

This example illustrates how common modules from deep ancestors of an entry point can be split into a separate common chunk

- `pageA` and `pageB` are dynamically required
- `pageC` and `pageA` both require the `reusableComponent`
- `pageB` dynamically requires `PageC`

You can see that webpack outputs five files/chunks:

- `output.js` is the entry chunk and contains
  - the module system
  - chunk loading logic
  - the entry point `example.js`
- `0.output.js` is an additional chunk
  - module `reusableComponent`
- `1.output.js` is an additional chunk
  - module `pageB`
- `2.output.js` is an additional chunk
  - module `pageA`
- `3.output.js` is an additional chunk
  - module `pageC`

## example.js

```
_({{example.js}})_
```

## pageA.js

```
_({{pageA.js}})_
```

## pageB.js

```
_({{pageB.js}})_
```

## pageC.js

```
_({{pageC.js}})_
```

## reusableComponent.js

```
_{{reusableComponent.js}}_
```

## webpack.config.js

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

## dist/output.js

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

## dist/366.output.js

```
_{{dist/366.output.js}}_
```

## dist/588.output.js

```
_{{dist/588.output.js}}_
```

## dist/145.output.js

```
_{{dist/145.output.js}}_
```

## dist/421.output.js

```
_{{dist/421.output.js}}_
```

## Info

### Unoptimized

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

### Production mode

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