


☐

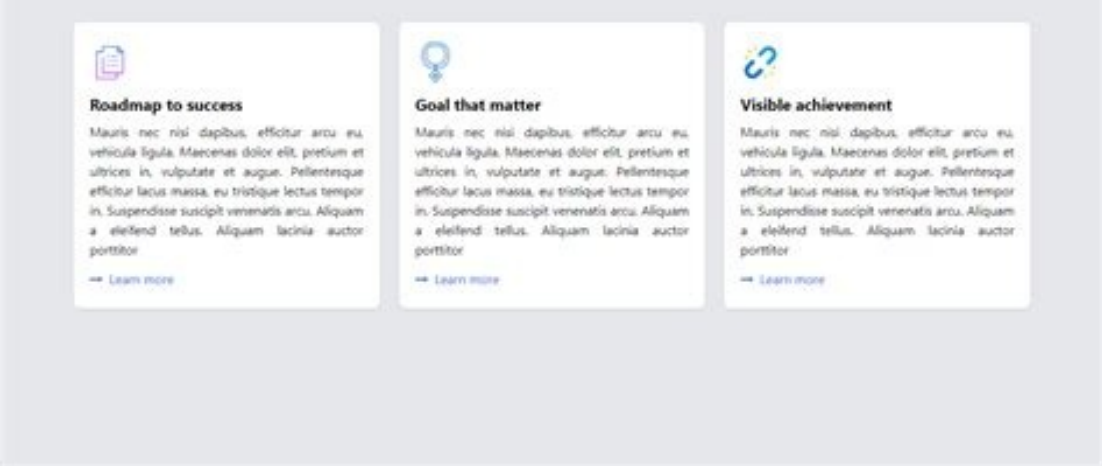
I'm not robot


reCAPTCHA

I'm not robot!

Tailwind css sample sites

In the world of CSS frameworks, the battle for the best approach to styling web applications continues. Some choose CSS modules with SCSS or not, others prefer CSS in JS, and still, others simply install Material Design. There are many possibilities and depending on the needs, everyone will surely find something for themselves.



In this article, however, I would like to focus on a solution that is completely different from those mentioned earlier. I’m talking about a framework recently gaining publicity – Tailwind CSS.In this article, I’m going to give you a really practical overview of the Tailwind CSS framework. And with the framework still going strong and only recently getting the biggest 2023 update with a new changelog template, it’s very much worthing learning more about it. What exactly are you going to get?What you’ll learn – the complete Tailwind CSS tutorial!The festivities include: general thoughts on the framework, including its intended objectives and target audience, its unique approach to writing CSS that may confuse some pros, the usage of pseudoselectors for states, handling responsiveness with the Tailwind CSS framework, writing a whole CSS Tailwind powered web appl, customizing the CSS Tailwind framework, creating your own classes in the framework, as well as trimming classes.Let’s get started.Tailwind CSS doesn’t provide ready-made building blocks for your layouts like Bootstrap or Material Design but also does not focus on writing everything manually offering only syntactic sugar. The creators themselves refer to it as “utility first” because what you mainly get in it is a set of existing classes that control individual aspects of the appearance and behaviour of your application elements. Do you remember mt-2 or invisible from Bootstrap?

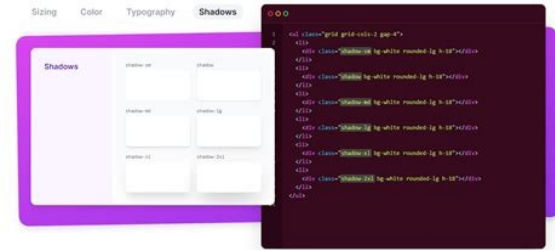


Then imagine Tailwind CSS only consists of this.As I already mentioned, styling an application in Tailwind CSS is based on a set of classes which are modifying one or two CSS attribute each. Each element will have a lot of such classes assigned to it. This may seem crazy at first, but it works pretty well in practice.Paradoxically, I’ve noticed that people who know CSS very well have the most difficulties with migrating to Tailwind CSS. Such developers may get discouraged when they have to check what class they should use to get the desired effect. After all, they could get the same result in a blink of an eye by writing CSS attributes manually, which they know perfectly well.From my own experience, however, I will say that it is worth making this mental shift, because after learning a few rules governing the naming of classes in Tailwind CSS, your work will significantly speed up, and styling the application may turn out to be faster than using traditional CSS.Let’s see how it looks in practice:You have several classes used in this example. Lets check them one-by-one: bg-green-300 - this class sets a green background (this the bg prefix). The value 300 in the name is just the shade number. border-green-600 - it’s similar to the previous class but it sets the elements borders colour to a slightly darker green. border-b - this class does nothing more than set the border to be visible only at the bottom (“b” stands for “bottom”) p-4 and m-4 those two classes make the margin and padding the size of 4 units. In Tailwind CSS, this is equivalent to 16px. rounded - finally, this sets the rounded corners for the element.With all those classes the final effect will look like this:if you don’t like the round corners, you can always adjust them. The rounded class has many varieties, such as rounded-sm or rounded-lg, that give the element corners a smaller or larger radius of rounding, respectively. It is also possible to round off only the selected corners.Also, as you can see, it is impossible to describe all CSS classes and their varieties here. Tailwind CSS supports all popular CSS attributes and has many CSS classes to handle each of them. Shadows, borders, grids, flexboxes, colours and fonts – all of this is well supported. If you are interested, I encourage you to explore the Tailwind CSS documentation.What you saw above are the easiest way to use Tailwind CSS classes. However, styling modern applications is more than just static appearance. Views must be responsive and must change according to the width of the device screen. Elements must also often react to mouse movement and, for example, highlight when hovering over them with the cursor.All such responsiveness and different states of various elements are often the weakest elements of many low-level CSS frameworks. Not that the pure CSS was any better (writing media queries can be a nightmare) but usually, even in frameworks, these syntaxes aren’t exactly elegant.You don’t need a crystal ball to predict that.

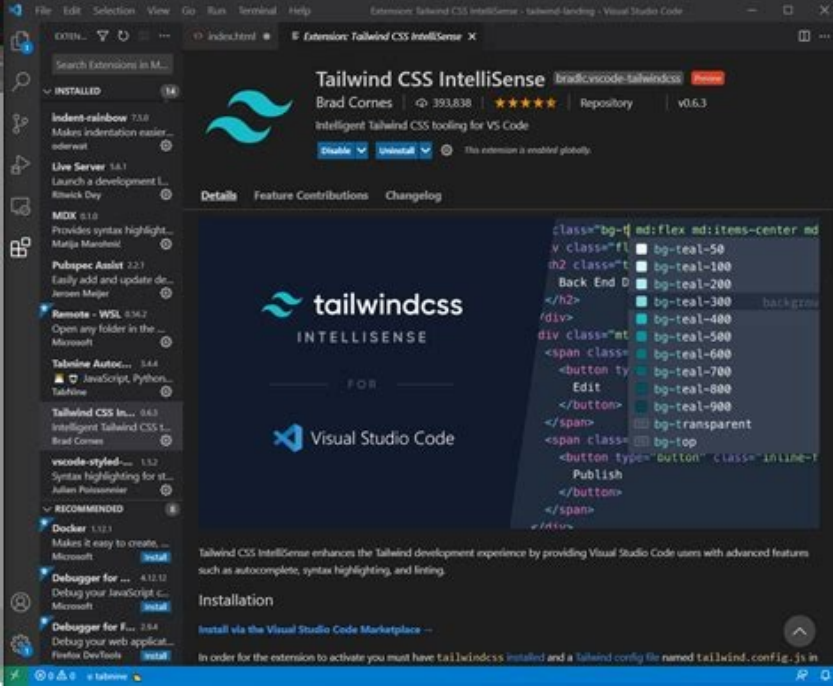


Check out our “State of Frontend” report based on a worldwide survey amongst 4500 developers.However, Tailwind CSS is different. Here you operate on classes and if, for example, you want a class to be active only when the user hovers the cursor over the button, it’s enough to precede such class with the prefix of the CSS pseudoselector, e.g. hover:The above example shows how the hover prefix was used to change the background of an element. Of course, Tailwind CSS supports other pseudo selectors as well. When it comes to more complex cases from the day-to-day battlefield, Tailwind CSS also tries to help. For example, you can very easily change the styles of the descendants of the selected element.For this purpose, you just use the group class in the element that is to be hovered over and the group-hover prefix in the descendant.If you begin to worry about the increasing length of class attributes then please remain calm. We will try to remedy this a bit later.It just so happens that handling responsiveness in Tailwind CSS is also fabulously simple. If you want some classes to be active for a given view, you precede them with a prefix denoting a specific screen size, e.g. ‘sm’ for larger phones, md for tablets and lg for monitors. The class without prefix will then only be used for the smallest screens.In this example, you have created a block – the larger the screen on which you display it, the larger its margin. Of course, nothing prevents you from adding more classes to an element at the same time, preceded by the screen size prefix or some pseudoselector. Different prefixes can also be concatenated to get classes like lg:hover:shadow-md.Here you may ask yourself since Tailwind CSS supports most CSS attributes with dozens of classes and each class can be preceded by one or more prefixes, will my application not grow by several megabytes after adding this framework?And that is a very good question.

It turns out that Tailwind CSS doesn’t generate all possible combinations by default. Moreover, it has the tools to remove unused classes and thus further reduce the size of the resulting source code.But to talk about it in detail, we need to explain how to install Tailwind CSS in your application and how it works first.Tailwind CSS doesn’t work as a regular CSS library. This isn’t just a collection of styles that you include in our CSS. You also need to properly transpile your code using Tailwind classes to get working styles as a result.While there are several ways to use Tailwind CSS, the recommended method is the PostCSS preprocessor and a dedicated plugin that will generate the styles you need. The aforementioned preprocessor works like other tools of this type, e.g. SCSS or LESS: it takes your code and with the help of appropriate logic transpiles it into pure CSS. PostCSS has the advantage of being fully extensible. You can write plugins for it and expand its capabilities depending on your needs. This is a situation similar to Babel or EsLint.If you use PostCSS, the exact way to install Tailwind CSS in your project will really depend on the technology stack, you can easily find the right instruction for installing everything on the Internet. A lot is also covered in the excellent Tailwind CSS documentation.For the purposes of this article, I will discuss how to install the CSS framework in question in projects generated with the popular Create React App starter. Assuming that the project itself has already been created, you can proceed to install the necessary libraries in the appropriate versions. Then, I’m also going to create a HTML file as well as a final CSS file.The problem with CRA is that this tool uses PostCSS internally and doesn’t allow to override its preprocessor configuration in any built-in way. However, this isn’t a hopeless situation because you can use CRACO, a tool for easily overwriting all CRA settings. So install CRACO in your project... and then we modify the commands in package.json so that your application launched with the help of this tool.The next step is to create a settings file for CRACO that will define how PostCSS configuration should be overwritten. For this purpose, in the main folder of your application, create the craco.config.js file and put the following content inside:In the code above we added two plugins to the PostCSS configuration. The first is your Tailwind CSS transpiler to generate all the classes you need, and the second is an autoprefixer that will ensure all CSS attributes will be generated in versions for all browsers you should support.Now that we have PostCSS configured, it’s time to add the Tailwind settings file itself. The fastest way to do this is to generate it with the command:After a while, the tailwind.config.js file will appear in the main folder of your project with the following content:This configuration is empty for now, but you will have time to expand it a bit before the end of the article. Now as long as you only add to the purge field the files in which you are going to use the Tailwind classes. You can use the glob expressions:This setting allows Tailwind CSS to check in the indicated files which classes you actually used in your project and which of them should be added to the production build of our application.



All others will be cut out, reducing the final size of your styles.For this mechanism to work properly, however, you must remember one rule: you mustn’t dynamically concatenate class names anywhere in the code.That is, instead of doing something like this:...you should always use uncut names:Finally, there is one thing left to do. You need to indicate where you want to include this framework in your application. And you can do it in two ways. Either by adding PostCSS understandable instructions in one of your CSS files...or import the all-in-one set into a JS/TS file:The latter solution is preferable if your stack allows you to directly import CSS files to JS. If you think Tailwind CSS may not be the best choice for your purpose, you are afraid it will limit you in some way, you can sleep well. Since all Tailwind classes are generated during the build, you can easily modify them from the configuration file.Let’s take a look at such adjustments of the breakpoints of a responsive layout:With the above code, you have overwritten the default Tailwind breakpoints, i.e. sm, md, lg, etc., and created your own. From now on, in your application, you can start using such classes as tablet:font-weight or desktop:text-white, that are activated on the resolutions indicated in the configuration file.



If you need to use classes that are active up to a given resolution or set some more advanced conditions, then everything is possible:For more examples and instructions on how to expand the breakpoint list without overwriting the default ones, please visit the official documentation pages.Another aspect of Tailwind CSS that many of you will want to customize is the availability of colours. The easiest way to do this is by creating a palette from ready-made sets of shades. For example, if you plan to use red and yellow and a few other primary colours, your palette could look like this:Nothing prevents you from defining your own shades or naming the colours in a specific way:After such a reconfiguration of Tailwind CSS, you can refer to defined shades in the code using classes such as bg-primary-light or text-secondary-dark.Ok, what if you want to further modify the content of Tailwind’s built-in classes? This is also possible and it is also done from the configuration file level. Let’s assume that according to your class shadow-lg casts too soft shadows for your taste and you want to strengthen this effect a bit. You can do it very easily by modifying the appropriate field in the theme settings:Note how we have overridden the styles only for the lg variant, and left the rest of the settings default, referring to the default theme. Of course, other classes can also be modified as shown.One more thing! I’ve mentioned at the beginning of the article that by default Tailwind doesn’t generate all prefix and class combinations. For example, without changing the settings, you cannot use the active:shadow class in your application, which should appear on the button when it is clicked (equivalent to button:active). Tailwind simply doesn’t generate this class for the default settings. How to change it?

Pretty straightforward. That’s why there is a variants field in the configuration, to set additional prefix combinations that we need.So far, you have only modified existing class sets. Tailwind, despite being a very universal tool, doesn’t have classes for absolutely all CSS attributes or cases of their use. Tailwind also doesn’t support e.g. CSS attributes which are not yet supported in all browsers. Can Tailwind help you to create your own classes? As it turns out - yes.You can order Tailwind to generate single classes or whole sets of classes with different prefixes. This is done in a CSS file using the @layers at-rule.For example, Tailwind doesn’t support CSS filters, so if you want to display elements in shades of grey on your website, you can add the missing filter support in the following way:Thanks to this, you can now combine your filter-grayscale and filter-none classes with the hover and focus prefixes and thus, for example, make colourful pictures only when they are indicated by the cursor.Generating classes works great. But in this form, it can be quite a chore to define if you want to do a whole range of classes (e.g. set a different filter power in the range from 0 to 100 in increments of 10). Do you have to write everything manually then? Fortunately, Tailwind comes to your aid here as well.You can write your own utility plugin for this framework that will generate definitions for a whole range of styles.To add such a plug, you have to go back to your configuration file and import a file in the plugins field, where you will store your styles generator.The second step is to write the generator itself that produces all the class descriptions you need. In my scenario, the content of filter.tailwind.js looks like this:When you look at it, there is nothing complicated here. First, you add a class that turns off grayscale to the object, i.e. filter:none. Later in the loop, you add more classes from filter-grayscale-10 all the way to filter-grayscale-100. Finally, the prepared data is sent to the addUtilities function, indicating in the second parameter that the created classes are to support hover prefixes and those related to responsiveness.And that’s it. All 100 classes with variants with different prefixes are ready for use. As you can see, writing Tailwind plugins is super easy. However, before you start doing it, I suggest that you always check if there is any ready-made solution on the Internet. There are many open-source plugins that are just waiting to be used, such as webnda/tailwindcss-aspect-ratio, which I highly recommend.There is one more thing that I promised to mention. How to deal with long, expanding class lists? In the end, sooner or later this will happen to you! used 11 classes to describe this button, and in fact, not much is going on here. And I didn’t even take care of the button’s responsiveness yet! It’s hard to think about what could happen with much more complicated web components.Fortunately, Tailwind CSS has solutions for this. First, you can define in your CSS file a so-called Tailwind component that will collect several atomic classes and combine them into one more complex class. There are two at-rules used here: @layer and @apply.If you don’t like creating components from ready-made classes, you can go back to writing plugins. Component plugins are just as easy to write as utility plugins, and they allow us to define new complex styles using CSS in JS.Creating components has another advantage over using atomic classes directly in your HTML code. When debugging you can better see what part of the code you are looking at because there are more telling names. Therefore, as I practice the method, I always create components to make the application code more readable and easier to develop. Please compare those two buttons yourself. Sample code! it worth getting more interested in using Tailwind CSS? Definitely.

I think this is a tool every web developer should have in their skills toolbox.Why? Even if you think that this framework is not suitable for creating large dedicated applications, it is worth knowing it for building quick prototypes, responsive design, interactive elements, large screens design, navigation bars etc. Once we learn its class nomenclature, including utility classes, Tailwind CSS will immediately speed up creating each project and allow you to focus on business values instead of the details of CSS styling. Among other CSS frameworks, Tailwind CSS really stands out with its ability to deliver business value quickly.Nevertheless, I believe Tailwind CSS also has its place in big projects. Developers don’t need to come up with names for many classes and aren’t obliged to check if someone ely created any utils anymore. When you use Tailwind, you get all the tools right away, including utility first classes, custom designs etc. Writing code with Tailwind CSS is really straightforward!In addition, Tailwind also helps you use a defined colour palette and maintain consistent spacing in the layout. And as we know: the bigger the project, the more people in the team, the harder it is to keep such things in check. Therefore, I propose to give this framework a chance. It will help to maintain the high quality of the code, which is a value for both developers and business.P.S.: I didn’t mention it before, but Tailwind CSS also gives you app a dark mode for free.Our frontend development team routinely delivers custom solutions in technologies such as React, Vue.js, or TypeScript. Check out the portfolio to see examples complete with deliverables and project achievements! I feel like an idiot for not using Tailwind CSS until now.If I had to recommend a way of getting into programming today, it would be HTML + CSS with Tailwind CSS.I have no design skills and with Tailwind I can actually make good looking websites with ease and it’s everything I ever wanted in a CSS framework.Tailwind CSS is the greatest CSS framework on the planet.I started using @tailwindcss. I instantly fell in love with their responsive modifiers, thorough documentation, and how easy it was customizing color palettes.Loved it the very moment I used it.There’s one thing that sucks about @tailwindcss - once you’ve used it on a handful of projects it is a real pain in the ass to write normal CSS again.Okay, I’m officially *all* in on the @tailwindcss hype train.

Never thought building websites could be so ridiculously fast and flexible.Okay, @tailwindcss just clicked for me and now I feel like a #!@%\$%\$ idiot.I’ve been using @tailwindcss the past few months and it’s amazing. I already used some utility classes before, but going utility-first... this is the way.After finally getting to use @tailwindcss on a real client project in the last two weeks I never want to write CSS by hand again. I was a skeptic, but the hype is real.I didn’t think I was going to like @tailwindcss... spent a day using it for a POC. I love it! I wish this had been around when we started, our company design system seriously considering a complete rebuild@tailwindcss looked unpleasant at first, but now I’m hooked on it!Once you start using tailwind, there is no going back! use @tailwindcss for every single project because it removes most of the annoyances of css and is multiple times quicker!t’s changed the trajectory of my business. I’m able to design better looking, better performing, and more accessible components in 1/3 of the time.My first tailwind project worked great but what really kicked ass was going back to it months later and saving so much time making new changes. I knew how everything fit together instantly.Tailwind looked like pure spaghetti until I used it in a real project. Now it’s the only way I make websites. Simple, fast, scalable.Tailwind is a classic example of why you need to put preconceptions aside when evaluating tech. The experience and productivity is streets ahead of what you might have believed based on old school CSS thinking!Tailwind CSS is a framework like no other. Rather than constraining you to a set design, it gives you the tools and the standardization to build exactly what you want.I remember being horrified the first time I saw utility first css. But these past months using Tailwind on an increasing number of projects has just been a joyful new way to build things on the web.I was initially skeptical as I began using @tailwindcss, until I now needed to copy a @sveltejs component to a different location and I didn’t need to worry about any of my styles breaking.@tailwindcss makes you better at CSS. Change my mind.Awesome stuff! I’m no designer or front-end developer; until I found Tailwind last year I hadn’t done any CSS since the early nineties. Tailwind, and Tailwind UI mean I can now create good looking front ends quickly, which is super empowering. Crazy impressive project I admit I was a big skeptic of @tailwindcss until last year. I thought “why would I ever write a million classes that just abstract single CSS properties?” By now, I feel like I’m twice as productive when building UIs. It’s really amazing.I’m nearing completion on my months-long project of rewriting my company’s frontend in TypeScript and @tailwindcss. Still, every time I re-implement a component, I think, “Wow, this was way easier this time.” Tailwind rocks.Co-Founder/CTO @LaunchPathInWith the amount of shipping we have to do, skipping the conversion of brainwaves to CSS, and being able to implement at the speed of thought using Tailwind, my life as a fullstack developer has never been more blissful.Tailwind makes it easy to bring new developers into the frontend project without having to worry about the mental exercise of understanding ‘some’ developer’s class hierarchy and thought process behind it.Tailwind has been a total game-changer for our dev team. It allows us to move faster, keep our UI consistent, and focus on the work we want to do instead of writing CSS.Have been working with CSS for over ten years and Tailwind just makes my life easier. It is still CSS and you use flex, grid, etc. but just quicker to write and maintain.Senior Program Manager at MicrosoftI’ve been writing CSS for over 20 years, and up until 2017, the way I wrote it changed frequently. It’s not a coincidence Tailwind was released the same year. It might look wrong, but spend time with it and you’ll realize semantic CSS was a 20 year mistake.Tailwind makes writing code feel like I’m using a design tool.Tailwind CSS is bridging the gap between design systems and products. It’s becoming the defacto API for styling.Staff Software Engineer @Algolia never want to write regular CSS again. Only @tailwindcss.I came into my job wondering why the last dev would ever implement Tailwind into our projects, within days I was a Tailwind convert and use it for all of my personal projects.Tailwind made me enjoy frontend development again and gave me the confidence that I can realize any design - no matter how complex it may be.Tailwind turned me into a complete stack developer.Tailwind is the easiest and simplest part of any project I work on. I can’t imagine I’ll build anything big without it.Tailwind CSS has alleviated so many problems we’ve all become accustomed to with traditional CSS that it makes you wonder how you ever developed websites without it.Having used other CSS frameworks, I always come back to Tailwind CSS as it gives me the ability to create a consistent and easy to use design system in my projects. Thanks to Tailwind CSS I only need one cup of coffee to get started on a new project.I’ve been using Tailwind CSS for many years, and yet they seem to still amaze us every year with the updates. It’s aided me in building websites super quickly, I could never go back to boring old CSS classes!Tailwind CSS is a design system implementation in pure CSS. It is also configurable. It gives developers super powers. It allows them to build websites with a clean consistent UI out of the box. It integrates well with any web dev framework because it’s just CSS! Genius.It’s super simple to slowly migrate to Tailwind from e.g. Bootstrap by using its prefix feature. Benefiting from its features while not having to spend a lot of time upfront is amazing!I wasn’t comfortable using CSS until I met Tailwind. Its easy to use abstraction combined with excellent documentation are a game changer!Head of Unplatform @aviasolutionsTailwind turns implementing designs from a chore to a joy.You’ll fall in love with building for the web all over again.Tailwind CSS helps you eject from the complexity of abstracting styles away. Having styles right there in your HTML is powerful, which gets even more obvious when using products like Tailwind UI.If Tailwind is like Tachyons on steroids, Tailwind UI is like Lance Armstrong winning the Tour de France (seven times).

Without, of course, the scandal and shame.Being burned by other abandoned CSS frameworks, my biggest fear was to bet on yet another framework that may disappear. However, I gave it a try and couldn’t be happier. They keep improving the framework in meaningful ways on a regular basis. It feels very much alive.Getting buy-in on Tailwind CSS from our entire team of developers took some time and discussion, but once we implemented company wide, it has made it a breeze for any developer to jump into any project and quickly make changes/enhancements.Tailwind CSS has at the same time made CSS enjoyable and drastically changed how I build out products. It’s rapid, efficient and an absolute joy to work with.Full Stack Developer & Designer!Using any CSS framework other than Tailwind seems like a step backward in web development at this point. Absolutely nothing else comes close to making me as productive during the design phase of development than Tailwind!Tailwind has completely revolutionized our devops pipeline. The CLI works consistently no matter what framework is in place.President & CTO @agilitycmsTailwind is like a really nice pair of socks. You think, “okay, how good can a pair of socks be”. Then you put socks on and you are like “%#@! these are socks”. Tailwind unified our css work across different client projects more than any other methodology, while letting us keep our bespoke designs, and even improved performance and stability of my sites.Tailwind is the only way to work with CSS at scale. Tailwind CSS has single-handedly been the biggest and most impactful change for our team’s development workflow.

I’m glad I live in a universe where Tailwind exists.Tailwind solves a complex problem in an elegant way. It provides a ready-to-use UI, all while not compromising on enabling developers to quickly build anything imaginable.Skip to the end. Use @tailwindcss.I was bad at front-end until I discovered Tailwind CSS. I have learnt a lot more about design and CSS itself after I started working with Tailwind. Creating web pages is 5x faster now.Full Stack Web Developer & Tech EducatorI don’t use it but if I would use something I’d use Tailwind!With Tailwind I can offer my clients faster turnaround times on custom WordPress themes, both for initial builds and for future revisions.Thanks to @tailwindcss, CSS started to make sense to me.Core Team Member @laravelphpTailwind clicked for me almost immediately.

I can’t picture myself writing another BEM class ever again. Happy user since the first public release! Productivity is at an all time high, thanks to @tailwindcss.CSS has always been the hardest part of offering a digital service. It made me feel like a bad developer. Tailwind gives me confidence in web development again. Their docs are my first port of call.I thought “Why would I need Tailwind CSS?” I already know CSS and use Bootstrap”, but after giving it a try I decided to switch all my projects to Tailwind.DevOps Engineer & Network AdministratorThe Tailwind docs are its real magic. It is actually better documented than CSS itself! It’s such a pleasure to use.I’ve never felt more confident designing and styling websites and web apps than when I’ve used Tailwind CSS.Tried it once, never looked back. Tailwindcss convert since 0.7 and it just keeps getting better and better.If you work at an agency and deal with hundreds of unique sites, each of them having their own custom CSS is a nightmare. Do your future self a favor and use Tailwind!Before Tailwind CSS I was banging my head against the wall trying to make sense of my CSS spaghetti. Now I am banging my head against the wall wondering why I didn’t try it earlier. My head hurts and my wall has a big hole in it. But at least using CSS is pleasant again!I was skeptical at first and resisted for a long time but after doing the first projects with Tailwind CSS this year, normal CSS just feels wrong and slow.Digital Designer & DeveloperAfter using Tailwind for the first time, I wondered why I used anything else. It’s now my go-to CSS framework for any application, production or prototype.Tailwind not only made me able to focus on building great UI’s but it also improved my overall CSS skills by having such a wonderful docs site when I needed to handwrite CSS.Using Tailwind is an accelerator for rapid prototyping design systems. Strong documentation, helpful community, and instant results.Design Operations Lead at BetterCloudI instinctively hated utility CSS, but Tailwind completely converted me. It reignited my excitement for front-end development and implementing custom designs!Tailwind CSS bridges the gap between design and dev more than anything else. It reintroduces context to development, limits cognitive load with choice architecture, grants access to a token library out of the box and is incredibly easy to pickup. It helped my design career so much.Co-Founder of FeatureBoardWhen I’m working on a project that isn’t using Tailwind, first I yell, then I take a deep breath, then I run npm install tailwindcss.Going back to a large website that doesn’t use Tailwind is like hopping out of a Tesla and into my dad’s rusted Minnesota farm truck. Sure, it works, but the clutch is slipping, the brakes barely work, and it’s filled with old tires we’re not even using anymore.Senior Front-End Developer @CrateandBarrell pushed back hard at the mention of Tailwind initially due to the number of classes in my code however within 5 minutes or using it I was hooked and now am the annoying guy pushing Tailwind on anyone who will listen. It has simplified my dev workflow beyond measurement.Head Engineer @SubscriptionBoxI never bothered to learn vanilla CSS because it’s a waste of time — why bother when I have Tailwind instead? Turns out I learned a ton of CSS anyway just by using Tailwind. It’s such a perfect middleground between thoughtful abstraction, while still letting you break free.Tailwind and the ecosystem around it is like a giant turbocharger for web agencies.

It helps teams of developers and designers develop a shared language and system of constraints that speeds up the entire process. It’s a game-changer for efficient teamwork.Tailwind provides the style of bespoke design, the constraint of a design system, and the flexibility to make it infinitely customizable, without being shoehorned into making every website look like it was cut from the same cloth.Tailwind completely changed my freelance career by allowing me to build out completely custom designs really fast without writing any CSS.Using Tailwind CSS will make you feel like you just unlocked a cheat code.Every developer I’ve convinced to give Tailwind a try has come back and said they are never going back. Every. Single. One.