🇪🇸 Proyecto UseState ReactJS

Introducción

Como primer proyecto vamos a crear un `Curriculum` con React haciendo uso de varias de las nociones aprendidas en las sesiones, como pueden ser los diferentes `componentes`, `useState` y traer información `mapeada` de distintos archivos.

Crear una nueva aplicación de React

Creamos una aplicación con `Vite` llamada myCV. De esta manera ya tendremos nuestro proyecto arrancado y listo para configurar.

En la carpeta src vamos a crear la carpeta correspondiente de `components` y una adicional en el mismo nivel llamada `CV`.

Dentro de components incluiremos los siguientes:

- `Hero`: nombre, información básica y contacto.
- `About`: más datos sobre nosotros.
- `Education` y `Experience`: educación y experiencia laboral actual y previa.
- `More`: idiomas, skills y, por ejemplo, voluntariados.

Estos componentes contendrán de forma individual una `hoja de estilos`, en este caso `.css`, para poder manipularlos uno a uno.

Dentro de la carpeta CV crearemos un archivo `Cv.js` donde almacenar un JSON con toda la información que queremos incluir en nuestro currículum. Esta información la iremos rescatando componente a componente según nuestras necesidades.

Hay que tener en cuenta que podéis conformar todo esto de la manera que queráis y componentizar más o menos según las necesidades, pero en este caso vamos a seguir este ejemplo para adquirir las nociones correctas.

Datos del CV

Vamos a cumplimentar la información de nuestro curriculum en el archivo `Cv.js`, en este ejemplo vamos a tomar como referencia a Tony Stark de la siguiente forma:

```
export const CV = {
  hero: {
    name: "Anthony",
    surname: "Edward Stark",
    city: "Avengers Tower / New York",
    email: "tony@starkindustries.com",
    birthDate: "29/05/1970",
    phone: "(+12) 767333841 NO PONGAIS EL VUESTRO POR FAVOR",
    image: "https://i.imgur.com/ZQAkED3.png",
    gitHub: "https://github.com/tonystark",
    aboutMe: [
      {
        info: "🤖 My armor, it was never a distraction or a hobby, it was a cocoon. I am Iron Man.",
      },
      {
        info: "🔩 CEO of Stark Industries.",
      },
      {
        info: "🕶️ Genius, billionaire, playboy, philanthropist.",
```

```
      },
      {
        info: "🦾 I do have a responsibility to keep my inventions from evil hands – but I have a greater responsibility to oppose that evi
      },
    ],
  },
  education: [
    {
      name: "Master in physics",
      date: "1985",
      where: "MIT",
    },
    {
      name: "Aeronautical Engineering",
      date: "1995",
      where: "ATI Vaughn College",
    },
    {
      name: "Weapons Engineer",
      date: "1998",
      where: "Firearm Collegue, Queens",
    },
  ],
  experience: [
    {
      name: "Consultant",
      date: "01/01/2013 – Nowadays",
      where: "S.H.I.E.L.D",
      description:
        "It builds the helicarriers used by S.H.I.E.L.D. It produces the Quinjets used by the Avengers.",
    },
    {
      name: "CEO",
      date: "01/01/2000 – 28/02/2012",
      where: "Stark Industries",
      description:
        "Manage the company, which is a multi-billion dollar multinational corporation that develops and manufactures advanced weapon and d
    },
  ],
  languages: {
    language: "English",
    wrlevel: "Native",
    splevel: "Native",
  },
  habilities: [
    "Robotics",
    "Robot Programming",
    "Physics",
    "Weaponery",
    "Engineer",
    "Money",
    "Dating",
    "Saving the world",
  ],
  volunteer: [
    {
      name: "September Foundation",
      where: "MIT",
      description:
        "The September Foundation is a program by Tony Stark to fund schools and young prodigies in their education. The foundation was nam
    },
    {
      name: "Damage Control",
      where: "U.S.A.",
      description:
        "The United States Department of Damage Control, occasionally known as the DODC, is a department of the United States of America. I
    },
  ],
};
```

Hemos almacenado todos nuestros datos en `arrays`, `objetos` y `arrays de objetos`. Es recomendable dividir nuestros datos en secciones para poder llamarlos más fácilmente y de forma ordenada en nuestra aplicación.

La ventaja de tener toda la información de nuestro CV en un mismo archivo es la facilidad de modificar, añadir o eliminar datos; ya que cualquier cambio que realicemos se reflejará en toda la aplicación al instante.

Ahora veamos como acceder a estos datos:

```
import { useState } from "react";
import "./App.css";
import { CV } from "./CV/CV";

const { hero, education, experience, languages, habilities, volunteer } = CV;
```

En nuestro componente `App` hemos importado `useState` para su posterior uso, la hoja de estilos de nuestro componente principal `App.css`, los `componentes` y la constante CV dentro de nuestro archivo `CV.js` para poder usarla a nivel global.

Justo debajo de las importaciones realizaremos un `destructuring` de nuestra constante `CV` para poder acceder a `hero`, `education`, `experience`, `languages`, `habilities` y `volunteer` directamente, consiguiendo un código más limpio y legible.

Componentes

Gracias al `destructuring` de nuestra constante CV podemos pasar por `prop` unos elementos u otros. La ventaja de haber componentizado nuestra aplicación y estructurado nuestra constante por `secciones` es la claridad de los `props` que tenemos que pasar a un componente:

```
import { useState } from "react";
import "./App.css";
import Hero from "./components/Hero.jsx"...
[...]
import { CV } from "./CV/CV";

const { hero, education, experience, languages, habilities, volunteer } = CV;

function App() {
  return (
    <div className="App">
        <Hero hero={hero} />
        <About hero={hero} />
        <Education education={education} />
        <Experience experience={experience} />
        <More
        languages={languages}
        habilities={habilities}
        volunteer={volunteer}
        />
    </div>
  );
}

export default App;
```

Tal y como aparece en el código, le estamos pasando por `props` la información que necesitamos a cada componente, por ejemplo: a nuestros componentes `<Hero/>` y `<About/>` le estamos pasando la información de `hero` a ambos ya que contiene el `nombre`, los `datos personales` y las frases almacenadas en `aboutMe.`

Al componente `<More/>` le estamos pasando por prop `languages`, `habilities` y `volunteer` porque todo estará incluido en el mismo componente. Todo es cuestión de organizar nuestra información y repartirla de la manera más conveniente y clara.

Vamos a echarle un vistazo a nuestro componente `<Hero/>`, ya que va a ser el primero que veamos nada más arrancar la aplicación:

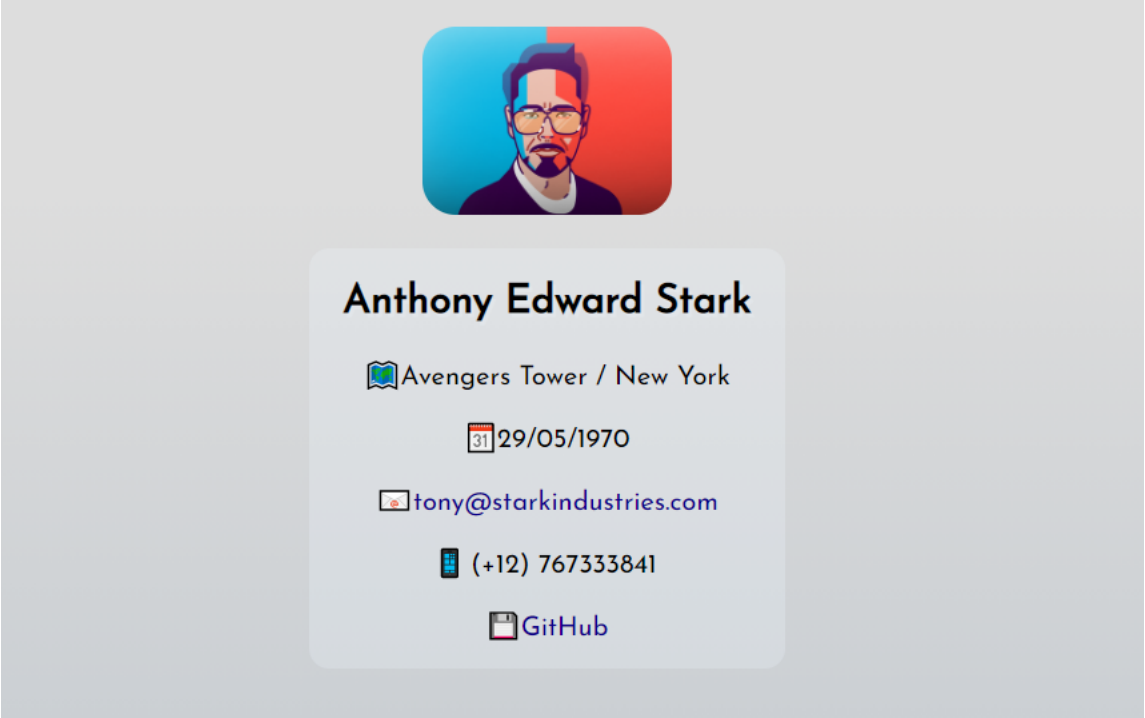
```
import React from "react";
import "./Hero.css";

const Hero = ({ hero }) => {
  return (
    <div className="hero">
      <img src={hero.image} alt="" />
      <div className="card">
      <h2>
        {hero.name} {hero.adress}
      </h2>
        <p>🗺{hero.city} </p>
        <p>🛅{hero.birthDate}</p>
        <p>
          📧
          <a href={"mailto:" + hero.email}>
          tony@starkindustries.com
          </a>
        </p>
        <p>📱 {hero.phone}</p>
        <p>💾<a href={hero.gitHub}>
            GitHub
          </a></p>
      </div>
    </div>
  );
};

export default Hero;
```

Este componente, como hemos visto, está recibiendo por prop la información de `hero`, por lo que la etiqueta `img` nos va a pintar el valor de `hero.image`, el primer `h2` nos pintará `hero.name` y `hero.adress` para tener el nombre completo y usando la misma lógica completaremos nuestro componente con la información que deseemos mostrar por pantalla.

En nuestra hoja de estilos `Hero.css` le hemos dado las siguientes propiedades para hacerlo más llamativo a la vista. Lo podéis tomar de referencia o crear el estilo que queráis:

```
.hero {
  display: flex;
  flex-direction: column;
  align-items: center;
}
.card {
  padding-left: 20px;
  padding-right: 20px;
  text-align: center;
  border-radius: 12px;
  background-color: rgba(240, 248, 255, 0.241);
}
img {
  width: 150px;
  margin: 20px;
  border-radius: 20px;
}
h2 {
  color: black;
  text-shadow: 2px 2px rgba(240, 248, 255, 0.241);
}
a {
  color: #0b0089;
  text-decoration: none;
}
```

Y así quedaría nuestro componente `<Hero/>` renderizado:



Usando la misma lógica, vamos a cumplimentar el resto de componentes recibiendo la información del prop deseado. Cuando tengamos `arrays` tendremos que mapear la información para pintar los datos correctamente.

Un ejemplo de este caso es el componente `<Education />`:

```
import React from "react";
import "./Education.css";

const Education = ({ education }) => {
  return (
    <div>
      <div className="education card">
        {education.map((item) => {
          return (
            <div key={JSON.stringify(item)}>
              <p className="name">🎓 {item.name}</p>
              <p>{item.where}</p>
              <p>{item.date}</p>
            </div>
          );
        })}
      </div>
    </div>
  );
};

export default Education;
```

Al mapear `education` vamos a generar un `item`, y por cada `item` nos va a crear 3 `<p>` con la información deseada. Recordad indicarle las `keys` a estos elementos tras realizar el `.map` para que React lea correctamente los datos.

Una vez realizado esto ya podemos ver por completo nuestra aplicación con todos los datos que hemos incluido en nuestro archivo `CV.js`.

Ahora vamos a practicar de manera muy sencilla `useState` para crear dos botones que nos muestre o bien el componente `<Education />` o el componente `<Experience/>` y así dinamizar un poco más nuestra aplicación.

useState

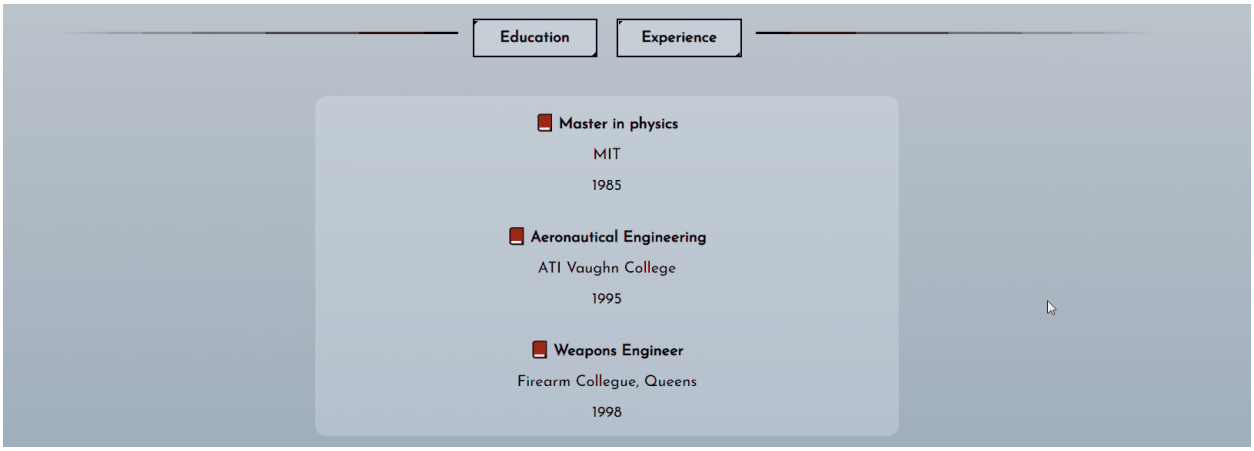
Como bien vimos, ya tenemos importado `useState` de React en nuestro componente `App`, por lo que solo nos quedaría crear una constante a la que cambiar el estado y dos botones que nos muestren un componente u otro según los clickemos:

```
import { useState } from "react";
import "./App.css";
//Importación de los componentes------
import { CV } from "./CV/CV";

const { hero, education, experience, languages, habilities, volunteer } = CV;

function App() {
  const [showEducation, setShowEducation] = useState(true);
  return (
    ...
            <button
              className="custom-btn btn-4"
              onClick={() => setShowEducation(true)}
            >
              Education
            </button>
            <button
              className="custom-btn btn-4"
              onClick={() => setShowEducation(false)}
            >
              Experience
            </button>
    ...
      <div>
        {showEducation ? (
          <Education education={education} />
        ) : (
          <Experience experience={experience} />
        )}
      </div>
   ...
  );
}

export default App;
```

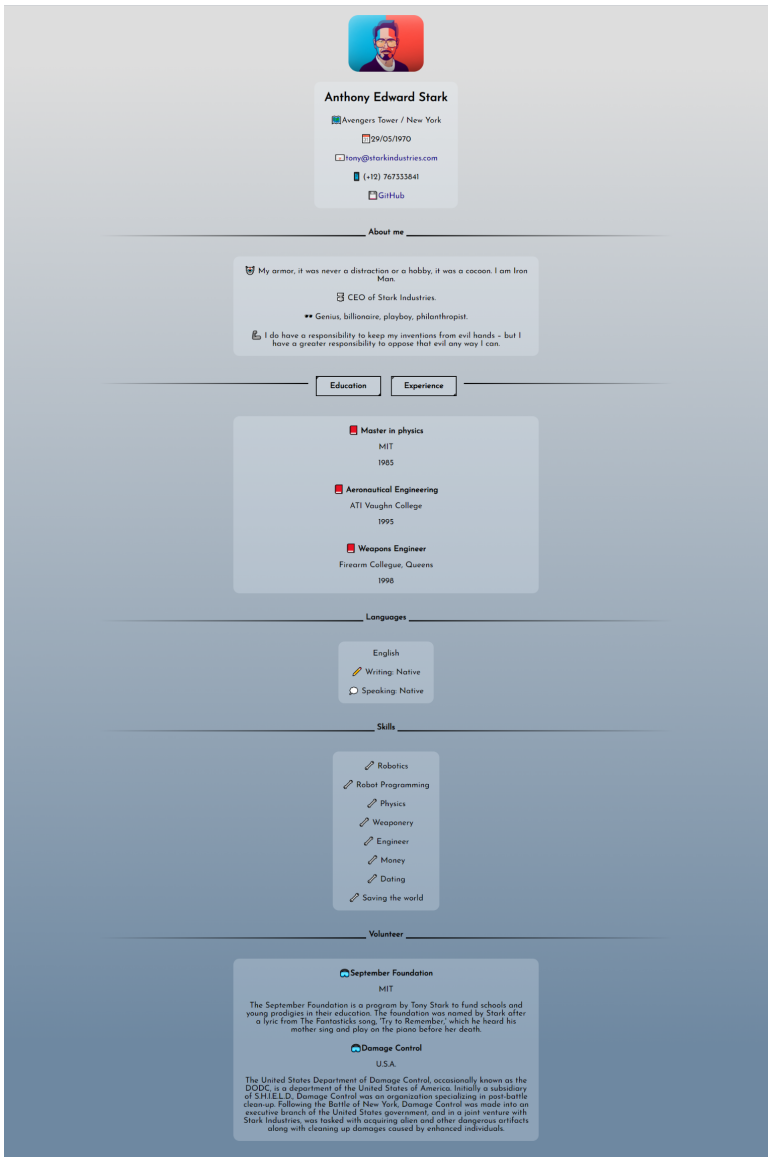
Por defecto vamos a setear nuestra variable `showEducation` a `true` para que nada más arrancar la aplicación nos muestre por defecto el componente `<Education/>`.

Siguiendo esta lógica vamos a crear un botón de `Education` que nos setée `showEducation` a `true` si no lo estuviera y un botón de `Experience` que nos lo vuelva `false` al clicar.

Gracias a estos dos botones crearemos un div posteriormente con un ternario que nos muestre Education si `showEducation = true` y nos muestre Experience si `showEducation = false`.



Una vez montemos todos nuestros componentes en `App` y arranquemos el proyecto tendremos un CV con enlaces de interés a nuestros perfiles y proyectos, secciones dinámicas y un aspecto más visual que los currículums habituales.


Anthony Edward Stark
Avengers Tower / New York
29/05/1970
tony@starkindustries.com
(+12) 767333841
GitHub
About me
My armor, it was never a distraction or a hobby, it was a cocoon. I am Iron Man.
CEO of Stark Industries.
Genius, billionaire, playboy, philanthropist.
I do have a responsibility to keep my inventions from evil hands - but I have a greater responsibility to oppose that evil any way I can.
Education
Experience
Master in physics
MIT
1985
Aeronautical Engineering
ATI Vaughn College
1995
Weapons Engineer
Firearm Collegue, Queens
1998
Languages
English
Writing: Native
Speaking: Native
Skills
Robotics
Robot Programming
Physics
Weaponery
Engineer
Money
Dating
Saving the world
Volunteer
September Foundation
MIT
The September Foundation is a program by Tony Stark to fund schools and young prodigies in their education. The foundation was named by Stark after a lyric from The Fantasticks song, 'Try to Remember,' which he heard his mother sing and play on the piano before her death.
Damage Control
U.S.A.
The United States Department of Damage Control, occasionally known as the DODC, is a department of the United States of America. Initially a subsidiary of S.H.I.E.L.D., Damage Control was an organization specalising in post-battle clean-up. Following the Battle of New York, Damage Control was made into an executive branch of the United States government, and in a joint venture with Stark Industries, was tasked with acquiring alien and other dangerous artifacts along with cleaning up damages caused by enhanced individuals.