

URL dinámicas

- Las URLs dinámicas (URI template) son usadas para obtener parte de ellas en un método de un controlador (parámetros).
- Una URL dinámica puede contener 1 o varios nombres de variables (parámetros con { }).
 - ✓ URL dinámica con un parámetro:
 - `http://localhost:8080/cineapp/detail/{id}`
 - **Se mandaría llamar como:** `http://localhost:8080/cineapp/detail/15`
 - ✓ URL dinámica con dos parámetros:
 - `http://localhost:8080/cineapp/detail/{id}/{fecha}`
 - **Se mandaría llamar como:** `http://localhost:8080/cineapp/detail/140/06-06-2017`
- Para vincular (binding) un parámetro de una URL dinámica a una variable en el controlador se utiliza la anotación `@PathVariable`

```
@RequestMapping(value = "/detail/{id}", method=RequestMethod.GET)
public String mostrarDetalle(@PathVariable("id") int idPelicula) {
    System.out.println("PathVariable: " + idPelicula);
    return "detalle";
}
```

- Un método puede tener cualquier número de anotaciones `@PathVariable`

```
public String mostrarDetalle(@PathVariable("id") int idPelicula, @PathVariable("fecha") Date fecha) {
```

URL dinámicas – Ejemplo

Controller

```
@RequestMapping(value = "/detail/{id}",method=RequestMethod.GET)
public String mostrarDetalle(@PathVariable("id") int idPelicula){
    System.out.println("PathVariable: " + idPelicula);
    return "detalle";
}
```

Link HTML (GET)

```
<a href="detail/${pelicula.id}">Consulta Horarios</a>
```

Generar URL dinámica

URL en el navegador (GET)

URL generada



Binding (vincular) Request Parameters

- Los parámetros de una petición HTTP pueden ser vinculados a una variable en un método en el controlador.
- Para vincular (binding) un parámetro de una petición se utiliza la anotación **@RequestParam** de Spring MVC. Se pueden vincular parámetros de una petición tipo GET y POST.

```
@RequestMapping(value = "/detalle", method=RequestMethod.GET)
public String verDetalle(@RequestParam("idMovie") int idPelicula){
    // Procesamiento del parámetro. Aquí, ya se hizo la conversión a String a int.
    System.out.println("RequestParam: " + idPelicula);
    return "someView";
}
```

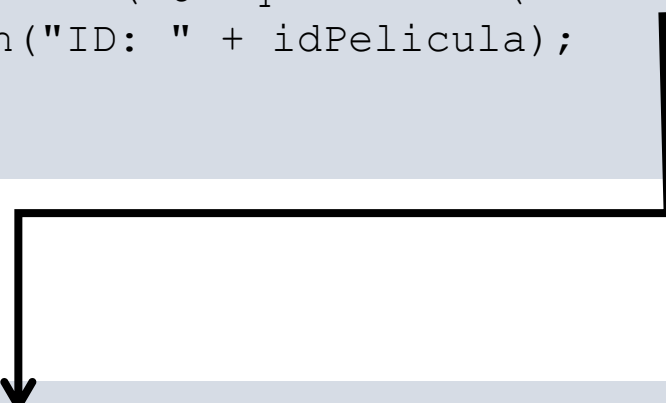
- Los parámetros usados con esta anotación son **REQUERIDOS** por default.
- Si se requiere que el parámetro sea opcional, se tiene que agregar el atributo "required" con el valor "false".
 - ✓ (ejemplo, @RequestParam(name="id", **required=false**)).
- Por default los parámetros de la petición llegan al servidor de tipo String (HTTP protocol basado en texto). Sin embargo, Spring MVC tiene un componente de conversión que convierte de String al tipo indicado en el método (int, double, date, etc.).

Binding (vincular) Request Parameters (GET)

Controller

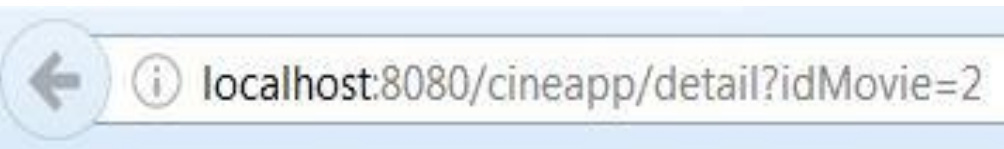
```
@RequestMapping(value = "/detalle",method=RequestMethod.GET)
public String verDetalle( @RequestParam("idMovie") int idPelicula){
    System.out.println("ID: " + idPelicula);
    return "detalle";
}
```

Link HTML (GET)



```
<a href="/detalle?idMovie=${pelicula.id}">Consulta Horarios</a>
```

URL en el navegador (GET)



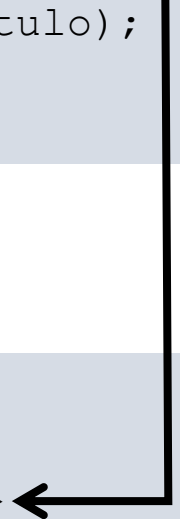
Binding (vincular) Request Parameters (POST)

Controller

```
@RequestMapping(value = "/save",method=RequestMethod.POST)
public String guardar( @RequestParam("titulo") String titulo ){
    System.out.println("Titulo:" + titulo);
    return "detalle";
}
```

FORM HTML (POST)

```
<form action="save" method="POST">
    Titulo
    <input type="text" name="titulo"/>
    <button type="submit" >Guardar</button>
</form>
```



Variantes de @RequestMapping

- Apartir de la versión 4.3 Spring agregó variaciones de la anotacion @RequestMapping

Adecuadas para aplicaciones web estándar:

- ✓ @GetMapping
- ✓ @PostMapping

Adecuadas para REST Web Services:

- ✓ @PutMapping
- ✓ @DeleteMapping
- ✓ @PatchMapping

- Solo pueden ser usadas a nivel de método.
- Ayudan a simplificar el mapeo para los métodos HTTP.
- Semántica más expresiva para cada método.

Variantes de @RequestMapping - Ejemplos

```
@Controller
public class NoticiasController {

    //@RequestMapping(value="/create",method=RequestMethod.GET)
    @GetMapping(value="/create")
    public String crear() {
        return "noticias/formNoticia";
    }

    //@RequestMapping(value="/save",method=RequestMethod.POST)
    @PostMapping(value="/save")
    public String guardar(@RequestParam("titulo") String titulo) {
        System.out.println("Guardando : " + titulo);
        return "noticias/formNoticia";
    }

    //@RequestMapping(value="/update/{id}",method=RequestMethod.GET)
    @GetMapping(value="/update/{id}")
    public String actualizar(@PathVariable("id") int idNoticia) {
        System.out.println("Actualizando: " + idNoticia);
        return "noticias/formNoticia";
    }
}
```

Anotación @RequestMapping – A nivel de la Clase

```
@Controller
@RequestMapping(value="/noticias")
public class NoticiasController {

    @GetMapping(value="/create")
    public String crear() {
        return "noticias/formNoticia";
    }

    @PostMapping(value="/save")
    public String guardar(@RequestParam("titulo") String titulo) {
        return "noticias/formNoticia";
    }
}
```



localhost:8080/cineapp/noticias/create

contextPath

Formulario HTML

```
<spring:url value="/" var="urlRoot" />
<form action="${urlRoot}noticias/save" method="post">
    Titulo de la noticia
    <input type="text" name="titulo">
    <input type="submit" value="Guardar">
</form>
```

Al renderizarse la vista, la URL del atributo action del formulario HTML, se convierte a:

action="/cineapp/noticias/save"

TinyMCE – Editor web (HTML-Javascript)

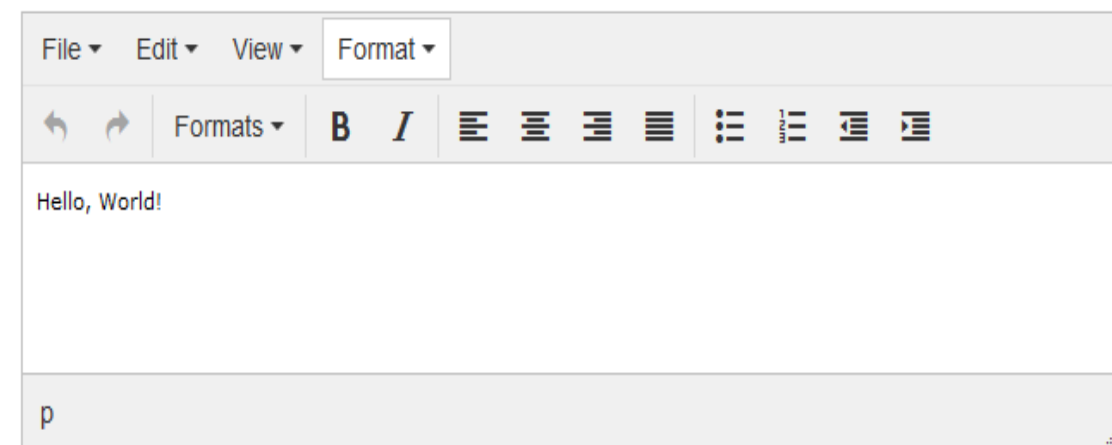
- Editor HTML basado en Javascript y HTML (independiente de la plataforma).
- Fácilmente convierte un elemento <textarea> en un editor HTML completamente funcional.
- Existe una versión Community y una versión de paga con más funcionalidad.
- Dos tipos de instalación (<https://www.tinymce.com>)
 - ✓ Vía CDN (Content Delivery Network). Se necesita Internet.
 - ✓ Descargar los archivos CSS, Javascript (esta forma la usaremos en el curso).

EJEMPLO CODIGO HTML

```
<!DOCTYPE html>
<html>
<head>
  <script src='https://cloud.tinymce.com/stable/tinymce.min.js'></script>
  <script>
    tinyMce.init({ selector: '#mytextarea' });
  </script>
</head>
<body>
<h1>TinyMCE Quick Start Guide</h1>
  <form method="post">
    <textarea id="mytextarea">Hello, World!</textarea>
  </form>
</body>
</html>
```

RESULTADO EN EL NAVEGADOR

TinyMCE Quick Start Guide



TinyMCE – Formulario de creación de Noticias/Novedades

➤ El archivo formNoticia.html de la plantilla HTML ya tiene incluido el formulario HTML.

Formulario HTML – Crear Noticia

Datos de la Noticia

Título

Julia Roberts protagonizara. The Bookseller

Estatus

Activa

Detalles

FileEditViewFormatTableTools

B

I

A

A

La novela de Cynthia Swanson **The Bookseller** será llevada al cine con **Julia Roberts** (**Los Pitufos: La aldea Escondida**) como protagonista.

La historia está ambientada en los sesenta y su protagonista es una mujer soltera, Kitty Miller, que lleva una librería. Sueña con una vida alternativa en la que ha encontrado el amor y está casada y con hijos, pero la línea que separa realidad y ficción comienza a estar demasiado dispersa para que la distinga.

Según informa **Moviehole** Roberts también producirá la película junto a Lisa Gillan y Marisa Yeres Hill.

Powered by TinyMCE

Guardar

La noticia renderizada en la sección Noticias y Novedades en la página principal.

El formato se conserva.