**GET Request**

**Purpose**: Retrieve data from the server.

 A GET request is used to request data without modifying any resources on the server.

 The response.json() method converts the response body into a JavaScript object or array.

**Syntax**

**fetch('https://api.example.com/data')**

**.then(response => {**

**if (!response.ok) {**

**throw new Error('Network response was not ok');**

**}**

**return response.json();**

**})**

**.then(data => {**

**console.log(data);**

**})**

**.catch(error => {**

**console.error('Fetch error:', error);**

**});**

## POST Request

**Purpose**: Send new data to the server.

* A POST request is typically used to create a new resource on the server.
* The body property contains the data being sent to the server, which is usually in JSON format.

Syntax

**const data = { key: 'value' };**

**fetch('https://api.example.com/data', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json',**

**},**

**body: JSON.stringify(data),**

**})**

**.then(response => {**

**if (!response.ok) {**

**throw new Error('Network response was not ok');**

**}**

**return response.json();**

**})**

**.then(data => {**

**console.log('Success:', data);**

**})**

**.catch(error => {**

**console.error('Error:', error);**

**});**

## PUT Request

**Purpose**: Update existing data on the server completely.

 A PUT request is used to replace the entire resource with the provided data.

 This method is idempotent; multiple identical requests will produce the same result.

### Syntax

### const updatedData = { key: 'newValue' };

### fetch('https://api.example.com/data/1', {

### method: 'PUT',

### headers: {

### 'Content-Type': 'application/json',

### },

### body: JSON.stringify(updatedData),

### })

### .then(response => {

### if (!response.ok) {

### throw new Error('Network response was not ok');

### }

### return response.json();

### })

### .then(data => {

### console.log('Updated:', data);

### })

### .catch(error => {

### console.error('Error:', error);

### });

## PATCH Request

**Purpose**: Partially update existing data on the server.

 A PATCH request is used to apply partial modifications to a resource.

 It only requires the changes to be sent, not the entire resource.

### Syntax

### const partialUpdate = { key: 'partiallyUpdatedValue' };

### fetch('https://api.example.com/data/1', {

### method: 'PATCH',

### headers: {

### 'Content-Type': 'application/json',

### },

### body: JSON.stringify(partialUpdate), })

### .then(response => {

### if (!response.ok) {

### throw new Error('Network response was not ok');

### }

### return response.json();

### })

### .then(data => {

### console.log('Partially Updated:', data)

### })

### .catch(error => {

### console.error('Error:', error);

### });

## DELETE Request

**Purpose**: Remove data from the server.

 A DELETE request is used to remove a specified resource from the server.

 Typically, it does not require a request body, although it can depending on server configuration.

### Syntax

### fetch('https://api.example.com/data/1', {

### method: 'DELETE',

### })

### .then(response => {

### if (!response.ok) {

### throw new Error('Network response was not ok');

### }

### console.log('Deleted successfully');

### })

### .catch(error => {

### console.error('Error:', error);

### });