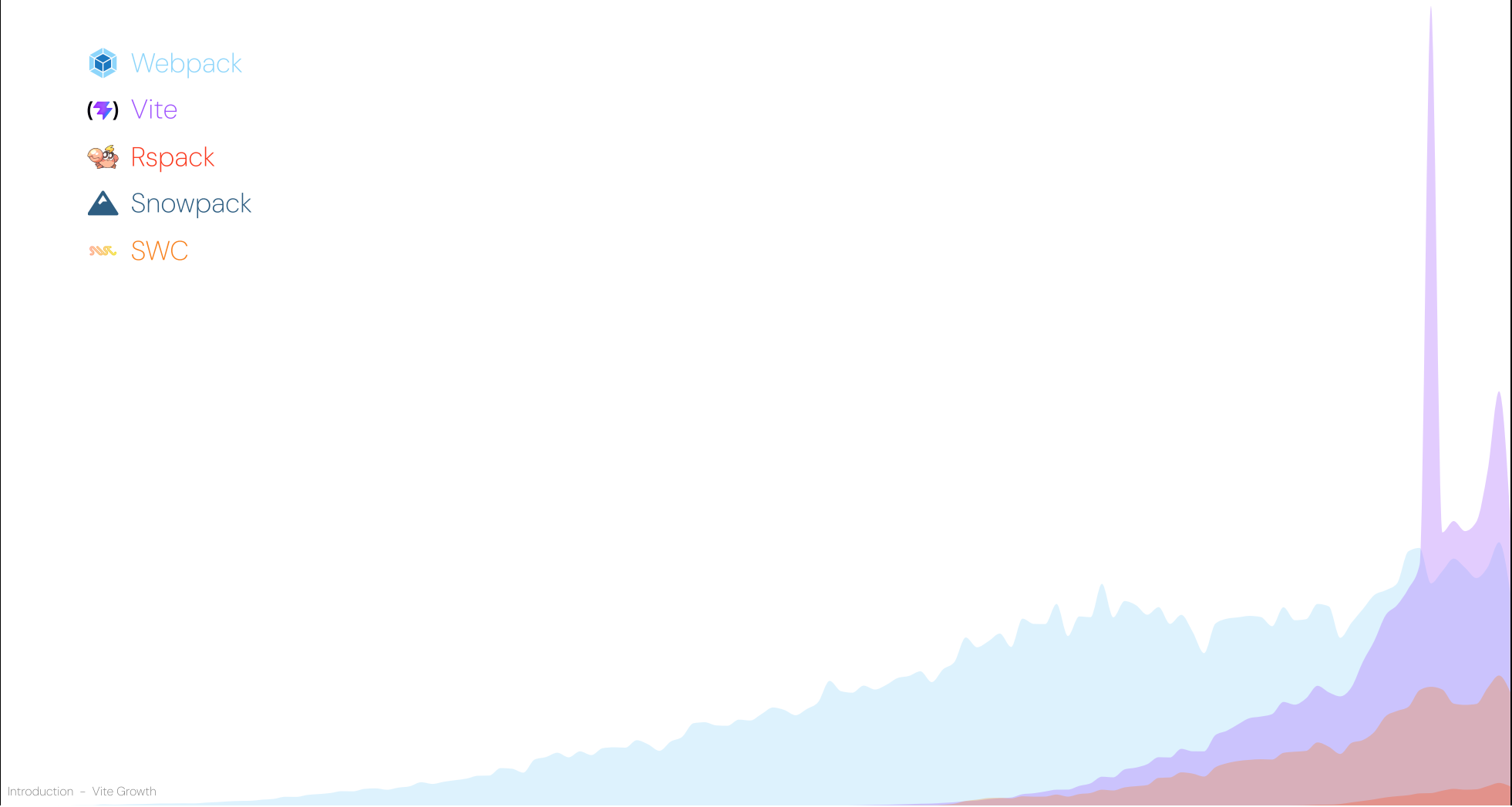
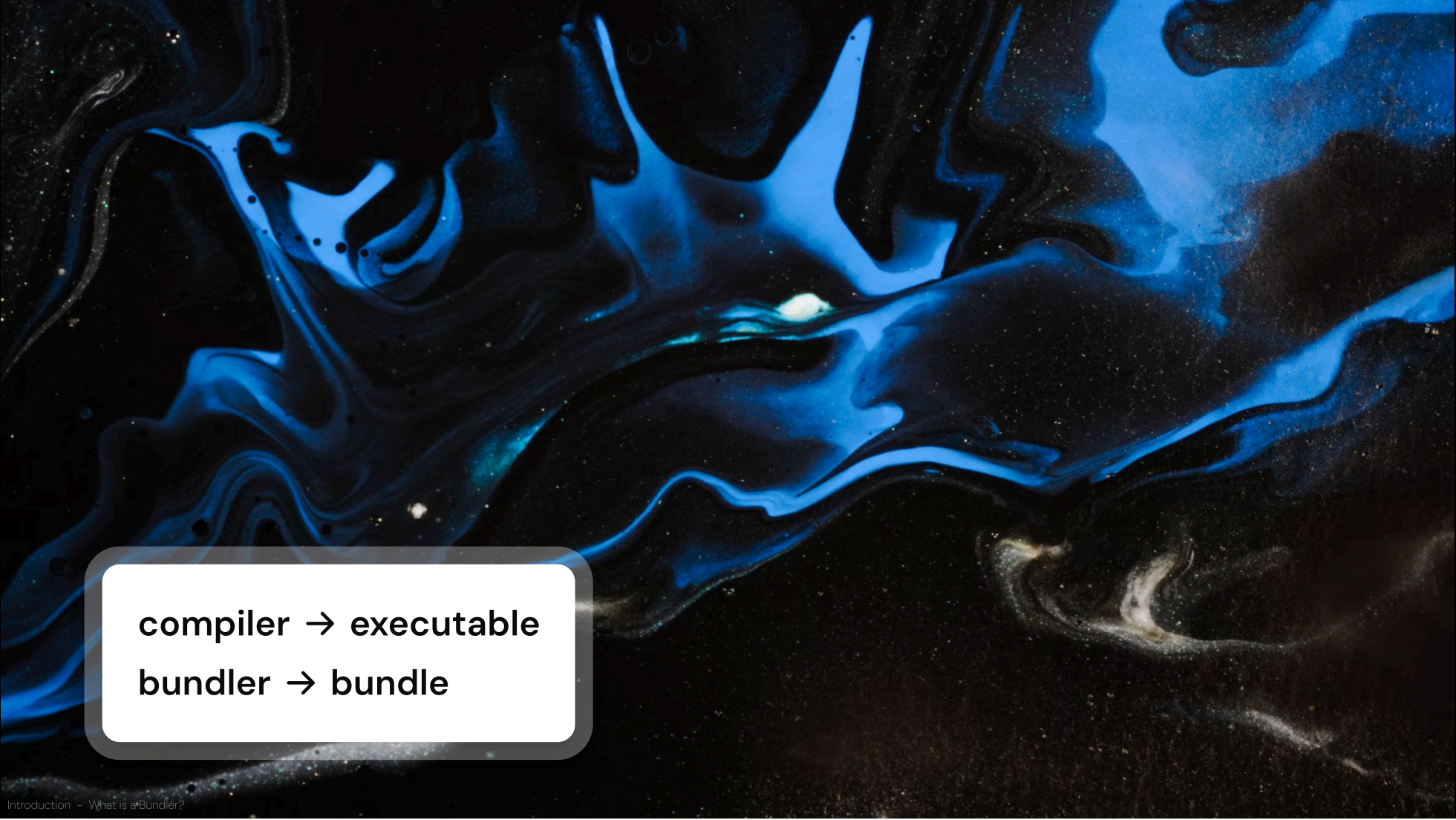
Au cœur d’une pipeline: démystifions Vite et ses plugins


VITE

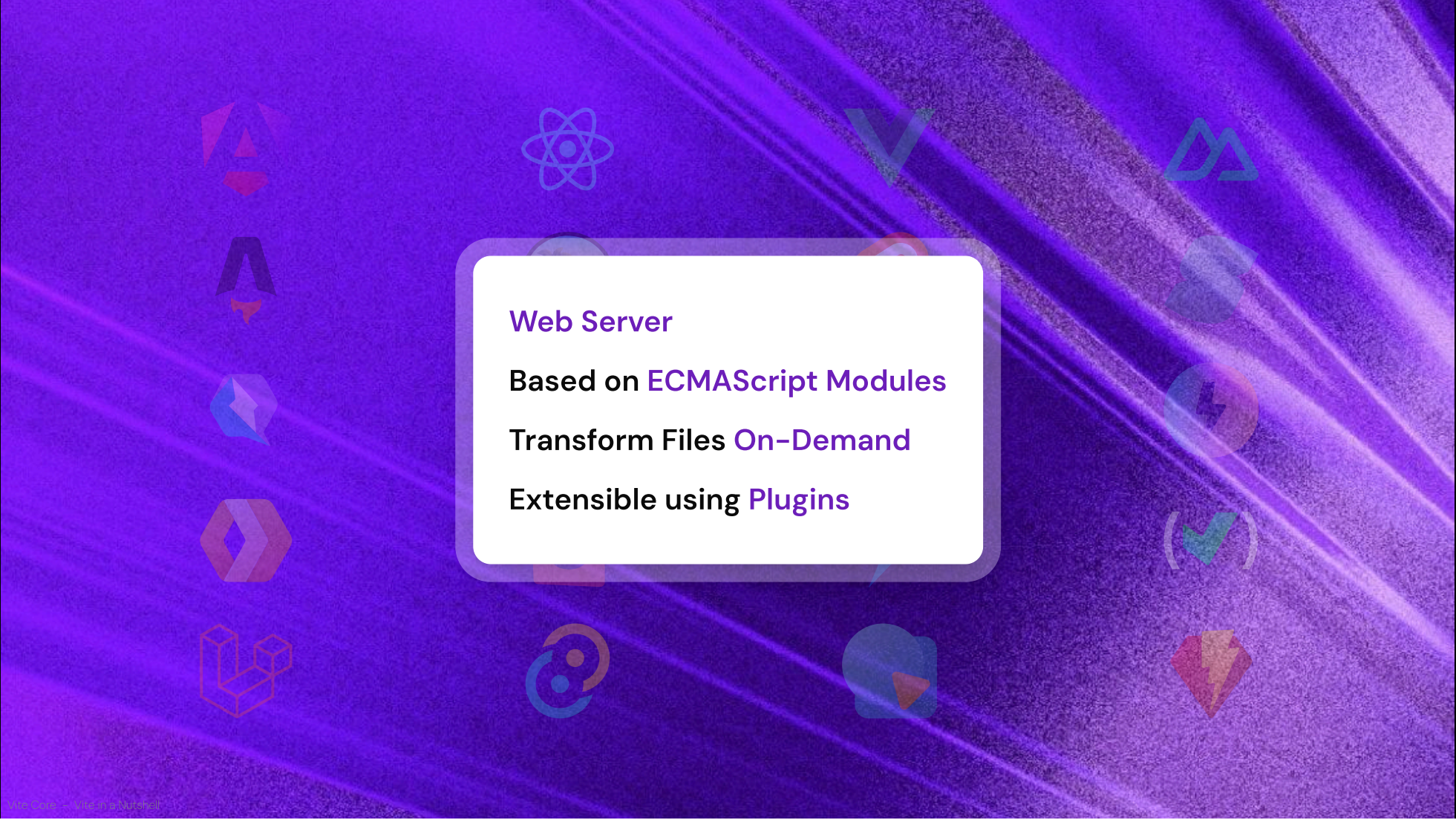
- Webpack
- Vite
- Rspack
- Snowpack
- SWC




compiler → executable
bundler → bundle



Type your message...


Web Server
Based on ECMAScript Modules
Transform Files On-Demand
Extensible using Plugins

HTTP Logs

```
GET http://localhost:5173/
GET http://localhost:5173/src/main.ts?t=1772882882631
GET http://localhost:5173/src/style.css?t=1772882882631
GET http://localhost:5173/node_modules/.vite/deps/vue.js?v=ed2c93c8
GET http://localhost:5173/src/App.vue
```

Native ESM instead of bundling-first

Dev server performance

On-demand transformation

Simpler configuration

Ecosystem alignment

ECMAScript module N.

A JavaScript file that uses `import` and `export`. The standard module format in modern JavaScript.

e.g. `main.ts`, `App.vue` *once transformed for the browser.*

module identifier N.

The string used in an import to request a module from Vite.

e.g. `'vue'`, `'./App.vue'`, `'virtual:my-module'`.

hook N.

A function exposed by a plugin to intervene at a specific step in the pipeline.

e.g. `resolveId`, `load`, `transform`.

virtual module N.

A module that does not exist on disk, but is generated on the fly by a plugin.

e.g. an import like `'virtual:my-module'`.

HTML

Use HTML as an entry point with automatic script and asset handling.

JSX

Use JSX with built-in transformation and framework support.

CSS

Import CSS with support for HMR, modules, and preprocessors.

Static Assets

Import static assets such as images, fonts, and media.

JSON

Import JSON files directly as JavaScript modules.

Glob Import

Import multiple modules using a file pattern.

Dynamic Import

Load modules dynamically with native code splitting.

WebAssembly

Import and run WebAssembly modules from JavaScript.

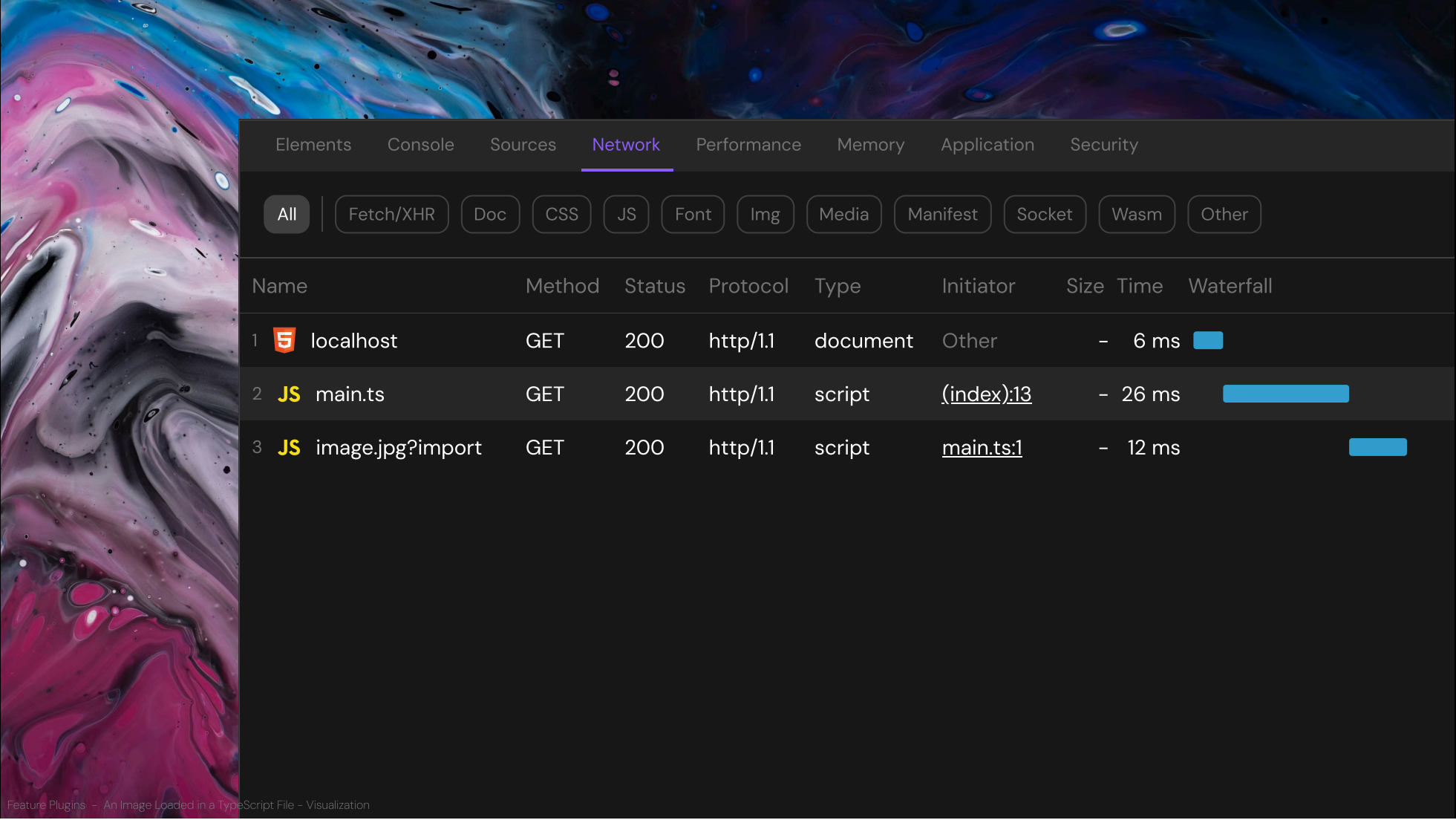
Web Workers

Run code in background threads using Web Workers.

Feature Plugins = Everything is a Plugin

```
import img from './image.png'
```





Feature Plugins – An Image Loaded in a TypeScript File – Visualization


Vite Build Output
dist
index.html
assets
image-BTTwLOJ2.jpg
JS
index-CglLWvq0.js

Vite in 3 Points



1

A web server for development

It handles requests that can be transformed on demand

2

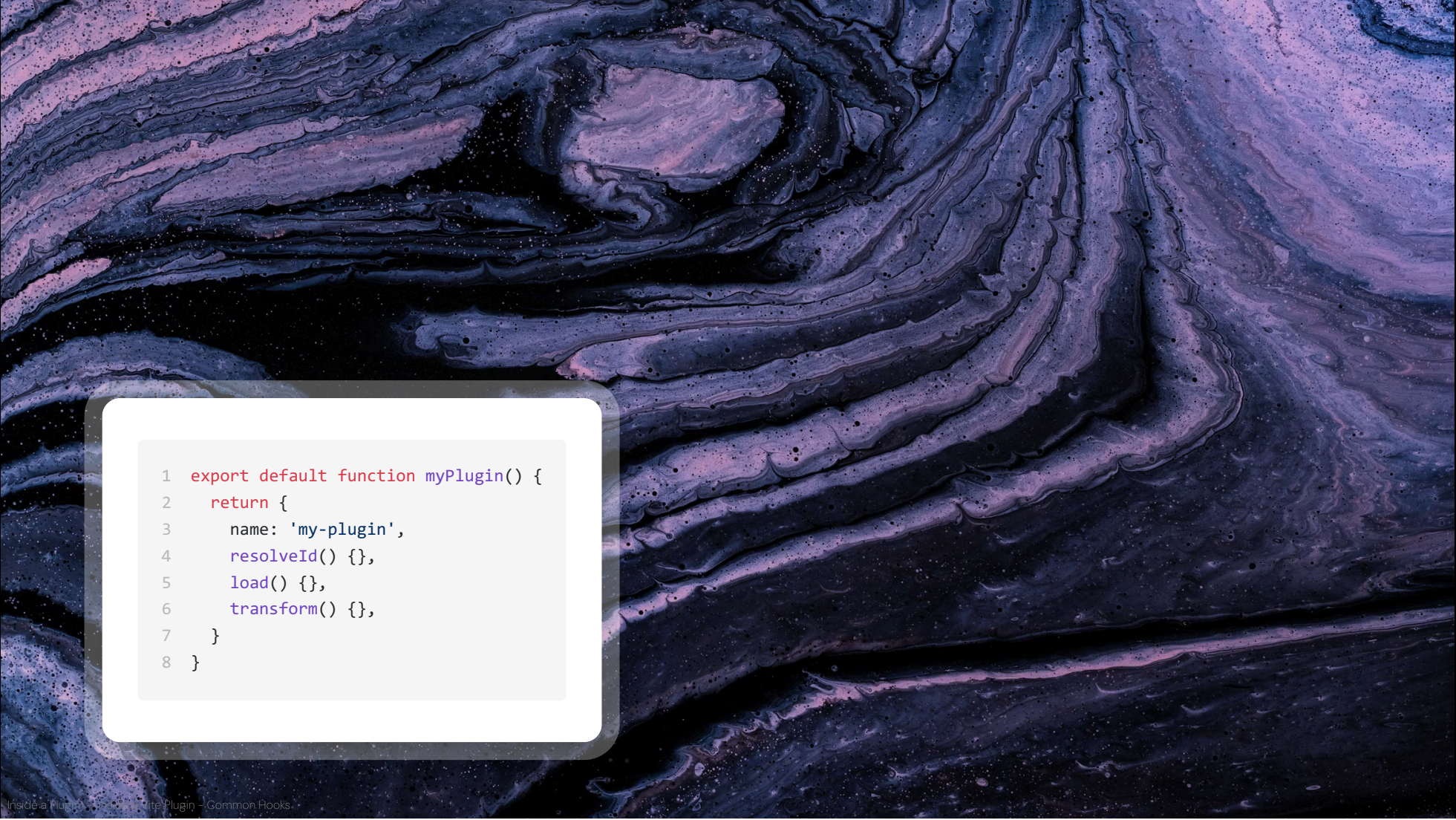
A bundler for production

It transforms our code for production

3

A plugin system

To extend its features and make it do whatever we want


export default function myPlugin() {
return {
name: 'my-plugin',
resolveId() {},
load() {},
transform() {},
}
}


export function myPlugin() {
return {
name: 'my-plugin',
resolveId(id) { // /src/components/HelloWorld.vue
if (id.endsWith('.vue')) {
return id
}
},
}
}


export function myPlugin() {
return {
name: 'my-plugin',
async load() {
return "export default 'Hello World'"
}
},
}


export function myPlugin() {
return {
name: 'my-plugin',
transform(code, id) {
if (id.endsWith('.vue')) {
return transformSFC(code, filename)
}
}
},
}

Multiple plugins?

request

resolveId `resolveId (id)`

VuePlugin

CustomPlugin

load `load (id)`

VuePlugin

CustomPlugin

transform `transform (code, id)`

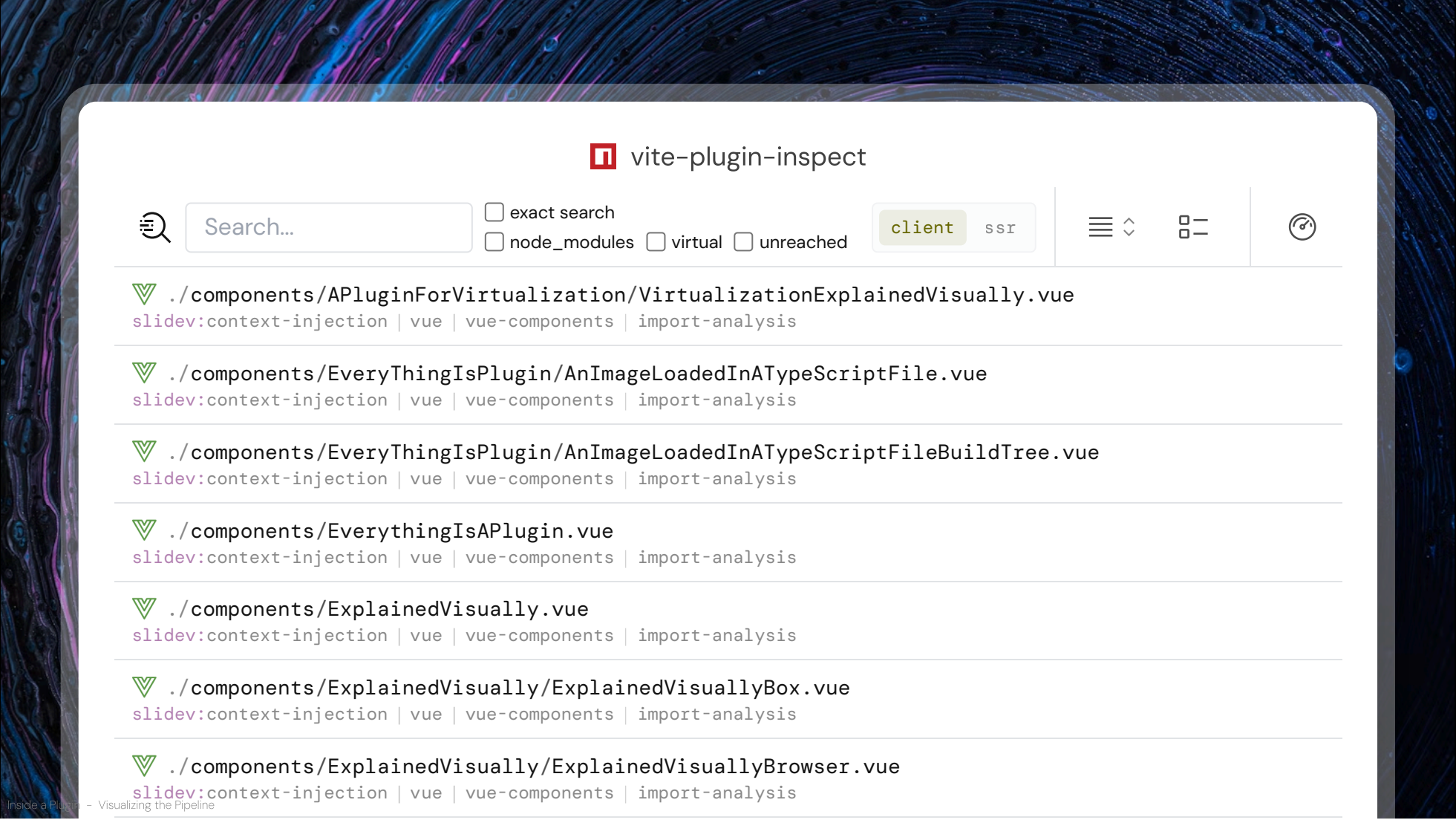
VuePlugin

CustomPlugin

response

input

output


vite-plugin-inspect
Search...
exact search
node_modules
virtual
unreached
client
ssr
./components/APluginForVirtualization/VirtualizationExplainedVisually.vue
slidev:context-injection | vue | vue-components | import-analysis
./components/EveryThingIsPlugin/AnImageLoadedInATypeScriptFile.vue
slidev:context-injection | vue | vue-components | import-analysis
./components/EveryThingIsPlugin/AnImageLoadedInATypeScriptFileBuildTree.vue
slidev:context-injection | vue | vue-components | import-analysis
./components/EverythingIsAPlugin.vue
slidev:context-injection | vue | vue-components | import-analysis
./components/ExplainedVisually.vue
slidev:context-injection | vue | vue-components | import-analysis
./components/ExplainedVisually/ExplainedVisuallyBox.vue
slidev:context-injection | vue | vue-components | import-analysis
./components/ExplainedVisually/ExplainedVisuallyBrowser.vue
slidev:context-injection | vue | vue-components | import-analysis

```
export default function myPlugin() {
  let config = null
  return {
    name: 'my-plugin',
    config(config) {
      config.root = 'src'
    },
    configResolved(resolvedConfig) {
      config = resolvedConfig
    },
    async buildStart() {
      await scanDir(config.root)
    },
    async buildEnd() {
      await generateSitemap(config)
    },
  }
}
```

The Vite Plugin in 3 Points



1. **A Vite plugin is a function that returns an object**
   That object must at least have a name property

2. **There are 3 main hooks to act on modules**
   resolveId, load, and transform

3. **There are also hooks for Vite's lifecycle**
   config, configResolved, buildStart, buildEnd, and closeBundle

A Plugin for Virtualization

HTTP Logs

```
GET http://localhost:5173/
GET http://localhost:5173/src/main.ts?t=1772882882631
GET http://localhost:5173/@id/__x00__virtual:my-module
```

```
export default function myVirtualModulePlugin() {
  const virtualModuleId = 'virtual:my-module'
  const resolvedVirtualModuleId = '\0' + virtualModuleId

  return {
    name: 'my-virtual-module-plugin',
    resolveId(id) {
      if (id === virtualModuleId) {
        return resolvedVirtualModuleId
      }
    },
    load(id) {
      if (id === resolvedVirtualModuleId) {
        return `export const msg = "from virtual module"`
      }
    },
  }
}
```

```
import { defineConfig } from 'vite'
import virtual from 'vite-plugin-virtual'

export default defineConfig({
  plugins: [
    virtual({
      'virtual:git:commit': () => {
        const commit = execSync('git rev-parse HEAD').toString().trim()
        return `export default "${commit}"`
      },
      'virtual:api:data': async () => {
        const res = await fetch('https://api.example.com/data')
        const data = await res.json()
        return `export default ${JSON.stringify(data)}`
      },
    })
  ]
})
```

Virtualization in 3 Points



1. **Vite can respond to requests with virtual modules**
   Modules that do not exist on the disk but are generated on the fly by a plugin

2. **To create a virtual module, use the resolveId and load hooks**
   Take the ownership of an import with resolveId and return the code with load

3. **Used in many plugins and very powerful**
   It allows you to inject dynamic data, generate code on the fly, and much more

Hot Module Replacement (HMR)

Update modules in the browser without a full page reload, preserving application state.

Middleware

Add additional routes to the Vite dev server to handle custom API requests or serve dynamic content.

Command Runner

Execute custom commands alongside the Vite dev server, with access to the Vite configuration and lifecycle.

Inter-process Communication

Allow different processes to communicate with each other to share data.

Macros

Transform source code at build time to enable new syntax or features that are not natively supported by JavaScript.

Server Framework

Extend Vite's dev server capabilities to build custom backends, APIs, or full-stack applications.

What to Keep in Mind



1. **A pipeline between your files and the browser**
   Each request goes through Vite, which can transform modules on demand

2. **Plugins can create modules that do not exist**
   Virtual modules, routes, icons, build info... everything can be generated on demand

3. **Everything can be extended by a plugin**
   Code transformation, HMR, middlewares, side effects, macros, backend...

In 20 minutes, you discovered what you need to build your Vite plugin.





C'était

Au cœur d'une pipeline : démystifions Vite et ses plugins

Pour **Paris TypeScript** (09 juin 2026)

Présenté par  Estéban Soubiran

⬇️ Votre feedback ⬇️



example.com

🎁 Contenu additionnel 🎁

