Welcome to CSS Frameworks. After watching this video, you will be able

to: differentiate between the different types

of CSS frameworks, list the advantages and disadvantages of each

type of popular framework and identify popular CSS frameworks and how to

use them. Recall that a framework provides a foundation

on which you can build your code. CSS frameworks, specifically, can provide

you with all the tools you need to spin up a basic user interface (or UI). This makes it easier to:

Implement various visual elements on a page. Which can include navigation bars, forms,

and grids and to create dynamic and responsive websites,

which can make a website suitable for any screen size. Since frameworks must be used from the creation

of your project, it is important to know about different types of frameworks available. Different frameworks focus on different features,

so it’s important to identify which are most important to your project when deciding

which framework to use. There are two types of CSS frameworks: utility

frameworks and component frameworks. Using no framework at all and just using plain

CSS (also called Vanilla CSS) requires you to write all the styling on your own. This gives you the freedom to style everything

exactly as you want it, but also requires a lot of time and effort, as

you must style every component. An alternative to this is to use a utility

framework, which gives you “utility” classes that

scope to a single CSS property. This makes it easier to apply CSS properties

directly in your HTML code, which can save a lot of time while still giving you the freedom to style components

as you wish. Component frameworks provide you with pre-styled components and

templates which are easy to add to any website. This requires little knowledge of CSS and

makes it easy to keep consistent styles, but also limits you to only the components

made available by the framework. Utility-first frameworks provide you with an easy way to reference

CSS properties. These typically come in the form of classes,

called utility classes, which scope to single-purpose CSS classes. Instead of having to write out the entire

CSS property, utility-first frameworks allow you to use a property by referencing

its corresponding class within the “class” attribute of your desired HTML element. For example, instead of using the “text-align:

center;” CSS property in your code, a utility-first framework might have a self-descriptive

class, such as “text-center”, which does the same thing when added to the ”class”

attribute of an HTML element. They make it easy to be consistent with color

choices, spacing, typography, shadows, and everything else that makes up a well-engineered

design system. However, having these styles mixed into your

HTML classes reduces the separations of concern within your code, making your HTML markup

more verbose. Since utility-first frameworks involve adding

many classes to your HTML markup, this often causes the download size of your

markup to increase, and potentially slows down your web pages. A popular utility-first CSS framework used

today is Tailwind CSS. It is packed with classes that can be composed

to build any design directly in your markup. For example,

say you want to style a link to indicate it’s dangerous. You could underline the link text and change

the text color to be red, as well as make the link a darker shade of red when hovered

over. Doing this in vanilla CSS would require this code, where you need to specify what

the link looks regularly and when hovered over in two separate sections. However, with Tailwind CSS, it can easily

be done within the “class” attribute, directly

in the HTML, which can save a developer time. The “hover” portion of the Tailwind CSS

is a special type of class, called a modifier, which can be used with any other Tailwind

class and is only applied when a user hovers over that

element. There are many instances of modifiers in Tailwind

CSS, some of which can make it very easy to create responsive websites fit to any screen

size. For example, adding “md:” before a class will only apply the class when a user’s

screen size is greater than 768 pixels wide, which is defined by Tailwind as a medium size

screen. This code will display an image with a width

of 16 (64px) by default, a width of 32 (128px) on medium screens, and a width of 48 (192px)

on large screens. This saves developers a lot of time when styling

many different elements, allowing them more time to focus on the back-end. It also provides them the freedom to style

elements however they want. Component frameworks provide pre-styled components

which can be easily added to your code. This results in the ability to develop well-styled

websites rapidly, as significantly less time needs to be spent styling each element. It also makes it easier to keep all related

elements styled uniformly, as you can simply choose the same or similar styles each time. However, having all these pre-defined styles

limits you only to what the framework provides, and doesn’t give you the freedom of customizing

everything exactly as you want it. They also provide a lot of overhead code that

you wouldn’t otherwise get if you choose not to use any frameworks, as component frameworks

will often provide you with more components than what you’ll use. One of the most popular component CSS frameworks

in use today is called Bootstrap. It allows you to build fast, responsive sites

with a feature-packed frontend toolkit. If we take our example of adding a red danger

link to your webpage which we saw earlier, we can see how much easier this would be with

Bootstrap. Recall how to add this with Vanilla CSS and

Tailwind CSS. Bootstrap provides a red danger link as one

of its many components, and so adding this can simply be done by using the “link-danger”

class. Seeing all these side by side, it becomes

clear that for this example, bootstrap has the easiest implementation of

a danger link. However, this may not always be the case. In addition to providing individual HTML components,

such as buttons and links, Bootstrap also supplies entire templates which

range from anything as comprehensive as an entire website, to a simple sign-in page. As a whole, all of Bootstrap’s components

and features save time designing webpages and make it easy to remain consistent with

styles throughout a website. It’s ease-of-use also means that you don’t

need to be an expert in CSS to use it, only limited knowledge is needed to use its basic features! However, it is important to keep in mind that

it’s limited in the components it can provide, and may not have the ones you’re looking

for. In this video, you learned CSS frameworks make it easier to implement

UI elements and create mobile-friendly webpages. Plain, or Vanilla, CSS lets you write all

the styles and layouts of a website on your own. Utility-First Frameworks provide you with

utility classes to help you build your own styles and layouts. Tailwind CSS consists of classes that are

composed to build designs directly on the markup. Component Frameworks provide you with a wide

selection of pre-styled components and templates, which can be implemented onto your website. Bootstrap allows developers to build fast,

responsive sites with its feature-packed front-end toolkit.