

0101010  
0100101  
1101010

# UD10.4- PASO DE INFORMACIÓN EN LAS REDIRECCIONES

0485 Programación - 1er DAW/DAM

# CONTENIDOS

---

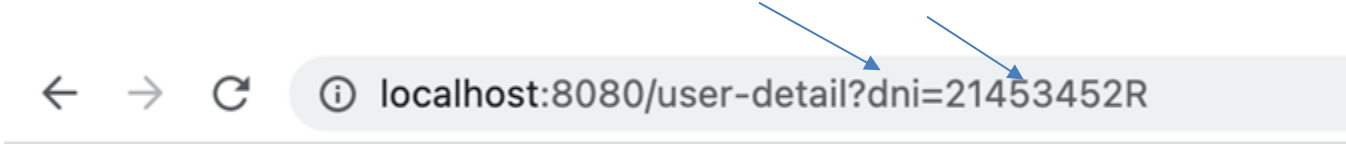
- Introducción
- El objeto **RedirectAttributes**
  - **Caso I.** Mostrar detalle de tarea tras su creación.
  - **Caso II.** Mensajes. De confirmación y de error.

# 1. INTRODUCCIÓN

---

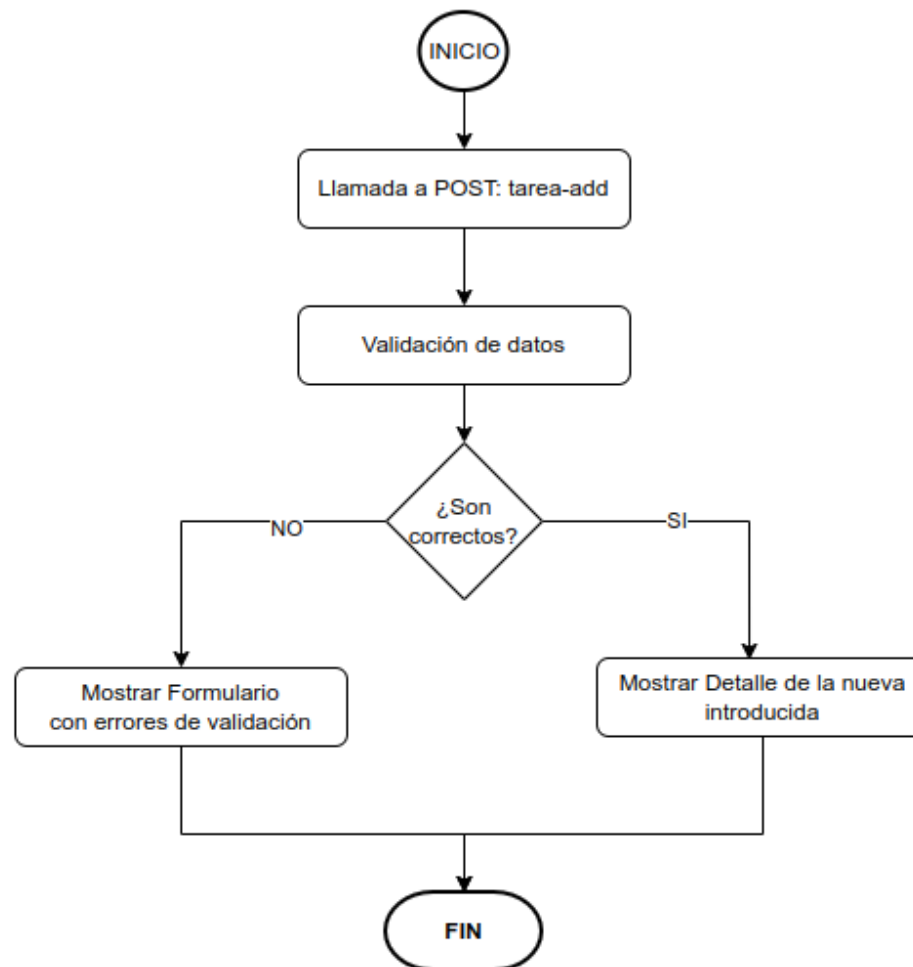
- En determinadas ocasiones, cuando **llevamos a cabo una redirección**, se hace necesario **pasar información al nuevo controlador**.
  - **Errores producidos**, al **validar un formulario** de introducción de datos.
  - **Mensajes de confirmación** sobre si una tarea se ha realizado correctamente.
  - Pasar los **datos** que contenía el **formulario** antes de que se produjera el **error de validación**.

## 2. EL OBJETO RedirectAttributes

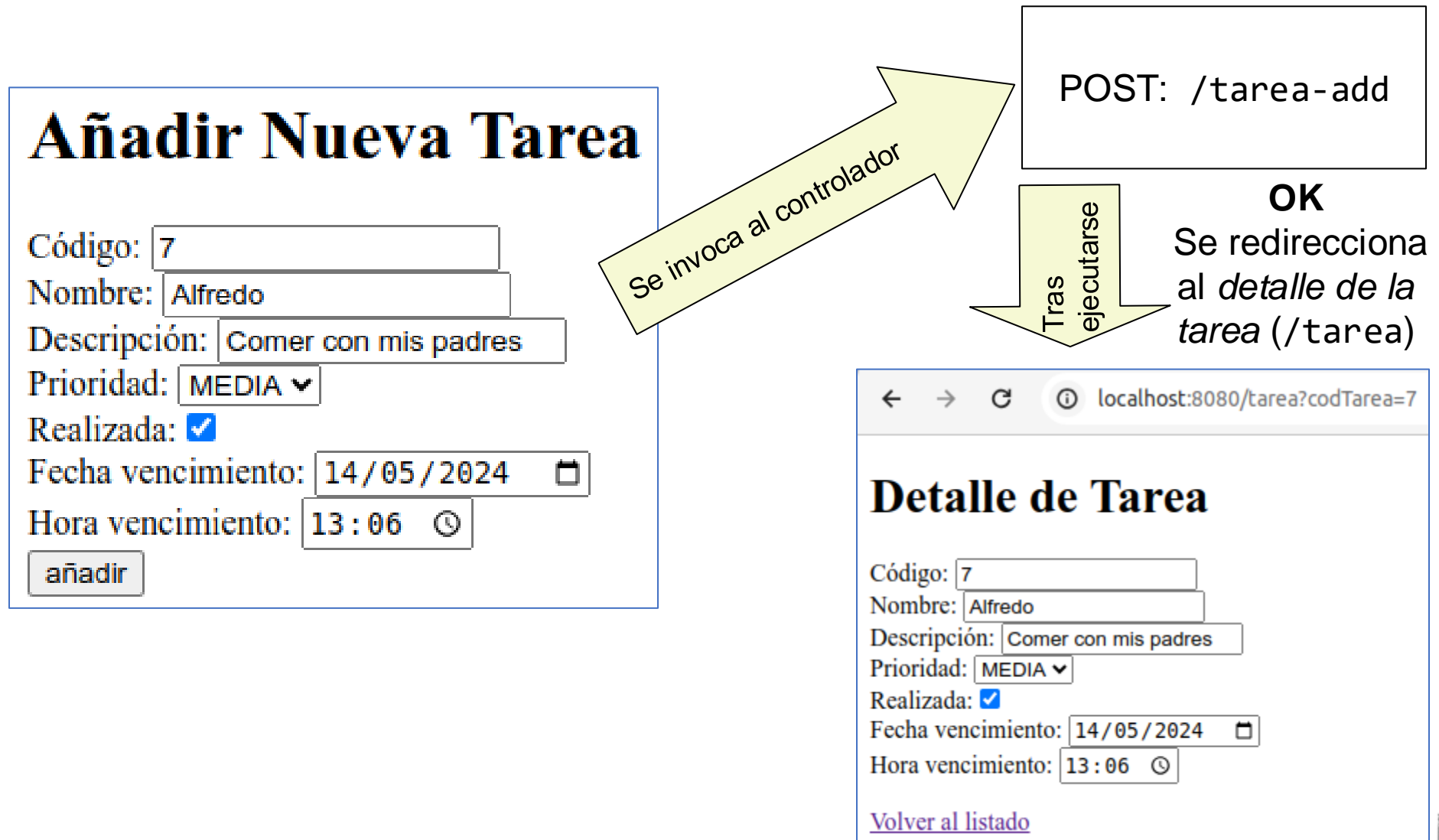
- Para **pasar parámetros** (información) en las **redirecciones** necesitamos acceder al objeto **RedirectAttributes**.
- Éste dispone de diferentes métodos:
  - `redirectAttributes.addAttribute(String key, Object value)`: Los parámetros viajan en la url en forma de **key=value**
  - `redirectAttributes.addFlashAttribute(String key, Object Value)`: Los parámetros se almacenan en memoria en la **sesión del usuario** y permanecerán ahí hasta que el navegador **acceda a la url especificada en la redirección**.

## 2.1 EJEMPLO GESTOR DE TAREAS

- Caso de uso "Introducir una tarea": La inserción de una tarea seguirá el siguiente flujo:



## 2.2 CASO I: MOSTRAR DETALLE DE LA TAREA (EN UN FORMULARIO) TRAS SU CREACIÓN



## 2.2 CASO I: MOSTRAR DETALLE DE LA TAREA (EN UN FORMULARIO) TRAS LA INTRODUCCIÓN

```
@GetMapping(value = "/tarea-form")
public String tareaFormActionView() {
    //cargamos el formulario
    return "user-add-form";
}

@PostMapping(value = "/tarea-add")
public String postAddAction(@RequestParam Map<String, String> params) {

    // Procesamos los datos y guardamos la tarea en el repositorio
    // Si todo es correcto redirigimos al usuario para que se muestre el
    detalle de la nueva tarea creada

    return "redirect:/tarea";
}
```

¿Necesitamos indicar al controlador  
/tarea el **código** de la tarea del que  
queremos visualizar el detalle?

## 2.2 CASO I: MOSTRAR DETALLE DE LA TAREA (EN UN FORMULARIO) TRAS LA INTRODUCCIÓN

```
@PostMapping(value = "/tarea-add")  
public String postAddAction(@RequestParam Map<String, String> params, RedirectAttributes  
redirectAttributes) {
```

```
    int code = Integer.parseInt(params.get("code"));  
    String user = params.get("user");
```

```
    // Recogemos los valores de la tarea
```

```
    .....
```

```
    // Creamos la tarea y la insertamos en el repositorio
```

```
    redirectAttributes.addAttribute("codTarea", code);
```

```
    return "redirect:/tarea";
```

```
}
```

**1. Indicamos al framework que necesitamos acceder al objeto RedirectAttributes**

**2. Añadimos los parámetros en forma de key-value. Éstos viajarán en la url del navegador**



← → ↻ ⓘ localhost:8080/tarea?codTarea=7



## 2.3 CASO II. MENSAJES DE CONFIRMACIÓN Y ERROR

Mensaje de confirmación

Tarea añadida con éxito

### Detalle de Tarea

Código:

Nombre:

Descripción:

Prioridad:

Realizada: ☐

Fecha vencimiento:

Hora vencimiento:

[Volver al listado](#)

Mensaje de error con campos incorrectos

**codigo:** Debe ser un valor numérico de máximo 3 dígitos y no estar vacío

**usuario:** El usuario debe empezar por mayúscula y no puede contener caracteres no alfabéticos o estar vacío

### Añadir Nueva Tarea

Código:

Nombre:

Descripción:

Prioridad:

Realizada: ☐

Fecha vencimiento:

Hora vencimiento:

## 2.3 CASO II. MENSAJES DE CONFIRMACIÓN Y ERROR

```
@GetMapping(value = "/tarea-form")
public String addUserFormAction() {
    //cargamos el formulario
    return "tarea-form-view";
}

@PostMapping(value = "/tarea-add")
public String postAddAction(@RequestParam Map<String, String> params,
    RedirectAttributes redirectAttributes) {
    // Procesamos los datos y guardamos la tarea en el repositorio teniendo en cuenta que...
    // - si se produce algun error redirigimos al formulario (vacío) mostrando los errores
    return "redirect:/tarea-form";

    // - si todo va bien lo redirigimos a la vista de detalle mostrando un mensaje de
    // confirmación
    redirectAttributes.addAttribute("codTarea", code);
    return "redirect:/tarea";
}
```

¿Cómo pasamos los errores detectados en la redirección?

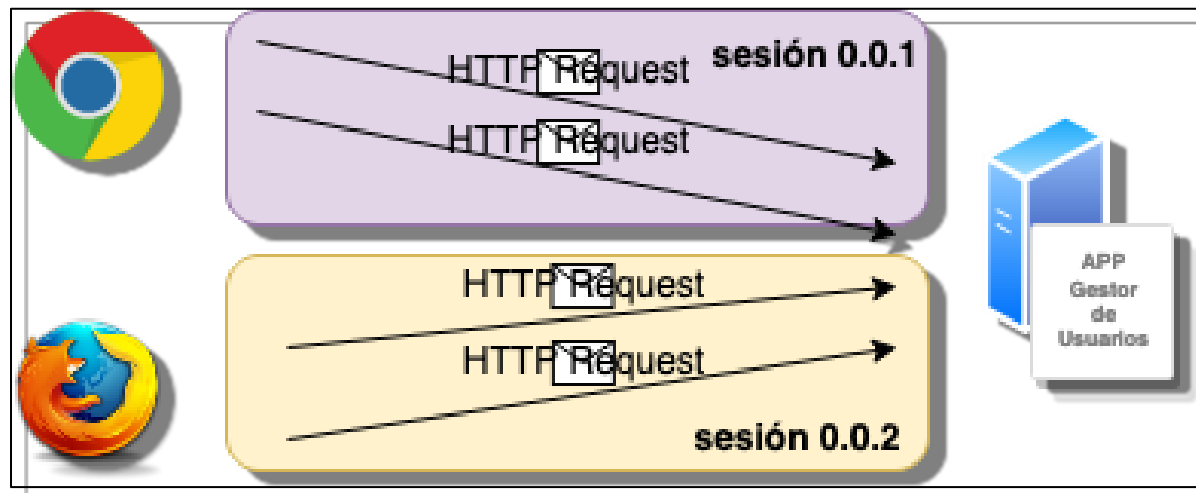
¿Cómo pasamos mensajes de confirmación en la redirección?



## 2.3 CASO II. MENSAJES DE CONFIRMACIÓN Y ERROR

### Atributos Flash

- Permite **guardar información** (objetos) en memoria que **estarán disponibles SÓLO en la ejecución del siguiente controlador**:
  - ✓ La **información se guarda** en lo que conocemos **como sesiones**.
  - ✓ **Se establecen** entre el **servidor** (máquina que ejecuta la aplicación) y el **cliente** (navegador) que lleva a cabo la petición.
  - ✓ Si acceden a la aplicación **2 clientes /navegadores** distintos cada uno **establece su propia sesión** con datos independientes.



## 2.3.1 CASO II. MENSAJES DE ERROR

```
@PostMapping(value = "/tarea-add")
public String postAddAction(@RequestParam Map<String, String> params, RedirectAttributes redirectAttributes) {
    ....
    HashMap<String, String> errors = new HashMap<>();
    if (!Validator.isValidNumberOfLength(code, 3)) {
        errors.put("código", "Debe ser un código numérico de máximo 3 dígitos");
    }
    if (!Validator.isValidText(user)) {
        errors.put("usuario", "El usuario no puede contener caracteres no alfabéticos");
    }
    if (errors.size() > 0) {
        redirectAttributes.addFlashAttribute("errors", errors);
        return "redirect:/tarea-form";
    }
    ...

    redirectAttributes.addAttribute("codTarea", code);
    return "redirect:/tarea";
}
```

El método **addFlashAttribute** mantendrá el objeto con los errores en memoria **hasta que se ejecute un controlador**.

## 2.3.1 CASO II. MENSAJES DE ERROR

tarea-form-view.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Gestor Tareas</title>
    <link rel="stylesheet" href="/main.css">
  </head>
  <body>
    <div class="errors" th:if="${errors}">
      <p th:each="item: ${errors}">
        <strong th:text="${item.key}"></strong>
        <span th:text="${item.value}"></span>
      </p>
    </div>
    <h1>Añadir Nueva Tarea</h1>
```

...

Si existe la  
variable **errors**, los  
mostramos.

## 2.3.1 CASO II. MENSAJES DE ERROR

### Añadir Nueva Tarea

Código:

Nombre:

Descripción:

Prioridad:

Realizada: ☒

Fecha vencimiento:

Hora vencimiento:

**codigo:** Debe ser un valor numérico de máximo 3 dígitos y no estar vacío

**usuario:** El usuario debe empezar por mayúscula y no puede contener caracteres no alfabéticos o estar vacío

### Añadir Nueva Tarea

Código:

Nombre:

Descripción:

Prioridad:

Realizada: ☐

Fecha vencimiento:

Hora vencimiento:

Mensaje  
campos incorrectos

## 2.3.2 CASO II. MENSAJES DE CONFIRMACIÓN

```
@PostMapping(value = "/tarea-add")
public String postAddAction(@RequestParam Map<String, String> params, RedirectAttributes redirectAttributes) {
    ....
    HashMap<String, String> errors = new HashMap<>();
    if (!Validator.isValidNumberOfLength(code, 3)) {
        errors.put("codigo", "Debe ser un código numérico de máximo 3 dígitos");
    }
    if (!Validator.isValidText(user)) {
        errors.put("usuario", "El usuario no puede contener caracteres no alfabéticos");
    }
    if (errors.size() > 0) {
        redirectAttributes.addFlashAttribute("errors", errors);
        return "redirect:/tarea-form";
    }
    ...

    redirectAttributes.addFlashAttribute("infoMessage", "Usuario insertado con éxito");

    redirectAttributes.addAttribute("codTarea", code);
    return "redirect:/tarea";
}
```

El método `addFlashAttribute` mantendrá el objeto `infoMessage` en memoria hasta que se ejecute un controlador.

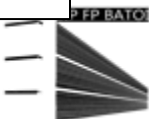
## 2.3.2 CASO II. MENSAJES DE CONFIRMACIÓN

tarea-details-view.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="/main.css">
    <title>Gestor Tareas</title>
  </head>
  <body>
    <div class="info" th:if="${infoMessage}">
      <p th:text="${infoMessage}"></p>
    </div>
    <h1>Detalle de Tarea</h1>
```

Si existe la variable **infoMessage**  
mostramos su contenido

...





## 2.3 CASO II. MENSAJES DE CONFIRMACIÓN

Mensaje de confirmación

Tarea añadida con éxito

### Detalle de Tarea

Código:

Nombre:

Descripción:

Prioridad:

Realizada: ☐

Fecha vencimiento:

Hora vencimiento:

[Volver al listado](#)

# ACTIVIDAD PREVIA

---

- A partir del ejemplo completo del **gestor de tareas** (todo-list) con gestión de **mensajes de error** y de **confirmación**, con validación de ciertos campos enviados en el formulario (utilizando expresiones regulares a través de la clase `Validator`) y todo lo realizado hasta el momento:  
  
-> Analiza su código y realiza la actividad entregable de **gestión de usuarios** que se propone a continuación.

Eso es todo... de momento :-)