UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES
Centro de e-Learning
Secretaría de Cultura y Extensión Universitaria

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Diplomatura en programación web full stack con React JS

UTN.BA | Centro de e-Learning
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES
Secretaría de Cultura y Extensión Universitaria

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Módulo 4:

Introducción Reactjs y Nodejs

Unidad 4:

Peticiones, respuestas y templates

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Presentación

En esta unidad veremos los conceptos de plantillas, peticiones, middleware y sesiones para incorporar funcionalidades avanzadas e interactuar con los datos enviados por los usuarios.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Objetivos

Que los participantes logren...

- Capturar datos enviados por el usuario en nuestros controladores.
- Utilizar el framework handlebars para manejar plantillas HTML
- Aprender a enviar datos al navegador del usuario.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Bloques temáticos

1. Captura de datos del request.
2. Templates / Vistas / Enviar datos al navegador.
3. Middleware.
4. Sesiones.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

1. Captura de datos del request

Los parámetros **POST** son aquellos que se envían desde la página web al servidor sin que sean visibles en la URL.

Lo primero que deberemos de conocer es que los parámetros POST no se envían en la URL si no que se envían en el cuerpo de la petición. Siendo el tipo de petición recibida por los servidores como `application/x-www-form-urlencoded`. Si bien el formato de los parámetros es el mismo que el utilizado por los parámetros GET.

Para conformar la petición **POST** vamos a utilizar el método **.post**

El objeto **req** representa la solicitud HTTP y tiene propiedades para la cadena de consulta de la solicitud, parámetros, cuerpo, encabezados HTTP, etc. En esta documentación y por convención, el objeto siempre se denomina req (y la respuesta HTTP es **res**), pero su nombre real está determinado por los parámetros de la función de devolución de llamada en la que estás trabajando.

Propiedades del objeto Request

req.body

Contiene pares de datos clave-valor enviados en el cuerpo de la solicitud. De forma predeterminada, no está definido y se rellena cuando utiliza middleware de análisis corporal como express.json() o express.urlencoded().

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

```
app.post('/saludo', function (req, res) {
    var nombre = req.body.nombre || '';
    var saludo = '';

    if (nombre != '')
        saludo = "Hola " + nombre;

    res.send(saludo);

})
```

req.query

A la hora de acceder al parámetro **GET** utilizamos el objeto **req.query** seguido del nombre que tenía el parámetro en el formulario (el nombre definido mediante el atributo name).

```
app.get('/saludo', function (req, res) {
    var nombre = req.query.nombre || '';
});
```

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

2. Templates / Vistas / Enviar datos al navegador

Templates con Handlebars

El **Motor de plantilla** (referido como "motor de vistas" por Express) le permite definir la estructura de documento de salida en una plantilla, usando marcadores de posición para datos que serán llenados cuando una página es generada. Las plantillas son utilizadas generalmente para crear HTML, pero también pueden crear otros tipos de documentos.

Express tiene soporte para numerosos motores de plantillas, como es el caso de **Handlebars** que es una extensión de Mustache.js y es un motor de plantillas muy popular ya que es basado en JavaScript y podemos utilizarlo tanto en lado servidor como en el cliente.

Handlebars nos permite escribir etiquetas HTML y luego dentro con código del motor propio podemos definir que imprime del contexto y la forma en que lo hace.

Instalar handlebars

Cuando nosotros instalamos express, instalamos también el motor de plantillas de handlebars, ya que queda prolijamente declarado en los archivos.

```
npx express-generator --view=hbs
```

Imprimir elementos del contexto

Nuestra vista genera como resultado datos que debemos mostrar al usuario, dichos datos los pasamos a través del contexto a nuestra plantilla y aquí es donde los imprimimos. Para imprimir esto simplemente debemos encerrar la variable o el elemento en llaves dobles como lo siguiente:

```
{{nombre}}
```

Eso nos lleva a ver el contenido de “nombre” que hayamos definido en nuestra vista, estas dobles llaves llevan el escape de caracteres de forma automática, de tal manera que no resulte la impresión de código no permitido por omisión del desarrollador.

Ahora si queremos imprimir un texto sin escapar debemos utilizar triples llaves, esto le indica a Handlebars que no debe escapar nada, veamos el ejemplo:

```
{{{nombres}}}
```

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Comentarios

Los comentarios se declaran así:

```
{{! esto es un comentario en Handlebars }}
```

Renderización de data (vistas)

Las plantillas se almacenan en el directorio **views** (como se especifica en app.js) y se les da la extensión de archivo **.hbs**.

El método **Response.render()** se utiliza para representar una plantilla específica junto con los valores de las variables nombradas que se pasan en un objeto y luego enviar el resultado como respuesta. En el siguiente código se puede ver cómo una ruta muestra una respuesta usando la plantilla "index".

```
/* GET home page. */
router.get('/', function(req, res) {
  res.render('index');
});
```

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Enviar datos al navegador

Podemos enviar datos al navegador del usuario pasando un objeto como segundo parametro del metodo render(). Dentro de este objeto incluiremos un conjunto de datos definidos como clave-valor donde cualquier tipo de dato es válido.

```
res.render('plantilla', {
    titulo: 'Lista de paises',
    lista: ['Argentina', 'Uruguay', 'Brasil'],
    activado: true,
    cantidad: 10
});
```

```
<h1>{{ titulo }}</h1>

<ul>
    {{#each lista}}
    <li>{{this}}</li>
    {{/each}}
</ul>

{{#if activado}}
<h3>Si estoy activo muestro {{cantidad}}</h3>
{{/if}}
```

En el primer archivo podemos ver como estamos llamando al método **render** para enviar al navegador el template llamado plantilla.hbs junto con **4 tipos** de datos distintos: un string, un array, un booleano y un número.
En el segundo archivo vemos cómo se interpretan estos valores en la plantilla y vemos 2 de los bloques principales de Handlebars. Tanto el string de título como el número de cantidad se representan directamente, mientras que el valor de activado

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

lo usamos dentro del bloque **{{#if}}{{/if}}** para evaluar si es verdadero, lo que hace que se muestre todo lo incluido en el bloque. En caso contrario el usuario nunca vería el h3 ni el valor de la variable cantidad.

Para el array lista el bloque empleado es el **{{#each}}{{/each}}** que nos permite recorrer arrays. Para acceder al valor de cada interacción del bucle Handlebars usa la variable this.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

3. Middleware

Un middleware es una función que se puede ejecutar antes o después del manejo de una ruta. Esta función tiene acceso al objeto Request, Response y la función next().

Las funciones middleware suelen ser utilizadas como mecanismo para verificar niveles de acceso antes de entrar en una ruta, manejo de errores, validación de datos, etc.

En resumen las funciones de middleware pueden realizar las siguientes tareas:

- Realizar cambios en la solicitud y los objetos de respuesta.
- Finalizar el ciclo de solicitud/respuestas.
- Invocar el siguiente middleware en la pila.

Si la función de middleware actual no finaliza el ciclo de solicitud/respuestas, debe invocar next() para pasar el control a la siguiente función de middleware. De lo contrario, la solicitud quedará colgada.

El siguiente ejemplo creamos un middleware sencillo para contar las vistas del usuario a una ruta determinada y almacenaremos dicho valor en una variable de

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

```
app.use(function(req, res, next) {
  // si no existe la variable de sesion vistas la
  // creamos como un objeto vacio
  if (!req.session.vistas) {
    req.session.vistas = {};
  }

  /**
   * buscamos una clave dentro session.vistas que
   * coincida con la url actual. Si no existe, la
   * inicializamos en 1. Si existe sumamos 1 al contador
   * de esa ruta
   */
  if (!req.session.vistas[req.originalUrl]) {
    req.session.vistas[req.originalUrl] = 1;
  } else {
    req.session.vistas[req.originalUrl]++;
  }

  next();
});


app.get('/pagina1', function(req, res) {
  /**
   * pasamos al template la variable vistas con el
   * numero devuelto por la clave que pedimos
   */
  res.render('pagina', {
    nombre: 'pagina1',
    vistas: req.session.vistas[req.originalUrl]
  });
});
```

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

4. Sesiones

Las sesiones son un método para hacer que algunas variables estén disponibles en múltiples controladores sin tener que pasarlas como parámetro. A diferencia de las cookies, las variables de sesión se almacenan en el servidor y tienen un tiempo limitado de existencia.

Para identificar al usuario que generó las variables de sesión, el servidor genera una clave única que es enviada al navegador y almacenada en una cookie. Luego, cada vez que el navegador solicita otra página al mismo sitio, envía esta cookie (clave única) con la cual el servidor identifica de qué navegador proviene la petición y puede rescatar de un archivo de texto las variables de sesión que se han creado. Después de un tiempo (configurable) sin peticiones por parte de un cliente (navegador) las variables de sesión son eliminadas automáticamente.

Una variable de sesión es más segura que una cookie ya que se almacena en el servidor. Otra ventaja es que no tiene que estar enviándole continuamente como sucede con las cookies.

Uno de los usos más comunes de las variables de sesión en las aplicaciones web es el proceso de autenticación y autorización de las distintas peticiones que el usuario realiza.

En primer lugar, debemos de conocer brevemente los términos de autenticación y autorización.

Autenticación es el proceso de verificar si el usuario es el mismo que él declara ser.

Autorización es el proceso de determinar si un usuario tiene los privilegios para acceder a un cierto recurso al que ha solicitado acceder.

Para poder utilizar variables de sesion debemos instalar el modulo **express-session** en nuestro proyecto e inicializarlo correctamente.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Para instalarlo bastará con escribir npm i express-session en nuestra consola. Para configurarlo primero debemos importar el módulo e inicializarlo.

A continuación vemos un pequeño fragmento de código Node.js en el que se ilustra de una manera muy simple el proceso de autenticación y autorización mediante sesiones de express.js. Como era de esperar, hay un punto de inicio de sesión, un punto de cierre de sesión. Para ver la página que está posteada debemos autenticarnos previamente, de esta forma nuestra identidad será verificada y guardada durante la sesión. Cuando cerremos la sesión lo que se producirá internamente es un borrado de nuestra identidad en dicha sesión.

Para poner en funcionamiento debemos instalar el módulo de express-session via npm.

```
npm i express-session
```

Vamos al archivo de configuración (**app.js**)

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

app.use(session({
    secret: 'inserte clave aqui',
    resave: false,
    saveUninitialized: true
}));
```

En este caso inicializamos el middleware de sesiones pasando como parámetro un objeto con las propiedades secret, resave y saveUninitialized. El único parámetro requerido de estos es secret, que será la cadena de texto o clave que se utilizará para generar el id de sesión que se envía en la cookie al usuario y luego verificar que el mismo no haya sido adulterado por el usuario. Este valor no debe ser sencillo de adivinar, por lo que se recomienda usar cadenas de texto aleatorias de por lo menos 20 caracteres.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

A partir de este momento el objeto res que reciben todos nuestros controladores contará con una propiedad llamada session la cual podemos tanto leer como escribir.

En el siguiente ejemplo hacemos una verificación sencilla de la existencia de la variable de sesion nombre. En caso de existir saludamos al usuario usando este valor. En caso contrario consideramos que el usuario es desconocido.

```
app.get('/ejemplo', function(req, res) {
    if (req.session.nombre) {
        res.send('Hola ' + req.session.nombre) ;
    } else {
        res.send('Hola usuario desconocido.') ;
    }
});
```

A continuación vemos un ejemplo más completo donde verificamos también la existencia de una variable de sesión llamada nombre y en base a su existencia mostraremos o no al usuario un formulario que le permita ingresar su nombre y que el mismo quede grabado en la sesión.

Así mismo le daremos al usuario un link que llevará al usuario a una ruta que se encarga de destruir la sesión, permitiendo iniciar el ciclo nuevamente.

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

```
app.get('/', function(req, res) {
  var conocido = Boolean(req.session.nombre);

  res.render('index', {
    title: 'Sesiones en Express.js',
    conocido: conocido,
    nombre: req.session.nombre
  });

});


app.post('/ingresar', function(req, res) {
  if (req.body.nombre) {
    req.session.nombre = req.body.nombre
  }
  res.redirect('/');
});

app.get('/salir', function (req, res) {
  req.session.destroy();
  res.redirect('/');
});
```

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

```
<h1>{{title}}</h1>
{{#if conocido}}
    <p>Hola {{nombre}} <a href="/salir">Salir</a></p>
{{/if}}

{{#unless conocido}}
    <p>No te conozco</p>
    <form action="/ingresar" method="post">
        <input type="text" name="nombre"><button>ingresar</button>
    </form>
{{/unless}}
```

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso
(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

Bibliografía utilizada y sugerida

Artículos de revista en formato electrónico:

Express. Disponible desde la URL: https://expressjs.com/es/

Handlebars. Disponible desde la URL:https://handlebarsjs.com/