

# Request API

□□

- [□□□□](#)
- [□□□□](#)
- [□□□□](#)
  - [□□□□□□□□](#)
- [clone\(\)](#)

□□□□□□      Request() □□□□□□□□□□□□      HTTP □□□□□□      Response  
□□□□□□      fetch() □□□□

□□□□□□      Request □□□□□□□□□□□□□□□□      HTTP □□□□□□□□  
fetch() □□□□□□□□□□

□□□□□□

Request □□□□□□□□□□□□      Request □□□□□

```
new Request(url: String, [init: Object]): Request
```

□□□□□□□□      URL □□□□□□□□□□□□□□□□□□      HTTP  
□□□□□□□□□□

```
{
  body: Object
  cache: String
  credentials: String
  headers: Object
  integrity: String
  keepalive: Boolean
  method: String
  mode: String
  redirect: □String
```

```

referrer: String
referrerPolicy: String
requestMode: String
requestCredentials: String
signal: AbortSignal
}

```

RequestInit

- body: HTTP RequestBody (Blob, BufferSource, FormData, String, URLSearchParams)
- cache: RequestCache
- credentials: RequestCredentials, omit: boolean, same-origin: boolean, include: boolean, Chrome 47
- headers: HeadersInit, HTTP Headers
- integrity: string, sha256-BpfBw7ivV8q2jLiT13fxDYAe2tJllusRSZ273h2nFSE=
- method: HTTP Method (GET, POST, DELETE, etc.)
- mode: RequestMode (cors, no-cors, navigate)
- redirect: RequestRedirect (error, follow, manual, follow)
- referrer: string, about:client

Request

```

// Create a new Request
const request = new Request('flowers.jpg');

// Create a new Request with options
const myInit = {
  method: "GET",
  headers: {
    "Content-Type": "image/jpeg",
  },
  mode: "cors",
  cache: "default",
};

const request = new Request('flowers.jpg', myInit);

```

Request() constructor

Request constructor

Request

```
new Request(request: Request, [init: Object]): Request
```



Request

1

HTTP  ReadableStream  null

```
const request = new Request('/myEndpoint', {  
  method: "POST",  
  body: "Hello world",  
});  
  
request.body; // ReadableStream 
```

Firefox

2

3

default  force-cache  no-cache  no-store  only-if-cached  reload

4

cookie  omit   
 include  same-origin  cookie

5

"    
'document'  'embed'  'font'  'frame'  'iframe'  'image'  'manifest'  'object'  'paintworklet'   
'report'  'script'  'sharedworker'  'style'  'track'  'video'  'worker'  'xslt'

6

Headers

7 integrity

integrity

8 method

method

GET POST

9 mode

mode

same-origin no-cors

cors

10 redirect

redirect

follow error manual

11 referrer

referrer

URL

12 referrerPolicy

referrerPolicy

referrer

13 signal

signal

AbortSignal

14 url

url

```
const myRequest = new Request('flowers.jpg');
const myURL = myRequest.url;
```



- arrayBuffer() Promise Request ArrayBuffer
- blob() Promise Request Blob
- json() Promise Request JSON

- `text()` `Promise` `Request` `FormData`
- `formData()` `Promise` `Request` `FormData`

`json()`

```
const obj = { hello: "world" };

const request = new Request("/myEndpoint", {
  method: "POST",
  body: JSON.stringify(obj),
});

request.json().then((data) => {
  // JSON
});
```

`.formData()` `Promise` `FormData`

`Service Worker`

```
self.addEventListener('fetch', event => {
  // 
  if (
    event.request.method === 'POST' &&
    event.request.headers.get('Content-Type') === 'application/x-www-form-urlencoded'
  ) {
    event.respondWith(handleFormSubmission(event.request));
  }
});

async function handleFormSubmission(request) {
  const formData = await request.formData();
  formData.append('extra-field', 'extra-value');

  const newRequest = new Request(request.url, {
    method: request.method,
    headers: request.headers,
    body: new URLSearchParams(formData)
  });

  return fetch(newRequest);
}
```

```
}
```

```
const
```

Service Worker

```
const
```

```
fetch()
```

```
const
```

# clone()

```
clone()
```

```
const
```

HTTP

```
const
```

```
const myRequest = new Request('flowers.jpg');
```

```
const newRequest = myRequest.clone();
```

```
const #2
```

```
const 2023-09-09 19:46:22 UTC
```

```
const 2023-09-09 23:54:09 UTC
```