Validando el Presupuesto

- Para manejar el presupuesto coloco en el onSubmit una función.
- La llamaré handlePresupuesto, suele ser el standard.
- Cuando ejecute el botón de submit se va a ejecutar esta función
- Uso el e.preventDefault para evitar el refresh
- Para comprobar que está todo correcto uso console.log
- Uso Number para pasar el output de strings a numeros; tengo presupuesto porque lo pasé por props

```
const NuevoPresupuesto = ({presupuesto, setPresupuesto}) => {

  const handlePresupuesto=(e)=>{
    e.preventDefault()
    console.log(Number(presupuesto))
  }

  return (
    <div className="contenedor-presupuesto contenedor sombra">

      <form action="" className="formulario"
      onSubmit={handlePresupuesto}>

        <div className="campo">
          <label>Definir Presupuesto</label>

         <input
          className="nuevo-presupuesto"
          type="text"
          placeholder="Añade tu presupuesto"
          value={presupuesto}
          onChange={(e)=>setPresupuesto(e.target.value)} />
        </div>

        <input type="submit" value="Añadir" />
      </form>
    </div>
  )
}

export default NuevoPresupuesto
```

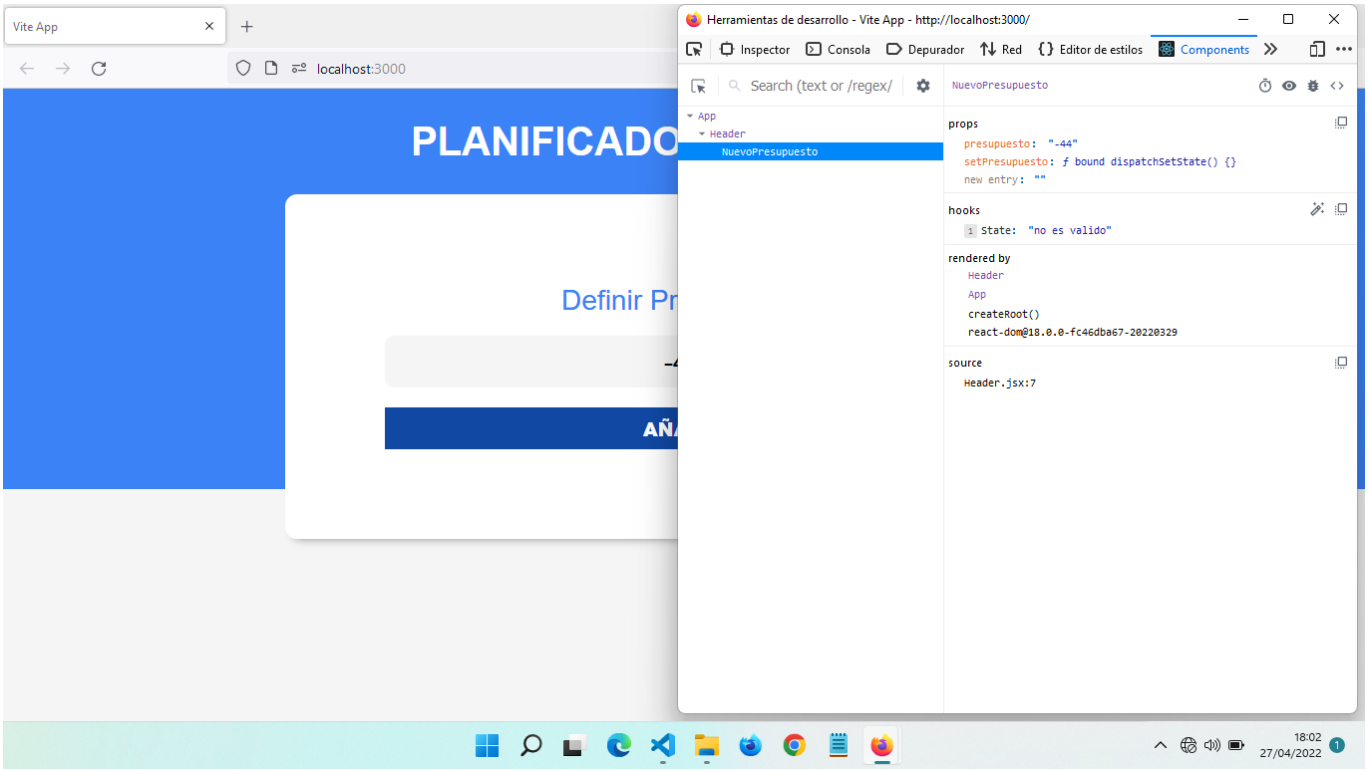
- Para hacer la validación, le digo si no es un numero o es menor que cero, console.log no válido

```
const handlePresupuesto=(e)=>{
  e.preventDefault()
  if(!Number(presupuesto) || Number(presupuesto) <0){
      console.log('no es valido')
  }else{
    console.log('es valido')
  }
}
```

- Los mensajes en consola es solo para probarlo. Voy a usar un state, no lo requiero en otros componentes.

```
const handlePresupuesto=(e)=>{
  e.preventDefault()
  if(!Number(presupuesto) || Number(presupuesto) <0){
      setMensaje('no es valido')
  }else{
      setMensaje('es valido')
  }
}
```

- Ahora si voy a components en devTools debería ver el state, por ejemplo poniendo un num negativo.



- Ahora se puede mostrar de forma booleana.
- Creo un nuevo componente llamado Mensaje
- Le paso children por las props, que es el contenido del componente

- También una variable tipo, que será la variable clase que inyecto en el template string
- La clase alerta es una predefinida

```
import React from 'react'

const Mensaje = ({children, tipo}) => {
  return (
    <div className= {`alerta ${tipo}`}>
        {children}
    </div>
  )
}

export default Mensaje
```

- Lo importo en NuevoPresupuesto, y digo: si hay mensaje ENTONCES imprime el componente Mensaje con el children, que es el contenido mensaje
- Por props le paso el tipo: error
- El mensaje es el que seteo con setMensaje

```
import { useState } from "react"
import Mensaje from "./Mensaje"


const NuevoPresupuesto = ({presupuesto, setPresupuesto}) => {

  const[mensaje,setMensaje] = useState('')


  const handlePresupuesto=(e)=>{
    e.preventDefault()
    if(!Number(presupuesto) || Number(presupuesto) <0){
        setMensaje('no es valido')
    }else{
        setMensaje('es valido')
    }
  }

  return (
    <div className="contenedor-presupuesto contenedor sombra">
      <form action="" className="formulario"
      onSubmit={handlePresupuesto}>
        <div className="campo">
          <label>Definir Presupuesto</label>
          <input
          className="nuevo-presupuesto"
          type="text"
          placeholder="Añade tu presupuesto"
          value={presupuesto}
          onChange={(e)=>setPresupuesto(e.target.value)}
```

```
          />
        </div>
        <input type="submit" value="Añadir" />
        {mensaje && <Mensaje tipo="error">{mensaje}</Mensaje>}
      </form>
    </div>
  )
}

export default NuevoPresupuesto
```

- De esta manera se inyecta una clase fija con una dinámica y se pasa el contenido con el children
- Lo próximo será definir el componente Gastos para cuando el presupuesto sea válido