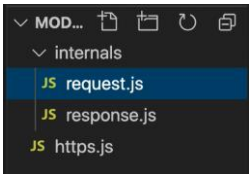
Module System – El archivo index.js

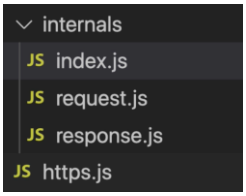
Al trabajar con proyectos Node, es posible que se encuentre con un archivo llamado *index.js*. Para explicar su propósito, vamos a crear una carpeta llamada *internals* en nuestro proyecto de ejemplo y donde reubicaremos los módulos *request* y *response*.



¿Qué pasaríamos si intentáramos hacer un *require* a la carpeta internals?

```
https.js > ...
1  const internals = require('./internal');
2  const { send } = require('./internals/request');
3  const { read } = require('./internals/response');
```

Tal como está, esto no funcionaría, ya que *internals* no es un módulo que la función *require* pueda cargar, pero podemos hacer que funcione, y la forma de hacerlo es usando el archivo *index.js*. Así que creamos un archivo llamado *index.js* dentro de la carpeta *internals*:



Index.js es un caso especial en Node. Te permite tratar una carpeta como un módulo y exportará como hemos estado haciendo un objeto que contiene los datos que queremos exportar de la capeta internals:

```
index.js > <unknown>
module.exports = {
  request: require('./request'),
  response: require('./response'),
};
```

De esta forma podemos exponer los módulos que están dentro de la carpeta *internals*, tratando a la misma carpeta como un módulo:

```
https.js > makeRequest
1  const internals = require('./internals');
2  // const { send } = require('./internals/
3  // const { read } = require('./internals/
4
5  function makeRequest(url, data) {
6    internals.request.send(url, data);
7    return internals.response.read();
8  }
```

Index.js es un caso especial que le permite exportar funciones de muchos módulos diferentes que viven en una sola carpeta hacia un único punto, haciendo un require a esa carpeta, o a la ruta a esa carpeta. De este modo.