Fetch y consumo de APIs

La API Fetch proporciona una interfaz JavaScript para acceder y manipular partes del canal HTTP, como peticiones y respuestas. Aquí hay una explicación y ejemplos básicos:

1. Haciendo una solicitud GET:

```
// Haciendo una solicitud GET a una API
fetch('https://jsonplaceholder.typicode.com/todos/1')
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```

2. Enviando datos con una solicitud POST:

```
// Haciendo una solicitud POST a una API
fetch('https://jsonplaceholder.typicode.com/posts', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
  },
  body: JSON.stringify({
    title: 'foo',
    body: 'bar',
    userId: 1,
```

```
  }),
})
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```

3. Uso de Async/Await con Fetch:

```
// Función que utiliza async/await con Fetch
async function obtenerDatos() {
  try {
    const response = await
fetch('https://jsonplaceholder.typicode.com/todos/1');
    const data = await response.json();
    console.log(data);
  } catch (error) {
    console.error('Error:', error);
  }
}

// Llamada a la función
obtenerDatos();
```

4. Trabajando con APIs autenticadas:

```
// Ejemplo de solicitud a una API con autenticación (usando un token)
const token = 'tu_token';
fetch('https://api.example.com/data', {
```

```
  headers: {
    'Authorization': `Bearer ${token}`,
  },
})
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```

Estos ejemplos muestran cómo usar Fetch para realizar solicitudes HTTP, cómo manejar respuestas JSON y cómo trabajar con promesas. Puedes adaptarlos según las necesidades específicas de la API que estés utilizando.