

# Tutoriales HTML, CSS y JavaScript

Grado en Ciencia de Datos e IA

Isaac Laaouaj  
Curso 2023 - 2024  
12/12/2023

# Índice

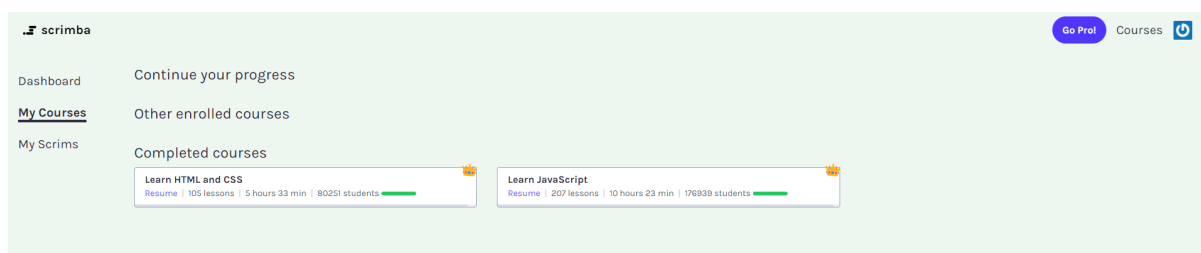
<b>1. Realización de los tutoriales</b>	<b>3</b>
1.1 HTML and CSS	3
1.2 Javascript	8
<b>2. Conclusión</b>	<b>13</b>

# 1. Realización de los tutoriales

Los tutoriales de Scrimba son en forma de video, con interacción directa con el código, es decir mientras avanzas con los videos a su vez puedes editar el código de manera personalizada.

Se ha realizado la parte gratuita de los siguientes tutoriales:

- **Learn HTML and CSS:** <https://scrimba.com/learn/htmlandcss>
- **Learn JavaScript:** <https://scrimba.com/learn/learnjavascript>



## 1.1 HTML and CSS

Este curso consta de 6 secciones, cada una de ellas diseñada para brindar un aprendizaje específico. Cada apartado contiene videos didácticos que cubren detalladamente el contenido presentado:

### What's inside

This course contains 105 interactive scrims spread across 6 modules.

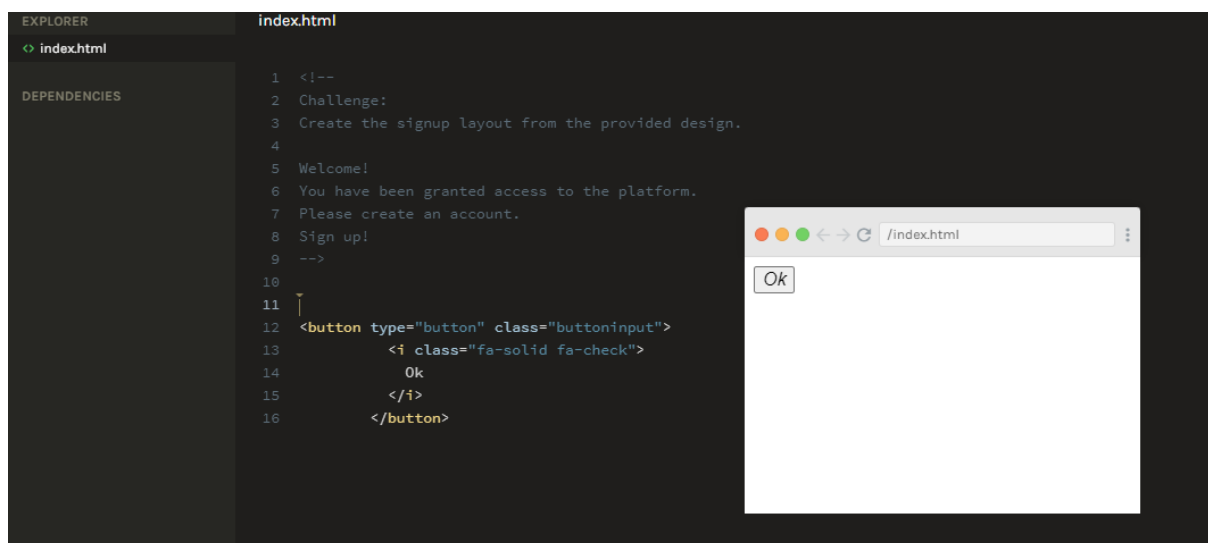
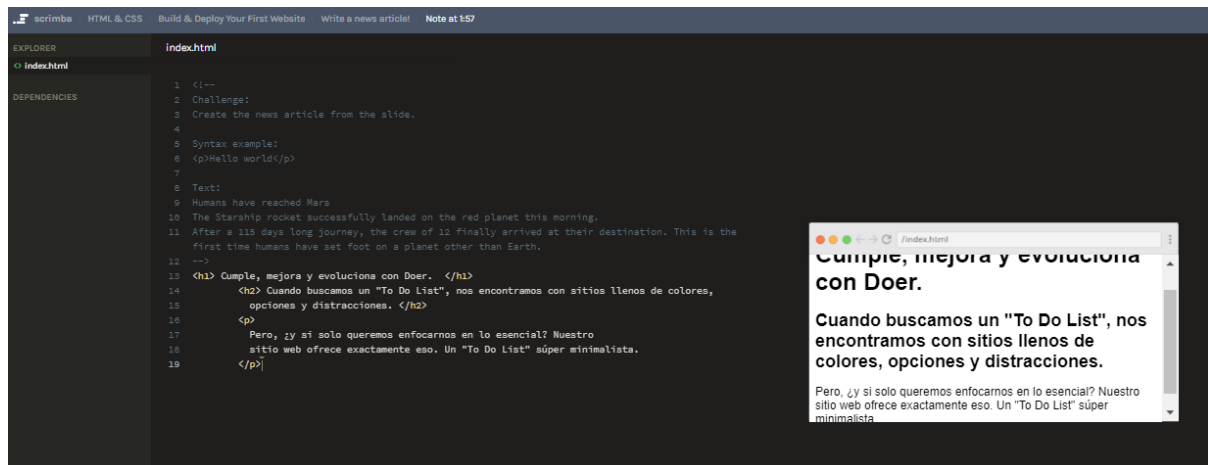


View/Hide All

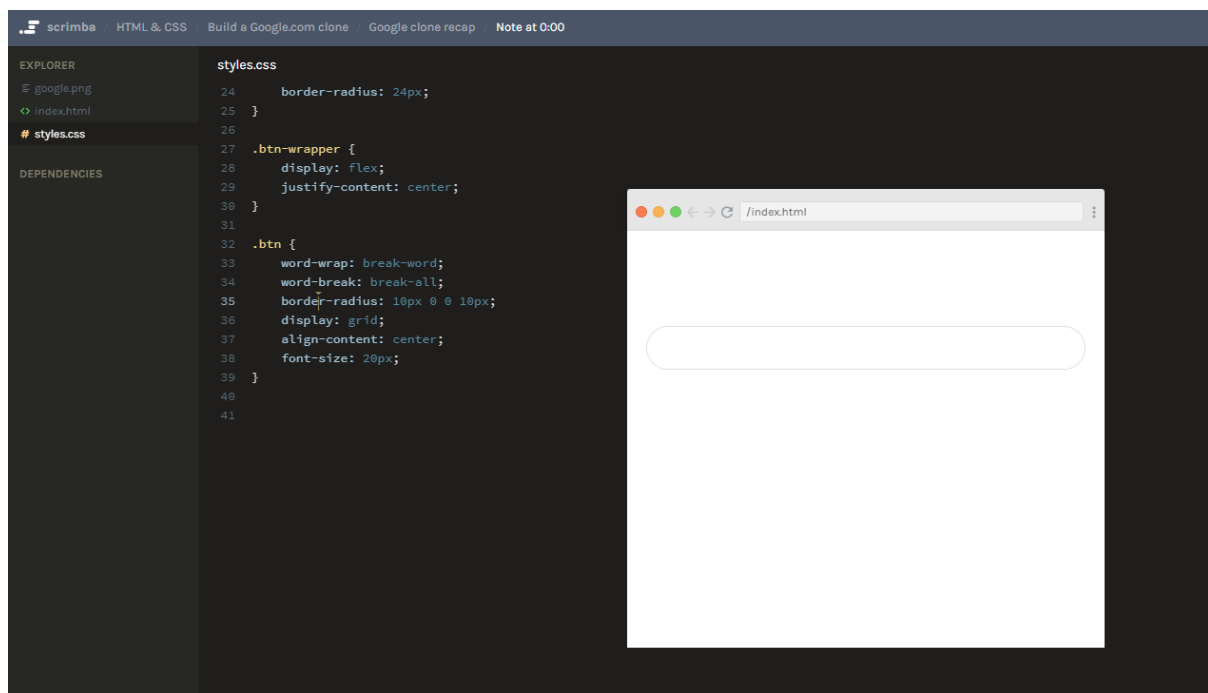
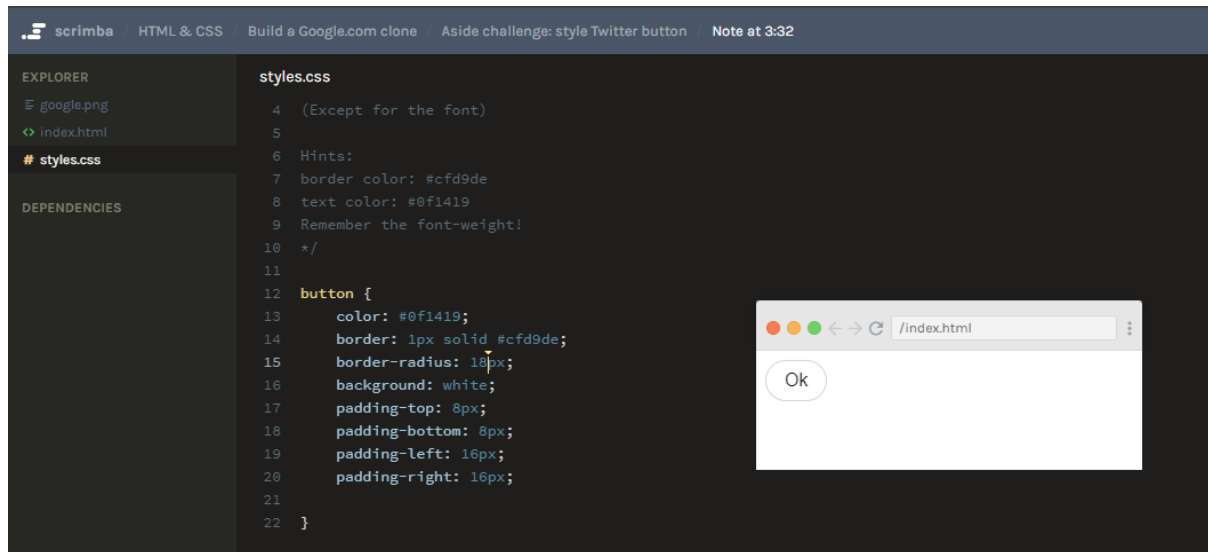
>	Build & Deploy Your First Website 100%	16 lessons 1 hour
>	Build a Google.com clone 98%	26 lessons 1 hour 21 min
>	Build a Digital Business Card 96%	20 lessons 58 min
>	Build a Space Exploration Site 97%	15 lessons 44 min
>	Build a Birthday GIFT Site 100%	23 lessons 1 hour 14 min
>	Solo Project: Hometown Homepage 73%	5 lessons 14 min

El tutorial de HTML y CSS ha sido de utilidad para poder realizar la parte de HTML y CSS necesaria para el trabajo final de la asignatura (Doer list), concretamente:

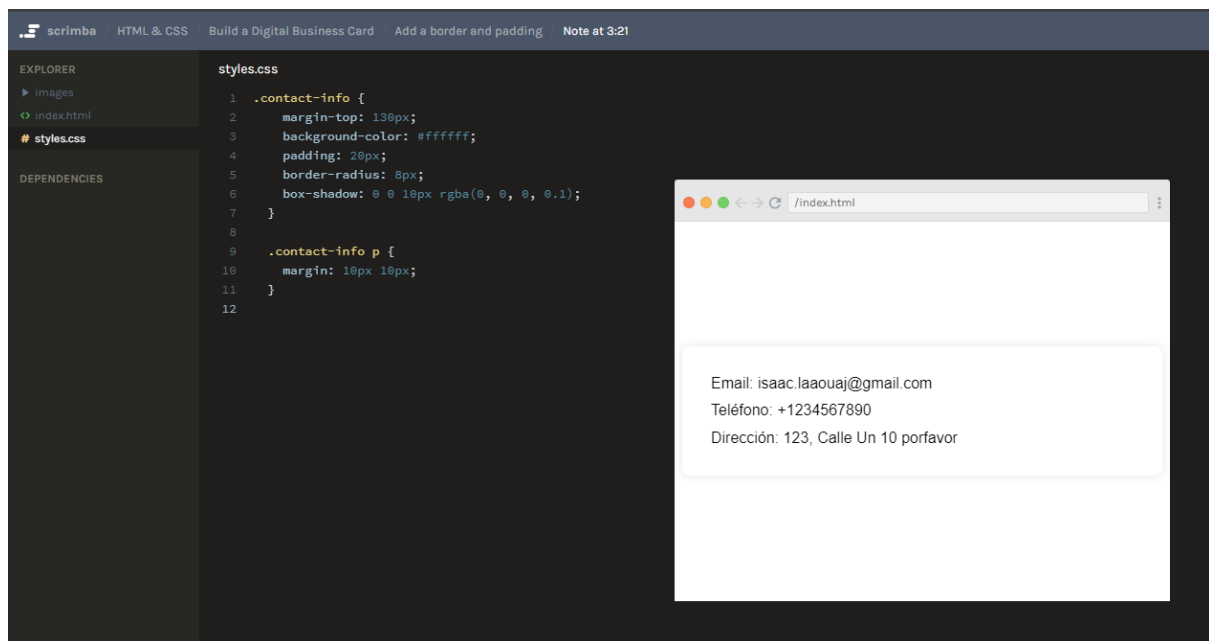
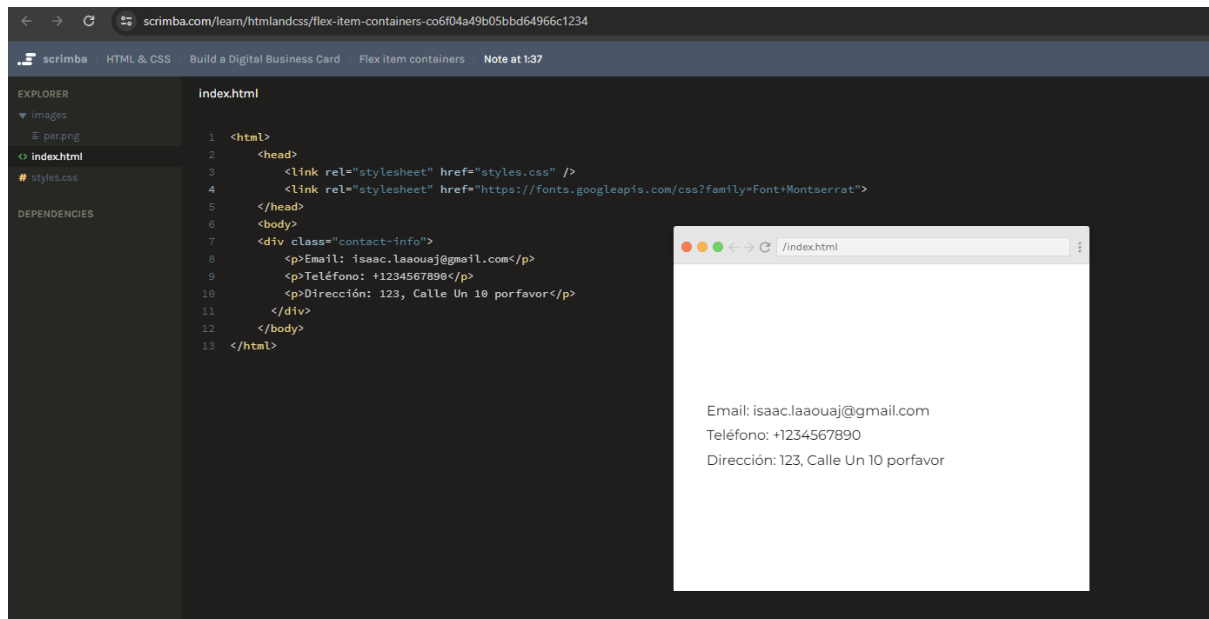
- Del primer apartado **“Build and Deploy your first Website”**: Se ha utilizado para realizar los títulos, segundos títulos, contenido en los párrafos, las imágenes mediante el tag **“<img>”**, la construcción de botones y la estructura básica HTML:



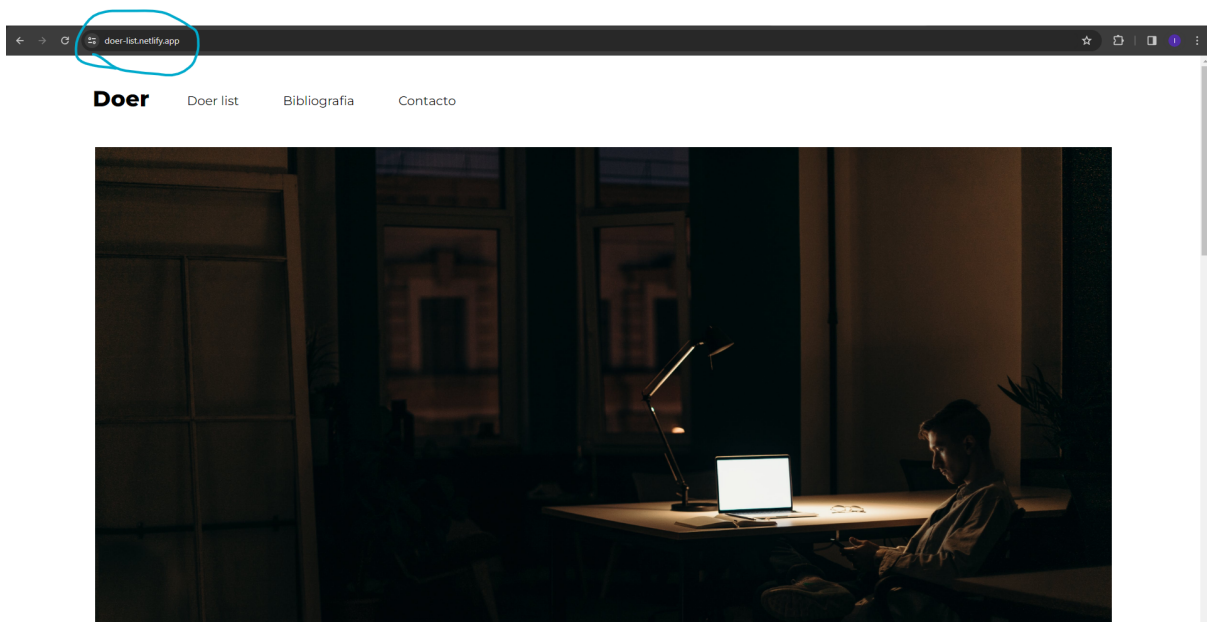
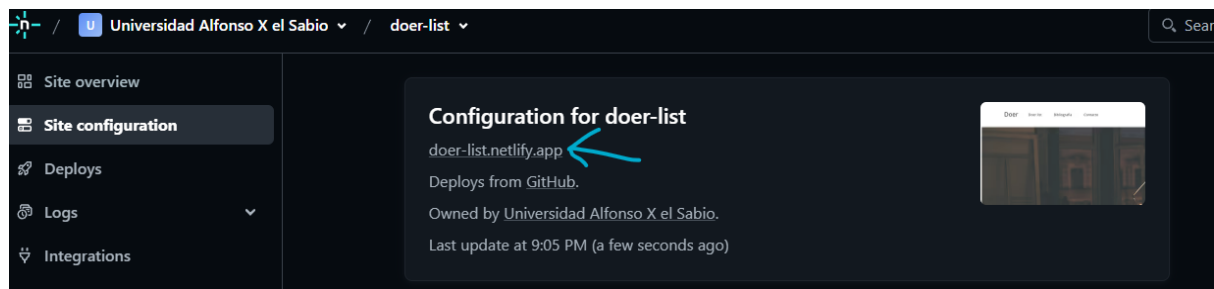
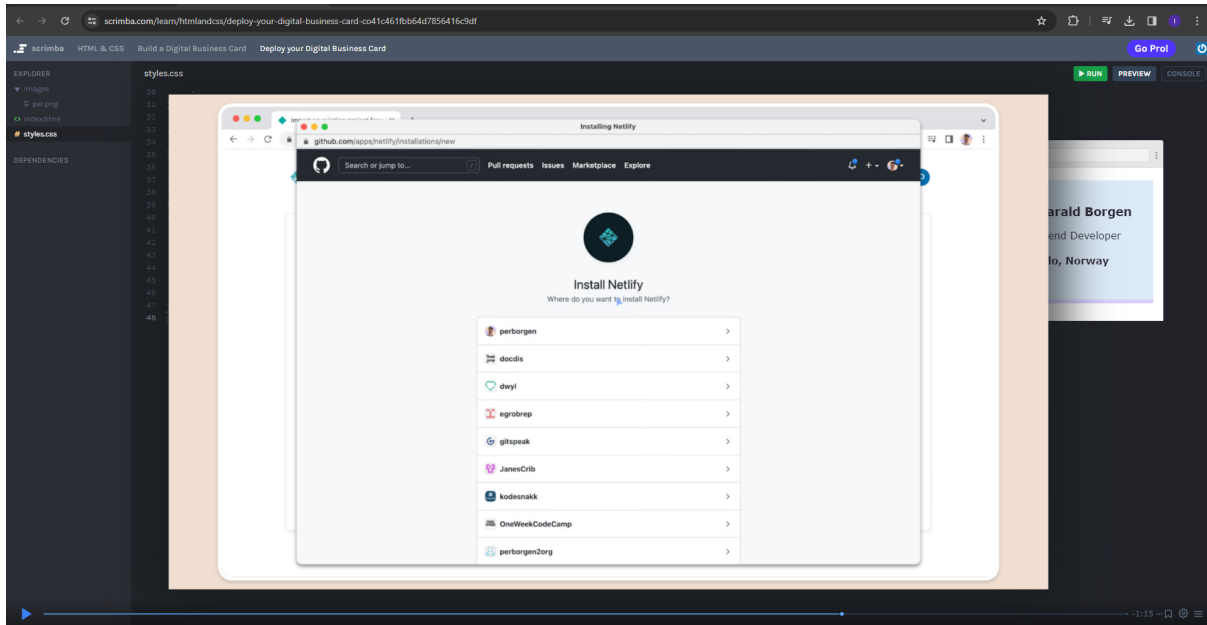
- Se ha utilizado el video **“Build a Google.com clone”** para la construcción del HTML y CSS de los botones y de la barra de inserción de la tarea de “Doer list”:



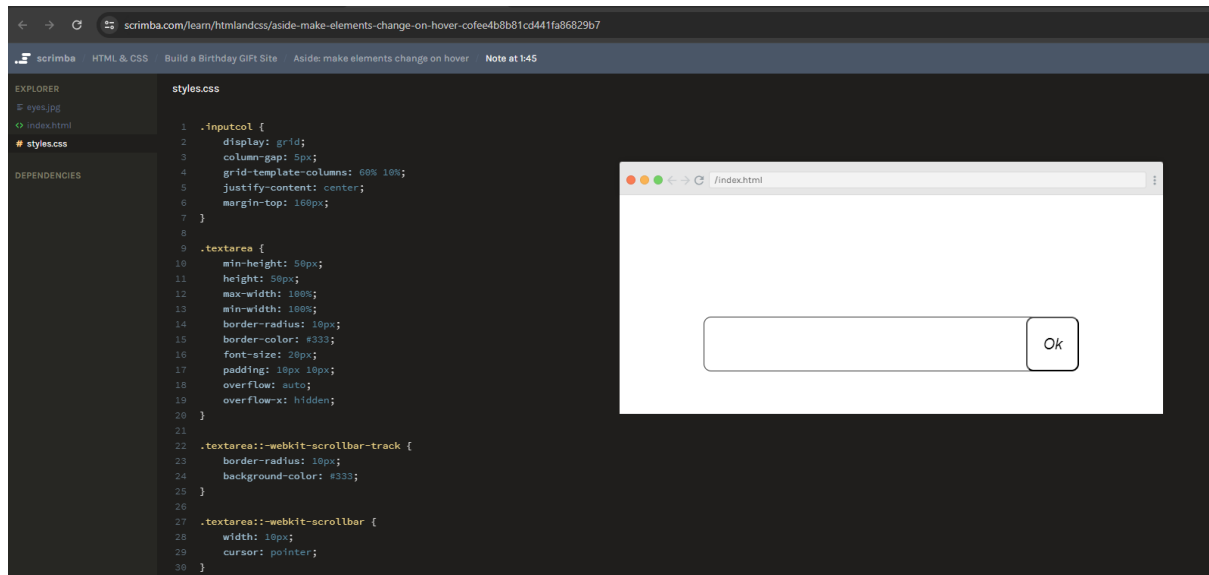
- Del apartado **“Build a Digital Business Card”**, se implementa lo explicado en el video **“Add a border and padding”** para importar una tipografía de **“Google Fonts”** y los bordes de la sección de contacto de nuestro proyecto final (Doer list):



- Del apartado **“Build a Digital Business Card”**, en el último video de **“Deploy your Digital Business Card”** utilizamos la plataforma de visualización de la web que nos recomiendan llamada **“Netlify”**, y obtenemos que nuestro sitio web se puede acceder via este link: <https://doer-list.netlify.app/>



- El apartado del tutorial “**Build a birthday GIfT Site**” se ha aplicado lo explicado en el video “**Aside: make elements change on hover**” que básicamente nos explica como hacer que el contorno de un botón cambie de color al pasar el ratón encima de él:



## 1.2 Javascript

El tutorial de Javascript se divide en 8 apartados y se han realizado toda la parte gratuita del curso:

### What's inside

This course contains 207 interactive scrims spread across 8 modules.



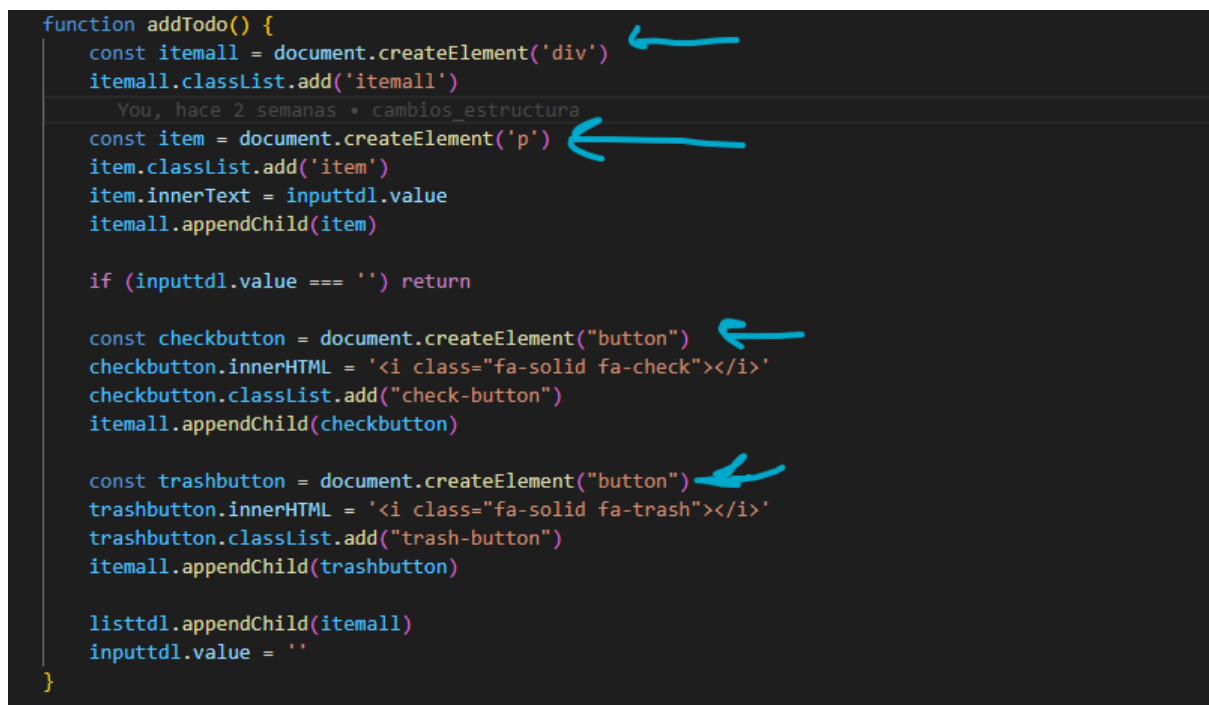
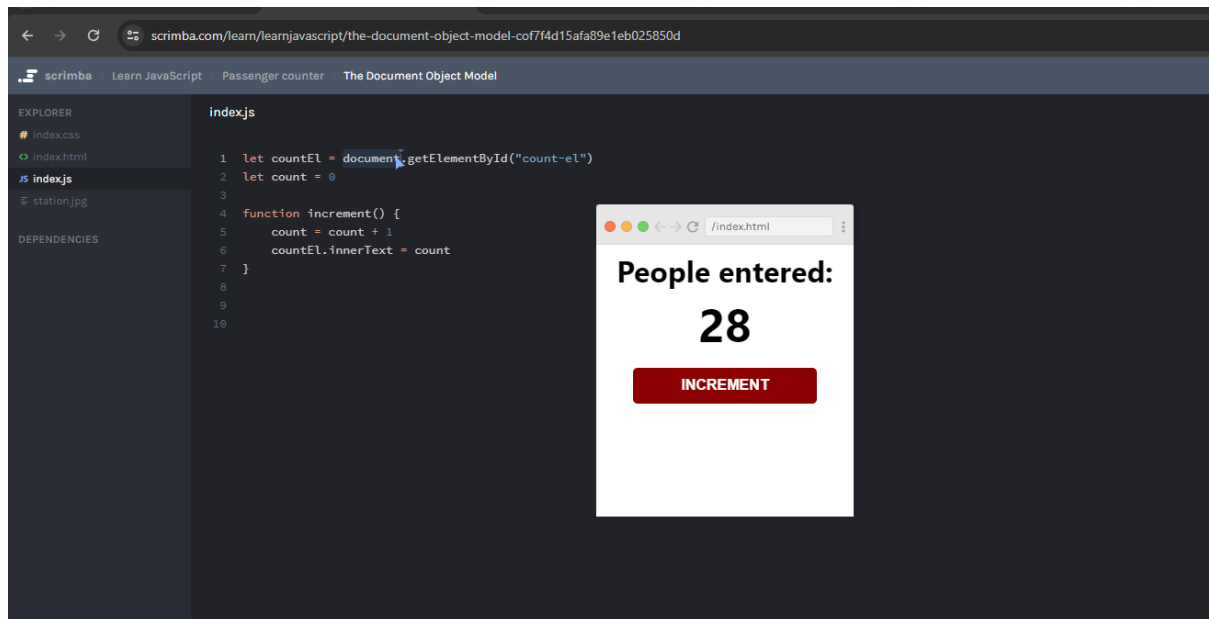
View/Hide All

> Build a passenger counter app 100%	31 lessons 1 hour 23 min
> Practice time - part 1 83%	9 lessons 25 min
> Setting up a local dev environment 100%	4 lessons 12 min
> Build a Blackjack game 100%	54 lessons 2 hours 40 min
> Practice time - part 2 90%	10 lessons 27 min
> Build a Chrome Extension 100%	54 lessons 2 hours 44 min
> Practice time - part 3 93%	9 lessons 26 min
> Build a Mobile App 97%	36 lessons 2 hours 3 min



Para desarrollar el código de la lista de tareas, el curso de JavaScript ha sido de ayuda en varios aspectos:

- Se necesitaba la comprensión sobre cómo seleccionar elementos del DOM (`document.querySelector`) para manipular su contenido y estructura. Del apartado de “**Passanger Counter**”, el video de “**The Document Object Model**” ha sido de ayuda para obtener referencias a los elementos de la lista de tareas y añadir nuevos elementos dinámicamente, que es lo que hace la función (`addTodo()`):



- El entendimiento de cómo funcionan los `addEventListener` para hacer la interacción del usuario, como el clic en el botón de añadir tarea (`buttonAdd.addEventListener('click', clickButton)`) y los clics en los botones de verificación y eliminación (`listItem.addEventListener('click', okdel)`). Han sido de ayuda los vídeos “The onclick event listener” y “Aside: The AND operator (&&)” del apartado “Build a BlackJack” del tutorial:

```
buttonAdd.addEventListener('click', clickButton)
listItem.addEventListener('click', okdel)
```

- Con los vídeos “Create the save button” y “Improve the message with string concatenation” he podido utilizar los métodos `createElement`, `appendChild`, y la manipulación del contenido de los elementos creados (`innerText`, `innerHTML`), que se añaden a los elementos de la lista de tareas (`addTodo()`) con sus botones asociados.

The screenshot shows a web browser with the URL `scrimba.com/learn/learnjavascript/create-the-save-button-co7534c79a0a694c6cae3f5da`. The page title is "Create the save button". The code editor shows the following JavaScript code in `index.js`:

```
1 let countEl = document.getElementById("count-el")
2 let count = 0
3
4 function increment() {
5   count = count + 1
6   countEl.innerText = count
7 }
8
9 // 1. Create a function, save(), which logs out the count when it's called
10
11 function save() {
12   console.log(count)
13 }
14
15
16
```

The screenshot shows a web browser with the URL `scrimba.com/learn/learnjavascript/improve-the-message-with-string-concatenation-co6714c6cae5360ed450d960`. The page title is "Improve the message with string concatenation". The code editor shows the following JavaScript code in `index.js`:

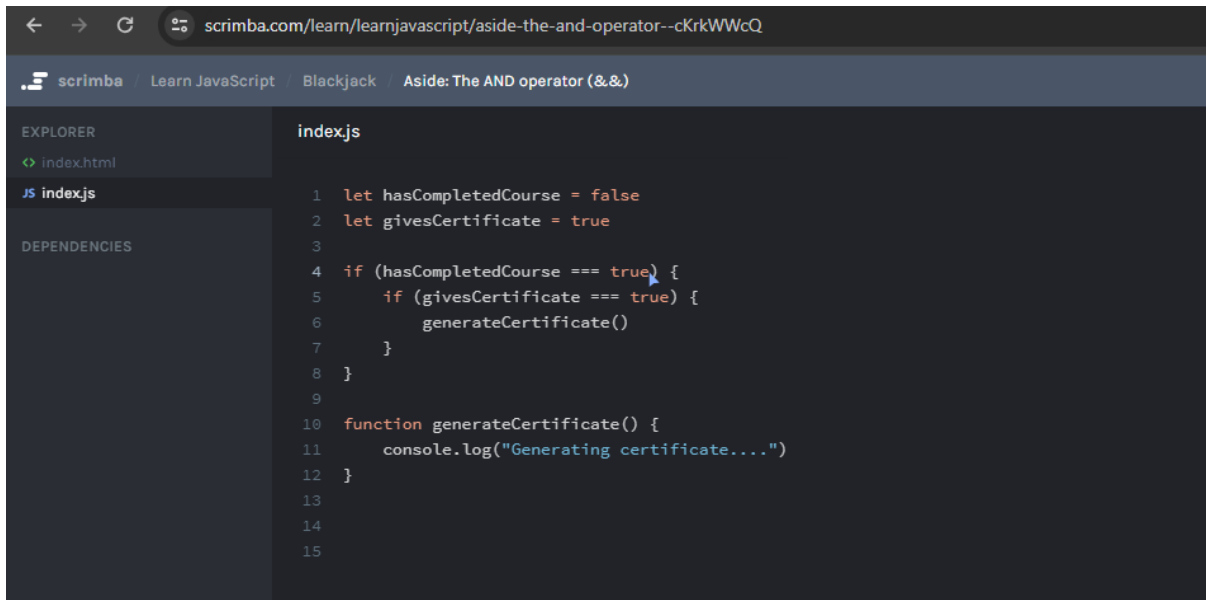
```
1 let welcomeEl = document.getElementById("welcome-el")
2
3 let name = "Per Harald Borgen"
4 let greeting = "Welcome back "
5
6 welcomeEl.innerText = greeting + name
7
8 // Add an emoji to the end! 🍌
9 // WRITE YOUR CODE BELOW HERE
10 // HINT: count = count + 1
11
12 welcomeEl.innerText += " 🍌"
13
14
```

```

JS script.js > addTodo > [x] itemall
You, hace 14 minutos | 1 author (You)
1  const inputtdl = document.querySelector('.textarea')
2  const buttontdl = document.querySelector('.buttoninput')
3  const listtdl = document.querySelector('.todolist')
4
5  function clickButton(e) {
6      e.preventDefault()
7      addTodo()
8  }
9
10
11  function addTodo() {
12      const itemall = document.createElement('div')
13      itemall.classList.add('itemall')
14
15      const item = document.createElement('p')
16      item.classList.add('item')
17      item.innerText = inputtdl.value
18      itemall.appendChild(item)
19
20      if (inputtdl.value === '') return
21
22      const checkbutton = document.createElement("button")
23      checkbutton.innerHTML = '<i class="fa-solid fa-check"></i>'
24      checkbutton.classList.add("check-button")
25      itemall.appendChild(checkbutton)
26
27      const trashbutton = document.createElement("button")
28      trashbutton.innerHTML = '<i class="fa-solid fa-trash"></i>'
29      trashbutton.classList.add("trash-button")
30      itemall.appendChild(trashbutton)
31
32      listtdl.appendChild(itemall)
33      inputtdl.value = ''
34  }
35

```

- Con “**Reassigning and incrementing**” y “**The AND operator (&&)**” he podido aplicar la condicional `if` que se utiliza para verificar si se ha ingresado un texto antes de añadirlo como una nueva tarea (“if (inputtdl.value === '') return”). También, se emplean para determinar si se ha hecho clic en el botón de verificación o de eliminación (“okdel()”):



```

1 let hasCompletedCourse = false
2 let givesCertificate = true
3
4 if (hasCompletedCourse === true) {
5   if (givesCertificate === true) {
6     generateCertificate()
7   }
8 }
9
10 function generateCertificate() {
11   console.log("Generating certificate....")
12 }
13
14
15

```



```

function addTodo() {
  const itemall = document.createElement('div')
  itemall.classList.add('itemall')

  const item = document.createElement('p')
  item.classList.add('item')
  item.innerText = inputtdl.value
  itemall.appendChild(item)

  if (inputtdl.value === '') return

  const checkbutton = document.createElement("button")
  checkbutton.innerHTML = '<i class="fa-solid fa-check"></i>'
  checkbutton.classList.add("check-button")
  itemall.appendChild(checkbutton)

  const trashbutton = document.createElement("button")
  trashbutton.innerHTML = '<i class="fa-solid fa-trash"></i>'
  trashbutton.classList.add("trash-button")
  itemall.appendChild(trashbutton)

  listtdl.appendChild(itemall)
  inputtdl.value = ''
}

function okdel(e) {
  const item = e.target

  // hecho You, hace 1 segundo • Uncommitted changes
  if (item.classList[0] === 'check-button') {
    const todolist = item.parentElement
    todolist.classList.toggle('checklist')
  }

  // eliminar
  if (item.classList[0] === 'trash-button') {
    const todolist = item.parentElement

```

## 2. Conclusión

Los cursos de HTML | CSS y Javascript, han sido útiles a la hora de saber como hacer y empezar con el proyecto final de la propia asignatura.

Opino que la metodología de “Scrimba.com”, es bastante disruptiva. Me ha gustado como a la vez que va transcurriendo la explicación puedes modificar el código a tu gusto, haciendo que el aprendizaje sea más ameno.