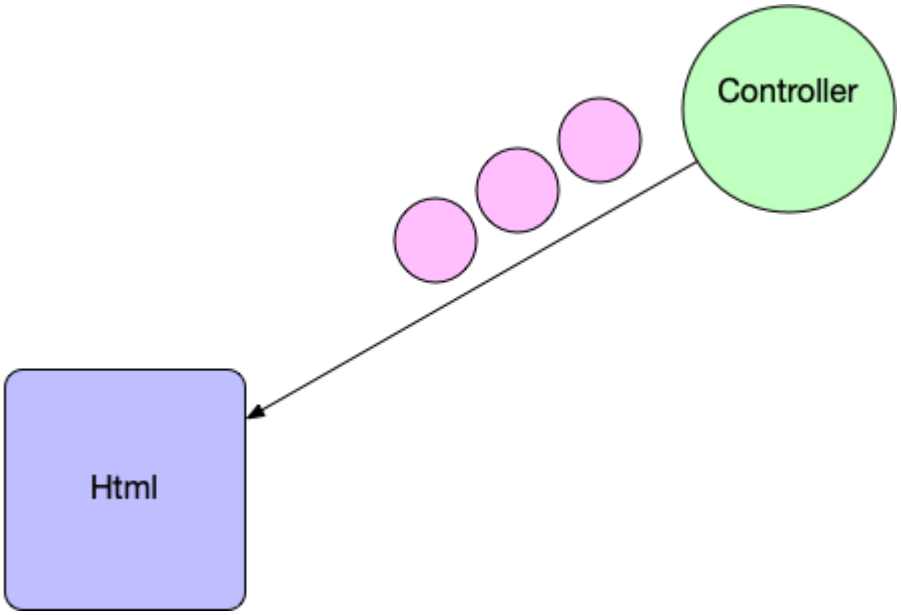
El uso de Thymeleaf for each es algo de lo más habitual cuando trabajamos con el motor de plantillas . Thymeleaf se encarga de renderizar páginas html con la información que obtenemos del servidor y permitir una integración natural entre los datos del modelo MVC y las páginas que soportan HTML5.



Vamos a ver unos ejemplos sencillos que usen este motor de plantillas. Para ello nos vamos a apoyar en un controlador que se encargue de pasar información a las plantillas.

```
@Controller
public class ProductoController {
        @RequestMapping("/hola")
        public String hola(Model modelo) {
        modelo.addAttribute("mensaje","hola desde spring mvc");
        return "hola";
        }
}
```

Este es el primer caso en el cual tenemos una url de /hola y a esta url mapeamos un método que nos redirija a una plantilla adjuntando un mensaje con un contenido:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<p th:text="${mensaje}"></p>

</body>
</html>
```

El resultado es que la variable ${mensaje} será substituida por la información que esta asociada ella en el servidor:



hola desde spring mvc

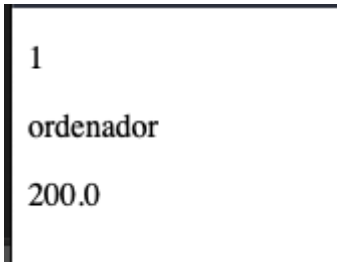
ThymeLeaf y Objetos

De la misma forma que podemos imprimir el valor de una variable no tenemos problema para gestionar también el uso de objetos e imprimir en una pagina HTML su contenido:

```
@RequestMapping("/objeto")
public String objeto(Model modelo) {
modelo.addAttribute("objeto",new Producto(1,"ordenador",200));
return "objeto";
}
```

En este caso lo que añadimos es un objeto completo . Es momento de ver como imprimir esa información a nivel de HTML:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<p th:text="${objeto.numero}"></p>
<p th:text="${objeto.concepto}"></p>
<p th:text="${objeto.importe}"></p>
</body>
</html>
```



ThymeLeaf For Each y listas de Objetos

Ya solo nos queda ver cómo gestionar una lista de objetos e imprimirla en una tabla . En este caso la operativa es un poco más compleja.

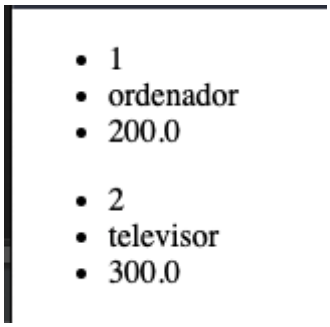
```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
```

```
<ul  th:each="producto : ${lista}" >

<li th:text="${producto.numero}"></li>
<li th:text="${producto.concepto}"></li>
<li th:text="${producto.importe}"></li>
</ul>

</body>
</html>
```

Como podemos observar tenemos que usar un Thymeleaf for each que se coloca en la etiqueta padre y permite recorrer una lista y acceder al valor de cada objeto y sus propiedades . El resultado lo podemos ver aquí :



El motor de plantillas de Thymeleaf permite una integración sencilla de y natural de un Modelo MVC a través del uso de atributos data.

Otros artículos relacionados

- Thymeleaf , un motor de plantillas
- Spring MVC Configuración
- Spring MVC static resources y su configuración
- Spring MVC Flash Attributes
- Spring MVC @RequestMapping