## Bạn không cần jQuery nữa đâu

Ngày nay, môi trường lập trình front-end phát triển rất nhanh chóng, các trình duyệt hiện đại đã cung cấp các API đủ tốt để làm việc với DOM/BOM. Bạn không còn cần phải học về jQuery nữa. Đồng thời, nhờ sự ra đời của các thư viện như React, Angular và Vue đã khiến cho việc can thiệp trực tiếp vào DOM trở thành một việc không tốt. jQuery đã không còn quan trọng như trước nữa. Bài viết này tổng hợp những cách để thay thế các hàm của jQuery bằng các hàm được hỗ trợ bởi trình duyệt, và hó cũng hoạt động trên IE 10+

## Danh mục



## Query Selector

Đối với những selector phổ biến như class, id hoặc thuộc tính thì chúng ta có thể sử dụng `document.querySelector` hoặc `document.querySelectorAll` để thay thế cho jQuery selector. Sự khác biệt của hai hàm này là ở chỗ:

- `document.querySelector` trả về element đầu tiên được tìm thấy
- `document.querySelectorAll` trả về tất cả các element được tìm thấy dưới dạng một instance của NodeList. Nó có thể được convert qua array bằng cách `[].slice.call(document.querySelectorAll(selector) || []);`
- Nếu không có element nào được tìm thấy, thì jQuery sẽ trả về một array rỗng `[]` trong khi đó DOM API sẽ trả về `null`. Hãy chú ý đến Null Pointer Exception. Bạn có thể sử dụng toán tử `||` để đặt giá trị default nếu như không có element nào được tìm thấy, ví dụ như `document.querySelectorAll(selector) || []`

> *Chú ý :* `document.querySelector` *và* `document.querySelectorAll` *hoạt động khá* ***CHẬM****, hãy thử dùng* `getElementById`, `document.getElementsByClassName` *hoặc* `document.getElementsByTagName` *nếu bạn muốn đạt hiệu suất tốt hơn.*

- 1.0 Query bằng selector

```
// jQuery
$('selector');

// Native
document.querySelectorAll('selector');
```

- 1.1 Query bằng class

```
// jQuery
$('.class');
```

```
// Native
document.querySelectorAll('.class');

// hoặc
document.getElementsByClassName('class');
```

- 1.2 Query bằng id

```
// jQuery
$('#id');

// Native
document.querySelector('#id');

// hoặc
document.getElementById('id');
```

- 1.3 Query bằng thuộc tính

```
// jQuery
$('a[target=_blank]');

// Native
document.querySelectorAll('a[target=_blank]');
```

- 1.4 Tìm bất cứ gì.
  - Tìm node

```
// jQuery
$el.find('li');

// Native
el.querySelectorAll('li');
```

  - Tìm body

```
// jQuery
$('body');

// Native
document.body;
```

  - lấy thuộc tính

```
// jQuery
$el.attr('foo');
```

```
// Native
e.getAttribute('foo');
```

- Lấy giá trị của thuộc tính `data`

```
// jQuery
$el.data('foo');

// Native
// using getAttribute
el.getAttribute('data-foo');
// you can also use `dataset` if only need to support IE 11+
el.dataset['foo'];
```

- 1.5 Tìm element cùng level/trước/sau
  - Element cùng level

```
// jQuery
$el.siblings();

// Native
[].filter.call(el.parentNode.children, function(child) {
  return child !== el;
});
```

  - Element ở phía trước

```
// jQuery
$el.prev();

// Native
el.previousElementSibling;
```

  - Element ở phía sau

```
// next
$el.next();
el.nextElementSibling;
```

- 1.6 Element gần nhất

  Trả về element đầu tiên có selector khớp với yêu cầu khi duyệt từ element hiện tại trở lên tới document.

```
// jQuery
$el.closest(queryString);

// Native
function closest(el, selector) {
```

```
  const matchesSelector = el.matches || el.webkitMatchesSelector ||
el.mozMatchesSelector || el.msMatchesSelector;

  while (el) {
    if (matchesSelector.call(el, selector)) {
      return el;
    } else {
      el = el.parentElement;
    }
  }
  return null;
}
```

- 1.7 Tìm parent

  Truy ngược một cách đệ quy tổ tiên của element hiện tại, cho đến khi tìm được một element tổ tiên ( element cần tìm ) mà element đó là con trực tiếp của element khớp với selector được cung cấp, Return lại element cần tìm đó.

```
// jQuery
$el.parentsUntil(selector, filter);

// Native
function parentsUntil(el, selector, filter) {
  const result = [];
  const matchesSelector = el.matches || el.webkitMatchesSelector ||
el.mozMatchesSelector || el.msMatchesSelector;

  // match start from parent
  el = el.parentElement;
  while (el && !matchesSelector.call(el, selector)) {
    if (!filter) {
      result.push(el);
    } else {
      if (matchesSelector.call(el, filter)) {
        result.push(el);
      }
    }
    el = el.parentElement;
  }
  return result;
}
```

- 1.8 Form
  - Input/Textarea

```
// jQuery
$('#my-input').val();
```

```
// Native
document.querySelector('#my-input').value;
```

- Lấy index của e.currentTarget trong danh sách các element khớp với selector `.radio`

```
// jQuery
$(e.currentTarget).index('.radio');

// Native
[].indexOf.call(document.querySelectAll('.radio'), e.currentTarget);
```

- 1.9 Nội dung Iframe

  `$('iframe').contents()` trả về thuộc tính `contentDocument` của iframe được tìm thấy

  - Nọi dung iframe

```
// jQuery
$iframe.contents();

// Native
iframe.contentDocument;
```

  - Query Iframe

```
// jQuery
$iframe.contents().find('.css');

// Native
iframe.contentDocument.querySelectorAll('.css');
```

**↑ Trở về đầu**

## CSS & Style

- 2.1 CSS

  - Lấy style

```
// jQuery
$el.css("color");

// Native
// NOTE: Bug đã được biết, sẽ trả về 'auto' nếu giá trị của style là
'auto'
const win = el.ownerDocument.defaultView;
// null means not return presudo styles
win.getComputedStyle(el, null).color;
```

- Đặt style

```
// jQuery
$el.css({ color: "#ff0011" });

// Native
el.style.color = '#ff0011';
```

- Lấy/Đặt Nhiều style

  Nếu bạn muốn đặt nhiều style một lần, bạn có thể sẽ thích phương thức setStyles trong thư viện oui-dom-utils.

- Thêm class và element

```
// jQuery
$el.addClass(className);

// Native
el.classList.add(className);
```

- Loại bỏ class class ra khỏi element

```
// jQuery
$el.removeClass(className);

// Native
el.classList.remove(className);
```

- Kiểm tra xem element có class nào đó hay không

```
// jQuery
$el.hasClass(className);

// Native
el.classList.contains(className);
```

- Toggle class

```
// jQuery
$el.toggleClass(className);

// Native
el.classList.toggle(className);
```

- 2.2 Chiều rộng, chiều cao

  Về mặt lý thuyết thì chiều rộng và chiều cao giống như nhau trong cả jQuery và DOM API:

  - Chiều rộng của window

```
// window height
$(window).height();
// trừ đi scrollbar
window.document.documentElement.clientHeight;
// Tính luôn scrollbar
window.innerHeight;
```

- Chiều cao của Document

```
// jQuery
$(document).height();

// Native
document.documentElement.scrollHeight;
```

- Chiều cao của element

```
// jQuery
$el.height();

// Native
function getHeight(el) {
  const styles = this.getComputedStyles(el);
  const height = el.offsetHeight;
  const borderTopWidth = parseFloat(styles.borderTopWidth);
  const borderBottomWidth = parseFloat(styles.borderBottomWidth);
  const paddingTop = parseFloat(styles.paddingTop);
  const paddingBottom = parseFloat(styles.paddingBottom);
  return height - borderBottomWidth - borderTopWidth - paddingTop -
paddingBottom;
}
// chính xác tới số nguyên (khi có thuộc tính `box-sizing` là `border-
box`, nó là `height`; khi box-sizing là `content-box`, nó là `height +
padding + border`)
el.clientHeight;
// Chính xác tới số thập phân (khi `box-sizing` là `border-box`, nó là
`height`; khi `box-sizing` là `content-box`, nó là `height + padding +
border`)
el.getBoundingClientRect().height;
```

- 2.3 Position & Offset

- Position

```
// jQuery
$el.position();

// Native
{ left: el.offsetLeft, top: el.offsetTop }
```

- Offset

```
// jQuery
$el.offset();

// Native
function getOffset (el) {
  const box = el.getBoundingClientRect();

  return {
    top: box.top + window.pageYOffset -
document.documentElement.clientTop,
    left: box.left + window.pageXOffset -
document.documentElement.clientLeft
  }
}
```

- 2.4 Scroll Top

```
// jQuery
$(window).scrollTop();

// Native
(document.documentElement && document.documentElement.scrollTop) ||
document.body.scrollTop;
```

**↑ Trở về đầu**

## Thao tác với DOM

- 3.1 Loại bỏ

```
// jQuery
$el.remove();

// Native
el.parentNode.removeChild(el);
```

- 3.2 Text
  - Lấy text

```
// jQuery
$el.text();

// Native
el.textContent;
```

  - Đặt giá trị text

```
// jQuery
$el.text(string);

// Native
el.textContent = string;
```

- 3.3 HTML
  - Lấy HTML

```
// jQuery
$el.html();

// Native
el.innerHTML;
```

  - Đặt giá trị HTML

```
// jQuery
$el.html(htmlString);

// Native
el.innerHTML = htmlString;
```

- 3.4 Append

  append một element sau element con cuối cùng của element cha

```
// jQuery
$el.append("<div id='container'>hello</div>");

// Native
let newEl = document.createElement('div');
newEl.setAttribute('id', 'container');
newEl.innerHTML = 'hello';
el.appendChild(newEl);
```

- 3.5 Prepend

```
// jQuery
$el.prepend("<div id='container'>hello</div>");

// Native
let newEl = document.createElement('div');
newEl.setAttribute('id', 'container');
newEl.innerHTML = 'hello';
el.insertBefore(newEl, el.firstChild);
```

- 3.6 insertBefore

  Chèn một node vào trước element được query.

  ```
  // jQuery
  $newEl.insertBefore(queryString);

  // Native
  const target = document.querySelector(queryString);
  target.parentNode.insertBefore(newEl, target);
  ```

- 3.7 insertAfter

  Chèn node vào sau element được query

  ```
  // jQuery
  $newEl.insertAfter(queryString);

  // Native
  const target = document.querySelector(queryString);
  target.parentNode.insertBefore(newEl, target.nextSibling);
  ```

**↑ Trở về đầu**

## Ajax

Thay thế bằng fetch và fetch-jsonp

**↑ Trở về đầu**

## Events

Để có một sự thay thế đầy đủ nhất, bạn nên sử dụng https://github.com/oneuijs/oui-dom-events

- 5.1 Bind event bằng on

  ```
  // jQuery
  $el.on(eventName, eventHandler);

  // Native
  el.addEventListener(eventName, eventHandler);
  ```

- 5.2 Unbind event bằng off

  ```
  // jQuery
  $el.off(eventName, eventHandler);

  // Native
  el.removeEventListener(eventName, eventHandler);
  ```

- 5.3 Trigger

```
// jQuery
$(el).trigger('custom-event', {key1: 'data'});

// Native
if (window.CustomEvent) {
  const event = new CustomEvent('custom-event', {detail: {key1: 'data'}});
} else {
  const event = document.createEvent('CustomEvent');
  event.initCustomEvent('custom-event', true, true, {key1: 'data'});
}

el.dispatchEvent(event);
```

**↑ Trở về đầu**

## Hàm tiện ích

- 6.1 isArray

```
// jQuery
$.isArray(range);

// Native
Array.isArray(range);
```

- 6.2 Trim

```
// jQuery
$.trim(string);

// Native
string.trim();
```

- 6.3 Object Assign

  Mở rộng, sử dụng object.assign https://github.com/ljharb/object.assign

```
// jQuery
$.extend({}, defaultOpts, opts);

// Native
Object.assign({}, defaultOpts, opts);
```

- 6.4 Contains

```
// jQuery
$.contains(el, child);
```

```
// Native
el !== child && el.contains(child);
```

**↑ Trở về đầu**

## Ngôn ngữ khác

- 한국어
- 简体中文
- Bahasa Melayu
- Português(PT-BR)
- Tiếng Việt Nam
- Русский
- Türkçe

## Các trình duyệt hỗ trợ

| Chrome | Firefox | IE | Opera | Safari |
|---|---|---|---|---|
| Latest ✓ | Latest ✓ | 10+ ✓ | Latest ✓ | 6.1+ ✓ |

# Giấy phép