

Response API

□□

- [□□□□](#)
- [□□□□](#)
 - [body□ bodyUsed](#)
 - [headers](#)
 - [ok](#)
 - [redirected](#)
 - [status□ statusText](#)
 - [type](#)
 - [url](#)
- [□□□□](#)
 - [□□□□](#)
 - [clone\(\)](#)
- [□□□□](#)
 - [Response.json\(\)](#)
 - [Response.error\(\)](#)
 - [Response.redirect\(\)](#)

□□□□□□

Response () □□□□□□□□□□

fetch () □□□□□□□□

Response □□□

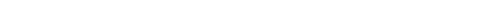
□□□□□

Response () □□□□□□□□□□

Response □□□

```
// □□  
new Response([body:Object, [init : Object]]): Response  
  
// □□
```

```
new Response()  
new Response(body)  
new Response(body, options)
```



`[]` `body []` `ArrayBuffer[]` `ArrayBufferView[]` `Blob[]`
`FormData[]` `ReadableStream[]` `String[]` `URLSearchParams[]`

```
{
  status: Number
  statusText: String
  headers: Object
}
```

[illegible]

```
const myBlob = new Blob();
const myOptions = { status: 200, statusText: "OK" };
const myResponse = new Response(myBlob, myOptions);
```


 JSON
 

```
const data = {
    hello: "world",
};

const json = JSON.stringify(data, null, 2);

const result = new Response(json, {
    headers: {
        "content-type": "application/json;charset=UTF-8",
    },
});
```

[illegible]



body bodyUsed

body  ReadableStream 

```
const res = await fetch('/fireworks.ogv');
const reader = res.body.getReader();

let result;
while (!(result = await reader.read()).done) {
  console.log('chunk size:', result.value.byteLength);
}
```

 body 

 body  Stream 

bodyUsed  body 

headers

headers  Headers 

```
const res = await fetch('/flowers.jpg');
console.log(...res.headers);
```

 res.headers 

ok

ok  200  299 

```
const res1 = await fetch('https://httpbin.org/status/200');
console.log(res1.ok); // true

const res2 = await fetch('https://httpbin.org/status/404');
console.log(res2.ok); // false
```

redirected

redirected

301

302

```
const res1 = await fetch('https://httpbin.org/status/200');
console.log(res1.redirected); // false

const res2 = await fetch('https://httpbin.org/status/301');
console.log(res2.redirected); // true
```

status statusText

status

```
const res1 = await fetch('https://httpbin.org/status/200');
console.log(res1.status); // 200

const res2 = await fetch('https://httpbin.org/status/404');
console.log(res2.status); // 404
```

statusText

200

statusText

OK

type

type

basic

cors

default

error

opaque

opaqueredirect

url

url

URL



- `.json()`  Promise  JSON 
- `.text()`  Promise 
- `.blob()`  Promise  Blob 
- `.arrayBuffer()`  Promise  ArrayBuffer 
- `.formData()`  Promise  FormData 

```
async function getRedditPosts() {
  try {
    const response = await fetch('https://www.reddit.com/r/all/top.json?limit=10');
    const data = await response.json();
    const posts = data.data.children.map(child => child.data);
    console.log(posts.map(post => post.title));
  } catch (error) {
    console.error(error);
  }
}
```

```
async function displayImageAsync() {
  try {
    const response = await fetch('https://www.example.com/image.jpg');
    const blob = await response.blob();
    const url = URL.createObjectURL(blob);
    const img = document.createElement('img');
    img.src = url;
    document.body.appendChild(img);
  } catch (error) {
    console.error(error);
  }
}
```

```
async function playAudioAsync() {
  try {
```

```

const response = await fetch('https://www.example.com/audio.mp3');
const arrayBuffer = await response.arrayBuffer();
const audioBuffer = await new AudioContext().decodeAudioData(arrayBuffer);
const source = new AudioBufferSourceNode(new AudioContext(), { buffer: audioBuffer });
source.connect(new AudioContext().destination);
source.start(0);
} catch (error) {
  console.error(error);
}
}

```

clone()

clone() 

Response 

```

const res1 = await fetch('/flowers.jpg');
const res2 = res1.clone();

```





Response.json()

Response.json() 

Response 

JSON 

Content-Type 

application/json 

```

Response.json(data)
Response.json(data, options)

```

Response.json() 

Response() 



```

const jsonResponse1 = Response.json({ my: "data" });

const jsonResponse2 = Response.json(

```

```
{ some: "data", more: "information" },
{ status: 307, statusText: "Temporary Redirect" },
);
```

Response.error()

Response.error()

Service worker

```
self.addEventListener('fetch', (event) => {
  const url = new URL(event.request.url);
  if (url.pathname === '/flowers.jpg') {
    event.respondWith(Response.error());
  }
});
```

Response.redirect()

Response.redirect()

Service worker

```
Response.redirect(url)
Response.redirect(url, status)
```

url

301 302

```
Response.redirect("https://www.example.com", 302);
```

#2

2023-09-09 19:46:22 UTC

2023-09-25 23:10:14 UTC