Carga de contenido estático

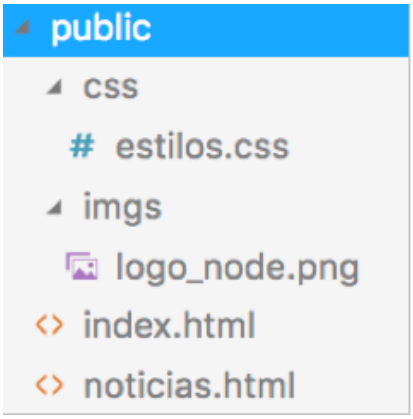

En sessions anteriors hem vist com estructurar una aplicació express per a desenvolupar un proveïdor de serveis REST. Hem emmagatzemat els esquemes i models de dades en una carpeta `models`, i els encaminadors que donen resposta a les diferents URLS en una carpeta `routes`, deixant l'aplicació principal en l'arrel del projecte.

Aquesta estructura és la que normalment se segueix per a desenvolupar una API REST (recorda, Express és un framework *unopinionated*, cosa que significa que podem triar qualsevol altra estructura que ens semble adequada). No obstant això, en aquest puzle falten algunes peces per a poder desenvolupar aplicacions web més completes, que permeten servir contingut HTML, més enllà de les simples dades JSON. Per a començar, veurem en aquest document com incorporar contingut estàtic (pàgines HTML, estils CSS i arxius JavaScript), i com estructurar-los en la nostra aplicació Express.

Veurem que amb Express la gestió del contingut estàtic és molt senzilla, a través del corresponent *middleware* que incorporarem als nostres projectes que el requerisquen. Per a això, farem una prova en un projecte anomenat "*ExpressEstatic*" en la nostra carpeta de "*ProjectesNode/Proves*".

1. Ubicació del contingut estàtic

A l'hora de situar el contingut estàtic de la nostra aplicació, és habitual deixar-lo tot en una única subcarpeta, que típicament se sol cridar "public". Imaginem que aquesta subcarpeta té aquesta estructura:



La imatge "logo_node.png" pot ser qualsevol que vulgueu. En aquest exemple usarem un logo de Node buscat en Internet:

La fulla d'estils té alguns estils bàsics (tipus de lletra i color de fons):

```
bodi
{
    background-color:rgb(245, 244, 201);
    font-family: Arial;
}
```

La pàgina `index.html` és molt simple, incorporant la fulla d'estils, la imatge, un encapçalat de primer nivell i un enllaç a l'altra pàgina:

```
<!doctype html>
<html>
    <head>
        <link href="css/estils.css" rel="stylesheet">
    </head>
    <body>
        <div align="center">
            <img src="imgs/logo_node.png">
        </div>
        <h1>Benvingut a Node.js</h1>
        <p><a href="noticies.html">Notícies de node</a></p>
    </body>
</html>
```

La pàgina de `noticies.html` és similar a l'anterior, mostrant un enllaç a la pàgina principal i un llistat d'exemple:

```
<!doctype html>
<html>
    <head>
        <link href="css/estils.css" rel="stylesheet">
    </head>
    <body>
        <div align="center">
            <img src="imgs/logo_node.png">
        </div>
        <h1>Benvingut a Node.js</h1>
        <p><a href="index.html">Pàgina d'inici</a></p>
        <ul>
            <li>Concurs JavaScript per a addictes a Node.js</li>
            <li>LinkedIn migra des de Rails cap a Node</li>
            <li>Conferència Node.js a Itàlia</li>
        </ul>
    </body>
</html>
```

2. Processament del contingut estàtic

Si volem que se servisquen automàticament aquests continguts en accedir a una URI concreta (que pot ser la pròpia uri "/public" o una altra), emprem el *middleware* **static**, integrat en Express. Com a primer paràmetre del comando `use` indiquem què URI volem utilitzar per a servir contingut estàtic (la que nosaltres vulguem), i en segon lloc, carreguem el *middleware* `static` indicant en quina carpeta estan realment aquests continguts (carpeta `/public` en el nostre cas):

```
const express = require('express');

let app = express();
app.use('/public', express.static(__dirname + '/public'));

app.listen(8080);
```

Opcionalment, també podem definir una ruta que redirigisca a la pàgina d'inici ("index.html") si s'intenta accedir a l'arrel de l'aplicació:

```
app.get('/', (req, res) => {
    res.redirect('/public/index.html');
});
```

Aquesta manera de servir contingut estàtic, com podem deduir, es queda molt curta. No hi ha quasi cap web que oferisca un contingut sense un cert dinamisme; el propi llistat de notícies podria extraure's d'una base de dades i bolcar-se en la pàgina... però per a això necessitem fer ús de plantilles, que veurem en documents posteriors.

3. Incloure estils predefinits: Bootstrap

És possible també incloure els estils de frameworks de disseny web, com per exemple *Bootstrap*, directament en la nostra aplicació Express, seguint aquests senzills passos:

1. En primer lloc, descarreguem Bootstrap com un mòdul més del nostre projecte:

```
npm install bootstrap
```

2. Després, apliquem el *middleware* de contingut estàtic perquè carregue per defecte els continguts de la carpeta `dist` d'instal·lació de Bootstrap, on es troben, entre altres coses, els estils.

```
app.use(express.static(__dirname + '/node_modules/bootstrap/dist'));
```

En aquest cas, els continguts de Bootstrap es carregaran associats a la ruta arrel del projecte (no hem indicat cap prefix com a primer paràmetre, cosa que sí hem fet en l'exemple anterior `/public`). Per tant, si volem incloure estils Bootstrap en qualsevol pàgina HTML o vista del nostre projecte, podem fer-lo així:

```
<link rel="stylesheet" href="/css/bootstrap.min.css">
```

Això accedirà a la carpeta `node_modules/bootstrap/dist/css` per a localitzar el fitxer CSS indicat. D'altra banda, si volem carregar estils propis de la nostra carpeta `public/css`, ho farem d'aquesta altra manera, i així no interferiran (cada ruta estàtica té un prefix diferent):

```
<link rel="stylesheet" href="/public/css/estils.css">
```