



UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Dipartimento di Scienze e Metodi
dell'Ingegneria

Tecnologie Web e Internet of Things

Stefania Monica
stefania.monica@unimore.it



Tecnologie Web e Internet of Things

React

React è un insieme di tecnologie prodotte da Facebook e rese disponibili open source per

- Realizzare applicazioni Web single-page
 - Un'applicazione single-page è un'applicazione in cui l'utente vede un'unica URL
 - E' formata da una piccola parte in HTML
 - Tutte le modifiche alla pagina avvengono tramite JavaScript Asincrono
- Realizzare applicazioni mobile
- Massimizzare il riuso di parti di codice tra le varie applicazioni

Per noi React è una tecnologia per realizzare applicazioni single-page

- Usando un'estensione di JavaScript (**JSX**) messa a disposizione da React
- Realizzando codice pensato per essere riusato in varie situazioni
- Realizzando codice robusto e cross-browser
- Producendo codice per il browser pensato per ottimizzare la costruzione della pagina
- Producendo codice per il browser ottimizzato per la velocità e la dimensione

L'utilizzo che facciamo di React è ormai comune per la realizzazione di applicazioni Web importanti

- Facebook
- Netflix
- ...

Per usare React per realizzare applicazioni Web sono necessari:

- **Node.js**, che offre gli strumenti per realizzare applicazioni Web in JavaScript che vengono compilate per massimizzare la velocità e la dimensione
- **React**, che si presenta come un insieme di pacchetti da installare all'interno di Node.js
- Strumenti per realizzare velocemente le applicazioni
 - Un editor efficace, ad esempio Visual Studio Code
 - Un Web Server specifico per lo sviluppo e il debugging, ad esempio **vite**
- Normalmente, una library di componenti grafici evoluti e che seguono gli standard grafici comuni
 - Ad esempio, la library **MUI** (Material User Interface)

In alternativa a tutto questo, si può usare <https://stackblitz.com>

JSX è un'estensione di ECMAScript 6

- Aggiunge ad ECMAScript 6 la possibilità di scrivere tag HTML direttamente nel sorgente
 - Usando () nella parte JavaScript si costruiscono delle espressioni tramite tag
 - Usando {} nella parte dei tag è possibile valutare un'espressione JavaScript
- Aggiunge una gestione dello stato dell'applicazione che si basa sulle promise, ma che viene offerta al programmatore in modo semplice
- Ha come scopo principale la produzione di single-page application ottimizzate per velocità e dimensione

Utilizzeremo solo una parte ridotta di JSX, ma sufficiente per scopi anche professionali

- JSX usa pesantemente i **moduli** di ECMAScript 6, che essenzialmente
 - Sono file con estensione JSX, che importano ed esportano funzioni
 - Per importare funzioni si usa la parola chiave **import**
 - Tutto quello che viene esportato viene etichettato con **export**
- JSX offre due tipi di export
 - Export di default, che vengono importati senza parentesi graffe
 - Export non di default, che vengono importati utilizzando le parentesi graffe

Tecnologie Web e Internet of Things

- React usa il termine **componente** per un qualsiasi albero HTML
- Per produrre i componenti, e quindi visualizzarli nella pagina, React richiede che vengano costruite delle funzioni JSX
- Queste funzioni devono
 - avere un nome che inizia con la maiuscola e può essere usato come tag
 - essere esportate da un modulo in modalità default
- Il nome di queste funzioni viene utilizzato in JSX come se fossero dei tag, che corrispondono all'albero generato dalla funzione
- Tutte le volte che React ha bisogno di ridisegnare una parte della pagina richiama la funzione che corrisponde al tag da rigenerare
- La modifica della pagina con queste chiamate a funzioni viene ottimizzata in modo da ricostruire unicamente le parti che servono

Tecnologie Web e Internet of Things

- Le funzioni che costruiscono i componenti devono essere senza stato
 - Il loro risultato dipende solo dagli argomenti
 - Il risultato non deve dipendere da variabili globali o da valori della pagina
- Questo serve perché così React può decidere in ogni momento se ricostruire il componente o no
- Per permettere di avere uno stato dell'applicazione React consente di associare dei valori di stato ai singoli componenti
- I valori di stato possono essere ad uso unicamente interno di un componente e quindi il componente stesso definisce i suoi valori
- Oppure, possono essere di interesse anche per un componente che contiene il componente e quindi è il componente più esterno a definire i valori

- Per definire un valore di stato si usa

```
const [get, set] = useState(init)
```

- Questa istruzione costruisce un valore di stato inizializzato a `init` e restituisce
 - una variabile `get` per leggere il valore di stato
 - una variabile `set` che è una funzione che serve per cambiare il valore di stato
- Questa istruzione verrà usata dal componente più esterno interessato allo stato
- I risultati verranno passati ai componenti più interni

Tecnologie Web e Internet of Things

- Siccome React decide autonomamente quando chiamare una funzione per costruire un componente, è vietato usare funzioni asincrone all'interno di componenti
 - Le funzioni asincrone vengono riattivate mediante la coda degli eventi, ma React vuole che tutte le modifiche alla pagina siano fatte prima di processare gli eventi in coda
- Quindi, React mette a disposizione la possibilità di chiamare funzioni dopo che la pagina è stata modificata completamente
- Di conseguenza
 - Se una di queste funzioni chiama una funzione asincrona, il risultato della funzione dovrà essere messo in uno stato
 - Lo stato sarà legato a una parte del componente
 - Quindi, quando il risultato della funzione asincrona è disponibile, verrà ridisegnata la parte del componente che dipendeva da quel risultato
- Queste funzioni eseguite dopo la costruzione dei componenti si chiamano **side effect**

Tecnologie Web e Internet of Things

- Per costruire un (side) effect si può usare

```
useEffect(f, {state})
```

- Questa chiamata forza l'esecuzione della funzione f quando uno dei membri della lista `state` cambia

Attenzione: f non può essere una funzione asincrona, ma può chiamare funzioni asincrone

- createRoot e StrictMode:

<https://react.dev/reference/react-dom/client/createRoot>

<https://react.dev/reference/react/StrictMode>

- useState:

<https://react.dev/reference/react/useState#usestate>

- useEffect:

<https://react.dev/reference/react/useEffect>

Tecnologie Web e Internet of Things

- Ormai le interfacce utente (*User Interface*, UI) sono così complesse che difficilmente si riescono a realizzare direttamente in HTML e CSS
- Normalmente, si utilizzano delle library realizzate appositamente per costruire UI complesse
- *Material User Interface* (MUI) è una library
 - Utilizzabile mediante React
 - Pensata per realizzare UI che si adattano sia al Web che all'ambiente mobile
 - Che offre un aspetto grafico che realizza il *Material Design*, un insieme di specifiche grafiche realizzate da Google e che tutte le App per Android dovrebbero seguire
- Mediante MUI è possibile quindi realizzare single-page application che hanno un aspetto grafico moderno sia per Web che per mobile

- Il sito Web di MUI è: <https://mui.com>
- La documentazione è accessibile da:
<https://mui.com/material-ui/getting-started>
- Per installare MUI in un progetto React è sufficiente scrivere nella Shell

```
npm install @mui/material
```

```
npm install @emotion/react
```

```
npm install @emotion/styled
```

```
npm install @mui/icons-material
```

- Una volta installato MUI, è necessario sostituire i tag HTML con i componenti grafici offerti da MUI

Tecnologie Web e Internet of Things

- Nei casi più semplici, i tag HTML vengono sostituiti come segue
 - `<div>` viene sostituito con [Box](#) o [Stack](#) da `@mui/material`
 - `<p>` viene sostituito con [Typography](#) da `@mui/material`
 - `<input type="text"/>` viene sostituito con [TextField](#) da `@mui/material`
 - `<select>` viene sostituito con [Select](#) da `@mui/material`, incluso in un [FormControl](#) da `@mui/material`
 - `<option>` viene sostituito con [MenuItem](#) da `@mui/material`
 - `<table>` viene sostituito con [Table](#) da `@mui/material` contenuto in un [TableContainer](#) da `@mui/material`
 - `<thead>` viene sostituito con [TableHead](#) da `@mui/material`
 - `<tbody>` viene sostituito con [TableBody](#) da `@mui/material`
 - `<th>` e `<td>` vengono sostituiti con [TableCell](#) da `@mui/material`
 - `<tr>` viene sostituito con [TableRow](#) da `@mui/material`
 - `` per le icone viene sostituito dal componente corrispondente all'icona che si vuole visualizzare, si veda <https://mui.com/material-ui/material-icons>

- Nel secondo esempio
 - [AppBar](#) da @mui/material
 - [IconButton](#) da @mui/material
 - [Toolbar](#) da @mui/material

 - [ThemeProvider](#) da @mui/material/styles
 - [createTheme](#) da @mui/material/styles ([palette](#))

 - [CssBaseline](#) da @mui/material
 - [useMediaQuery](#) da @mui/material
- Brightness7Icon da @mui/icons-material
- Brightness4Icon da @mui/icons-material
- WbSunnyIcon da @mui/icons-material
- AcUnitIcon da @mui/icons-material
- CloudIcon da @mui/icons-material
- ThunderstormIcon da @mui/icons-material