



Seguir

Sé el primero de tus amigos  
en seguir a Miríada X.

Follow @miriadax



Mi Página

Cursos

Universidades e instituciones

Conócenos

Soporte

## Desarrollo de servicios en la nube con HTML5, Javascript y node.js

[Inicio](#) | [Syllabus](#) | [Foro](#) | [Blog](#)

### Módulos

- **Modulo 0. Introducción al curso, al programa y al Sistema Operativo UNIX**
- **Modulo 1. Introducción a JavaScript de servidor y a node.js. Sentencias, Variables, Booleanos, Números, Strings y Funciones**
- **Modulo 2. Introducción a JavaScript de servidor y a node.js. Bucles, Clases predefinidas, Objetos, Propiedades y Métodos; Prototipos y Clases; Arrays; JSON; Funciones como Objetos y Cierres (Closures)**
- **Modulo 3. Modulos node.js; Expresiones Regulares; Eventos, Entorno de Ejecución y Concurrencia en node.js; Ficheros y Flujos**
- **Modulo 4. Introducción a HTTP y a los Servidores**

Modulo 4. Introducción a HTTP y a los Servidores Web; Introducción a express y al Middleware Static; Introducción a REST; Aplicaciones express.js y Composición de Middlewares; Formularios GET y POST; Parámetros Ocultos

## Tema 6. Cuestionario obligatorio

Prueba realizada

Tu resultado en el test ha sido: 100%

**Has superado el test.**

### Tus respuestas

Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 || ''));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:

Web; Introducción a express y al Middleware Static; Introducción a REST; Aplicaciones express.js y Composición de Middlewares; Formularios GET y POST; Parámetros Ocultos

- ✓ Tema 0: Transparencias del módulo
- ✓ Tema 1. introducción a HTTP
- ✓ Tema 1. Cuestionario obligatorio
- ✓ Tema 2. Servidor Web
- ✓ Tema 2. Cuestionario obligatorio
- ✓ Tema 3. Introducción a express.js y al middleware static
- ✓ Tema 3. Cuestionario obligatorio
- ✓ Tema 4. Introducción a REST
- ✓ Tema 4. Cuestionario obligatorio
- ✓ Tema 5. Aplicaciones REST con express.js
- ✓ Tema 5. Cuestionario obligatorio
- ✓ Tema 6. Acceso a campos de la ruta
- ➡ Tema 6. Cuestionario obligatorio
- ✓ Tema 7. Composición y ejecución de middlewares
- ✓ Tema 8. Formulario GET
- ✓ Tema 9. URL encode
- ✓ Tema 10. Formulario POST
- ✓ Tema 11. Parámetro oculto y method override

http://localhost/P2

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"
- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☒ "Nadie"



Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 | | ""));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:  
http://localhost/37/signals

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☒ "37signals"
- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☐ "Nadie"



Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

- **Modulo 5. Gestión de versiones de proyectos con git y GITHUB; Proyecto, Espacio de Trabajo y Versiones (Commit); Arboles y Ramas de un proyecto; Repositorios Remoto y colaboración a través de GITHUB**
- **Modulo 6. Proyecto Quiz I: Patrón Modelo-Vista-Controlador (MVC); generación del proyecto con express-generator; Primera Página y Primera Pregunta; Despliegue en la nube (Heroku)**
- **Modulo 7. Proyecto Quiz II: La Base de Datos (DB), Tablas, sequelize.js y SQLite; Despliegue en Heroku utilizando Postgres; Presentación de Listas de Quizes y Autoload**
- **Modulo 8. Proyecto Quiz III: Gestión de Listas de Quizes, Creación, Edición y Borrado**
- **Modulo 9. Proyecto Quiz IV: Creación y Moderación de Comentarios a Quizes; Relaciones entre Tablas de la Base de Datos; Sesiones, Autenticación y Autorización; HTTP Seguro (HTTPS)**

```
var express = require('express');
var app = express();
app.get('/:id1(\\d+)/:id2?', function (req,res){
res.send( req.params.id1 + (req.params.id2 | | ""));
});
app.get('*', function (req, res){res.send( 'Nadie' );});
app.listen(80);
```

-----

Con qué texto responderá esta aplicación a la siguiente URL:  
http://localhost/signals

---

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"
- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☒ "Nadie"



Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

-----

```
var express = require('express');
var app = express();
app.get('/:id1(\\d+)/:id2?', function (req,res){
res.send( req.params.id1 + (req.params.id2 | | ""));
});
app.get('*', function (req, res){res.send( 'Nadie' );});
app.listen(80);
```

-----

Con qué texto responderá esta aplicación a la siguiente URL:  
http://localhost/3P2

---

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"

- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☒ "Nadie"



---

Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 | | ""));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:  
<http://localhost/3/P2>

- 
- ☒ "3P2"
  - ☐ "3"
  - ☐ "P2"
  - ☐ "37signals"
  - ☐ "4caminos"
  - ☐ "cuatrocaminos"
  - ☐ "Nadie"



---

Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 | | ""));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:<http://localhost/cuatro/caminos>

---

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"
- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☒ "Nadie"



Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 || ''));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:  
<http://localhost/4caminos>

---

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"
- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☒ "Nadie"



Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 || ""));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:  
<http://localhost/37signals>

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"
- ☐ "4caminos"
- ☐ "cuatrocaminos"
- ☒ "Nadie"




Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 || ""));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:  
<http://localhost/3>

- ☐ "3P2"
- ☒ "3"
- ☐ "P2"

- 
- ☐ "37signals"
  - ☐ "4caminos"
  - ☐ "cuatrocaminos"
  - ☐ "Nadie"



Supongamos que en nuestro ordenador local arrancamos la siguiente aplicación express:

```
-----  
var express = require('express');  
var app = express();  
app.get('/:id1(\\d+)/:id2?', function (req,res){  
  res.send( req.params.id1 + (req.params.id2 || ""));  
});  
app.get('*', function (req, res){res.send( 'Nadie' );});  
app.listen(80);  
-----
```

Con qué texto responderá esta aplicación a la siguiente URL:  
<http://localhost/4/caminos>

- ☐ "3P2"
- ☐ "3"
- ☐ "P2"
- ☐ "37signals"
- ☒ "4caminos"
- ☐ "cuatrocaminos"
- ☐ "Nadie"



**Enhorabuena. Terminaste este módulo.**

 anterior

Siguiente 

