Envío-Recepción de Formularios

Cuando vamos a hacer una solicitud **`POST`** para enviar un formulario tenemos dos opciones para el encabezado **`Content-Type`** dependiento del tipo de dato a enviar y su tamaño.
De este encabezado depende como se enviará la información en el **`body`**:

`Content-Type: application/x-www-form-urlencoded`

Util si vamos a enviar poca información.
La información será enviada en una cadena formada por pares **`name=value`**, separados por ampersand **`&`**
Es similar el envío de parámetros en el query string de una petición HTTP,**`VariableUno=ValorUno&VariableDos=ValorDos....`**
Carácteres especiales serán sustituidos por **`%HH`** (dos hexadecimales que represetan el código ASCII)

> Si vamos a enviar un archivo:
> La conversión de carácteres alfanuméricos a tres caracteres tripicará el tamaño de la petición.

`**content-Type: multipart/form-data; boundary={boundary string}`

Util si vamos enviar mucha información o también si necesitamos enviar archivos.
Cada par **`name=value`** será enviado como parte de un mensaje MIME y separados por una cadena llamada **`boundary`**
`boudary string`: Es una cadena aleatoria que genera el navegador (Ejem: **`----WebKitFormBoundaryKGUmWkAsjo5nUBp2`**)

> Si vamos a enviar pocos campos de texto:
> Agregarles el **`Boudary string`** hará que aumente el tamaño de la petición.

Envío-Recepción de Formularios

Envío del Formulario (Html)

Usando el atributo `enctype` de la etiqueta `<form>`:

`Content-Type: application/x-www-form-urlencoded`

```
<form id="idForm" action="http://..." method="POST" enctype="application/x-www-form-urlencoded">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre">
    <label for="apellido">Nombre:</label>
    <input type="text" id="apellido">
    <button type="submit" value="Enviar">
</form>
```

`**content-Type: multipart/form-data; boundary={boundary string}`

```
<form id="idForm" action="http://..." method="POST" enctype="multipart/form-data">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre">
    <label for="apellido">Nombre:</label>
    <input type="text" id="apellido">
    <button type="submit" value="Enviar">
</form>
```

Envío-Recepción de Formularios

Envío del Formulario (Javascript)

Usando el API Fetch de Javascript:

`Content-Type: application/x-www-form-urlencoded`

```
const datosForm = new FormData(document.getElementById("idForm"));
const cadenaUrl = new URLSearchParams(datosForm).toString();

//Para enviar el formulario como application/x-www-form-urlencoded
fetch('http://...',{ method: 'POST',
                     headers: {"Content-Type": "application/x-www-form-urlencoded"}, //Debemos incluir header Content-Type
                     body: cadenaURLSearch })
  .then(response=>response.json())
    .then(dato=>console.log(dato));
```

`content-Type: multipart/form-data; boundary={boundary string}`

```
const datosForm = new FormData(document.getElementById("idForm"));

//Para enviar el formulario como multipart/form-data
fetch('http://...',{ method: 'POST',              //No debemos incluir header Content-Type, el navegador lo agregará
                     body: datosForm })           //automáticamente ya que debe generar una cadena "Boundary" en el body
  .then(response=>response.json())                //y también incluirla en el encabezado Content-Type
    .then(dato=>console.log(dato));
```

Envío-Recepción de Formularios

Envío del Formulario en el body la petición (payload)

Dependiendo del contenido ded header `Content-Type`, en el body de la petición
el formulario se enviará de la siguiente manera:

`Content-Type: application/x-www-form-urlencoded`

```
campo1=valor1&campo2=valor2
```

`**content-Type: multipart/form-data; boundary={boundary string}`

```
--{boundary string}
Content-Disposition: form-data; name="campo1",

valor1
--{boundary string}
Content-Disposition: form-data; name="campo2",

valor2
--{boundary string}
```

Envío-Recepción de Formularios

Recepción del Formulario (Javascript)

Si fue enviado con `Content-Type: application/x-www-form-urlencoded`

- **`express.urlencoded()`** *Middleware ya incorporado en Express*
- **`body-parser`** *Paquete de npm que también interpreta json y texto*

```
const express = require('express');
const app = express();                              // Ejecutamos la funcion express()

app.use( express.urlencoded({extended: true,}) );  // Funcion Middleware Para interpretar un body codificado como URLEncoded

app.post('/usuario/',(req,res) => {
    console.log(req.body);
    res.send('Se recibio el formulario:'+JSON.stringify(req.body));
});

app.listen(8082,()=>{
    console.log("Servidor Express corriendo y escuchando en el puerto 8082")
});
```

Envío-Recepción de Formularios

Recepción del Formulario (Javascript)

Si fue enviado con **`content-Type: multipart/form-data; boundary={boundaryString}`**
Podemos instalar estos paquetes del registro npm:

- **multer**
- **busboy**
- **formidable**
- **multiparty**

Envío-Recepción de Formularios

Recepción del Formulario (Javascript-Multer)

```
const express = require('express');
const multer  = require('multer');
const app     = express();                          // Ejecutamos la funcion express()

const folder = path.join(__dirname+'/archivos/'); //Definir una ruta para almacenar archivos que se envian del cliente
const upload = multer({dest:folder});             //Pasarle esa ruta a multer()

app.use(upload.single('archivo'));                // Funcion Middleware que interpreta un body codificado como FormData
                                                  // Suponiendo que tuviesemos un campo <input type=file" name="archivo">

app.use(upload.none());                           // Funcion Middleware que interpreta un body codificado como FormData
                                                  // Suponiendo que tuviesemos solamemte campos <input type=text>

//Como la solicitud ya paso por las funciones Middleware
//El objeto req ya tiene una propiedad file con el archivo y en la propiedad body los datos del formulario
//Podemos procesar esos datos en nuestros metodos/rutas que siguen abajo

app.post('/usuario/',(req,res) => {
    console.log(`Se recibio el archivo: ${req.file.originalname}`);
    console.log(req.body);
    console.log('Se recibio el formulario:'+JSON.stringify(req.body));
    res.json(req.body);
});

app.listen(8082,()=>{
    console.log("Servidor Express corriendo y escuchando en el puerto 8082")
});
```