

# React

Intelligenza Artificiale e Tecnologie Web - Laboratorio

# Programma

- Esercizio di creazione di una pagina web statica:
  - uso delle componenti a classe
  - uso delle componenti a funzione
  - uso dello standard ECMAScript 6
- Novità sintattiche introdotte da ECMAScript 6
- Introduzione a React Router per la creazione di Single-Page Applications (SPA) con React.
- Esercizi di creazione di Single-Page applications.

# Esercizio 1 – Galleria di immagini

- Creare una galleria di immagini con due componenti React:
  - *ImageComponent*, che prende in input un'immagine con un url e una descrizione e le inserisce in un div.
  - *Gallery*, che prende in input una lista di immagini e per ognuna crea un image component e le inserisce in un *div*. È fornita di un bottone per aggiungere un bottone
- Usare le componenti React a classe:
  - Effettuare il binding dei metodi per gestire gli eventi.
  - Usare lo stato interno delle componenti per tenere traccia delle immagini caricate.

# Esercizio 1 – Galleria di immagini

- Creare una galleria di immagini con due componenti React:
  - *ImageComponent*, che prende in input un'immagine con un url e una descrizione e le inserisce in un div.
  - *Gallery*, che prende in input una lista di immagini e per ognuna crea un image component e le inserisce in un *div*. È fornita di un bottone per aggiungere un immagine tramite l'inserimento di un url e di un titolo.
- Usare le componenti React a classe:
  - Effettuare il binding dei metodi per gestire gli eventi.
  - Usare lo stato interno della Galleria per tenere traccia delle immagini caricate.

# Esercizio 1 – Galleria di immagini



Colosseo



Linguaggio dei fiori



Intelligenza Artificiale

Aggiungi Immagine

# Esercizio 1 – Galleria di immagini

W3schools Tutorials Exercises Certificates Services Search... Plus For Teachers Spaces Get Certified Sign Up Log in

HTML CSS JAVASCRIPT SQL PYTHON JAVA PHP HOW TO W3.CSS C C++ C# BOOTSTRAP **REACT** MYSQL JQUERY EXCEL XML DJANGO NUMPY

## React Tutorial

- React Home
- React Intro
- React Get Started
- React Upgrade
- React ES6
- React Render HTML
- React JSX
- React Components
- React Class**
- React Props
- React Events
- React Conditionals
- React Lists
- React Forms
- React Router
- React Memo
- React CSS Styling
- React Sass Styling

## React Class Components

[< Previous](#) [Next >](#)

Before React 16.8, Class components were the only way to track state and lifecycle on a React component. Function components were considered "state-less".

With the addition of Hooks, Function components are now almost equivalent to Class components. The differences are so minor that you will probably never need to use a Class component in React.

Even though Function components are preferred, there are no current plans on removing Class components from React.

This section will give you an overview of how to use Class components in React.

Feel free to skip this section, and use Function Components instead.

## React Components

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML via a render() function.

Sol Marina Beach Crete

Creta

Da 123 €

# Esercizio 1 – Galleria di immagini

- Convertire le componenti React da classi a funzioni:
  - Eliminare l'uso dello *state* per tenere traccia delle immagini caricate. Usare le *props*.
  - La funzione *setState* non può più essere usata. Il render dell'elemento *Gallery* dovrà essere esplicitamente richiamato ogni volta che una nuova immagine viene inserita.

# React Router

- Libreria per la gestione delle navigazioni in applicazioni React.
- Permette di creare single-page applications (SPA) con routing dinamico.
- Gestisce la navigazione senza ricaricare l'intera pagina.



# React Router

- React Router introduce un approccio dichiarativo al routing:
  - Le route sono definite come componenti React.
  - Ad ogni route è associato un componente che viene automaticamente renderizzato quando si naviga verso quella route → evita il ricaricamento della pagina
  - La logica di navigazione è semplificata poiché non deve essere più gestita manualmente.
  - Da includere con il link: <https://unpkg.com/react-router-dom@5.2.0/umd/react-router-dom.min.js>

# React Router

```
var BrowserRouter = ReactDOM.BrowserRouter;  
var Route = ReactDOM.Route;  
var Link = ReactDOM.Link;  
var Switch = ReactDOM.Switch;
```

- Deve essere definita una componente per l'implementazione del routing.
  - Solitamente è la componente *root* che graficamente è renderizzata come un menu di navigazione

```
function App() {  
  return (  
    <Router>  
      <div>  
        <nav>  
          <ul>  
            <li>  
              <Link to="/">Home</Link>  
            </li>  
            <li>  
              <Link to="/about">About</Link>  
            </li>  
            <li>  
              <Link to="/users">Users</Link>  
            </li>  
          </ul>  
        </nav>  
  
        <Switch>  
          <Route path="/about">  
            <About />  
          </Route>  
          <Route path="/users">  
            <Users />  
          </Route>  
          <Route path="/">  
            <Home />  
          </Route>  
        </Switch>  
      </div>  
    </Router>  
  )  
}
```

# Esercizio 2 – Routing dinamico

- Trasformare il sito in una single-page application:

## Welcome to the Homepage

This is the main page of the website.

[About](#) | [Contact](#)

## About Us

This is the about page.

[Home](#) | [Contact](#)

## Contact Us

You can contact us at:

Email: [example@email.com](mailto:example@email.com)

[Home](#) | [About](#)

```
1 from flask import Flask, render_template
2
3 app = Flask(__name__)
4
5 @app.route('/')
6 def index():
7     return render_template('index.html')
8
9 @app.route('/about')
10 def about():
11     return render_template('about.html')
12
13 @app.route('/contact')
14 def contact():
15     return render_template('contact.html')
16
17 if __name__ == '__main__':
18     app.run(debug=True)
```

# Esercizio 2 – Routing dinamico

- Trasformare le pagine in componenti React.
- Impostare un routing dinamico per navigare le componenti.
- Sostituire i link di navigazione presenti con un menu di navigazione in alto.
- Nota bene:
  - In questo esempio, è preferibile non usare JSX per il rendering delle componenti.
  - Integrare React su Flask senza il supporto di node.js limita la possibilità di utilizzo di JSX.
  - Nel mondo reale React viene quasi sempre usato con il supporto di node.js. Questo è solo un esempio didattico.

# Esercizio 3 – Flask API + React

- Trasformare la pagina del team in una componente react che riceve le informazioni sul team in formato json dal server tramite AJAX:
  - Creare l'endpoint su flask 'api/team' che restituisce.
- Trasformare la pagina dei libri in una componente react:
  - Collegarlo all'endpoint 'api/books' tramite AJAX per ricevere i libri.
  - Impostare la componente inserire un nuovo libro e inviarlo alle API sempre tramite l'endpoint 'api/books'

# Esercizio 3 – Flask API + React

- Trasformare la pagina del team in una componente react che riceve le informazioni sul team in formato json dal server tramite AJAX:
  - Creare l'endpoint su flask 'api/team' che restituisce.
- Trasformare la pagina dei libri in una componente react:
  - Collegarlo all'endpoint 'api/books' tramite AJAX per ricevere i libri.
  - Impostare la componente inserire un nuovo libro e inviarlo alle API sempre tramite l'endpoint 'api/books'

# React Hooks

- Gli **hook** di React sono una funzionalità introdotta in React 16.8 per gestire lo **stato** e i **cicli di vita** dei componenti basati su funzione, offrendo la possibilità di usare caratteristiche avanzate di React che prima erano disponibili solo nei componenti basati su classe.
- Permettono di:
  - **Gestire lo stato** delle componenti (state management).
  - **Eseguire effetti collaterali** (side effects).
  - **Accedere ad altre funzionalità avanzate** di React come il contesto o i riferimenti (refs).

# useState Hook

- È l'hook che permette di gestire lo stato di un componente function. Restituisce due elementi:
  - Il valore attuale di stato.
  - La funzione per aggiornarlo.

```
var _useState = React.useState(null),  
    teamData = _useState[0],  
    setTeamData = _useState[1];
```

- *setTeamData(newTeamData)* aggiorna la variabile *teamData*.



# *useEffect* Hook

- Gestisce gli effetti collaterali di un componente basato su funzione. Esegue codice in risposta a:
  - Al montaggio, modifica o smontaggio di un componente.
  - Al fetch di dati tramite API
  - Aggiunta di nuovi listener

```
React.useEffect(function () {  
  fetch('/api/team')  
    .then(function (response) { return response.json(); })  
    .then(function (data) {  
      setTeamData(data); })  
    .catch(function (error) { console.error(error); });  
}, []);
```

- UseEffect fa partire l'esecuzione del fetch automaticamente al montaggio del componente.