**React Hooks - הגדרה**

פונקציות מיוחדות של REACT שמאפשרות תכונות מיוחדות לכך Component ללא צורך לכתוב Class מיוחד לצורך כך.

**useState** Hook

הפונקציה החשובה ביותר היא: useState

תפקיד: לשמש כמאגר שמחזיק מידע שניתן לעדכן אותו באמצעות פונקציה מיוחדת: setState כאשר משתנה זה מתעדכן, ה – Component מתעדכן באופן אוטומטי.

פקודה בסיסית:  *const [value, setValue] = useState(initialValue);*

בכל פעם שנפעיל:*setValue(newValue)* ; value = ה – Component יצטייר מחדש.

אם לא היינו משתמשים ב - **useState** ופועלים בפקודות JS רגילות ה – Component לא היה עובד נכון כי בכל render המשתנה היה מאותחל מחדש ולא היה זוכר את הערך הקודם שלו.

לדוגמא: אם היינו כותבים כך:

*function MyComponent() {*

*let count = 0*

*function setCount() {*

*count++;*

*console.log(count); // This will increase in your console*

*}*

*return <div>*

*{count} {/\* This won't update in the UI \*/}*

*<button onClick={setCount}>Click me</button>*

*</div>;*

*}*

זה לא היה עובד כי:

1. בכל Render המשתנה Count יהיה שווה ל – 0.
2. React לא יודע על השינוי.

השימוש ב – - useState

1. גורם למשתנה להפוך ל – Persistent מבחינת REACT.
2. השימוש בפונקציה set…() מידע את REACT שה – component צריך להצטייר מחדש.

**תחליף לשימוש ב – פונקצית useState – שימוש ב – Class React.Component**

**class test extends React.Component**

|  |  |
| --- | --- |
| **שימוש בפונקציה** | **שימוש ב - Class** |
| **import { useState } from 'react';** | **class test extends React.Component** |
| **const [name, setName] = useState('Taylor')**  **const [age, setAge] = useState(42);** | **state = {**  **name: 'Taylor',**  **age: 42,**  **};** |
| **setName(e.target.value);**  **setAge(age + 1);** | **this.setState({**  **name: e.target.value**  **});**  **this.setState({**  **age: this.state.age + 1**  **});** |