



Desarrollo de Aplicaciones Informáticas

Tema: Router en NextJS

Docente: Ing. Pablo Morandi

Ayudante: Matías Marchesi

Curso: 5to año

Especialidad: Informática

Redactado por: Ing. Pablo Morandi

-
- 1. ¿Qué es el Router en Next.js?
 - 2. useRouter - Navegación Programática
 - 3. Métodos del Router
 - 4. useSearchParams - Parámetros de URL
 - 5. Diferencias entre Link y useRouter
-

1. ¿Qué es el Router en Next.js?

El **Router** en Next.js nos permite navegar entre páginas de forma programática, es decir, a través de código JavaScript en lugar de solo usar enlaces (`<Link>`).

¿Cuándo usamos cada uno?

- **Link:** Para navegación directa (como un botón o enlace normal)
- **useRouter:** Para navegación programática (después de validaciones, formularios, etc.)

Historia del Router en Next.js:

- Antes se usaba `next/router` (Router de páginas)
- Ahora en Next.js 13+ se usa `next/navigation` (Router de aplicaciones)

Importante: Nosotros usamos `useRouter` de `next/navigation` ya que trabajamos con el App Router.

2. useRouter - Navegación Programática

El hook `useRouter` nos da acceso a métodos para controlar la navegación desde nuestro código.

Importación y Uso Básico

```
"use client";
import { useRouter } from "next/navigation";

export default function MiComponente() {
  const router = useRouter();

  const irAotraPagina = () => {
    router.push("/otra-pagina");
  };

  return <button onClick={irAotraPagina}>Ir a otra página</button>;
}
```

Nota: Necesitamos `'use client'` porque `useRouter` solo funciona en componentes del cliente.

3. Métodos del Router

El objeto `router` nos proporciona varios métodos útiles para la navegación.

`router.push()`

Navega a una nueva página (como hacer clic en un enlace).

```
const router = useRouter();

// Navegación simple
router.push("/login");

// Con parámetros de consulta
router.push("/perfil?usuario=juan");

// Con rutas dinámicas
router.push(`/usuario/${userId}`);
```

`router.back()`

Vuelve a la página anterior (como el botón "Atrás" del navegador).

```
const router = useRouter();

const volverAtras = () => {
  router.back();
};

return <button onClick={volverAtras}>← Volver</button>;
```

router.forward()

Va hacia adelante en el historial (como el botón "Adelante" del navegador).

```
const router = useRouter();

const irAdelante = () => {
  router.forward();
};
```

router.refresh()

Actualiza la página actual.

```
const router = useRouter();

const actualizarPagina = () => {
  router.refresh();
};
```

router.replace()

Navega a una página pero **reemplaza** la entrada actual del historial (no se puede volver con "Atrás").

```
const router = useRouter();

// En lugar de agregar una nueva entrada, reemplaza la actual
router.replace("/dashboard");
```

4. useSearchParams - Parámetros de URL

El hook `useSearchParams` nos permite leer los parámetros de la URL (lo que viene después del `?`).

Uso Básico

```
"use client";
import { useSearchParams } from "next/navigation";

export default function MiComponente() {
  const searchParams = useSearchParams();

  // URL: /perfil?nombre=juan&edad=25
  const nombre = searchParams.get("nombre"); // 'juan'
  const edad = searchParams.get("edad"); // '25'

  return (
    <div>
      <h2>Perfil de {nombre}</h2>
      <p>Edad: {edad} años</p>
    </div>
  );
}
```

Métodos de useSearchParams

```
const searchParams = useSearchParams();

// Obtener un parámetro específico
const usuario = searchParams.get("usuario");

// Verificar si existe un parámetro
const tieneUsuario = searchParams.has("usuario");

// Obtener todos los parámetros
const todosLosParams = searchParams.toString();

// Iterar sobre todos los parámetros
searchParams.forEach((value, key) => {
  console.log(`${key}: ${value}`);
});
```

5. Diferencias entre Link y useRouter

Ambos sirven para navegar, pero tienen diferentes propósitos.

Cuándo usar Link

```
import Link from "next/link";

export default function Menu() {
  return (
    <nav>
      <Link href="/home">Inicio</Link>
      <Link href="/perfil">Perfil</Link>
      <Link href="/contacto">Contacto</Link>
    </nav>
  );
}
```

Usar Link cuando:

- Navegación directa (menús, botones simples)
- No necesitas lógica antes de navegar
- SEO importante (los crawlers pueden seguir los enlaces)

Cuándo usar useRouter

```
"use client";
import { useRouter } from "next/navigation";

export default function FormularioLogin() {
  const router = useRouter();

  const manejarLogin = async () => {
    // 1. Validar datos
    const esValido = validarFormulario();

    if (esValido) {
      // 2. Enviar datos al servidor
      const exito = await enviarLogin();

      if (exito) {
        // 3. Solo entonces navegar
        router.push("/dashboard");
      }
    }
  };

  return <button onClick={manejarLogin}>Iniciar Sesión</button>;
}
```

Usar useRouter cuando:

- Navegación después de validaciones
 - Lógica compleja antes de navegar
 - Navegación condicional
 - Formularios y procesos multi-paso
-