React

תוכנת קוד פתוח שפורסמה על ידי פייסבוק (מטא) שמטרתה בניית אתרי אינטרנט מורכבים.

ריאקט היא פריימוורק (סביבת פיתוח) מבוססת קומפוננטות.

קומפוננטה היא רכיב ויזואלי.

ה-ui מחולק לחלקים קטנים, כל אחד מהם הוא קומפוננטה, שגם יכולה להכיל בתוכה קומפוננטות אחרות.

ריאקט היא SPA – Single Page Application

התקנות

* Npm I –g create-react-app
* בדיקה שעובד: create-react-app –version
* יצירת אפליקציית ריאקט חדשה:

create-react-app kitten-website --template typescript

* יצירת קומפוננטות אוטומטית: npm i -g react-cli-snippets
  + ואז: create fc home/Home

הרצת האפפ: npm start

Tsx – typescript extended

טייפסקריפט שאפשר לכתוב בתוכו קוד html

סוגי קומפוננטות:

* פעם נהגו לבנות Class components, הסתבר שזו עבודה מאד מורכבת וסזיפית
* כיום נהוג לעבוד עם Function components

ניתן לרנדר רשימות בריאקט באמצעות ה-HOF map וריאקט חוסכת לנו את הצורך ב-reduce. עם זאת, ריאקט מכריחה אותנו לתייג כל תגית מרשימה באמצעות מאפיין הנקרא key. בדרך כלל נרנדר לתוך ה-key את שדה ה-id של הרשומה.

על button ניתן להוסיף onClick ולעשות אינטרפולציה לפונקציה. למשל:

            <button onClick={displaySale}>Display Sale</button>

State

מצב. ספציפית בעולם התוכנה, מתאר את חתימת הדאטה הנוכחית שיש לאובייקט מסויים.

במקרה שלנו, האובייקט הוא קומפוננטה. בריאקט, הקומפוננטה מנהלת את הדאטה וכדי לשנות, ליצור, למחוק, כל דבר שקשור לדאטה, עלינו להשתמש בכלים של ריאקט ולא לנסות למשל לשנות דאטה על ידי שינוי ערך של משתנים. פשוט כי זה לא יעבוד.

Side Effect

לפעמים, קומפוננטה צריכה לבצע פעולות מחוץ לתחום גזרתה: ללכת להביא json משרת מרוחק. למשל לשאוב מידע מקומפוננטה אחרת. למשל להשתמש במשתנים גלובליים (כמו window).

כדי לבצע פעולות כאלה, אסור בתכלית האיסור לבצע אותן בצורה ישירה, מכיוון שאם הן יסתיימו ברינדור (שיבלול) מחדש של הקומפוננטה, מה שיגרום לקריאה מחדש לפעולה החיצונית. וזה דבר לא רצוי.

פעולה חיצונית כזו נקראת side effect והדרך החוקית היחידה לבצע אותה, היא דרך פונקציה של ריאקט שנקראת useEffect

הפונקציית callback שאנחנו מספקים ל-useEffect מתבצעת אך ורק פעם אחת בעת הרינדור הראשון של הקומפונטטה ולא באף רינדור אחר.

אלא אם כן, אחד המשתנים שהעברנו במערך שהוא הפרמטר השני, משתנה.

למשל:

useEffect(() => {

        setInterval(() => {

            console.log('displaying time')

            displayTime()

        }, 1000)

    }, [time])

האפקט בדוגמה הזו, יתרחש פעם אחת בעת רינדור ראשוני של הקומפוננטה, ואז רק אם משתנה ה-time יעודכן. (במקרה זה שוב ניצור סוג של לולאה אינסופית של רינדור – להיזהר ממצבים כאלה)

React strict mode

מוד עבודה דיפולטי של ריאקט, נוצר במהלך create-react-app כתגית הבסיס לאפליקציה ב-index.tsx. ניתן להיפטר ממנו לעת עתה.

Routing

זהו ניתוב פנימי באתר, ונועד להציג את הדף הנכון לפי ה-url שהמשתמש הקליד (או לחץ) וגם, להציג בשורת ה-url את הנתיב הנוכחי.

<https://mysite.com/products> in this case, /products is the route

https://mysite.com/sports/basketball in this case, /sports/basketball is the route

כדי להשתמש בראוטינג, עלינו להתקין npm חדש.

npm i react-router-dom @types/react-router-dom

התקנת שרת מקומי של northwind:

npm i –g northwind-back-end

הבאת נתונים משרת מרוחק:

1. לבנות קלאס עבור קונפיגורציה של המערכת. נהוג לקרוא לקלאס הזה AppConfig ולמקם אותו בתיקיית utils
2. לבנות קלאס מסוג Model. מודלים הם קלאסים שמתארים נתונים, כמו למשל נתונים שמגיעים משרת מרוחק.
3. בניית קלאס מסוג service. קלאסים מסוג זה מכילים קוד לוגי בלבד, למשל שמיישם גישה לשרת מרוחק, אין להם שום UI, והם משמשים כשכבה הלוגית שבאמצעותה הקומפוננטות מיישמות לוגיקה.
4. במילים אחרות: קוד לוגי שלא קשור לתצוגה, לא אמור לשבת בתוך קומפוננטה, אלא בתוך service, והקומפוננטה תשתמש ב-service

npm i axios