SubtleCrypto: digest() method

Secure context: This feature is available only in secure contexts (HTTPS), in some or all supporting browsers.

The `digest()` method of the `SubtleCrypto` interface generates a digest of the given data. A digest is a short fixed-length value derived from some variable-length input. Cryptographic digests should exhibit collision-resistance, meaning that it's hard to come up with two different inputs that have the same digest value.

It takes as its arguments an identifier for the digest algorithm to use and the data to digest. It returns a `Promise` which will be fulfilled with the digest.

Note that this API does not support streaming input: you must read the entire input into memory before passing it into the digest function.

Syntax

```
digest(algorithm, data)
```

Parameters

`algorithm`

This may be a string or an object with a single property `name` that is a string. The string names the hash function to use. Supported values are:

- `"SHA-1"` (but don't use this in cryptographic applications)
- `"SHA-256"`
- `"SHA-384"`
- `"SHA-512"`.

`data`

An `ArrayBuffer`, a `TypedArray` or a `DataView` object containing the data to be digested.

Return value

A `Promise` that fulfills with an `ArrayBuffer` containing the digest.

Supported algorithms

Digest algorithms, also known as cryptographic hash functions, transform an arbitrarily large block of data into a fixed-size output, usually much shorter than the input. They have a variety of applications in cryptography.

Algorithm	Output length (bits)	Block size (bits)	Specification
SHA-1	160	512	FIPS 180-4, section 6.1
SHA-256	256	512	FIPS 180-4, section 6.2
SHA-384	384	1024	FIPS 180-4, section 6.5
SHA-512	512	1024	FIPS 180-4, section 6.4

Warning: SHA-1 is now considered vulnerable and should not be used for cryptographic applications.

Note: If you are looking here for how to create a keyed-hash message authentication code (HMAC), you need to use the SubtleCrypto.sign() instead.

Examples

For more examples of using the `digest()` API, see Non-cryptographic uses of SubtleCrypto.

Basic example

This example encodes a message, then calculates its SHA-256 digest and logs the digest length:

```
const text =
  "An obscure body in the S-K System, your majesty. The inhabitants refer to it as the planet
Earth.";

async function digestMessage(message) {
  const encoder = new TextEncoder();
  const data = encoder.encode(message);
  const hash = await crypto.subtle.digest("SHA-256", data);
  return hash;
}

digestMessage(text).then((digestBuffer) =>
  console.log(digestBuffer.byteLength)
);
```

Converting a digest to a hex string

The digest is returned as an `ArrayBuffer`, but for comparison and display digests are often represented as hex strings. This example calculates a digest, then converts the `ArrayBuffer` to a hex string:

```
const text =
  "An obscure body in the S-K System, your majesty. The inhabitants refer to it as the planet
Earth.";

async function digestMessage(message) {
  const msgUint8 = new TextEncoder().encode(message); // encode as (utf-8) Uint8Array
  const hashBuffer = await crypto.subtle.digest("SHA-256", msgUint8); // hash the message
  const hashArray = Array.from(new Uint8Array(hashBuffer)); // convert buffer to byte array
  const hashHex = hashArray
    .map((b) => b.toString(16).padStart(2, "0"))
    .join(""); // convert bytes to hex string
  return hashHex;
}

digestMessage(text).then((digestHex) => console.log(digestHex));
```

Specifications

Specification
Web Cryptography API # SubtleCrypto-method-digest

Browser compatibility

Report problems with this compatibility data on GitHub

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android	Deno	Node.js
digest	✓ 37	✓ 79 …	✓ 34	✓ 24	✓ 7	✓ 37	✓ 34	✓ 24	✓ 7	✓ 3.0	✓ 37	✓ 1.11	✓ 15.0.0

Tip: you can click/tap on a cell for more information.

✓ Full support Partial support … Has more compatibility info.

See also

- Non-cryptographic uses of SubtleCrypto
- Chromium secure origins specification
- FIPS 180-4 specifies the SHA family of digest algorithms.