Explications liées au programme Test1

Architecture MVC : Model-View-Controller

@SpringBootApplication
public class DemoTest1Application {
public static void main(String[] args) {
SpringApplication.run(DemoTest1Application.class, args);
}
}
	Le point de démarrage de l'application SpringApplication :la classe de démarrage d'une application Spring et qui va créer une instance de la classe **ApplicationContext**

ApplicationContext : l'interface centrale d'une application Spring permettant de fournir des informations de configuration `a l'application.

@SpringBootApplication : englobe les 3 annotations suivantes :

- **@Configuration** : fait partie du noyau de Spring Framework et indique que la classe annotée peut contenir des m´méthodes annotées par @Bean. Ainsi,Spring Container peut traiter la classe et générer des beans qui seront utilisés par l'application.
- **@EnableAutoConfiguration** : permet, au d´démarrage de Spring, de générer automatiquement les configurations n´nécessaires en fonction des d´dépendances ajoutées.
- **@ComponentScan** : permet de scanner les package contenant des composants

@Controller
public class HomeController {
 	
@GetMapping("/test")
public String home()
{ // code spécifique ....
return "home"; }
 	
@GetMapping("/test2")
public String home2()
{return "home2";}
}
 	
Par défaut, une méthode du contrôleur annotée par **@VerbMapping("path")** retourne le nom de la page html qui sera renvoyée au client
	Un contrôleur est une classe annotée par **@Controller** qui indique à Spring qu'il doit gérer cette classe (on ne doit pas l'instancier explicitement). C'est un des composants d'une architecture MVC.
	
	Un contrôleur Spring reçoit une requête du contrôleur frontal dont le rôle est d'aiguiller les requêtes émises par les utilisateurs d'une application vers les contrôleurs applicatifs adéquats.
	
	On y trouve principalement des méthodes annotées par **@VerbMapping("path")** où Verb correspond aux verbes du protocole http : **Get**, **Post**, **Put**, **Delete** et **Patch.**
	
	Spring exécutera la méthode correspondant à la requête du client qui « matche » avec l'annotation utilisée.

<html> <head> <title>TODO supply a title</title> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> </head> <body> Bonjour , bienvenue il est </body> </html>	Page home.html qui utilise le moteur de templates Thymeleaf

@Controller
public class WelcomeController {
 	
@GetMapping("/welcome")
public String welcome**(@RequestParam String name**,
Model model)
{ **model.addAttribute("nom",name);**
return "welcome";
}
}
 	
Requête : **http://ad_IP:8080/welcome?name= Severs**
	Ce contrôleur contient une méthode appelée par une requête Get qui possède un paramètre de requête nommé name.
	
	Ce paramètre est stocké comme attribut appelé « nom » dans le modèle qui sera utilisé par la vue appelée

<html> <head> <title>TODO supply a title</title> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> </head> <body> Bonjour **** </body> </html>	**${nom}** est une expression langage Spring. Si Spring trouve un attribut nommé « nom », il utilise la valeur de l'attribut trouvé.