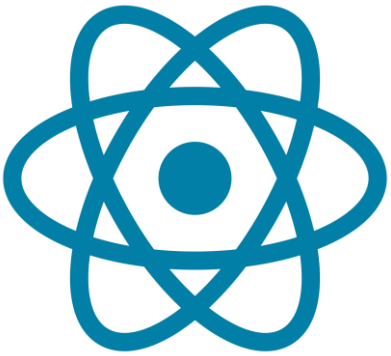
Rauscher Gábor

React alapok
Front-End fejlesztéshez



BBS-INFO Kiadó, 2025.

A könyv nagyobb mennyiségben megrendelhető a kiadónál:
BBS-INFO Kiadó, www.bbs.hu, Tel.: +36-14-07-17-07

Papírkönyv: ISBN 978-615-6364-40-1
E-book: ISBN 978-615-6364-41-8

Kiadja a BBS-INFO Kft. Budapest

Tartalomjegyzék

1. Bevezetés

Mi a React?

A React egy JavaScript könyvtár (framework), amely eszközt kínál a felhasználói felületek komponenseinek leírására. Segítségével olyan felhasználói felületet hozhatunk létre – jellemzően HTML alapú alkalmazást –, amelyben JavaScript működik, és ehhez kapcsolódik a React. A React lehetővé teszi újrafelhasználható összetevők, azaz komponensek megírását a kódban. Ezért a React elsősorban Front-End fejlesztők számára ajánlott, akik a JavaScript lehetőségeit szeretnék kibővíteni.

Milyen előfeltétele van e könyv tanulmányozásának?

A könyv anyagának elsajátításához szükséges a HTML, a CSS és a JavaScript alapismerete. Emellett ajánlott egy kódszerkesztő használata, például a Visual Studio Code. Az összetettebb alkalmazások fejlesztéséhez már nem célszerű csupán egy HTML fájlban hivatkozással behívni a React könyvtárát egy külső webhelyről, hanem érdemes több fájlból álló projektkörnyezetet kialakítani. Ehhez szükséges, hogy a Node.js és az npm (Node Package Manager) telepítve legyenek, használatukhoz pedig parancssort (terminált) kell alkalmaznunk. Ezek telepítése, valamint a terminál használata operációs rendszertől függő, és általános informatikai ismeretnek számít. A könyv feltételezi ezen tudás meglétét az összetettebb eljárások bemutatásakor. A React különféle környezetekben történő elindításáról a 2.2. pontban olvashatunk részletesen.

Miről lesz szó a könyvben?

A könyv első része az ES6 új eszközeit mutatja be. Az ES6 az ECMAScript hatodik verziója, amely a JavaScript szabványosítására jött létre. Bár az itt szereplő eljárások a JavaScriptet jól ismerők számára már ismerősek lehetnek, ebben az esetben a React szempontjából tekintjük át őket. Ez tulajdonképpen egy tudásfelfrissítés a JavaScriptből, amely hasznos alapot nyújt a React eljárások későbbi bemutatásához.

A könyv második fontos része a klasszikus React eljárásokat ismerteti, köztük a React Class típusú komponenseit is. Ennek külön tárgyalása azért indokolt, mert bár a Class típusú komponensek már teljes mértékben helyettesíthetők függvény típusú komponensekkel (Hook-okkal), mégis érdemes mindkét megközelítést megismerni. Gyenge szójátékkal élve: a klasszikus Class jó bevezetés a tömörebb leírású Hook-ok megértéséhez.

A harmadik rész a Hook-okkal foglalkozik, amelyek a React 16.8-as verziójával kerültek bevezetésre. Fontos megjegyezni, hogy a Class típusú komponensek az újabb verziókban is megmaradtak.

A könyv utolsó fejezetében gyakorló feladatokat találunk, amelyek segítenek elmélyíteni és alkalmazni a korábban megszerzett ismereteket.

2. Kezdetek

2.1. A React célja

A React célja, hogy a weboldalak a lehető legtakarékosabban jöjjenek létre. Ezt két módszerrel éri el: az első, hogy az oldalak elemeit a fejlesztő a lehető legtöbb komponensre bontja szét, amelyek aztán újrahasznosíthatók; a második, hogy a komponensek közül a kódfeldolgozó csak azokat hozza létre, amelyek az adott időben és helyen valóban szükségesek, így feleslegesen nem töltődnek be oldalelemek. Kicsit kézzelfoghatóbban: egyszerű JavaScript kódolásban, ha nem akartuk, hogy egy gomb (button) elem látszódjon egy adott feltétel mellett, akkor valamilyen kóddal láthatatlanná tettük, de az elem ettől még létezett a weboldalon, csak nem volt látható. A React ezzel szemben a gombot nem hozza létre, nem tölti be, ha nem szükséges, amíg az adott feltétel fennáll. Ezt a React úgy éri el, hogy először egy virtuális DOM-ot hoz létre a memóriájában, és nem közvetlenül hat a böngésző DOM-jára, csak a szükséges módosításokkal. A React a weboldalon bekövetkező változások esetén kizárólag azokat az elemeket módosítja, amelyek szükségesek, nem pedig az egész weboldal újratöltését kezdeményezi.

2.2. A fejlesztési környezet

Kétféle megoldás használható. Az egyszerűbb módszer a tananyag végéig megfelelő, amíg nem kerül sor több fájl kezelésére; rövidebb kódok esetén jól alkalmazható, amikor mindent egyetlen HTML fájlban írunk. A React kódokat a script elemek, címkék (tag-ek) közé helyezzük egy kódszerkesztő segítségével, és a futás eredményét a kódszer-

kesztőhöz kapcsolt localhost-on (helyi szerveren) látjuk. A szükséges kiegészítések a második eszköz bemutatása után következnek.

A másik megoldás egy projekt indítása, amely több fájlt hoz létre egy könyvtárban (alkönyvtárakkal együtt). Ezek egy része automatikusan generálódik a létrehozás során (a projekt kezeléséhez), míg más fájlokat a saját kódjaink megírására használunk. A projekt létrehozásához szükség van az npm és a Node.js előzetes telepítésére, ezek megléte alapfeltétel. Ebben az esetben a projekt létrehozása és elindítása a parancssorból (terminálból) történik, amely egyúttal elindítja a localhost-ot is. A kódok és fájlok szerkesztése ezután egy kódszerkesztőben könnyebb, mivel az a projekt fájljait faszerkezetben is megjeleníti. Nem tanulás, hanem munka során a projektmódszer használatos. A részletek szintén az alábbiakban következnek.

2.2.1. Közvetlen HTML fájl használata

A közvetlenül HTML fájlba írandó React kódok előtt a következőket tesszük:

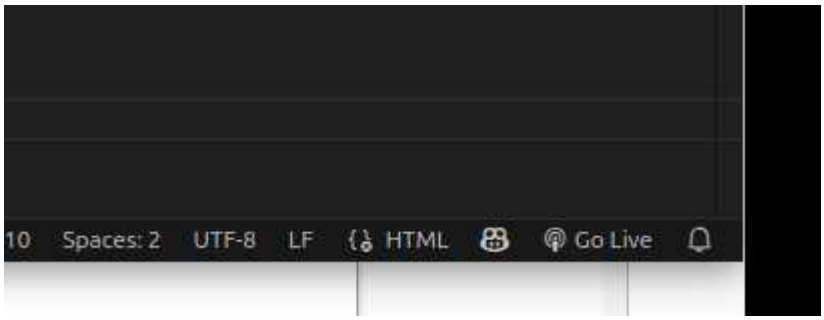
A head részbe három scriptet illesztünk be, melyből az első kettő a React könyvtárakat hozzák, a harmadik a JSX nyelvhez szükséges Babel fordítót. Ezek a CDN-ek (Content Delivery Network – tartalomszétosztó netes hálózat) a cdnjs.com vagy az unpkg.com webhelyekről jönnek, mint források, az alábbi példában mindjárt látszik. A JSX-ről külön lesz majd szó, a lényege, hogy ezzel a szintaktikai kiegészítéssel React-ben írhatunk HTML-t.

A React kódunkat a body részbe illesztett script tag-ek közé írjuk majd. Itt a nyitó script tag-ben meg kell adni a script típusát, mely most: text/babel.

Ezek alapján, így néz ki a kezdeti fájlunk a React 18-as verziójának behívásával:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>

<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
</head>
<body>
    <div id="root">
    </div>

    <script type="text/babel">

</script>
</body>
</html>
```

A localhost indítása Visul Studio Code szerkesztő esetén a jobb alsó sarokban található: „Go Live” gomb megnyomásával történhet.



2.2.2. Projekt használata a create-react-app parancs után

Projekt létrehozása a parancssorból az `npx create-react-app` paranccsal történik.

Ehhez azonban előtte installálni kell ezt a létrehozó eszközt az npm-ben. Vagyis: a Node.js telepítésével megkapjuk a kezelő npm-et is, azonban a projekt létrehozó eszközét is telepítenünk kell az első React projekt előtt egy ehhez hasonló paranccsal:

```
npm install -g create-react
```

A konkrét parancs a node.js verziótól (operációs rendszertől) és a választott eszköztől függ. (Van például eszköz, mely a React mellett a hasonló Vue framework-öt is kezeli.)

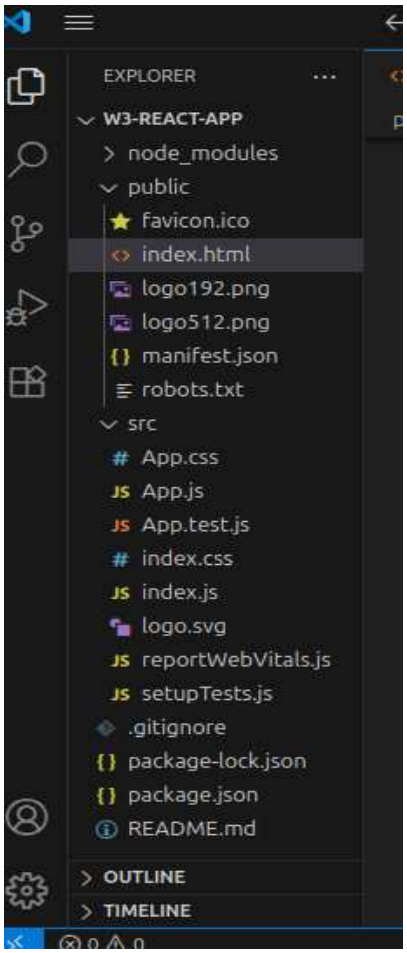
Tehát fentiek megléte után írhatjuk a `npx create-react-app` parancsot, mégpedig abban a könyvtárban, ahol a React projektünket futtatni szeretnénk. Azaz a parancssorban belépünk a könyvtárunkba, majd egyszerűen a parancs után adjuk meg a projektünk nevét, célszerűen utalva arra is, hogy egy React applikációt hozunk létre. Így a `create-react-app` létrehoz egy alkönyvtárat a projektünk nevével és a szükséges fájlokkal. Példa:

```
npx create-react-app elso-react-app
```

A projekt futtatása pedig a parancssorból a következőképpen lehetséges: Először elmegyünk a projektünk könyvtárába, majd egyszerűen az `npm start` parancs indítja a szerkeszthető és futtatható projektet, továbbá megnyitja a helyi szervert (localhostot, általában a 3000-es porton) a böngészőben.

```
cd elso-react-app
npm start
```

A képernyőn, a böngészőben megjelenik a React logoja és tanulásra, szerkesztésre buzdító felirat.



Mivel ezek a kezdő elemek zavaróak lehetnek, az alábbiak szerint ezeket töröljük, csakúgy, mint a kezdés által a tanulásra kínált felesleges fájlokat. Ehhez és a továbbiakhoz kódszerkesztő használata indokolt, a könyv a Visual Code Studio-ból mutatja az eredményeket.

A szerkesztőben látjuk, hogy milyen könyvtárakat és fájlokat hozott létre a create-react-app a projektünkhöz, nagyjából (eszköztől, verziótól függően) úgy, ahogy a csatolt kép is mutatja.

A számunkra most lényeges könyvtárak a public és az src. A többi fájl és könyvtár többnyire a React futását biztosítja a háttérből. Ezekkel most nem kell törődnünk.

A public könyvtár tartalmazza a megjelenő fájlokat, mindenekelőtt az index.html fájlt, ezt látjuk majd futás után