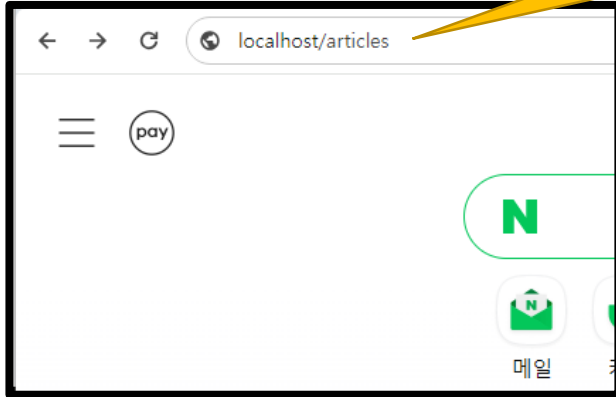


R : read : 전체조회

클라이언트(브라우저)

브라우저에 입력하는것은
HTTP GET 방식입니다.



HTTP GET
localhost/articles
요청

서버(스프링)

```
@GetMapping("/articles")
public String index( Model model ){
    // 1. p.175 DAO에게 요청해서 데이터 가져온다.
    List<ArticleForm> result = articleDao.index();
    // 2. p.175 뷰템플릿(머스테치)에게 전달할 값을 model 담아준다.
    model.addAttribute( attributeName: "articleList", result);
    // 3. p.175 뷰 페이지 설정
    return "articles/index";
}
```

머스테치
model.addAttribute("articleList", result);

[View (HTML) 렌더링]
머스테치를 HTML 만드는과정(자동)

Index.mustache

HTTP GET
localhost/articles
응답

localhost/articles

헤더(메인메뉴)
글쓰기(New Article)

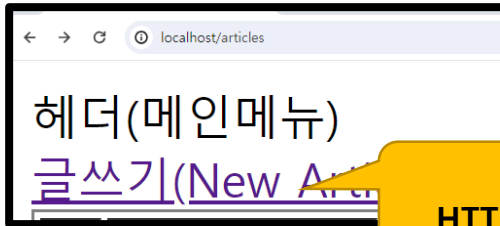
ID	Title	Content
2	<u>안녕하세요</u>	하하하하
3	<u>안녕하세요2</u>	하하하

푸터(홈페이지 정보)

C : create

클라이언트(브라우저)

서버(스프링)

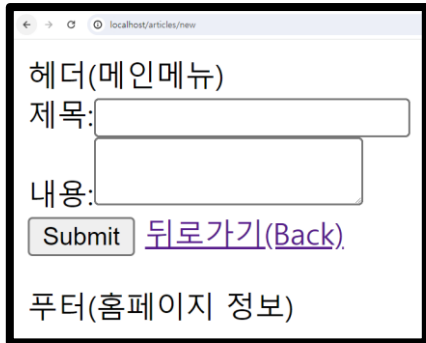


`글쓰기(New Article)`

`<a>` 클릭은
HTTP GET 방식입니다.

HTTP GET
localhost/articles/new
요청

```
@GetMapping("/articles/new") // HTTP 요청 경로 : GET  
public String newArticleForm(){  
    return "articles/new"; // 리다이렉트  
} // m end
```

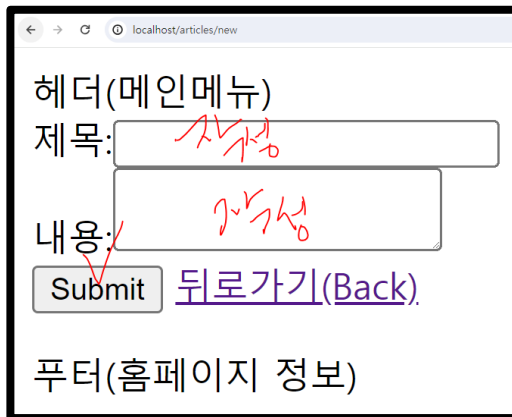


HTTP GET
localhost/articles/new
응답

머스태치

[View (HTML) 렌더링]
머스태치를 HTML 만드느과정(자동)

new.mustache



`<form action="/articles/create" method="post">`

HTTP POST
localhost/articles/create
요청

```
@AllArgsConstructor // 컴파일  
@NoArgsConstructor // 컴파일  
@ToString // 컴파일  
@Getter@Setter // 컴파일  
public class ArticleForm {  
    // 1. 필드  
    private long id;  
    private String title;  
    private String content;  
}
```

```
@PostMapping("/articles/create") // HTTP 요청 경로 : POST;  
public String createArticle( ArticleForm form ){  
    // DAO에게 요청하고 응답 받기.  
    ArticleForm saved = articleDao.createArticle( form );  
    return "redirect:/articles/"+saved.getId() ;  
} // m end
```

리다이렉트

제목:`<input name="title" />` `
`
내용:`<textarea name="content">` `</textarea>` `
`

R : Read 개별조회

클라이언트(브라우저)

서버(스프링)

리다이렉트

```
@GetMapping("/articles/{id}") // 클라이언트 요청 예시 : /articles/  
public String show(@PathVariable Long id , Model model ){  
    ArticleForm form = articleDao.show( id );  
    model.addAttribute( attributeName: "article" , form );  
    model.addAttribute( attributeName: "name" , attributeValue: "유재석" );  
    return "articles/show";  
}
```

머스टे치

model.addAttribute("article", form);

[View (HTML) 렌더링]
머스टे치를 HTML 만드느과정(자동)

show.mustache

HTTP GET
localhost/articles/4
응답

localhost/articles/4

헤더(메인메뉴)

[목록보기\(Go to Article List\)](#)

[수정\(edit\)](#)

[삭제>Delete\)](#)

ID	Title	Content
4	ddsD	ASDASD

푸터(홈페이지 정보)

U : update

클라이언트(브라우저)

서버(스프링)

localhost/articles/4

수정(edit)

헤더(메인메뉴)

목록보기(Go to Article List)

수정(edit)

HTTP GET
localhost/articles/4/edit
요청

```
@GetMapping("/articles/{id}/edit") // GetMapping이름: <a> 이:  
public String edit( @PathVariable long id , Model model ){  
    System.out.println("id = " + id);  
    // 1. DAO에게 요청하고 응답 받는다.  
    ArticleForm form = articleDao.findById( id );  
    // 2. 응답결과를 뷰 템플릿에게 보낼 준비 model  
    model.addAttribute( "article" , form );  
    // 3. 뷰 페이지 설정  
    return "articles/edit";  
}
```

머스테치
model.addAttribute("article", form);

[View (HTML) 렌더링]
머스테치를 HTML 만드는과정(자동)

edit.mustache

localhost/articles/4/edit

헤더(메인메뉴)

제목: ddsD

ASDASD

내용:

Submit 뒤로가기(Back)

푸터(홈페이지 정보)

HTTP GET
localhost/articles/4/edit
응답

localhost/articles/4/edit

헤더(메인메뉴)

제목: ddsD수정

ASDASD수정

내용:

Submit 뒤로가기(Back)

푸터(홈페이지 정보)

<form action="/article/update" method="post">

HTTP POST
localhost/articles/update
요청

```
@PostMapping("/article/update") // @PatchMapping @Put  
public String update( ArticleForm form ){  
    // * form 입력 데이터를 Dto 매개변수로 받을때  
    // 1. form 입력상자의 name 과 Dto의 필드명 동일  
    // 2. Dto의 필드 값을 저장할 생성자 필요  
    System.out.println("form = " + form);  
    // 2.DAO에게 요청하고 응답받기  
    ArticleForm updated = articleDao.update( form );  
    // 3. 수정 처리된 상세페이지로 이동  
    return "redirect:/articles/" + updated.getId();  
}
```

리다이렉트

<input name="id" type="hidden" value="{{id}}"/>
제목:<input name="title" value="{{ title }}" />

내용:<textarea name="content" >{{ content }}</textarea>


```
@AllArgsConstructor // 컴파일  
@NoArgsConstructor // 컴파일  
@ToString // 컴파일  
@Getter@Setter // 컴파일  
public class ArticleForm {  
    // 1. 필드  
    private long id;  
    private String title; //  
    private String content;
```

R : Read 개별조회

클라이언트(브라우저)

서버(스프링)

리다이렉트

```
@GetMapping("/articles/{id}") // 클라이언트 요청 예시 : /articles/  
public String show(@PathVariable Long id , Model model ){  
    ArticleForm form = articleDao.show( id );  
    model.addAttribute( attributeName: "article" , form );  
    model.addAttribute( attributeName: "name" , attributeValue: "유재석" );  
    return "articles/show";  
}
```

머스टे치

model.addAttribute("article", form);

[View (HTML) 렌더링]
머스टे치를 HTML 만드느과정(자동)

show.mustache

HTTP GET
localhost/articles/4
응답

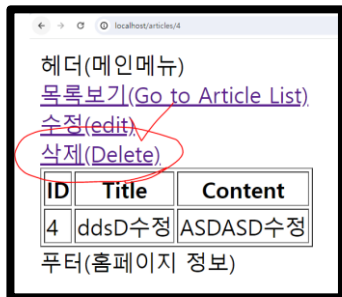
헤더(메인메뉴)
[목록보기\(Go to Article List\)](#)
[수정\(edit\)](#)
[삭제>Delete\)](#)

ID	Title	Content
4	ddsD	ASDASD

푸터(홈페이지 정보)

D : delete

클라이언트(브라우저)



삭제(Delete)

서버(스프링)

HTTP GET
localhost/articles/4/delete
요청

```
@GetMapping("/articles/{id}/delete")
public String delete(@PathVariable long id ){
    System.out.println("id = " + id);
    // 1. 삭제할 대상
    // 2. Dao 삭제 요청하고 응답받기
    boolean result = articleDao.delete( id );
    // 3. 결과 페이지로 리다이렉트 하기.
    return "redirect:/articles";
}
```

리다이렉트

```
@GetMapping("/articles")
public String index( Model model ){
    // 1. p.175 DAO에게 요청해서 데이터 가져온다.
    List<ArticleForm> result = articleDao.index();
    // 2. p.175 뷰템플릿(머스태치)에게 전달할 값을 model 담아준다.
    model.addAttribute( attributeName: "articleList", result);
    // 3. p.175 뷰 페이지 설정
    return "articles/index";
}
```

머스태치
model.addAttribute("articleList", result);

HTTP GET
localhost/articles

[View (HTML) 렌더링]
머스태치를 HTML 만드는과정(자동)

index.mustache



HTTP

- URL : HTTP 규약의 통신 식별 경로

- method : HTTP 규약의 통신 방법

1. GET 2. POST 3. PUT 4. DELETE

- HTTP 이용한 자원 제공방법

1. path : URL 사이의 문자형식 데이터 넣어서 보내는 방법

1. 요청

localhost/articles/{매개변수1}/{매개변수2}

localhost/articles/4/edit

2. 스프링 받을때.

매핑함수(@PathVariable 타입 매개변수명){ }

2. contentType : HTTP 제공하는 내용물(payload)을 body 포함해서 보내는방법

1. application/x-www-form-urlencoded < HTML form 기본형태>

주로 폼 전송시 사용

1. 요청

<form action="/articles/create" method="post">

제목:<input name="title" />

내용:<textarea name="content"></textarea>

<button type="submit">Submit</button>

뒤로가기(Back)

</form>

2. 스프링 받을때

매핑함수(객체타입 매개변수명){ }

2. *application/json

< JQUERY AJAX 이용한 형태 >