divotion

tech radar

2024 volume 1

Our view on recent developments in our field

4th tech radar edition

Introduction

Our experienced specialists participate every day in the development of many different software projects all over the Netherlands. Every six months our engineers gather to discuss the latest emerging trends and developments in software engineering. We seek to capture these trends in a technology radar. Each edition of the radar shows how these trends change, compared to the previous edition. A shift can indicate that we see a technology becoming more, or less, relevant for certain use cases.

If a trend does not show up in later editions, it signifies that there are no relevant developments or experiences that cause us to shift our assessment. A certain technique or framework that is not currently on the radar does not mean we do not like it or do not use it; it only says that we think it has not moved on the radar.

With this document, we will discuss the shifts we have observed over the past six months. This analysis guides us in deciding which technologies we use and recommend.

Tech Radar

The idea to create a tech radar was initiated by ThoughtWorks. They periodically showcase their view on new trends and development. In addition, they advise all to define their own radar.

At Divotion we fully support that view. Building a tech radar is an instructive and valuable experience, where we mutually share our knowledge and create a common awareness around technology. We believe that specialists should be able to create and maintain their own toolset to perform their job to their best ability. When composing a radar, you facilitate a broad discussion on technology, so organisations can strike the right balance between the risks and rewards of innovation. We can help you to kick-start these discussions in your organisation. Let your teams innovate and inspire each other. Together they can draw up a set of technologies and techniques that can accelerate your business.

Overview

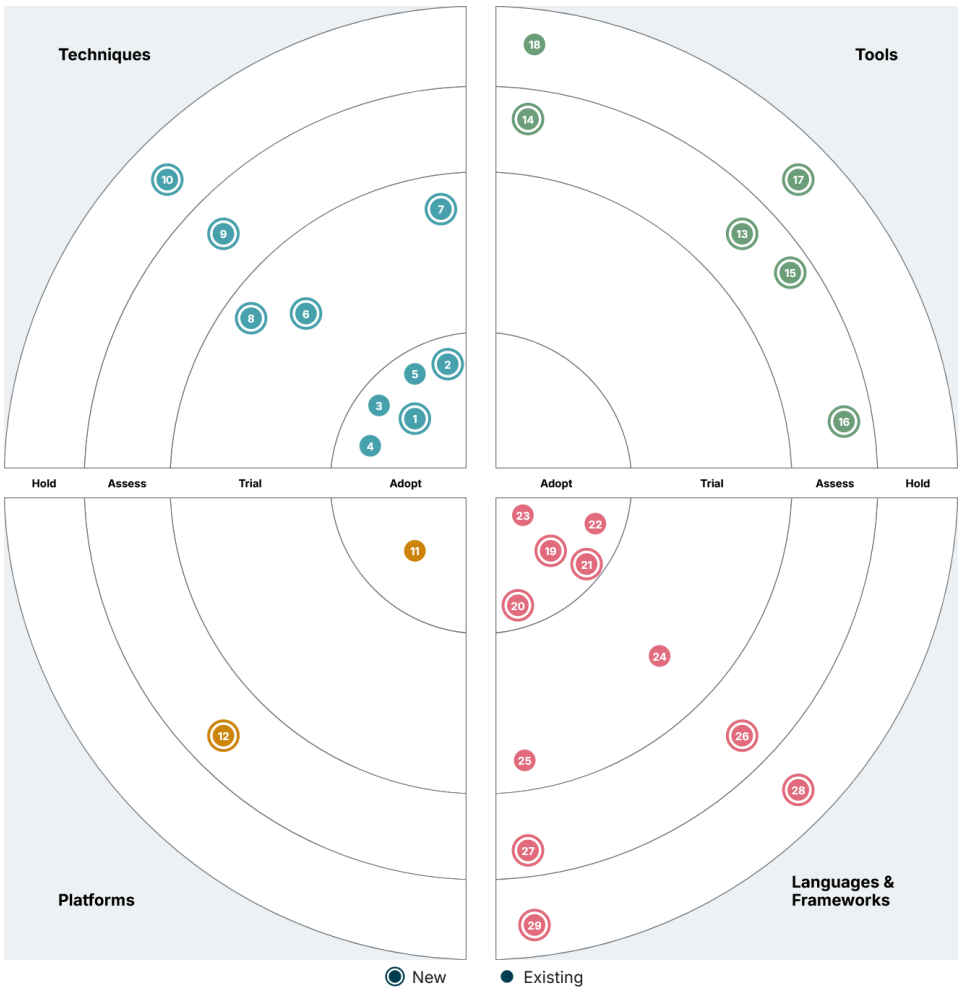
The radar consists of quadrants and rings, containing blips to indicate technologies of interest. The quadrants subdivide the different subjects into categories:

- **Techniques** help developers create better software.
- **Tools** aid in the development and delivery process.
- **Platforms** to deploy and execute programs.
- **Languages and frameworks** that support developers in their daily tasks.

The rings in each quadrant indicate which phase of the adoption cycle we believe a technology is currently at:

- **Adopt**: We recommend using this technology, wherever it fits the requirements.
- **Trial**: We advise to gain experience with this technology, wherever a project allows for a certain degree of risk.
- **Assess**: Interesting topic to learn more about and assess its future impact; yet too early to use in production.
- **Hold**: Do not deploy in a project not already using this technology.

In the subsequent sections we will delve into our views on recent developments in the field of software engineering more closely.



focus on front-end development

Techniques	Tools
ADOPT 1. AI Code Assist 2. Angular's renaissance 3. Design Systems 4. HTTP/3 5. TypeScript BFF **TRIAL** 6. Container queries 7. Release Please 8. WebAuthn **ASSESS** 9. Server Actions **HOLD** 10. DDD	**ADOPT** - **TRIAL** - **ASSESS** 13. Biome 14. Changesets 15. Guidepup 16. V0 **HOLD** 17. ESLint formatting 18. Rome
Platforms	**Languages & frameworks**
ADOPT 11. Deno **TRIAL** - **ASSESS** 12. Bun **HOLD** -	**ADOPT** 19. HTML 20. Tailwind CSS 21. WebAssembly 22. Remix 23. SvelteKit **TRIAL** 24. Astro 25. Qwik **ASSESS** 26. Analog 27. Yew **HOLD** 28. CSS in JS 29. Sass

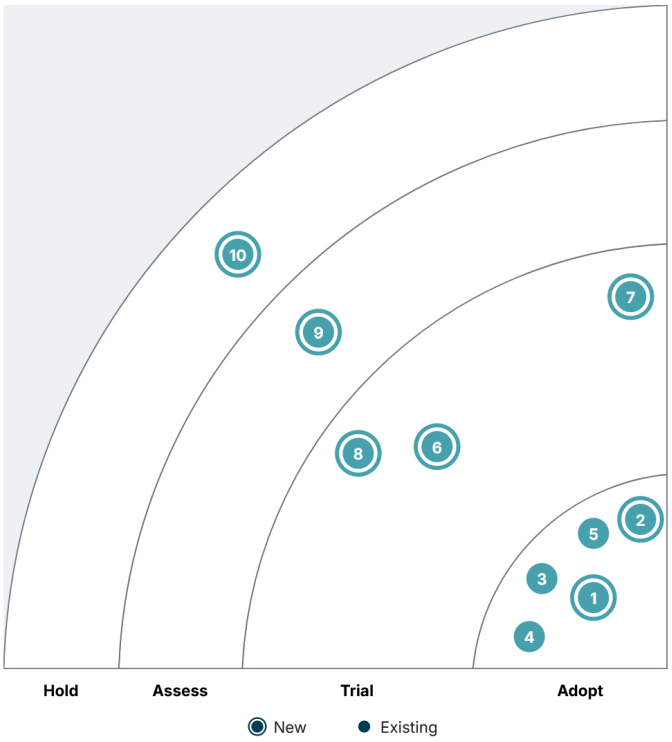
Techniques

Adopt
1. AI Code Assist
2. Angular's renaissance
3. Design Systems
4. HTTP/3
5. TypeScript BFF

Trial
6. Container queries
7. Release Please
8. WebAuthn

Assess
9. Server Actions

Hold
10. DDD



AI Code Assist

ADOPT

Starting to use AI Code Assist tools can really change the way you work, offering many benefits. These tools come with cool features like helping you complete code faster, checking your syntax, finding errors, and giving you tips to improve your code. They're easy to blend into your usual coding setup and can help with organizing your code better and writing documentation, which is helpful for both beginners and expert coders.

When you're looking at options for AI Code Assist tools, you can choose between Open Source Software (OSS) and vendor-based APIs. OSS is great if you like to tweak things yourself and want to be part of a community, but it can be a bit technical. Vendor-based APIs are more straightforward to use, often have better support, and are updated regularly, but they might cost money and don't offer much room for customization.

One important thing to keep in mind with vendor-based APIs is that they might use your data to help improve their future AI models. This is something to think about if you're concerned about how your data is used. In essence, adopting AI Code Assist tools is a smart idea. They ease the coding process, speed up your tasks, and open up new learning opportunities.

Angular’s renaissance

ADOPT

Angular has recently released version 17 of the web framework, calling it their "renaissance". With this release comes a lot of improvements and modernizations: SSR/SSG and a new control flow for templates, on top of the earlier released standalone components, signals and the inject function. These features, also used internally by Angular, improved performance, require less boilerplate code and provide easier testing

We recommend taking full advantage of the new Angular features as it looks like this is the way forward for the framework. More internal Angular components are being rewritten to use the new features, and we hope the Angular team continues to improve the framework in this new and exciting direction. Angular is calling this a renaissance, and we agree. The new features are a breath of fresh air and make Angular feel like a modern framework again.

Design Systems

ADOPT

Design systems have grown more popular in recent years. A well-designed design system can significantly improve an organization’s design workflow, efficiency, and consistency, ultimately leading to a better user experience. A design system is maintained and implemented by both a UI designer and a developer, to keep designed components and their software counterparts in sync. This means that a design system is something that keeps on living and growing. A typical and core part of a design system is a set of design tokens. These tokens define the styling defaults of primitives like color, font and margins. Updating a primitive or token results in an instant change throughout all the designs and, when updated in code, updates all components as well. This saves a lot of time and keeps both design and software consistent in look and feel.

We recommend the use of a design system to improve the user and development experience. However, setting up a design system requires some experience and time. Implementing one is not a one-time thing, it is a long-term investment. This means that it is not suitable for all projects. Some off-the-shelf design systems including their software counterpart might save you some time. Make sure to do a cost-benefit analysis to see whether a design system suits your project’s needs.

HTTP/3

ADOPT

HTTP/3 is a new standard for web browser and server communication. Where HTTP/1 is not efficient in communication, HTTP/2 and HTTP/3 are much more efficient and faster. HTTP/3 uses a new transport protocol called QUIC and solves some problems with HTTP/2. For example, smartphones switching from Wi-Fi to cellular data, and it prevents the blocking of all other network communication when packets are lost.

HTTP/3 is quite new and is now in the Request for Comments (RFC) phase. Despite it is not an official standard yet, Google already uses it for Gmail in production for example. Currently all major modern browsers, except for Safari, support HTTP/3. To use it, the backend infrastructure has to be ready for it. This is not as trivial as it seems, because all reverse-proxies and other infrastructure components (if applicable) have to support it well. In our previous tech radar, we had HTTP/3 on assess, but we think the time is there to have it adopted. Although it not being a final standard yet, it is widely used and when implemented well on all sides, it can improve the experience a user has with the web application you developed. We would suggest asking the infrastructure team inside your company to have a look into HTTP/3 and find out what is needed to implement it on the infrastructure side.

TypeScript BFF

ADOPT

TypeScript can be used as the language to develop a backend for frontend for web applications. A backend for frontend (BFF) is a backend service that simplifies the frontend code by fetching and processing data from multiple APIs and providing a single, simplified API. This allows you to build a backend for frontend that is strongly typed and easier to maintain. When types are shared between the frontend and the backend for frontend, this makes for a powerful setup. TypeScript in combination with NodeJs or Deno is particularly suited to build scalable and efficient real-time applications.

We recommend using TypeScript for your backend for frontend, so you can develop a strongly typed backend for the frontend that is simpler to maintain. When used in conjunction with NodeJs or Deno, TypeScript is especially well-suited for constructing real-time applications that are both scalable and efficient. Some frontend frameworks already incorporate a server-side runtime, which makes using them as a backend for frontend even more accessible and powerful.

Container queries

TRIAL

Container queries in CSS are a big step forward in making responsive web design look good on different devices. Instead of changing styles based on the whole screen size, they change styles based on the size of the part of the responsive web design they're in. This new way is more flexible and lets designers create better, more adaptable responsive web designs. You set up container rules, similar to how you use media queries. As of this year, container queries are supported in all major browsers. These queries are really helpful for making parts of responsive web design respond to different sizes, changing how designers approach making responsive web design flexible and improving how parts of responsive web design work and look on different screens.

Now's a great time to try out container queries, but keep in mind that some of the latest features aren't fully implemented in all the major browsers yet. If you're having a hard time making a component with media queries, why not give container queries a go? It's a good idea to experiment with them, and that's why we place container queries in the "adopt" category.

Release Please

TRIAL

Release Please is a tool that automates the generation of CHANGELOGs, GitHub releases, and version bumps for your projects. It achieves this by parsing your git history, specifically looking for Conventional Commit messages, and subsequently creating release pull requests. The generated release notes offer valuable information regarding changes, improvements, bug fixes, and new features introduced in each software version. This documentation serves as a communication tool for users, administrators, and developers. For those not using GitHub, an alternative option is Commit and Tag Version.

Release Please simplifies the process of updating the changelog and version based on your commits. However, its effectiveness depends on the commitment of your team to consistently use Conventional Commits. Consider experimenting with it to determine its suitability for your team's workflow.

WebAuthn

TRIAL

Web Authentication (WebAuthn) is a web standard that enables strong authentication with asymmetric cryptography, enabling passwordless authentication and secure multi-factor authentication (MFA). Benefits of WebAuthn over password-based authentication include: protection against phishing, reduced impact of data breaches and password (guessing) attacks. WebAuthn has been available for a while as an MFA option but is now also available as a primary authentication method.

With support in Firefox coming very soon, all modern browsers can now make full use of the capabilities of WebAuthn. With the clear benefits over password-based authentication, we expect WebAuthn to become the new standard for authentication on the web. The user experience, however, is not yet optimal. With differences in interface between browsers and operating systems, it is difficult to teach users how to use WebAuthn. We expect this to improve over time, as the standard matures.

Server Actions

ASSESS

Several frontend frameworks are incorporating a server-side runtime solution, along with the frontend that runs within the browser. This allows for developing a full-stack application. Using these frameworks, the frontend can be rendered on the server. It will serve HTML to the browser, rather than the classic Single Page App approach. By leveraging this power, data can be loaded and processed on the server. Processing data received from the browser in such a framework on the server-side is called Server Actions. Because these are tightly integrated in the frontend framework, it makes for more elegant solutions with tight integration.

If you are developing a web application, you should consider using a full-stack framework. It will empower you to build more elegant solutions for typical problems that arise in an application, such as data validation, state management and error handling.

DDD

HOLD

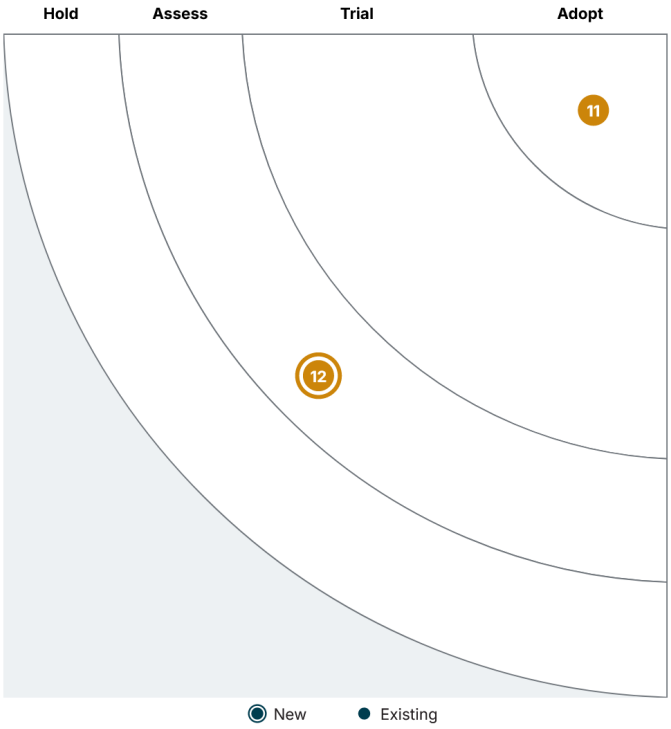
Domain-Driven Design (DDD) is an approach to software development that focuses on understanding and modeling the complexities of a business domain. It emphasizes collaboration between technical and domain experts to create software that directly reflects the real-world problem space. DDD provides a set of principles, patterns, and practices to design and build software systems that are centered around the core domain and its intricacies.

We advise not to use DDD in its complete form, particularly in existing projects. Implementing DDD demands significant time and has a steep learning curve. We've observed limited return on investment, often resulting in increasing the complexity of the solution which pose various challenges for maintenance. DDD could be more suitable for new complex projects. And we've observed benefits when DDD is merely used as a communication tool to improve the collaboration between the business and the development teams.

Platforms

Adopt
11. Deno

Assess
12. Bun



Deno

ADOPT

Deno is a modern runtime for JavaScript and WebAssembly powered by Chrome's V8. Created by Ryan Dahl (the creator of Node.js) to solve fundamental design and security flaws and embraces modern best practices like ES Modules and TypeScript. It's very much 'batteries included' with TypeScript support, a built-in linter and code formatter, and IDE integration. As opposed to, Node.js, it offers much broader and spec-compliant support for platform / Web APIs. It also features significant and broad performance optimizations like TypeScript type checking that's 4.7x faster.

We rate Deno with adopt due to its improved out-of-the-box experience, the much-improved platform / Web APIs, and its significant and broad performance optimizations. Deno improved their security model and offers a more secure runtime by default. Over the last year, they increased the backwards compatibility and the ecosystem is growing. But we also see a rise in the use of Bun, and we expect that to be a viable alternative in the near future.

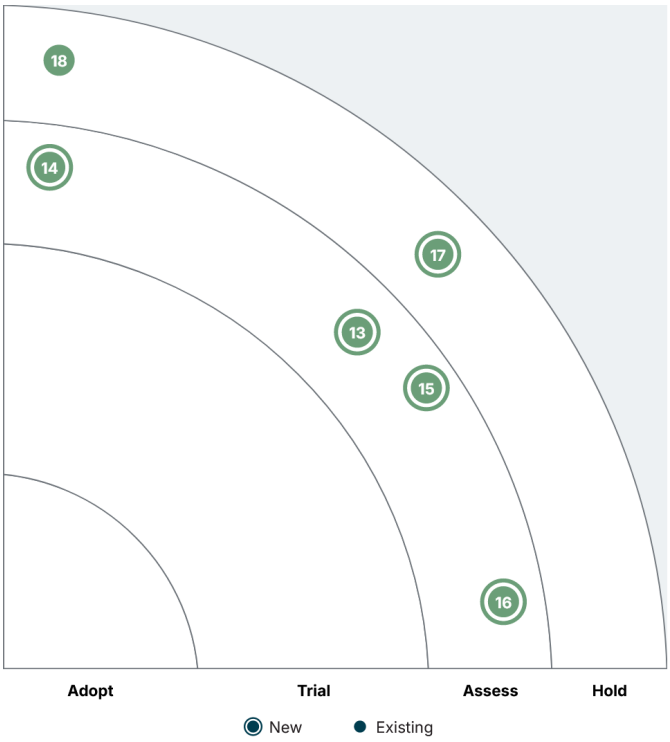
Bun

ASSESS

Bun is an all-in-one toolkit for JavaScript and TypeScript apps. At its core is the Bun runtime, a fast JavaScript runtime designed as a drop-in replacement for Node.js. It dramatically reduces startup times and memory usage. In addition, Bun also implements a test runner, script runner, and NodeJS-compatible package manager.

Bun has great potential because it's super fast (for example, installation is 25x faster), Jest compatible, and supports snapshot testing out of the box. Why do we categorize it for assessment, and why should you avoid using it in production already? Version 1 of Bun has just been released, and that's why we advise against using it in production at this time. You can assess it in side projects or development scripts if you're interested in experimenting with new tooling.

Tools



Assess
13. Biome
14. Changesets
15. Guidepup
16. V0

Hold
17. ESLint formatting
18. Rome

Biome

ASSESS

Biome is the official fork of Rome, since the Rome Tools Inc company has been shut down. Biome is a kit of development tools built from the ground up in Rust with no dependencies, so it does not require Node.js to function. They have a vision where combining the functionality of separate tools will lead to simpler configuration, amazing performance, and adaptability to any stack. Biome aims to support all main languages of modern web development.

As of now, Biome includes a formatter, a linter and an analyzer. We will be awaiting the release of the remaining tools, consisting of a compiler, bundler, and testing solution. Once those are ready this may prove to be a great stack for developing your applications, but for now we recommend to hold off on it until it has matured.

Changesets

ASSESS

The changesets workflow is designed to help when people are making changes, all the way through to publishing. It lets contributors declare how their changes should be released, then we automate updating package versions, and changelogs, and publishing new versions of packages based on the provided information. Changesets focuses on the changes that are being made, rather than the packages that are being changed, and so it is a good fit for monorepos. With their CLI tool, you can use changesets with any package manager, and any CI.

We think that changesets is a great tool when you are working with a monorepo and for that reason we place it in adopt. Changesets help you to keep track of the changes that are being made and to make sure that you are releasing the right versions of packages. It helps you to keep your changelogs up-to-date and to make sure that you are publishing the right versions of packages.

Guidepup

ASSESS

Guidepup is a screen reader driver for test automation. A screen reader is an assistive technology that translates digital text and content into synthesized speech or Braille output, enabling individuals with visual impairments to access and navigate web content. With Guidepup you can automate screen reader test workflows the same you would for mouse or keyboard-based scenarios. It can be used to create (unit)tests that can check for expected screen reader behaviour. This way it is also a self-describing way of documenting the expected screen reader behaviour of a component. Guidepup can be used with VoiceOver, NVDA or the "Guidepup Virtual Screen Reader". The virtual screen reader is operating system agnostic and can be used on Mac, Windows and Linux.

Although we do not yet have hands-on experience with Guidepup, we think you should have a look at it if accessibility is important to your organisation. The use of the Virtual Screen Reader appeals to us to be able to have one set of tests that can be run locally and in the CI/CD pipeline. We do not see Guidepup as a replacement for Axe or Pa11y, but as an extra tool to help better convey what the expected behaviour of a component or page is.

V0

ASSESS

Vercel Labs introduces v0.dev, an innovative tool that simplifies website building using AI. You can easily turn text into web interfaces and receive ready-to-use React code. v0.dev seamlessly integrates with two popular web design frameworks: Shadcn UI and Tailwind CSS. This ensures that your web design process is smooth and accessible.

v0.dev revolutionizes website creation, streamlining the process with its advanced AI. This tool enables rapid website development, significantly reducing the time required to build a site. Its user-friendly interface welcomes everyone, from coding novices to seasoned web developers. This accessibility and efficiency make v0.dev a must-try tool, placing it firmly in the "Assess" category.

ESLint formatting

HOLD

ESLint is a popular linting tool for JavaScript and TypeScript that can be used to enforce coding conventions. It is highly configurable and can be used to enforce a common coding style for a project. Back in 2020 ESLint froze all current formatting rules and halted development of new ones due to the maintenance cost, and has now deprecated those rules.

"ESLint is not the right tool for source code formatting", wrote Nicholas Zakas, creator of ESLint, in an issue on GitHub. Since all formatting rules have been deprecated and will be removed in a future version of ESLint, we do not recommend using those code formatting rules at this time. There are a number of tools that are faster and, in our opinion, better suited for formatting source code. Prettier, js-beautify and dprint, to name a few.

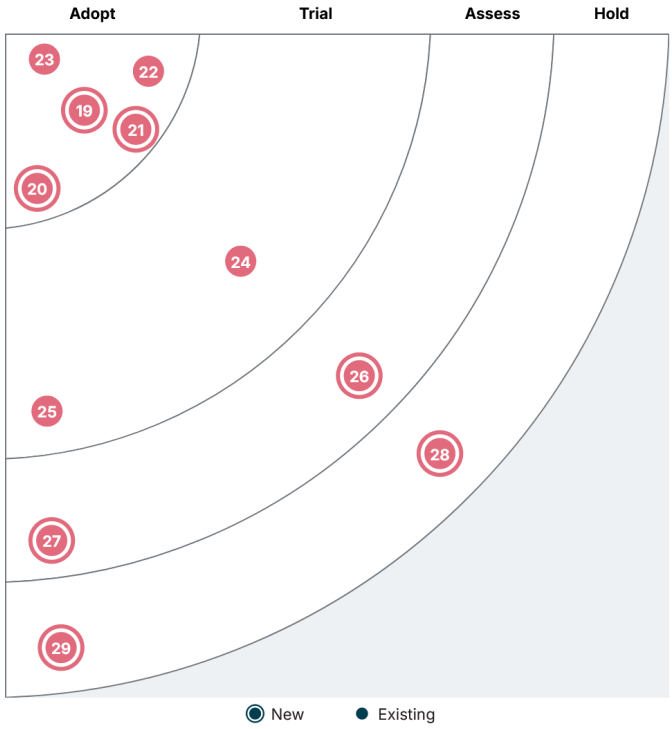
Rome

HOLD

Rome was a kit of development tools, created by the Rome Tools Inc company. Unfortunately, the company did not survive and the project is now dead.

In our last tech radar Rome was on the Assess ring. Because of its discontinuity, we now place it on Hold Rome now has an official fork called Biome, which is led and maintained by the former Rome team. You can read more about Biome in this Tech Radar.

Languages & frameworks



Adopt
19. HTML
20. Tailwind CSS
21. WebAssembly
22. Remix
23. SvelteKit

Trial
24. Astro
25. Qwik

Assess
26. Analog
27. Yew

Hold
28. CSS in JS
29. Sass

HTML

ADOPT

HTML, or HyperText Markup Language, is a standard language used to structure and present content on the web. Its primary goal is to define the logical structure of a document by utilizing a system of tags that represents various elements. These elements, such as headings, paragraphs, links, images, and forms, provide a hierarchical structure to organize content. HTML documents are interpreted by web browsers, which use the markup to render text, images, and multimedia elements in a visually cohesive manner. Through attributes and tags, HTML enables developers to create semantically rich documents, fostering accessibility, search engine optimization, and consistent user experiences across different platforms and devices.

Of course, HTML is nothing new, but we too often see that developers are not aware of the importance of using the right elements, and attributes and with it the right semantics. With the correct use of elements and attributes, a website or web application can have dramatically better search engine optimization and accessibility. HTML is a living standard, meaning that new features can be added. One of these features is the relatively new dialog element that is now well-supported by all browsers since March 2022. We advise you to have a look at the new features and to implement them if it is possible. Even the old, important, features are missed a lot by developers, like using the "for"-attribute on a label element that relates to the id of an input element. You can also think about an online training focussed on HTML, or have an experienced developer in your company organize one. A lot of developers think they know all about HTML, but there is more to it than meets the eye.

Tailwind CSS

ADOPT

Tailwind CSS is a utility-first CSS framework that emphasizes a pragmatic and customizable approach to styling web interfaces. It provides a set of small, single-purpose utility classes that can be directly applied in the HTML markup. This allows developers to quickly prototype and build interfaces without having to write custom CSS, which can significantly speed up the development process, especially for small to medium-sized projects. Unlike some other CSS frameworks that require a preprocessor like Sass or Less, Tailwind can be used directly in your HTML without the need for additional build steps.

In our opinion, Tailwind CSS is worthy of serious consideration for both new and existing projects, although there is much debate on this matter. Tailwind CSS has built-in support for many popular front-end frameworks, which in itself is a huge positive. It also has a thriving community, which means there are numerous resources, tutorials, and plugins available. Do keep in mind that while Tailwind CSS has its strengths, it may not be the best fit for every project or development team. The utility-first approach may be a matter of preference, and some developers may prefer a more traditional component-based framework. Ultimately, the choice of CSS framework depends on the specific needs and preferences of the project and its developers.

WebAssembly

ADOPT

WebAssembly, in short "Wasm". In the past, there have been several attempts to create a fast low level language for the web (asm.js and dart). This has been difficult because of its backward compatibility and versionless requirements because of the web. The World Wide Web Consortium (W3C) named Wasm a web standard in 2019, thus becoming the fourth web standard with HTML, CSS and JavaScript. Wasm is a binary language which can run in it's own secure sandbox/vm. It allows to combine C++ or rust with html and javascript and run it directly on a CPU. Wasm finds application in performing resource-intensive calculations within web pages, executing directly in the browser. Notably, applications like Figma and Photoshop are able to run in the browser, thanks to Wasm. Another interesting use case is where it replaces Docker in, for example, edge computing or IOT applications. It's secure because it doesn't rely on external packages and is super fast in starting up because it doesn't have to initiate Docker.

We strongly advise you to have a look at Wasm if you are porting an existing C++ application to the web, or if you want to do heavy calculations in the browser.

Remix

ADOPT

Remix Focuses on web standards and modern web app UX to build better websites. As a full-stack web framework in React it lets you focus on the user interface and work back through web standards to deliver a fast, slick, and resilient user experience. Remix creates a bridge between static site generated websites and dynamic server side rendered content.

We think Remix is a great fit when writing a React application. Remix is also listed as a good option by the official React documentation. One of the best features of Remix is that it allows you to write to your server by creating a form like in PHP and then handling the data submitted on the server-side. This makes Remix a good framework for high performant React applications.

SvelteKit

ADOPT

Powered by Svelte and Vite, SvelteKit is a framework for building a reactive application. It replaces Sapper which was the previous attempt at building a framework for Svelte. It is very flexible and fast, supports various rendering patterns and comes with zero-configuration tooling. SvelteKit being a meta-framework for Svelte already comes with a bundler, router, TypeScript support and various deployment options out of the box.

SvelteKit is moving to Adopt on our radar because we've seen adoption grow and expect that trend to continue. We think SvelteKit is a good alternative to Next.js, Nuxt and other meta-frameworks for usage in smaller applications where website speed and development iteration speed are a priority. More improvements to both Svelte and SvelteKit are coming in the near future, and we're excited to see how the framework evolves.

Astro

TRIAL

Astro is a new kind of static site builder for the modern web. It uses a modern server-side templating language that renders directly to HTML & CSS, eliminating heavy JavaScript automatically. Astro uses partial hydration to eliminate dead code from your website and only hydrates your essential, interactive UI. Use a frontend UI component built with React, Preact, Svelte, Vue, SolidJS, AlpineJS or Lit and Astro will automatically render it to HTML at build-time and strip away all JavaScript.

Astro has recently released version 3 and is steadily becoming a more mature product and expanding its toolbox of features. We think the customizability and flexibility of Astro are huge selling points, and we've seen the usage of Astro steadily increase over the past few months. It currently looks like this trend will continue, and we are excited to see what the future holds for Astro.

Qwik

TRIAL

Qwik is an HTML-first framework that automatically lazy-loads all JavaScript code. It uses static HTML generated from server-side rendering and ships with less than 1kb on page load. All other JavaScript code is only lazy-loaded on a needed basis. Due to its Qwik optimizer, 'splitting points' for lazy loading are automatically created without relying on the developer to specify any loading boundaries.

Qwik is a promising new framework that has the potential to become a major player in web development. Its 1.0 version was released in May 2023. However, given its relatively young age, we rate it with trial. Its resumable execution model is a huge improvement over having to ship megabytes of JavaScript code to the browser and then hydrating the server state on the client. Additionally, its ecosystem is quite vibrant with third-party support for Storyboard, Vitest, and others. And, last but not least... there's Qwik-React which allows the reuse of existing React components inside a Qwik app.

Analog

ASSESS

Analog is a full-stack meta-framework for building applications and websites with Angular. It is very similar to other meta-frameworks such as Next.js, Nuxt, SvelteKit, and others, but building on top of Angular, instead. Like other meta-frameworks, Analog implements file-based routing, SSR/SSG, and has support for many plugins. At the time of writing, Analog is part of the GitHub Accelerator program, receiving funding and support from GitHub directly.

A number of features in Analog are similar to what Angular 17 now offers, but Analog makes some different choices in plugins and is overall more configurable. The choices made are to make the development of applications and websites with Angular faster. With out-of-the-box support for Tailwind CSS, Markdown content, tRPC, and more, Analog looks to be a great choice for building your future Angular application. While promising, Analog is still in early development and is not yet ready for production use. The documentation, error messages, and overall experience are still a work in progress, but the project is moving quickly and is worth keeping an eye on.

Yew

ASSESS

Yew is a Rust-based framework for building fast, multi-threaded web apps using WebAssembly, ideal for complex and performance-sensitive projects. It offers a component-based structure similar to React, making it easy for developers to create interactive UIs. With features like minimal DOM API calls, web worker support, and JavaScript interoperability, it enhances app performance and flexibility. While Yew excels in projects needing Rust's advanced features, especially in sectors requiring high reliability.

The growing interest in Rust and WebAssembly has made Yew, a Rust-based framework, quite popular recently. Yew is still being developed and isn't fully ready for use in big projects, but it's one of the most promising options among similar Rust frameworks being created right now. Even though Yew isn't new, it's getting a lot of updates, especially for making web apps. This makes it a good time for web developers to start looking into Rust and Yew to build fast and powerful web apps.

CSS in JS

HOLD

CSS-in-JS is a web development approach where styles for a web application are written directly within JavaScript files instead of separate CSS files. This technique aims to address challenges related to scoping, global namespace conflicts, and the dynamic nature of styling in modern web applications. Various libraries and frameworks, such as Styled-components, Material UI, and Chakra UI, facilitate CSS-in-JS by allowing developers to write styles using JavaScript syntax. This approach promotes better component encapsulation, theming, and the ability to use JavaScript logic for conditional styling.

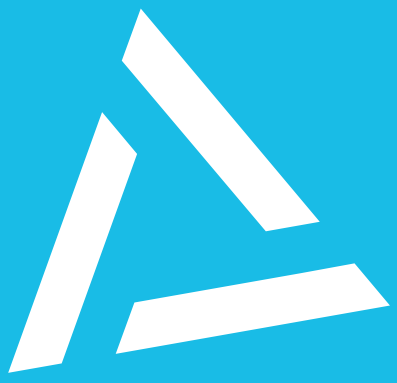
Considering that CSS now has native support for many functionalities traditionally achieved using JavaScript, it is worth giving serious consideration to using native CSS over CSS-in-JS. When using CSS-in-JS, the additional tooling and dependencies introduced can lead to an increased build size and complexity, affecting performance and build times. Mixing styles with JavaScript logic may make the code less readable and harder to maintain, compromising the traditional separation of concerns. Taking this information into account, in our opinion, you should avoid using CSS-in-JS due to the drawbacks associated with its approach. It's important to note that the decision between native CSS and CSS-in-JS depends on the specific needs of your project, team preferences, and the advantages each approach brings.

Sass

HOLD

Sass is a CSS preprocessor introduced in 2006. Code written in Sass is compiled into standard CSS that browsers can interpret. Sass enriches CSS by incorporating additional features to facilitate the creation of well-organized, maintainable, and reusable code. Key features of Sass include: inception, support for variables, nesting style rules, mixins, and inheritance.

Sass has been widely used for the past decade and remains a popular choice for front-end development, but is it still necessary? Many Sass features that were not supported by CSS back in 2006 have been added in recent years. Custom properties, calculations, and grids have been supported by CSS for a couple of years now. Additionally, as of this year, CSS also supports one of our favourite pre-processor features: the ability to nest style rules. Although CSS still does not support Sass features like mixins and inheritance, for most use cases, these are not essential. If needed, alternatives are available. It's worth noting that the Remix framework does not have built-in support for Sass, although it does support alternatives. As a result, we recommend against using Sass in new projects.



focus on
front-end
development