크로스 브라우징

웹 표준에 따라 개발을 하여 서로 다른 OS 또는 플랫폼에 대응하는 것을 말한다. 즉, 브라우저의 렌더링 엔진이 다른 경우에 인터넷이 이상없이 구현되도록 하는 기술이다. 웹 사이트를 서로 비슷하게 만들어 어떤 **환경** 에서도 이상없이 작동되게 하는데 그 목적이 있다. 즉, 어느 한쪽에 최적화되어 치우치지 않도록 공통요소를 사용하여 웹 페이지를 제작하는 방법을 말한다.

크로스 브라우징 이슈에 대응하는 프론트엔드 개발자들의 전략

대한민국 프론트 개발의 현 주소...

안타깝게도 IE 구 버전을 사용하는 고객이 정말 많다...

사실 사용하시면서도 IE 구 버전인지는 모르고 인터넷이라고 알고 계실 것이다.

개발자들은 어떻게 해야하는가? 가능만하다면, 직접가서 브라우저를 바꿔주고 싶다.

또 모바일 기기들이 등장하면서, 모바일 기기내에 탑재되는 브라우저까지 생각해야 한다.

다행히, (다행인지는 모르겠으나... N-Screen...)

모바일 기기 내부에 탑재되는 브라우저들은 대체로 HTML5나 ECMAScript5와 같은 최신 기능을 대부분 지원한다.

즉, PC보다 라이브러리의 필요성이 상대적으로 덜하다.

크로스 브라우징 이슈에 대한 전략같지 않은 전략을 세워보자!
**

**

1.* 적용 기능의 지원 브라우저를 파악하자!

개발 기획 시에 우리가 개발할 기능들을 정의하고, 그 기능들을 **지원할 브라우저를 우선 파악**한다. 제공하는 서비스에서 핵심적인 부분은 모든 브라우저에서 동작해야 하지만, 사용자 경험을 고려한 부가적인 기능들은 조금 빠져도 되지 않을까? (개인적인 의견) 이것 저것 신경쓰다가 정작 중요한 것(ex 비즈니스 로직)을 놓치게 되는 '우'를 범하지 말자! 추가적으로 가능하다면, 서비스를 제공하는 주 타겟층을 예측하고 그 타겟층이 주로 사용하는 브라우저를 집중적으로 공략해보자.

> 구 브라우저 대상으로는, 핵심기능만 쓸만하게 동작하게 하자!
>
> 단, 동작이 안되면 안된다. 정적으로라도 렌더링은 되야한다.
>
> 신 브라우저 대상으로는 지원되는 최신 기술을 적용하여 멋지고 향상된 UX를 제공하자.

서비스를 제공하는데 있어서, 차이는 있지만 핵심적인 내용은 제공해야 한다. 내용을 똑같이 다 보여주기 보다는 조금 다르게 보여주더라도 최신 기술을 적용하자.

2*. 모든 환경에서 지원해야 한다면 라이브러리를 사용하자!

라이브러리는 호환성 이슈를 해결하기 위한 아주 좋은 전략이다. (jQuery, underscore.js, extJS, HTML5 polyfill 라이브러리) 하지만 이 라이브러리를 사용하기 위해 저 라이브러리를 사용하고, 저 라이브러리를 사용하려면 또 저 라이브러리를 추가해야 한다.

라이브러리가 엄청 비대해질 수 있다. 이 라이브러리를 잘 관리해야하는 cost가 발생한다. (각 브라우저마다 코드를 작성하는 일보다는 더 나을 수 있다.)

HTML5 Polyfill 정리된 사이트

3.* 직접 구현시에는 '기능 탐지'를 이용하자!

각 브라우저에서 지원하는 함수, 메소드, 그리고 기능들이 다르기 때문에, if 문을 통해서 분기를 설정한 다음에 함수를 적용해야 할 것이다.

12345	if(isIE){ attatchEvent();} else if(isFirefox){ addEventListener();}	cs

isIE가 무슨 변수인지는 모르겠지만, 딱 보면 현재 브라우저가 IE이니?라고 물어보는 boolean 변수인 것 같다. 이렇게 하면 분기가 브라우저 별로 나뉘었다. 그런데 나중에 ie가 addEventListener를 지원하기 시작한다면? 구 브라우저가 미래에 업데이트 될 상황을 대비해야 한다! 정말 골치아픈 녀석이다.

그래서!

feature detection!!이라는 방법이 있다.

어떤 브라우저인지는 우리가 알 필요가 없다 그저 너의 브라우저에 내가 제공하고자 하는 기능이 있는지를 확인한다.

1234567891011121314	addHandler: function(element, type, handler){ //modern browsers : chrome, safari, firefox if(element.addEventListener){ element.addEventListener(type, handler, false); } //IE else if(element.attachEvent){ element.attachEvent("on" + type, handler); } //Old Browsers else { element["on" + type] = handler; }}Colored by Color Scripter	cs

***4*. 각종 툴들이 많이 나오고 있다. Tool 들을 사용하자!**

웹 표준을 지키고 있는지 검사하기!

HTML5 CSS는 오류가 발생하지 않고 화면이 깨진다. 그래서 다음과 같은 서비스들이 있다. 이 이외에도 정말 많은 서비스들이 존재한다. Cross-Browser Compatibility 라고 구글에 검색해보자.

* Can I use?

* HTML5 Markup Validation Service

* CSS-Validator

reset.css (or normalize.css)를 사용하자.

CSS의 경우, browser의 기본 스타일이 제각각인 경우가 있다. 동일한 스타일을 적용하기 위해선 defalut 값을 초기화시킬 필요가 있다. 이것도 하나의 라이브러리로, 검색해보면 여러 개를 찾을 수 있다. 무작정 Copy & Paste 하지 말고, 어떤 의미에서 초기화를 시키는지 알고 사용하자.

Prefix 를 적어주자.

아직 웹 표준이 안된 기능을 브라우저들이 가져다 지원하기 때문에 기능 앞에 각 브라우저들의 엔진을 prefix로 적어주는 것. 모든 브라우저에서 지원하는 호환 프로퍼티를 먼저 정의하고 CSS3에서 지원하는 프로퍼티를 나중에 정의하는 방법을 사용해야 한다. 이와 같은 이유로 접두어가 없는 속성은 가장 나중에 추가해줘야 한다!

123456789	#menu { border-radius: 15px;}#menu { -webkit-border-radius: 15px; -moz-border-radius: 15px; border-radius: 15px;}Colored by Color Scripter	cs

이상으로 크로스 브라우징 이슈에 대응하는 프론트 엔드 개발자들의 전략 포스팅을 마치겠습니다.

막짤은 소신껏.