

JavaScript – Programmation avancée

Intervenant : Jean-Frédéric VINCENT





Module 9

Interroger une API avec Fetch

09 Interroger une API avec fetch

Les fonctions asynchrones

Les promesses

Les littéraux de gabarits

Mise en place de Fetch

Les fonctions asynchrones

- Une fonction synchrone **async** peut contenir une expression **await** qui interrompt l'exécution de la fonction asynchrone et attend la résolution de la promesse passée Promise.
La fonction asynchrone reprend ensuite puis renvoie la valeur de résolution.
- Le mot-clé **await** est uniquement valide au sein des fonctions **asynchrones**.

```
const go async ()=>{  
  const response = await fetch("https://swapi.dev/api/people/1");  
  // on récupère la partie JSON de la response  
  const data = await response.json();  
  console.log(data);  
}  
go();
```

*Dès que ma fonction contient **await**, elle doit être prefixer par **async***

Les promesses

- Une Promise est un objet qui va prendre un certain temps pour se résoudre ou rejeter.

```
const repondApresDeuxSeconde= () => {  
  return new Promise((resolve) => {  
    setTimeout(() => {  
      resolve('ok');  
    }, 2000);  
  });  
}  
  
const go= async()=> {  
  const result = await repondApresDeuxSeconde();  
  console.log(result); // affiche : ok  
}  
  
go();
```

Les littéraux de gabarits

- Les littéraux de gabarits **Template literals** sont une alternative intéressante à la concaténation
 - Le "back quote" : ALT GR + 7 (2 fois)

```
const age =18;  
console.log (`Votre age ${age}`);  
// avec un if ternaire  
console.log (`Votre age ${age} statut :${age >= 18? 'ok':'interdit'}`);
```

Les littéraux de gabarits

- Les littéraux de gabarits **Template literals** sont une alternative intéressante à la concaténation

Le back quote : ALT GR + 7 (2 fois)

```
const age =18;
console.log (`Votre age ${age}`);
// avec un if ternaire
console.log (`Votre age ${age} statut :${age >= 18? 'ok':'interdit'}`);
```

Utilisation de Fetch

- Mise en place d'une requête AJAX avec Fetch
- Pour récupérer le contenu JSON de la réponse on utilise **.json()**
- **Attention** de pas oublié le **await** devant !

```
const response = await fetch("https://swapi.dev/api/people/1");  
// on récupère la partie JSON de la response  
const data = await response.json();  
console.log(data);
```