Introduzione

React non è un *framework*. Molti framework usano React, ma React non lo è. ReactNative, Next.js, Gatsby e Remix sono tutti esempi di framework basati su React. Questi framework usano React per creare interfacce utente, perché React è una libreria specializzata nella *creazione* di queste interfacce. Questo semplice concetto, però, è una delle maggiori cause di confusione per chi si avvicina a React per la prima volta, e non è la sola… Qual è la differenza tra un framework e una libreria? Dove finisce React e dove inizia il framework?

Proprio il suo essere "solo una libreria" ha portato allo sviluppo di un vasto e rigoglioso ecosistema di librerie da usare insieme a React per costruire applicazioni.

Tutte questa possibilità hanno, però, portato a un aumento della complessità. Infatti, ci sono molti modi diversi per fare la stessa cosa con React. Per esempio, scegliere il modo in cui gestire lo stile dei componenti mette di fronte ad almeno tre opzioni diverse; per comunicare con delle API ce ne sono altrettante e per la gestione dello stato dell'applicazione le possibilità diventano ancora di più. Come è possibile compiere una scelta corretta?

Per tutti questi motivi, se chiedete a un esperto se React sia semplice, è probabile che vi risponderà dicendovi che "React è semplice se lo sai usare già". Infatti, nonostante la sua enorme diffusione, React rimane una tecnologia "impegnativa". Perché React è semplice, sì, ma non banale, e lavorarci richiede di pensare seguendo la sua logica che, anche se può sembrare un paradosso, spesso risulta più difficile da comprendere a chi ha già molta esperienza di sviluppo. Una volta superato lo scoglio iniziale, la difficoltà principale che rimane è quella rappresentata dal *come creare* e *come far comunicare* tra loro i componenti dell'interfaccia, in modo che siano riutilizzabili, flessibili e semplici da usare.

La prima versione open source di React risale al 2013 ed è stata creata all'interno di Facebook proprio con l'obiettivo di migliorare lo sviluppo e la manutenzione delle sue interfacce. Funzionalità dopo funzionalità, le applicazioni Facebook erano cresciute in dimensioni e complessità, e le interfacce erano diventate lente. Per risolvere questi problemi di performance e ridurre la complessità delle interfacce, Jordan Walke, uno sviluppatore di Facebook, nel 2011 creò un prototipo di quello che, due anni più tardi, sarebbe diventato React.

Nel 2019, con la versione 16.8.0 e l'introduzione degli *hook*, React ha subito una trasformazione radicale, che lo ha fatto diventare come lo conosciamo oggi. Questa versione ha avviato un processo di mutamento dell'intero ecosistema React, e i cambiamenti positivi che ha provocato non si sono ancora esauriti; purtroppo, però, questa nuova

via intrapresa da React ne ha aumentato la complessità, perché lo ha spostato verso un approccio più funzionale, abbandonando il precedente modo di utilizzo, che era simile a quello dei linguaggi di programmazione a oggetti, con classi e metodi, cui la maggior parte degli sviluppatori era abituato.
Questa trasformazione ha reso, in un attimo, non più adeguato molto del materiale disponibile per l'apprendimento di React. Infatti, decine di volumi, guide e video creati fino al 2019 non sono aggiornati, e gli esempi riportati sono spesso contrastanti tra loro e creano ulteriore confusione a chi si sta avvicinando a React.
L'obiettivo di questo libro è quello di fornire un'introduzione a React che sia pratica e strutturata. Per motivi di spazio non è possibile, oltre che superfluo, cercare di coprire in poco spazio tutte le possibili caratteristiche di React, poiché alcune di queste sono avanzate e richiedono altre conoscenze, ben comprese e consolidate. Questo volume cerca, quindi, di illustrare i concetti base di React, così da permettere al lettore di sviluppare in autonomia le intuizioni sul suo funzionamento. Attraverso le spiegazioni e la pratica di sviluppo di un'applicazione, il lettore potrà comprenderne il reale funzionamento, come usarlo in modo produttivo e avere una base solida da cui iniziare il proprio percorso personale all'interno del suo vasto ecosistema. Il volume è completato da quattro appendici che contengono ulteriori informazioni su React e il suo ecosistema.
Grazie alla conoscenza di React, il lettore potrà anche accedere a tutte quelle tecnologie contemporanee che lo usano per costruire applicazioni più grandi e complesse. React è quindi una conoscenza chiave per chi sviluppa software.

A chi è rivolto questo libro

Il mondo dello sviluppo front-end è spesso circondato da un'aura di magia che sembra voler celare quello che è veramente, cioè normale sviluppo software. Allo stesso modo, molti sviluppatori tendono a sottovalutare la complessità e le conoscenze necessarie per poter realizzare un front-end moderno, perché pensano che si tratti ancora di codice *jQuery* messo tra due tag `<script>` dentro un pagina HTML che fa delle chiamate a `show()` e `hide()`. In realtà, lavorare oggi sul front-end richiede non solo ottime conoscenze nel linguaggio utilizzato (per esempio, JavaScript), ma anche una padronanza completa dello strumento usato e delle sue caratteristiche.
Questo libro si rivolge, quindi, a tutti coloro che non conoscono ancora React e che vogliono comprenderne le basi, a tutti coloro che hanno lavorato o che lavorano con altre librerie per la creazione di interfacce utente e ora vogliono ampliare le loro conoscenze e a tutti coloro che già conoscono o lavorano con React e vogliono verificare le proprie conoscenze per migliorare la qualità del proprio lavoro.

Prerequisiti

La conoscenza richiesta per leggere questo libro è quella minima di HTML, CSS, JavaScript. Se necessario, potete leggere uno dei tanti libri disponibili su questi argomenti nel catalogo Apogeo o consultare i tanti siti disponibili online. Per quanto riguarda JavaScript è consigliato il volume *JavaScript – la guida definitiva* di David Flanagan, poiché tratta tutte le caratteristiche moderne di JavaScript necessarie per lavorare con React.

Nel corso del volume saranno comunque approfonditi alcuni aspetti di JavaScript che sono fortemente legati a React.

L'approccio

In tutto il libro svilupperemo un singolo progetto partendo da zero, così da esaminare ogni aspetto di React in un contesto d'uso reale e non solo tramite esempi isolati e decontestualizzati. Inoltre, useremo solo pochissime librerie esterne, così da ridurre al minimo la confusione iniziale tra quello che è possibile fare solo con React e cosa no. Il progetto che realizzeremo è un'applicazione per la gestione delle attività, o *Todo App*. Questa applicazione per gestire elenchi di "cose da fare" è simile alle tante già esistenti e che probabilmente avete già usato almeno una volta: *Microsoft To Do*, *Todoist*, *Any.do*, …

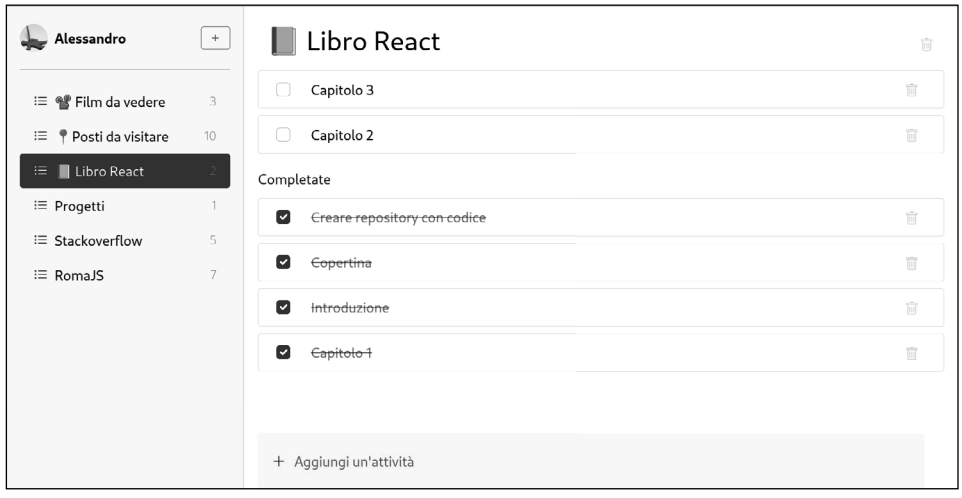

Figura I.1 Una schermata dell'applicazione che svilupperemo.

Sviluppando l'applicazione tratteremo tutti gli argomenti principali che riguardano React, guardando ognuno di essi nella sua interezza, andando a fondo di ogni aspetto e cercando di capire non solo i *come*, ma anche i *perché*: cercheremo di spiegare ogni riga di codice che scriveremo e ogni cambiamento che faremo.

Il codice

Tutto il codice usato nel libro è scaricabile dal sito di Apogeo all'indirizzo `https://bit.ly/apo-react` e da GitHub all'indirizzo `https://github.com/lifeisfoo/react-guida-pratica`. Nel *repository*, troverete tante cartelle quanti sono i capitoli del libro: ogni cartella contiene la versione finale del codice scritto o modificato nel capitolo corrispondente. Inoltre, ogni capitolo si sviluppa a partire dal codice creato in quello precedente. Quindi, per esempio, per lavorare sul Capitolo 3, partiremo dal codice contenuto nella cartella del Capitolo 2.

Nel momento in cui leggerete queste pagine, è probabile che saranno state pubblicate altre versioni di React o delle librerie usate. Se una di queste versioni porterà cambiamenti che rendono errato il codice proposto, il repository sarà aggiornato appena possibile. Per ricevere una notifica ogni volta che il codice viene modificato, potete usare la funzione "watch" nel repository di Github.

La struttura

Questo libro è composto da undici capitoli e quattro appendici. Ogni capitolo ha un preciso scopo ed è dedicato a uno o più aspetti di React.

Nel Capitolo 1 realizzeremo la prima ed elementare versione della nostra app, che mostrerà un semplice messaggio di benvenuto. Nel farlo, introdurremo alcuni concetti essenziali legati a React, come il *virtual DOM*, *JSX*, gli attributi, gli elementi e, infine, i componenti. Non sarà necessario installare nulla.

Nel Capitolo 2 porteremo la nostra applicazione in locale, prima in un singolo file e poi in un progetto strutturato in più file con aggiornamento automatico a ogni modifica. Durante questo processo vedremo qual è il ruolo di *Babel* in un'applicazione React, come si collega un'applicazione a un elemento del *DOM* e come dividerla in più componenti usando più file.

Nel Capitolo 3 personalizzeremo il messaggio di benvenuto dell'applicazione, mostrando il nome e l'immagine di profilo dell'utente collegato. Lo faremo sfruttando le *prop*, di cui vedremo a fondo le caratteristiche, il modo in cui vengono passate e ricevute dai componenti, e come impiegare i valori di default. Infine, vedremo come usare l'*interpolazione* in JSX.

Nel Capitolo 4 daremo all'applicazione un *layout*, aggiungendo una *sidebar* e dello stile: questo ci permetterà di capire come usare un file *CSS* esterno, come aggiungere uno stile *inline* ai componenti e come usare la prop *children*. A questo punto, l'applicazione avrà la *sidebar* visivamente completa, con il nome, l'immagine dell'utente e gli elenchi di cose da fare.

Nel Capitolo 5 completeremo l'interfaccia dell'applicazione, ma non prima di aver rifattorizzato tutti i componenti già creati, così da nascondere all'interno dei componenti tutti i dettagli relativi allo stile. Vedremo inoltre la differenza tra componenti generici e specializzati, fino ad arrivare al concetto di composizione. Facendo questo inizieremo a capire che cosa vuol dire *pensare in React* e qual è il flusso da seguire per sviluppare rapidamente interfacce con React.

Nel Capitolo 6 renderemo interattiva l'applicazione, permettendo all'utente di scegliere quale lista di attività visualizzare. Nel fare ciò, scopriremo come mantenere uno stato all'interno di un componente con l'hook `useState`, come reagire alle azioni dell'utente e aggiornare di conseguenza l'applicazione. Questo ci permetterà di guardare nel dettaglio in che modo React aggiorna e mantiene aggiornati i componenti visualizzati a ogni *rendering*.

Nel Capitolo 7 permetteremo all'utente di creare una nuova attività usando un campo di input. Nel farlo scopriremo che cosa sono i controlled component e come funziona il flusso di aggiornamento dell'interfaccia di React. Infine, aggiungeremo all'applicazione una vista da mostrare quando non è selezionata nessuna lista, così da indicare all'utente che cosa fare per proseguire.

Nel Capitolo 8 aggiungeremo tutte le azioni mancanti sulle attività. Inizieremo con la possibilità di completare un'attività usando una checkbox interattiva e continueremo con la cancellazione di un'attività. Quest'ultima azione richiederà una conferma esplicita da parte dell'utente, con una piccola modal inline. Durante queste modifiche vedremo la differenza tra stato locale di un componente e stato globale dell'applicazione.
Nel Capitolo 9 permetteremo all'utente di creare, modificare ed eliminare una lista. Per avere la conferma della cancellazione, aggiungeremo una modal di conferma a tutto schermo. Per modificare il nome della lista creeremo un componente generico riutilizzabile. Vedremo come sfruttare la prop key per forzare l'aggiornamento dell'interfaccia.
Nel Capitolo 10 collegheremo l'applicazione a delle API, per creare, modificare, leggere o eliminare le attività e le liste. Nel farlo scopriremo l'hook `useEffect` e come funziona. Inoltre, vedremo come usare CRA come proxy tra l'applicazione locale e delle API remote. Infine, capiremo come mostrare un errore personalizzato in caso di problemi con le API.
Nel Capitolo 11 faremo una build dell'applicazione, così da poterla usare anche in produzione. Vedremo poi come fare il deploy dell'applicazione usando un server web dedicato o come renderla disponibile in un servizio web che espone anche delle API.
Nell'Appendice A vedremo come lavorare in modo sicuro con React. In particolare, scopriremo come usare lo strict mode durante lo sviluppo, per trovare in anticipo potenziali bug. Poi, useremo i prop types per aggiungere una validazione sui tipi di valore ricevuti dai componenti. Vedremo come configurare il progetto per poter usare TypeScript. Infine, scriveremo i nostri primi due test unitari, per verificare in modo automatico il funzionamento dei componenti e accenneremo a quali strumenti usare per eseguire i test end-to-end.
Nell'Appendice B faremo una panoramica di alcune caratteristiche di React che non abbiamo affrontato durante lo sviluppo dell'applicazione. Inizieremo facendo un approfondimento sugli hook, su come crearne di personalizzati e sul funzionamento di altri hook comuni. Vedremo poi il context, i componenti definiti come classi, gli error boundary, gli eventi, e gli uncontrolled component.
Nell'Appendice C vedremo come installare Node e npm sulla nostra macchina, quale editor usare per lavorare con React. Infine, elencheremo alcune caratteristiche degli strumenti che abbiamo usato per creare il progetto.
Nell'Appendice D cercheremo di capire come procedere nel vasto mondo di React, quali librerie studiare e utilizzare e soprattutto perché farlo.
Il libro è scritto per essere letto dall'inizio alla fine, perché segue lo sviluppo dell'applicazione.

Convenzioni tipografiche

Il codice sorgente viene mostrato usando un carattere tipografico dedicato come `let react = true`. In alcuni casi il codice compare direttamente nel testo, in altri casi in modo separato all'interno di listati. Alcuni di questi riportano l'intero contenuto del file, altri solo una parte, per brevità: in questi casi vedrete un commento `//...` prima e dopo il codice mostrato.

Terminologia

Il nome dei concetti legati a React o, più in generale, allo sviluppo, sono riportati nella loro forma inglese. Quindi, per esempio, non troverete scritto *funzione freccia* ma *arrow function*. Il motivo di questa scelta è l'utilizzo ormai dominante della forma inglese in tutti i testi e siti o nella documentazione; in questo modo sarà più facile accedere a ulteriori informazioni cercando il nome originale nei motori di ricerca o nei siti dedicati.

Errori e feedback

Potete segnalare errori, affermazioni non chiare o passaggi complicati scrivendo nella sezione Discussions del repository pubblico indicato sopra. Per altre richieste potete contattare direttamente l'autore all'indirizzo `lifeisfoo@gmail.com`.
L'autore cercherà di rispondervi e chiarire per voi e per gli altri lettori tutto quello che non risulterà chiaro dal testo.

Ringraziamenti

Questo volume non sarebbe stato possibile senza tutte le persone che hanno partecipato negli ultimi anni ai miei talk e workshop su React. Grazie ai loro feedback è stato possibile capire alcune delle caratteristiche che rendono React così complicato da comprendere.
Grazie a tutti i colleghi e le colleghe con i quali ho avuto modo di lavorare su React: tutti i problemi incontrati e risolti nei vari progetti hanno contribuito in modo positivo alla comprensione di React.
Grazie a tutte le persone che, dal 2013, portano avanti una delle più longeve community dedicate a JavaScript e al mondo del front-end: RomaJS (`https://romajs.org`). Senza le possibilità di condivisione e confronto offerte da questa comunità, questo libro non avrebbe mai visto la luce.
Un ringraziamento speciale va a Guido D'Orsi, Fabrizio Vitale e Emanuele De Cupis per il tempo dedicato alla revisione di alcuni capitoli e le numerose correzioni suggerite. Qualsiasi errore o imprecisione rimane responsabilità unica dell'autore.