React

שיעור רביעי

ניווט בין עמודים באמצעות useNavigate()

- ה useNavigate hook משמש לניווט עמודים בצורה הטובה ביותר מכמה סיבות:
  + מעביר לעמוד שונה לגמרי. ל URL שונה.
  + שומר את העמוד הקודם ממנו בא, כך השימוש הנפוץ של ניווט קדימה או אחורה קלה ביותר.
- useNavigate מחליף את useHistory בשנים האחרונות. והוא המתקדם ביותר.
- שלבים ב components:

ייבא את useNavigae מ react-router-dom

```
import {useNavigate} from "react-router-dom"
```

ב JS - יצירת משתנה המכיל את פונקציית useNavigate

```
const navigate = useNavigate();
```

בכפתור ממנו ננווט לעמוד הבא נשתמש במשתנה navigate לעמוד אליו נרצה לנווט:

```
<button onClick={()=>navigate("/about")}>About</button>
```

ניווט לעמוד אחורה useNavigate()

- בשביל לחזור לעמוד הקודם – בניווט העמוד נשים את הערך מינוס (-) ומספר לפי העמוד אליו נרצה לנווט מהשורש ועד לעמוד הנוכחי. לדוגמא – אם מהעמוד home עברנו לעמוד about ונרצה משם לעבור חזרה ל home נשים את הערך (-1) כי זה חזרה לעמוד אחד אחורה.

```
<button onClick={()=>navigate(-1)}>Go Back Home</button>
```

- ואם מתוך ה home ננווט לעמוד about, מהעמוד about ננווט לעמוד נוסף (לדוגמא history), ומתוך העמוד השלישי history נרצה לחזור חזרה ל home – נשים 2- ב navigate כי זה חזרה ל2 עמודים אחורה.

```
<button onClick={() => navigate(-2)}>Go Back Home</button>
```

useNavigate() הפעלה (לדוגמא בApp)

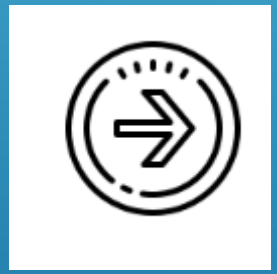
- על מנת לנווט את העמודים נשים את העמודים ביניהם מנווטים בתוך אובייטים של react-router-dom. כדוגמת הצורה הבאה:

```
<BrowserRouter>
  <Routes>
    <Route exact path="/" element={<Home />} />
    <Route exact path="/about" element={<About />} />
    <Route exact path="/history" element={<History />} />
  </Routes>
</BrowserRouter>
```

- נותן את כח הניווט בין העמודים, עמוד שיופעל ולא יכלל באובייקטים אלו לא ינווטץ

צורות ניווט בין עמודים מסקנה

- הדרך הראשונה שלמדנו – ניווט בין עמודים מתאימה לניווט כמו בnav bar וכד'.
  כך את כל הלינקים רואים תמיד ורק תוכן העמוד משתנה.
- בניווט בין העמודים עצמם עם צורך ניווט קדימה ואחורה לדף אחר – נשתמש ב useNavigate()



React

Timer

- באתרים רבים נציג timer למשתמש. הרבה פעמים לאחר זמן מסוים שלא נעשה פעולה באתר מתבצע יציאה מהאתר לצורך אבטחת מידע של המשתמש.
- לבצע timer בדרך הפשוטה ב react hooks נשתמש בספריית react-timer-hook שלבים:
התקנת הספרייה: הרצת הפקודה הבאה: npm i react-timer-hook
ניתן להיכנס לקישור הבא ולהעתיק דוגמא מוכנה עם שימוש ב Timer:
https://www.npmjs.com/package/react-timer-hook הסבר על הדוגמא נמצא ג"כ בקישור.
- המון פעמים בהם צריך לבצע פיצ'ר מסוים מומלץ לבדוק קודם בספריות ה npm בד"כ תמיד נמצא מדריך כזה עם התקנה וקוד לדוגמא עם הסבר מפורט.

Reach Children

- React Children מאפשר לנו שימוש בהפעלת קומפוננט עם רכיבים הקיימים בתוכו.
- props.children הוא אביזר מיוחד, המועבר אוטומטית לכל רכיב, שניתן להשתמש בו כדי להציג את התוכן הכלול בין תגי הפתיחה והסגירה בעת הפעלת קומפוננט.
- שלבים:

1. נשים ערך/ ערכים בתוך אובייקט ה component (במיקום הפעלת הקומפוננט כמו ב App.js)

```
<FirstChild><b>This is children</b></FirstChild>
```

2. בתוך ה component נקבל props. במיקום שנרצה נשים {props.childten} - יציג את הערך ששמנו בהפעלת הcomponent.

```
function FirstChild(props) {
    return (
        <div>
            <div>{props.children}</div>
        </div>
    );
}
export default FirstChild;
```

React

פיצול ה childrens

- ה childrens יכולים להיות גם קומפוננטות ממש.
- ניתן גם בקומפוננט המכילה childrens לדעת כמה childrens קיימים, ולשלוט על איזה child יוצג ואיפה. בשביל זה:
- ניצור משתנה המכיל את מספר ה childrens:

```
const numberOfChildrens = React.Children.count(props.children);
```

יש לשים לב, כשאנחנו משתמשים באובייקט React יש לייבא אותו מ react

```
import React from "react";
```

- בשביל להציג child מסוים נצטרך לבצע כמה שלבים:

ליצור מהchildrens מערך:

```
const childrens = React.Children.toArray(props.children);
```

להוציא מהמערך אובייקט על המיקום במערך אותו נרצה להציג:

(cloneElement) היא פונקציה מובנת המאפשרת את מה שמקבלת לאובייקט המתאים.

```
const singleChild = React.cloneElement(childrens[1])
```

- הצגת child בודד:

```
<div>{singleChild}</div>
```

שימוש נפוץ – slider – עזרה בשיעורי הבית

- לדוגמא slider הבנוי מקומפוננט המכיל כמה רכיבים בתוכו (מספר קומפוננטות) בכל פעם מציג עמוד אחד – child אחד.
- נרצה לשים שני כפתורים next, previous בלחיצה על הכפתור נרצה לשלוח אינדקס המכיל איזה child להציג:
  1. בשביל לקבל child של אינדקס מסוים – ניצור פונקציה המקבלת אינדקס ועושה את הפקודות הרלוונטיות לשליפת child ספציפי כמפורט בעמוד הקודם על האינדקס אותו קיבלה.
  2. נגדיר משתנה המכיל את מספר העמוד עליו עומדים עכשיו.
  3. בכל לחיצה על כפתור next – נגדיל משתנה זה ב 1, ובכל לחיצה על previous נקטין אותו ב 1.
  4. נציג את ה child הנוכחי באמצעות קריאה לפונקציה עם שליחת המשתנה המכיל את האינדקס.
- בונוס – בשביל להציג את האייקון של > ו < בכפתורים next, previous בצורה הבאה:

```
<button className="btn btn-danger"> Next &gt; </button>
```

```
<button className="btn btn-danger"> &lt; Previous</button>
```

React

שיעורי בית:

- Slider של תמונות עם כפתור next, previous. בלחיצה על הכפתורים התמונה משתנה. שימוש ב react children המכיל קומפוננטות כמספר התמונות (לכל תמונה – קומפוננט)
- ניתן להיעזר בהסברים בעמוד הקודם.
- **אתגר** בלחיצה על הכפתורים הכפתור next לא יהיה מאופשר כאשר תוצג התמונה האחרונה (ה child האחרון) והפוך ב previous.

בהצלחה!!