스프링입문-1 (김영한)

2021년 7월 29일 목요일 오후 5:19

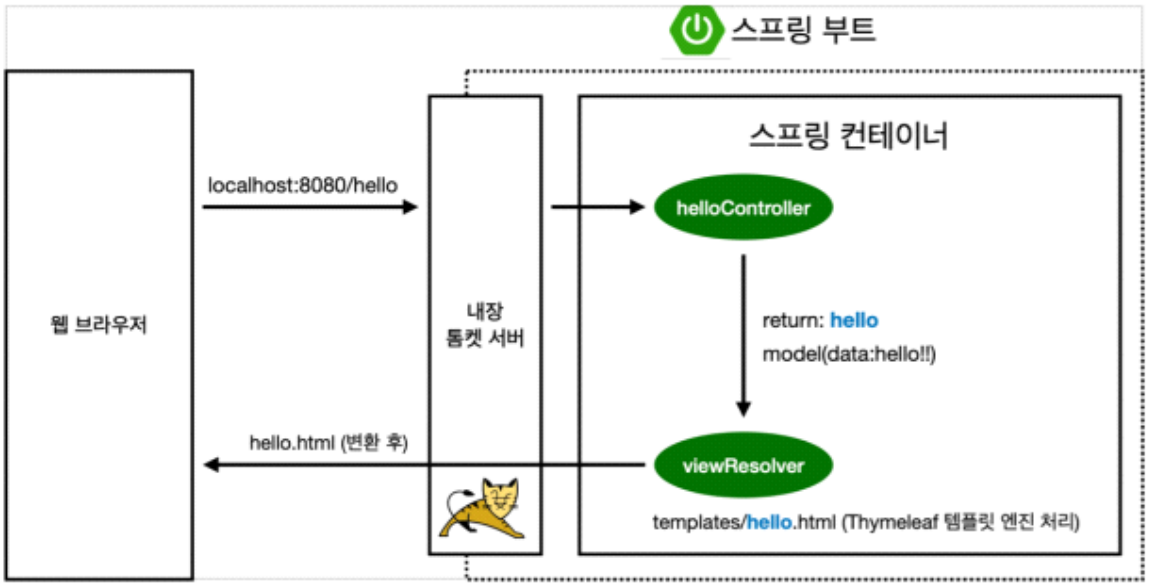
resources/static/ 에 index.html 만들면 welcome page로 인식하고 띄워줌

localhost:8080

템플릿 엔진을 이용해서 모양이나 기능 수정 추가 가능
템플릿 엔진 종류 대표적으로 (FreeMarker, Groovy, Thymeleaf, Mustache)

Thymeleaf의 장점 : html을 그대로 쓰고 서버없이 바로 열어봐도 볼 수 있음

동작 환경 그림



컨트롤러에서 리턴 값으로 문자를 반환하면 viewResolver 가 화면을 찾아서 처리한다
resource:templates/{ViewName}+.html

예시에서는 hello 리턴 받고 hello.html 찾아서 출력
+ model 로 {data} 부분 hello!!로 치환해주는 작업까지 해줌

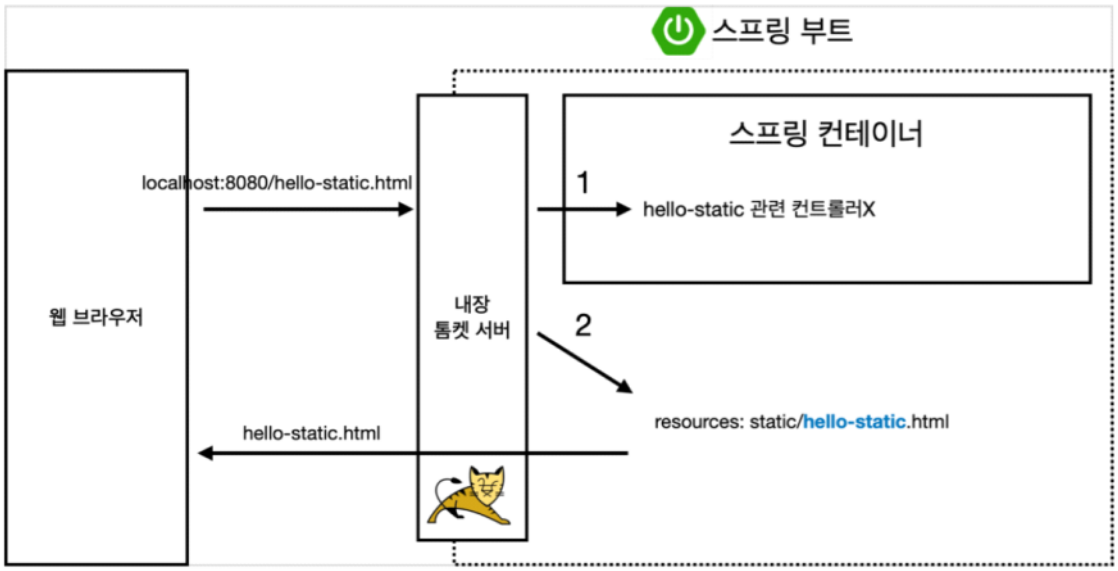
spring-boot-devtools 라이브러리 추가하면, html파일 컴파일시 서버 재시작 없이 변경사항 반영

스프링 웹개발 기초

- 정적 컨텐츠 (서버에서 하는 것 없이 웹브라우저에 그대로 전달)
- MVC(Model, View, Control) 와 템플릿 엔진(jsp, php 같이 html 을 그냥 주는게 아니라 서버에서 프로그래밍하여 동적으로 바꿔서 전달)

- API (Json 이라는 데이터 구조 포멧으로 데이터 전달하는 방식)

정적 컨텐츠

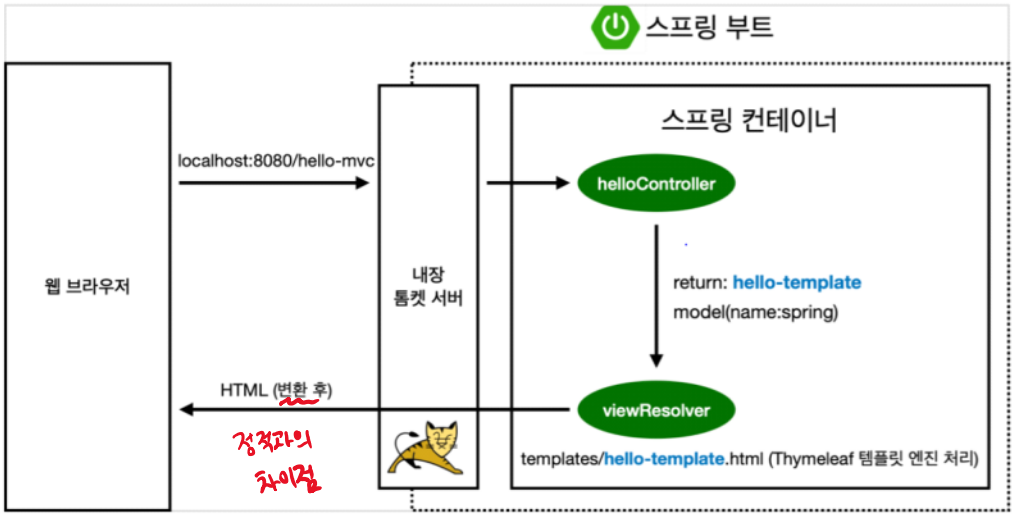


1. spring이 hello-static.html 요청 받음
2. hello-static.html 을 컨트롤에서 찾아봄
3. 없으면 resources 에서 찾아봄
4. 있으면 반환

MVC : Model, View, Controller

View 는 화면을 그리는데 모든 역량 집중
Model 과 Controller 는 비즈니스 모델,내부적인 것 처리하는데 집중

MVC, 템플릿 엔진 이미지



viewResolver : 뷰를 찾아주고 템플릿 엔진 연결시켜줌

ex)

```
@GetMapping("hello-mvc")
public String helloMvc(@RequestParam("name") String name, Model model){
    //매개변수 받을 때 @RequestParam 사용
    model.addAttribute( attributeName: "name",name );
    return "hello-template";
}
```

API :

ex)

```
@GetMapping("hello-string")
@ResponseBody //http 의 통신 프로토콜의 body 부분에 직접 넣어주겠다는 의미
//view 거치지 않고 직접 데이터 전달
public String helloString(@RequestParam("name") String name){
    //매개변수 받을 때 @RequestParam 사용
    return "hello "+name;
}
```

@ResponseBody 사용하여 viewResolver 사용X
대신 Http의 body에 직접 반환

```java
@GetMapping("hello-api")
@ResponseBody //http 의 통신 프로토콜의 body 부분에 직접 넣어주겠다는 의미
public Hello helloApi(@RequestParam("name") String name){
    //매개변수 받을 때 @RequestParam 사용
    Hello hello = new Hello();
    hello.setName(name);
    return hello;
}
```

@ResponseBody 사용하고, 객체 반환하면 객체가 JSON으로 변환됨
(문자, 객체, byte 등 여러 타입에 따라 어떻게 변환할지 미리 정해져있음)