디지털컨버전스 기반 UXUI Front 전문 개발자 양성과정

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 - JungHyun LEE(이정현)
akdl911215@naver.com

@RestController : @Controller + @ResponseBody 의 축약형으로써, 리턴값을 뷰리졸버로 매핑하지 않고 그대로 출력해줍니다.

@GetMapping : @RequestMapping(method = RequestMethod.GET) 의 축약형으로써, 애너테이션만 보고 무슨 메소드 요청인지 바로 알아볼 수 있다.

#Cotroller 생성
Controller를 하나 생성한다. 생성할 때는 @RestController를 이용한다.

```
@RestController // REST API Controller 사용한다는 것을 프레임워크에 알려줌
@RequestMapping("api") // localhost:8080/api 형태로 매핑 됨
public class GetController {

}
```

@RestController를 붙여줘야, Spring Framework 는 이 클래스가 Controller라는 것을 알 수 있다. 클래스 위의 @RequestMapping(“api”)를 입력하면, httl://[ip]:[port]/api와 같이 url 이 매핑된다.

#@RequestMapping
간단하게 화면에 문자열을 출력하는 API를 만들면, 아래와 같이 만들 수 있다.

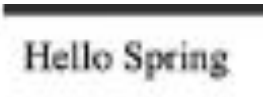
```java
@RestController
@RequestMapping("api")
public class GetController {

  // GET Method 로 통신
  // 경로는 api/getMethod 와 매핑됨
  @RequestMapping(method = RequestMethod.GET, path = "/getMethod")
  public String getRequest() {
    return "Hello Spring";
  }

}
```

@RquestMapping안에 method값으로 RequestMethod.GET을 넣어주면, HTTP GET 요청이 왔을때 동작을 한다. 그리고, path 값을 URL에 매핑한다.(위에서는 /api/getMethod와 매핑됨)

브라우저를 통해 확인해보면, 아래와 같이 화면에 Hello Spring이 나타난다.



#@GetMapping
GET 요청 방식의 API를 만들때, @RequestMapping(method = RequestMethod.GET …) 방식도 있지만, @GetMapping 을 이용하는 방법도 있다.

```
@RestController
@RequestMapping("api")
public class GetController {

  ...

  // GET Method 통신
  // 경로는 api/getParam과 매핑됨
  @GetMapping("/getParam")
  public String getParameter() {
    return "Hello Spring";
  }

}
```

단순히 @GetMapping 을 사용하면 @RequestMapping(method = RequestMethod.GET…)과 동일한 효과를 볼 수 있다.

@RequestParam
Parameter를 입력받는 API를 만드는 경우도 있는데, 그때는 @RquestParam을 사용한다.

```
@RestController
@RequestMapping("api")
public class GetController {

  ...

  @GetMapping("/getParam")
  public String getParameter(@RequestParam String id, @RequestParam(name = "password")
    return "ID: " + id + ", Password: " + pwd;
  }
}
```

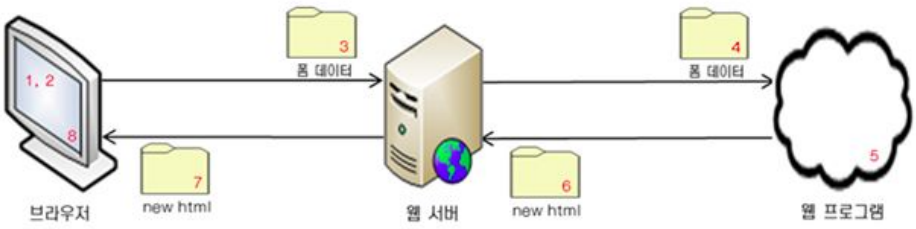
위 코드와 같이 입력 받을 파라미터를 메소드의 인자값을 넣어준다. 그리고, 그 앞에 @RequestParam을 넣는다.

기본적으로 인자 변수명을 파라미터명으로 받는다.
하지만, @RquestParam(name=”원하는 파라미터 명")을

HTML: 폼(form) 이해
폼은 알게 모르게 웹에서 많이 사용합니다. 사용자 의견이나 정보를 알기 위해 입력할 큰 틀을 만드는 데 사용되기 때문입니다. 폼은 입력된 데이터를 한 번에 서버로 전송합니다. 전송한 데이터는 웹 서버가 처리하고, 결과에 따른 또 다른 웹 페이지를 보여줍니다.

-폼 태그 동작방법
1. 폼이 있는 웹페이지를 방문합니다.
2. 폼 내용을 입력합니다.
3. 폼 안에 있는 모든 데이터를 웹 서버로 보냅니다.
4. 웹 서버는 받은 폼 데이터를 처리하기 위해 웹 프로그램으로 넘깁니다.
5. 웹 프로그램은 폼 데이터를 처리합니다.
6. 처리결과에 따라 새로운 html페이지를 웹서버에 보냅니다
7. 웹 서버는 받은 html 페이지를 브라우저에 보냅니다.
8. 브라우저는 받은 html 페이지를 보여줍니다.

위 설명을 그림 1과 같이 표현할 수 있습니다.



[그림 1]

-폼 태그 속성

폼 태그 속성에는 name, action, method, target 등이 있습니다. 폼 속성을 이용하여 전송할 때 어디로 보내야 하는지 그리고 어떤 방법으로 보낼지 정합니다. 폼 태그 속성은 다음과 같습니다.

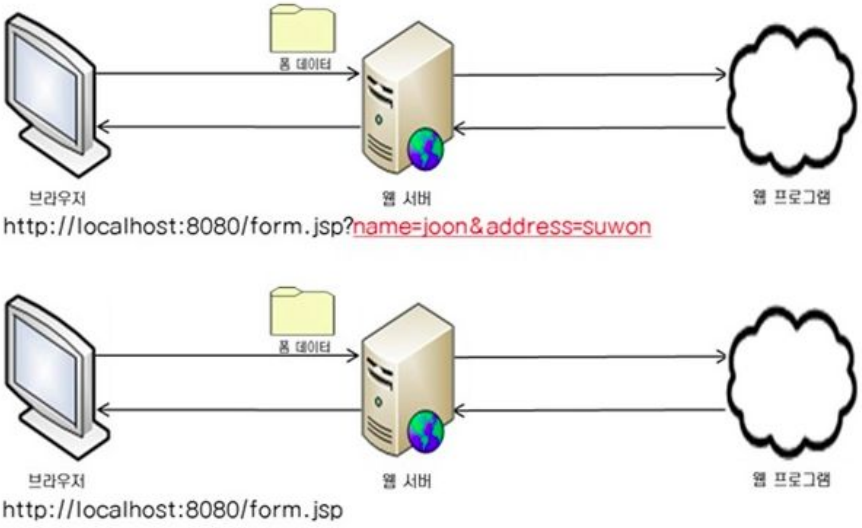
- action: 폼을 전송할 서버 쪽 스크립트 파일을 지정합니다.
- name: 폼을 식별하기 위한 이름을 지정합니다.
- accept-charset: 폼 전송에 사용할 문자 인코딩을 지정합니다.
- target: action에서 지정한 스크립트 파일을 현재 창이 아닌 다른 위치에 열도록 지정합니다.

```
<html>
    <head>
    </head>

    <body>
        <form action = "http://localhost:8080/form.jsp" accept-charset="utf-8"
              name = "person_info" method = "get">

        </form>
    </body>
<html>
```

전송할 http 메소드 종류인 GET 과 POST 는 브라우저에서 폼 데이터를 가져와 서버로 보내는 똑같은 기능을 수행하지만, 방식을 다르다. GET은 폼 데이터를 URL 끝에 붙여 보

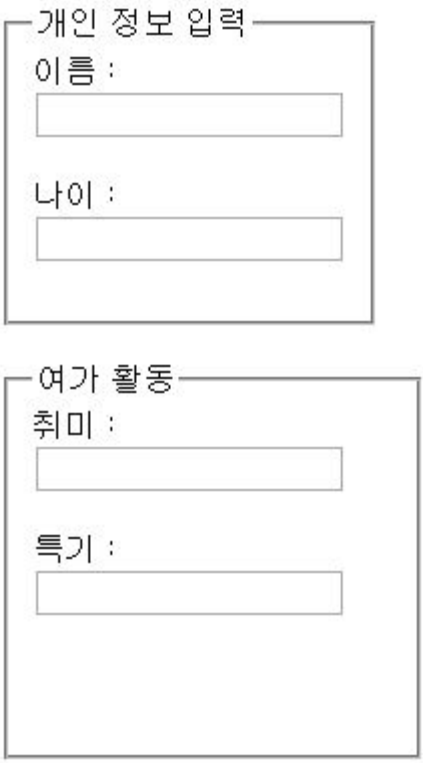


금방 그림의 위쪽은 GET방식이고 아래쪽은 POST 방식입니다.

URL 끝에 데이터를 붙여 보내는 GET 방식은 데이터가 외부에 노출되어 보안에 취약합니다. 그래서 보내려는 개인정보나 보안을 해야 하는 경우는 POST 방식을 사용해야 합니다. 또한, HTTP 메소드 정의에서 GET 방식은 지정된 리소스에서 데이터를 요청하는 경우에는 읽을 때 사용하는 메소드입니다. 반면 POST 방식은 지정된 리소스에서 데이터를 처리할 경우인 쓰고, 수정, 삭제 할 때 사용됩니다.

보안이 필요하지 않으면서 지정된 리소스에서 자원을 읽을 경우에는 GET방식을 사용하고, 그렇지 않다면 POST 방식을 사용하면 됩니다.

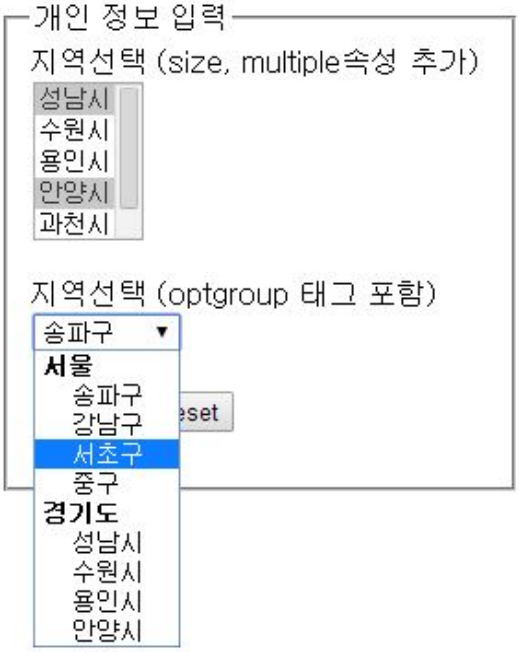
-폼 엘리먼트 그룹 <field>,<legend>태그 <fielset>태그는 폼 태그 안에 관련 있는 폼 엘리먼트들을 그룹화 할때 사용합니다. 그리고 <filedset> 태그 하위에 <legend> 태그를 [illegible] 컨트들을 목적에 맞게 이름



input 태그
- type : 태그 모양을 다양하게 변경할 수 있습니다. type 에는 text, radio, checkbox, password, button, hidden, fileupload, submit, reset 등을 지정할 수 있습니다.
- name : 태그 이름을 지정합니다.
- readonly : 태그를 읽기전용으로 합니다.
- required : 해당 태그가 필수태그로 지정됩니다. 필수 태그를 입력하지 않고, submit 버튼을 누르면 에러메시지가 웹 브라우저에 출력됩니다. (HTM5 추가사항)
- autofocus : 웹페이지가 로딩되자마자 이 속성을 지정한 태그로 포컬스가 이동됩니다. (HTML5 추가사항)
- placeholder : 태그에 입력할 값에 대한 힌트를 줍니다 (HTML5 추가사항)
- patterm : 정규표현식을 사용하여 특정범위 내의 유효한 값을 입력 받을 때 사용합니다(HTML5 추가사항)

목록태크

- select : 항목을 선택할 수 있는 태그입니다. 속성 중에 size와 multiple속성이 있습니다. size는 한 번에 표시할 항목 수를 의미하고, multiple은 다중선택을 허용할 것인지를 지정하는 속성입니다.
- select : 태그 하위에 <optgroup> 태그와 <option>태그가 있습니다. <optgroup> 태그는 <select> 태그 안에서 목록들을 그룹화할 경우 사용됩니다. label 속성을 사용하여 그룹 이름을 지정합니다. <optgroup>태그 하위에 <option> 태그를 포함합니다.
- option : 태그는 목록을 나타내는 태그입니다.

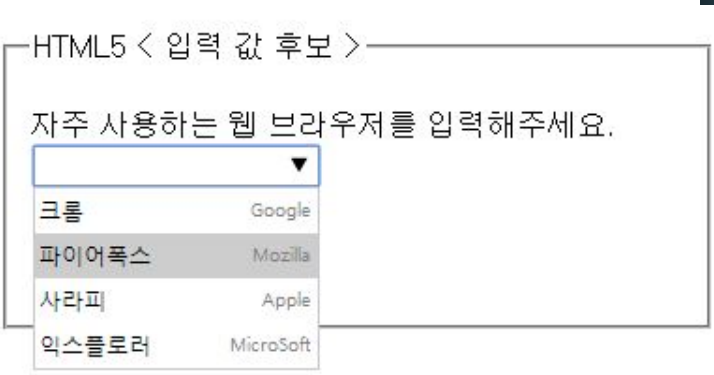


여러줄 글상자 태그

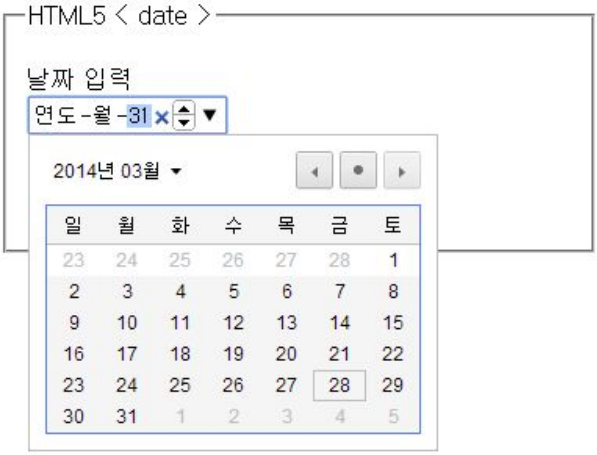
- textarea : 여러 줄을 입력받는 태그입니다. 속성 중에 rows와 cols가 있습니다. rows는 줄을, cols는 한 줄에 입력될 크기를 지정합니다.

입력 값 후보 태그

- datalist : 텍스트 상자에 입력 값 후보 목록을 지정할 경우 사용합니다.

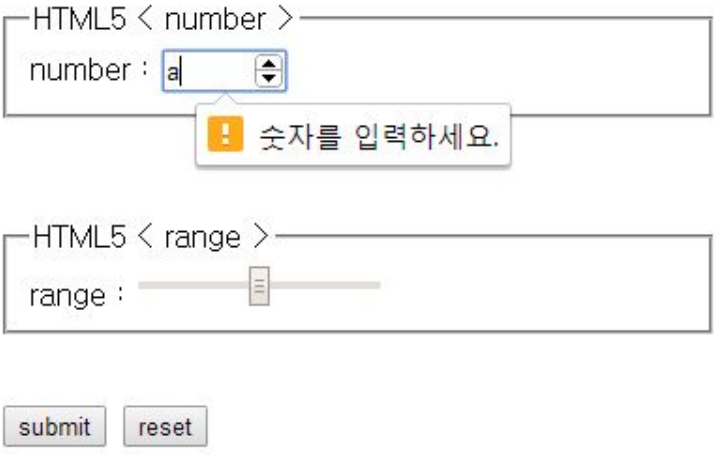


<input>태그의 date : 날짜를 입력받기 위한 속성값입니다. 날짜 선택을 위한 달력도 함께 표시됩니다. 이 값이 서버 프로그램에 전달되면 자바 date 객체에 바로 데이터가 전달됩니다. 그래서 쉽게 date 데이터를 서버 프로그램에서 받을 수 있는 장점이 있습니다.



날짜와 관련된 것에는 date 말고 month, week, datatime, datatime-local이 추가 되었습니다. date 속성과 비슷하게 지정하면 사용자가 원하는 결과를 볼 수 있습니다.

<input>태그의 number 와 range :
number 와 range는 둘 다 숫자를 입력할 때 사용□□□니다. 차이점으로 range 태그는 슬라이더 형태의 UI로 렌더링 된다는 점입니다. min, max 속성을 사용하여 최소 최댓값을 지정합니다.



위처럼 숫자만 입력 받을 수 있습니다. 숫자가 아닌 값을 입력하고 submit 버튼을 누르면 에러메시지가 나타납니다. range는 슬라이더 모습으로 숫자르 □□정하는 것을 확인할 수 있습니다.

<input> 태그의 color :
색상을 입력받을 때 사용합니다. color타입은 아직 모든 웹 브라우저에서 지원하지 않지만, 일부 웹 브라우저에서 Color Picker 형태의 UI로 렌더링 됩니다.

HTTP 란?
HTTP(Hypertext Transfer Protocol)는 클라이언트와 서버 간의 통신을 가능하게 하도록 설계되었습니다. HTTP는 클라이언트와 서버 간의 요청-응답 프로토콜로 작동합니다.

ex) 클라이언트(브라우저)가 서버에 HTTP 요청을 보냅니다. 그런 다음 서버는 클라이언트에 응답을 반환합니다. 응답에는 요청에 대한 상태 정보가 포함되며 요청 된 콘텐츠도 포함될 수 있습니다.

HTTP Methods
-GET
-POST
-PUT
-HEAD
-DELETE
-PATCH
-OPTIONS
가장 일반적으로 사용되는 2가지 HTTP 메서드는 GET 및 POST 입니다.

GET Methods
GET은 지정된 리소스에서 데이터를 요청하는 데 사용됩니다.
GET은 가장 일반적인 HTTP 메서드 중 하나입니다.
쿼리 문자열(이름/값 쌍)은 GET요청의 URL로 전송됩니다.

```
/test/demo_form.php?name1=value1&name2=value2
```

GET 요청에 대한 기타 참고 사항:
-GET요청을 캐시 할 수 있습니다.
-GET 요청은 브라우저 기록에 남아 있습니다.
-GET 요청을 북마크 할 수 있습니다.
-민감한 데이터를 처리 할 때 GET 요청을 사용해서는 안됩니다.
-GET 요청에는 길이 제한이 있습니다.
-GET 요청은 데이터를 요청하는 데만 사용됩니다(수정하지 않음)

POST Methods
POST는 리소스를 생성/업데이트 하기 위해 서버로 데이터를 보내는 데 사용됩니다.
POST 를 통해 서버로 전송 된 데이터는 HTTP 요청의 요청 본문에 저장됩니다.

```
POST /test/demo_form.php HTTP/1.1
Host: w3schools.com
name1=value1&name2=value2
```

POST는 가장 일반적인 HTTP 방법 중 하나입니다.
POST 요청에 대한 기타 참고 사항:
-POST 요청은 캐시되지 않습니다.
-POST 요청은 브라우저 기록에 남아 있지 않습니다.
-POST 요청은 북마크 할 수 없습니다.
-POST 요청은 데이터 길이에 제한이 없습니다.

PUT Methods
PUT는 리소스를 생성 / 업데이트 하기 위해 서버로 데이터를 보내는 데 사용됩니다.

POST와 PUT의 차이점은 PUT 요청이 실행 가능하다는 것입니다.
즉, 동일한 PUT 요청을 여러 번 호출하면 항상 동일한 결과가 생성됩니다.
반대로 POST 요청을 반복적으로 호출하면 동일한 리소스를 여러번 생성하는 부작용이 있습니다.

HEAD Methods
HEAD는 GET과 거의 동일하지만 응답 본문이 없습니다.
즉, GET / users가 사용자 목록을 반환하면 HEAD / users 는 동일한 요청을 하지만 사용자 목록은 반환하지 않습니다.
HEAD 요청은 큰 파일이나 응답 본문을 다운로드 하기 전과 같이 실제로 GET 요청을 하기 전에 GET 요청이 반환 할 내용을 확인하는 데 유용합니다.

DELETE Methods
DELETE 메서드는 지정된 리소스를 삭제합니다.

OPTIONS Methods
OPTIONS메소드는 대상 자원에 대한 통신 옵션을 설명합니다