## Ziel

Die Lernenden kennen die JavaScript-Objekte, mit denen sich Informationen über den Browser ermitteln lassen. Die Lernenden können mit eigenen Worten beschreiben, was eine "Browserweiche" ist und welchen Zweck diese hat.

## Ausgangslage

Es gibt viele verschiedene Browser und davon wiederum viele unterschiedliche Versionen. Die wichtigsten sind sicherlich Internet Explorer, Firefox, Chrome, Safari und Opera. Leider ist es so, dass man sich als Webentwickler nicht darauf verlassen kann, dass ein JavaScript-Programm in allen Browsern einwandfrei läuft. Es gibt immer mal wieder Inkompatibilitäten, die ein Webentwickler berücksichtigen muss. Und leider betrifft das nicht nur JavaScript-Features, sondern auch HTML und CSS. So können wir uns z.B. nicht darauf verlassen, dass ein Browser die neuen HTML5-Tags alle kennt und richtig darstellt.

Deshalb muss ein Webentwickler auf irgendeine Art erkennen können, welchen Browser verwendet und auch abfragen können, was dieser Browser kann oder eben nicht kann. Dies ermöglicht es, auf inkompatible Browser reagieren zu können und z.B. eine benutzerfreundliche Fehlermeldung auszugeben oder den Benutzer auf eine spezielle, für seinen Browser optimierte Version der Webseite umzuleiten zu können.

## Hilfsmittel

Internet:

* Statistiken zur Verwendung der verschiedenen Browser: <http://gs.statcounter.com/>
* Browser Support Tables von <http://caniuse.com/>
* MDN (JavaScript Tutorial & Referenz): <https://developer.mozilla.org/de/docs/Web/JavaScript>
* JavaScript bei w3schools: <http://www.w3schools.com/js/default.asp>
* etc.

## Vorgehen

Arbeiten Sie die nachfolgenden Kapitel der Reihe nach durch. Sie werden viel theoretischen Input erhalten und zwischendurch auch die eine oder andere Aufgabe lösen müssen.

## Aufgabe: Am häufigsten verwendete Browser in der CH identifizieren

Wenn Sie eine Website für zum Beispiel ein Schweizer Unternehmen entwickeln, ist es wichtig, dass Sie wissen, welche Browser die zukünftigen Besucher der Website benützen, damit Sie die Website für diese Browser optimieren und Browserinkompatibilitäten umgehen können.

Finden Sie heraus, welche Browser von den Schweizern aktuell am meisten verwendet werden. Gehen Sie dazu auf die Webseite <http://gs.statcounter.com/> und identifizieren Sie die 4 Browser, die in der Schweiz in den letzten 12 Monaten am häufigsten verwendet worden sind. Sie können sich dabei auf die Desktop-Browser beschränken und die mobilen Browser erst mal ignorieren.

Füllen Sie die folgende Tabelle aus.

**Top 4 Desktop Browser in der Schweiz**im Zeitraum von \_\_\_\_\_\_\_\_\_\_\_ bis \_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| **Rang** | **Browser** | **Anteil in %** |
| 1. |  |  |
| 2. |  |  |
| 3. |  |  |
| 4. |  |  |

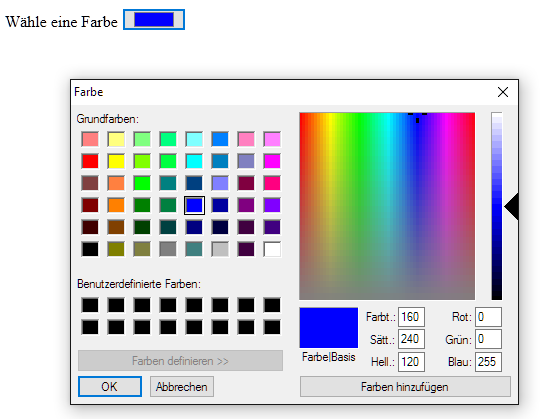
## Aufgabe: Browser Support Tabellen auf *caniuse.com* nachschlagen

Wie können wir herausfinden, ob ein bestimmter Browser ein Sprachelement von HTML, CSS oder JavaScript unterstützt? Eine Möglichkeit ist es, im Internet auf bestimmten Webseiten nachzuschauen. Sie lernen nun eine solche Webseiten kennen. Lösen Sie dazu die folgende Aufgabe.

Seit HTML5 gibt es den *input type* "color", mit dem ein Farbauswahl-Dialog aufgerufen werden kann. Hier der HTML-Code:

<label for="color-picker">Wähle eine Farbe</label>  
<input id="color-picker" type="color"/>

Und hier, was dieser HTML-Code bewirkt, sofern der Browser ihn unterstützt:



Leider unterstützen nicht alle Browser den *input type* "color". Ihre Aufgabe ist es, herauszufinden, ob die folgenden Browser dieses HTML Element unterstützen (OK) oder nicht unterstützen (NOK). Nutzen Sie dazu die Webseite <http://caniuse.com/>.

|  |  |  |
| --- | --- | --- |
| **Browser** | **OK** | **NOK** |
| Internet Explorer 11 (IE11) | □ | □ |
| Firefox Version 47 | □ | □ |
| Chrome 50 | □ | □ |
| Safari 9.1 | □ | □ |
| Safari 10 | □ | □ |

## Aufgabe: Browserinformationen mit JavaScript auslesen

Wie kann man Browsertyp und Browserversion des Benutzers erkennen, um auf allfällige Inkompatibilitäten des Browsers reagieren zu können? Für die Erkennung der Browserversion gibt es in JavaScript ein eigenes Objekt:

* navigator

Auf dieses Objekt und seine Eigenschaften und Methoden kann direkt zugegriffen werden. Es gibt drei interessante Eigenschaften, die wir uns genauer ansehen werden:

* navigator.**appName**
* navigator.**userAgent**
* navigator.**appVersion**

Um zu sehen, was es mit diesen drei Eigenschaften auf sich hat, erstellen Sie nun bitte eine neue HTML-Datei und fügen Sie innerhalb des <body> folgendes Script ein.

<script>  
 document.write("<p>appName --> " + navigator.appName + "</p>");  
 document.write("<p>userAgent --> " + navigator.userAgent + "</p>");  
 document.write("<p>appVersion --> " + navigator.appVersion + "</p>");  
</script>

Öffnen Sie die Datei dann mit mindestens den Browsern Chrome, Firefox, Microsoft Internet Explorer (IE) und Microsoft Edge und tragen Sie in die untenstehende Tabelle die Werte ein, die das Script bei jedem der Browser ausgibt.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Eigenschaft | Chrome | Firefox | Internet Explorer | Edge |
| navigator.appName |  |  |  |  |
| navigator.userAgent |  |  |  |  |
| navigator.appVersion |  |  |  |  |

## Theorie: Browser erkennen

Sie werden feststellen, dass die navigator.appName Eigenschaft sowohl bei Chrome, als auch bei Firefox und sogar teilweise auch beim IE 11 den Wert "Netscape" ausgibt. Die appName Eigenschaft ist also nicht verlässlich genug, um einen Browser erkennen zu können. Daher greifen wir auf navigator.userAgent zurück. Diese Eigenschaft hat den Aufbau *Mozilla/Versionsnummer* gefolgt von plattformspezifischen Informationen.

**Wann ist es Firefox**?

Alle Mozilla-Browser basieren auf der Gecko-Engine und haben im navigator.userAgent eine Angabe der Machart *Gecko/Versionsnummer*. Ausserdem werden Sie feststellen, dass sich der Firefox mit der Angabe *Firefox/Versionsn*ummer meldet.

Damit kann mit JavaScript wie folgt geprüft werden, ob es sich um den Firefox handelt:

var ua = navigator.userAgent.toLowerCase();  
 var isFirefox = (ua.indexOf("gecko/") > -1 && ua.indexOf("firefox/") > -1);  
  
**Wann ist es Internet Explorer?**

Der Internet Explorer ist relativ einfach zu erkennen, zumindest wenn seine Version kleiner als 11 ist. Dann schreibt der IE seine Versionsnummer nämlich immer in die *navigator.userAgent* Eigenschaft in der Form: *MSIE Versionsnummer* (als der Text "MSIE" gefolgt von einem Leerschlag und dann der Versionsnummer). Es kann also einfach auf das Vorhandensein der Zeichenkette "MSIE" geprüft werden bzw. nach der Umwandlung in Kleinbuchstaben auf die Existenz der Zeichenkette "msie".

var ua = navigator.userAgent.toLowerCase();  
 var isIE = ua.indexOf("msie") > -1;

Beim Internet Explorer 11 können wir uns leider nicht darauf verlassen, dass die Zeichenkette "MSIE 11" vorhanden ist. Der IE11 meldet sich manchmal auch mit folgendem User Agent: *Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; AS; rv:11.0) like Gecko*. Hier müsste man sich dann auf die Zeichenkette "Trident/" verlassen (die Layout-Engine des Internet Explorers) plus die Zeichenkette "rv:11.0" (die Browser Version).

**Wann ist es Edge?**

Microsofts Edge-Browser ist erkennbar an der Zeichenkette "Edge/".

Folgender JavaScript-Code identifiziert den Edge-Browser:

var ua = navigator.userAgent.toLowerCase();  
 var isEdge = ua.indexOf("edge/") > -1;

**Wann ist es Chrome?**

Googles Chrome-Browser scheint man auf den ersten Blick einfach zu erkennen, da in der navigator.userAgent Eigenschaft Text in der Form: *Chrome/Versionsnummer* zu finden ist. Doch aufgepasst! Schauen Sie sich einmal an, welchen Wert Microsofts Edge in die userAgent Eigenschaft schreibt. Sie werden feststellen, dass beim Edge-Browser nebst einem Text in der Form *Edge/Versionsnummer* ebenfalls die Zeichenkette "Chrome/" zu finden ist. Das bedeutet, dass es sich nur dann um Chrome handeln kann, wenn zwar die Zeichenkette "Chrome/" zu finden ist, nicht aber die Zeichenkette "Edge/".

Die Erstellung eines entsprechenden Skripts sei ihnen überlassen ;).

## Aufgabe: Weiterleitung mit dem location Objekt

Um mit JavaScript einen Client-Request auf eine neue Seite umleiten zu können, benötigen Sie das location Objekt, welches alle Informationen und Angaben zur URL der aktuellen Seite enthält. Das location Objekt ist ein Unterobjekt des window Objektes, welches Sie ja schon ein wenig kennen, bspw. von window.alert(). Das *window* Objekt ist das Standardobjekt unter JavaScript, sein Name kann also weggelassen werden. Ein einfaches location (bzw. alert()) reicht aus für den Aufruf.

Das location Objekt hat mehrere Eigenschaften und Methoden. Für diese Aufgabe ist die Eigenschaft href interessant, welche die gesamte URL enthält. Mit folgender Anweisung rufen Sie die Website vom Tagesanzeiger auf:

location.href = "http://www.tages-anzeiger.ch/";

Programmieren Sie nun eine sogenannte "Browserweiche", d.h. eine HTML-Datei, die ein JavaScript beinhaltet, welches die verschiedenen Browser auf verschiedene Seiten umleitet.

* Der Firefox soll auf die Seite *firefox.htm*l umgeleitet werden.
* Der Chrome soll auf die Seite *chrome.html* umgeleitet werden.
* Der IE soll auf die Seite *ie.html* umgeleitet werden.
* Der Edge soll auf die Seite *edge.html* umgeleitet werden.
* Ein unbekannter Browser soll als unbekannt gemeldet werden.

Nutzen Sie die Informationen aus dem vorherigen Kapitel, um diese Aufgabe zu lösen.

## Theorie: Browserversion erkennen

Mit der Erkennung des Browsers ist es noch nicht getan. Denn jeder Browser-Hersteller gibt von Zeit zu Zeit neue Versionen seines Browsers heraus. Und leider setzen nicht alle Benutzer immer die neuste Version eines Browsers ein, sodass immer mehrere Versionen eines Browsers in Gebrauch sind.

Da unterschiedliche Versionen eines Browsers auch unterschiedlich kompatibel sind, kommt der ambitionierte Webprogrammierer nicht darum herum, auch auf die Version eines Browsers zu reagieren.

Aus dem Wert der Eigenschaft navigator.appVersion kann die Versionsnummer herausgeschnitten werden. Die Versionsnummer endet vor dem ersten Leerzeichen:

var av = navigator.AppVersion ;   
 var version = av.substring(0, av.indexOf(" "));  
 var version2 = av.substring(0, av.indexOf("."));

In der Variablen version wird die komplette Version gespeichert und in der Variablen version2 lediglich die Hautpversion, d.h. die Versionsnummer vor dem Punkt.

**Besonderheiten beim Firefox**

Firefox meldet sich jeweils mit der Versionsnummer 5.0. Ältere Firefox-Versionen nicht; denn bei älteren Versionen handelt es sich noch um Netscape, den Vorgänger-Browser von Firefox, der heute keine Rolle mehr spielt. Das heisst auch, um die detaillierte/genauere Versionsnummer des Firefox zu erhalten, müssen Sie die Eigenschaft navigator.userAgent bemühen.

var ua = navigator.userAgent.toLowerCase();  
 var ffoxString = ua.substring(ua.indexOf("firefox/"));  
 var ffoxVersion = ffoxString.substring(ffoxString.indexOf("/") + 1) \* 1;   
 // Hinweis zu obiger Codezeile: \*1 wandelt den *string* in eine *number* um

**Besonderheiten beim IE**

Die Versionsbestimmung beim IE gestaltet sich etwas mühsam. Nachfolgend kurz die Regeln, woran man IE 4 und höher erkennt:

* IE4: version2 muss 4 sein.
* IE5: version2 ist 4, und navigator.userAgent enthält die Zeichenkette "MSIE 5"
* IE6: version2 ist 4, und navigator.userAgent enthält die Zeichenkette "MSIE 6"
* IE7: version2 ist 4, und navigator.userAgent enthält die Zeichenkette "MSIE 7"
* IE8: version2 ist 4 oder 5, und navigator.userAgent enthält die Zeichenkette "MSIE 8"
* IE9: version2 ist 5, und navigator.userAgent enthält die Zeichenkette "MSIE 9"
* IE10: version2 ist 5, und navigator.userAgent enthält die Zeichenkette "MSIE 10"
* IE11: version2 ist 5, und navigator.userAgent enthält die Zeichenkette "MSIE 11" oder die Zeichenkette "rv:11.0"

|  |
| --- |
| **Gut zu wissen:** Sie können sich die User Agents der verschiedenen Internet Explorer Versionen emulieren lassen. Öffnen Sie dazu den Edge-Browser und lasse Sie sich mit F12 die Entwicklerwerkzeuge anzeigen. Dort aktivieren Sie den Reiter "Emulation". Sodann können Sie in der Dropdownliste "Benuter-Agents" verschieden IE Versionen emulieren. |

## Aufgabe: Browserweiche für Safari 7 programmieren

Schreiben Sie den JavaScript-Code, mit dem Apples Browser **Safari, Version** **7** erkannt werden kann.

Sie haben den Safari-Browser auf Ihrem Rechner nicht installiert und haben darum keinen Plan, wie die Eigenschaft navigator.userAgent bei Safari aussieht? Sie können die Informationen, die Sie benötigen, auf der folgenden Webseite finden:

<http://www.useragentstring.com/pages/useragentstring.php>

**Hinweis:** Berücksichtigen Sie beim Programmieren Ihrer Browserweiche, dass auch der Chrome sich mit der Zeichenkette "Safari/" in der userAgent Eigenschaft meldet!

Ihre Lösung:

-----------------------------------------------------------------------------------------------

-----------------------------------------------------------------------------------------------

-----------------------------------------------------------------------------------------------

-----------------------------------------------------------------------------------------------

## Theorie: Objektabfragen mit JavaScript (JavaScript-Feature-Erkennung)

Von modernen, professionellen JavaScript-Programmen wird erwartet, dass sie browser-übergreifend funktionieren. Da nicht jeder Browser JavaScript gleich gut respektive im vollen Umfang unterstützt, ist es nötig, zu prüfen, ob ein bestimmter JavaScript-Befehl im betreffenden Browser funktioniert oder nicht. Wie Sie gesehen haben, können dazu sogenannte "Browserweichen" eingesetzt werden, die mithilfe von des navigator-Objektes versuchen, den Browser zu identifizieren.

**Der Weg über das navigator –Objekt ist in der heutigen Weblandschaft, in der die Browser in ihren verschiedenen Versionen unzählige Techniken unterschiedlich umsetzen, nicht mehr gangbar**, zumindest, was die Unterstützung von JavaScript betrifft. Der Browsermarkt und die Entwicklung neuer, in JavaScript nutzbarer Techniken ist zu stark in Bewegung, als dass eine solche "Browserweichen" zuverlässig funktionieren könnte.

An die Stelle von "Browserweichen" sind heute "**Fähigkeitenweichen"** getreten. Diese ermitteln nicht die Browserversion, sondern sie prüfen konkret die Existenz derjenigen JavaScript-Objekte, die im Laufe des Scripts verwendet werden sollen. Entscheidend ist dann, ob die nötigen Objekte existieren und korrekt funktionieren, also etwa, dass Eigenschaften einen erwarteten Wert haben oder Methoden das spezifizierte Ergebnis liefern. Dieses Abfragen von Objekten, Methoden und Eigenschaften spielt heutzutage eine zentrale Rolle in JavaScript. Objektabfragen fallen im Einzelfall allerdings sehr unterschiedlich aus und es ist nicht immer ohne Weiteres klar, wie eine Abfrage programmiert werden muss, damit sie effizient und zuverlässig das tut, was man von ihr erwartet.

An dieser Stelle wird darauf verzichtet, die Funktionsweise und die geeigneten Anwendungen der verschiedenen Objektabfrage-Techniken zu erklären. Sie werden sich in Ihrer Karriere als JavaScript-Programmierer später damit auseinandersetzen. **Bis auf weiteres reicht es vollkommen, wenn Sie wissen, dass mit JavaScript Objektabfragen gemacht werden können, um die JavaScript-Kompatibilität von Browsern zu testen.**

Für den neugierigen und/oder ambitionierten JavaScript-Einsteiger sei hier dennoch an einen Artikel im Internet verwiesen, der gut in die "Geheimnisse" der Objektabfragen mit JavaScript einführt:

 <https://wiki.selfhtml.org/wiki/JavaScript/Objektabfragen>