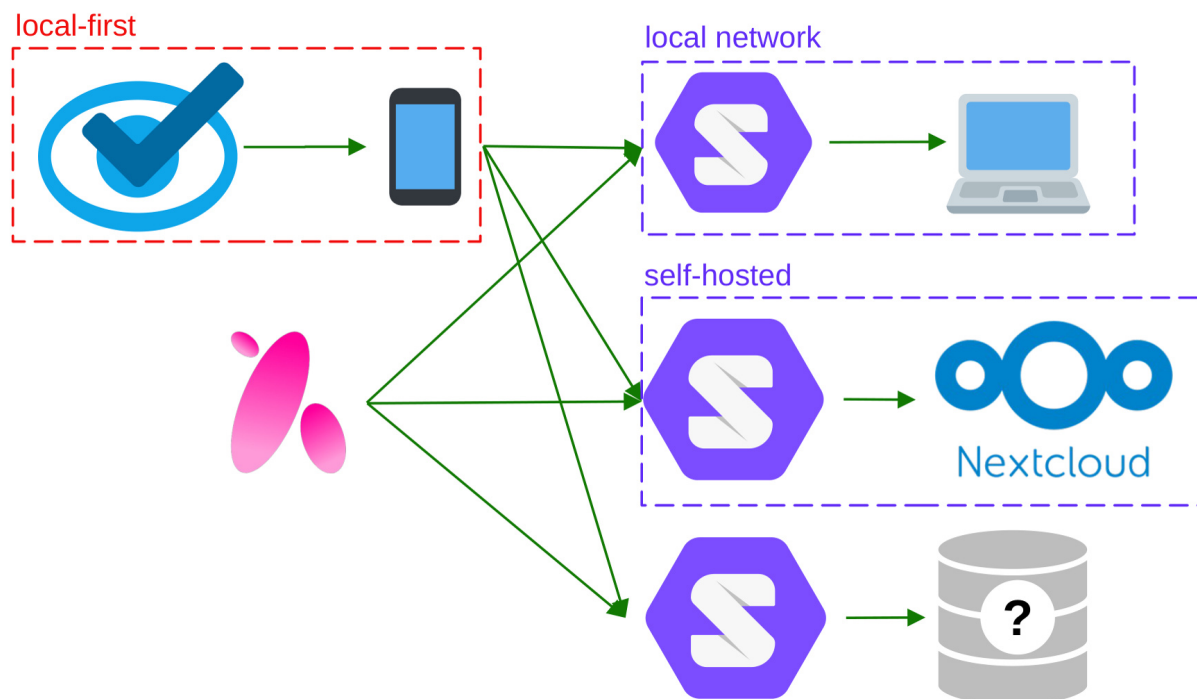
- **Learn Chinese** (open)

Your "Install Ànima" task is still completed, so no action needed there. Want me to help break down the Chinese learning goal into smaller steps, or set any details for it?

Ànima answers using personal data · Frame 00:27:18

Now we have two tasks. I can ask it to look again: 'Can you take a look? Do we have any new tasks?' It should read the data again and find the new task. Yes, now you have 'Learn Chinese,' and it is currently open. The idea is that you use an AI as you normally would, choosing whichever one you want, but it is connected to your Solid Pod and has access to your data.

Privacy conditions and a recap of the data flow



Local-first, local networks and self-hosting · Frame 00:28:48

In this case, I was using Anthropic, a third-party model. But if you use only local models, or models you control, everything is completely private. In fact, this can work with the internet disconnected, because everything can run locally. That's the basic idea of how Ànima and Solid applications work. Let me recap what we just did. First, we started using Solid Focus. You can use it entirely on your own device: it's local-first.

You don't even need to create an account. If you want to use my apps that way, you can. At some point, though, you will probably want to synchronize the data or store it somewhere. In this case, we used the local network: everything happened on localhost, and the data was stored in the pod on my laptop. Then we opened Ànima, which could communicate with this pod. Here, the pod was also provided by Ànima, but it could have been any other pod.

Storage choices and a Nextcloud-backed pod

00:28:18 · Noel De Martin

Ànima can connect to any Solid Pod. In my own production setup, for example, I have a self-hosted pod that stores data in Nextcloud. That's how I work with my data. Since both are Solid Pods, all my applications—Ànima, Focus and the others—can connect to that pod too. In my production environment, I store data in Nextcloud. Because Solid is a protocol, you can choose whichever storage you want.

00:28:53 · Noel De Martin

Depending on the storage you want, an adapter may not exist yet, so you may need to implement your own. I have to say it is fairly easy: the Nextcloud adapter is something I implemented myself, and you can see my Solid Pod on GitHub. One good thing about Solid is that the API isn't very complicated. That's the basic idea of how this works. If you want to build an application or understand how my applications are built, I have a couple of libraries I've been working on.

Soukai: a local-first data layer

My Tools



Framework agnostic

TypeScript

Models
(Active Record)

Engines
(Solid POD, IndexedDB, ...)

Interoperability
(Type Indexes, Schemas, RDF aliases)

Local-first
(CRDTs)

noeldemartin.com/local-first-2026

Soukai: models and synchronization · Frame 00:30:50

00:29:31 · Noel De Martin

The first is SoukaiJS. It is framework-agnostic: a TypeScript library. I use it in Ànima on both the server and the frontend. It's pure TypeScript, so wherever JavaScript works, this library can work, with no other requirements. Because it's written in TypeScript, it has full type definitions. It uses the Active Record pattern.

00:30:01 · Noel De Martin

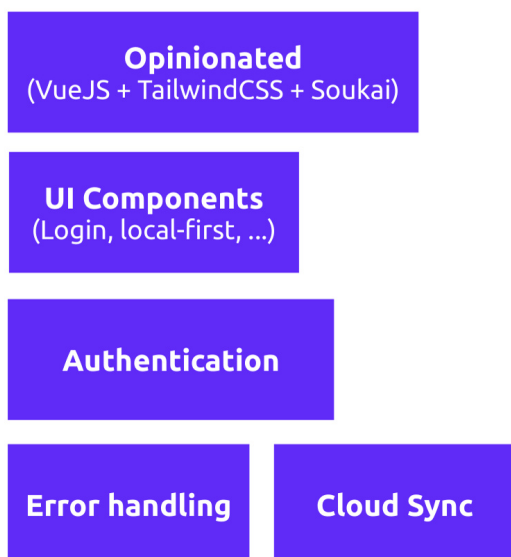
You may already know this if you have used Ruby on Rails or Laravel. It's a programming pattern for working with data. One thing the pattern gives you is the ability to swap the engine. My applications use IndexedDB by default, so anything you write in the application is stored there. If you want to synchronize with a pod, you can use a different engine, and it can happen automatically.

00:30:33 · Noel De Martin

Your code doesn't have to know whether the data is being stored in the pod or in IndexedDB. That's the magic of Active Record. It also comes with interoperability features for working with Solid. As I mentioned, it is local-first, so CRDTs are built in as well; you don't have to implement them again. Another tool I'm working on is Aerogel. This is much more experimental and less finished, but I'm already using it for all my apps.

Aerogel: an application framework

My Tools



Aerogel

The Lightest Solid

Work in Progress!

aerogel.js.org

noeldemartin.com/local-first-2026

The Aerogel application framework · Frame 00:31:57

00:31:10 · Noel De Martin

Aerogel is a full framework for making applications. Soukai is only the data layer; Aerogel builds the whole app. It is opinionated: it uses Vue.js, Tailwind CSS and Soukai for the data. It comes with many UI components. If you want to log in with Solid and have the interface for synchronizing data, that's already done, and you don't have to implement it again. Authentication is included, as is some error handling.

00:31:41 · Noel De Martin

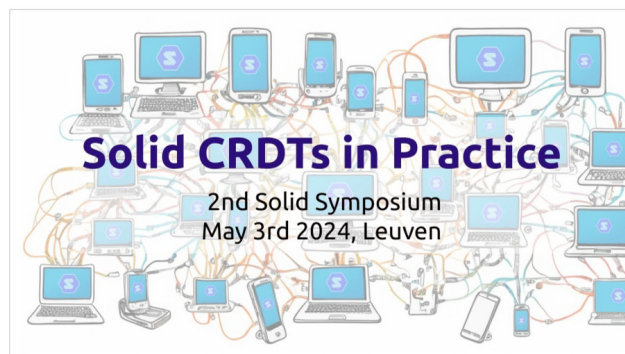
Some errors are very specific to Solid. The framework handles those issues, shows the appropriate response to the user or deals with them accordingly. Cloud synchronization is also implemented. If you're working with Aerogel, you don't have to think about synchronizing data: that mechanism happens automatically, and you can configure it. If you want to learn more about these tools, I have given a couple of talks in the past.

Further talks and development journal

My Tools



noeldemartin.com/solid-symposium-dx



noeldemartin.com/solid-symposium-crdts

noeldemartin.com/local-first-2026

Developer experience and CRDT resources · Frame 00:32:10

00:32:12 · Noel De Martin

You can watch them to dig deeper. One is about the developer experience of making Solid apps. The other is about CRDTs: how they are implemented in this library and how I use them with Solid. That's basically it. You can follow me online. As I mentioned, on my website I talk about what I'm working on. You can follow my development journal to keep up to date with all of this. That's it. Thank you.

See you online!



Follow my work at
noeldemartin.com/now

You can also find me on the fediverse,
Bluesky, Twitter/X, etc.

noeldemartin.com/local-first-2026

Following Noel's work · Frame 00:32:28

00:32:46 · Noel De Martin

I'll stop sharing for now.

00:32:50 · Noel De Martin

Thank you so much.

Questions and answers

00:32:53 · Host

Yes. Are there any questions? Would anyone like to ask something?

00:33:06 · Noel De Martin

I hope it was clear. I don't know if I went too quickly.

00:33:11 · Host

Yes, very clear, I think.

Does Solid provide synchronization?

00:33:15 · Audience

Hi, can I ask a question? As I understand it, you're building a local-first application and using Solid as the coordination and storage layer. Is that right?

00:33:34 · Noel De Martin

Yes. They are local-first, so everything lives in the browser. They use Solid to synchronize the data. I implemented a CRDT algorithm that works with Solid. If you want to learn more, I gave a talk about it; it's the one I listed at the end.

00:33:52 · Audience

Does the Solid protocol provide real-time synchronization between your client and the pod server?

00:34:11 · Noel De Martin

Solid does not provide synchronization out of the box. As I mentioned, it is like a database or Dropbox: you have a collection of files. I built this CRDT on top of Solid. Solid itself lets you read and write data, but it is generic data. It's like having a database with SQL: you can build CRDTs on top of SQL, but you can also build a normal application that isn't local-first.

00:34:43 · Noel De Martin

In this case, Solid only acts as a database or file store. Synchronization is what I implemented in my libraries.

RDF data and CRDT operations in the same document

00:34:53 · Audience

In Solid, RDF describes the schema, or semantics, of the stored data. But with CRDTs, the data you transfer is more like fragments or delta operations. Does your RDF describe the CRDT, or does it describe the schema of the data itself?

00:35:32 · Noel De Martin

I created an RDF vocabulary for CRDTs, so the CRDT is described in RDF. The Solid Pod knows nothing about CRDTs; it only sees that I'm writing RDF to it. I tried to do this so that another application could still use the same data even if it wasn't local-first. One problem with CRDTs or custom vocabularies is that, if you make things too custom, other applications won't understand the data unless they also use the same library.

00:36:16 · Noel De Martin

The way I implemented it is that the basic data remains plain RDF. If you create a task, it's a schema.org Action, for example. On top of that, the application creates the CRDT operations. Let me share the screen again and show you the task. If we open this task, as you can see, this is RDF. The basic data is a schema.org Action.

```
1 <#it> a <https://schema.org/Action>, <http://www.w3.org/2002/12/cal/ical#Vtodo>.  
2 <http://www.w3.org/2002/12/cal/ical#summary> "Install Anima".  
3 <https://schema.org/ActionStatus> <https://schema.org/CompletedActionStatus>.  
4 <https://schema.org/name> "Install Anima".  
5 <http://www.w3.org/2002/12/cal/ical#completed> "2026-10-02T12:30:27.425Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
6 <#it-metadata> a <https://vocab.noeldemartin.com/crdt/Metadata>.  
7 <https://vocab.noeldemartin.com/crdt/createdAt> "2026-10-02T12:28:07.638Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
8 <https://vocab.noeldemartin.com/crdt/resource> <#it>.  
9 <https://vocab.noeldemartin.com/crdt/updatedAt> "2026-10-02T12:30:27.436Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
10 <#it-operation-2678f454-637e-4ebe-81e9-421c1ffc2619> a <https://vocab.noeldemartin.com/crdt/SetPropertyOperation>.  
11 <https://vocab.noeldemartin.com/crdt/resource> <#it>.  
12 <https://vocab.noeldemartin.com/crdt/date> "2026-10-02T12:28:07.638Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
13 <https://vocab.noeldemartin.com/crdt/property> <https://schema.org/name>.  
14 <https://vocab.noeldemartin.com/crdt/value> "Install Anima".  
15 <#it-operation-666e4d6f-d676-4385-a7eb-a1aaf013501> a <https://vocab.noeldemartin.com/crdt/SetPropertyOperation>.  
16 <https://vocab.noeldemartin.com/crdt/resource> <#it>.  
17 <https://vocab.noeldemartin.com/crdt/date> "2026-10-02T12:28:07.638Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
18 <https://vocab.noeldemartin.com/crdt/property> <http://www.w3.org/2002/12/cal/ical#summary>.  
19 <https://vocab.noeldemartin.com/crdt/value> "Install Anima".  
20 <#it-operation-cel3545e-c5b4-40ba-9d41-42793b7cb6d0> a <https://vocab.noeldemartin.com/crdt/SetPropertyOperation>.  
21 <https://vocab.noeldemartin.com/crdt/resource> <#it>.  
22 <https://vocab.noeldemartin.com/crdt/date> "2026-10-02T12:28:07.638Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
23 <https://vocab.noeldemartin.com/crdt/property> <https://schema.org/actionStatus>.  
24 <https://vocab.noeldemartin.com/crdt/value> <https://schema.org/PotentialActionStatus>.  
25 <#it-operation-0515ee6b-750f-4620-9d69-f97fb6684d96> a <https://vocab.noeldemartin.com/crdt/SetPropertyOperation>.  
26 <https://vocab.noeldemartin.com/crdt/resource> <#it>.  
27 <https://vocab.noeldemartin.com/crdt/date> "2026-10-02T12:30:27.436Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
28 <https://vocab.noeldemartin.com/crdt/property> <http://www.w3.org/2002/12/cal/ical#completed>.  
29 <https://vocab.noeldemartin.com/crdt/value> "2026-10-02T12:30:27.425Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
30 <#it-operation-e02f19db-5ece-4201-b025-c41df4d2a53e> a <https://vocab.noeldemartin.com/crdt/SetPropertyOperation>.  
31 <https://vocab.noeldemartin.com/crdt/resource> <#it>.  
32 <https://vocab.noeldemartin.com/crdt/date> "2026-10-02T12:30:27.436Z"^^<http://www.w3.org/2001/XMLSchema#dateTime>.  
33 <https://vocab.noeldemartin.com/crdt/property> <https://schema.org/actionStatus>.  
34 <https://vocab.noeldemartin.com/crdt/value> <https://schema.org/CompletedActionStatus>.
```

Task state and CRDT operations in an RDF file · Frame 00:37:00

00:36:46 · Noel De Martin

This is something any Solid application should understand. It doesn't have to know anything about CRDTs. Here we also have the operations, using the CRDT vocabulary I created. If you implement a task manager using Solid, you are very likely to use schema.org, because it is one of the most popular ontologies and people already know it. You should be able to understand this data even if you know nothing about what my application is doing, or about CRDTs.

00:37:22 · Noel De Martin

That's how I implemented it so it can interoperate with other applications. The CRDT operations are written into the document as well. If you want more details, you can watch this presentation, Solid CRDTs in Practice, where I explain how it all works.

00:37:48 · Audience

Wow, that's amazing.

00:37:51 · Noel De Martin

Thank you. Any more questions, opinions or anything else?

Why protocol stability can be a good thing

00:38:01 · Host

Can I ask a question about Solid? Solid has been around and in development for a long time, maybe nearly a decade. I imagine there are many interesting ideas and features behind it. What do you personally enjoy most about building apps on top of Solid?

00:38:32 · Noel De Martin

Yes, Solid has been around for a while. If you look at its history and how it was created, it wasn't something that happened overnight. It's the evolution of the web according to Tim Berners-Lee. First he created the web. Then he created semantic data, which works with the web and follows the same idea. Then he created Solid as a way to exchange, edit and store semantic data.

00:39:04 · Noel De Martin

The boundary of when Solid was created is therefore a little blurry; it isn't black and white. At the same time, I would say Solid is very stable. To be honest, that was a little disappointing for me at first. When I started using it, my first impression was that Tim Berners-Lee had created a new protocol to do this storage-related work.

00:39:35 · Noel De Martin

Because it was new, I thought it would improve a lot, change and add new features. In reality, during all the time I've worked with Solid, almost nothing new has happened at the protocol level. There have been many new projects, people and libraries, but the Solid protocol itself hasn't evolved very much. At first that was disappointing, but now I think it's a good thing, because I think of Solid as being like RSS.

00:40:09 · Noel De Martin

The specification for RSS feeds hasn't changed in many years. That's also a good thing, because it's why many applications work with RSS. Now we have podcast applications, and everybody is doing podcasts, but everything still works with the basic RSS specification. The way I see Solid is that it is a protocol that is still evolving.

00:40:40 · Noel De Martin

Some things may still change, but I don't think the basic ideas will change very much. The basic concept is that Solid is a store, like Dropbox or Nextcloud, except the files are RDF. So it's not just files; it's semantic data. That's the cornerstone of Solid, and I don't think the protocol itself is going to change too much.

00:41:11 · Host

Thank you so much. I see a question in the chat box.

00:41:22 · Noel De Martin

Okay. I think only you can see it, because I can't see it.

Why I started: applications that disappeared

00:41:27 · Host

The question is: how did you first get into Solid, and what made you want to start building on it?

00:41:41 · Noel De Martin

That's interesting, because if you want to know what I thought of Solid at the beginning, you can read it on my website. I've been talking about it in my development journal ever since I started learning Solid, so you can go back through the journal. The basic idea is that I had started working on something similar, which I called autonomous data. I was very annoyed with applications that stopped working. I had been using Wunderlist, a task manager that was acquired by Microsoft.

00:42:19 · Noel De Martin

I also used Evernote—specifically Evernote Food, a layer on top of Evernote for managing recipes. Both applications were discontinued. I was very unhappy, because I was fine with the apps: I didn't need any new features or want anything to change. I was fed up with applications disappearing even when I didn't need anything new. Following that thought, I started working on what I called autonomous data.

00:42:54 · Noel De Martin

While working on that, I came across Solid. It was very similar to what I was doing, so I decided to stop working on my own idea and follow Tim Berners-Lee's idea, which was already close to what I wanted to achieve. The main drawback I found was that the developer experience wasn't very good. That's why I ended up implementing libraries such as Soukai and Aerogel: I wasn't very happy with the libraries available at the time.

00:43:30 · Noel De Martin

There are more libraries to choose from now, but I still use mine because they give me the developer experience I like. That's how I started: I wanted to solve the problem of applications disappearing out of nowhere.

00:43:51 · Host

Nice. Thank you so much. Any other questions from the audience? If you think of something, please feel free to ask.

Focus, Umai and MediaKraken

00:44:19 · Noel De Martin

Maybe I can mention something I didn't explain, in case you're curious about the applications. One of them is Focus, the task manager. Of the other two, one is called Umai, a recipe manager. That's what I used to replace Evernote Food; the idea is to manage your cooking recipes. The other is called MediaKraken, a movie-tracking website. I keep my collection of movies there: everything I've watched or want to watch.

00:44:53 · Noel De Martin

I keep track of them there. I'm currently rewriting MediaKraken to support TV shows as well. If you try it today, you'll notice it isn't as smooth as the other two, because it's the older application I have right now and I still need to improve it. That's what the applications do, in case you're interested in trying one.

Closing thanks

00:45:32 · Host

Cool. Any other questions? If there are no other questions, thank you, Noel.

00:45:53 · Noel De Martin

This was great. Thank you for organizing, and thank you everyone for attending.

Noel De Martin: <https://noeldemartin.com/> · Slides: <https://noeldemartin.com/local-first-2026/> · Solid: <https://solidproject.org/>