

MHW3

Simone Squillaci

O46001862

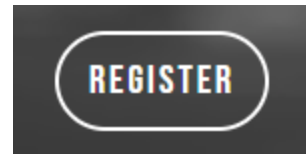
26/04/2022

Descrizione Progetto

Integrazione delle API REST in un sito

API REST Con API KEY

Per la prima API, ho creato una sorta di finestra di registrazione (utilizzando la vista modale e si accede a quest'ultima attraverso il tasto Register) e verifica se l'email appartiene ad un dominio valido.



Registrazione

Email

Password

Avanti Chiudi

API REST Con API KEY

Per riuscire a validare una mail, viene effettuata una richiesta all'endpoint di emailvalidator. Nella richiesta viene aggiunta la key ricevuta dopo la registrazione al sito.



Il tasto chiudi, non fa altro che chiudere la finestra di registrazione una volta cliccato.



Il tasto avanti, invece innesca l'evento di richiesta.

API REST Con API KEY

Al click sul bottone 'Avanti', si attiva l'evento verifica che come prima cosa, prende il testo all'interno del box di testo dell'email e converte attraverso encodeURIComponent la stringa di testo in una stringa elaborabile da URL.

```
// EMAIL  
function Verifica(event){  
  const text = document.querySelector('#email-content').value;  
  console.log(text);  
  const encodedText = encodeURIComponent(text);  
  
  email_request=email_key_endpoint+email_key+'&email='+encodedText;  
  fetch(email_request).then(onResponse).then(onJson);  
}
```

Dopo di che, nell'email_request, si forma l'url che verrà passato come parametro nella richiesta.

API REST Con API KEY

La fetch come prima cosa chiama la funzione onResponse

```
function onResponse(response){  
  //Trasformo la risposta in un json per poter leggere i campi  
  if(response.status===400){  
    console.log('Errore');  
    let result= document.querySelector('#register-window');  
    const controllo=document.querySelector('h3');  
    if(controllo!=null){  
      controllo.textContent='Inserire Email';  
    }  
    else{  
      let result= document.querySelector('#register-window');  
      let text = document.createElement('h3');  
      text.textContent='Inserire Email';  
      result.appendChild(text);  
    }  
    return;  
  }  
  return response.json();  
}
```

Questa controlla se la richiesta è andata a buon fine o no. L' if controlla se la richiesta non contiene una mail o semplicemente se non è andata a buon fine (error == 400) ritornando il messaggio di errore 'Inserire Email' e non ritornando nulla alla funzione onJson. Oppure se andata a buon fine, ritorna il json della promise.

API REST Con API KEY

Alla funzione onJson viene passato il json della risposta.

```
function onJson(json){  
  console.log(json);  
  let result= document.querySelector('#register-window');  
  const controllo=document.querySelector('h3');  
  const validita=json.deliverability;  
  if(controllo!=null){  
    if(validita === 'DELIVERABLE'){  
      console.log('Valido');  
      controllo.textContent='Email Valida';  
    }  
    if(validita === 'UNDELIVERABLE'){  
      console.log('Valido');  
      controllo.textContent='Email non valida';  
    }  
  }  
  return;  
}
```

Nella costante validita, viene assegnato il valore all'interno del json che contiene il risultato della richiesta (se la mail è valida o meno). Mentre alla costante controllo viene assegnato se esiste l'elemento h3 utilizzato per inserire nell'HTML la risposta.

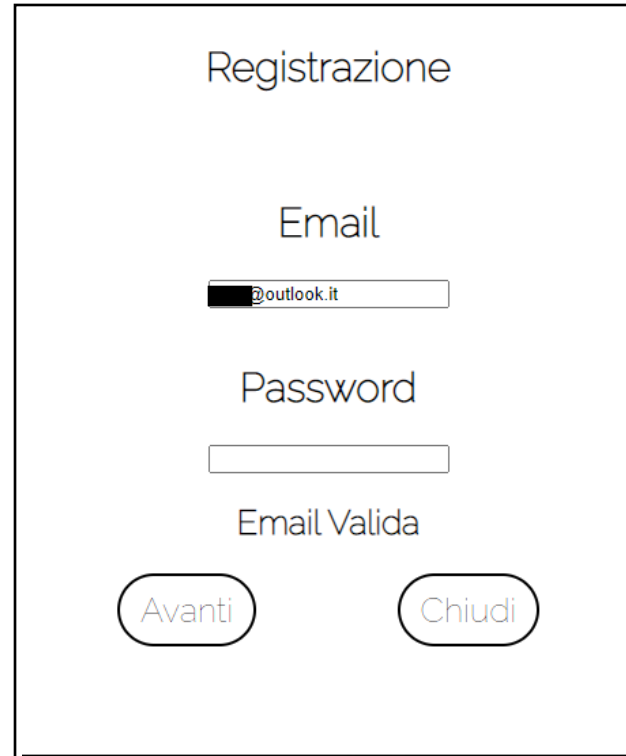
API REST Con API KEY

Successivamente la funzione controlla se il risultato appartiene alla categoria DELIVERABLE o UNDELIVERABLE

```
}  
if(validita === 'DELIVERABLE'){  
  console.log('Valido');  
  let result= document.querySelector('#register-window');  
  let text = document.createElement('h3');  
  text.textContent='Email Valida';  
  result.appendChild(text);  
}  
  
if(validita === 'UNDELIVERABLE'){  
  console.log('Non Valido');  
  let result= document.querySelector('#register-window');  
  let text = document.createElement('h3');  
  text.textContent='Email non valida';  
  result.appendChild(text);  
}
```

Ps: l'if contenente la condizione (controllo != null) controlla se era stata già effettuata una richiesta e quindi se a schermo era già presente una delle tre notifiche possibili.

API REST Con API KEY



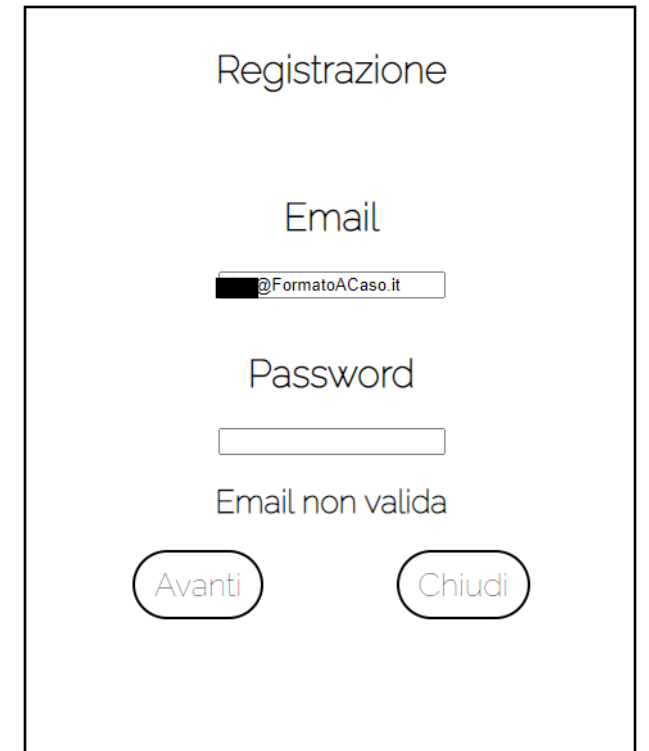
Registrazione

Email

Password

Email Valida

Richiesta andata a buon fine. Caso in cui la mail è valida



Registrazione

Email

Password

Email non valida

Richiesta andata a buon fine. Caso in cui la mail non è valida

API REST Con API KEY

Registrazione

Email

Password

Inserire Email

Avanti Chiudi

Richiesta fallita.

API REST Oauth 2.0

La seconda API consiste nel visualizzare le informazioni principali di una playlist scelta da me (trattata quindi come una costante). Per prima cosa si effettua la richiesta di autenticazione e quindi l'acquisizione del token da allegare alle richieste. Sempre se la richiesta di prima autenticazione va a buon fine.

```
let token;

fetch('https://accounts.spotify.com/api/token',{
  method: 'post',
  body: 'grant_type=client_credentials',
  headers:{
    'Content-Type': 'application/x-www-form-urlencoded',
    'Authorization': 'Basic ' + btoa(client_id+':'+client_secret)
  }
}).then(onResponseToken).then(onToken);
```

API REST

Oauth 2.0

La fetch richiama la funzione on ResponseToken.

```
let token;

fetch('https://accounts.spotify.com/api/token',{
  method: 'post',
  body: 'grant_type=client_credentials',
  headers:{
    'Content-Type': 'application/x-www-form-urlencoded',
    'Authorization': 'Basic ' + btoa(client_id+':'+client_secret)
  }
}).then(onResponseToken).then(onToken);
```

Questa funzione non fa altro che ritornare alla funzione onToken il json della promise.

API REST Oauth 2.0

La funzione onToken assegna alla nostra variabile token il token di ritorno della richiesta di autenticazione.

```
function onToken(json){  
    token = json;  
}  
  
function onResponseToken(response){  
    if(response.status=== 401){  
        console.log('ERRORE AUTORIZZAZIONE SPOTIFY');  
    }  
    return response.json();  
}  
  
function onResponseSpotify(response){  
    if(response.status === 400){  
        console.log('ERRORE RICHIESTA SPOTIFY');  
    }  
    return response.json();  
}
```

PS: Errore 401 gestisce l'errore in caso la richiesta di autenticazione fallisce. Errore 400 gestisce l'errore della richiesta del contenuto.

API REST Oauth 2.0

L'evento di richiesta e successiva visualizzazione delle informazioni della playlist viene scatenato dal click sull'apposito bottone.

Scopri la playlist usata per gli allenamenti!

Clicca qui!

API REST Oauth 2.0

La funzione correlata all'evento assegnato al bottone è l'onClickPlaylist:

```
function onClickPlaylist(event){  
  fetch('https://api.spotify.com/v1/playlists/37i9dQZF1DXdcBWuJkbcy?si=231f68f805854e10',{  
    headers:{  
      'Authorization':'Bearer '+ token.access_token  
    }  
  }).then(onResponseSpotify).then(onSpotify);  
}
```

Questa funzione semplicemente fa la richiesta attraverso una fetch con parametro passato l'url completo (Endpoint + id playlist). Nell'headers correlato alla richiesta passiamo il token ricevuto in precedenza.

```
function onToken(json){  
  token = json;  
}  
  
function onResponseToken(response){  
  if(response.status=== 401){  
    console.log('ERRORE AUTORIZZAZIONE SPOTIFY');  
  }  
  return response.json();  
}  
  
function onResponseSpotify(response){  
  if(response.status === 400){  
    console.log('ERRORE RICHIESTA SPOTIFY');  
  }  
  return response.json();  
}
```


API REST Oauth 2.0


La funzione onSpotify ricava le informazioni del json di risposta, assegna le informazioni alle varie variabili, nasconde il testo e il bottone e per finire aggiunge il tutto all'HTML

```
function onSpotify(json){  
  console.log(json);  
  //Hidden Elementi Precedenti  
  const bottone = document.querySelector('#clickSpoty');  
  bottone.classList.add('hidden');  
  const title = document.querySelector('#SpotyTitle');  
  title.classList.add('hidden');  
  //Inserimento Dati  
  let result= document.querySelector('.random');  
  let text = document.createElement('h4');  
  text.textContent=json.name;  
  result.appendChild(text);  
  let img = document.createElement('img');  
  img.src=json.images[0].url;  
  img.classList.add('flex');  
  result.appendChild(img);  
  let par = document.createElement('p');  
  par.textContent=json.description;  
  result.appendChild(par);  
  let link = document.createElement('a');  
  link.textContent=json.href;  
  result.appendChild(link);  
}
```


API REST Oauth 2.0

Ottenendo come risultato finale questo:

Motivation Mix

The image shows the cover art for the 'Motivation Mix' playlist on Spotify. It features a man in black athletic wear jumping in the air against a light orange background. A small Spotify logo is in the top left corner of the image. The text 'Motivation Mix' is at the bottom of the image.

Motivation Mix

Uplifting and energetic music that helps you stay motivated.

<https://api.spotify.com/v1/playlists/37igdqZFtDXdxcBWuJkbcy>