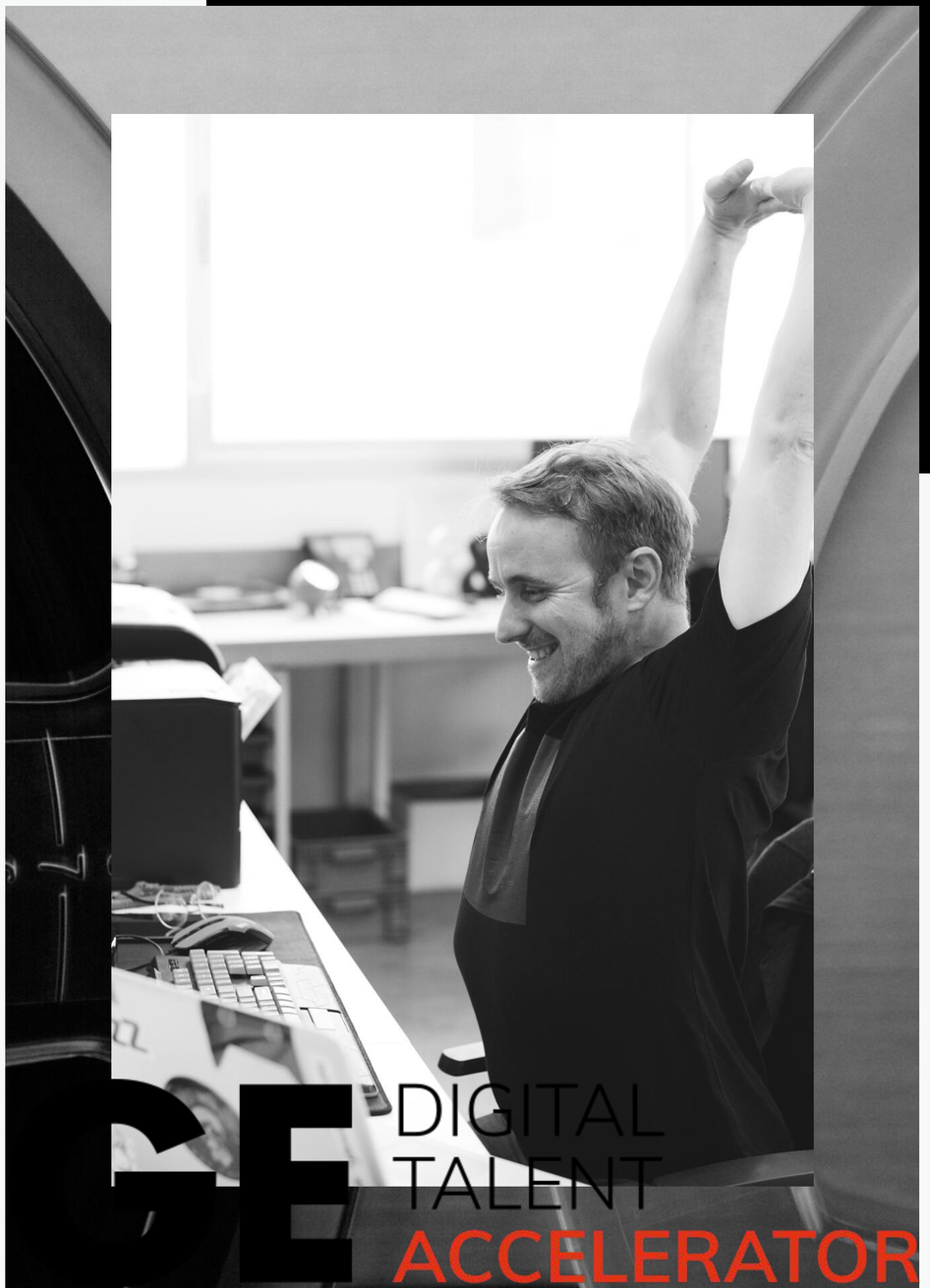
Express

Fundación ONCE | Inserta
Cofinanciado por la Unión Europea
MINISTERIO DE TRABAJO Y ECONOMÍA SOCIAL
Fondos Europeos

THE BRIDGE
DIGITAL TALENT ACCELERATOR

Índice



THE BRIDGE DIGITAL TALENT ACCELERATOR

¿Qué es un framework?

La traducción literal de framework es ‘marco de referencia’, y explica muy bien lo que significa. Un framework es como un **patrón o esquema que ayuda a la programación a estructurar el código y a ahorrar tiempo y esfuerzos a los programadores**.

Se trata de una herramienta de **programación versátil**, ya que está incompleta y, al añadir líneas de código, la convertimos en una determinada aplicación o nos permite incorporar nuevas funciones.

¿Qué es Express?

Express.js es un **framework de Node.js**.
Gracias a Express crear una API robusta es rápido y fácil.

Express es un **framework del "lado del servidor"** o "backend".
No es comparable con frameworks del lado del cliente como Angular o Vue. Se puede usar en combinación con estos frameworks para crear aplicaciones full stack completas.

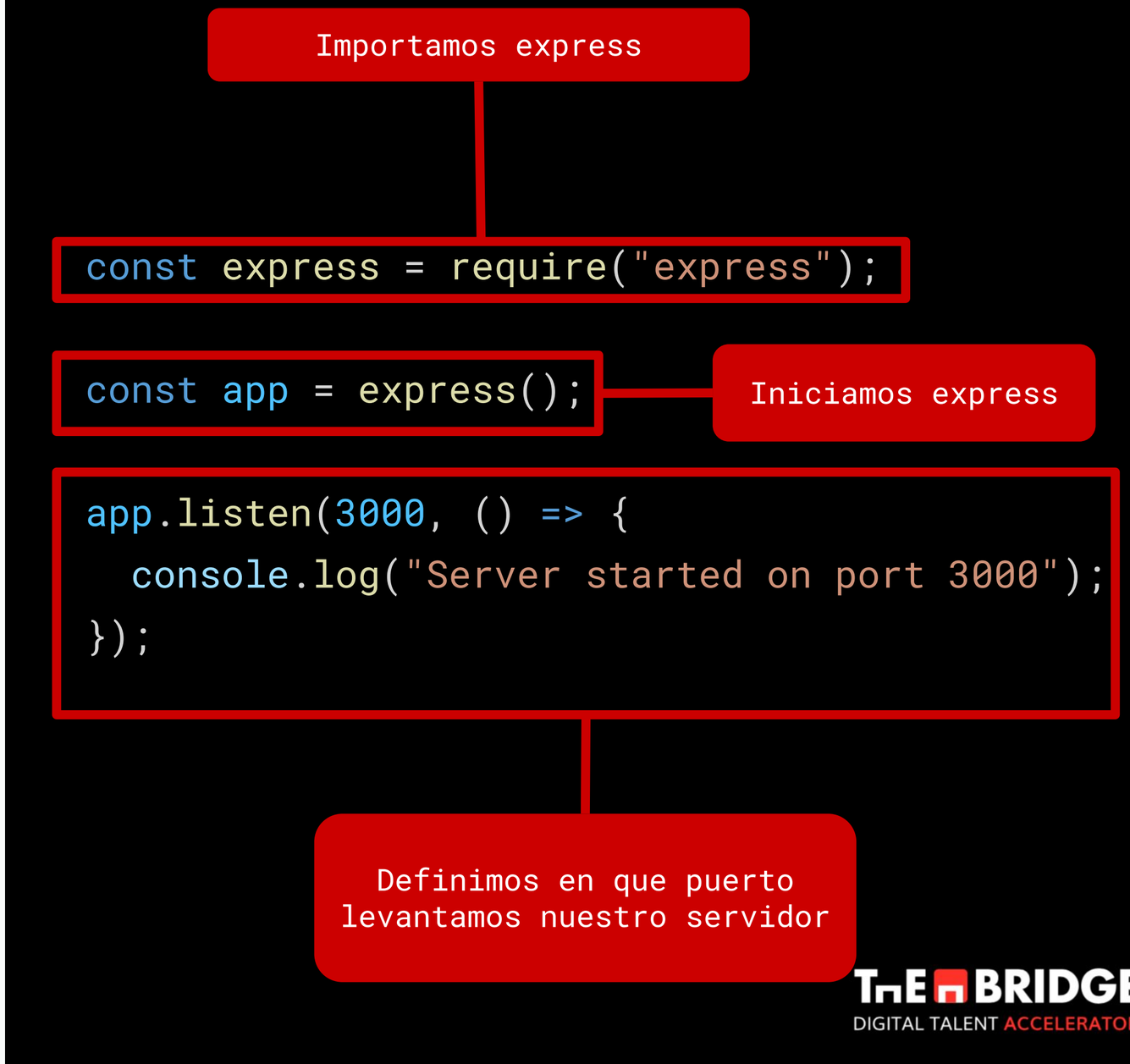
Instalación

Express

```
$ npm install express
```

Empezando

- Creamos un archivo llamado "index.js" y agregamos el siguiente código:



Endpoints

- **El manejo de endpoints/rutas es simple**
  - app.get()
  - app.post()
  - app.put()
  - app.delete()

```
app.get('/',(req,res)=>{
    // Obtener datos de la bd
    // Cargar páginas
    // Devolver un json
    // Total acceso a la request & response
})
```

(información que enviamos) representa la respuesta HTTP, lo que se devuelve como datos JSON etc

(información que nos llega) representa las propiedades de solicitud HTTP para cosas como parametros de URL, cualquier dato enviado por el body,etc.

Nuestro primer Endpoint

- **Endpoint.** Los endpoints son las URLs de un API o un backend que responden a una petición.

```
app.get('/',(req,res)=>{
    res.send('Hola The Bridge')
})
```

Creamos nuestro propio endpoint/ruta

Request

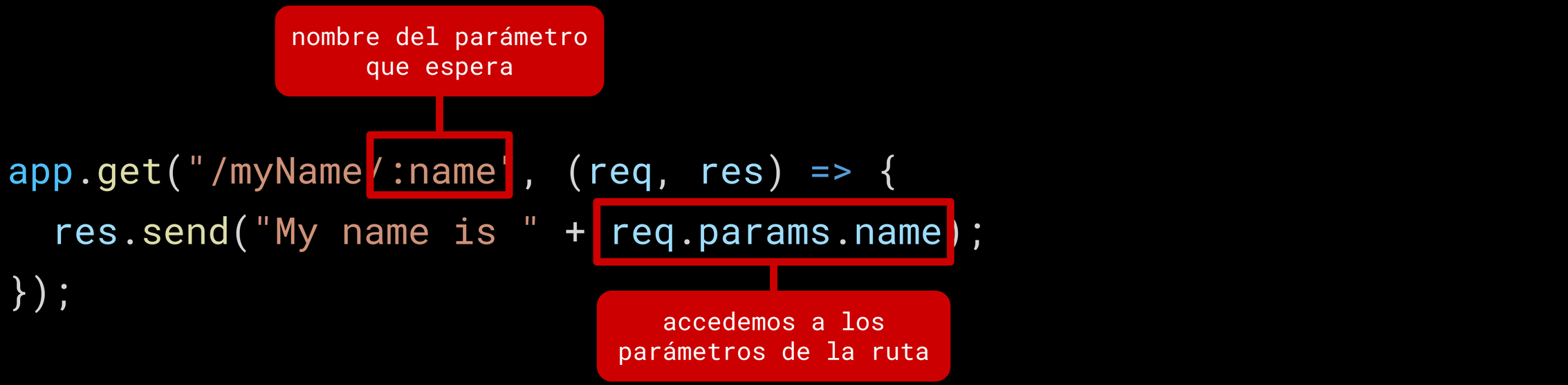
Vamos a profundizar un poco más en Request, el método que pasa toda la información del navegador al servidor.

Desde una petición **REQUEST**, nos puede llegar la información de distinta manera:

- **En la ruta** (req.params.nombreDelParametro)
  - http://localhost:3000/inicio/82
- **Como Query String**(req.query.nombreQuery)
  - http://localhost:3000/inicio?id=82
- **En el Body** (req.body.nombreCampoEscritoEnElBody)
  - http://localhost:3000/inicio
    {nombre: 'Gustavo Pascual Falcó'}

En la ruta

Es importante tener en cuenta que cuando utilicemos req.params accederemos al nombre del parámetro que hemos definido antes en el endpoint:



Query String

Es importante tener en cuenta que cuando utilicemos req.query accederemos al nombre que hemos definido en la ruta:

```
app.get("/myName", (req, res) => {
  res.send("My name is " + req.query.name);
});
// localhost:3000/myName/?name=pedro
```

definimos la query

¿Qué es Postman?

Nos permite realizar peticiones de una manera simple para testear APIs propias o de terceros.

Instalación:
https://www.postman.com/downloads/

¿Qué es un middleware?

Son esos métodos/funciones/operaciones que se llaman ENTRE el procesamiento de la Solicitud y el envío de la Respuesta en su método de aplicación.

Body

Recogemos la información pasada por el body de la siguiente forma:

```
app.use(express.json())

app.post("/myName", (req, res) => {
  res.send("My name is " + req.body.name);
});
```

método incorporado en express para reconocer el objeto de solicitud entrante como **objeto JSON**

accedemos a la información que nos llega por el body

Cargando archivos HTML

- En esta ruta crearemos un endpoint que enviará un archivo utilizando **res.sendFile()**

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

const app = express();

app.get('/',(req,res)=>{
    res.sendFile(path.join(__dirname,'public','index.html'))
})

app.listen(3000, () => {
  console.log("Server started on port 3000");
});
```

Creamos nuestra ruta para que nos muestre una página html

Cargando archivos HTML

Cargando archivos HTML de forma estática

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

const app = express();

app.use(express.static(path.join(__dirname, 'public')));

app.listen(3000, () => {
  console.log("Server started on port 3000");
});
```

Members CRUD

```
const members = [
  {
    id: 1,
    name: "John Doe",
    email: "john@gmail.com"
  },
  {
    id: 2,
    name: "Bob Williams",
    email: "bob@gmail.com"
  },
  {
    id: 3,
    name: "Shannon Jackson",
    email: "shannon@gmail.com"
  },
];
```