Mostrando el Componente de Gastos

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

Si elimino el else y pongo un console.log fuera del if, puedo ver que si le doy a Añadir, aparece en consola aunque sea un presupuesto no valido

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

log(...data: any[]): void

Por ello le coloco un return, para que no siga ejecutando nada más y rompo el ciclo

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

Cómo sale de esa condición, no se ejecuta el return y ejecuta la otra parte

- Remuevo el console.log que me servía para la comprobación
- Voy a remover el mensaje de alerta por si se mostró anteriormente con setMensaje y un string vacío

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

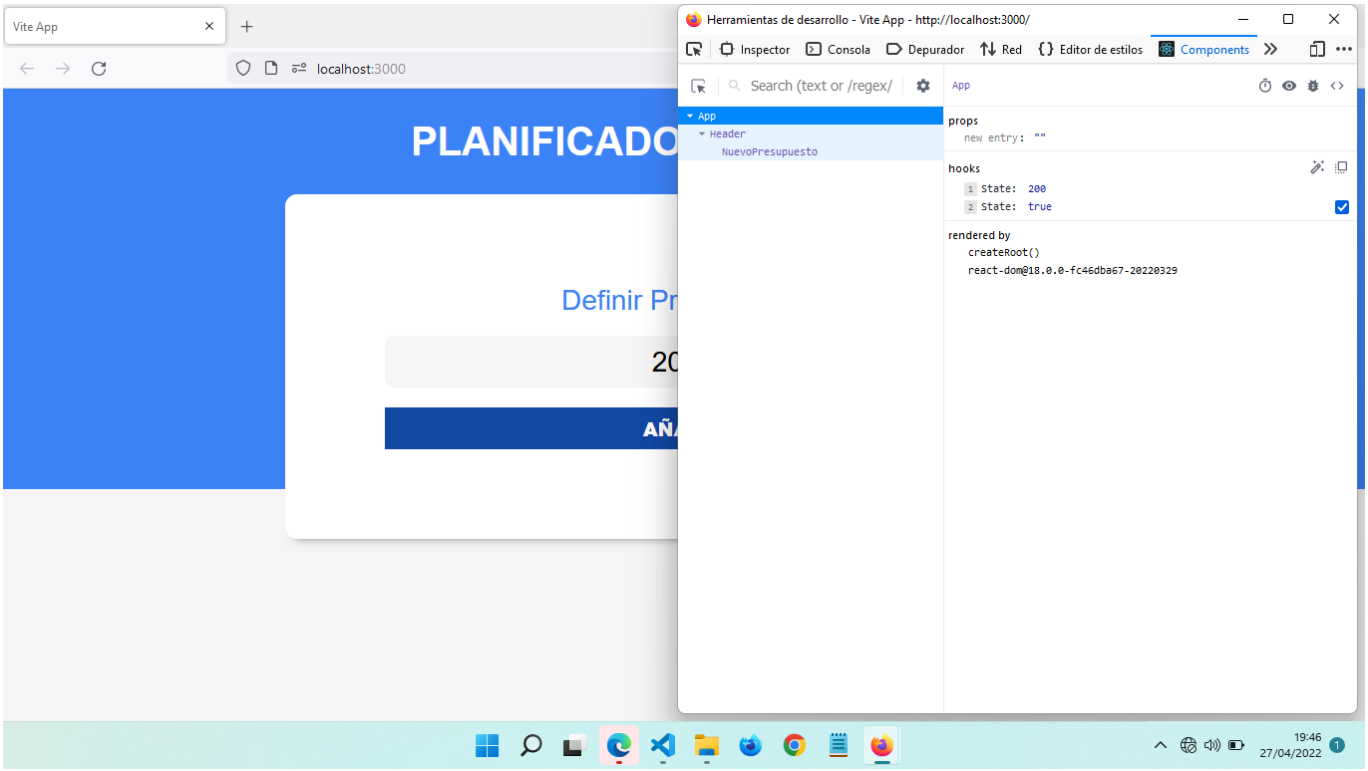
```
  }
```

- Ocurre que la cantidad que yop introduzco en el input, al pasarlo al state los pasa como string.
- Para pasarlo a numero, uso NUmber en el setPresupuesto

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


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

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


  const handlePresupuesto=(e)=>{
    e.preventDefault()
    if(!presupuesto || presupuesto <0){  //No se necesita el Number aqui
        setMensaje('no es valido')
        return
    }
    setMensaje('')
  }

  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="number"  //cambio el type!
          placeholder="Añade tu presupuesto"
          value={presupuesto}
          onChange={(e)=>setPresupuesto(Number(e.target.value))}
                                              //uso el Number!

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

export default NuevoPresupuesto
```

- Ahora solo se pueden introducir números en el input

- Creo un state en App.jsx con isValidPresupuesto y setIsValidPresupuesto
    - El state con el valor de false ya que carga en 0 el presupuesto
- Lo paso por props al Header y a NuevoPresupuesto solo el setIsValidPresupuesto
- Lo coloco fuera del if con el return y lo cambio a true.
- Esto me dará la posibilidad de abrir el otro componente
- Puedo ver en components de devtools como un state es la cantidad y el otro cambia a true cuando le doy añadir y el presupuesto es valido



- Entonces, si el presupuesto es válido muestrame el otro componente "Control Presupuesto"
- Si no vamos a definir el NuevoPresupuesto
- Esto lo puedo hacer con un ternario en el componente Header

```
import NuevoPresupuesto from "./NuevoPresupuesto"

const Header = ({
  presupuesto,
  setPresupuesto,
  isValidPresupuesto,
  setIsValidPresupuesto}) => {
  return (
    <header>
        <h1>Planificador de Gastos</h1>

        {isValidPresupuesto ?(
          <p>Control Presupuesto</p>
        ): (
          <NuevoPresupuesto
            presupuesto={presupuesto}
            setPresupuesto={setPresupuesto}
            setIsValidPresupuesto={setIsValidPresupuesto}
```

```
            />

          )}

      </header>
    )
}

export default Header
```

- Aparece algo así, quitando el otro componente con el input y muestra el párrafo