byoa (bring your own algorithm) NFTs

Putting apps inside NFTs to control and enhance your UX in web3 apps and the metaverse

August 11, 2021

@RyanBerckmans
@TheCryptoPeter
@mallowsxyz

Abstract

The metaverse is first and foremost about interoperability and personal property rights and freedoms. Web2 apps have a vertical integration and black box around data and user experience. The move to web3 offers an opportunity to build out enhanced data and UX ownership for users.

We explore, propose, and implement byoa (bring your own algorithm), an ecosystem to help maximize interoperability and personal property rights and freedoms in a cross-platform and multi-substrate manner.

byoa is an open ecosystem of apps inside NFTs to control and enhance your UX in web3 apps and the metaverse. byoa NFTs run using the byoa-sdk, a credibly neutral integration point for the byoa ecosystem. Web3 apps opt into byoa by choosing to integrate byoa-sdk.

The potential for byoa NFTs range from accessibility, mental wellness, DeFi opportunities, and more.

Definitions

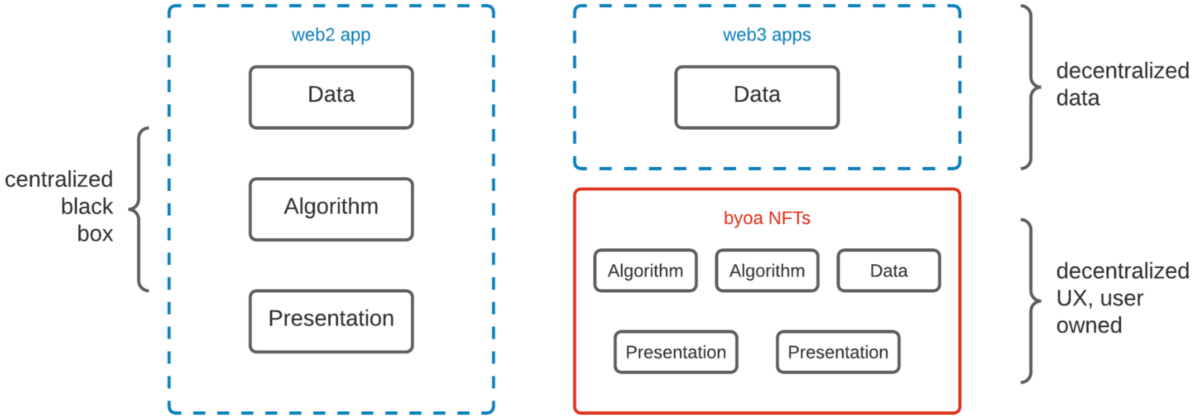
- **metaverse** - a unification of substrates that are focused on interoperability and personal property rights and freedoms. Starting with web3 apps.
- **byoa** - bring your own algorithm, an umbrella term that we use to describe the practice of packaging apps inside of NFTs that, to start, run alongside and inside web3 apps. We envision byoa as being a sort of user-directed operating system inside any app.
- **byoa NFT** - an NFT that contains (inside the NFT's immutable metadata) an algorithm and/or data which helps a user to control and enhance their UX.
- **byoa algorithm** - an algorithm packaged inside a byoa NFT.
- **byoa ecosystem** - the broader ecosystem of byoa tooling, byoa NFTs, decentralized app stores for byoa NFTs, byoa algorithms, the byoa SDK itself, and anything else related to byoa.
- **host web3 app** - the web3 app that runs the byoa sdk which then runs a user's byoa NFTs. For example, if you were using byoa sdk on Uniswap with a byoa NFT that's a custom Uniswap price chart, in that case the host web3 app would be Uniswap.

Motivation

Consider a web app with data providers, data manipulators, and data presenters. In web2, this is largely all closed and black-box. For example, as a user of facebook or twitter, you play an important role providing inputs into their system, but you can't bring your own inputs or functionality. In web2, the property owner decides your UX for you.

Web3 is usually decentralized, open, and not a black-box. Yet today, web3 has only a nascent ability for users to bring their own data providers, manipulators, and presenters.

Unlike web2, web3 (and we think, the metaverse) is philosophically rooted in interoperability and personal property rights and freedoms. In our view, these roots have cultivated a natural trend of your crypto wallet's non-custodial ownership expanding to include an increasingly diverse array of digital property. We see byoa as part of this trend, and its core contribution is to add non-custodial "installed apps" to your crypto wallet.



web3 byoa NFTs are decentralized, owned by the user, composable, and run on multiple substrates vs. web2 black-box experiences

How byoa works

byoa (bring your own algorithm) is an open ecosystem of apps inside NFTs that are used to control and enhance your UX in web3 apps and the metaverse.

As a user grows their collection of byoa NFTs in their crypto wallet, they will bring an increasingly personalized UX with them to any web3 app or place in the metaverse.

By themselves, byoa NFTs have no relationship to the web browser, a specific crypto wallet, or any particular web3 app. Instead, byoa NFTs need a specific integration point with the host web3 app. The integration point we've built is the new byoa-sdk, which acts as a userland operating system to run byoa NFTs.

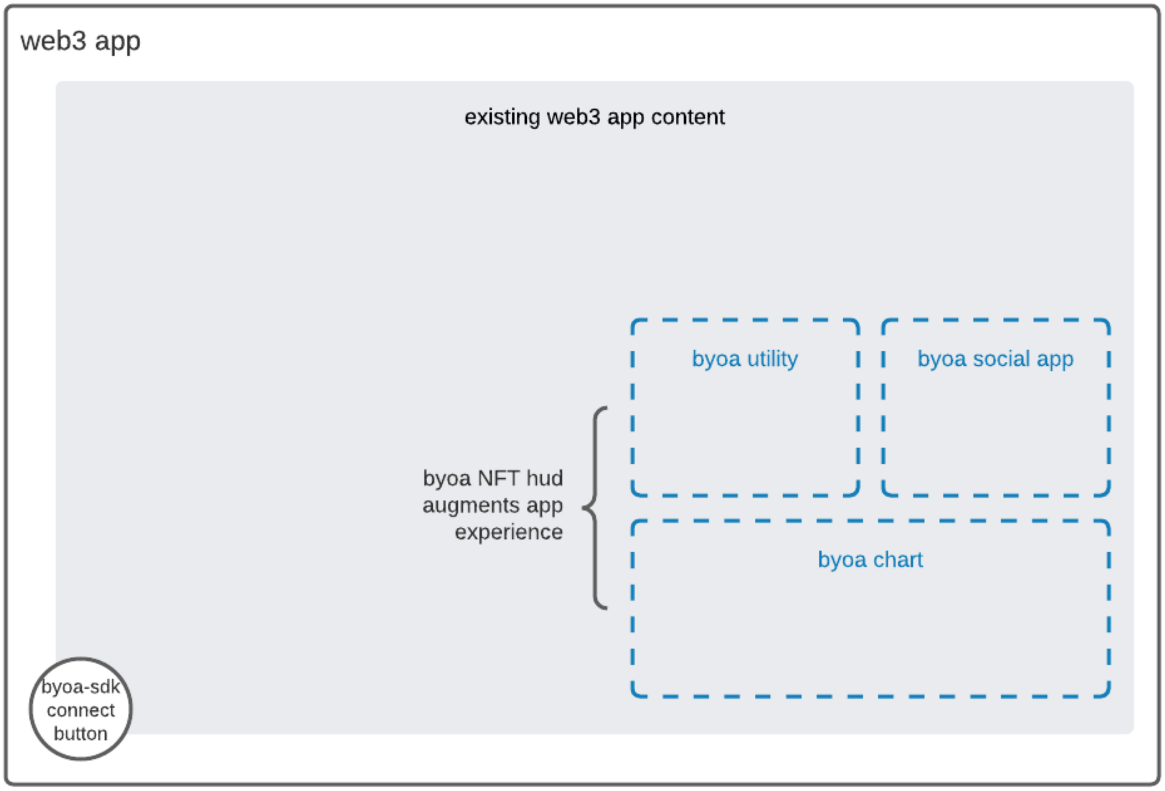
We're proposing that every web3 app should (and eventually, will want to) integrate byoa-sdk, which means it's important that byoa-sdk is a public good, ie. credibly neutral and non-monopolistic.

The high-level plan for byoa is to grow the ecosystem and multi-sided market of byoa NFTs, web3 apps that integrate byoa-sdk, and decentralized app stores that coordinate the minting (ie. installing) of byoa NFTs into user wallets.

A look into the future of byoa

In the future, as the byoa SDK grows into a mature userland operating system with a rich ecosystem of byoa NFTs, we might imagine that a user's personal interface, like their phone's home screen, may be entirely configured and run via their byoa NFTs in their crypto wallet.

Here is a concept for what that future end-state might look like:

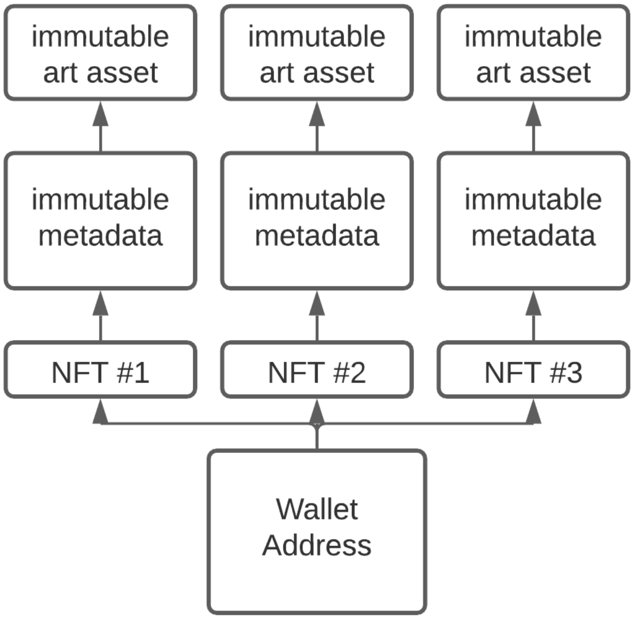


Applying the byoa-sdk to a web3 app enables a user with byoa NFTs to bring their own experiences (data, processing, and presentation) throughout web3 and the metaverse in a consistent and interoperable manner.

How byoa fits into today's common NFT architecture

Here's how NFTs tend to work today:

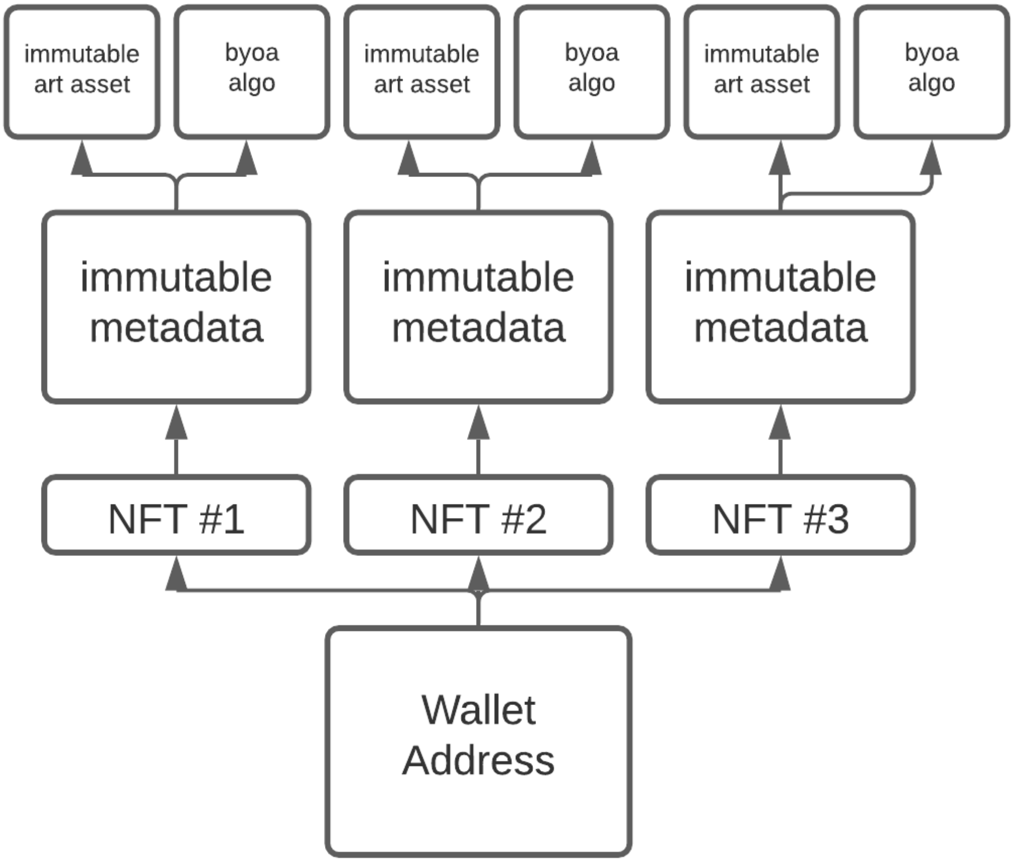
An ERC-721 smart contract stores a pointer to an immutable metadata structure stored in a decentralized file system like IPFS. That immutable metadata has attributes referencing more immutable and decentralized data, such as an image asset.



Architecture for a typical NFT

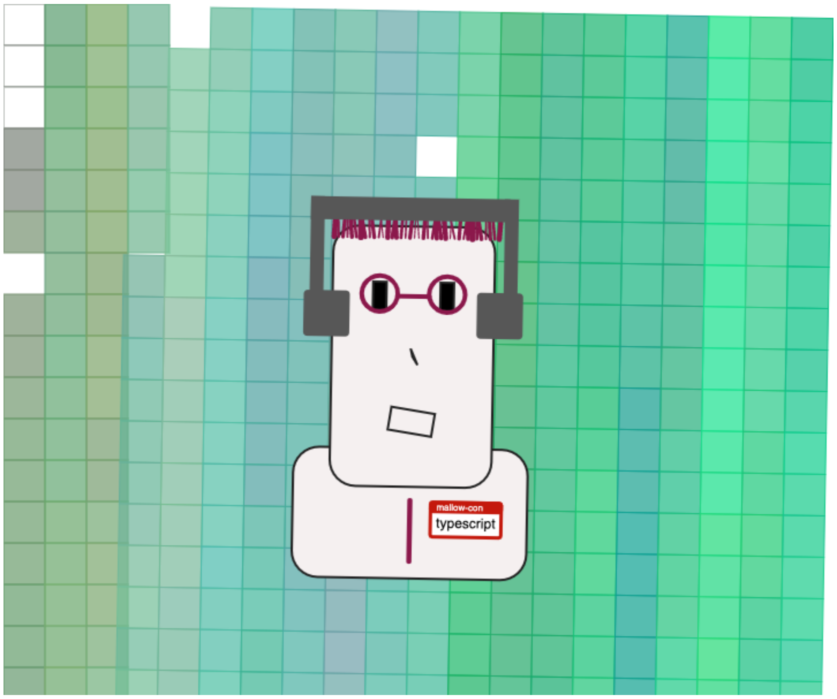
And here's how byoa NFTs work:

byoa NFTs are regular NFTs. But, their metadata additionally points to an immutable and decentralized algorithm.



Architecture for a byoa NFT

We wanted to develop a proof of concept of byoa and came up with Mallows, an avatar/profile-photo NFT collection of toy collectibles. Each Mallow has one byoa algorithm inside of it and may be loaded into any byoa-sdk-compatible web3 app.



The Mallows NFT project includes generative art with different traits for each of the Mallows (nerdy marshmallows from the mallow-verse). Each mallow also has one byoa algorithm stored with their NFT.

Mallows architecture deep dive

Why Mallows?
We wanted a proof of concept for byoa, and thought that an NFT avatar toy collection would be a fun way to start.

Mallows are toy collectibles and the first byoa NFTs. Each mallow has one byoa algorithm inside of it, most algorithms are inspired by crypto culture and current events.

Mallows architecture
Mallows uses a standard ERC-721 NFT contract that associates each NFT tokenId with off-chain, content-addressed (and thus immutable) token metadata.

To satisfy making Mallows a proof of concept of byoa NFTs, each tokenId's metadata includes new fields that point to off-chain, content-addressed algorithm code.

Although byoa algorithms are content-addressed and immutable, any byoa algorithm may choose to include centralized dependencies. For example, one of our byoa demos makes a run-time network connection to DeepAI, a centralized SaaS. In general, a byoa NFT that has a centralized dependency is known as a semi-centralized byoa NFT.

Mallows run-time execution
As an initial setup, a host web3 app, such as Uniswap, integrates the byoa-sdk.

To help bootstrap the availability of DeFi app UIs that support byoa, we plan to offer forks of popular DeFi apps with the only modification being the addition of byoa-sdk. We may also offer a browser extension that only loads the byoa-sdk into any page.

User journey (demo)

1. arrive on host web3 app page, eg. Uniswap
2. connect wallet --> byoa-sdk finds and loads the user's byoa NFTs
3. byoa-sdk draws the byoa control widget on the page
4. user clicks the byoa control and chooses a byoa NFT to activate (in future, byoa NFTs whitelisted by the user may auto-activate)
5. byoa-sdk loads the selected byoa NFT's algorithm and executes it, modifying the user experience

Examples of Mallows toy byoa algorithms
Each mallow toy collectible has one algorithm inside of it. Here are some of the algorithms:

1. a real-time flickering flames visualization of EIP-1559 fee burn.

2. an NFT profile viewer that auto-loads profile photos from your NFT collection, displaying them in a standard floating window provided by the byoa-sdk.

3. an "accessibility hub" with prototype controls to adjust colors for color blindness, activate night mode, or apply a custom CSS theme to the page.

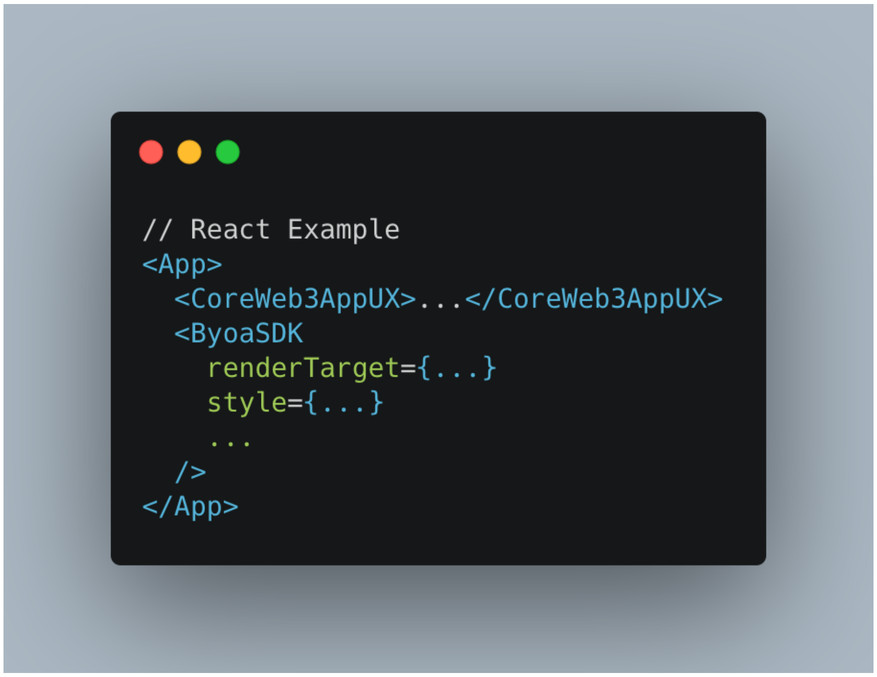
byoa-sdk v1 architecture

The v1 architecture defines an initial implementation for all decentralized byoa algorithms. It is designed with the knowledge that it will be improved with future versions while maintaining backwards compatibility.

The byoa-sdk establishes a wallet connect button which a user can click in order to access byoa functionality. It fetches a user’s NFTs from a whitelisted set of byoa NFTs, currently Mallows. Future work would include considering adopting a byoa standard for solidity ERC721 contracts, such as ERC721Algorithmable and then no whitelisting would be needed.

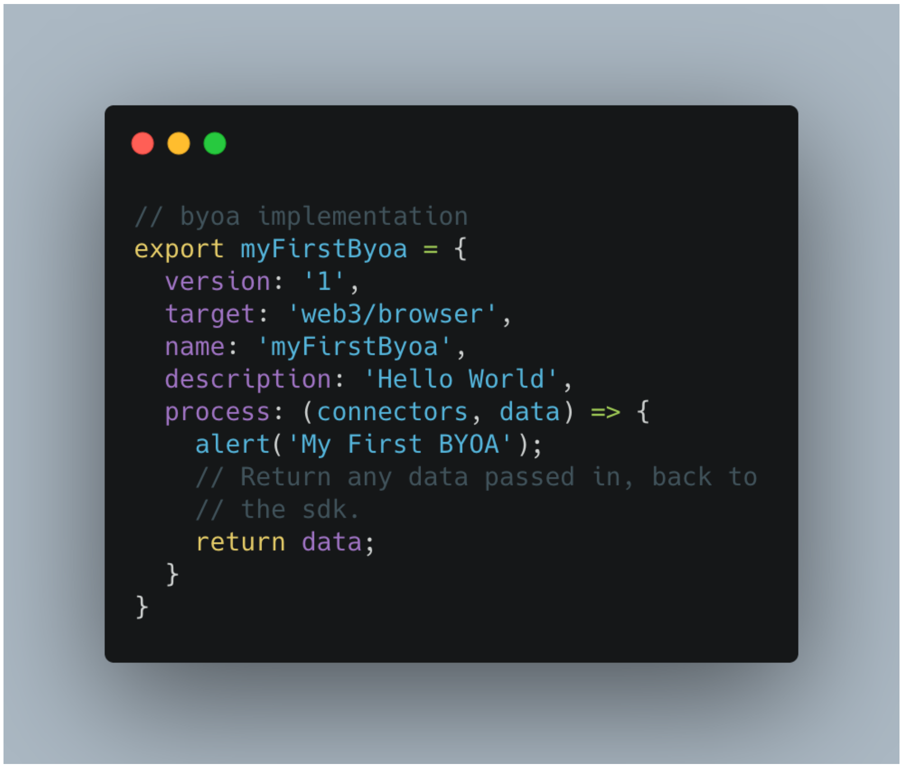
The byoa-sdk establishes a dedicated HUD overlaid in the DOM of the host web3 app of which the byoa algorithm has access to render into while passing through all click events. The byoa-sdk also establishes a suite of connectors which it makes accessible as parameters to the byoa algorithm. Such connectors include jsonRPCProviders for ethereum network calls.

The byoa-sdk then facilitates the users' use of the byoa NFTs they have brought with them to the host web3 app.



```
// React Example
<App>
  <CoreWeb3AppUX>...</CoreWeb3AppUX>
  <ByoaSDK
    renderTarget={...}
    style={...}
    ...
  />
</App>
```

A web3 app includes the byoa-sdk library of choice in their application.
Above is pseudocode of a React example.



```
// byoa implementation
export myFirstByoa = {
  version: '1',
  target: 'web3/browser',
  name: 'myFirstByoa',
  description: 'Hello World',
  process: (connectors, data) => {
    alert('My First BYOA');
    // Return any data passed in, back to
    // the sdk.
    return data;
  }
}
```

Above is pseudocode of a byoa algorithm living in decentralized immutable storage linked to by a user's NFT. The above will be consumed by the byoa-sdk when a user connects their wallet on a byoa-sdk enabled web3 property.

Towards a steady-state byoa SDK

Cross-platform and multi-substrate
This whitepaper focuses on web browsers as an initial primary substrate for the byoa SDK and ecosystem.

At steady state, the byoa SDK would be expanded to multiple substrates, such as VR/AR headset and phone operating systems.

As well, byoa NFTs are initially written in JavaScript and TypeScript, but may be expanded to support multiple substrates. A single byoa NFT instance may support multiple substrates, similar to a universal binary. Future substrates may include platforms like wasm, Unity, or iOS. Ie., the current byoa SDK may be thought of as the first byoa interpreter that's focused on JavaScript and web3.

Security
byoa NFTs have security concerns similar to browser plugins and mobile apps.

A malicious byoa NFT is like a malicious mobile app. The byoa ecosystem must facilitate security features to identify malicious or broken apps and protect users.

For example, a malicious byoa NFT running in a host web3 app's JavaScript main thread may craft a malicious transaction and trick a user into signing it.

There may also be malicious web3 apps that may attack byoa NFTs. byoa algorithms may contain secrets, such as API keys or cookies. A malicious web3 app could steal these data if the byoa algorithm isn't run in a secure sandbox, such as in a content-addressed iframe using a subdomain-based IPFS gateway.

For a steady-state byoa SDK, potential security features include

- Token Lists for byoa NFTs.
- asking the user to whitelist a byoa NFT for it to run automatically. This whitelist could be tied to the browser session, minted into a “whitelist NFT” to reside in the wallet, etc.
- requiring the user to sign a transaction to "take receipt" of a byoa NFT. For example, if the user was gifted a malicious byoa NFT, it may be ignored by the byoa SDK until the user reviewed their alerts/inbox and signed a transaction to accept the new byoa NFT.
- sandboxing byoa NFT algorithms into iframes. Note that use of an iframe also allows the byoa algorithm to protect itself from the host web3 app.
- sandboxing byoa NFT algorithms into wasm VMs. Figma eventually landed on this solution.
- helpers to coordinate decentralized audits, ratings systems, and developer reputation scores.

byoa SDK standard library

We think of the byoa SDK as a userland operating system that may provide a feature-rich standard library to any byoa NFT.

For a steady-state byoa SDK, potential standard library components include

- NFT avatar utilities
- ethrpc utilities
- L2 and cross-chain utilities
- byoa UI components, such as a windowing system with floating, collapsible windows
- a domain-specific language for web3 apps to declare the data and hooks for byoa
- infrastructure for decentralized and semi-centralized messaging. For example, a byoa NFT reader for messages sent to your crypto wallet. Or, an embedded chat room.
- payment utilities
- a package manager for dependencies between byoa NFTs

Growing the byoa ecosystem

We see the byoa ecosystem as being a multi-sided market of at least (i) byoa NFTs, (ii) web3 apps that integrate byoa-sdk, and (iii) decentralized app stores that coordinate the minting (ie. installing) of byoa NFTs into user wallets.

The byoa ecosystem is highly reliant on the byoa-sdk as the integration point for users to run their byoa NFTs in host web3 apps. For that reason, it seems important that byoa-sdk be an open, credibly neutral, non-monopolistic public good.

To help bootstrap web3 apps supporting byoa-sdk, we plan to launch forks of popular DeFi app UIs with the only modification being the addition of byoa-sdk. We may also offer a browser extension whose only job it is to load the byoa-sdk into any page.

We'll also engage in these byoa growth activities:

- foster creator programs and materials to help others build byoa NFTs
- attract contributors to byoa-sdk
- investigate EIP standards tracks for byoa
- facilitate minting byoa NFTs on L2s
- develop real-world byoa NFTs that are useful to the community

byoa vs. browser plugins

What are some of the differences between byoa vs. traditional browser plugins, such as the Google Chrome Web Store plugins?

Differences include

1. byoa is intended to be cross-platform and multi-substrate. For example, byoa-sdk runs anywhere that JavaScript runs vs. browser plugins only run in browsers.

2. byoa promotes creator ownership. The byoa ecosystem has a decentralized marketplace of app stores to buy, trade, and mint byoa NFTs. In contrast, a browser plugin store is a walled garden where the megacorp must approve the app and may restrict app functionality and commercial terms.

3. byoa promotes user ownership. You own your byoa NFTs, whereas, for example, Google may take down a Chrome browser plugin that you're enjoying.

byoa vs. video game modding communities

What may be the differences between byoa vs. a video game modding community?

The byoa-sdk sort of unlocks a modding community for any web3 app. But, there seems to be some differences between the byoa ecosystem and video game modding communities:

Typically, a video game modding community is siloed to a single game because they are dedicated players of that game (unlike crypto people using many web3 apps), and a game modding platform is usually built specifically for that one game.

In contrast, any web3 app may integrate the byoa SDK to gain access to the marketplace of byoa NFTs, which is a bit like a generalized modding framework for any video game. Note that when a host web3 app integrates byoa, it may specify its own preferences for that integration, such as which structured data or hooks it exposes to byoa NFTs.

byoa vs. mobile app store apps

How might we compare byoa vs. Apple and Google's mobile app stores?

A web3 app that facilitates minting or installing byoa NFTs into a user's wallet may be thought of as a type of decentralized or semi-centralized app store.

Traditional app stores offer utility such as app security reviews, 5-star ratings, and general merchandising around top apps, categories of apps, etc. In the byoa ecosystem, these benefits of traditional app stores may be substantially replicated in a decentralized and/or semi-centralized way.

The main drawback of traditional app stores is that they are walled gardens that choose to limit certain app functionality or commercial terms. These drawbacks may be wholly eliminated in a decentralized marketplace of byoa NFTs.

Traditional app stores are single-platform, with app-level frameworks such as react native gaining popularity to allow developers to build a cross-platform app with one codebase. In contrast, byoa NFTs are innately designed to become cross-platform and multi-substrate.

Another benefit of traditional app stores is that the apps are native mobile apps, which often confer a superior UX vs. web apps on mobile. We envision that byoa NFTs may grow to become cross-platform and include native app functionality, although it might require something like a crypto-native fork of Android to support byoa at the OS level.

Criticisms of byoa

Here are some criticisms of byoa that we came up with:

1. Security concerns broadly. Apple and Google's mobile app stores and Firefox and Google's browser plugin stores provide substantial security protections. It's incumbent on the byoa ecosystem to learn from the security successes of other app platforms, build these capabilities, and protect end-users.

2. It might be said that the EVM is already a byoa ecosystem, and that web3 doesn't need a new semi-off-chain, content-addressed byoa system. In our view, the distinctiveness of byoa is that it's a userland system that runs inside other apps, giving end-users control over their last-mile UX.

3. Why build an ecosystem around the idea that every web3 app should integrate byoa-sdk? Ie. isn't this an unrealistic expectation and/or undesirable outcome for an entire app ecosystem to depend on a single library that must be voluntarily integrated by every web3 app? In short, we think the benefits of a mature byoa ecosystem outweigh the cost of every web3 app needing to integrate byoa-sdk. For example, Apple's App Store and SDK makes iPhones more useful and has resulted in a world where people generally want many apps on their phones. We think that people will want to control and enhance their UX by keeping apps in their crypto wallet, and that's why web3 apps will integrate byoa-sdk, because users want it. As well, we recognize the importance that byoa-sdk must be open, credibly neutral, and non-monopolistic.

Design considerations for building new byoa NFTs

For creators of new byoa NFTs, here are some high-level design considerations that may be useful:

- **Algorithms and/or data?** Will the new byoa NFT contain algorithms, data, or both?

- **Public or private?** Is the new byoa NFT intended to execute in a public, semi-private, or secret context? For example, maybe the new byoa NFT contains API keys that should be hidden from the host web3 app?

- **Decentralized or semi-centralized?** Is the new byoa NFT fully decentralized or semi-centralized? For example, a byoa algorithm that depends on a centralized SaaS is semi-centralized, even if the centralized SaaS doesn’t require an API key.

- **General or specific?** Does the new byoa NFT run on any site, such as an accessibility helper, or is it designed to work only on a specific site, like a Uniswap price chart plugin?

Conclusion

To date, web2 apps have maintained vertical black box integration from data to user experience.

As decentralization increasingly enables interoperability and data ownership, many web3 apps have done well to open up data ownership within the apps. Yet, these apps still control much of the user experience.

byoa (bring your own algorithm) is an ecosystem founded on interoperability and personal data ownership that allows the user to control and enhance their UX in web3 apps and the metaverse, and does so in a permissionless, trustless, and interoperable way.

Appendix: byoa NFT examples

Here are some examples of what you may be able to do with byoa NFTs:

1. **byoa as a content distribution platform.** Rent a movie by minting an NFT that gives you the right to watch the movie as often as you want for the next two days.

2. **Limited runs of the byoa algorithm.** For example, pay $5 and get to run the algorithm inside the NFT N times - could be a song, movie, image filter, etc.

3. **Decentralized bookmarks.** For example, I might mint a byoa NFT that represents a bookmark for Compound Finance, and then that bookmark shows up on my personal web3 landing page.

4. **Mental wellness.** A semi-centralized byoa NFT that allows me to add trigger words or concepts, which then get automatically filtered out via recommendations from a centralized AI service. We gave a demo of a prototype.

5. **Content moderation.** Similar to mental wellness, a centralized AI service could be used in a semi-centralized byoa NFT to provide parental controls in any UX context.

6. **Accessibility**. A byoa NFT may be used to carry a user's accessibility preferences to any UX context. For example, a prototype "accessibility hub" is inside some Mallows byoa NFTs.

7. **Custom HUDs.** A host web3 app may expose data or hooks in its integration with byoa-sdk, and those data and hooks can power a user-defined HUD or windowing system. For example, a small floating HUD on Compound Finance that automatically appraises the user of competing lending rates.

8. **NFT preferences.** A byoa NFT whose only job it is to contain your preference of which of your other avatar NFTs should be used as your metaverse profile photo.

9. **Data usage rights.** A byoa NFT that informs the host app of your stated preferences for handling your data. Similar to a browser's "Do not track" setting, except more expressive and you take it with you in any UX context.

10. **Global, user-defined cookies.** For example, a byoa NFT that replicates certain web2 session cookies, encrypted, so that you may use them on other platforms.

11. **Embedded and decentralized social networks.** Lately, there's been a lot of talk about decentralized social media. We're excited for it. As well, we think it'd be cool to embed a social network in a byoa NFT, such that your current web3 app is your "location" in the social network or virtual world. For example, when you load the page, you see the avatars of other users who are currently visiting your app/page (adjusted for the other users' privacy and activity sharing preferences). Each avatar you see might be based on an "NFT preference" byoa NFT in the other users' wallets, which is an example of composability and dependencies among byoa NFTs.

12. **Custom embedded financial charts.** For example on Uniswap, a byoa NFT for a custom real-time price chart that auto-displays the current pool you're working with.

13. **Delivering centralized product functionality into an embedded byoa context.** In the previous example of "custom embedded financial charts", the byoa NFT contained the code to draw the chart. But, the ethereum community already has a good collection of charting and data science tools. Instead of bundling the code to draw the chart in the byoa NFT, we might use a byoa NFT to serve an embedded chart from an existing product.

14. **TradFi payment credentials.** A semi-centralized byoa NFT that's run by a compliant company. It uses the byoa-sdk to offer your credit card number to the host page in a compliant way. It's like Chrome storing your credit card numbers, except it's semi-centralized, cuts out the middle man (Google), and is more voluntary and lean because the user controls the decision to own that byoa NFT and the company controls the implementation of that byoa NFT in a direct customer relationship.

15. **Self-contained billing and payments.** The code to process a DeFi payment is a vertical slice of UI, off-chain processing, and smart contract code. We can embed all that code into a byoa NFT, so that the NFT itself would have everything it needs - including the UI - to process a payment in a fully autonomous and self-sovereign way. Note that the byoa algorithm may be executed by the user before they pay a fee to mint the NFT. For example, the user might execute a byoa algorithm to submit a payment and generate a zk-proof of that payment, and then the byoa NFT's contract consumes that proof as a required input into minting the NFT. An NFT that literally sells itself.

16. **NFT collection governance of immutable, off-chain logic.** Make a crypto punks DAO, where each 1 punk NFT gets 1 vote in governance. They might then vote to upgrade the canonical implementation of a punk 3D renderer that's used in the metaverse. For example, you'd package a 3D renderer into a byoa NFT, and then the byoa-sdk would offer that renderer to other byoa NFTs as the canonical punk 3D renderer. When you enter a new room in the metaverse, the 3D punks you see are based on their collective decision to appear that way.

17. **Micropayments.** The byoa-sdk may coordinate micropayments, as well as help humans to sign policies for automated micropayments within safety thresholds. For example, your browser might automatically pay a newspaper $0.10 for reading an article, without bothering you at all, because you signed a policy to agree to pay "$0.10 per article on Tier 4 American News Sites".

18. **Advertising.** A user could opt into advertising by minting an advertising byoa NFT. They'd then see ads in a window drawn by byoa-sdk, and get paid to view these ads. The user may express preferences over which ads they see, how often they see the ads, as well as which data from the host app or their wallet may be shared with the advertising byoa NFT. The user has control over their ad experience and keeps 100% of the money. Another potential benefit is that the advertising byoa implementation may be fully decentralized and content-addressed, and then an auditor could prove that the advertising code is private and will always be private.