

This example demonstrates how to build a library with webpack that has dependencies on other libraries which should not be included in the compiled version.

We use the `libraryTarget: "umd"` option to build a UMD module that is consumable in CommonJS, AMD and with script tags. We don't specify the `library` option so the library is exported to the root namespace.

We use the `externals` option to define dependencies that should be resolved in the target environment.

In the simple case we just need to specify a string ( `"add"` ). Then it's resolved as `"add"` module in CommonJS and AMD, and as global `add` when used with the script tag.

In the complex case we specify different values for each environment:

environment	config value	resolved as
CommonJS (strict)	<code>["./math", "subtract"]</code>	<code>require("./math").subtract</code>
CommonJS (node.js)	<code>"./subtract"</code>	<code>require("./subtract")</code>
AMD	<code>"subtract"</code>	<code>define(["subtract"], ...)</code>
script tag	<code>"subtract"</code>	<code>this.subtract</code>

## example.js

```
_{{example.js}}_
```

## webpack.config.js

```
_{{webpack.config.js}}_
```

## dist/output.js

```
_{{dist/output.js}}_
```

## Info

### Unoptimized

```
_{{stdout}}_
```

### Production mode

```
_({production:stdout})_
```