Why do we have mini-apps?

- Hypothesis: because in China we have:
  - fragmentation of software platforms
  - dominance of individual apps

- To test this: let’s compare the US and Chinese markets

Why don’t we have mini-apps in the US?

Example of dominant US companies:

- Google:
  - started as a (set of) very popular website(s)
  - Ends-up a dominant app platform:
    - Chrome
    - Android
- Apple:
  - Started as a (set of) popular device(s) and Os
  - Ends-up a dominant app platform:
    - IOS app store
    - Safari

Why do we have mini-apps in China?

- Fragmentated app platforms:
  - Lots of browsers
  - Lots of app stores on lots of flavors of Android
  - Nobody dominates, except perhaps Apple.

But: dominant apps, like Wechat, Alipay, Taobao.

If you can’t dominate with an app store or a browser,
what do you do?

⇒ You build the app platform inside of your app.
⇒ Mini-apps.

Who, outside of China, may be interested in Standardization?

Apple and Google: probably not

Meta? Also a strong set
of apps without a platform(but they’re trying)

New AI entrants?

- Claude Artifacts
- Github Sparks(Microsoft)

Mini-apps and Web standards

- Current implementations with lots of non-standard features
- But, most features map to standard Web features
- See gap-analysis-api.md from Martin Alvarez-Espinar

Conclusion: mostly an Embedder(super-app,
aka MiniApp user agent) and Webview API thing

Doesn’t mean it cannot be standardized: see for example Browser user interface considerations (#nav-traversal-ui) in the HTML spec: “Browser user agents should provide the ability to…” (replace “browser” with “mini-app”).

Mini-apps and Servo

Servo is not a browser but only a Web engine; it does have an embedding API

We can try to build a “mini-app user agent” that would embed Servo

Mini-apps and Servo

Conceptually, we can follow the Web Application manifest spec: https://www.w3.org/TR/appmanifest/

Main difference is not tying the loading and applying of a manifest to a link in a document.

Mini-apps and Servo

I chose to follow the spec in this way:

- Processing the manifest without a document
- Ignore the “the user agent SHOULD ensure that at least at some point in the past” part(previous same origin doc with a link to a manifest).
- So I’m just loading a mini-app as a folder of assets locally, including a manifest, and processing this into a “mini-app”, which is then stored as a HTML file.
- One can imagine searching and loading from an app store.
- Launching the application immediately, with a file url pointing to the mini app HTML file.
- Doing this on Mac OS. No idea how this would work on IOS(can you store a file somewhere?).

Time for a demo.