Created by Firejune at 2006/12/16

Ajax 기반 웹사이트 구축에 필요한 UI 디자인

이번 프로젝트를 진행하면서 가장 답답했던 것은 Ajax 기반 웹사이트를 구축할 때 필요한 요소에 대해 이해하지 못하고 접근했다는 것이다. 보통 웹사이트에 비해 오히려 불편하거나 오해를 사는 요인이 증가했으며, 이것은 UT(User Test)를 통해 명백히 증명되었다. 시중에는 Ajax를 구현하는 것에 급급한 책들이 대부분이기에 스팩을 작성하거나 UI(User Interface)를 디자인 할 때 Ajax를 어디서 부터 어떻게 구현 해야 할지 마땅히 참조할 곳도 없는 현실이다. Ajax사이트를 기획하거나 디자인할 때 최소한 반영해야 할 항목들을 두서없이 정리해 본다.

Ajax란, Asynchronous Javascript And XML의 약자이며 웹 개발 기법의 일부분이다. 전체 페이지를 다시 로드하지 않고 필요한 데이터만을 주고 받도록 도록 웹 서버에 요청할 수 있게 해 준다.

대부분의 브라우저는 Ajax 통신방식을 지원하기는 하지만 Ajax를 위한 UI를 지원하지는 않는다. 예를 들어 Ajax로 불러 들인 데이터는 브라우저의 뒤로가기 버튼을 사용해 다시 되돌릴 수 없다. 그리고 페이지를 읽는 동안의 프로세스는 표현하지만 Ajax가 통신하는 동안에는 그렇지 않다.
브라우저에서 기본으로 제공하는 UI를 웹상에서 구현해야하는 것은 선택이 아닌 필수라 할 수 있겠다.

1. Ajax 액션 통신 시각효과
2. Ajax 액션 버튼 시각효과
3. Ajax 메시지 처리
4. 뒤로/앞으로 이동버튼 문제
5. 페이지의 새로 고침
6. 파일 업로드 프로그레시브
7. 반복 요청/수신 반환

Ajax 액션 통신 시각효과
앞서 말했듯이 Ajax가 작동하게 되면 서버로부터 응답이 오기까지 페이지는 시각적인 변화가 거의없다.(파이어폭스의 FireBug와 같은 Ajax 송수신을 모니터링 할 수 있는 플러그인을 사용하지 않는 한) 사용자는 실수(또는 통신장애)로 오인하고 같은 액션을 재차 시도할 수 있기 때문에 Ajax로 통신을 시작하는 시점에 인디케이터(작동 중임을 알리는 애니메이션 이미지)를 사용하여 웹사이트가 작동 중임을 알려야 한다. 인디케이터의 위치는 사이트 레이아웃 디자인 과정에 신중히 고려하여 선정해야 하며 항상 눈에 띄는 곳이 좋다. 추천하는 포지션은 액션이 일어나는 버튼과 가까운 곳 또는 Ajax로 불러들인 데이터가 표시될 곳의 상단이다.

Ajax 액션 버튼 시각효과
스태틱(한자리에 고정 된) 버튼이라면 DIM(작동 불능)상태로 변경하는 조치도 동시에 필요하다. 데이터의 반복 입/출력을 의도적으로 시도할 수 있기 때문이다. 액션에 사용되는 버튼 또는 링크는 DIM 상태를 표현할 수 있는 모양을 갖추고 있어야 하며, 기획자 뿐만 아니라 디자이너도 버튼의 액션이 어느 시점에 어떻게 사용되는지에 대해 정확히 이해하고 있어야

한다.(다이어그램으로 전달하면 효과적이다.)

Ajax 메시지 처리

HTTP 500 - 서버나 프로그램에 문제가 있습니다. HTTP 503 - 일시적으로 사용할 수 없습니다. HTTP 404 - 파일을 찾을 수 없습니다. HTTP 403 - 접근할 수 없습니다.

우리는 위와 같은 HTTP 오류메시지를 자주 접해 보았다. Ajax 송/수신 또한 예외는 아니다. 혹은 애플리케이션이나 데이터베이스 오류에 대한 메시지 처리도 필요하며, 사용자가 범한 오류를 처리하기 위해서도 필요하다. 보통 자바스크립트의 alert 메서드(조금 식상하지만...)를 사용하여 비교적 글로벌하게 처리할 수 있다. 보다 효율적으로 메시지를 전달하고자 한다면, 인디케이터와 마찬가지로 오류 메시지 처리영역을 액션이 일어나는 곳 주변에 비치해 두는 것도 좋은 방법다.

뒤로/앞으로 이동버튼 문제

사용자는 이전 컨텐츠를 조회하기 위해 브라우저의 백(뒤로가기)버튼을 사용한다. 매우 자연스러운 플로우이며 브라우저에서 당연시 여겨지고 있다. Ajax가 사용되는 환경에서 백버튼을 사용하면 사용자는 의도와 전혀 다른 결과물을 볼 수 있으며 심지어는 사이트를 떠나 버릴수도 있다. 이는 컨텐츠 자체를 Ajax로 다룰때 나타나는 고질적인 이슈이다. GWT(Google Web Toolkit) 등의 프레임웍을 사용하면 자체적으로 해결 할 수 있다고 말하고 있지만 표준을 준수하는 방법이 아닌 트릭에 가깝기에 구현시 약간의 위험을 감수해야한다. 해시를 사용해 해결하는 방법을 언젠가

작성한 'Ajax 요청시 뒤로가기(백버튼) 문제 해결하기' 포스트를 참고하자. 이 문제는 특수한 상황에 속하는 것으로 보고 Ajax를 이용해 페이지를 통째로 전환하려는 생각은 가급적 피하는 것이 좋다.

페이지의 새로 고침
사용자는 페이지를 리프래시(새로 고침)하여 전체 데이터를 다시 갱신 할 수 있다. Ajax로 불러들인 데이터를 새로 고치게 되면 모든 내용이 초기화 된다. 이 문제를 해결하는 것은 매우 까다롭다. 최종으로 내린 XMLHttpRequest의 응답을 쿠키에 저장하고 새로 고침 할 때 최종으로 불러온 결과물를 쿠키에서 뽑아내거나, 주소를 쿼리에 저장하거나, 다시금 최종 위치의 정보를 서버에 요청하게 처리하는 꼼수를 생각할 수도 있겠다. (얼마전 디자인을 새롭게 한 굿찌닷컴은 해시를 이용하여 새로 고침 및 고유주소에 대한 문제를 효과적으로 처리하고 있다.)

파일 업로드 프로그레시브
서버로부터 응답이 오기까지 오랜 시간(5초 이상)이 필요한 부분은 인디케이터로 처리하는 것을 피해야 한다. 단순히 반복되는 인디케이터가 5초이상 지속되면 사용자는 사이트의 오류로 생각할 수 있다. 특히, 오랜 시간을 소모하는 업로드 과정에 Ajax를 사용할 때 파일 업로드의 진행과정(남은시간/속도/프로그레스바)을 구현하면 매우 효과적이다. 최근 몇몇 Ajax 애플리케이션에는 Perl을 이용한 실시간 업로드 프로그레시브(진행상황)를 데스크탑 애플리케이션에 못지 않은 수준으로까지 구현하고 있다.

반복 요청/수신 반환
이미 수신한 결과물을 또다시 요청하는 경우에 대한 처리도 필요하다. 이

것은 매우 귀찮은 작업이지만, Ajax를 사용함으로 얻을 수 있는 효율성을 극대화 할 수 있다.(위자드닷컴은 이 문제를 비교적 잘 구현했다.) 이미 수신한 바 있는 데이터와 같은 결과를 서버로부터 응답받는 것은 매우 비효율적이기 때문에 마지막으로 수신한 시간과 비교하여 갱신 여부를 파악할 수 있는 아키텍쳐가 필요하다. 다시말해 서버는 클라이언트의 마지막 수신 시간값을 기준으로 갱신 여부를 파악하여 그 결과에 적절히 응답해야 한다. 물론, 갱신 여부가 필요없는 데이터에 이러한 처리는 불필요하며, 이미 완성된 DOM을 그대로 보여주는 것으로 충분하다.

이 밖에도 고유주소에 대한 처리, 논 자바스크립트 환경의 브라우징, 접근성을 파괴하지 않는 Ajax사용 등 고질적인 기술이슈들을 모두 나열하자면 한도 끝도 없지만 이정도에서 정리하자. 웹 개발자(디자이너, 플래너, 프로그래머) 는 위와 비슷한 작업을 상당히 귀찮게 여기는 경향이 있다. 수백, 수십여가지 예외처리가 필요할 수도 있는 브라우저 컨벤션을 일일이 정의/구현하는 것을 미친짓(?)으로 생각할 수도 있다는 말이다. 하지만 편리하고 실용적인 Ajax 웹사이트 디자인을 꾀한다면 적어도 최초에 열거한 7가지 사항을 간과해서는 않될 것이다.