Vamos a crear un JWT Token con Node.js para ver un ejemplo práctico del artículo anterior de JSON Web Tokens . En este caso he elegido Javascript ya que la implementación es muy sencilla. Lo primero que tendremos que hacer es instalar varias librerías de Node.

- npm install express
- npm install body-parser
- npm install jsonwebtoken
- npm install express-jwt

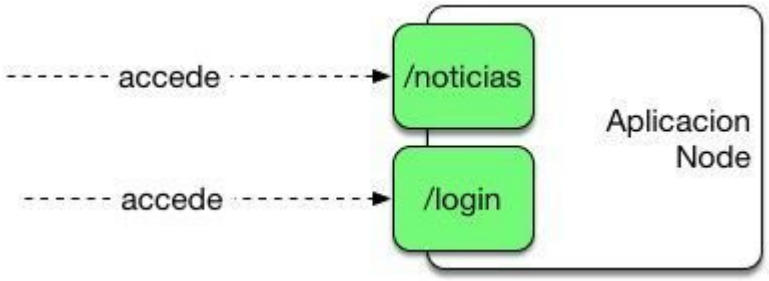
JWT Token y Node

Es momento de construir nuestra aplicación con Node.js:

```
var express = require('express');
var bodyParser=require('body-parser');
var jwt=require('jsonwebtoken');
var expressJwt=require('express-jwt');
var app = express();

var jwtClave="laclave_de_cecilio";
app.use(express.static('publica'));
app.use(bodyParser.json());
app.use(expressJwt({secret:jwtClave}).unless({path: ["/login"]}));


var usuario= {
 nombre:"cecilio",
 clave:"cecilio"
}
var noticias = [{
```

```
 id: 1,
 titulo: "noticia 1"
}];

app.get('/noticias', function(req, res) {
 res.send(noticias);
});


app.post("/login",function(request,response) {

if (request.body.nombre==usuario.nombre ||
request.body.clave==usuario.clave) {

 var token=jwt.sign({
 usuario:"cecilio"
 },jwtClave);

 response.send(token);

}else {

 response.status(401).end("usuario incorrecto")
}

});

app.listen(3000, function() {
 console.log('aplicacion en el puerto 3000!');
});
```

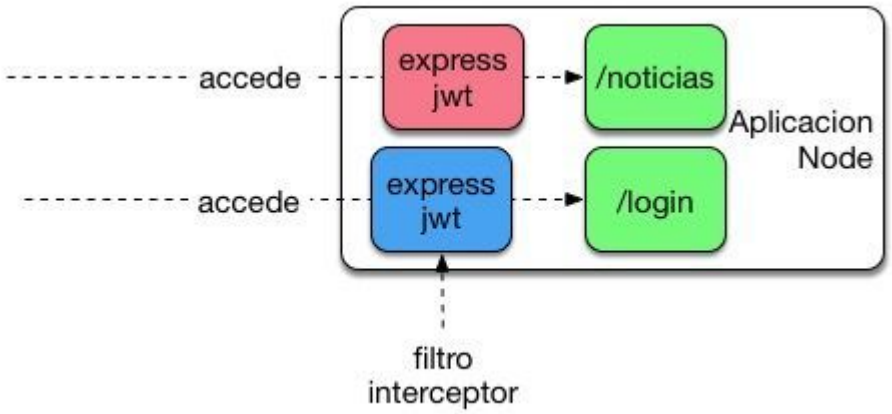
No es mucho código pero vamos a explicarlo. En primer lugar lo que más destaca es que disponemos de dos URL de acceso. Una es /login que servirá para logearnos en el servidor la otra es /noticias que nos devuelve una noticia.



Ambas URL están protegidas por el módulo de express-jwt que se registra previamente y que hace la función de filtro o interceptor:

```
var jwtClave="laclave_de_cecilio";
app.use(expressJwt({secret:jwtClave}).unless({path: ["/login"] }));
```

Este módulo protege todas las URL salvo la url de Login y los recursos estáticos :



Cada vez que nosotros solicitemos la url de noticias nos devolverá acceso no permitido ya que no tenemos un token:

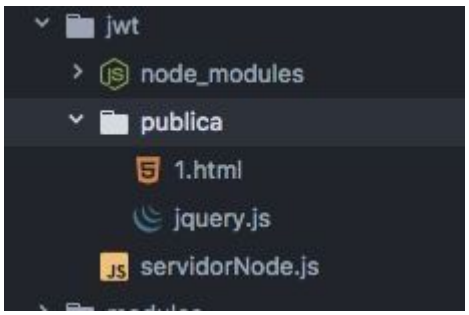
```
UnauthorizedError: No authorization token was found
    at middleware (/Users/cecilio/Dropbox/Cursos Jav
    at /Users/cecilio/Dropbox/Cursos Javascript/ES6/
    at Layer.handle [as handle_request] (/Users/ceci
    at trim_prefix (/Users/cecilio/Dropbox/Cursos Ja
    at /Users/cecilio/Dropbox/Cursos Javascript/ES6/
    at Function.process_params (/Users/cecilio/Dropb
    at next (/Users/cecilio/Dropbox/Cursos Javascrip
    at expressInit (/Users/cecilio/Dropbox/Cursos Ja
    at Layer.handle [as handle_request] (/Users/ceci
    at trim_prefix (/Users/cecilio/Dropbox/Cursos Ja
```

Así pues tenemos que construirnos una aplicación cliente que envíe una petición POST al servidor a la URL de Login y pase usuario “cecilio” y clave “cecilio”, esto nos devolverá un token:

```
app.post("/login",function(request,response) {

if (request.body.nombre==usuario.nombre ||
request.body.clave==usuario.clave) {

 var token=jwt.sign({
 usuario:"cecilio"
 },jwtClave);

 response.send(token);

}else {

 response.status(401).end("usuario incorrecto")
}
```

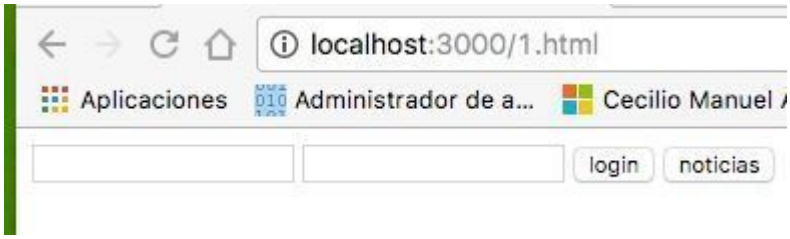
```
});
```

Vamos a construir un documento html en la carpeta pública del servidor que se apoye en jQuery y se encargue de realizar tanto la petición de login como la de las noticias:




```
<body>
<input type="text" id="nombre" />
<input type="text" id="clave" />

<input type="button" id="login" value="login" />
<input type="button" id="noticias" value="noticias" />
</body>
```

Unicamente tenemos dos cajas de texto y dos botones (login,noticias) .



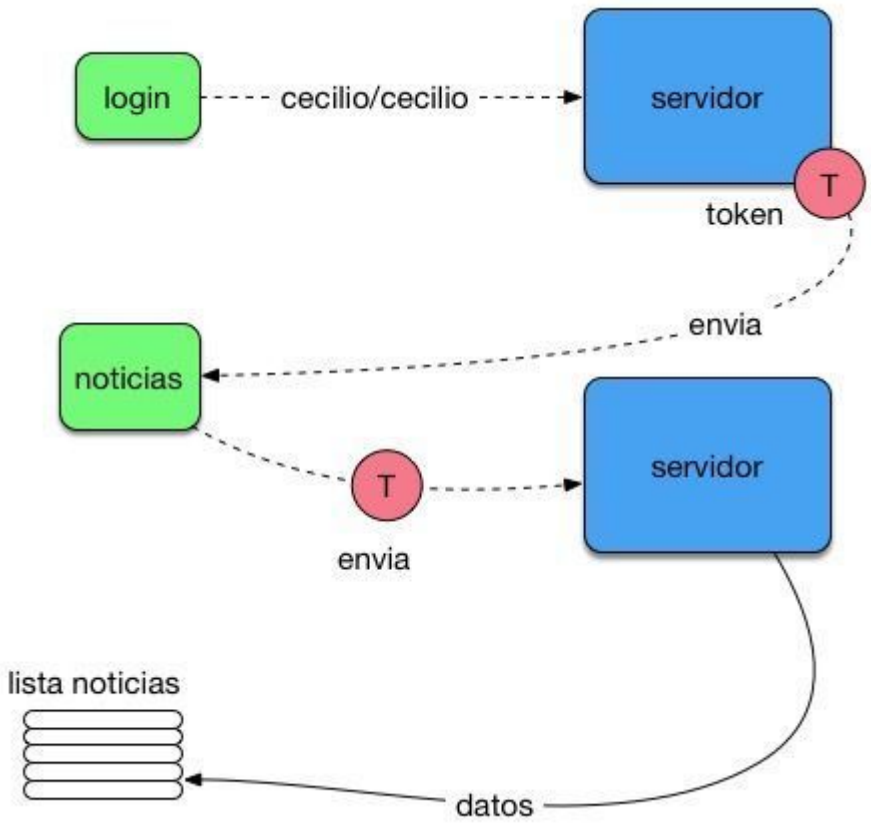
El primer botón realiza una petición Ajax al servidor y nos devuelve un token. El segundo botón solicita las noticias. Vamos a ver el código de JavaScript creado con jQuery que es algo que todos entendemos.

```
$(document).ready(function() {

            var token = "";

            $("#login").click(function() {

                var usuario = {
                    nombre: $("#nombre").val(),
                    clave: $("#clave").val()
                };

                $.ajax({
                    url: 'login',
                    type: 'post',
                    contentType: 'application/json; charset=utf-8',
                    data: JSON.stringify(usuario),
                    success: function(data) {
```

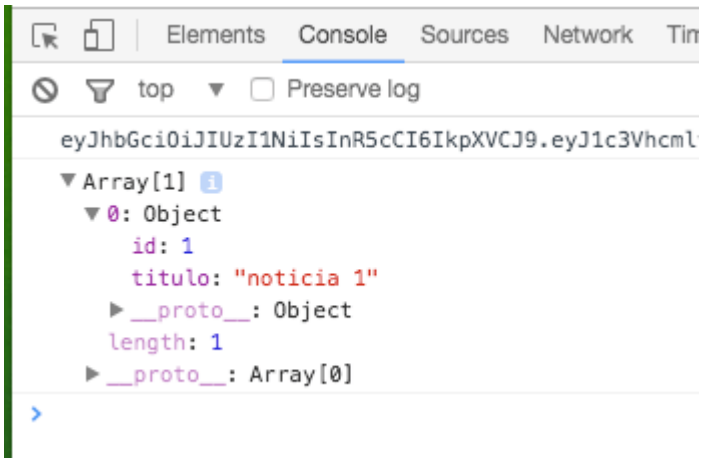
```
                token=data;


                },
                error:function(error) {

                  console.log(error);
                }

            });

        });

        $("#noticias").click(function() {


            $.ajax({
                url: 'noticias',
                type: 'get',
                contentType: 'application/json; charset=utf-8',
                dataType: 'json',
                success: function(data) {
                    console.log(data);
                },
                beforeSend: function(xhr) {

                    console.log(token);

                    xhr.setRequestHeader("Accept",
"application/json");
                    xhr.setRequestHeader("Authorization", "Bearer
```

```
"+token);
                    },

                });
            });

        });
```

El código no es muy complicado y únicamente gestiona dos eventos “login” y “noticias” . El primero de ellos solicita un token al servidor realizando una petición AJAX y enviando los datos en JSON. Cuando recibe la respuesta almacena el token en una variable global. El evento de noticias hace una petición al servidor solicitando las noticias y pasando el token que acabamos de obtener como parámetro en la cabecera.



De esta forma podremos acceder a los datos solicitando primero el JWT token para despues imprimirlos en la consola:



Otros artículos relacionados:

- JSON Web Tokens
- JSON JQuery y Templates
- Arquitecturas Web Abiertas vs Cerradas