

Next Generation *Web*
CONFERENCE

Mar. 13~14, 2006

Ajax 프로그래밍



이창신, 책임 연구원

티맥스소프트

- XMLHttpRequest 소개
 - ✓ 오브젝트 생성
 - ✓ 핵심 API
 - 비동기 콜백을 구현하기 위한 기본 기능
 - 응답 처리 기능
 - ✓ 부가 API
 - **HTTP** 프로토콜 조작

- Ajax 기본 예제
 - ✓ 텍스트 기반 클라이언트·서버 송수신
 - ✓ **XML** 기반 웹 서비스 호출

- Ajax 실전 예제
 - ✓ **ID** 중복 확인 기능
 - ✓ 프로그래스 바 기능

- 결론

▣ XMLHttpRequest 생성의 신비

```

function createHttpRequest()
{
    if(window.ActiveXObject){
        //Win e4,e5,e6용
        try {
            return new ActiveXObject("Msxml2.XMLHTTP");
        } catch (e) {
            try {
                return new ActiveXObject("Microsoft.XML
HTTP");
            } catch (e2) {
                return null ;
            }
        }
    } else if(window.XMLHttpRequest){
        //Win Mac Linux m1,f1,o8 Mac s1 Linux k3용
        return new XMLHttpRequest() ;
    } else {
        return null ;
    }
}

```

▣ XMLHttpRequest의 콜백 방식 비동기 송수신

- 연결 초기화
 - ✓ request.open("GET", "/test.xml");
 - 필수 인수: HTTP 요청 메소드, 대상 URL
 - 추가 인수들: 비동기 여부, 인증 정보
- 요청 송신
 - ✓ request.send("요청 데이터 문자열");
- HTTP 요청 메소드에 따라 처리 방식이 다르다
 - ✓ GET: 요청 데이터를 매개변수화하여 URL에 붙여 open한다
 - URL 쿼리는 application/x-www-form-urlencoded 콘텐츠 타입 인코딩이 필요
 - ✓ POST: 요청 데이터를 send에 넘긴다

▣ GET과 POST의 차이

■ GET

- ✓ `var encName = encodeURIComponent(매개변수 이름)`
- ✓ `var encValue = encodeURIComponent(매개변수 값)`
- ✓ `data = '?' + encName + '=' + encValue;`
- ✓ `request.setRequestHeader(`
- ✓ `'Content-Type', 'application/x-www-form-urlencoded; charset=UTF-8'`
- ✓ `);`
- ✓ `request.open("GET", "/test.cgi" + data);`
- ✓ `request.send("");`

■ POST

- ✓ `request.open("POST", "/test.cgi");`
- ✓ `request.send("name=test&data=123");`

▣ 응답 수신

■ 콜백 함수를 지정

```
✓ request.onreadystatechange = callbackFunction;  
✓ function callbackFunction() {  
✓   if (request.readyState == 4) {  
✓     // 수신 완료시의 처리  
✓   }  
✓ }
```

```
✓ request.onreadystatechange = function() {  
✓   if (request.readyState == 4) {  
✓     // 수신 완료시의 처리  
✓   }  
✓ }
```

■ ※request.onload

▣ XMLHttpRequest.readyState

- **0 = UNINITIALIZED:** 오브젝트 생성됨, 그러나 **open**으로 초기화되진 않음
- **1 = LOADING:** **open**은 되었지만 **send**는 되지 않음
- **2 = LOADED:** **send**는 호출되었지만 **status**와 응답 헤더가 도착하지 않음
- **3 = INTERACTIVE:** 데이터를 받고 있는 중
- **4 = COMPLETED:** 데이터를 완전히 받았음

▣ readyState과 더불어 사용

- **status:** HTTP 응답 코드
 - ✓ 200 OK: 요청 성공
 - ✓ 401 Unauthorized: 권한 없음
 - ✓ 403 Forbidden: 접근 거부
 - ✓ 404 Not Found: 요청 리소스 없음
 - ✓ 500 Internal Server Error: 서버 내부 오류
- **statusText:** 응답 코드에 따른 메시지

▣ **responseText VS responseXML**

- 수신 완료시의 처리에서 사용

- ✓ 텍스트의 경우

- 일반적인 문자열로 다룰 수 있다
- JSON(JavaScript Object Notation) 이용

```
» {  
» "test1": "hello1",  
» "test2": "hello2"  
» }
```

```
» eval("res = "+request.responseText)
```

```
» alert(res.test2)
```

※ 배열도 가능: [a, b, c, ...]

- ✓ XML의 경우

- res = request.responseXML
- var msgs = res.getElementsByTagName("msg")
- // 최초의 msg 요소의 firstChild의 값을 표시한다
- alert(msgs[0].firstChild.nodeValue)

※ DOM 처리의 어려움: XML -> JSON 변환 유틸리티도 응용 가능

▣ HTTP 프로토콜 관련 API

- 요청 헤더
 - ✓ `setRequestHeader("key", "value")`: `open`과 `send` 사이에서 유효
- 응답 헤더
 - ✓ `getAllResponseHeaders()`
 - ✓ `getResponseHeade("key")`
- 제어 관련
 - ✓ `abort`: 요청을 취소. `readyStatus`를 0으로 한다
 - 재사용하려면 새로 `open`을 해야한다(바로 `send`하면 오류 발생)

▣ 크로스 브라우저 고려: UTF-8의 `responseText`를 해석하지 못하는 브라우저

- 서버에서 UTF-8으로 URI 인코딩 후
- `var res = decodeURIComponent(oj.responseText)`

▣ 예제와 함께 하는 데모를 시작하기 전에...

- 노트북이 있으신 분들은 텍스트 에디터를 열고 따라해보세요
- 없으셔도 좋습니다, 머리속에 가장 좋은 노트북이 있으니까요 ^^

▣ 데모: 텍스트 응답의 처리

▣ 데모: 웹 서비스 소비

▣ 데모: 실전 예제 1 – ID 중복 확인 기능

▣ 데모: 실전 예제 2 – 진짜 프로그래스 바

▣ 결론

- **Ajax의 핵심 XMLHttpRequest를 꺾고 있자!**
 - ✓ 안전한 콜백 처리
 - ✓ 부가적인 HTTP 프로토콜 처리
- **서비스 호출**
 - ✓ 텍스트 방식 VS XML 방식
 - ✓ 웹 서비스 소비
- **다양한 응용**
 - ✓ 더욱 인터랙티브한!
 - ✓ 서비스의 상태를 더욱 자세히!
- **Ajax는 이제 시작입니다**
 - ✓ 저보고 그 많은 기능을 다 짜라는 건 아니겠죠? Apache Kabuki
 - ✓ Ajax용으로는 이클립스같은 툴이 없나요? Eclipse ATF