MALE: Now we're ready to put everything into practice that we have learned.

This is what our form will look like when we're done.

It's a form for proposing a new online course for a digital training company.

Please start by downloading a copy of the Start file and unzipping it.

Then open the text editor of your choice.

I will be using Brackets.

Open the index file, and we'll start by creating an open and closing form tag.

When you fill out a form and click on the button that submits the data,

two things typically happen.

First, the data is moved from the web form to a processing script page.

Here, the data could be added to a database or sent in an e-mail message.

Next the user is shown a page that informs them that their submission was received.

You will need to instruct the form what method to use when the data

is being transferred from the form to the backend.

One option is to use Get, which is sent through a URL.

Here's an example of Get from YouTube.

Do you see the question mark followed by a name-value pair?

Another option is to use Post,

which is hidden from view and slightly more secure.

If you're using GitHub to host your pages, do not use Post.

If you're using a web-hosting company like BlueHost, then either one would work.

In this page, we're not going to worry about processing of the form data.

So, we'll skip that middle step and just show a Thanks page.

Let's add the two parameters we just talked about to the opening form tag.

Please add method equals get.

Next, add action equals thanks dot html.

If you look down here, you can see that I already have a Thanks page ready for you.

We are going to be building a fairly long form so that we

can practice lots of different form input types.

Therefore, we will follow one of our best practices by

breaking our form into chunks using field sets.

Please type an opening and closing field set tag.

Just inside the field set, add an open and closing legend with applicant information.

This is what it looks like.

We'll make it pretty later on using CSS.

Every input needs to consider six things: the input type;

Are you asking for text, for numbers, for a date, for selection from a list?

the name of the input--each name needs to be unique--the instructions

to the user so they know what information you are

asking for; the value--sometimes this is supplied by the user, such as a person's name.

Other times, it is supplied by the page and the user just clicks a box--whether

this item is required or not; and finally, it may be helpful to provide

an example of the information you're looking for.

Let's build our first input and make it a type equals text.

Notice that there is not a closing input tag.

This is what we have so far.

You can type, but at this point the user has no idea what is being asked of them.

Let's add name equals F name, which stands for "first name."

If you use the correct name, it helps the browser auto-fill your form,

which can save a lot of time and make your customers happy.

Google has provided a list of names to use to help with auto-complete.

We will also add user instructions before the input of first name.

Let's also add an asterisk to indicate that this is a required field.

Now when I look at my page,

it's pretty clear that I should type my first name in the box.

I really don't need to use a placeholder for first name.

I think it's pretty obvious.

I also want to add "required" to the input, and you do that by just typing required.

This makes it so the form will not be submitted without something being typed in here.

Now let's demonstrate the label.

When I am looking at a web page,

I click the input; you can see it as a blue outline indicating that I can now type.

If I click the word "first" or "name," it does not get a blue outline.

Let's make the input work by clicking on either the user instructions or the input box.

Please add an opening label before the user instructions.

Then move the closing label after the input.

Now when I click the user instructions, the input activates with the blue line.

While this may not be very impressive now,

it will become more important when we use radio buttons with your fingers as the pointer.

Please copy the label for the first name and paste it below itself.

Change the user instructions to last name and the name to L name.

This is what it looks like so far.

Please try to ignore the ugly layout;

we'll fix that using CSS.

Now let's copy and paste again for the phone.

Change the user instructions to "phone" and remove the asterisk, since this is not required.

Change the type to tel, which is short for "telephone."

Change the name to phone.

Remember to remove the required attribute from the input.

Let's add a placeholder for a phone of

123-555-1234 to let them know that they do not need to include dashes.

Now we're ready for the e-mail.

Please copy the phone and paste it again.

Change the user instructions to e-mail with an asterisk and the type to e-mail.

Also change the name to "e-mail," and add a placeholder to someone at gmail dot com.

Finally, we need to add the required attribute back.

Let's do one more.

Please copy and paste again.

Change the user instructions to website.

Change the type and name to URL.

Change the placeholder to HTTP colon forward slash forward slash.

This is what we have so far, and section 1 is done.

You may be wondering why we are using

different types when they look exactly the same in the browser?

That's a good question.

The answer can be seen when you view the page on a mobile device.

Now there will be some differences depending on the age of your device,

if it's a phone or a tablet, and of course what brand it is.

Here's what I see on my iPhone.

A type of "text" will show you the full keyboard with the space bar.

A type of "tel" will give you a phone keypad.

A type of "e-mail" will add the at symbol and a period in the bottom row.

A type of "URL" will remove the space bar

add a forward slash and a dot com.

Now let's move on to section 2 of our form.

Please create another field set and legend, and we'll title this "Course Information."

This time we're going to create a radio group.

We still need user instructions for this group, so we'll place that in a division tag.

Start with a self-closing input tag, then add type equals radio, name equals audience.

Here we can see a dot.

Since the user cannot type inside a radio button,

we must supply the value ourselves.

In your HTML, add value equals beginner.

The little circle is meaningless without user instructions,

but this time we'll place them after the input.

Notice that when we click on the dot, it activates.

However, when we clicked the word "beginner" nothing happened.

This is a situation where wrapping it in a label tag is really important.

Please return to the HTML.

Add an opening and closing label tag.

Now when we click the word "beginner,"

the radio button activates.

Please copy that label and paste it two more times below itself.

Change the value and user instructions to

intermediate and then advanced for the last one.

Notice that we left the name identical for all three.

This allows us to click one and have the other one deactivate.

If I were to change the names so they were different like this then you see that I can

select them all at once, and that defeats the whole purpose of a radio button.

I will just undo that so they match again.

Below this section, we will have a series of check boxes which are designed to

aelect one, two, or more.

Add a division for software needed.

Next copy and paste one of the radio buttons from above.

Change the type to check box.

The value to "yes," meaning it has been checked.

Change the name to editing.

Finally, the user instructions to photo editing.

If we look at the result,

we can see that there is now a square next to photo editing, and we can

click the word "photo editing" or the checkbox to make it respond.

We need four of these, so copy and paste three more times.

On the second one, change the name to "code" and the value to "coding."

On the third one, change the name to "process" and the value to "preprocessor."

Finally, change the last one to "client" and "FTP client."

Now we have four functioning but really ugly check boxes ready to go.

Please create the third section with a legend of "funding."

Let's start this one with a label, then add user instructions of

"anticipated completion date" followed by

an open input tag with a type of "date" and a name of "completed date."

Let's take a look.

Notice I get a date picker when I click this input.

Next let's add a drop-down,

which we have not used yet.

Again we start with a label and core subject for the user instructions.

Instead of using an input,

we will use a select with a name of "subject."

Each option in the drop-down will become an option with a value.

According to our best practices,

we would not use a drop-down unless there were more than five options.

So, I will copy and paste in eight of them.

Now you can use as many as you want or as you have patience to type.

When we refresh the page, we can see that the first option,

software development, is already selected.

This is not good.

If the user were to forget to make a selection, then

this option would be submitted, and we'd get a value that was not correct.

Therefore, we need to create another option.

Please jump back to your HTML and add an option that says "please select,"

and it has no value.

Now we will not have an inaccurate response if the user forgets to make a choice.

The last kind of input we need is number.

Please copy and paste the completion date and paste it below the closing label.

Change the instructions to "How many authors?"

Change the type to "number" and the name to "authors."

In order for this form to process, we need a button at the bottom.

After the closing field set but above the closing form tag,

add an input of type "submit."

We then say to ourselves,

"I want to, in this case, propose my course."

We'll then set the value of this input to propose my course.

Open the HTML page and click the button.

Notice that it will not advance to the Thanks page

without the required fields being filled in.

I have a user profile set-up for Zen.

The browser auto-fills the first name, the last name, the phone, and the e-mail.

Now when I click the button, I see the Thanks Page.

Our form is now complete, it just looks terrible.

In the next video, we'll do some serious CSS magic.