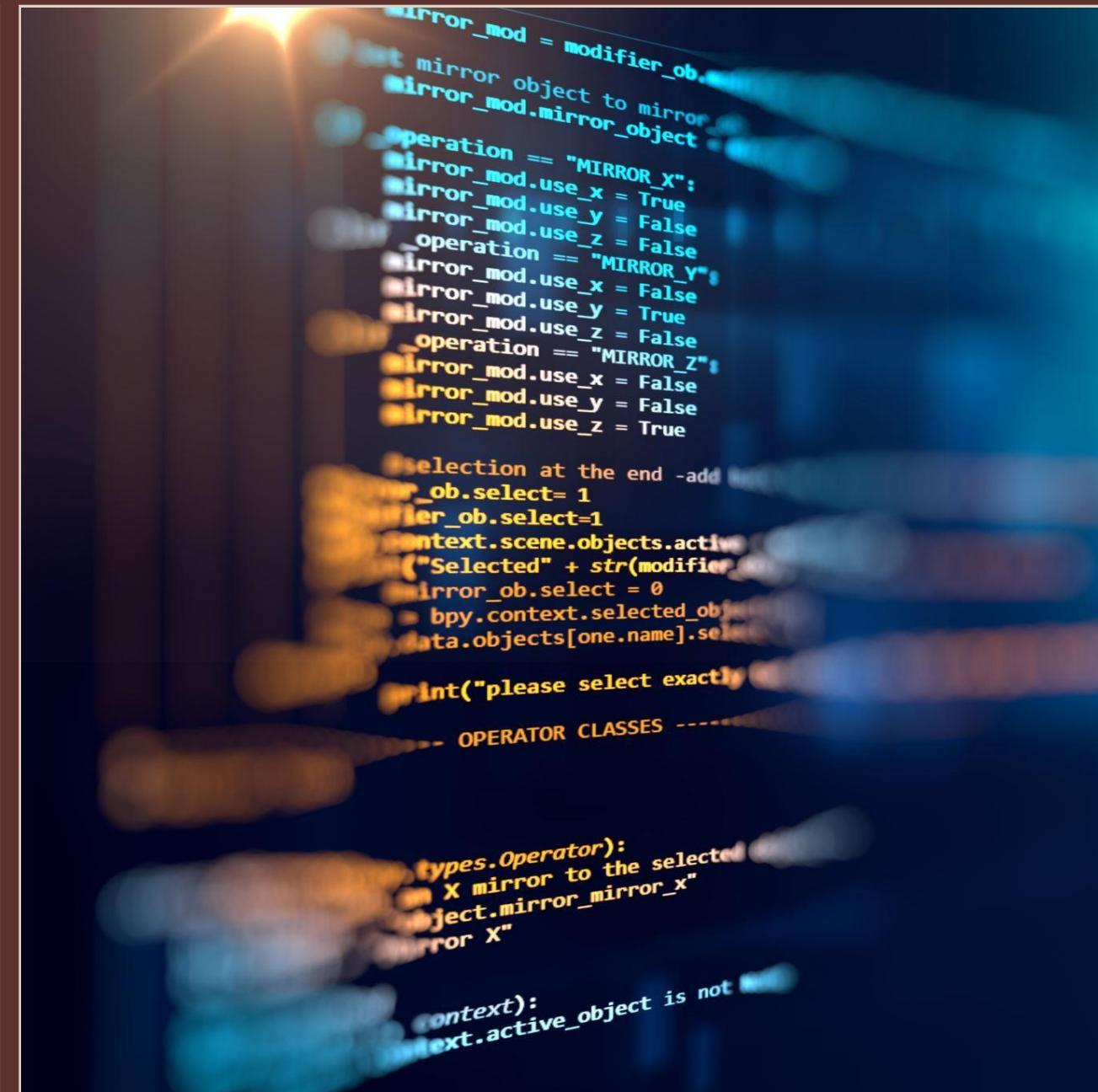


# Introducción a React

Conceptos básicos para crear  
interfaces web interactivas



The image shows a person's hand pointing towards a computer monitor. The monitor displays a dark-themed Python script. The script includes logic for mirroring objects in 3D space along X, Y, or Z axes, handling selection validation, and defining operator classes. A tooltip or callout bubble is visible near the bottom right of the screen, providing a detailed explanation of one of the script's functions.

```
mirror_mod = modifier_obj.modifiers.new("MIRROR", type='MIRROR')
# set mirror object to mirror
mirror_mod.mirror_object = bpy.context.scene.objects.active
if operation == "MIRROR_X":
    mirror_mod.use_x = True
    mirror_mod.use_y = False
    mirror_mod.use_z = False
elif operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
elif operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

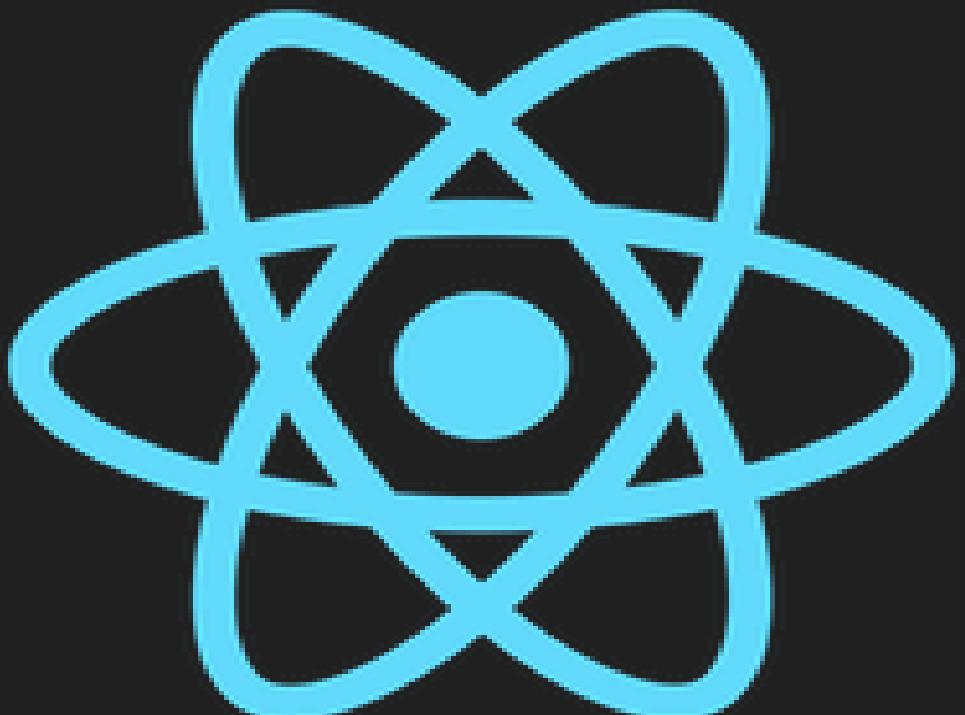
# selection at the end - add
if ob.select= 1
    ob.select=1
context.scene.objects.active = ("Selected" + str(modifier))
    mirror_ob.select = 0
    - bpy.context.selected_objects[0]
    data.objects[one.name].select = 1
    print("please select exactly one object")
else:
    print("please select exactly one object")

-- OPERATOR CLASSES ----

# types.Operator:
# X mirror to the selected
# object.mirror_mirror_x"
# or X"
# context):
# context.active_object is not
```

# Introducción a React

# Índice



## Visión general de React

React es una biblioteca popular para crear interfaces de usuario en aplicaciones web modernas.

## Configuración del entorno

Configurar un entorno de desarrollo adecuado es esencial para trabajar eficientemente con React.

## Creación y estructura de proyectos

Aprender métodos para crear proyectos y la estructura básica de una aplicación React es clave para empezar.

## Recursos y recomendaciones

Se proveen recursos útiles para continuar aprendiendo y mejorar el dominio de React.

# ¿Qué es React?

# Características y ventajas



## Componentes Reutilizables

React utiliza componentes independientes que facilitan la modularidad y el mantenimiento del código.

## JSX para Legibilidad

JSX permite escribir código similar a HTML dentro de JavaScript, mejorando la integración y legibilidad.

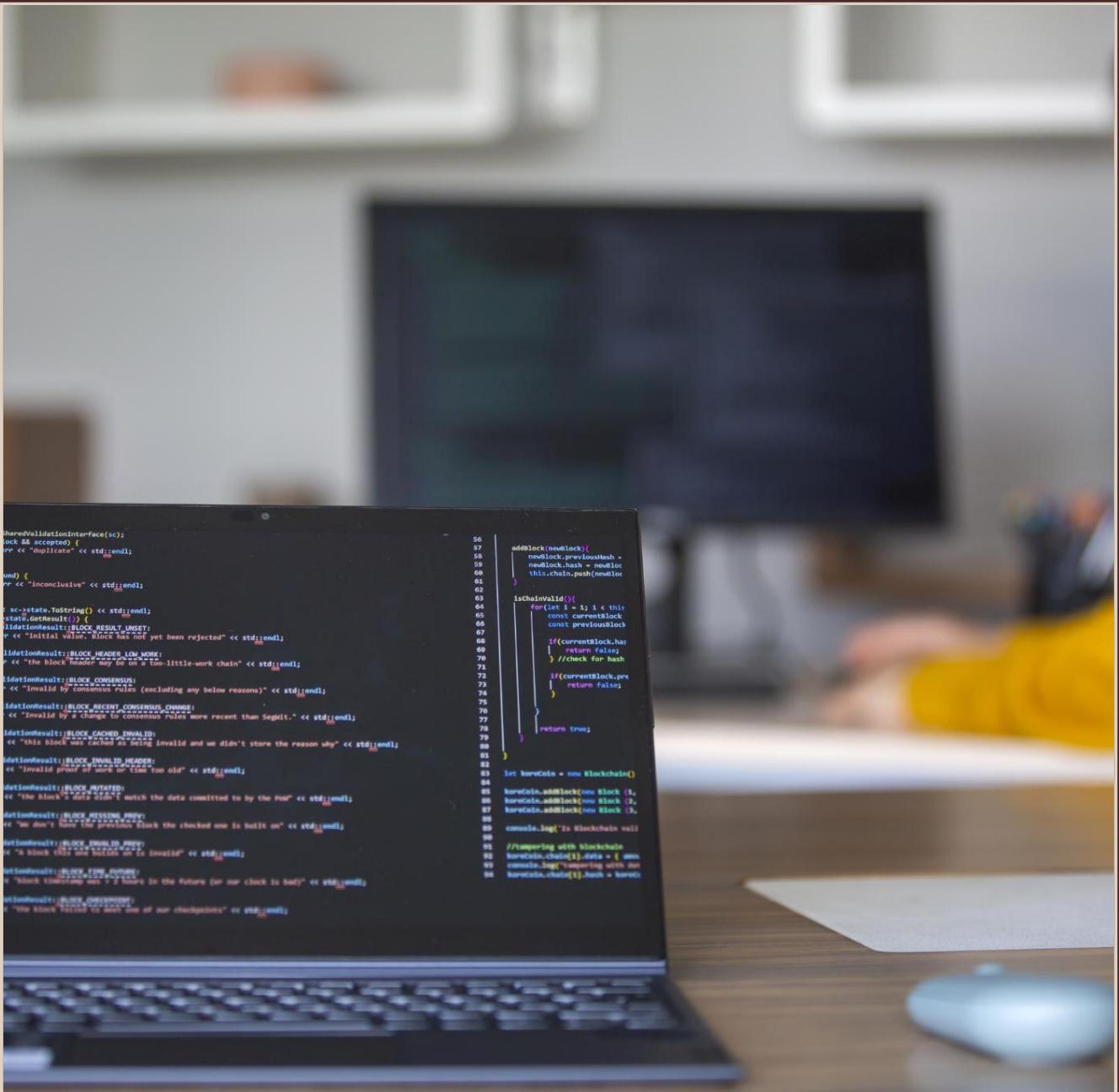
## Virtual DOM Optimizado

Virtual DOM actualiza solo los elementos necesarios, mejorando el rendimiento de la interfaz.

## Flujo de Datos Unidireccional

Los datos fluyen de componentes padres a hijos, facilitando el control y depuración del estado.

# Configuración del entorno



# Instalación y herramientas recomendadas

## Instalación de Node.js y npm

Node.js incluye npm, esencial para React. Se recomienda la versión LTS y añadir Node al PATH para facilitar su uso.

## Verificación de instalación

Comandos node -v y npm -v permiten comprobar la correcta instalación y versiones de Node.js y npm.

## Entorno de desarrollo Visual Studio Code

Visual Studio Code es popular por su flexibilidad, con extensiones como ESLint, Prettier y React Developer Tools para mejorar productividad.

# Creación de proyectos en React

## Create React App (CRA)

```
C:\> npx create-react-app mi-app  
C:\> cd mi-app  
C:\> npm start
```

## Vite

```
C:\> npx create vite@latest mi-app  
C:\> cd mi-app  
C:\> npm run dev
```

## Next.js

```
C:\> npx create-next-app@latest mi-app  
C:\> cd mi-app  
C:\> npm run dev
```

## Create React App (CRA)

CRA automatiza la configuración de Webpack y Babel, facilitando el desarrollo pero con menor flexibilidad y mayor peso.

## Vite

Vite es una herramienta ligera y rápida que ofrece arranque instantáneo y recarga en caliente eficiente para proyectos React.

## Next.js

Next.js permite desarrollo avanzado con renderizado del lado del servidor, rutas automáticas y optimización para producción.

# Estructura inicial del proyecto

FICHERO/DIRECTORIO	DESCRIPCIÓN
<b>mi-app/</b>	Directorio contenedor de la aplicación
<b>Node_modules/</b>	Dependencias instaladas
<b>Public/</b>	Archivos estáticos (favicon, index.html)
<b>Src/</b>	Código fuente (App.js, App.css, index.js, index.css)
<b>Package.json</b>	Configuración del proyecto y dependencias
<b>Package-lock.json</b>	Consistencia de dependencias
<b>README.md</b>	Documentación inicial

Directarios y archivos principales

```
mi-app/
├── node_modules/
├── public/
│   ├── assets/          # Recursos públicos servidos tal cual
│   │   ├── images/
│   │   │   └── fonts/
│   │   ...
│   └── _redirects / vercel.json / netlify.toml # Config de hosting (opcional)
└── src/
    ├── assets/          # Recursos empaquetados por el bundler (importables)
    │   ├── images/
    │   │   └── ...
    │   └── styles/
    ├── components/
    │   └── Button/
    ├── pages/            # (opcional) si organizas por vistas
    ├── routes/           # (opcional) configuración de react-router
    ├── hooks/
    ├── context/
    ├── services/         # APIs, clients, utils de datos
    ├── utils/
    ├── styles/
    ├── App.jsx
    ├── main.jsx          # Vite
    ├── index.jsx          # CRA usa index.js como entry
    └── vite-env.d.ts      # si usas TypeScript con Vite
    .env                  # variables de entorno (NUNCA subir claves sensibles)
    .env.local             # ignorado por git (por defecto)
    .gitignore
    package.json
    tsconfig.json          # si usas TypeScript
    vite.config.ts | webpack.config.js
    README.md
```