This example demonstrates how to build a complex library with webpack. The library consists of multiple parts that are usable on its own and together.

When using this library with script tags it exports itself to the namespace `MyLibrary` and each part to a property in this namespace (`MyLibrary.alpha` and `MyLibrary.beta`). When consuming the library with CommonsJS or AMD it just exports each part.

We are using multiple entry points (`entry` option) to build every part of the library as a separate output file. The `output.filename` option contains `[name]` to give each output file a different name.

We are using the `libraryTarget` option to generate a UMD (Universal Module Definition) module that is consumable in CommonsJS, AMD and with script tags. The `library` option defines the namespace. We are using `[name]` in the `library` option to give every entry a different namespace.

You can see that webpack automatically wraps your module so that it is consumable in every environment. All you need is this simple config.

Note: You can also use the `library` and `libraryTarget` options without multiple entry points. Then you don't need `[name]`.

Note: When your library has dependencies that should not be included in the compiled version, you can use the `externals` option. See externals example.

webpack.config.js

```
var path = require("path");
module.exports = {
    // mode: "development || "production",
    entry: {
        alpha: "./alpha",
        beta: "./beta"
    },
    output: {
        path: path.join(__dirname, "dist"),
        filename: "MyLibrary.[name].js",
        library: ["MyLibrary", "[name]"],
        libraryTarget: "umd"
    }
};
```

dist/MyLibrary.alpha.js

```
(function webpackUniversalModuleDefinition(root, factory) {
    if(typeof exports === 'object' && typeof module === 'object')
```

```
        module.exports = factory();
    else if(typeof define === 'function' && define.amd)
        define([], factory);
    else if(typeof exports === 'object')
        exports["MyLibrary"] = factory();
    else
        root["MyLibrary"] = root["MyLibrary"] || {}, root["MyLibrary"]["alpha"] = factory();
})(self, function() {
return /******/ (() => { // webpackBootstrap
/******/     var __webpack_modules__ = ([
/* 0 */
/*!******************!*\
  !*** ./alpha.js ***!
  \******************/
/*! unknown exports (runtime-defined) */
/*! runtime requirements: module */
/*! CommonJS bailout: module.exports is used directly at 1:0-14 */
/***/ ((module) => {

module.exports = "alpha";

/***/ })
/******/     ]);
```

/* webpack runtime code */

```
/************************************************************************/
/******/     // The module cache
/******/     var __webpack_module_cache__ = {};
/******/
/******/     // The require function
/******/     function __webpack_require__(moduleId) {
/******/         // Check if module is in cache
/******/         var cachedModule = __webpack_module_cache__[moduleId];
/******/         if (cachedModule !== undefined) {
/******/             return cachedModule.exports;
/******/         }
/******/         // Create a new module (and put it into the cache)
/******/         var module = __webpack_module_cache__[moduleId] = {
/******/             // no module.id needed
/******/             // no module.loaded needed
/******/             exports: {}
/******/         };
/******/
/******/         // Execute the module function
/******/         __webpack_modules__[moduleId](module, module.exports, __webpack_require__);
/******/
```

```
/******/         // Return the exports of the module
/******/         return module.exports;
/******/     }
/******/
/************************************************************************/

/******/
/******/     // startup
/******/     // Load entry module and return exports
/******/     // This entry module is referenced by other modules so it can't be inlined
/******/     var __webpack_exports__ = __webpack_require__(0);
/******/
/******/     return __webpack_exports__;
/******/ })()
;
});
```

dist/MyLibrary.beta.js

```
(function webpackUniversalModuleDefinition(root, factory) {
    if(typeof exports === 'object' && typeof module === 'object')
        module.exports = factory();
    else if(typeof define === 'function' && define.amd)
        define([], factory);
    else if(typeof exports === 'object')
        exports["MyLibrary"] = factory();
    else
        root["MyLibrary"] = root["MyLibrary"] || {}, root["MyLibrary"]["beta"] = factory();
})(self, function() {
return /******/ (() => { // webpackBootstrap
/******/     var __webpack_modules__ = ([
/* 0 */,
/* 1 */
/*!******************!*\
  !*** ./beta.js ***!
  \******************/
/*! unknown exports (runtime-defined) */
/*! runtime requirements: module */
/*! CommonJS bailout: module.exports is used directly at 1:0-14 */
/***/ ((module) => {

module.exports = "beta";

/***/ })
/******/     ]);
```

/* webpack runtime code */

```
/************************************************************************/
/******/     // The module cache
/******/     var __webpack_module_cache__ = {};
/******/
/******/     // The require function
/******/     function __webpack_require__(moduleId) {
/******/         // Check if module is in cache
/******/         var cachedModule = __webpack_module_cache__[moduleId];
/******/         if (cachedModule !== undefined) {
/******/             return cachedModule.exports;
/******/         }
/******/         // Create a new module (and put it into the cache)
/******/         var module = __webpack_module_cache__[moduleId] = {
/******/             // no module.id needed
/******/             // no module.loaded needed
/******/             exports: {}
/******/         };
/******/
/******/         // Execute the module function
/******/         __webpack_modules__[moduleId](module, module.exports, __webpack_require__);
/******/
/******/         // Return the exports of the module
/******/         return module.exports;
/******/     }
/******/
/************************************************************************/

/******/
/******/     // startup
/******/     // Load entry module and return exports
/******/     // This entry module is referenced by other modules so it can't be inlined
/******/     var __webpack_exports__ = __webpack_require__(1);
/******/
/******/     return __webpack_exports__;
/******/ })()
;
});
```

Info

Unoptimized

```
asset MyLibrary.beta.js 2.07 KiB [emitted] (name: beta)
asset MyLibrary.alpha.js 2.06 KiB [emitted] (name: alpha)
```

```
chunk (runtime: alpha) MyLibrary.alpha.js (alpha) 25 bytes [entry] [rendered]
  > ./alpha alpha
  ./alpha.js 25 bytes [built] [code generated]
    [used exports unknown]
    cjs self exports reference ./alpha.js 1:0-14
    entry ./alpha alpha
    used as library export
chunk (runtime: beta) MyLibrary.beta.js (beta) 24 bytes [entry] [rendered]
  > ./beta beta
  ./beta.js 24 bytes [built] [code generated]
    [used exports unknown]
    cjs self exports reference ./beta.js 1:0-14
    entry ./beta beta
    used as library export
webpack 5.51.1 compiled successfully
```

Production mode

```
asset MyLibrary.alpha.js 429 bytes [emitted] [minimized] (name: alpha)
asset MyLibrary.beta.js 425 bytes [emitted] [minimized] (name: beta)
chunk (runtime: alpha) MyLibrary.alpha.js (alpha) 25 bytes [entry] [rendered]
  > ./alpha alpha
  ./alpha.js 25 bytes [built] [code generated]
    [used exports unknown]
    cjs self exports reference ./alpha.js 1:0-14
    entry ./alpha alpha
    used as library export
chunk (runtime: beta) MyLibrary.beta.js (beta) 24 bytes [entry] [rendered]
  > ./beta beta
  ./beta.js 24 bytes [built] [code generated]
    [used exports unknown]
    cjs self exports reference ./beta.js 1:0-14
    entry ./beta beta
    used as library export
webpack 5.51.1 compiled successfully
```