JavaScript: la proprietà innerHTML

`innerHTML` è una proprietà non standard in lettura e scrittura introdotta originariamente in Internet Explorer ed in seguito adottata dagli altri browser. La particolarità di questa proprietà è che i browser usano il loro parser HTML quando accedono a tale proprietà. Come molti sanno, questa proprietà restituisce in lettura una stringa contenente il markup dell'elemento selezionato e imposta il contenuto HTML di un elemento come stringa in scrittura. Poichè questa proprietà richiede l'uso del parser HTML da parte di un browser, sono necessari alcuni accorgimenti per renderne più efficace l'utilizzo.

Performance

In scrittura non facciamo altro che assemblare una stringa da usare con questa proprietà. Tuttavia occorre prestare attenzione a come `innerHTML` viene usata. Per esempio, il seguente codice:

```
for(var i = 0; i < 10; i++) {
  element.innerHTML += '<li>' + i '</li>';
}
```

ci penalizza in termini di performance, in quanto usiamo questa proprietà dieci volte e quindi abbiamo dieci chiamate al parser HTML. Invece, dobbiamo usare un approccio incrementale:

```
var html = '':
for(var i = 0; i < 10; i++) {
  html += '<li>' + i + '</li>';
```

```
}
element.innerHTML = html;
```

In questo caso, invece, abbiamo una sola chiamata al parser HTML con un notevole miglioramento della performance.

Marcatura ben formata

Questa proprietà genera una nuova struttura DOM all'interno di un elemento, quindi è di vitale importanza che la stringa usata contenga marcatura ben formata, vale a dire con i tag correttamente annidati. Ricordiamo che se serviamo il nostro documento come `application/xhtml+xml` un errore del genere farà fallire il nostro script. Inoltre Internet Explorer potrebbe restituire un errore nel caso di marcatura malformata.

Spazio bianco

In lettura `innerHTML` restituisce la marcatura così com'è, incluso lo spazio bianco in essa presente. Caratteri come `\t`, `\r`, `\n`, `\s` che fanno parte dello spazio bianco vengono quindi restituiti insieme ai tag.

Possiamo normalizzare la stringa restituita usando le espressioni regolari e il metodo `replace()`:

```
var html = document.getElementById('test').innerHTML;
var trimSpace = html.replace(/^\s+|\s+$/g, '');
```

Quando si usa il metodo `replace()` si raccomanda di usare il flag g nell'espressione regolare al fine di operare una ricerca su tutta la stringa, senza fermarsi alla prima ricorrenza del pattern usato.