# Node.js  Loaders Team Meeting 2024-07-02

## Links

* **Recording**: https://www.youtube.com/watch?v=dgqUAngs1zg
* **GitHub Issue**: https://github.com/nodejs/loaders/issues/207

## Present

* Loaders team: @nodejs/loaders
* Geoffrey Booth: @GeoffreyBooth
* Chengzhong Wu: @legendecas
* Guy Bedford: @GuyBedford
* James Sumners: @jsumners-nr
* Joyee Cheung: @JoyeeCheung
* Marco Ippolito: @marco-ippolito

## Agenda

## Announcements

*Extracted from **loaders-agenda** labeled issues and pull requests from the **nodejs org** prior to the meeting.

### nodejs/node

* Proposal for a simple, universal module loader hooks API to replace require() monkey-patching [#52219](https://github.com/nodejs/node/issues/52219)
* ESM Hook deadlocks since 21.2.0 a3e09b3 [#50948](https://github.com/nodejs/node/issues/50948)

### nodejs/loaders

* Proposal: Moving hooks on thread [#205](https://github.com/nodejs/loaders/pull/205)
* Hooks thread direction [#201](https://github.com/nodejs/loaders/issues/201)
* doc: add synchronous hooks proposal [#198](https://github.com/nodejs/loaders/pull/198)
  * Guy: tc39 module harmony proposals aim to improve ES modules. I am working on the JavaScript module source https://github.com/tc39/proposal-esm-phase-imports
  * Guy: to support a better worker instantiation workflow. Tools like bundlers can understand where workers can be instantiated. Will present the proposal to HTML WG soon. One problem is import-maps across threads, like `new Worker(source, { importMap: ‘inherit’ })`.
  * Guy: Trying to suggest setting some default to make `import source myModule from

‘./mod.js’; new Worker(myModule)` work.
  * Guy: A lot of invariance to maintain across toolings on resolution.
  * Guy: The module source declaration might advance later this year.
  * Guy: Being difficult to know who to contact in Node.js project, despite I am the delegate of OpenJS foundation to TC39
  * Geoffrey: What is possible to do register hook and put that into worker? People assumes worker to be isolated by default
  * Guy: Tools like vercel’s are hard to understand what loaders have registered.
  * Joyee: At least for Node.js the ship has shipped that you can do anything, like spawning a process. Browsers are supposed to run untrusted code. Node.js runs trusted code, that is the security model.
  * Geoffery: @joyee what’s your opinion on inheriting main thread’s loader settings?
  * Joyee: in the thread, people want it to be able to configure at library level and user level.
  * Geoffery: a user case is that TypeScript loaders. As an author of TypeScript library, a user register TypeScript loader at the entry, within the library, it can decide if it wants register in the worker or child process. What would the downside be the confiurability here.
  * Guy: I think loader authors won’t think about worker that much. On the API side, needs clarification on how the registration works.
  *

### Open Discussion

* TypeScript support, Marco and Matteo work on integrating swc into Node.js.
  * Geoffrey: Purely type stripping won’t work for enum/decorator/esm etc.; those are transforms. But we don’t want transformation, we should go with https://github.com/tc39/proposal-type-annotations
* Marco: there are tests on enum
* Geoffrey: they should not work or else it is not type stripping
* Chengzhong: concerned about the proprietary syntax of TypeScript being bound with Node.js major versions without TypeScript team buy-in
* Chengzhong: Champions of type-annotations proposal and the TypeScript team expressed interest in experimenting with annotations proposal in Node.js
* Joyee: can we go with undici’s
* Marco: swc already does that
* Joyee: ??
* Marco: I will only put efforts on this if this is agreed to be eventually in core. Otherwise I will not continue this
* Joyee: I am not against putting this in core. I think it should go with undici’s pattern to release a userland package first. The experiments might be hindered by the release-cycle of the core.
* Chengzhong: I’d be happy to set up a meeting to include type-annotations and TypeScript teams.