# Consultando una API

- En CryptoCompare hay una API con endpoints que se pueden utilizar
- Top List by Market Cap Full Data
- Puedo cambiar de 10 a 20 los resultados

> Endpoint: https://min-api.cryptocompare.com/data/top/mktcapfull?limit=10&tsym=USD

- Importo useEffect con un arreglo vacío en el formulario, es un buen lugar para llamar a una APi
- De esta forma cuando esté cargado el componente, como no tiene ninguna dependencia va a ejecutarse una sola vez cuando esté listo, va a consultar la API y traer los resultaods
- Cómo no sé si tengo respuesta uso el await con el fetch a la url
- Puedo comprobarlo con un console.log(respuesta)
- De nuevo, el resultado lleva un await y retorna un json
- Lo que interesa dela respuesta es el .Data


FOTO9

- Ahora hay que crear un arreglo que tenga la misma forma que las monedas que escribí (dólar, euro, peso mexicano, libra esterlina)

---

# Formateando un Array para pasarlo como opciones

- CoinInfo es la parte del json de respuesta que me interesa
- Tiene FullName: "Bitcoin" y el formato de tres dígitos Name: "BTC"
- .map me crea un nuevo arreglo (forEach solo lista) que se va a ir llenando en arrayCriptos
- Si le pongo un console.log aparece en consola cada una de las criptos

```
const arrayCriptos = resultado.Data.map(cripto=>{
            console.log(cripto)
            })
```

De nuevo lo que me interesa esta en CoinInfo, puedo añadirlo

```
    >console.log(cripto.CoinInfo.Name)
    >console.log(cripto.CoinInfo.FullName)
```

-El Hook que construí itera sobre id y sobre Nombre, así que puedo asignarlos en un nuevo objeto para posteriormente hacer la relación para la conversión

- Puedo usar esos 3 dígitos como key ya que no se repiten
- Pero si yo imprimo el arrayCriptos va a devolver un array vacío
- Por eso coloco el return objeto, que es lo que me interesa
- Ahora, lo que estoy construyendo en esta parte de mi código, se va a ir retornando y llenando el objeto arrayCriptos

```
useEffect(()=>{

  const consultarAPI= async()=>{
    const url= ' https://min-api.cryptocompare.com/data/top/mktcapfull?limit=10&tsym=USD'
    const respuesta= await fetch(url)
    const resultado = await respuesta.json()


    const arrayCriptos = resultado.Data.map(cripto=>{
        const objeto = {
          id: cripto.CoinInfo.Name,
          nombre: cripto.CoinInfo.FullName
        }
        return objeto
    })

  }
  consultarAPI()
}, [])
```

Creo un state con criptos y setCriptos, lo inicio como un arreglo vacío

- Entonces, la función se llama setCriptos, que va a llenar ese arreglo
- Le meto el arrayCriptos

```
 useEffect(()=>{

    const consultarAPI= async()=>{
      const url= ' https://min-api.cryptocompare.com/data/top/mktcapfull?
limit=10&tsym=USD'
      const respuesta= await fetch(url)
      const resultado = await respuesta.json()


      const arrayCriptos = resultado.Data.map(cripto=>{
          const objeto = {
            id: cripto.CoinInfo.Name,
            nombre: cripto.CoinInfo.FullName
          }
          return objeto
      })
     setCriptos(arrayCriptos)
    }
    consultarAPI()
  }, [])
```

Por la forma en la que escribí el hook va a ser muy sencillo implementar un segundo state

- Duplico la linea y en el label pongo Elije tu criptomoneda

```
 const [state, SelectMonedas] = useSelectMonedas('Elige tu moneda', monedas)
  const [cripto, SelectCriptoMoneda] = useSelectMonedas('Elige tu Criptomoneda',
criptos)
```

- Ahora ya tengo los dos states
- Cuando le de a cotizar, puedo tener un handle submit, llevarlo hacia otro componente y consular la conversión