ajax

ajax

ajax

ajax

- 서버로부터 데이터를 가져와 전체 페이지를 새로 고치지 않고 일부만 로드할 수 있게 하는 기법으로 비동기식 요청을 보내는데 필요한 기술
- AJAX : Asynchronous JavaScript And XML

ajax 장단점

- 장점
  1) 비동기식 방식으로 웹서버의 응답을 기다리지않고 데이터를 빠르게 처리
  2) 페이지 리로딩 없이 처리
- 단점
  1) 한 페이지에 지속적으로 사용 시 리소스가 쌓여 페이지가 느려짐
  2) 스크립트로 되어 있어 에러 발생 시 디버깅이 어려움

ajax

동기식 처리모델

- 페이지가 로드 되는 동안 브라우저는 script문이 실행되면 그 실행이 종료될때 까지 기다렸다가 종료되면 나머지 페이지를 로드하는 방식

비동기식 처리모델

- 페이지가 로드 되는 동안 브라우저는 먼저 서버데이터 요청 script문을 실행 한 후 나머지 페이지를 계속 로드하고 페이지와 상호작용을 처리하며, script요청 데이터를 기다리지 않는다. 그리고 요청데이터가 도착하면 그때 이벤트가 발생하면서 지정된 함수가 호출되어 실행되는 방식

Javascript
ajax

Javascript ajax

처리 절차

1. script문에 요청을 위한 XMLHttpRequest 객체 생성
2. 서버의 응답을 처리할 함수 생성 및 지정
   → onreadystatechange에 함수 지정
3. open메소드로 요청할 방법 및 요청할 대상(Server)선정
   → 요청메소드, 요청주소, 동기/비동기 설정
4. send메소드로 대상(Server)에 전송
   → post일때 파라미터 값 설정/get일때는 매개변수 없음
5. 응답상태에 따른 처리

Javascript ajax

XMLHttpRequest

- 비동기식으로 서버에 요청(Request)을 보내기 위한 객체로 요청 및 응답을 처리

속성

속성명	내용
onreadystatechange	readyState속성이 변경될 때 호출되는 메소드를 저장하는 변수
readyState	객체의 상태를 저장하는 변수
responseText	응답 결과를 문자열로 저장하는 변수
responseXML	응답 결과를 XML data로 저장하는 변수
status	전송/응답 결과를 저장하는 변수(코드값)
statusText	전송/응답 결과를 저장하는 변수(문자열)

Javascript ajax

readyState 속성 값

속성명	내용
0	요청이 시작되지 않은 상태 / open메소드가 호출되지 않음
1(loading)	서버와 접속된 상태 / send메소드가 호출되지 않음
2(loaded)	send메소드가 호출되고 헤더는 도착하지 않은 상태
3(interactive)	일부 데이터를 받은 상태
4(completed)	요청을 완료하고 응답하는 상태

status 속성 값

속성명	내용
200(OK)	요청 성공
404(Not Found)	페이지 없음
500(Internal Server Error)	서버 오류 발생

Javascript ajax

1. XMLHttpRequest 객체 생성

- IE7 이상, safari, firefox, opera, chrome

  var httpRequest = new XMLHttpRequest();

- IE6 이하

  var httpRequest = new ActiveXObject(Microsoft.XMLHTTP);

Javascript ajax

2. 응답 처리 함수 설정

- XMLHttpRequest객체 생성 후 속성값에 함수를 저장

  var httpRequest = new XMLHttpRequest();

  httpRequest.onreadystatuschange = 실행할 함수명;

  또는

  ```
  httpRequest.onreadystatuschange = function(){
          처리로직
  }
  ```

Javascript ajax

3. 요청 대상 설정 / 요청 처리

- XMLHttpRequest객체 생성 후 open메소드로 요청대상 설정

```
var httpRequest = new XMLHttpRequest();

//요청 대상 설정
httpRequest.open(전송방법,요청페이지,동기식/비동기식설정);
//요청 처리
httpRequest.send("param값");
```

→ send시 param값은 post인경우에는 필수, get인경우에는 요청페이지에서 처리 가능

Javascript ajax

4. 응답처리

- XMLHttpRequest객체 생성 후 속성값으로 응답 처리(text)

```
var httpRequest = new XMLHttpRequest();
httpRequest.onreadystatechange = function(){
    if(httpRequest.readyState == 4){//요청이 완료되었고
        if(httpRequest.status==200){//정상적으로 결과가 수신되었을 때
            //서버에서 보내준 데이터를 자바스크립트 변수에 저장
            var value = httpRequest.responseText;
            //서버에서 보내준 데이터를 이용하여 HTML페이지 변경
        }
    }
}
```

Javascript ajax

4. 응답처리

- XMLHttpRequest객체 생성 후 속성값으로 응답 처리(XML)

```
var httpRequest = new XMLHttpRequest();
httpRequest.onreadystatechange = function(){
    if(httpRequest.readyState == 4){//요청이 완료되었고
        if(httpRequest.status==200){//정상적으로 결과가 수신되었을 때
            //서버에서 보내준 데이터를 자바스크립트 변수에 저장
            var value = httpRequest.responseXML;
            //XML에서 원하는 데이터만 추출하는 법
            var xml = value.getElementsByTagName("태그명");
        }
    }
}
```

Javascript ajax

JSON과 XML

- JSON : JavaScript Object Notation의 약자로 자바스크립트 객체를 표현하기 위한 표기법으로 각언어별로 객체 표현방법이 달라서 통일하기 위해 사용
  형식
  {key1:value1,key2:value2,key3:value3}

- XML : Extensible Markup Language의 약자로 HTML과 매우 비슷한 문자기반의 마크업 언어로 사람과 기계가 동시에 읽기 편한 구조로 되어있음
  형식
  <태그명1>값1</태그명1>
  <태그명2>값2</태그명2>
  <태그명3>값3</태그명3>

jQuery ajax

jQuery ajax

$.ajax()를 이용하여 처리

1. url 속성을 통해 전송할 url 주소 설정
2. data 속성을 통해 전달할 데이터 설정
3. 성공, 실패 시 처리할 로직을 함수로 선언
4. 반드시 처리할 로직을 선언

jQuery ajax

$.ajax()의 주요 속성

속성명	내용
url	데이터를 전송할 URL의 주소 설정
data	서버에 전송할 데이터를 key:value 형식으로 설정(js객체)
datatype	서버가 리턴하는 데이터의 타입 설정(text,xml,json,html)
type	서버로 전송하는 형식 지정(GET, POST)
success	통신 성공했을 때 처리할 로직을 함수로 작성
error	통신 실패했을 때 처리할 로직을 함수로 작성
complete	통신 시 반드시 실행할 로직을 함수로 작성

jQuery ajax

$.ajax()를 이용하여 처리

```
$.ajax({
    url : “/test”,                  //1. 전달할 servlet url mapping
    data : {id: “idid”},            //2. 전달할 데이터
    type : “get”,                   //3. 전달 방식 지정
    success : function(data){       //4-1. 성공 시 처리할 함수
        //서버에서 보내준 데이터는 매개변수인 data로 받음
    },
    error : function(){             //4-2. 실패 시 처리할 함수

    },
    complete:function(){            //5. 반드시 처리할 절차

    }
)}
```