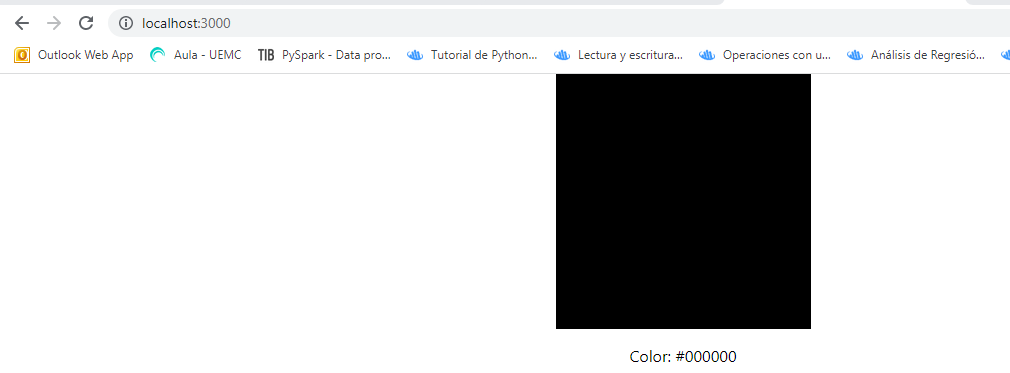
**Ejercicios sesiones 10, 11 y 12**

En este ejercicio tendréis que crear un componente el cual contenga un elemento, puede ser un rectángulo, cuadrado.

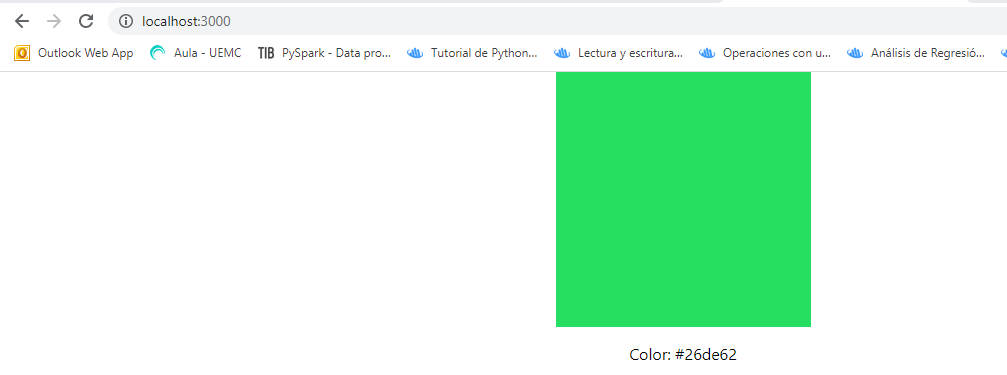
Las dimensiones del elemento serán de ancho 255px y de alto 255px y su color inicial tiene que ser negro.



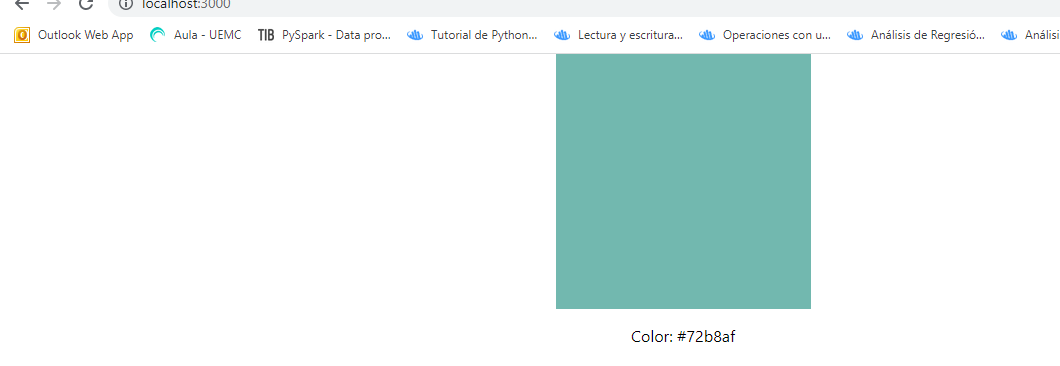
Dentro del componente crearéis varios métodos de captura del ratón que harán lo siguiente para que el color del elemento cambie:

Un método para que cuando el ratón entre en el contenedor, se dará un valor aleatorio (color RGB entre 0 y 255) para cambiar el color del componente.

Un método para que cuando salga el ratón del componente se detenga por completo el cambio de color.



Por último, un método en el que cuando se pulsa dos veces en el componente, se tiene que detener el cambio de color.



**App.css**

.App {

text-align: center;

}

.App-logo {

height: 40vmin;

pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

.App-logo {

animation: App-logo-spin infinite 20s linear;

}

}

.App-header {

background-color: #282c34;

min-height: 100vh;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

font-size: calc(10px + 2vmin);

color: white;

}

.App-link {

color: #61dafb;

}

@keyframes App-logo-spin {

from {

transform: rotate(0deg);

}

to {

transform: rotate(360deg);

}

}

**App.js**

import { useState } from 'react';

import './App.css';

function App() {

const [color, setColor] = useState('#000000');

const [manageInterval, setManageInterval] = useState(0);

const [doubleClick, setDoubleClick] = useState(0);

const getColor = () => Math.floor(Math.random() \* 256);

const getHex = (red, green, blue) => {

return (

'#' +

[red, green, blue]

.map((c) => {

const hex = c.toString(16);

return hex.lenght === 1 ? '0' + hex : hex;

})

.join('')

);

};

const generateHex = () => {

const rgb = {

r: getColor(),

g: getColor(),

b: getColor(),

};

return setColor(getHex(rgb.r, rgb.g, rgb.b));

};

const onChangeColor = () => {

return setManageInterval(setInterval(generateHex, 500));

};

const onStopChangeColor = () => {

return clearInterval(manageInterval);

};

const onClickChangeColor = () => {

return clearInterval(manageInterval);

};

return (

<div className="App" style={{ margin: 'auto' }}>

<div

id="square"

onMouseOver={onChangeColor}

onMouseLeave={onStopChangeColor}

onDoubleClick={onClickChangeColor}

style={{ width: '255px', height: '255px', backgroundColor: color, margin: 'auto' }}

></div>{' '}

<p style={{ color: 'black' }}>Color: {color} </p>

</div>

);

}

export default App;