신기술/기능 요구사항 지원 보고서
[제목: WEB2.0 기능분석]

한국소프트웨어진흥원
공개SW기술지원센터

<Revision 정보>

일자	VERSION	변경내역	작성자
2007.07.02	0.1	초기 작성	허종윤

목 차

<그림 차례>

1. 문서 개요

Web2.0에서 HTML, JavaScript™, DHTML, DOM으로 구성된 Ajax는 웹 인터페이스를 인터랙티브 Ajax 애플리케이션으로 변형하는 획기적인 방식이다. 본 문서에서는 이러한 기술들이 어떻게 작용하는지 전체적인 개요를 비롯하여 세부사항 까지 설명한다. 또한 XMLHttpRequest 객체 같은 Ajax의 중심적인 개념들을 소개한다.

가. 문서의 목적

다음과 같은 세부적인 목적을 달성하기 위하여 작성되었다.

○ AJAX 소개 및 새로운 기능 분석
○ 관련 아키텍쳐 구조 설명

나. 본 문서의 사용방법

다음과 같은 방법으로 사용할 수 있다.

○ 국내 웹개발자에게 새롭게 추가된 기술을 소개할 수 있다.
○ 사용자들에게 새로운 기능들에 대한 내용을 홍보할 수 있다.

다. 참고사항

○ 개발지원시 시스템 관련사항, 기술적 배경 등 참고사항들을 기술한다.

2. 신기술/기능 요구사항

가. 대상 업체

구 분		비 고
기관명	-	
담당자	-	
연락처	-	
회사 위치	-	

나. 요구 사항

항목	지원 내용	비고
WEB2.0 기술 소개	● 한국소프트웨어진흥원 공개SW기술지원센터 자체 신기술 조사요청에 의해 WEB2.0 기술 검토	

3. WEB2.0 개요

HTML, JavaScript™, DHTML, DOM으로 구성된 Ajax는 볼품없는 웹 인터페이스를 인터랙티브 Ajax 애플리케이션으로 변형하는 획기적인 방식이다. 본 문서에서는 이러한 기술들이 어떻게 작용하는지 전체적인 개요를 비롯하여 세부사항 까지 설명한다. 또한 XMLHttpRequest 객체 같은 Ajax의 중심적인 개념들을 소개한다.

Ajax는 일시적으로 유행하는 툴이 아니다. 웹 사이트를 구축하는 강력한 방식이며 완전히 새로운 언어를 배우는 것보다는 그다지 어렵지 않다.

Ajax에 관해 자세히 들어가기 전에 잠시 Ajax의 기능에 대해 알아보자. 오늘날 애플리케이션을 작성할 시 두 가지 애플리케이션이 있다.

- 데스크톱 애플리케이션
- 웹 애플리케이션

두 애플리케이션은 다 친숙한 것들이다. 일반적으로 데스크톱 애플리케이션은 CD상에 배치된 다음 (또는 웹 사이트에서 다운로드) 컴퓨터에 완전 설치된다. 이 애플리케이션은 인터넷을 이용해 업데이트를 다운로드하기도 하지만 애플리케이션 실행 코드는 데스크톱 상에 상주해 있다. 웹 애플리케이션은 웹서버 상에서 실행되며 웹 브라우저 상에서 접속된다.

하지만 두 애플리케이션에 대한 코드 실행 위치보다 애플리케이션 작동방식 및 애플리케이션과 사용자와의 상호작용방식이 중요하다. 일반적으로 데스크톱 애플리케이션은 상당히 빠르고 (컴퓨터 상에서 실행되고 인터넷 상에서 대기 중인 상태가 안 나온다.), 대형 사용자 인터페이스(일반적으로 운영체제와 상호작용)를 갖추며 상당히 동적이다. 거의 대기시간 없이 메뉴 및 하위 메뉴를 클릭, 지시, 입력하고 풀업한다.

반면 웹 애플리케이션은 가장 최신 것이며 데스크톱에서는 전혀 얻을 수 없는 서비스를 제공한다. 하지만 웹 애플리케이션 기능으로 인해 서버 응답 대기, 스크린 재생 대기, Request 컴백 및 새 페이지 생성에 관한 대기 기능 등이 부수된다.

Ajax는 데스크톱 애플리케이션 및 항상 업데이트 되는 웹 애플리케이션의 기능 및 상호작용 간의 차이를 줄여주는 역할을 한다. 마치 데스크톱 애플리케이션에서 찾은 것처럼 동적 사용자 인터페이스 및 가상 제어기능을 사용한다. 하지만 웹 애플리케이션 상에서 데스크톱 애플리케이션을 이용할 수 있다. 그러면 대기 중인 것이 무엇인가? Ajax가 응답 Ajax 애플리케이션으로 변환되는 과정에 대해 살펴보기로 하자.

4. 새로운 기법

Ajax에 관해 살펴보면 Ajax는 많은 기술들이 응집되어 있다. Ajax의 기본을 마치고 넘어가려면 몇 가지 다른 기술들을 면밀히 살펴보아야 한다.

Ajax 애플리케이션에 포함된 기본기술은 다음과 같다.

- 웹 양식을 구축하고 애플리케이션 완료 때까지 사용되는 필드를 식별하는 데 HTML을 사용한다.

- 자바 스크립트 코드는 Ajax 애플리케이션을 실행하는 중심 코드이며 서버 애플리케이션과의 커뮤니케이션을 용이하게 한다.
- DHTML(동적 HTML)은 웹 양식을 동적으로 업데이트 한다. div, span및 기타 동적 HTML 요소를 사용해 HTML을 마크업 한다.
- 서버에서 복귀된 HTML 및 (때로) XML 구조를 다루는 데 있어 DOM, 즉 문서 객체 모델(Document Object Model)을 사용한다.

이 기술들에 대해 간략히 요약하고 각 기술의 기능에 대해 좀 더 알아보기로 하는데 각 기술에 관한 자세한 사항은 차후 글에서 다룰 것이다. 자바 스크립트에 익숙할수록 Ajax에 담긴 기술에 관한 일반적인 지식 단계에서 각 기술에 관한 자세한 지식으로 넘어가는 게 더 쉬워진다.

가. XMLHttpRequest 객체

Ajax의 첫 번째 객체는 XMLHttpRequest 이고, 자바스크립트의 일종이고, <그림 1. XMLHttpRequest 생성>과 같이 간단하게 생성된다.

```
<script language="javascript" type="text/javascript">
var xmlHttp = new XMLHttpRequest();
</script>
```

<그림 1. XMLHttpRequest 생성>

자바 스크립트 객체는 XMLHttpRequest를 통해 서버에 전달하는 자바 스크립트 기술의 일종이다. 이 객체는 애플리케이션 흐름이 정상적이지 않으며 Ajax 기술의 많은 부분을 차지하고 있다.

정상적인 웹 애플리케이션에서 사용자는 양식 필드를 기입하며 제출 버튼을 클릭한다. 그러면 모든 양식을 서버에 보내며 서버는 처리과정을 통해 양식을 스크립트(일반적으로 PHP, 자바 또는 CGI 과정/이와 유사한 과정)에 전송한다. 스크립트를 실행할 때 스트립트를 통해 완전히 새로운 페이지가 전송된다. 그 페이지는 데이터가 작성된 새로운 양식의 HTML/확인 페이지 또는 원 양식에 기입된 데이터에 근거해 선택된 옵션이 포함된 페이지일 수 있다. 물론, 서버상의 스크립트/프로그램이 처리되면서 새로운 양식을 다시 보내는 동안 사용자는 대기해야 한다. 서버로부터 데이터를 다시 받을 때까지는 스크린 상에 아무 것도 없게 되며 결국 대화성은 낮게 된다. 사용자는 즉각적으로 응답을 받지 못하며 데스크톱 애플리케이션 상에서 작업하는 기분이 들지 않게 된다.

Ajax는 근본적으로 자바 스크립트 기술 및 웹 양식 및 서버 간의 XMLHttpRequest 객체를 결합한다. 사용자가 웹 양식을 기입할 때 데이터는 직접 서버 스크립트에 전송되지 않고 자바 스크립트 코드에 전달된다. 대신 자바 스크립트 코드는 양식 데이터를 포착해 Request를 서버에 전송한다. 이 과정이 일어나는 동안, 사용자 스크린 상의 양식은 순식간에 나타나거나 깜빡이거나 사라지거나 정지하지 않는다. 즉 자바 스크립트 코드는 몰래 Request를 전송하며 사용자는 Request가 만들어졌는지도 알지 못한다. 게다가 Request를 비동기적으로 전송하기

때문에 더 좋은 상황이 된다. 이는 자바 스크립트에서 서버 응답을 그냥 대기하지 않는다는 것을 의미한다. 따라서, 사용자는 데이터를 계속 기입하고 화면이동하고 애플리케이션을 사용한다.

그런 다음 서버는 자바 스크립트 코드(웹 양식에 대해 아직도 대기 중임)에 데이터를 다시 전송한다. 자바 스크립트 코드에서는 데이터와의 상호기능을 결정하며 연속적으로 양식 필드를 업데이트 하면서 애플리케이션에 즉각적인 응답을 준다. 결국 사용자는 양식을 제출/재생하는 작업 없이 새로운 데이터를 얻게 된다. 자바 스크립트 코드는 데이터를 얻고 계산도 수행하며 또 다른 Request를 전송하며 이런 모든 과정은 사용자 개입 없이도 된다. 이것이 바로 XMLHttpRequest 객체의 장점이다. XMLHttpRequest 객체는 서버와 같이 커뮤니케이션을 주고받고 사용자는 그 과정에서 벌어지는 과정을 알지 못한다. 이로 인해 데스크톱 애플리케이션과 마찬가지로 동적, 상호 반응적인 고도의 양방향 경험을 얻게 되지만 그 속에 인터넷의 모든 장점이 담겨 있다.

일단 XMLHttpRequest에 대해 다루게 되면 나머지 자바 스크립트 코드는 상당히 평범한 것들이다. 사실 다음과 같은 기본적인 작업에 자바 스크립트 코드를 이용한다.

- 양식 데이터 얻기: 자바 스크립트 코드로 HTML 양식에서 데이터를 꺼내 이를 서버에 전송하는 작업이 간단해진다.
- 양식 상의 값 변환: 필드 값 설정에서 연속적인 이미지 교체작업에 이르는 양식 업데이트 작업 또한 간단하다.
- HTML 및 XML 구문분석: 자바 스크립트 코드를 이용해 DOM을 처리하고 서버에서 다시 전송하는 HTML 양식 및 임의의 XML 데이터에 관한 구조를 다루게 된다.

```
// Get the value of the "phone" field and stuff it in a variable called phone
var phone = document.getElementById("phone").value;

// Set some values on a form using an array called response
document.getElementById("order").value = response[0];
document.getElementById("address").value = response[1];
```

<그림 2. 자바스크립트 코드에서 필드값 설정>

XMLHttpRequest만 이해하면 Ajax 애플리케이션에서 나머지는 대부분<그림 2. 자바스크립트 코드에서 필드값 설정>에 나온 바와 같이 HTML과 결합된 단순 자바 스크립트 코드다.

나. Request 객체

XMLHttpRequest 객체를 작동시키기 위해선 몇 가지 다른 작업을 해야 한다.

○ Microsoft 브라우저

Microsoft 브라우저, Internet Explorer는 XML을 다룰 시 MSXML를 사용한다. Internet

Explorer 상에서 다뤄야 할 Ajax 애플리케이션을 작성할 시 독특한 방식으로 XMLHttpRequest 객체를 작성해야 한다. IE에 설치된 JavaScript 기술 버전에 따라 MSXML 버전도 변하게 되며 실지로 2개의 버전이 있다. 따라서 두 경우를 다루는 코드를 작성해야 한다. Microsoft 브라우저 상에서 XMLHttpRequest 객체를 생성하는 데 필요한 코드에 관해선 <그림 3. Microsoft 브라우저에서 XMLHttpRequest 객체 생성>과 같다.

```
var xmlHttp = false;
try {
  xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");
} catch (e) {
  try {
    xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
  } catch (e2) {
    xmlHttp = false;
  }
}
```

<그림 3. Microsoft 브라우저에서 XMLHttpRequest 객체 생성>

xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");

xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");.

상기의 2개의 라인코드로 MSXML의 한 버전을 이용해 XMLHttpRequest 객체 생성을 한다. 하지만 객체가 생성되지 않는 경우 다른 버전을 사용해 XMLHttpRequest 객체를 생성한다. 두 코드 다 작동되지 않는 경우 xmlHttp 변수는 거짓으로 설정되고 작동되지 않는 것이 있다는 것을 코드에 알려 준다. 그럴 경우, 비-Microsoft 브라우저가 있을 가능성이 있다. 따라서 객체 생성을 위해선 다른 코드를 사용해야 한다.

○ Mozilla 및 비-Microsoft 브라우저

인터넷 브라우저를 선택하지 않거나 비-Microsoft 브라우저를 작성할 경우 다른 코드가 필요하다.

var xmlHttp = new XMLHttpRequest object;.

이 단순한 라인으로 Mozilla, Firefox, Safari, Opera 및 임의의 양식/형태에서 Ajax애플리케이션을 지원하는 기타 비-Microsoft 브라우저에서 XMLHttpRequest 객체를 생성한다.

```
/* Create a new XMLHttpRequest object to talk to the Web server */
var xmlHttp = false;
/*@cc_on @*/
/*@if (@_jscript_version >= 5)
try {
  xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");
} catch (e) {
  try {
    xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
  } catch (e2) {
    xmlHttp = false;
  }
}
@end @*/

if (!xmlHttp && typeof XMLHttpRequest != 'undefined') {
  xmlHttp = new XMLHttpRequest();
}
```

<그림 4. 다중 브라우저 방식으로 XMLHttpRequest 객체 생성>

<그림 4. 다중 브라우저 방식으로 XMLHttpRequest 객체 생성>에서 코드에 관한 핵심은 세 가지 과정으로 요약된다.

1. 변수 xmlHttp를 생성해 앞으로 생성할 XMLHttpRequest 객체를 참조한다.
2. Microsoft 브라우저에서의 객체를 시도, 생성한다.

- Msxml2.XMLHTTP 객체를 사용해 XMLHttpRequest 객체를 시도, 생성한다.
- 과정이 실패할 경우, Microsoft.XMLHTTP 객체를 사용해 XMLHttpRequest 객체를 시도, 생성한다.

3. xmlHttp가 아직도 설정되지 않은 경우 비-Microsoft 방식으로 XMLHttpRequest 객체를 생성한다.

위과정 끝단계 시 사용자가 실행하는 브라우저 종류에 관계없이 xmlHttp의 경우 유효한 XMLHttpRequest 객체를 인용한다.

다. AJAX에서의 Request/Response

Ajax 애플리케이션에 대해 이해하고 XMLHttpRequest 객체 및 객체 생성에 관한 기본적인 개념을 알아본다. Ajax 애플리케이션은 웹 애플리케이션에 제출되는 HTML 양식보단 서버상의 임의의 웹 애플리케이션에 대화하는 JavaScript 기술이라는 사실을 알게 된다.

XMLHttpRequest 객체는 작성하는 각각의 Ajax 애플리케이션에서 일정 형태로 사용하는 중요 코드라 Ajax 애플리케이션이 포함된 기본 Request/Response 모델을 통해 객체 사용법을 알아본다.

○ Request 만들기

새로운 XMLHttpRequest 객체가 있는 경우 이를 시험해 보자. 먼저 웹 페이지에서 호출하는 JavaScript 방법이 필요하다.(사용자가 텍스트에서 입력하거나 메뉴에서 옵션을 선택할 시와 같음.) 그 다음, 거의 모든 Ajax 애플리케이션에서의 동일한 기본 아웃라인을 따른다.

① 웹 양식으로부터 필요한 모든 데이터 얻기
② 연결할 URL 구축
③ 서버 연결
④ 서버 실행 종료 시 서버 실행 기능 설정
⑤Request 전송

<그림 5. Ajax가 포함된 Request 생성>는 위의 순서대로 5단계를 진행하는 Ajax 방법의 예에 관해 나와 있다.

```
function callServer() {
  // Get the city and state from the web form
  var city = document.getElementById("city").value;
  var state = document.getElementById("state").value;
  // Only go on if there are values for both fields
  if ((city == null) || (city == "")) return;
  if ((state == null) || (state == "")) return;

  // Build the URL to connect to
  var url = "/scripts/getZipCode.php?city=" + escape(city) + "&state=" +
escape(state);
  // Open a connection to the server
  xmlHttp.open("GET", url, true);

  // Setup a function for the server to run when it's done
  xmlHttp.onreadystatechange = updatePage;

  // Send the request
  xmlHttp.send(null);
}
```

<그림 5. Ajax가 포함된 Request 생성>

Ajax 코드의 첫번째 비트는 몇 가지 양식 필드 값을 포착하는 기본 JavaScript 코드를 사용한다. 그런 다음 이 코드에서는 연결 최종 목적지로 PHP 스크립트를 설정한다.
PHP 스크립트의 URL을 지정한 다음 GET 매개변수를 이용해 이 URL에 도시 및 국가를 추가한다. 그 다음 연결하면 먼저 XMLHttpRequest 객체가 작동되는 것을 보게 된다. 연결방법은 연결 URL 뿐만 아니라, GET 매개변수에도 나와 있다. 최종 매개변수를 true로 설정한 경우, 이 매개변수에선 비동기식 연결(Ajax를 만든다.)을 요구한다. false로 설정한 경우엔 Request를 만들 시 서버 상에서 Ajax에서의 JavaScript 코드가 대기하고 응답을 받을 때 코드가 지속된다. 사용자는 최종 매개변수를 true로 설정하면서 서버에서 배경에 있는 Request를 처리하는 동안 사용자는 웹 양식(기타 JavaScript 방식 포함)을 여전히 사용한다.
한편 xmlHttp(이것은 XMLHttpRequest 객체의 인스턴스)의 onreadystatechange 속성으로 서버 실행이 종료될 시(5분/5시간 내에 종료될 수 있음) 서버 기능을 명령한다. 이 코드는 서버 상에서 대기하지 않기 때문에 서버가 기능을 인식해 서버에 응답할 수 있도록 하는 게 필요하다. 이 경우 서버에서 Request를 처리하면서 종료 시 이른바 updatePage()라 불리는 특수 방법을 트리거한다.
최종적으로 send() 코드를 0(null) 값으로 호출한다. 데이터를 추가해 이를 서버에 전송하므로 Request에는 추가해서 보낼 게 없다. 이렇게 되면 Request를 발송하고 서버는 서버에 요구된 기능을 실행한다.
이 코드에서 나오는 것이 없는 경우, 코드가 상당히 간단하다는 것을 명심하라. 이 코드는 Ajax 애플리케이션의 비동기적 특성을 제외하고는 상당히 단순하다. 이 코드를 통해 복잡한 HTTP Request/응답 코드보다는 근사한 애플리케이션 및 인터페이스에 완전 초점을 맞추도록 한다는 사실을 여러분은 높게 평가할 것이다.
<그림 5. Ajax가 포함된 Request 생성>에서 코드는 코드를 얻는 방법만큼이나 쉽다. 데이터는 단순 텍스트이고 Request URL의 일부로 포함된다. GET 매개변수는 더 복잡한 POST대신 Request를 전송한다. 여기에 덧붙일 XML/컨텐츠 헤더가 없고 Request 본체에 전송할 데이터도 없다. 이게 바로 Ajax 유토피아다.

이럴경우에는 POST Request를 전송하는 방법, Request 헤더 및 컨텐츠 형식을 설정하는 방법, 메시지에 XML을 설정하는 방법 및 Request에 보안기능을 추가하는 방법을 사용한다.

○ Response 만들기

서버 응답에서는 다음의 2가지 사항이 중요하다.

① xmlHttp.readyState 속성이 4와 같을 때까지는 어떤 작업도 해선 안 된다.
② 서버는 xmlHttp.responseText 속성에 응답한다.

```
function updatePage() {
  if (xmlHttp.readyState == 4) {
```

```
    var response = xmlHttp.responseText;
    document.getElementById("zipCode").value = response;
  }
}
```

<그림 6. 서버 Response 생성>

<그림 6. 서버 Response 생성>은 <그림 5. Ajax가 포함된 Request 생성>에서 전송된 값에 근거해 서버에서 호출하는 방법에 관한 예를 보여준다.

xmlHttp.readyState 코드는 서버에서 해당 준비 상태로의 호출을 대기하고 서버에서 다시 복귀되는 값(이 경우, 사용자 기입 도시 및 국가에 대한 ZIP 코드)을 사용해 또 다른 형태의 양식 필드를 설정한다. 그 결과, zipCode 필드는 ZIP 코드와 함께 갑자기 나타난다. 하지만 사용자는 버튼을 클릭해서는 안 된다! 그게 바로 이전에 말했던 데스크톱 애플리케이션이다. Ajax 코드에는 응답성, 동적 상태 외의 더 많은 것이 있다. 일단 서버에서 zipCode를 복귀시키고 updatePage() 방식으로 도시/국가 ZIP 코드와 함께 zipCode 필드 값을 설정하는 경우 사용자는 값을 무효로 한다. 값을 무효로 하는 데는 두 가지 이유가 있다. 예에서 나오는 상황을 단순화시키고, 때로는 사용자가 서버에서 명령하는 것을 무효로 하기 위해서다.
서버에 전송하고 응답에 관해 취급할 방법은 <그림 7. Ajax 프로세스 시작>과 같이 JavaScript 기술을 활용한다. 사용자가 도시/국가 필드에 관한 새로운 값을 입력할 경우 callServer() 방식을 전송한 다음 Ajax 애플리케이션이 시작된다.

```
<form>
 <p>City: <input type="text" name="city" id="city" size="25"
       onChange="callServer();" /></p>
 <p>State: <input type="text" name="state" id="state" size="25"
       onChange="callServer();" /></p>
 <p>Zip Code: <input type="text" name="zipCode" id="zipCode" size="5" /></p>
</form>
```

<그림 7. Ajax 프로세스 시작>

5. 참고자료

- http://www.ibm.com/developerworks/web/library/wa-ajaxintro1.html