Content-Security-Policy: default-src directive

- Syntax
- Examples
  - No inheritance with default-src
  - Firefox default-src: none SVG sprite blocking issue
- Specifications
- Browser compatibility
- See also

The HTTP Content-Security-Policy (CSP) `default-src` directive serves as a fallback for the other CSP fetch directives. For each of the following directives that are absent, the user agent looks for the `default-src` directive and uses this value for it:

- child-src
- connect-src
- font-src
- frame-src
- img-src
- manifest-src
- media-src
- object-src
- prefetch-src
- script-src
- script-src-elem
- script-src-attr
- style-src
- style-src-elem

- style-src-attr
- worker-src

CSP version	1
Directive type	[Fetch directive](https://developer.mozilla.org/en-US/docs/Glossary/Fetch%20directive)

Syntax

```
Content-Security-Policy: default-src 'none';
Content-Security-Policy: default-src <source-expression-list>;
```

This directive may have one of the following values:

- `'none'`
  - : No resources may be loaded. The single quotes are mandatory.
- `<source-expression-list>`
  - : A space-separated list of *source expression* values. Resources may be loaded if they match any of the given source expressions. For this directive, any of the source expression values listed in Fetch directive syntax are applicable.

Examples

No inheritance with default-src

If there are other directives specified, `default-src` does not influence them. The following header:

```
Content-Security-Policy: default-src 'self'; script-src https://example.com
```

is the same as:

```
Content-Security-Policy: connect-src 'self';
                         font-src 'self';
                         frame-src 'self';
                         img-src 'self';
                         manifest-src 'self';
```

```
                    media-src 'self';
                    object-src 'self';
                    script-src https://example.com;
                    style-src 'self';
                    worker-src 'self'
```

Firefox `default-src: none` SVG sprite blocking issue

> **NOTE:** This issue was fixed in Firefox 132; see bug 1773976.

When creating a CSP, you can start with `default-src 'none'` to lock down all resource loading and then add further directives to open up the policy, allowing you to load just the resources you need. For example, to allow same-origin loading of images only:

```
Content-Security-Policy: default-src 'none'; img-src 'self'
```

However, there is a problem here. If you are embedding SVG sprites defined in external files via the `<use>` element, for example:

```
<svg>
  <use href="/images/icons.svg#icon"/>
</svg>
```

your SVG images will be blocked in Firefox if you have a `default-src 'none'` policy set. Firefox does not treat the SVG as an embedded image like other browsers do, therefore `img-src 'self'` will not allow them to be loaded. You need to use `default-src 'self'` if you want your external sprites to load in Firefox.

Alternatively, if the `default-src 'none'` policy is a hard requirement, you can include the SVG sprites inline in the HTML page:

```
<body>
  <svg style="display: none">
    <symbol id="icon" viewBox="0 0 24 24">
      <path d="…" />
    </symbol>
```

```
  </svg>
  …
  <svg>
    <use href="#icon" />
  </svg>
</body>
```

Specifications

Browser compatibility

See also

- Content-Security-Policy
- CSP directives (https://w3c.github.io/webappsec-csp/#csp-directives):
  - Fetch directive
  - Document directive
  - Navigation directive
  - Reporting directive
  - `upgrade-insecure-requests`
  - `block-all-mixed-content`

□□□ #3
□ □□□ □□□ 2026-10-31 03:27:53 UTC
□ cocobook-sync □□□ 2026-11-04 17:35:40 UTC