

prototype.js의 엘리먼트(Element) 다루기 (insert, update, remove)

항상 될하다보면 자기 중심으로 생각하게 되는것 같습니다. 상대방의 입장에서 생각해야 되는데 그게 쉽지 않습니다. 더군다나 웹에서는 상대방(?)의 대한 정보가 보통은 전무한 편이라서 더 자기중심적으로 생각하게 되는 경우가 많죠.

초창기에 개발에 대해 전혀 모를때 인터넷에 있는 많은 정보들을 볼 때 가장 답답한 부분중 하나가 설명하는 사람이 당연히 내가 알려라고 생각하고 그냥 넘어가는 부분을 경작 나는 몰라서 그사람의 설명이나 소스까지 도달하기도 어려웠다는 것이지요. 머 그렇다고 항상 모든 것을 다 설명한다는 것도 어렵겠지만 블로그에서는 최대한 자세하게 쓰려고 하고 있기는 하지만 좀 초심을 잃어버린 생각이 들었습니다. 그리고 새로 알게된걸 항상 포스팅으로 하게되지는 않고 그러다 보면 포스팅타이밍을 놓쳐버린 것들도 많죠.

쿠이님의 질문을 받으면서 생각난 김에 JS로 엘리먼트를 다룰때 가장 많이 사용할 추가, 삭제에서 prototype.js가 지원하는 부분에 대해서 다시 한번 정리합니다.

prototype.js에는 Element 클래스가 있습니다. HTML의 DIV, INPUT, LI같은 엘리먼트에 관련된 클래스입니다. Element의 API 문서를 보면Element클래스가 지원하는 많은 메서드를 볼 수 있습니다. 엘리먼트와 관련된 속성에 대한 것이라든지, 추가, 삭제, 생성등등의 기능을 지원하고 있습니다.

일반적인 방법

Html

```
<div id="testDivision" name="testDivision">
  <a href="#">Link 1</a>
</div>
```

간단한 HTML을 가지고 보겠습니다. 엘리먼트를 핸들링할때 JS에서 제공하는 2가지방법이 있습니다. innerHTML을 이용하는 방법과 DOM을 이용하는 방법입니다.

일반적으로는 innerHTML을 이용하는 것이 간단하면서도 편하기 때문에 많이 사용합니다.

JavaScript

```
var htmlStr = "<a href='#'>Link 2</a>";
var obj = document.getElementById("testDivision");
obj.innerHTML = obj.innerHTML + htmlStr;
```

사실 설명할 내용도 없긴 하지만 innerHTML은 이름 그대로 해당 엘리먼트의 안에 있는 HTML을 의미하고 있습니다. 내부 HTML을 새로지정하면 새로 지정하는 내용으로 replace가 됩니다. 기존의 innerHTML값과 새로추가할 HTML스트링을 문자열로 이어붙여서 다시 innerHTML으로 할당해서 HTML의 구조를 동적으로 바꾸는 것입니다. js를 실행하면 HTML이 아래와 같은 모습입니다.

Html

```
<div id="testDivision" name="testDivision">
  <a href="#">Link 1</a>
  <a href="#">Link 2</a>
</div>
```

두번째 방법은 DOM입니다. DOM은 Document Object Model의 약자로 표준이 정해져 있고 대부분의 브라우저가 지원하고 있습니다. 엘리먼트를 추가/삭제같은 DOM Level 2에 정의가 되어 있습니다.(DOM Level 1에서 추가확장되어 현재는 대부분 Level 2를 지원하고 있는 것으로 알고 있습니다.) 이름대로 DOM을 다루기 위해서 만들어진 것이기 때문에 다양한 메서드를 제공하고 있고 좀 더 세련된 느낌의 사용법을 가지고 있습니다.

JavaScript

```
var elem = document.createElement("a")
elem.setAttribute("href", "#");
elem.appendChild(document.createTextNode("Link 2"));
document.getElementById("testDivision").appendChild(elem);
```

같은 기능을 DOM으로 구현한 것입니다. 엘리먼트 만들고 속성추가하고 TextNode만들어서 A엘리먼트 안에 넣은 다음에 추가한 것입니다. 메서드로 다양하게 다룰 수 있기는 하지만 저 간단한 소스도 소스로는 꽤 복잡하게 작성되고 있고 한눈에 어떻게 동작할 것인지를 파악하기가 쉽지 않고 복잡한 HTML의 경우에는 훨씬 복잡하게 됩니다. innerHTML에 비해서 특정엘리먼트만 지

우는 remove관련 메서드도 지원하고 있기는 하지만 브라우저의 따라 DOM Level 1/2의 지원차이가 있습니다.

DOM은 Dom Test Pages에서 지원여부를 테스트해 볼 수 있고 PPK의 W3C DOM Compatibility에서 브라우저별 차이를 파악할 수 있습니다.

prototype.js 이용하기

당연히 prototype.js를 로드해야하고 그 후에 다음과 같이 사용하면 됩니다.
JavaScript

```
$("#testDivision").insert("<a href='#'>Link 2</a>");
$("#testDivision").update("<a href='#'>Link 2</a>");
$("#testDivision").remove();
```

너무 간단한 코드라서 그냥 한 코드블럭에 다 썼습니다. 첫줄 insert만으로 위에서 설명했던 기능과 동일한 기능을 합니다. update는 replace의 기능을 합니다. 2번째 줄을 실행하면 Div안에 link2만 존재하게 되었고 remove를 실행하면 testDivision DIV 자체가 삭제되어버립니다. 간단한 기능만 소개했지만 그외에도 Element관련 다양한 기능을 제공하고 있습니다. prototype.js내부적으로 크로스 브라우징을 지원하기 때문에 브라우저호환에 대해서 큰 걱정없이 사용할 수 있습니다.

추가적으로 좀 더 세밀하게 컨트롤 할 수 있는 기능을 제공하고 있습니다.
JavaScript

```
$("#testDivision").insert({top:"<a href='#'>Link 2</a>"});
// 결과
// <div id="testDivision" name="testDivision">
//   <a href="#">Link 2</a><a href="#">Link 1</a>
// </div>

$("#testDivision").insert({bottom:"<a href='#'>Link 2</a>"});
// 결과
// <div id="testDivision" name="testDivision">
//   <a href="#">Link 1</a><a href="#">Link 2</a>
// </div>

$("#testDivision").insert({before:"<a href='#'>Link 2</a>"});
// 결과
// <a href="#">Link 2</a>
// <div id="testDivision" name="testDivision">
//   <a href="#">Link 1</a>
// </div>

$("#testDivision").insert({after:"<a href='#'>Link 2</a>"});
// 결과
// <div id="testDivision" name="testDivision">
//   <a href="#">Link 1</a>
// </div>
// <a href="#">Link 2</a>
```

위 코드처럼 JSON형태로 사용해서 insert되는 위치를 지정해 줄 수 있습니다. 사용할 수 있는 position값은 top, bottom, before, after입니다. 이것 이용하면 쉽게 원하는 위치에 엘리먼트의 삽입/삭제가 가능합니다.

좀 복잡한 HTML의 경우에는 script.aculo.us의 Builder을 사용하는 것도 괜찮은 방법 같습니다.

10만원에 프로그래밍 해드립니다 - 크

웹/어플/워드프레스/코딩/프로그램 개발 국내랭키 1위, 30만건 완료 kmon

📅 2009/08/11 02:43

📁 Javascript/Framework

🔖 DOM, Element, innerHTML, Prototype Framework, prototype.js, script.aculo.us



트윗

좋아요 0개





Comments List

1. Jasper | 2011/08/04 18:12 + M/D Reply
좋은글 잘 보았습니다.

마지막 예제코드 14번 라인의...
\$("testDivision").insert({bottom:"Link 2"});
를

\$("testDivision").insert({before:"Link 2"});
로


해야 할 것 같은데요.
1. Outsider | 2011/08/05 11:19 + M/D
아~ 그러네요... 소스에 오류가 있었군요. 수정해 놓겠습니다. 감사합니다.

Facebook Comments

댓글 0개


정렬 기준

날짜 오름차순

 댓글 달기...

Facebook 댓글 플러그인

- My Github My Twitter My Facebook My LinkedIn My portfolio
- 전체 글 보기
- ATOM List 1314
 - ATOM BlaBlaBla~ 173
 - ATOM JAVA 184
 - ATOM Java 60
 - ATOM Framework 81

- ATOM Tools 43
- ATOM Scala 62
- ATOM .NET 21
- ATOM ASP 12
- ATOM ASP.NET 2.0 7
- ATOM C# 2
- ATOM PHP 1
- ATOM Database 36
- ATOM Commons 9
- ATOM MS SQL 9
- ATOM NoSQL 11
- ATOM Oracle 4
- ATOM PostgreSQL 2
- ATOM etc... 1
- ATOM Programming 227
- ATOM Publishing 57
- ATOM Javascript 177
- ATOM Javascript 110
- ATOM Framework 52
- ATOM Library 17
- ATOM node.js 128
- ATOM CoffeeScript 10
- ATOM Python 5
- ATOM Ruby on Rails 11
- ATOM RIA 10
- ATOM Silverlight 9
- ATOM Flex & AIR 1
- ATOM Web 2.0 & Semantic 48
- ATOM Ubuntu 8
- ATOM Mobile 26
- ATOM Mobile 2
- ATOM Mobile Web 6
- ATOM Android 6
- ATOM iOS 13
- ATOM DevOps 36
- ATOM Newsletter 91
- 2017/12
- 2017/11
- 2017/10
- 2017/09
- 2017/08
- 방명록
-  RSS