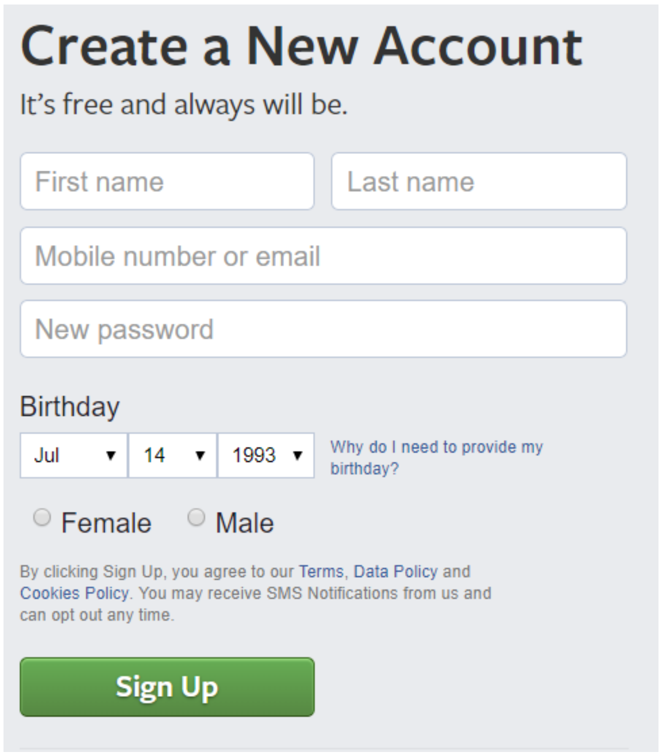
Week 2_4(en)

HTML Forms

> Let's learn how to make HTML forms where users can enter different data into different fields!

There are times we need to make the user fill in some information and submit it to the server.
To do this, we need entries and a submit button like below.



Create a New Account

It's free and always will be.

First name	Last name
Mobile number or email	
New password	

Birthday

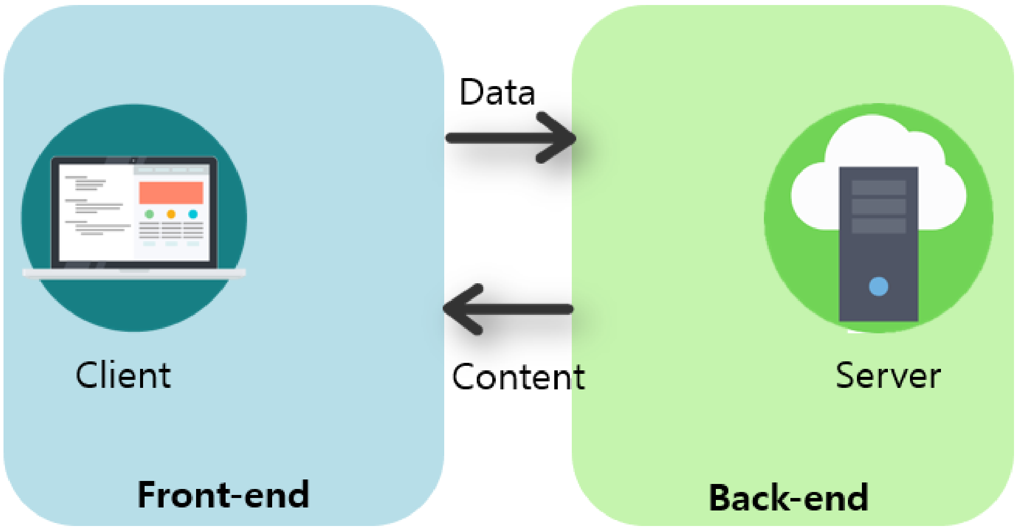
Jul ▾ 14 ▾ 1993 ▾ Why do I need to provide my birthday?

○ Female ○ Male

By clicking Sign Up, you agree to our Terms, Data Policy and Cookies Policy. You may receive SMS Notifications from us and can opt out any time.

Sign Up

While actually sending the data and saving it in the server is related to the back end, which is out of the scope of this course, showing the view and designing the user experience side like above is our job.



Before starting, let's make a new file named `form.html`.

Form with Simple Input

Making Entries

We use the `<form>` tag to indicate a whole form with one submit button.
An `<input>` tag creates an input entry, and we use the `<label>` tag to indicate the label of the entry.

```
<form>
  <label>First Name:</label>
  <input type="text">
</form>
```

As you can see, the `<input>` element is an empty element.
This might look good since the label and the entry comes side by side visually.
But we need to associate the label of the input and the `<input>` element in a code perspective.
We want to make the cursor automatically move to the field when we click on the label name.
To do this, we set a `for` attribute of the `<label>` tag as some kind of value and give the same value to the `id` attribute of the `<input>` tag.

```
<form>
  <label for="firstName">First Name:</label>
  <input type="text" id="firstName">
</form>
```

Check it on your browser!
An alternative way to associate the label and the entry is to nest the `<input>` element inside the `<label>` element like below.

```
<form>
  <label for="firstName">
    First Name:
    <input type="text" id="firstName">
  </label>
</form>
```

This looks exactly same as the prior code, but this code syntax is *not supported on some screen readers.*
So I recommend you to use the first practice.

Let's add another label and input for last name.

Making a Button

Then we need to add the submit button. To do this, we use the `<input>` tag with the `type` attribute set as `"submit"`.
You can put the button label as the `value` attribute.

```
<input type="submit" value="Save!">
```

Remember that the submit button will not actually work!
This is related to the back end, and here we are only making the part the user can view.

However, there is one final attribute that we need to add to each of our inputs to make the form accessible to programming languages (back end) so they can actually do something useful with the data.
This is the `name` attribute.

We use the `name` attribute for each input field (excluding the one for submit button) to indicate the name of each entry, by which the back end developer will access each entry.

```
<form>
  <label for="firstName">First Name:</label>
  <input type="text" id="firstName" name="firstName">

  <label for="lastName">Last Name:</label>
  <input type="text" id="lastName" name="lastName">

  <input type="submit" value="Save!">
</form>
```

Different Types of Inputs

Email

While you can simply use the `"text"` type `<input>` tag to receive an email from the user, there is a special element type of input specifically for emails, which is the `"email"` type.

```
<input type="email">
```

Let's add tags and right attributes as we learned above to complete the form.

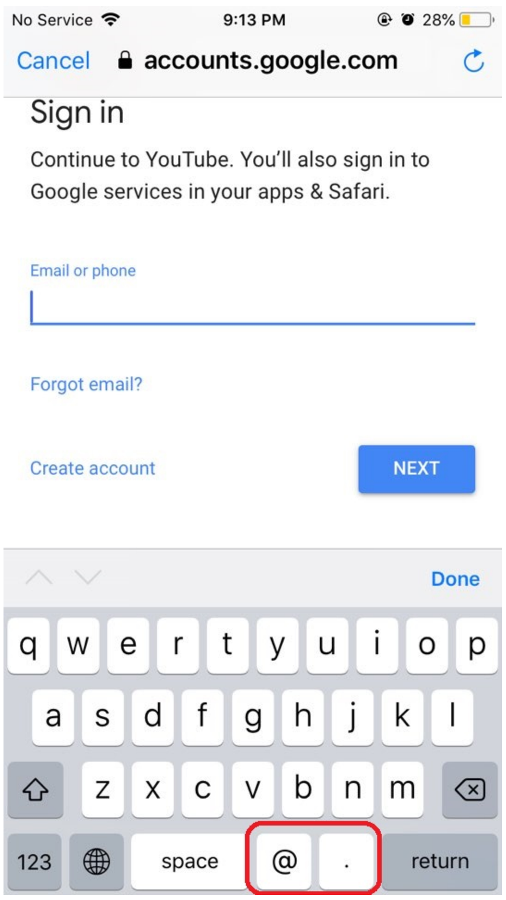
```
  <label for="email">Email:</label>
  <input type="email" id="email" name="email">

  <input type="submit" value="Save!">
```

It is useful to use an `email` type input because

- The browser automatically validates the content.

  If the user tries to submit text that is not in the form of an email, it will alert the user of this.
- It shows the right keyboard on touchscreen devices.

  For touchscreen users, the `@` and `.` symbols will come out automatically on the onscreen keyboard when typing in an `email` type input.



Sign in

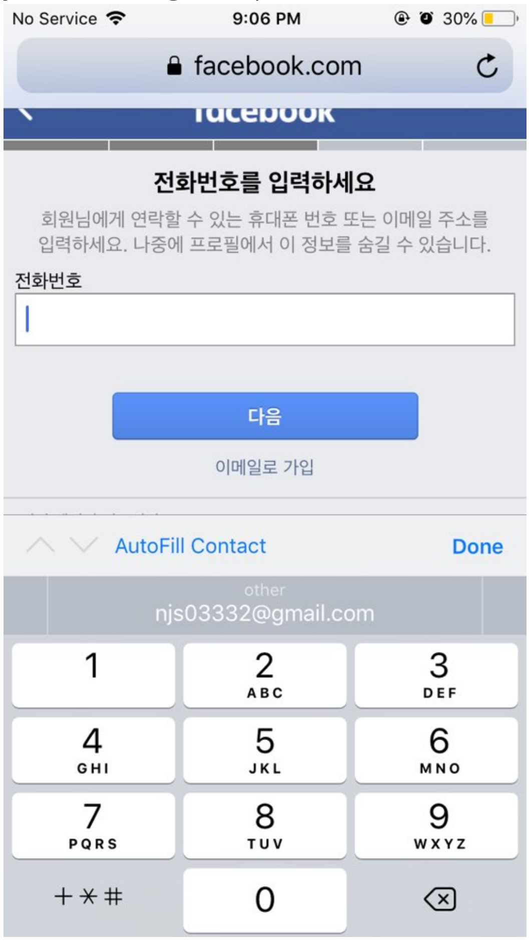
Continue to YouTube. You'll also sign in to Google services in your apps & Safari.

Telephone Number

We use the input type `"tel"` for entries for telephone numbers.
Applying what we learned above, we can write the code below.

```
<label for="telephone">Telephone:</label>
<input type="tel" id="telephone" name = "telephone">

<input type="submit" value="Save!">
```

If we use the `<input>` tag with the `type` value `"tel"`, the onscreen keyboard will resemble that of when you are dialing a telephone number.



전화번호를 입력하세요

회원님에게 연락할 수 있는 휴대폰 번호 또는 이메일 주소를 입력하세요. 나중에 프로필에서 이 정보를 숨길 수 있습니다.

See more types of inputs

Receiving Multiple Lines of Text

There is a special tag `<textarea>` for entries in which you want to receive multiple lines of text.

- The `<textarea>` tag is not an empty element, as opposed to the `<input>` element.
- By putting content inside a `<textarea>` element, we can set the default input of the entry.
- The `<textarea>` element does not need the `type` attribute, because it is its own unique element.

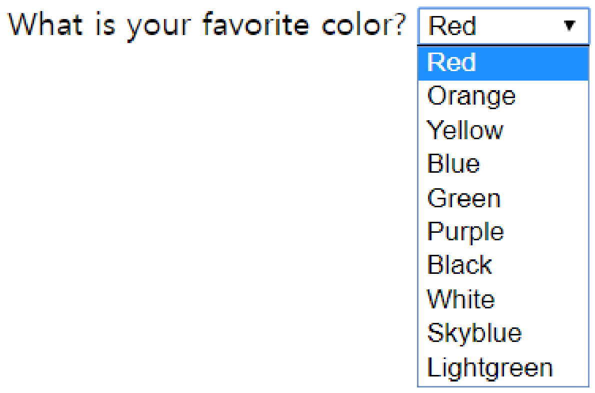
```
<label for="yourMessage">Your Message:</label>
<textarea id="yourMessage" name="yourMessage">Thank you.</textarea>
```

Check it on your browser!
The user can write multiple lines or paragraphs by pressing enter inside the entry.

Choosing Between a Set of Options

There are 3 types of form fields which let users choose between a set of options.

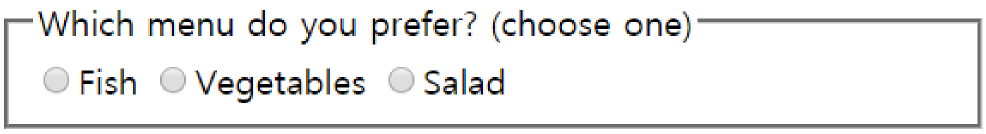
1. Select (dropdown)



The selectbox is useful when

- users already know what they are going to select
- there are so many options that might cover a lot of space

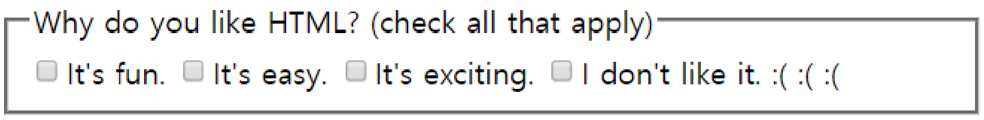
2. Radio



The radio field is useful when

- it is an important decison
- there are few options and the screen can afford it

3. Checkbox


Why do you like HTML? (check all that apply)
It's fun.
It's easy.
It's exciting.
I don't like it. :(:(:(

The checkbox field is useful when you want to let the user select 2 or more fields simultaneously.

Select Field

We use the `<select>` tag to wrap the options and the `<option>` tag to indicate each option.

```
<select>
  <option>Red</option>
  <option>Green</option>
  <option>Blue</option>
</select>
```

We need to give each option a `value` attribute which contains the actual data we want to pass along the server (as apposed to the content the user sees).
Let's say in this case, we need to send the data of the choice in the form of `colorRed` for the Red option, `colorGreen` for the Green option, and so on.

```html
<select>
  <option value="colorRed">Red</option>
  <option value="colorGreen">Green</option>
  <option value="colorBlue">Blue</option>
</select>
```

Now let's add a `name` attribute to the `<select>` tag, through which the back end developer can access the tag.

```html
<select name="favoriteColor">
```

Let's add a label for this set of options and associate the label and the options by using the `for` and `id` attribute as we learned above.

```html
<label for="favoriteColor">What is your favorite color?</label>
<select id="favoriteColor" name="favoriteColor">
  <option value="colorRed">Red</option>
  <option value="colorGreen">Green</option>
  <option value="colorBlue">Blue</option>
</select>
```

Radio Field

By typing the code below, you will be able to see a single radio button on your browser.

```html
<input type="radio">
```

Let's add labels.

```html
  <input type="radio" id="fish"><label for="fish">Fish</label>
  <input type="radio" id="vegetables"><label for="vegetables">Vegetables</label>
  <input type="radio" id="salad"><label for="salad">Salad</label>
```

By adding the same name to each `<input>` elements, the browser will know that the three options are **associated** and that the user should be able to **select only one** of the three.
Also, just like when using select fields, we need to indicate the actual data to be sent to the server using the `value` attribute.

```html
  <input type="radio" id="fish" name="favoriteMenu" value="menu1"><label for="fish">Fish</label>
  <input type="radio" id="vegetables" name="favoriteMenu" value="menu2"><label
for="vegetables">Vegetables</label>
  <input type="radio" id="salad" name="favoriteMenu" value="menu3"><label
for="salad">Salad</label>
```

Now let's add the question for these set of fields.
We use the `<legend>` tag to indicate the question of the choices and the `<fieldset>` tag to wrap up the question and the choices together.

```html
<fieldset>
  <legend>Which menu do you prefer? (choose one)</legend>
  <input type="radio" id="fish" name="favoriteMenu" value="menu1"><label for="fish">Fish</label>
  <input type="radio" id="vegetables" name="favoriteMenu" value="menu2"><label
for="vegetables">Vegetables</label>
  <input type="radio" id="salad" name="favoriteMenu" value="menu3"><label
for="salad">Salad</label>
</fieldset>
```

Checkbox Field

The checkbox field goes almost exactly the same way as the radio field.
We use the `radio` type input instead.

```html
<fieldset>
  <legend>Why do you like HTML? (check all that apply)</legend>
  <input type="checkbox" id="fun" name="htmlReason" value="reason1"><label for="fun">It's fun.
</label>
  <input type="checkbox" id="easy" name="htmlReason" value="reason2"><label for="easy">It's
easy.</label>
  <input type="checkbox" id="exciting" name="htmlReason" value="reason3"><label
for="exciting">It's exciting.</label>
  <input type="checkbox" id="no" name="htmlReason" value="reason4"><label for="no">I don't like
it. :( :( :(</label>
</fieldset>
```