How to detect Safari, Chrome, IE, Firefox and Opera browsers?

Asked 12 years ago Modified 1 month ago Viewed 1.2m times

1085

I have 5 addons/extensions for Firefox, Chrome, Internet Explorer(IE), Opera, and Safari.

How can I correctly recognize the user browser and redirect (once an install button has been clicked) to download the corresponding addon?

javascript browser-detection

Share Edit Follow

edited Jul 10, 2022 at 22:22
IT goldman
16.2k ●2 ●14 ●29

asked Mar 23, 2012 at 23:06
FrankC
11.3k ●4 ●17 ●21

4 try detectjs, it can be used for all browsers – dude Mar 9, 2016 at 10:54

1 Possible duplicate of Browser detection in JavaScript? – Matthijs Wessels Sep 2, 2016 at 10:23

4 detect.js is no longer maintained (according to github.com/darcyclarke/Detect.js), they recommend github.com/lancedikson/bowser – YakovL Apr 25, 2018 at 17:02

1 Possible duplicate of How can you detect the version of a browser? – KyleMit ♦ Feb 14, 2019 at 14:05

4 I'd like to re-surface this question - is there a reliable & simple answer to this yet? There are many (old) questions like this on SO and yet most of them work off the `userAgent` function which even w3schools acknowledges is inaccurate. I've tested some of the answers to this question and similar ones, and none of them seem reliable. If I'm better off asking a new question please let me know. – AutoBaker Mar 17, 2021 at 16:20

32 Answers

Sorted by:

Highest score (default)

1 2 Next

2003

Googling for browser reliable detection often results in checking the User agent string. This method is **not** reliable, because it's trivial to spoof this value.
I've written a method to detect browsers by duck-typing.

Only use the browser detection method if it's truly necessary, such as showing browser-specific instructions to install an extension. **Use feature detection when possible.**

Demo: https://jsfiddle.net/6spj1059/

```
// Opera 8.0+
var isOpera = (!!window.opr && !!opr.addons) || !!window.opera ||
navigator.userAgent.indexOf(' OPR/') >= 0;

// Firefox 1.0+
var isFirefox = typeof InstallTrigger !== 'undefined';

// Safari 3.0+ "[object HTMLElementConstructor]"
var isSafari = /constructor/i.test(window.HTMLElement) || (function (p) {
return p.toString() === "[object SafariRemoteNotification]"; })
(!window['safari'] || (typeof safari !== 'undefined' &&
window['safari'].pushNotification));

// Internet Explorer 6-11
var isIE = /*@cc_on!@*/false || !!document.documentMode;

// Edge 20+
var isEdge = !isIE && !!window.StyleMedia;

// Chrome 1 - 79
var isChrome = !!window.chrome && (!!window.chrome.webstore ||
!!window.chrome.runtime);

// Edge (based on chromium) detection
var isEdgeChromium = isChrome && (navigator.userAgent.indexOf("Edg") != -1);

// Blink engine detection
var isBlink = (isChrome || isOpera) && !!window.CSS;


var output = 'Detecting browsers by ducktyping:<hr>';
output += 'isFirefox: ' + isFirefox + '<br>';
output += 'isChrome: ' + isChrome + '<br>';
output += 'isSafari: ' + isSafari + '<br>';
output += 'isOpera: ' + isOpera + '<br>';
output += 'isIE: ' + isIE + '<br>';
output += 'isEdge: ' + isEdge + '<br>';
output += 'isEdgeChromium: ' + isEdgeChromium + '<br>';
output += 'isBlink: ' + isBlink + '<br>';
document.body.innerHTML = output;
```

▶ Run code snippet Expand snippet

Analysis of reliability

The previous method depended on properties of the rendering engine (`-moz-box-sizing` and `-webkit-transform`) to detect the browser. These prefixes will eventually be dropped, so to make detection even more robust, I switched to browser-specific characteristics:

- Internet Explorer: JScript's Conditional compilation (up until IE9) and `document.documentMode`.
- Edge: In Trident and Edge browsers, Microsoft's implementation exposes the `StyleMedia` constructor. Excluding Trident leaves us with Edge.
- Edge (based on chromium): The user agent include the value "Edg/[version]" at the end (ex: "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.16 Safari/537.36 **Edg/80.0.361.9**").
- Firefox: Firefox's API to install add-ons: `InstallTrigger`

- Chrome: The global `chrome` object, containing several properties including a documented `chrome.webstore` object.
- Update 3 `chrome.webstore` is deprecated and undefined in recent versions
- Safari: A unique naming pattern in its naming of constructors. This is the least durable method of all listed properties and guess what? In Safari 9.1.3 it was fixed. So we are checking against `SafariRemoteNotification`, which was introduced after version 7.1, to cover all Safaris from 3.0 and upwards.
- Opera: `window.opera` has existed for years, but will be dropped when Opera replaces its engine with Blink + V8 (used by Chromium).
- Update 1: Opera 15 has been released, its UA string looks like Chrome, but with the addition of "OPR". In this version the `chrome` object is defined (but `chrome.webstore` isn't). Since Opera tries hard to clone Chrome, I use user agent sniffing for this purpose.
- Update 2: `!!window.opr && opr.addons` can be used to detect Opera 20+ (evergreen).
- Blink: `CSS.supports()` was introduced in Blink once Google switched on Chrome 28. It's of course, the same Blink used in Opera.

Successfully tested in:

- Firefox 0.8 - 61
- Chrome 1.0 - 71
- Opera 8.0 - 34
- Safari 3.0 - 10
- IE 6 - 11
- Edge - 20-42
- Edge Dev - 80.0.361.9

> Updated in November 2016 to include detection of Safari browsers from 9.1.3 and upwards
>
> Updated in August 2018 to update the latest successful tests on chrome, firefox IE and edge.
>
> Updated in January 2019 to fix chrome detection (because of the window.chrome.webstore deprecation) and include the latest successful tests on chrome.
>
> Updated in December 2019 to add Edge based on Chromium detection (based on the @Nimesh comment).

Share Edit Follow

edited Nov 23, 2020 at 20:38

community wiki
25 revs, 16 users 39%
Rob W

10 FYI This doesn't work with Chrome Extensions as `window.chrome.webstore` is undefined there. Haven't checked it with Firefox Extensions. `is.js` mentioned elsewhere does work in both Chrome and Firefox Extensions. – nevf Aug 31, 2016 at 7:11

87 `isSafari` doesn't work with Safari 10. I'm going to argue this is a bad solution (not that I have a good one). There's no guarantee many of the things your checking for won't be removed OR won't be added by other browsers. Every site that was using this code for check for Safari just broke with macOS Sierra or Safari 10 upgrades :(– gman Sep 29, 2016 at 5:16

13 But has this been tested on the **mobile versions** of those browsers as well as the **desktop versions** too? And truthfully, there are different mobile versions and different desktop versions per platform. So really, firefox has 3 binaries for Windows, Linux, Mac OS, and 2 binaries for Android and iOS. – DrZ214 Dec 23, 2016 at 3:53

6 The current `isSafari` does not work under `<iframe>` under Safari 10.1.2 – Mikko Ohtamaa Sep 14, 2017 at 19:24

37 window.chrome.webstore is deprecated starting from Chrome ver. 71: blog.chromium.org/2018/06/... – st_bk Oct 29, 2018 at 6:48

211

You can try the following way to check the browser:

```
function myFunction() {
  if ((navigator.userAgent.indexOf("Opera") ||
navigator.userAgent.indexOf('OPR')) != -1) {
    alert('Opera');
  } else if (navigator.userAgent.indexOf("Edg") != -1) {
    alert('Edge');
  } else if (navigator.userAgent.indexOf("Chrome") != -1) {
    alert('Chrome');
  } else if (navigator.userAgent.indexOf("Safari") != -1) {
    alert('Safari');
  } else if (navigator.userAgent.indexOf("Firefox") != -1) {
    alert('Firefox');
  } else if ((navigator.userAgent.indexOf("MSIE") != -1) ||
(!!document.documentMode == true)) //IF IE > 10
  {
    alert('IE');
  } else {
    alert('unknown');
  }
}
```

```
<!DOCTYPE html>
<html>

<body>
  <p>What is the name(s) of your browser?</p>
  <button onclick="myFunction()">Try it</button>
  <p id="demo"></p>

</body>

</html>
```

▶ Run code snippet   Expand snippet

And if you need to know only IE Browser version then you can follow below code. This code works well for version IE6 to IE11

```
function getInternetExplorerVersion() {
  var ua = window.navigator.userAgent;
  var msie = ua.indexOf("MSIE ");
  var rv = -1;

  if (msie > 0 || !!navigator.userAgent.match(/Trident.*rv\:11\./)) // If
Internet Explorer, return version number
  {

    if (isNaN(parseInt(ua.substring(msie + 5, ua.indexOf(".", msie))))) {
      //For IE 11 >
      if (navigator.appName == 'Netscape') {
        var ua = navigator.userAgent;
        var re = new RegExp("Trident/.*rv:([0-9]{1,}[\.0-9]{0,})");
        if (re.exec(ua) != null) {
          rv = parseFloat(RegExp.$1);
          alert(rv);
        }
      } else {
        alert('otherbrowser');
      }
    } else {
      //For < IE11
      alert(parseInt(ua.substring(msie + 5, ua.indexOf(".", msie))));
    }
    return false;
  }
}
```

```
<!DOCTYPE html>
<html>

<body>

  <p>Click on Try button to check IE Browser version.</p>

  <button onclick="getInternetExplorerVersion()">Try it</button>

  <p id="demo"></p>

</body>

</html>
```

▶ Run code snippet   Expand snippet

Share Edit Follow

edited Sep 16, 2023 at 21:28
corn on the cob
2,206 ● 3 ● 21 ● 32

answered Oct 14, 2014 at 10:44
Nimesh
3,510 ● 1 ● 29 ● 36

5 Why would one write so many lines of code? userAgent is ambiguous. – igauravsehrawat Oct 15, 2015 at

7:00

7 What about Microsoft Edge? – user6031759 Jun 15, 2016 at 23:25

4 the answer above checks for chrome before checking for safari. because safari will not have `chrome` keyword in the useragent. example of safari useragent - `mozilla/5.0 (macintosh; intel mac os x 10_11_5) applewebkit/601.6.17 (khtml, like gecko) version/9.1.1 safari/601.6.17` – Golak Sarangi Aug 17, 2016 at 4:07

30 Stackoverflow uses this method – vityavv Sep 7, 2016 at 14:02

7 When testing this in Opera (latest version), this returns 'Chrome' for me. To fix this, I changed the Opera if statement to: `if(navigator.userAgent.indexOf("Opera") != -1 || navigator.userAgent.indexOf('OPR') != -1 )` – Kyle Vassella Dec 5, 2017 at 23:05

83

Here are several prominent libraries that handle browser detection as of Dec 2019.

Bowser by lancedikson - 4,065★s - Last updated Oct 2, 2019 - 4.8KB

```
var result = bowser.getParser(window.navigator.userAgent);
console.log(result);
document.write("You are using " + result.parsedResult.browser.name +
               " v" + result.parsedResult.browser.version +
               " on " + result.parsedResult.os.name);
```

```
<script src="https://unpkg.com/bowser@2.7.0/es5.js"></script>
```

▶ Run code snippet   Expand snippet

**supports Edge based on Chromium*

Platform.js by bestiejs - 2,550★s - Last updated Apr 14, 2019 - 5.9KB

```
console.log(platform);
document.write("You are using " + platform.name +
               " v" + platform.version +
               " on " + platform.os);
```

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/platform/1.3.5/platform.min.js">
</script>
```

▶ Run code snippet   Expand snippet

jQuery Browser by gabceb - 504★s - Last updated Nov 23, 2015 - 1.3KB

```
console.log($.browser)
document.write("You are using " + $.browser.name +
               " v" + $.browser.versionNumber +
               " on " + $.browser.platform);
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js">
</script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-
browser/0.1.0/jquery.browser.min.js"></script>
```

▶ Run code snippet   Expand snippet

Detect.js (Archived) by darcyclarke - 522★s - Last updated Oct 26, 2015 - 2.9KB

```
var result = detect.parse(navigator.userAgent);
console.log(result);
document.write("You are using " + result.browser.family +
               " v" + result.browser.version +
               " on " + result.os.family);
```

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/Detect.js/2.2.2/detect.min.js">
</script>
```

▶ Run code snippet   Expand snippet

Browser Detect (Archived) by QuirksMode - Last updated Nov 14, 2013 - 884B

```
console.log(BrowserDetect)
document.write("You are using " + BrowserDetect.browser +
               " v" + BrowserDetect.version +
               " on " + BrowserDetect.OS);
```

```
<script src="https://kylemit.github.io/libraries/libraries/BrowserDetect.js">
</script>
```

▶ Run code snippet   Expand snippet

Notable Mentions:

- WhichBrowser - 1,355★s - Last updated Oct 2, 2018

- Modernizr - 23,397★s - Last updated Jan 12, 2019 - To feed a fed horse, feature detection should drive any canIuse style questions. Browser detection is really just for providing customized images, download files, or instructions for individual browsers.

Further Reading

- Stack Overflow - Browser detection in JavaScript?
- Stack Overflow - How can you detect the version of a browser?

Share Edit Follow

edited Dec 2, 2019 at 15:09

answered Mar 30, 2018 at 21:13
KyleMit ♦
34.5k ● 70 ● 477 ● 685

5 Well worth a few Kb of overhead to not reinvent the wheel. – 111 Mar 13, 2020 at 21:36

72

I know it may be overkill to use a lib for that, but just to enrich the thread, you could check is.js way of doing this:

```
is.firefox();
is.ie(6);
is.not.safari();
```

Share Edit Follow

answered May 13, 2015 at 20:55
Rafael Eyng
4,900 ● 2 ● 35 ● 36

15 Just worth noting that under the hood it's primarily doing User-Agent detection. Supplemented with vendor detection / some feature detection in places. – TygerKrash Oct 20, 2015 at 9:50

1 @TygerKrash sure, you are absolutely right. That is actually what I meant with my answer: open the source code of `is.js` and check how they do it. – Rafael Eyng Apr 17, 2016 at 16:26

4 jQuery used to include similar properties: $.browser.msie... Were removed from version 1.9 api.jquery.com/jquery.browser – Ricardo stands with Ukraine Jun 29, 2016 at 9:08

1 This is definitely the most robust approach when you assume that the UA string hasn't been modified. It also properly detects OS (android, win, mac, linux); device type (desktop, tablet, mobile). It can also test for version of browser. – kashiraja Feb 17, 2019 at 7:23

50

In case anyone finds this useful, I've made Rob W's answer into a function that returns the browser string rather than having multiple variables floating about. Since the browser also can't really change without loading all over again, I've made it cache the result to prevent it from needing to work it out the next time the function is called.

```
/**
 * Gets the browser name or returns an empty string if unknown.
 * This function also caches the result to provide for any
 * future calls this function has.
 *
 * @returns {string}
 */
var browser = function() {
    // Return cached result if avalible, else get result then cache it.
    if (browser.prototype._cachedResult)
        return browser.prototype._cachedResult;

    // Opera 8.0+
    var isOpera = (!!window.opr && !!opr.addons) || !!window.opera ||
navigator.userAgent.indexOf(' OPR/') >= 0;

    // Firefox 1.0+
    var isFirefox = typeof InstallTrigger !== 'undefined';

    // Safari 3.0+ "[object HTMLElementConstructor]"
    var isSafari = /constructor/i.test(window.HTMLElement) || (function (p) {
return p.toString() === "[object SafariRemoteNotification]"; })
(!window['safari'] || safari.pushNotification);

    // Internet Explorer 6-11
    var isIE = /*@cc_on!@*/false || !!document.documentMode;

    // Edge 20+
    var isEdge = !isIE && !!window.StyleMedia;

    // Chrome 1+
    var isChrome = !!window.chrome && !!window.chrome.webstore;

    // Blink engine detection
    var isBlink = (isChrome || isOpera) && !!window.CSS;

    return browser.prototype._cachedResult =
        isOpera ? 'Opera' :
        isFirefox ? 'Firefox' :
        isSafari ? 'Safari' :
        isChrome ? 'Chrome' :
        isIE ? 'IE' :
        isEdge ? 'Edge' :
        isBlink ? 'Blink' :
        "Don't know";
};

console.log(browser());
```

▶ Run code snippet   Expand snippet

Share Edit Follow

edited Oct 26, 2017 at 21:19
HoldOffHunger
19.8k ● 10 ● 111 ● 139

answered Jul 17, 2015 at 15:19
Will Squire
6,307 ● 7 ● 46 ● 62

3 in Edge browser, it returns *Chrome* – Riz Dec 29, 2016 at 11:25

3 @eFriend I updated the answer to the latest browser tests. – pilau Feb 8, 2017 at 10:11

5 I like this, but I would have preferred a fallback to userAgent(), instead of "Don't know". – HoldOffHunger Oct 26, 2017 at 19:39

6 Property `window.chrome.webstore` was removed in Chrome 71, so this approach is no longer working.

– Bedla Dec 19, 2018 at 10:34 ✎

2 I tried this in Safari, Firfox, and Chrome, always returns `Don't know`. – nightrain Jun 1, 2020 at 19:40

▲

49

▼

Short variant (**update 10 july 2020 mobile browser detection fix**)

```
var browser = (function() {
    var test = function(regexp) {return
regexp.test(window.navigator.userAgent)}
    switch (true) {
        case test(/edg/i): return "Microsoft Edge";
        case test(/trident/i): return "Microsoft Internet Explorer";
        case test(/firefox|fxios/i): return "Mozilla Firefox";
        case test(/opr\//i): return "Opera";
        case test(/ucbrowser/i): return "UC Browser";
        case test(/samsungbrowser/i): return "Samsung Browser";
        case test(/chrome|chromium|crios/i): return "Google Chrome";
        case test(/safari/i): return "Apple Safari";
        default: return "Other";
    }
})();
console.log(browser)
```

▶ Run code snippet   Expand snippet

Typescript version:

```
export enum BROWSER_ENUM {
  EDGE ,
  INTERNET_EXPLORER ,
  FIRE_FOX ,
  OPERA ,
  UC_BROWSER ,
  SAMSUNG_BROWSER ,
  CHROME ,
  SAFARI ,
  UNKNOWN ,
}

const testUserAgent = (regexp: RegExp): boolean =>
regexp.test(window.navigator.userAgent);

function detectBrowser(): BROWSER_ENUM {
  switch (true) {
    case testUserAgent(/edg/i): return BROWSER_ENUM.EDGE;
    case testUserAgent(/trident/i): return BROWSER_ENUM.INTERNET_EXPLORER;
    case testUserAgent(/firefox|fxios/i): return BROWSER_ENUM.FIRE_FOX;
    case testUserAgent(/opr\//i): return BROWSER_ENUM.OPERA;
    case testUserAgent(/ucbrowser/i): return BROWSER_ENUM.UC_BROWSER;
    case testUserAgent(/samsungbrowser/i): return BROWSER_ENUM.SAMSUNG_BROWSER;
    case testUserAgent(/chrome|chromium|crios/i): return BROWSER_ENUM.CHROME;
    case testUserAgent(/safari/i): return BROWSER_ENUM.SAFARI;
    default: return BROWSER_ENUM.UNKNOWN;
  }
}

export const BROWSER: BROWSER_ENUM = detectBrowser();
export const IS_FIREFOX = BROWSER === BROWSER_ENUM.FIRE_FOX;
```

Functional algorithm, just for fun:

```
 var BROWSER =  new Array(
     ["Microsoft Edge", /edg/i],
     ["Microsoft Internet Explorer", /trident/i],
     ["Mozilla Firefox", /firefox|fxios/i],
     ["Opera", /opr\//i],
     ["UC Browser", /ucbrowser/i],
     ["Samsung Browser", /samsungbrowser/i],
     ["Google Chrome", /chrome|chromium|crios/i],
     ["Apple Safari", /safari/i],
     ["Unknown", /.+/i],
 ).find(([, value]) => value.test(window.navigator.userAgent)).shift();
```

Share Edit Follow

edited Feb 21, 2023 at 6:41

answered Mar 23, 2018 at 11:02

Alex Nikulin

8,468 ● 4 ● 35 ● 37

5 this will show 'safari' while browsing with 'chrome' on ios – Reza Mar 13, 2019 at 19:06

2 This method worked with Chrome and Firefox on desktop and Safari on iOS. It fails with Chrome and Firefox on iOS. – BReddy Jul 9, 2020 at 1:09

1 @Gangula it currently reliable for all browsers that you see in a code)Microsoft Edge,Microsoft Internet Explorer,Opera,Firefox,Uc,samsung,chrome,safari – Alex Nikulin Oct 3, 2022 at 17:14

15

No idea if it is useful to anyone but here is a variant that would make TypeScript happy:

```
export function getBrowser() {

 // Opera 8.0+
    if ((!!window["opr"] && !!["opr"]["addons"]) || !!window["opera"] ||
navigator.userAgent.indexOf(' OPR/') >= 0) {
        return 'opera';
    }

    // Firefox 1.0+
    if (typeof window["InstallTrigger"] !== 'undefined') {
        return 'firefox';
    }

    // Safari 3.0+ "[object HTMLElementConstructor]"
    if (/constructor/i.test(window["HTMLElement"]) || (function(p) { return
p.toString() === "[object SafariRemoteNotification]"; })(!window['safari'] ||
(typeof window["safari"] !== 'undefined' &&
window["safari"].pushNotification))) {
        return 'safari';
    }

    // Internet Explorer 6-11
    if (/*@cc_on!@*/false || !!document["documentMode"]) {
        return 'ie';
    }
    // Edge 20+
    if (!(/*@cc_on!@*/false || !!document["documentMode"]) &&
!!window["StyleMedia"]) {
        return 'edge';
    }

    // Chrome 1+
    if (!!window["chrome"] && !!window["chrome"].webstore) {
        return 'chrome';
    }

    // Blink engine detection
    if (((!!window["chrome"] && !!window["chrome"].webstore) ||
((!!window["opr"] && !!["opr"]["addons"]) || !!window["opera"] ||
navigator.userAgent.indexOf(' OPR/') >= 0)) && !!window["CSS"]) {
        return 'blink';
    }
}
```

Share Edit Follow

edited Aug 9, 2020 at 2:05
Emma
27.5k ● 11 ● 44 ● 70

answered Dec 5, 2018 at 16:21
Tony Smith
889 ● 1 ● 12 ● 27

1 Why do you have some ifs with "false" as a condition? – CodeMonkey Oct 28, 2019 at 11:45

13

Thank you, everybody. I tested the code snippets here on the recent browsers: Chrome 55, Firefox 50, IE 11 and Edge 38, and I came up with the following combination that worked for me for all of them. I'm sure it can be improved, but it's a quick solution for whoever needs:

```
var browser_name = '';
isIE = /*@cc_on!@*/false || !!document.documentMode;
isEdge = !isIE && !!window.StyleMedia;
if(navigator.userAgent.indexOf("Chrome") != -1 && !isEdge)
{
    browser_name = 'chrome';
}
else if(navigator.userAgent.indexOf("Safari") != -1 && !isEdge)
{
    browser_name = 'safari';
}
else if(navigator.userAgent.indexOf("Firefox") != -1 )
{
    browser_name = 'firefox';
}
else if((navigator.userAgent.indexOf("MSIE") != -1 ) ||
(!!document.documentMode == true )) //IF IE > 10
{
    browser_name = 'ie';
}
else if(isEdge)
{
    browser_name = 'edge';
}
else
{
   browser_name = 'other-browser';
}
$('html').addClass(browser_name);
```

It adds a CSS class to the HTML, with the name of the browser.

Share Edit Follow

edited Apr 24, 2021 at 7:53

answered Jan 31, 2017 at 8:12

dnns
326 ●4 ●10

12

You can use `try` and `catch` to use the different browser error messages. IE and edge were mixed up, but I used the duck typing from Rob W (based on this project here: https://www.khanacademy.org/computer-programming/i-have-opera/2395080328).

```
var getBrowser = function() {
    try {
        var e;
        var f = e.width;
    } catch(e) {
        var err = e.toString();

        if(err.indexOf("not an object") !== -1) {
            return "safari";
        } else if(err.indexOf("Cannot read") !== -1) {
            return "chrome";
        } else if(err.indexOf("e is undefined") !== -1) {
            return "firefox";
        } else if(err.indexOf("Unable to get property 'width' of undefined or
null reference") !== -1) {
            if(!(false || !!document.documentMode) && !!window.StyleMedia) {
                return "edge";
            } else {
                return "IE";
            }
        } else if(err.indexOf("cannot convert e into object") !== -1) {
            return "opera";
        } else {
            return undefined;
        }
    }
};
```

Share Edit Follow

edited Feb 8, 2018 at 18:23

answered Oct 25, 2016 at 17:50

Mason Jones
161 ●2 ●7

1 That's a great idea: you don't need neither "window", nor "navigator" objects! – Vadim Dec 13, 2017 at 15:42

11

Here's a 2016 adjusted version of Rob's answer, including Microsoft Edge and detection of Blink:

(*edit*: I updated Rob's answer above with this information.)

```
// Opera 8.0+ (UA detection to detect Blink/v8-powered Opera)
isOpera = !!window.opera || navigator.userAgent.indexOf(' OPR/') >= 0;
// Firefox 1.0+
isFirefox = typeof InstallTrigger !== 'undefined';
// Safari 3.0+
isSafari = /constructor/i.test(window.HTMLElement) || (function (p) { return
p.toString() === "[object SafariRemoteNotification]"; })(!window['safari'] ||
safari.pushNotification);
// Internet Explorer 6-11
isIE = /*@cc_on!@*/false || !!document.documentMode;
// Edge 20+
isEdge = !isIE && !!window.StyleMedia;
// Chrome 1+
isChrome = !!window.chrome && !!window.chrome.webstore;
// Blink engine detection
isBlink = (isChrome || isOpera) && !!window.CSS;

/* Results: */
console.log("isOpera", isOpera);
console.log("isFirefox", isFirefox);
console.log("isSafari", isSafari);
console.log("isIE", isIE);
console.log("isEdge", isEdge);
console.log("isChrome", isChrome);
console.log("isBlink", isBlink);
```

▶ Run code snippet   Expand snippet

The beauty of this approach is that it relies on browser engine properties, so it covers even derivative browsers, such as Yandex or Vivaldi, which are practically compatible with the major browsers whose engines they use. The exception is Opera, which relies on user agent sniffing, but today (i.e. ver. 15 and up) even Opera is itself only a shell for Blink.

Share Edit Follow   edited Feb 8, 2017 at 10:02   answered Jan 17, 2016 at 16:27

pilau
6,673 ● 4 ● 57 ● 69

1 @NoR Oh you're using the VM... The `MSAssertion` trick is adjusted to version 25. But since many devs rely on the VMs, I'll try to adjust it this older version. Good call. Thanks. – pilau Jan 20, 2016 at 17:42

1 @NoR Updated - should be futureproof. The `StyleMedia` (capitalized) object is specific to IE and Edge and doesn't seem to be going anywhere. – pilau Jan 20, 2016 at 20:07

1 I have also found UAParser a js plugin that still maintained and has be highly accurate and easy to use. – Issac Gable Aug 14, 2017 at 16:08

▲
8
▼

If you need to know what is the numeric version of some particular browser you can use the following snippet. Currently it will tell you the version of Chrome/Chromium/Firefox:

```
var match = $window.navigator.userAgent.match(/(?:Chrom(?:e|ium)|Firefox)\/([0-
9]+)\./);
var ver = match ? parseInt(match[1], 10) : 0;
```

Share Edit Follow   edited May 29, 2018 at 18:34   answered Mar 22, 2016 at 5:55

HoldOffHunger
19.8k ● 10 ● 111 ● 139

Valera Tumash
639 ● 7 ● 17

7

There is also a less "hacky" method which works for all popular browsers. Google has included a browser-check in their Closure Library. In particular, have a look at `goog.userAgent` and `goog.userAgent.product`. In this way, you are also up to date if something changes in the way the browsers present themselves (given that you always run the latest version of the closure compiler.)

Share Edit Follow

answered Mar 23, 2017 at 3:56

Albert
195 ● 2 ● 9

7

UAParser is one of the JavaScript Library to identify browser, engine, OS, CPU, and device type/model from userAgent string.

There's an CDN available. Here, I have included a example code to detect browser using UAParser.

```
<!doctype html>
<html>
<head>
<script src="https://cdn.jsdelivr.net/npm/ua-parser-js@0/dist/ua-
parser.min.js"></script>
<script type="text/javascript">
    var parser = new UAParser();
    var result = parser.getResult();
    console.log(result.browser);     // {name: "Chromium", version:
"15.0.874.106"}
</script>
</head>
<body>
</body>
</html>
```

Now you can use the value of `result.browser` to conditionally program your page.

Source Tutorial: How to detect browser, engine, OS, CPU, and device using JavaScript?

Share Edit Follow

edited Aug 5, 2022 at 21:05

answered May 23, 2018 at 6:30

Luzan Baral
3,698 ● 5 ● 39 ● 69

5 a thousand lines of code you call lightweight? – deathangel908 Jun 25, 2018 at 15:44

6

Detecting Browsers on Desktop and Mobile : Edge, Opera, Chrome, Safari, Firefox, IE

> I did some changes in @nimesh code now it is working for Edge also, and Opera issue fixed:

```
function getBrowserName() {
    if ( navigator.userAgent.indexOf("Edge") > -1 &&
navigator.appVersion.indexOf('Edge') > -1 ) {
        return 'Edge';
    }
    else if( navigator.userAgent.indexOf("Opera") != -1 ||
navigator.userAgent.indexOf('OPR') != -1 )
    {
        return 'Opera';
    }
    else if( navigator.userAgent.indexOf("Chrome") != -1 )
    {
        return 'Chrome';
    }
    else if( navigator.userAgent.indexOf("Safari") != -1)
    {
        return 'Safari';
    }
    else if( navigator.userAgent.indexOf("Firefox") != -1 )
    {
        return 'Firefox';
    }
    else if( ( navigator.userAgent.indexOf("MSIE") != -1 ) ||
(!!document.documentMode == true ) ) //IF IE > 10
    {
        return 'IE';
    }
    else
    {
        return 'unknown';
    }
}
```

Thanks @nimesh user:2063564

Share Edit Follow

edited Jan 7, 2019 at 8:22

answered Jan 4, 2019 at 11:50

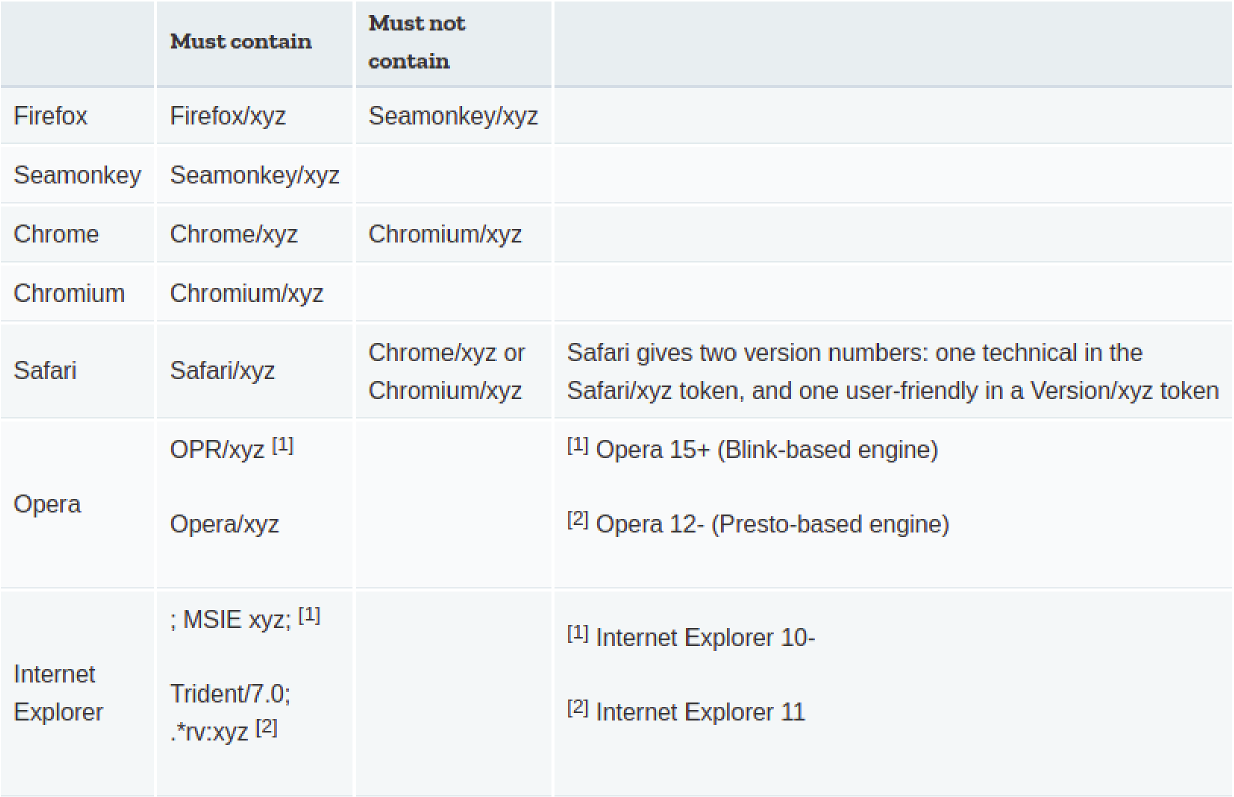
Irshad Khan
5,876 ● 2 ● 47 ● 40

1 It's picking up Safari when I'm in Chrome... – Devin B. Oct 21, 2021 at 20:46

6

Detecting Browser and Its version

This code snippet is based on the article from [MDN](). Where they gave a brief hint about various keywords that can be used to detect the browser name.



	Must contain	Must not contain	
Firefox	Firefox/xyz	Seamonkey/xyz	
Seamonkey	Seamonkey/xyz		
Chrome	Chrome/xyz	Chromium/xyz	
Chromium	Chromium/xyz		
Safari	Safari/xyz	Chrome/xyz or Chromium/xyz	Safari gives two version numbers: one technical in the Safari/xyz token, and one user-friendly in a Version/xyz token
Opera	OPR/xyz [1] Opera/xyz		[1] Opera 15+ (Blink-based engine) [2] Opera 12- (Presto-based engine)
Internet Explorer	; MSIE xyz; [1] Trident/7.0; .*rv:xyz [2]		[1] Internet Explorer 10- [2] Internet Explorer 11

The data shown in the image above is not sufficient for detecting all the browsers e.g. *userAgent of Firefox will have Fxios as a keyword rather than Firefox.*

A few changes are also done to detect browsers like **Edge** and **UCBrowser**

The code is currently tested for the following browsers by changing userAgent with the help of dev-tools (How to change userAgent):

- FireFox
- Chrome
- IE
- Edge
- Opera
- Safari
- UCBrowser

```
getBrowser = () => {
    const userAgent = navigator.userAgent;
    let browser = "unkown";
    // Detect browser name
    browser = (/ucbrowser/i).test(userAgent) ? 'UCBrowser' : browser;
    browser = (/edg/i).test(userAgent) ? 'Edge' : browser;
    browser = (/googlebot/i).test(userAgent) ? 'GoogleBot' : browser;
    browser = (/chromium/i).test(userAgent) ? 'Chromium' : browser;
    browser = (/firefox|fxios/i).test(userAgent) && !
(/seamonkey/i).test(userAgent) ? 'Firefox' : browser;
    browser = (/; msie|trident/i).test(userAgent) && !
(/ucbrowser/i).test(userAgent) ? 'IE' : browser;
    browser = (/chrome|crios/i).test(userAgent) && !
(/opr|opera|chromium|edg|ucbrowser|googlebot/i).test(userAgent) ? 'Chrome' :
browser;;
    browser = (/safari/i).test(userAgent) && !
(/chromium|edg|ucbrowser|chrome|crios|opr|opera|fxios|firefox/i).test(userAgent)
 ? 'Safari' : browser;
    browser = (/opr|opera/i).test(userAgent) ? 'Opera' : browser;

    // detect browser version
    switch (browser) {
        case 'UCBrowser': return
`${browser}/${browserVersion(userAgent,/(ucbrowser)\/([\d\.]+)/i)}`;
        case 'Edge': return
`${browser}/${browserVersion(userAgent,/(edge|edga|edgios|edg)\/([\d\.]+)/i)}`;
        case 'GoogleBot': return
`${browser}/${browserVersion(userAgent,/(googlebot)\/([\d\.]+)/i)}`;
        case 'Chromium': return
`${browser}/${browserVersion(userAgent,/(chromium)\/([\d\.]+)/i)}`;
        case 'Firefox': return
`${browser}/${browserVersion(userAgent,/(firefox|fxios)\/([\d\.]+)/i)}`;
        case 'Chrome': return
`${browser}/${browserVersion(userAgent,/(chrome|crios)\/([\d\.]+)/i)}`;
        case 'Safari': return
`${browser}/${browserVersion(userAgent,/(safari)\/([\d\.]+)/i)}`;
        case 'Opera': return
`${browser}/${browserVersion(userAgent,/(opera|opr)\/([\d\.]+)/i)}`;
        case 'IE': const version =
browserVersion(userAgent,/(trident)\/([\d\.]+)/i);
            // IE version is mapped using trident version
            // IE/8.0 = Trident/4.0, IE/9.0 = Trident/5.0
            return version ? `${browser}/${parseFloat(version) + 4.0}` :
`${browser}/7.0`;
        default: return `unknown/0.0.0.0`;
    }
}

browserVersion = (userAgent,regex) => {
    return userAgent.match(regex) ? userAgent.match(regex)[2] : null;
}

console.log(getBrowser());
```

Run code snippet   Expand snippet

Share Edit Follow

edited Aug 8, 2020 at 21:45

answered Aug 8, 2020 at 21:25

Ankit
612 ●7 ●13

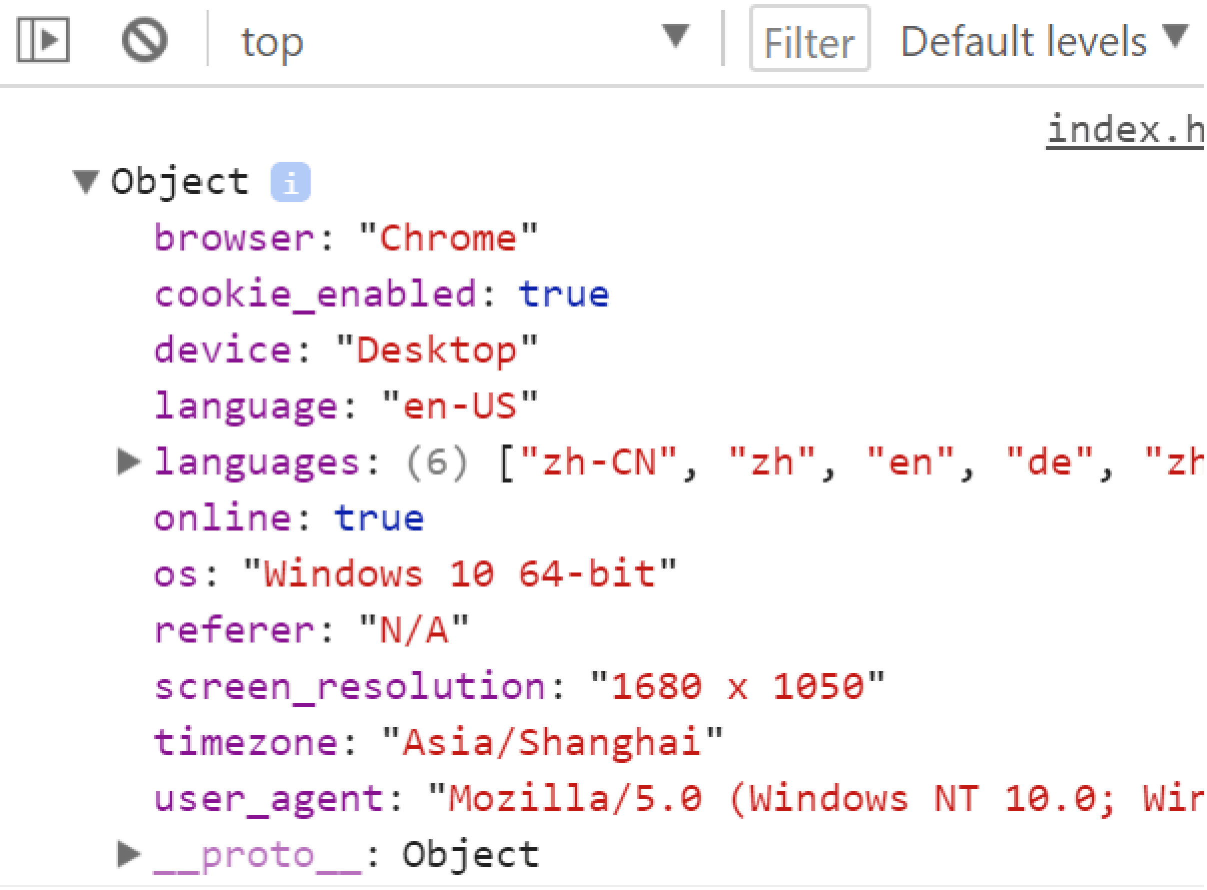
1 Only this one worked for me. Thank you. – Eray Apr 13, 2022 at 18:32

5

You can use Detect-browser.js, JavaScript library that detects and prints an object of browser information including browser language/name, user agent, device type, user OS, referer, online/0ffline, user timezone, screen resolution, and cookie enabled.

Get it from here detect-browser.js

it will give you something like that:



```
▼ Object
    browser: "Chrome"
    cookie_enabled: true
    device: "Desktop"
    language: "en-US"
  ▶ languages: (6) ["zh-CN", "zh", "en", "de", "zh
    online: true
    os: "Windows 10 64-bit"
    referer: "N/A"
    screen_resolution: "1680 x 1050"
    timezone: "Asia/Shanghai"
    user_agent: "Mozilla/5.0 (Windows NT 10.0; Wir
  ▶ __proto__: Object
```

Share Edit Follow

answered Jun 11, 2020 at 9:52

Malki Mohamed

1,648 ● 2 ● 25 ● 41

5

Here is my customized solution.

```
        const inBrowser = typeof window !== 'undefined'
        const UA = inBrowser && window.navigator.userAgent.toLowerCase()
        const isIE =
          UA && /; msie|trident/i.test(UA) && !/ucbrowser/i.test(UA).test(UA)
        const isEdge = UA && /edg/i.test(UA)
        const isAndroid = UA && UA.indexOf('android') > 0
        const isIOS = UA && /iphone|ipad|ipod|ios/i.test(UA)
        const isChrome =
          UA &&
          /chrome|crios/i.test(UA) &&
          !/opr|opera|chromium|edg|ucbrowser|googlebot/i.test(UA)
        const isGoogleBot = UA && /googlebot/i.test(UA)
        const isChromium = UA && /chromium/i.test(UA)
        const isUcBrowser = UA && /ucbrowser/i.test(UA)
        const isSafari =
          UA &&
          /safari/i.test(UA) &&

!/chromium|edg|ucbrowser|chrome|crios|opr|opera|fxios|firefox/i.test(UA)
        const isFirefox = UA && /firefox|fxios/i.test(UA) &&
!/seamonkey/i.test(UA)
        const isOpera = UA && /opr|opera/i.test(UA)
        const isMobile =
          /\b(BlackBerry|webOS|iPhone|IEMobile)\b/i.test(UA) ||
          /\b(Android|Windows Phone|iPad|iPod)\b/i.test(UA)
        const isSamsung = UA && /samsungbrowser/i.test(UA)
        const isIPad = UA && /ipad/.test(UA)
        const isIPhone = UA && /iphone/.test(UA)
        const isIPod = UA && /ipod/.test(UA)

        console.log({
          UA,
          isAndroid,
          isChrome,
          isChromium,
          isEdge,
          isFirefox,
          isGoogleBot,
          isIE,
          isMobile,
          isIOS,
          isIPad,
          isIPhone,
          isIPod,
          isOpera,
          isSafari,
          isSamsung,
          isUcBrowser,
        }
      }
```

Share Edit Follow

edited Dec 4, 2020 at 18:33

answered Dec 1, 2020 at 19:14

Henrique Van Klaveren

1,610 ● 16 ● 24

3

```
var BrowserDetect = {
        init: function () {
            this.browser = this.searchString(this.dataBrowser) || "Other";
            this.version = this.searchVersion(navigator.userAgent) ||
this.searchVersion(navigator.appVersion) || "Unknown";
        },
        searchString: function (data) {
            for (var i = 0; i < data.length; i++) {
                var dataString = data[i].string;
                this.versionSearchString = data[i].subString;

                if (dataString.indexOf(data[i].subString) !== -1) {
                    return data[i].identity;
                }
            }
        },
        searchVersion: function (dataString) {
            var index = dataString.indexOf(this.versionSearchString);
            if (index === -1) {
                return;
            }

            var rv = dataString.indexOf("rv:");
            if (this.versionSearchString === "Trident" && rv !== -1) {
                return parseFloat(dataString.substring(rv + 3));
            } else {
                return parseFloat(dataString.substring(index +
this.versionSearchString.length + 1));
            }
        },

        dataBrowser: [
            {string: navigator.userAgent, subString: "Edge", identity: "MS
Edge"},
            {string: navigator.userAgent, subString: "MSIE", identity:
"Explorer"},
            {string: navigator.userAgent, subString: "Trident", identity:
"Explorer"},
            {string: navigator.userAgent, subString: "Firefox", identity:
"Firefox"},
            {string: navigator.userAgent, subString: "Opera", identity:
"Opera"},
            {string: navigator.userAgent, subString: "OPR", identity: "Opera"},

            {string: navigator.userAgent, subString: "Chrome", identity:
"Chrome"},
            {string: navigator.userAgent, subString: "Safari", identity:
"Safari"}
        ]
    };

    BrowserDetect.init();


    var bv= BrowserDetect.browser;
    if( bv == "Chrome"){
        $("body").addClass("chrome");
    }
    else if(bv == "MS Edge"){
     $("body").addClass("edge");
    }
    else if(bv == "Explorer"){
     $("body").addClass("ie");
    }
    else if(bv == "Firefox"){
     $("body").addClass("Firefox");
    }


$(".relative").click(function(){
$(".oc").toggle('slide', { direction: 'left', mode: 'show' }, 500);
$(".oc1").css({
   'width' : '100%',
   'margin-left' : '0px',
   });
});
```

Share Edit Follow

answered May 4, 2017 at 10:29

neel upadhyay

354 ● 2 ● 10

3

Chrome & Edge introduced a new User-Agent Client Hints API for this:

```
navigator.userAgentData.brands.map(item => item.brand).includes('Google
Chrome')
```

Firefox & Safari don't support it yet unfortunately.

Share Edit Follow

answered Apr 7, 2022 at 8:45

Ollie Williams

2,215 ● 3 ● 26 ● 47

2

To check for IE browser using following code.

```
console.log(/MSIE|Trident/.test(window.navigator.userAgent))
```

OR

```
var isIE = !!document.documentMode;

console.log(isIE)
```

Thanks

Share Edit Follow

edited Jul 7, 2021 at 9:48

answered Jun 23, 2021 at 17:04

Rahul Daksh
266 ●2 ●8

2

This method is currently valid for detecting all browsers. I quoted the vue-element-admin template

```
export function fnBrowserDetect() {
     var browserName = (function(agent) {
       switch (true) {
         case agent.indexOf('edge') > -1: return 'MS Edge'
         case agent.indexOf('edg/') > -1: return 'Edge (chromium based)'
         case agent.indexOf('opr') > -1 && !!window.opr: return 'Opera'
         case agent.indexOf('chrome') > -1 && !!window.chrome: return 'Chrome'
         case agent.indexOf('trident') > -1: return 'MS IE'
         case agent.indexOf('firefox') > -1: return 'Mozilla Firefox'
         case agent.indexOf('safari') > -1: return 'Safari'
         default: return 'other'
       }
     })(window.navigator.userAgent.toLowerCase())
     return browserName.toLowerCase()
   }
```

Share Edit Follow

answered Jan 21, 2023 at 14:16

M.skr
108 ●2 ●9

2 @CarloNyte window.chrome seems to be a Chrome extension and not something defined in the standard. I would be slightly worried if Google did a Microsoft and defined itself as the standard - and got away with it – SimonC Jul 31, 2023 at 8:22

1

```
import getAgent, { getAccurateAgent } from "@egjs/agent";

const agent = getAgent();
getAccurateAgent().then((accurate_agent)=>{
    console.log(agent,'accurate.');
})
console.log(agent);
```

https://github.com/naver/egjs-agent

Share Edit Follow

answered Aug 26, 2021 at 7:37
JaeIL Ryu
169 ●1 ●11

0

Perhaps the real answer is that 90% of the time, you can't reliably detect any specific browser.

The most reliable way for your use case is to provide a download button for each browser and let the user - who knows *for certain* what browser they are on "tell you." *All* of the detection methods shown in these other answers will either eventually stop working, or can easily be faked.

User-agent strings can be overridden by the user.

Browsers, over time, trend toward removing unique API differences. (Source: look at Internet Explorer over the years)

If you *must* use browser detection, use a well-updated library to do so. So long as you keep that library up-to-date, you can be assured that the detection will not fail - so long as it's not too reliant upon UA strings.

For other use cases, don't use browser detection - use feature detection. Modern Firefox supports everything - if not more - that Chrome does, and generally is closer to the w3 & etc. specs. A tool called "browserlist" is a great tool to determine which browsers' features your code should use. Tools like Webpack, Parcel, Babel, and more use this - so you can use the latest browser features without having to write the compatibility code yourself.

Share Edit Follow

edited Feb 15 at 15:44

answered Feb 15 at 15:37
Lazerbeak12345
326 ●5 ●21

-1

This combines both Rob's original answer and Pilau's update for 2016

```
    var isOpera = !!window.opera || navigator.userAgent.indexOf(' OPR/') >= 0;
    // Opera 8.0+ (UA detection to detect Blink/v8-powered Opera)
var isFirefox = typeof InstallTrigger !== 'undefined';   // Firefox 1.0+
var isSafari =
Object.prototype.toString.call(window.HTMLElement).indexOf('Constructor') > 0;
    // At least Safari 3+: "[object HTMLElementConstructor]"
var isChrome = !!window.chrome && !isOpera;              // Chrome 1+
var isIE = /*@cc_on!@*/false || !!document.documentMode;
// Edge 20+
var isEdge = !isIE && !!window.StyleMedia;
// Chrome 1+
var output = 'Detecting browsers by ducktyping:<hr>';
output += 'isFirefox: ' + isFirefox + '<br>';
output += 'isChrome: ' + isChrome + '<br>';
output += 'isSafari: ' + isSafari + '<br>';
output += 'isOpera: ' + isOpera + '<br>';
output += 'isIE: ' + isIE + '<br>';
output += 'isIE Edge: ' + isEdge + '<br>';
document.body.innerHTML = output;
```

Share Edit Follow

answered Jan 21, 2016 at 9:48

Joe Borg

63 ● 2 ● 12

1 @HoldOffHunger it's main intention was more to adapt the most compatible code to the active browser, rather than to inform the user which browser they're using. Unless the browser they're using is super outdated and has been excluded from backwars compatibility, in which it wouldn't hurt to let the user know they can benefit from a better experience should they switch to something more up to date – Joe Borg Nov 2, 2017 at 6:16

-1

Here you find out which browser is it running.

```
function isValidBrowser(navigator){

            var isChrome =  navigator.indexOf('chrome'),
            isFireFox= navigator.indexOf('firefox'),
            isIE = navigator.indexOf('trident') ,
            isValidChromeVer = parseInt(navigator.substring(isChrome+7,
isChrome+8)) >= 4,
            isValidFireForVer = parseInt(navigator.substring(isFireFox+8,
isFireFox+9)) >= 3,
            isValidIEVer = parseInt(navigator.substring(isIE+8, isIE+9)) >= 7;

            if((isChrome > -1 && isValidChromeVer){ console.log("Chrome
Browser")}

            if(isFireFox > -1 && isValidFireForVer){ console.log("FireFox
Browser")}

            if(isIE > -1 && isValidIEVer)){ console.log("IE Browser")}


        }
```

Share Edit Follow

answered Dec 6, 2017 at 4:21

Ajay Kumar

5,070 ● 1 ● 41 ● 44

-1

We can use below util methods

```
utils.isIE = function () {
        var ver = navigator.userAgent;
        return ver.indexOf("MSIE") !== -1 || ver.indexOf("Trident") !== -1; //
need to check for Trident for IE11
    };

    utils.isIE32 = function () {
        return (utils.isIE() && navigator.appVersion.indexOf('Win64') === -1);
    };

    utils.isChrome = function () {
        return (window.chrome);
    };

    utils.isFF64 = function () {
        var agent = navigator.userAgent;
        return (agent.indexOf('Win64') >= 0 && agent.indexOf('Firefox') >= 0);
    };

    utils.isFirefox = function () {
        return (navigator.userAgent.toLowerCase().indexOf('firefox') > -1);
    };
```

Share Edit Follow

answered Feb 20, 2019 at 5:28
NPE
459 ● 2 ● 16

-1

```
const isChrome = /Chrome/.test(navigator.userAgent)
const isFirefox = /Firefox/.test(navigator.userAgent)
```

Share Edit Follow

answered May 8, 2019 at 11:50
Landaida
132 ● 2 ● 8

2 Unfortunately, it's not a good answer. Edge has Chrome in his userAgent message. It's better to use !!window.chrome && (!!window.chrome.webstore || !!window.chrome.runtime); – Przemo Apr 18, 2020 at 13:49

-1

Simple:

```
var OSName="Unknown OS";
if (navigator.appVersion.indexOf("Win")!=-1) OSName="Windows";
if (navigator.appVersion.indexOf("Mac")!=-1) OSName="MacOS";
if (navigator.appVersion.indexOf("X11")!=-1) OSName="UNIX";
if (navigator.appVersion.indexOf("Linux")!=-1) OSName="Linux";
if (navigator.appVersion.indexOf("Linux x86_64")!=-1) OSName="Ubuntu";

var nVer = navigator.appVersion;
var nAgt = navigator.userAgent;
var browserName  = navigator.appName;
var fullVersion  = ''+parseFloat(navigator.appVersion);
var majorVersion = parseInt(navigator.appVersion,10);
var nameOffset,verOffset,ix;

// In Opera, the true version is after "Opera" or after "Version"
if ((verOffset=nAgt.indexOf("Opera"))!=-1) {
 browserName = "Opera";
 fullVersion = nAgt.substring(verOffset+6);
 if ((verOffset=nAgt.indexOf("Version"))!=-1)
   fullVersion = nAgt.substring(verOffset+8);
}
// In MSIE, the true version is after "MSIE" in userAgent
else if ((verOffset=nAgt.indexOf("MSIE"))!=-1) {
 browserName = "Microsoft Internet Explorer";
 fullVersion = nAgt.substring(verOffset+5);
}
// In Chrome, the true version is after "Chrome"
else if ((verOffset=nAgt.indexOf("Chrome"))!=-1) {
 browserName = "Chrome";
 fullVersion = nAgt.substring(verOffset+7);
}
// In Safari, the true version is after "Safari" or after "Version"
else if ((verOffset=nAgt.indexOf("Safari"))!=-1) {
 browserName = "Safari";
 fullVersion = nAgt.substring(verOffset+7);
 if ((verOffset=nAgt.indexOf("Version"))!=-1)
   fullVersion = nAgt.substring(verOffset+8);
}
// In Firefox, the true version is after "Firefox"
else if ((verOffset=nAgt.indexOf("Firefox"))!=-1) {
 browserName = "Firefox";
 fullVersion = nAgt.substring(verOffset+8);
}
// In most other browsers, "name/version" is at the end of userAgent
else if ( (nameOffset=nAgt.lastIndexOf(' ')+1) <
          (verOffset=nAgt.lastIndexOf('/')) )
{
 browserName = nAgt.substring(nameOffset,verOffset);
 fullVersion = nAgt.substring(verOffset+1);
 if (browserName.toLowerCase()==browserName.toUpperCase()) {
  browserName = navigator.appName;
 }
}
// trim the fullVersion string at semicolon/space if present
if ((ix=fullVersion.indexOf(";"))!=-1)
   fullVersion=fullVersion.substring(0,ix);
if ((ix=fullVersion.indexOf(" "))!=-1)
   fullVersion=fullVersion.substring(0,ix);

majorVersion = parseInt(''+fullVersion,10);
if (isNaN(majorVersion)) {
 fullVersion  = ''+parseFloat(navigator.appVersion);
 majorVersion = parseInt(navigator.appVersion,10);
}

document.write(''
 +'Hey! i see you\'re using '+browserName+'! <br>'
 +'The full version of it is '+fullVersion+'. <br>'
 +'Your major version is '+majorVersion+', <br>'
 +'And your "navigator.appName" is '+navigator.appName+'. <br>'
 +'Your "navigator.userAgent" is '+navigator.userAgent+'. <br>'
)

document.write('And, your OS is '+OSName+'. <br>');
```

▶ Run code snippet   Expand snippet

Share Edit Follow

answered Jul 11, 2020 at 13:13

Alon

311 ● 3 ● 9

▲

You can detect it like:

-1

```
if (typeof navigator !== 'undefined' && navigator.userAgent &&
navigator.userAgent.toLowerCase().match(/firefox\/(\d+)/)) {
   alert('Firefox');
}
```

Share Edit Follow

answered Sep 10, 2020 at 9:38

CodeWhisperer

1,171 ● 2 ● 19 ● 40

-1

```
var BrowserType;
(function (BrowserType) {
    BrowserType["OPERA"] = "Opera";
    BrowserType["OPERA2"] = "OPR";
    BrowserType["EDGE"] = "Edg";
    BrowserType["CHROME"] = "Chrome";
    BrowserType["SAFARI"] = "Safari";
    BrowserType["FIREFOX"] = "Firefox";
    BrowserType["UNKNOWN"] = "unknown";
})(BrowserType || (BrowserType = {}));
const detectBrowser = () => {
    return Object.values(BrowserType).find((browser) =>
navigator.userAgent.indexOf(browser) != -1);
};
console.log(detectBrowser());
```

▶ Run code snippet   Expand snippet

Share Edit Follow

answered Jan 27, 2023 at 20:14

satyam_naithani

57 ● 1 ● 3

1 2 Next

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.