

Maier, Thomas; www.css4.at
JavaScript. Teacher edition

Graz : *css4.at* 2022, 197 S.



Quellenangabe/ Reference:

Maier, Thomas; www.css4.at: JavaScript. Teacher edition. Graz : *css4.at* 2022, 197 S. - URN:
urn:nbn:de:0111-pedocs-247759 - DOI: 10.25656/01:24775

<https://nbn-resolving.org/urn:nbn:de:0111-pedocs-247759>

<https://doi.org/10.25656/01:24775>

Nutzungsbedingungen

Dieses Dokument steht unter folgender Creative Commons-Lizenz:
<http://creativecommons.org/licenses/by-nc-sa/4.0/deed.de> - Sie dürfen das Werk bzw. den Inhalt unter folgenden Bedingungen vervielfältigen, verbreiten und öffentlich zugänglich machen sowie Abwandlungen und Bearbeitungen des Werkes bzw. Inhaltes anfertigen: Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen. Dieses Werk bzw. der Inhalt darf nicht für kommerzielle Zwecke verwendet werden. Die neu entstandenen Werke bzw. Inhalte dürfen nur unter Verwendung von Lizenzbedingungen weitergegeben werden, die mit denen dieses Lizenzvertrages identisch oder vergleichbar sind. Mit der Verwendung dieses Dokuments erkennen Sie die Nutzungsbedingungen an.

Terms of use

This document is published under following Creative Commons-Licence:

<http://creativecommons.org/licenses/by-nc-sa/4.0/deed.en> - You may copy, distribute and transmit, adapt or exhibit the work in the public and alter, transform or change this work as long as you attribute the work in the manner specified by the author or licensor. You are not allowed to make commercial use of the work. If you alter, transform, or change this work in any way, you may distribute the resulting work only under this or a comparable license.

By using this particular document, you accept the above-stated conditions of use.



Kontakt / Contact:

peDOCS
DIPF | Leibniz-Institut für Bildungsforschung und Bildungsinformation
Informationszentrum (IZ) Bildung
E-Mail: pedocs@dipf.de
Internet: www.pedocs.de

Mitglied der


Leibniz-Gemeinschaft

```
// Teacher Edition  
JavaScript {  
    verfassung = "Thomas Maier";  
    inhalt[0] = "Unterrichtsplanung";  
    inhalt[1] = "Lernhandouts";  
    inhalt[2] = "Übungshandouts";  
    inhalt[3] = "Zielformulierungen";  
    inhalt[4] = "Didaktische Gedanken";  
}
```

2022



JavaScript von Thomas Maier

Zeppelinstraße 12A/7, 8055 Graz
www.css4.at | javascript@css4.at
Copyright: CC Lizenz BY NC SA 2022, Version 2

Diese Arbeit wurde mit LibreOffice auf einem Linux Betriebssystem erstellt und entspricht damit einem Grundsatz der Creative Commons. Alle Inhalte sind als Open Educational Resources gekennzeichnet. Bitte prüfen Sie auf <https://oer.css4.at> ob eine neuere Version dieses Scriptes vorhanden ist!

Inhaltsverzeichnis

Allgemeine didaktische Konzeption.....	6
Unterrichtsgrundsätze.....	6
Allgemeine didaktische Grundsätze.....	6
Unterrichtsprinzipien.....	7
Eingangsvoraussetzungen.....	7
Vorkenntnisse in HTML und CSS.....	8
Flankierende Methoden.....	9
Raumausstattung.....	9
EDV Infrastruktur.....	10
Blendet Learning.....	12
Sozialformen.....	13
Exemplarisches Stundenbild.....	14
Intensionen der Studierenden.....	15
Kompetenzbegriff und Zielsetzungen.....	15
Grobziele und Kompetenzzuwachs.....	16
Handouts und Assignments.....	17
Lernhandout 1.0 Einführung.....	18
Lernhandout 2.0 Ausgabe im Browser.....	20
Lernhandout 3.0 Variablen.....	22
Lernhandout 4.0 Rechenoperatoren.....	24
Übungsblatt 4.0 Operatoren.....	26
Lernhandout 5.0 Funktionen.....	28
Lernhandout 6.0 Eingabe.....	30
Übungsblatt 6.0 Eingabe.....	32
Lernhandout 6.1 Ereignis Attribute.....	34
Lernhandout 7.0 Verzweigung.....	36
Übungsblatt 7.0 Verzweigung.....	38
Lernhandout 7.1 switch case.....	40
Übungsblatt 7.1 switch case.....	42
Lernhandout 8.0 Schleifen.....	44
Lernhandout 8.1 for Schleifen.....	46
Übungsblatt 8.1 Schleifen.....	48
Lernhandout 9.0 Arrays.....	50
Übungsblatt 9.0 Arrays.....	52
Lernhandout 9.1 Array Methoden.....	54
Lernhandout 9.2 Arrays sortieren.....	56
Übungsblatt 9.2 Arrays sortieren.....	58
Lernhandout 10.0 Text Operationen.....	60
Übungsblatt 10.0 Text Operationen.....	62
Lernhandout 10.1 String Methoden.....	64
Übungsblatt 10.1 String Methoden.....	66

Lernhandout 10.2 Datum.....	68
Lernhandout 11.0 .style.....	70
Lernhandout 11.1 .style Objekt.....	72
Übungsblatt 11.1 .style Objekt.....	74
Lernhandout 11.2 Selektoren.....	76
Lernhandout 11.3 .setAttribute.....	78
Übungsblatt 11.3 .setAttribute.....	80
Lernhandout 12.0 Nodes im DOM.....	82
Übungsblatt 12.0 Nodes im DOM.....	84
Lernhandout 12.1 Nodes einfügen.....	86
Lernhandout 12.2 Nodes ersetzen.....	88
Lernhandout 12.3 Nodes entfernen.....	90
Lernhandout 12.4 Nodelist.....	92
Übungsblatt 12.4 Nodelist.....	94
Lernhandout 13.1 document.....	96
Lernhandout 13.2 Timer.....	98
Übungsblatt 13.2 Timer.....	100
Lernhandout 14.0 Window Objekt.....	102
Übungsblatt 14.0 Window Objekt.....	104
Lernhandout 14.1 Pop-Up Window.....	106
Übungsblatt 14.1 Pop-Up Window.....	108
Lernhandout 14.2 Window Methoden.....	110
Lernhandout 14.3 pageOffset.....	112
Übungsblatt 14.3 pageOffset.....	114
Lernhandout 15.1 Navigator.....	116
Lernhandout 15.2 Location.....	118
Übungsblatt 15.2 Location.....	120
Lernhandout 15.3 Wertübergabe GET.....	122
Übungsblatt 15.3 Wertübergabe GET.....	124
Lernhandout 15.4 localStorage.....	126
Übungsblatt 15.4 localStorage.....	128
Lernhandout 15.5 Cookies.....	130
Übungsblatt 15.5 Cookies.....	132
Lernhandout 16.1 EventListener.....	134
Übungsblatt 16.1 EventListener.....	136
Lernhandout 16.2 Keyboard Events.....	138
Übungsblatt 16.2 Keyboard Events.....	140
Lernhandout 16.3 Mouse Event Typen.....	142
Lernhandout 16.4 Mouse Eigenschaften.....	144
Übungsblatt 16.4 Mouse Eigenschaften.....	146
Lernhandout 16.5 Touch Events.....	148
Lernhandout 17.1 Audio.....	150
Übungsblatt 17.1 Audio.....	152
Lernhandout 17.2 Video.....	154
Lernhandout 17.3 Audio Video Events.....	156
Lernhandout 18.1 Canvas.....	158
Lernhandout 18.2 Canvas Pfade.....	160
Lernhandout 18.3 Canvas Text & Bild.....	162
Übungsblatt 18.3 Canvas.....	164
Lernhandout 19.1 Geolocation.....	166
Übungsblatt 19.1 Geolocation.....	168

Lernhandout 20.1 OOP Einführung.....	170
Lernhandout 20.2 OOP Literale.....	172
Lernhandout 20.3 OOP Konstruktor.....	174
Übungsblatt 20.3 Konstruktor.....	176
Lernhandout 20.4 JSON.....	178
Lernhandout 21.1 Projekte.....	180
Übungsblatt 21.1 Projekte.....	182
Übungsblatt 21.2 Projekte Wirtschaft.....	184
Übungsblatt 21.3 Projekte Entertainment.....	186

Pädagogischer Beipackzettel.....188

Gedanken zur Leistungsbewertung.....	188
Warum Blendet Learning?.....	189
Sozialformen in der IKT.....	189
Der konsequente Lehrplanbezug.....	189
Die kritische Sozialformierung.....	190
Jenseits von Gut und Sehr Gut.....	190
Mit Imperativen [programmieren] lernen.....	193

Quellenverzeichnis.....195

Internetquellen und online Tools.....	195
Literaturverzeichnis.....	195

Allgemeine didaktische Konzeption

Unterrichtsgrundsätze

Im Unterricht soll ein Klima der Wertschätzung sowie des Respekts und der Gleichbehandlung vorherrschen.

Das Klassenzimmer soll ein Ort der offenen Kommunikation sein. Der Unterricht ein Forum für Fragen und Diskussionen. Fragen aller Art sind erwünscht. Getreu: "Es gibt keine dummen Fragen – die einzige dumme Frage ist jene, die nicht gestellt wird".

Im Unterricht sollte Zeit für individuelles Fördern gefunden werden. Leistungsschwächere Studierende sollen gefördert werden. Leistungsstärkere Studierende gefordert. Dabei sollte dem "Schüler-Schüler-Coaching" Vorrang gegeben werden.

Der Fokus liegt auf den Leistungen der Studierenden. Die besondere Würdigung von Qualitäten soll einer fehlerzentrierten Wahrnehmung Vorzug gegeben werden. Fehler machen ist erlaubt – jedoch sollen diese kommuniziert werden.

Die Leistungen der Studierenden sind sowohl auf der Prozessebene als auch auf der Produktebene zu würdigen. Es wird nicht nur das aktive Lernen gefordert, sondern auch das passive Lernen akzeptiert.

Allgemeine didaktische Grundsätze

- Unterrichtsbeispiele mit Praxis- und Lebensbezug
- Methodenvielfalt und kooperatives offenes Lernen
- Förderung von Selbstständigkeit, Selbstverantwortung, Einzel- und Teamarbeit, sozialem Lernen.
- Einsatz von Blended Learning (IT-Bezug), Standardsprache und Fachterminologie
- Individuelles Fördern bei einer Maximierung der reinen Lernzeit
- Handlungsorientierte Unterrichtsplanung und Einhaltung der Progressionslogik
- Produkt und Prozessorientierte Leistungsbewertung

Unterrichtsprinzipien

Es gelten die allgemeinen Unterrichtsprinzipien laut dem Bundesministerium für Bildung. Die fettgedruckten Prinzipien wurde in den Detailplanungen Rechnung getragen. Andere sollten unbedingt im Rahmen des pädagogischen/situationsgemäßen Gesprächs vermittelt werden. Vergessen Sie niemals diese äußerst wichtigen Prinzipien – sie sind weit mehr als einfaches fächerübergreifendes Lernen/Lehren.

- Erziehung zur Gleichstellung von Frauen und Männern
- Gesundheitserziehung
- **Interkulturelles Lernen**
- **Leseerziehung**
- Medienbildung
- **Politische Bildung**
- Sexualerziehung
- **Umweltbildung**
- Verkehrserziehung
- **Wirtschaftserziehung und Verbraucher_innenbildung**

Eingangsvoraussetzungen

Das Erlernen von JavaScript kann nicht sofort ohne weiteres starten. Es sind Kompetenzen wie das Englische auf Niveau A2 sowie das Schreiben mit 10-Fingern klar von Vorteil. Viel wichtiger ist, grundsätzliche Erkenntnisse der imperativen Programmierung und sämtliche Kompetenzen der Algebra sowie einem (wohl am wichtigsten) Ordnungsbewusstsein als Eingangskompetenzen anzuführen. Es sollte auch ein Grundverständnis für Logik vorhanden ein. Da nun also, JavaScript als eine Einführung der konzeptionellen Erkenntnisse der Softwareentwicklung darstellt, ist es erst nach der Kenntnisvermittlung einer allgemeinen Programmierung als diskussionswürdig darzustellen.

Exemplarisch: Startet man gerne mit der Darstellung und den Ablauf des Kaffeekochens. Dieses wäre ein einfacher Algorithmus, den die Studierenden dekonstruieren. Ebenso muss das EVA Prinzip verstanden werden.

Wissen und das kognitive Verständnis von mathematischer Methodik ist gänzlich unverzichtbar für das Erlernen einer Programmiersprache. So ist die Zahlenlehre notwendig um das Prinzip einer Variablendeklaration zu verstehen – oder um mit den Rechengesetzten eine Variablen-Wert-Zuweisung auszuführen. Es wirkt wechselseitig und verlangt zugleich ein kritisches Bewusstsein.

Kenntnisse der formalen Aussagenlogik (ob nun aus dem Philosophieunterricht oder aus dem Mathematikunterricht) sind selbstverständlich kein Nachteil. Die informatische Logik basiert auf der klassischen Aussagenlogik. Es wird ein Verständnis für logische Operatoren vermittelt (Und, Oder, Wenn-Dann usw.) sowie die Gültigkeitsprüfung von Argumenten über Wahr oder Falsch, bzw. in der Informatik: True or False!

Das Vorkenntnisse in der allgemeinen Informatik als Grundvoraussetzung betrachtet werden soll, liegt doch klar auf der Hand. Ebenso sollte der_ die Studierende ein Office Paket beherrschen. Einblicke in die Datenbankmodellierung (z. B. über Access) sind von Vorteil. Der_ die Studierende muss das eigene Betriebssystem kennen.

Vorkenntnisse in HTML und CSS

Die Studierenden müssen bereits Kenntnisse in HTML und CSS mitbringen. Auf der Lernplattform www.css4.at befindet sich ein Tutorial für HTML und CSS. Als Grundlage um mit JavaScript zu arbeiten sollten folgende Kapitel beherrscht werden:

Basiskenntnisse (Grundlagen) für die Ausgabe	Unbedingt notwendige Erweiterungen für die Eingabe
<ul style="list-style-type: none"> 1.1 – HTML Grundgerüst 2.1 – Basis Tags 2.2 – Basis a href, img 3.1 – Tabellen 4.1 – style, span, color 4.2 – Rahmen 4.3 – Höhe und Breite 6.1 – Ausrichtung 6.2 – Abstände 8.1 – Positionierung 8.2 – Display 	<ul style="list-style-type: none"> 12.1 – Formulare 12.2 – Textarea 12.3 – Input 12.4 – Input Typen 12.5 – Formular Buttons 12.6 – Dialogeingaben 12.7 – Auswahleingaben 12.8 – Auswahllisten
<p>Aufbauend auf diesen Kenntnissen, ist der selbstständige Erwerb von Vertiefungen problemlos möglich.</p>	<p>Zusätzlich zu den Basiskenntnissen sind die Formularelemente notwendig um eine Eingabe-Ausgabe-Maske zu erstellen.</p>

Flankierende Methoden

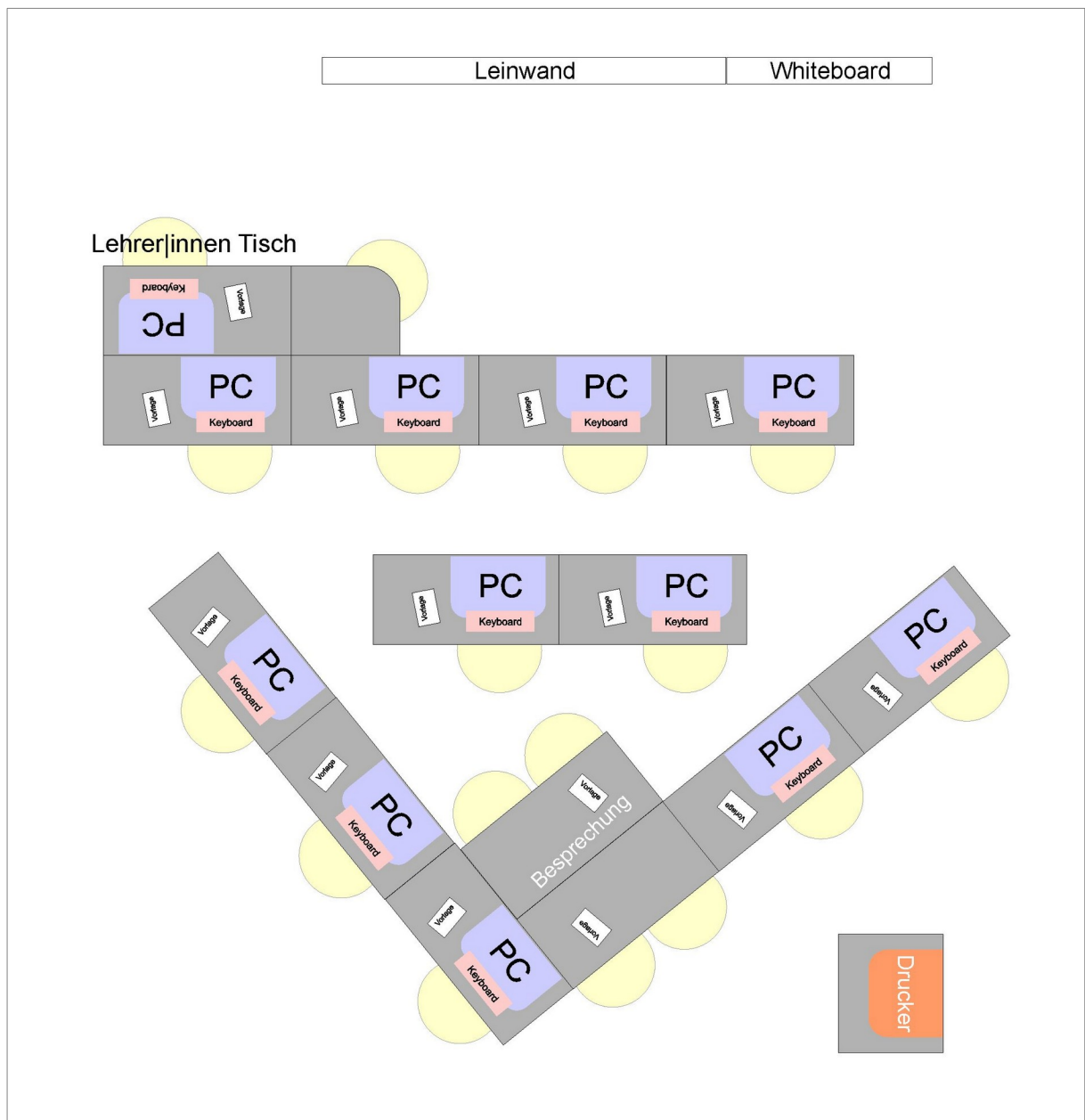
Neben den klassischen Methoden Frontalunterricht, Erklären, Lehrervortrag und Vorzeigen-Nachmachen, sind folgende Unterrichtsmethoden ebenfalls denkbar:

- Aufgabenorientiertes Lernen
- Fächerübergreifendes und –verbindendes Lernen
- Handlungsorientiertes Lernen
- Einzelarbeit
- Konstruktionsaufgaben
- Projektlernen
- projektorientierter Unterricht
- Freie Arbeit
- Entdeckendes Lernen
- Mind-Mapping
- Brainwriting
- uvm.

Raumausstattung

Folgende Medien und Ausstattungen sind für den Unterricht empfehlenswert:

- Großes Whiteboard
- Beamer (Projektor)
- Besprechungstische
- ausreichend Bürostühle
- jeweils einen Bürotisch pro PC (Arbeitsplatz – PC sind rechts angeordnet, damit die Linke Hälfte des Tisches Platz für Vorlagen bietet).
- ein Drucker



Vorschlag für eine Raumgestaltung.

EDV Infrastruktur




JavaScript ist äußerst genügsam was die Leistung der Hard- und Software betrifft. Im Grunde reicht ein gängiger PC mit einem Editor und einem Browser. Das Betriebssystem spielt nur eine untergeordnete Rolle, da JavaScript plattformunabhängig ist. Ob nun Windows, OSX oder Linux – egal! Hier eine Darstellung der benötigten Software.

- Ein schneller Texteditor
- Eine umfangreiche IDE-Software (HTML Editor)
- sämtliche gängige Browser
- Eine Textverarbeitungssoftware
- weitere Tools: Color Picker, ZIP Software, Screenshot-Tools, PDF Viewer (Acrobat odgl.)
- kolaborative Anwendungen: z. B. Office 365, OneNote, Google, TeamViewer

- didaktische Tools: iTalc, Veyon, Moodle
- FTP Software (z. B. FileZilla)
- Virtualisierungssoftware
- CMS Editoren (z. B. Adobe Muse, Joomla, Wordpress)
- Entwicklungsumgebungen (z. B. Visual Studios)
- Bildbearbeitungssoftware
- Audio und Videobearbeitungssoftware**

Eine Webanbindung des Schulservers (z. B. über IIS, Apache odgl.) wäre sicher ein Pluspunkt, damit die Studierenden ihre JavaScripte sofort online stellen können. Die Planung eines Webservers ist aber ein anderes Thema, das den Rahmen dieser Arbeit sprengen würde. Die Studierenden können aber ihre Lernprodukte auf gratis Webspaces anbieten, wie z. B. mur.at oder Lima-City.de publizieren (wenn nötig und wenn sie es wollen).

Beispiele für Basisinstallationen

		
Microsoft Windows	Apple OSX	Linux
<ul style="list-style-type: none"> • XAMPP (WAMPP) • Notepad ++ • Adobe Dreamweaver • ATOM • Edge, Chrome, Firefox • MS Office • MS Access • Color Picker • 7Zip • Snipping Tool • Adobe PDF Acrobat • FileZilla • OneNote • TeamViewer • Veyon (iTalc) • Gimp bzw. Adobe Photoshop • Corel Draw, Adobe Illustrator, Inkscape • Audacity • Oracle VirtualBox • MS SQL Server 	<ul style="list-style-type: none"> • XAMPP • TextWrangler • Adobe Dreamweaver • ATOM • Safari, Chrome, Firefox • Apple iWork • MS Access • Color Picker • 7Zip • Skitch • Adobe PDF Acrobat • FileZilla • Evernote • TeamViewer • Veyon (iTalc) • Gimp bzw. Adobe Photoshop • Corel Draw, Adobe Illustrator, Inkscape • Audacity • Oracle VirtualBox • SQL Server 	<ul style="list-style-type: none"> • XAMPP • Sublime • Bluefish • ATOM • Chrome, Firefox • LibreOffice • Kexi • KColorChooser • 7Zip • Shutter • Adobe Reader • FileZilla • Evernote • TeamViewer • Veyon (iTalc) • Gimp • Inkscape • Audacity • Oracle VirtualBox • SQL Server

Die Hardware ist definitiv genügsamer:

- Bei Laptops muss ein externes Keyboard bereitgestellt werden. Ebenso eine Maus.
- Ein zweiter Monitor ist ein Nice-to-Have, aber nicht zwingend notwendig.
- Eine Breitband-Internet-Verbindung.
- Ein funktionierendes Netzwerk mit Serverseitigen Profilen und Netzlaufwerken.
- Audio-Ausgabe: Entweder über Lautsprecher oder besser über Kopfhörer.
- USB Anschlüsse für die Verwendung von USB-Speicher-Sticks.
- Ein Netzwerk-Drucker
- Mobile Devices (Smartphone und Tablets) um die Scripte zu testen

Blendet Learning



Abbildung: Die Bannernavigation von www.css4.at

Auf der Lernplattform www.css4.at befindet sich links oben ein Suchfeld. Über dieses kann man nach notwendigen Dateien, Suchbegriffen und Refcodes suchen.

Auf der Startseite befindet sich der Lernbereich als dunkelblauer Kasten. Dort findet man unter Web Development einen Link zur Lernplattform: JavaScript!

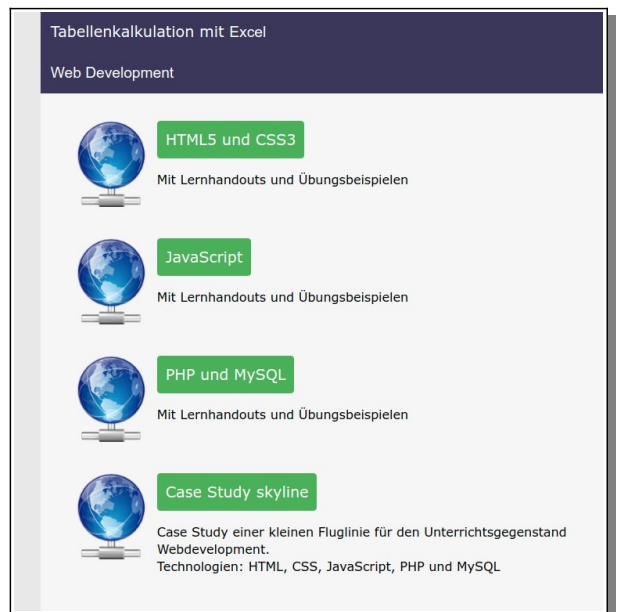


Abbildung: Im Lernbereich

Die Lernplattform für JavaScript erlaubt zu jeder Einheit die Downloads der Lern- und Übungshandouts und der benötigten Dateien für die Übungsbeispiele.

Zusätzlich findet man auf der Lernplattform noch hilfreiche Links zum Unterricht. In der linken Spalte findet man die Lernhandouts und korrespondierend rechts die dazu passenden Übungshandouts.

Zu allen Übungsbeispielen gibt es selbstverständlich eine Lösung, die in den meisten Fällen nur als Referenz dient und nicht als absolute Lösung (ohne Abweichung von Komma und Beistrich). Der Zugang zu den Lösungen erfolgt passwortgeschützt über <https://solved.css4.at/> - dort gibt es auch ein Antragsformular für den Zugriff auf den Lösungsbereich.

Die Verwendung von Lern- und Übungshandouts, sowie der Lösungen ist selbstverständlich für Lehrer_innen von öffentlichen Schulen oder von Organisationen mit öffentlichen Charakter kostenlos und frei verwendbar.

Sollten Sie das Unterrichtsmaterial von www.css4.at kommerziell verwenden (z. B. für bezahlte Kurse in der Erwachsenenbildung) dann bitte ich Sie recht höflich mir zuvor eine Mail an webmail@css4.at mit dem didaktischen Umfang ihres Unterrichts (didaktische Konzeption) und dem damit verbunden finanziellen Volumen zuzusenden. Wir finden sicher einen Modus für die Verwendung des Unterrichtsmaterials.

Sozialformen

Das Unterrichtsmaterial ist auf Einzelarbeit ausgelegt. Dennoch ist Partnerarbeit bei Bildschirmknappheit denkbar. Gruppenarbeit kommt kategorisch nicht vor. Das Plenum wird nur im Rahmen des Frontalunterrichts aktiviert.

Methoden, die andere Sozialformen bedingen, unterliegen der didaktischen Kreativität des_der Lehrerin. Gegen Ende des Unterrichts kann gerne eine Projektarbeit angesetzt werden, die auch in der Gruppe gelöst werden kann.

Exemplarisches Stundenbild

↓	Dauer	Wann? Zeit bis	Was? Lerninhalte	Wie? Methoden	Wodurch? Medien	Warum? Methode	
Einstieg	2 min	2'	Begrüßung	päd. Gespräch	---	---	
	5 min	7'	Vorbesprechung	päd. Gespräch	---	Erhebung von Interessen	
	6 min	13'	Wiederholung	Mind-Mapping	A3 Bogen oder Whiteboard	Wiederholung der letzten UE.	
Erarbeitung	10 min	23'	Einheit x	Vorzeigen – Nachmachen	Bildschirm	Vermittlung der Skills	
	30 min	53'	Übungsbeispiele Einheit x	freies Arbeiten	Bildschirm	Kombiniert mit individuellen Coaching	
	5 min	58'	PAUSE				
	10 min	68'	Einheit y	Vorzeigen Nachmachen	Bildschirm	Vermittlung der Skills	
	30 min	98'	Übungsbeispiele Einheit y	freies Arbeiten	Bildschirm	Kombiniert mit individuellen Coaching	
Erfolgssicherung	15 min	113'	Individuelles Coaching und freies Arbeiten	Coaching	Bildschirm	Leistungsbeobachtung, Weiterführende Fragen beantworten	
	5 min	118'	Ausblick auf die nächste UE.	päd. Gespräch	---	Die nachgelagerten Themen ansprechen	
	2 min	120'	Verabschiedung	päd. Gespräch	---	Verabschiedung und Herstellung der Arbeitsplatz-Ordnung. (z. B. Herunterfahren des PC)	

Intensionen der Studierenden

Was genau in den Köpfen der Studierenden vorgeht können wir nicht wissen. Würden wir alle Intensionen, Gedanken und Affekte der Studierenden kennen, dann würde auch unsere Didaktik gänzlich anders aussehen – aber so wie es ist, ist es auch die große Herausforderung pädagogischer Arbeit. Hier eine Aufstellung möglicher Intensionen:

Die Studierenden wollen ...

- grundlegende und vertiefende Kenntnisse des Webdevelopements erwerben.
- vertiefende Kenntnisse des Webdevelopements erwerben.
- eigene Webseiten und Apps nach eigenen Vorstellungen entwerfen und diese mit JavaScript dynamisch verändern.
- das Internet mitgestalten.
- eine positive Note bzw. eine gezielte Note für das Unterrichtsfach bekommen.
- die Ausbildung abschließen.
- das erworbene Wissen aus dem Webdevelopment fächerübergreifend in anderen Gegenständen einsetzen.
- mit dem Wissen und den Fähigkeiten Geld verdienen.
- soziale Kontakte erleben.

Kompetenzbegriff und Zielsetzungen

Die Konzeption der Lern- und Lehrunterlagen dieser Arbeit unterliegen dem strengen Gebot der Progressionslogik. Alle Themenbereiche sind aufbauend gegliedert und meist von ihrem Vorgängerthema abhängig. Wichtige Elemente des didaktischen Aufbaus zur Erlernung von JavaScript wiederholen sich in den Übungsbeispielen um diese zu festigen.

Die Unterrichtsplanung entspricht nicht gängigen Vorstellungen von Bildungsstandards ist aber sehr wohl kompetenzorientiert aufgebaut. Es gibt keine Deskriptoren die in eine 4.0 Skala zur Leistungsbewertung übertragen werden können. Dafür gibt es klassische Zielformulierungen (Grob- und Feinziele), die um die Taxonomien "Verstehen", "Anwenden", "Analysieren" und "Entwickeln" erweitert wurden. Die Zielarten werden in "Affektiv, Kognitiv, Psychomotorisch, Selbstkompetenz und Inhaltsebene" differenziert.

Grobziele und Kompetenzzuwachs

Der_die Studierende ...	Zielart	Taxonomie
... erlernt die Verwendung von JavaScript.	I	II
... nutzt das Wissen zur Gestaltung von dynamischen Webseiten.	I	II
... erlernt neue Syntax und Anwendung von Objekten, Methoden, Eigenschaften, Befehlen, Literalen und Notationen.	I	II
... erhält das Rüstwerkzeug um individuelle Vorstellung vom Web Development (jenseits von Vorlagen und Templates) umzusetzen.	SIK A	II
... erhält einen Einblick in die darunterliegende Ebene der Softwareentwicklung und betrachtet den kritischen letzten Eindruck eines Lernprodukt.	KI	III
... schreiben JavaScripte nach schriftlichen und graphischen Vorlagen.	SI	III
... nutzt die erlernten Fähigkeiten um imperative Algorithmen, Vorstellungen und Gedanken in schriftlich Form darzustellen.	KI	IV
... entwickelt ein höheres Professionsbewusstsein.	ASK	IV
... erwirbt die Fähigkeit selbstständig sein_ihr Wissen zu erweitern.	SIK	III
... ist befähigt ein mittleres Software-Projekt zu erledigen.	SI	IV
... steigert seine_ihre kognitive Leistungen im Bereich des linearen und logischen Denken.	KS	IV
... entwickelt Problemlösungsstrategien.	KS	IV
... nutzt Prinzipien der Programmierung mit JavaScript um eine andere imperative Programmiersprache leichter zu erlernen.	KS	III
... steigert sein_ihr Bewusstsein für Ordnung und Genauigkeit.	AS	IV
... versteht komplexe Zusammenhänge (z. B. Mathematische Formeln) und wandelt diese in einen programmierbaren Algorithmus.	KI	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

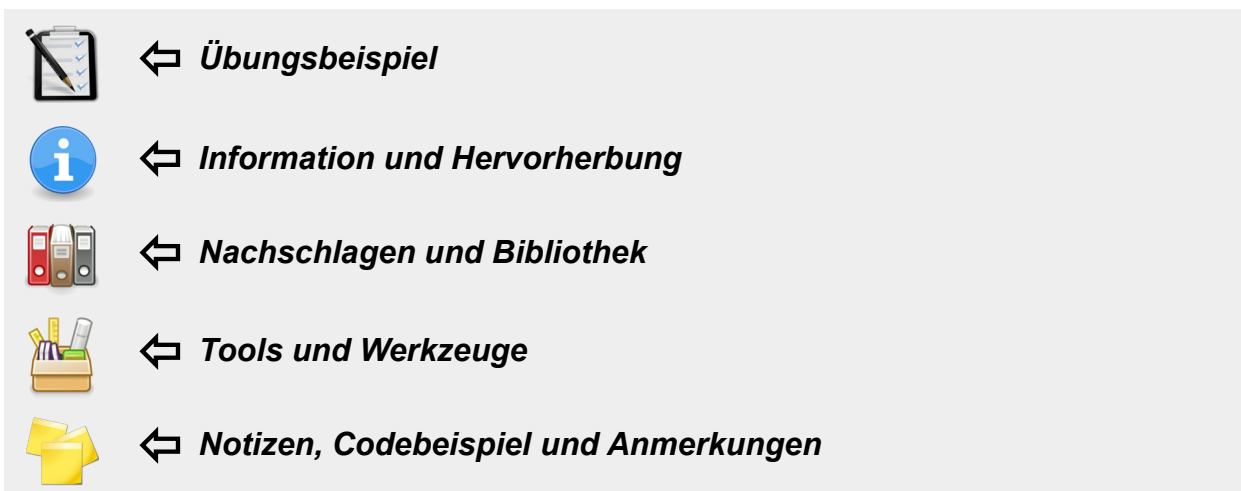
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Handouts und Assignments

Die Handouts sind in Einheiten gegliedert. Jede Einheit beschreibt einen Themenkomplex aus dem jeweiligen Bezugssystem. Grundsätzlich unterliegen die Einheiten einer Progressionslogik, können aber auch voneinander unabhängig unterrichtet werden (je nach pädagogischen Bedarf).

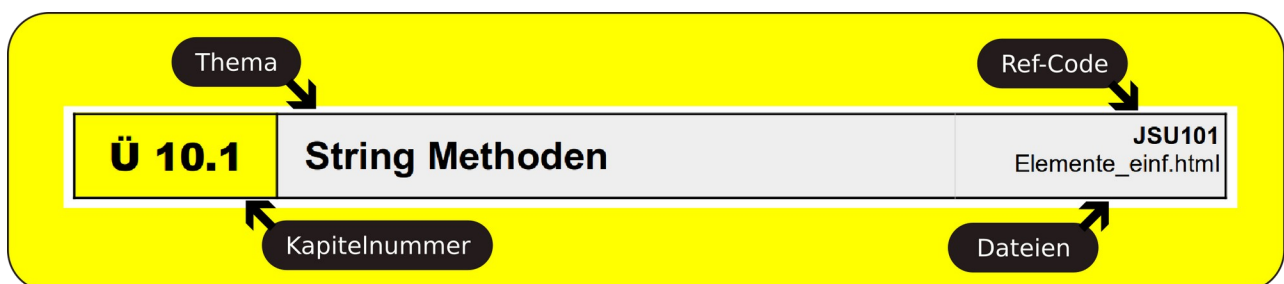
Alle Handouts wurden nach dem Prinzip des "One-Page-Management" erstellt und können auch als Kopiervorlage genutzt werden. Die Handouts wurden mit Screenshots bebildert. Die Anleitungen sind auf die Übungsbeispiele hin konzipiert.

Die Symbole auf den Handouts sollen den Studierenden eine Ordnung im Lernprozess vermitteln.



A legend box with a light gray background and rounded corners. It contains five entries, each with an icon and a text label. The labels are: 'Übungsbeispiel' (with a clipboard icon), 'Information und Hervorhebung' (with an 'i' in a blue circle icon), 'Nachschlagen und Bibliothek' (with a book icon), 'Tools und Werkzeuge' (with a toolbox icon), and 'Notizen, Codebeispiel und Anmerkungen' (with a yellow arrow icon). Each icon is followed by a left-pointing arrow and the text label.

Dieses One-Page-Management wurde der Wirtschaft entnommen. Es ist eine Methode um Führungspersonlichkeiten einen raschen Überblick über die aktuellen Fakten zu geben. Rasch und Einfach – was nun also für Manager_innen von großen Betrieben stimmig ist, sollte für den Schüler bzw. die Schülerin genauso gelten. Ein neues Thema – ein neues Blatt Papier.



Über die Kapitelnummer werden Themenbereiche gegliedert. Übungsblätter haben zusätzlich noch ein "Ü" bei der Kapitelnummer. Der Ref-Code identifiziert ein Handout eindeutig und kann im Suchfeld von www.css4.at abgerufen werden. Dateien, die für das Thema notwendig sind, findet man als rechts, unter dem Ref-Code. Die Dateien können von www.css4.at herunter geladen werden! In dieser Arbeit folgt immer zuerst die didaktische Konzeption (Feinzielen und Anmerkungen) und im Anschluss daran das Lernhandout bzw. Übungsblatt.

Lernhandout 1.0 Einführung

Referenzcode	JSL010
Technologien	HTML, JavaScript
Dateinamen	L010_Start.odt, L010_Start.pdf
Konvergenz	HCL011

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... unterscheiden zwischen Scriptsprachen und Programmiersprachen	K	I
... unterscheiden zwischen Serverseitig und Clientseitig.	K	I
... verstehen die Oberfläche eines Texteditors und arbeiten damit.	I	II
... kennen die Entwicklertools eines Browsers.	I	II
... erstellen ein HTML Grundkonstrukt als Vorlage.	I	II
... arbeiten mit dem <script> und <noscript> Element.	I	II
... nutzen die selbst erstellte Vorlage für alle weiteren Arbeiten.	I	II
... arbeiten an eigenen Organisationsfähigkeit (Dateimanagement).	S	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

Neben beliebigen Unterrichtsmethoden zum Einstieg in ein neues Thema, sollten folgende technische und didaktische Grundvoraussetzungen geklärt werden:

- Ist ein Texteditor und ein Browser vorhanden?
- Sind die Bedienelemente des Texteditors und des Browsers bekannt?
- Ist der Texteditor für die Arbeit mit HTML, CSS und JavaScript voreingestellt? (z. B. bei Notepad++ kann die Sprache JavaScript eingestellt werden)
- Ist die Kodierung des Texteditors auf UTF-8 eingestellt?
- Haben die Studierenden eine Möglichkeit ihre Lernprodukte abzuspeichern? (Beispiele: USB-Stick, USB-Festplatte, lokal am Rechner, Netzlaufwerk, Cloud usw).

Die Studierenden sollten sich angewöhnen, ihre Lernprodukte mit Meta-Tags im <head> zu beschreiben – besonders der Name des Autors und das Datum sollten das Minimum darstellen.

Das HTML Konstrukt vom Lernhandout stellt ein rudimentäres Grundkonstrukt dar. Es bezieht sich auf HTML5 (!doctype html) in der deutschen Sprache (lang = "de").

Da die meisten Browser mit der Programmiersprache JAVA erstellt wurden, erlauben diese eine Schnittstelle um mit JavaScript zusätzliche Befehle einer Webseite abzuwickeln. Wie schon im Namen JavaScript steht, handelt es sich dabei nicht um eine Programmiersprache sondern eine Scriptsprache die clientseitig abgearbeitet wird. Im Vergleich zu PHP, welches serverseitig abgearbeitet wird, ist der Quellcode für alle lesbar.

Für die Arbeit mit JavaScript benötigen wir einen Texteditor (z. B. Notepad++, oder Dreamweaver, Atom) und einen Browser (z. B. Firefox, Chrome). JavaScript harmoniert wunderbar mit HTML und CSS, deshalb werden wir es in einem HTML-Konstrukt verwenden.



HTML Konstrukt mit `<script>` Tag im Body

```
<!doctype html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <title>Titel</title>
    <meta name="author" content="Thomas Maier">
    <meta name="date" content="2022-11-10">
    <style> ... </style>
  </head>
  <body>
    <script> ... </script>
    <noscript> ... </noscript>
  </body>
</html>
```

HTML



```
<script> ... </script>
<noscript> ... </noscript>
```

Innerhalb des `<script>` Tag wird der JavaScript Code geschrieben.
Das `<script>` Element kann auch im `<head>` stehen.

Innerhalb des `<noscript>` Tag befindet sich jener Content, der angezeigt wird wenn JavaScript in den Browsereinstellungen deaktiviert ist.

z. B. `<noscript> <h2> JavaScript aktivieren! </h2> </noscript>`



Eine Vorlage erstellen

- Starte einen Texteditor (z. B. Notepad++, Dreamweaver, Atom odgl).
- Schreibe das HTML Konstrukt wie oben dargestellt.
- Schreibe in den Meta-Tag `author` deinen Vor- und Nachnamen.
- Speichere den Code als `vorlage.html` ab.**



Wir werden die `vorlage.html` öfters brauchen. Also speichere das File so ab, damit es jederzeit und schnell gefunden wird. (z. B: USB-Stick, Cloud, Netzlaufwerk). Bei jeder neuen Übung sollte das aktuelle Datum und ein Titel eingetragen werden!

Lernhandout 2.0 Ausgabe im Browser

Referenzcode **JSL020**

Technologien HTML, JavaScript

Dateinamen L020_Ausgabe.odt, L020_Ausgabe.pdf

Konvergenz HCL051

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen das EVA-Prinzip über die Ausgabe.	K	I
... nutzen <code>window.alert()</code> für eine Ausgabe in einem Dialogfenster.	I	II
... nutzen <code>document.write()</code> für eine sofortige Ausgabe ins Dokument.	I	II
... nutzen <code>.innerHTML</code> und einem ID-Selektor für eine Ausgabe in ein HTML Element.	I	II
... nutzen <code>console.log()</code> für eine Ausgabe in die Konsole des Browsers.	I	II
... arbeiten mit der Konsole des Browsers für eine etwaige Fehlersuche.	I	III
... schließen jede JavaScript Anweisung mit einem Semikolon.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Das EVA-Prinzip steht für EINGABE - VERARBEITUNG - AUSGABE. Da JavaScript eine asynchrone Scriptsprache ist, bekommt das Prinzip eine besondere Bedeutung. JavaScript nimmt alle Eingaben, verarbeitet diese und gibt dann das Ergebnis aus. Das merkt man bei Schleifen. Synchrone Programmiersprachen würden z. B. in einer Schleife jeweils eine Ausgabe erlauben - JavaScript tut dieses nicht. JavaScript verarbeitet zuerst alles bevor es ausgegeben wird. Mehr dazu in der Thematik DOM.

Die JavaScript Befehle werden innerhalb des `<script>` Elements ausgeführt.
Für die Ausgabe von Text gibt es vier Möglichkeiten.

- 1: `window.alert()` ;
- 2: `document.write()` ;
- 3: `.innerHTML = ""` ;
- 4: `console.log()` ;



Alle Anweisungen in JavaScript enden mit einem Semikolon ;
Der Text befindet sich innerhalb von zwei Anführungszeichen "text"

JS

```
window.alert();
```

Die Ausgabe erfolgt über ein Dialogfenster. Das Aussehen des Dialogfensters ist vom Browser abhängig, in den meisten Fällen gibt es eine OK Schaltfläche.
`alert()` ist eine Methode des Window Objekts.

```
<script>
  window.alert("Hallo Welt");
</script>
```

JS

```
document.write();
```

Schreibt an der Stelle im Textfluss wo sich das `<script>` Element befindet.
Man kann HTML Code in dieser Ausgabe mitgeben:

z. B: `document.write("<h1>Hallo Welt</h1>");`

```
<div> <script>
  document.write("Hallo Welt");
</script> </div>
```

JS

```
document.getElementById().innerHTML = "";
```

Schreibt in ein bestimmtes HTML Element, das mit einer ID versehen ist.
Sollte im HTML Element ein Content sein, wird dieser überschrieben!
`.getElementById("")` ist ein Selektor. `.innerHTML` ist die Methode!

```
<p id="ausgabe"> </p>
<script>
  document.getElementById("ausgabe").innerHTML = "Hallo Welt";
</script>
```

JS

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

Für die Fehlersuche (debugging) kann man eine Ausgabe in die Konsole des Browsers anweisen.
Die Konsole befindet sich in den Entwicklertools (z. B. bei Firefox).

```
<script>
  console.log("Hallo Welt");
</script>
```

Lernhandout 3.0 Variablen

Referenzcode **JSL030**

Technologien JavaScript

Dateinamen L030_Variablen.odt, L030_Variablen.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen die Variablen in JavaScript und vergleichen diese mit der Mathematik.	K	III
... vereinbaren Variablen (bzw. deklarieren, dimensionieren).	I	III
... wissen, was case-sensitive bedeutet und beachten es in Folge für ihre Lernprodukte.	I	II
... verstehen die richtige Wertzuweisung für eine Variable. (von rechts nach links).	I	I
... schreiben ein erstes Script mit Verarbeitung und Ausgabe	I	II
... sorgen jetzt schon für Ordnung und Übersichtlichkeit im Script.	SI	II
... erstellen Konstante.	I	II
... erkennen, dass ein Punkt für ein Dezimalzeichen steht.	KI	I
... arbeiten mit der Kommentarfunktion.	I	II
... erkennen die Kommentarfunktion als fundamentales Werkzeug der Programmierung (z. B. Auskommentieren von Code, Beschreiben usw.).	KSI	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Der Begriff "Behälter" wurde aus didaktischen Überlegungen gewählt. Selbstverständlich handelt es sich bei einer Variablenvereinbarung um eine Reservierung im Hauptspeicher.
- Für die Variablennamen sind alle Alphanummerischen Zeichen (ausgenommen Sonderzeichen, wie den Umlauten oder dem scharfen ß) sowie Underline und das Dollarzeichen erlaubt. Man kann auch mit `let` Variablen vereinbaren.
- Eine Typdefinition ist nicht notwendig, weil JavaScript erst durch die Zweisung den Typ bestimmt (gleich wie in PHP). Um den Überblick zu bewahren kann man den Typ als Zeichen in den Variablennamen einfügen. z. B. `izahl` für eine Integer-Variable oder `sort` für eine String-Variable.

Mit Variablen erzeugt man eine Art von Behälter der einen Wert haben kann. Gleich wie im Mathematikunterricht, wo x und y einen Wert haben, sind Variablen in JavaScript mit Zahlenwerte, Text (Strings) oder Objekten zuzuweisen. Eine Variable wird mit `var` vereinbart (erstellt, man spricht auch von deklarieren, dimensionieren).

JS



`var` Variablenname ;

Der *Variablenname* ist case-sensitive (Groß- und Kleinschreibung beachten) und sollte sich vom JavaScript-Code klar unterscheiden. Am besten verwendet man deutsche Worte.

Ohne Zuweisung, wird eine Variable als `undefined` typisiert und kann damit alle Werte übernehmen. z. B. `var flaeche;`

`var` **ausgabe** = "Das Ergebnis: "; **flaeche** = **breit** * **hoch**;

Eine Wertzuweisung erfolgt immer von rechts nach links.

```
<script>
  var breit = 15;
  var hoch = 5;
  var flaeche;
  var ausgabe = "Das Ergebnis: ";
  flaeche = breit * hoch; //← Der Stern * steht für eine Multiplikation
  document.write(ausgabe);
  document.write(flaeche);
</script>
```



Um nicht den Überblick zu verlieren, sollte man die Variablen immer ganz oben im Code deklarieren.

JS



`const` Konstantenname = Wert;

Mit `const` wird eine Konstante (ein Wert der sich nicht verändert) definiert.

```
const euler = 2.718; //← Der Punkt steht für ein Dezimalzeichen.
const Pi = 3.142;
```



Mit Kommentaren kann man den Code übersichtlicher gestalten. Sie sind manchmal auch recht nützlich bei der Fehlersuche, weil man damit einen Codeabschnitt auskommentieren kann.

Einzeilige Kommentare werden mit `//` definiert

Mehrzeilige Kommentare werden mit `/* ... */` definiert.

```
// Berechnungen
/* Ausgabe im Dokument
   document.write(ausgabe); */
```


Lernhandout 4.0 Rechenoperatoren

Referenzcode **JS1040**

Technologien JavaScript

Dateinamen L040_Operatoren.odt, L040_Operatoren.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen und nutzen die Rechenoperatoren in JavaScript.	I	II
... experimentieren mit den Beispielen zu den Rechenoperationen.	KI	III
... schreiben ein Script zur Berechnung der Umsatzsteuer.	I	II
... können Texte (Stings) miteinander verketteten.	I	II
... nutzen das Math. Objekt für weitere Mathematische Operationen.	I	II
... schreiben ein Script zur Berechnung des Flächeninhalts eines gleichseitigen Dreiecks.	KI	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

■ Zum Beispiel Umsatzsteuer:

Die 20 % entsprechen dem aktuellen Österreichischen Umsatzsteuer-Satz. In anderen Ländern kann dieser Wert verändert werden um auch zu zeigen, wie sich eine Veränderung im Code auf das Ergebnis auswirkt.

Die Unterscheidung von Brutto und Netto ist nicht nur für wirtschaftliche Schulen von Interesse. Brutto bedeutet inklusive Steuern und Netto ohne Steuern. Man kann die Unterscheidung auch anhand des Bruttogehalts vs. Nettogehalts erklären.

Ebenso ist ein "Philosophieren" über Sinn und Unsinn von Steuern durchaus denkbar!

■ Zum Flächeninhalt des Dreiecks:

Das Ergebnis hat viele Nachkommastellen. Mit der Methode `Math.round()`; können diese bereinigt werden. z. B.

```
flaeche = Math.round(flache * 100) / 100;
```

Bei einer mathematischen Operation in JavaScript gelten auch die Rangfolgen:
Zuerst die Berechnung einer Klammer und dann Punkt- vor Strichrechnung.

Operation	Zeichen Operator	Beispiel
Addition	+	<code>i = i + 1;</code>
Subtraktion	-	<code>var drei = 6 - 3;</code>
Multiplikation	*	<code>x = y * 3;</code>
Division	/	<code>zinssatz = 8 / 100;</code>
Modulo (Rest einer Division)	%	<code>var rest = 5 % 3;</code> Lösung: <code>rest = 2</code>



Script zur Berechnung der Umsatzsteuer, wenn man den Bruttowert angibt.

```
var brutto = 240;
var USt = 20;
var steuer = brutto / (100 + USt) * USt;
document.write("Die Umsatzsteuer beträgt € ");
document.write(steuer);
```



Ein einfaches Pluszeichen + dient auch zur Verkettung eines Strings (Text).

```
var ausgabe = " bei einem Steuersatz von " + USt + " Prozent";
document.write(ausgabe);
```

JS



Math.

Das Standard Objekt **Math.** bietet weitere Methoden wie z. B:

Methode	Beispiel
Potenz Math.pow(,);	<code>quadrat = Math.pow(4,2);</code> Lösung: 16
Wurzel Math.sqrt();	<code>wurzel = Math.sqrt(9);</code> Lösung: 3
Zufallszahl Math.random();	<code>var zufall = Math.random();</code>
Runden Math.round();	<code>b = Math.round(12.6);</code> Lösung: 13



Berechnung des Flächeninhalts eines gleichseitigen Dreiecks.

$$A = \frac{a^2}{4} \times \sqrt{3}$$

```
var a = 8;
var flaeche = Math.pow(a, 2) / 4 * Math.sqrt(3);
var ausgabe = "Der Flächeninhalt beträgt " + flaeche;
document.write(ausgabe);
```

Übungsblatt 4.0 Operatoren

Referenzcode **JSU040**

Technologien HTML, JavaScript

Dateinamen U040_Operatoren.odt, U040_Operatoren.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... ergänzen eine Tabelle um die korrekte Erstellung von Variablen.	I	IV
... vereinbaren korrekt Variablen.	I	II
... unterscheiden konkret bei der Wahl von Variablennamen.	III	I
... nutzen das Internet zum Selbststudium (Internetrecherche)	SIA	II
Übung B		
... übersetzen eine mathematische Formel in einen JavaScript-Code.	IK	III
... deklarieren die notwendigen Variablen.	I	II
... schreiben ein Script zur Umrechnung von Celsius in Fahrenheit.	I	IV
Übung C		
... übersetzen eine mathematische Formel in einen JavaScript-Code.	IK	III
... deklarieren die notwendigen Variablen.	I	II
... schreiben ein Script zur Lösung von quadratischen Gleichungen (Mitternachtsformel).	IK	IV
... runden das Ergebnis auf zwei Kommastellen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Formel für die Umrechnung von Fahrenheit in Celsius lautet:
 $^{\circ}\text{F} = ^{\circ}\text{C} * 1,8 + 32$



Übung A: Variablentypen

- Ergänze die Tabelle um die korrekte Erstellung von Variablen.
- Finde passende Variablennamen zum Typ.
- Erfinde passende Werte zum Typ.
- Solltest du einen Typ nicht kennen, dann recherchiere im Internet.**

Typ	Erstellung (bzw. Vereinbarung)
Fließkomma	<code>var fZahl = 4.334;</code>
Integer	
String	
Boolean	
Undefined	



Übung B: Fahrenheit und Celsius

- Übersetze die mathematische Formel der Umrechnungen von Celsius in Fahrenheit in einen JavaScript Code.
- Deklariere die notwendigen Variablen.
- Speichere den Code in ein HTML Dokument.
- Teste den Code: 45 °C sind 113 °F**



Übung C: Mitternachtsformel

- Übersetze die a-b-c Formel zur Lösung von quadratischen Gleichungen in einen JavaScript Code.
- Deklariere die notwendigen Variablen.
- Speichere den Code in ein HTML Dokument.
- Runde die Ergebnisse von x_1 und x_2 auf zwei Kommastellen!
- Teste den Code: $a = 3, b = 5, c = 1 \rightarrow x_1 = -0.23, x_2 = -1.43$**

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Lernhandout 5.0 Funktionen

Referenzcode **JSL050**

Technologien HTML, JavaScript

Dateinamen L050_Funktionen.odt, L050_Funktionen.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen Funktionsaufbau, Abruf, Übergabeparameter und Rückgabe.	I	I
... beachten die korrekte Schreibweise einer Funktion (geschwungene Klammern).	I	II
... arbeiten mit <code>return</code> ; als Rückgabewert.	I	II
... nutzen das <code>onClick</code> Attribut um Funktionen zu starten.	I	II
... schreiben eine "Runden" Funktion.	I	II
... rufen Funktionen auf mit Übergabeparameter und Rückgabewerte.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Im ersten Beispiel kann der Übergabewert geändert werden, um klar zu erklären wie ein String übergeben wird. z. B. statt 'Absenden' --> 'Schalter1', oder man übergibt einen Integerwert: `onClick="geklickt(1) ;"`
- `onClick` Attribute können in fast jedem HTML Element vorkommen (z. B. in `a href`)
- Im zweiten Beispiel liegt der Fokus auf dem Funktionsaufruf. Mit einer Änderung bei den Wertübergaben kann man die Auswirkungen gut betrachten (z. B. statt 3 nur 2 Stellen).
- Aus Platzgründen sind in den Handouts die Funktionen nicht ansehnlich schön. Hier die etwas schöner Konvention (Schreibweise) einer Funktion:

```
function aufrufname()  
{  
    Anweisung 1;  
    Anweisung 2;  
    return;  
}
```

- Eine Funktion kann selbstverständlich auch ohne Parameter sein. Man sollte sich aber angewöhnen eine Funktion immer mit `return`; zu beenden - egal ob etwas zurückgegeben wird oder nicht.

Eine Funktion kann man mit einem Unterprogramm im Quellcode vergleichen. Die Funktion wird nicht automatisch ausgeführt, sondern muss durch einen Funktionsaufruf gestartet werden. Sie hat keinen, einen oder mehrere Übernahmeparameter und meist einen oder keinen Rückgabewert. Funktionen können auch im `<head>` stehen.

JS



```
function Funktionsname (Parameter1, Parameter2, ...)
  {Anweisungen ...; return;};
```

Eine Funktion beginnt immer mit dem Befehl `function`.

Der Funktionsname soll eindeutig sein und sich von anderen Funktionen unterscheiden. Der Funktionsname ist case-sensitive.

Die Parameter sind lokale Variablen, die in der Funktion abgearbeitet werden.

Der Anweisungsblock steht innerhalb von zwei geschwungenen Klammern `{}`. Jede Anweisung endet mit einem Semikolon `;`

Mit `return` übergibt man einen Wert oder beendet die Funktion!
z. B. `return ausgabe;` oder `return true;`

```
<script>
  function geklickt(welcher) {
    window.alert("Button " + welcher + " wurde geklickt!");
    return; }
</script>

<button onClick="geklickt('Absenden');" >Senden</button>
<button onClick="geklickt('Loeschen');" >Löschen</button>
```



Das Attribut `onClick=""` im HTML Element `<button>` führt JavaScript Code aus, wenn man auf das Element klickt. Hier wird die Funktion `geklickt()` mit einem String (Absenden bzw. Loeschen) aufgerufen.



Der Funktionsaufruf `geklickt('Absenden');` wurde mit einfachen Anführungszeichen für die Wertübergabe geschrieben, weil das `onClick=""` Attribut die doppelten Anführungszeichen benötigt.



Eine eigene Rundenfunktion mit der Übergabe einer Fließkommazahl und der Anzahl der Stellen.

```
<script>
  function runden(zahl, stellen) {
    zahl = zahl * Math.pow(10, stellen);
    zahl = Math.round(zahl) / Math.pow(10, stellen);
    return zahl;}
  var ausgabe = runden(34.3862345, 3); // <-- Start der Funktion
  window.alert(ausgabe);
</script>
```



Der Funktionsaufruf wurde mit einer Variablen-Vereinbarung aufgerufen. Die lokale Variable `zahl` wurde abgearbeitet und mit `return` an die Variable `ausgabe` übergeben.

Lernhandout 6.0 Eingabe

Referenzcode **JSI060**

Technologien HTML, JavaScript

Dateinamen L060_Eingabe.odt, L060_Eingabe.pdf

Konvergenz HCL121 bis HCL128

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... realisieren Benutzereingaben über <code>window.prompt</code> .	I	II
... implementieren Benutzereingaben über Formularelemente (<code><input></code>).	I	II
... berücksichtigen das EVA Prinzip im Scripting-Prozess.	IK	III
... kennen und verwenden den Syntax von <code>window.prompt</code>	I	II
... unterscheiden zwischen einer erzwungenen Zeilenschaltung in JavaScript und dem <code>
</code> Tag als Zeilenschaltung im HTML Code.	I	III
... arbeiten mit der Selektor-Methode <code>.getElementById()</code>	I	II
... schreiben ein einfaches Script mit einem Formularelement als Eingabe, einem Buttonelement zum Start der Funktion und einer Ausgabe in ein HTML Element (vollständiges EVA-Prinzip):	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Gerade bei `document.getElementById()` muss auf die Groß-Klein-Schreibung (case-sensitive) geachtet werden. Hier entstehen oft Fehler!

Damit der User etwas auf einer Webseite eingeben kann, gibt es zwei praktikable Möglichkeiten. Erstens, eine Eingabe mit der `window.prompt()` ; Methode oder eine Eingabe über `<input>` Elemente, die mit `document.getElementById().value` ; abgerufen werden.

In der Programmierung gilt das EVA-Prinzip. EVA steht für Eingabe – Verarbeitung - Ausgabe. Das gilt sowohl für Programmier- als auch für Scriptsprachen.

JS



```
window.prompt("text", "default text");
```

Der Browser stellt ein Eingabe-Dialog-Fenster zur Verfügung. `prompt()` ; ist eine Methode des Window-Objekts. Der User kann etwas eingeben, mit OK bestätigen und diese Eingabe wird dann einer Variable zugewiesen.

`text`..... Ein Text der den Abfrage Dialog begleitet.

`default text`..... Ein Text der im Eingabe Feld steht (Placeholder).

```
<script>
  var ort = window.prompt("Ihr Wohnort?", "Wien");
  document.write("Wir senden das Paket nach " + ort);
</script>
```



Mit `\n` erzwingt man eine Zeilenschaltung.
Im HTML Code sollte man jedoch `
` verwenden.

```
window.prompt("Willkommen \n Bitte gib deinen Vornamen ein!");
```

JS



```
document.getElementById().value;
```

Ermittelt den Wert eines HTML Elements (z. B. `<input>`, `<textarea>` usw.) über seine eindeutige ID. Es ist sinnvoll, die Abfrage in eine Funktion zu schreiben und über einen Button zu starten.

```
<script>
function deinName ()
{
  var ausgabe = document.getElementById("derVorname").value;
  ausgabe = "Hallo " + ausgabe;
  document.getElementById("dieAusgabe").innerHTML = ausgabe;
  return;
}
</script>

<input id="derVorname" placeholder="Dein Vorname?">
<button onClick="deinName();">Start</button>
<p id="dieAusgabe"></p>
```


Übungsblatt 6.0 Eingabe

Referenzcode **JSU060**

Technologien HTML, CSS, JavaScript

Dateinamen U060_Eingabe.odt, U060_Eingabe.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen ein Script zur Berechnung eines Dreiecks mit dem Pythagoras.	I	IV
... verwenden Eingabefelder, Buttons und Ausgabefelder.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen ein Webdokument mit einer Berechnung von Zinseszinsen.	I	IV
... arbeiten mit komplexen Formeln der Finanzmathematik.	I	II
... erkennen die Auswirkung von Zinseszins in finanziellen Angelegenheiten.	KS	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erstellen ein JavaScript zu Deltoid Berechnungen.	I	IV
... verwenden Eingabefelder, Buttons und Ausgabefelder.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- **Übung B:** Zinseszins-Berechnungen können weiter thematisiert werden → z. B. in Verbindung mit Krediten, Giro-Konto-Überziehungen, verantwortliches Konsumverhalten (Ratenkauf) usw.



Übung A: Pythagoras

Im rechtwinkligen Dreieck gilt für die drei Seiten: $a^2 + b^2 = c^2$
Der Benutzer gibt die Werte von a und b ein und erhält die Länge von c.
Erstelle auch einen Button zum Starten der Funktion.

- HTML Zwei Eingabefelder, ein Button, ein Ausgabefeld
- JS Verarbeitung in einer Funktion
- css Ansprechende Gestaltung (Freies Design)
- Teste deine Seite [a = 4, b = 3, c = 5]



Übung B: Zinseszins

Schreibe ein Skript, welches durch Eingabe von Anfangskapital, Zinssatz und einer Laufzeit in Jahren das **Endkapital** einer Spareinlage errechnet und ausgibt.

Die Formel für eine Zinseszins-Rechnung lautet:

- HTML Drei Eingabefelder, ein Button und ein Ausgabefeld
- JS Verarbeitung in einer Funktion
- css Ansprechende Gestaltung (Freies Design)
- Teste deine Seite

$$K_0 = 1200, p = 4, n = 5, K_n = 1459,98$$

$$K_n = K_0 \times \left(1 + \frac{p}{100}\right)^n$$

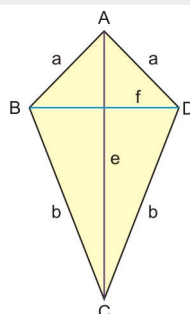
Kn _____ Endkapital
K0 _____ Anfangskapital
p _____ Zinssatz (z. B. 4 %)
n _____ Laufzeit in Jahren



Übung C: Deltoid

Auf einer Webseite sollen die Längen von a, b, und f eines Deltoids eingegeben werden. Ausgegeben werden soll der Flächeninhalt.

- HTML Drei Eingabefelder, ein Button, ein Ausgabefeld
- JS Verarbeitung in einer Funktion
- css Ansprechende Gestaltung (Freies Design)
- Teste deine Seite [a = 5, b = 8, f = 6, A = 34,25]



$$A = \frac{e \cdot f}{2}$$

$$e = \sqrt{a^2 - \left(\frac{f}{2}\right)^2} + \sqrt{b^2 - \left(\frac{f}{2}\right)^2}$$

Lernhandout 6.1 Ereignis Attribute

Referenzcode **JSL061**
Technologien HTML, JavaScript
Dateinamen L061_Event.odt, L061_Event.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... nutzen Ereignis-Attribute (HTML) um JavaScript Code bzw. Funktionen auszuführen.	I	II
... beachten die Schreibweise von Anführungszeichen (einfache in doppelten).	KI	II
... nutzen Window-Event-Attribute.	I	II
... können die Höhe des Viewports ermitteln.	I	II
... nutzen Form Event-Attribute.	I	II
... nutzen Keyboard und Mouse-Event-Attribute.	I	II
... schreiben ein Script um ein "verhülltes" Passwort anzuzeigen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Da jedes Ereignis-Attribut zwei doppelte Anführungszeichen benötigt, werden einfache Anführungszeichen innerhalb der Anweisung verwendet:
`<body onLoad="window.alert('Servus ');">`

Ereignis Attribute (event attributes) werden für den Aufruf von JavaScript (Funktionen oder direkt ein Code) verwendet. Achte dabei auf die Anführungszeichen.

```
<body onLoad="window.alert('Servus');">
<button onClick="MeineFunktion();">Absenden</button>
```

Window Events	onBeforePrint	← Bevor ein Dokument gedruckt wird.
	onError	← Wenn ein Fehler auftritt.
	onLoad	← Nachdem ein Dokument/Objekt geladen wurde.
	onResize	← Wenn sich die Größe des Browserfenster verändert.
	onUnload	← Nachdem ein Browserfenster geschlossen wurde.

```
<body onResize="WieHoch();">
<p>Die Viewport-Höhe: <span id="brz"></span> Pixel</p>
<script>
function WieHoch() {
    var hoch = window.innerHeight;
    var ausgabe = "Viewport-Höhe: " + hoch + "px";
    document.getElementById("brz").innerHTML = ausgabe; }
</script>
```



window.innerHeight ermittelt die Höhe des Viewports in Pixel.

Form Events	onChange	← Wenn sich der Wert verändert.
	onFocus	← Wenn der Fokus auf dem Element ist.
	onSelect	← Wenn ein Text in der Eingabe markiert wird.
	onBlur	← Wenn man das Eingabefeld verlässt.
	onInput	← Bei jeder Eingabe im Eingabefeld (Tastendruck).
Keyboard und Mouse Events	onKeyDown	← Bei einem Tastendruck.
	onKeyUp	← Nach einem Tastendruck.
	onClick	← Bei einem Mausklick (linke Maustaste).
	onDBLclick	← Bei einem Doppelklick (linke Maustaste).
	onMouseOut	← Mauszeiger verlässt ein Element.
	onMouseOver	← Mauszeiger bewegt sich über dem Element.
onWheel	← Wenn das Mousrad gedreht wird.	

```
<input type="password" id="Passwort" onKeyUp="zeige();">
<p id="Anzeige">Hier ist das Passwort lesbar!</p>
<script>
function zeige() {
    var ausgabe = document.getElementById("Passwort").value;
    document.getElementById("Anzeige").innerHTML = ausgabe; }
</script>
```

Lernhandout 7.0 Verzweigung

Referenzcode **JSL070**

Technologien JavaScript

Dateinamen L070_Verzweigung.odt, L070_Verzweigung.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... begreifen den Aufbau einer Verzweigung.	K	III
... arbeiten mit if-else Verzweigungen und kennen die Syntax.	I	II
... unterscheiden zwischen Bedingung und Anweisung.	I	I
... kennen die unterschiedlichen Vergleichsoperatoren in einer Verzweigung.	I	II
... schreiben ein Script mit einer Verzweigungen um die größere Zahl von zwei Zahlen zu ermitteln.	I	III
... nutzen mehrere Bedingungen in Verbindung mit logischen Operatoren.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Verzweigungen kann man auch als Fallunterscheidungen verstehen.
- Der == Operator (istgleich) wird in JavaScript immer mit mind. zwei Ist-Gleich-Zeichen definiert. Ein Ist-Gleich-Zeichen wird für Wertzuweisungen verwendet. Wenn drei Ist-Gleich-Zeichen (===) genutzt werden dann muss neben dem Wert auch die Typdefinition gleich sein.
- Mit Number() wird die Eingabe in eine Zahl gewandelt. Das ist deshalb notwendig, weil sonst eine Eingabe von 111 kleiner wäre als 22.
- Ein oft gemachter Fehler entsteht die Semikolonsetzung. Nach der abschließenden geschwungenen Klammer } darf kein Semikolon gesetzt werden!

Eine Verzweigung besteht aus Bedingungen und Anweisungen. Wird eine Bedingung erfüllt (true), dann werden auch die Anweisungen ausgeführt. Wird die Bedingung (bzw. die Bedingungen) nicht erfüllt (false), dann werde die Anweisungen des Else Block ausgeführt, wenn ein solcher vorhanden ist.

JS



```
if (Bedingung) {Anweisung; Anweisung; Anweisung; usw.}
else {Anweisung; Anweisung; Anweisung; usw.}
```

Für die Definition einer Bedingung gibt es Vergleichsoperatoren. Die Operatoren == **istgleich** und != **ungleich** können für alle Typen (Text, Boolean, Nummer usw.) verwendet werden.

```
var eingabe = window.prompt("Wie viel ist 7 x 8?");
if (eingabe == 56) {window.alert("Richtig"); }
else {window.alert("Falsch"); }
```

Operator	Bedeutung	Beispiel
==	istgleich	if(x == 2) {window.alert"Zwei gewählt";}
!=	ungleich	if(y != 10) {x = x / 100;}
>	größer	if(eingabe > 50) {document.write(eingabe);}
>=	größergleich	if(breite >= 1024) {window.alert"Zu groß";}
<	kleiner	if(wert < 0) {ausg = "Keine negativen Zahlen";}
<=	kleinergleich	if(breite <= 1024) {schalter = true;}



Beispiel: Die größere von zwei Zahlen ermitteln.
Number () wandelt die Eingabe in eine Zahl!

```
var benutzer = window.prompt("Welcher Benutzer?", "User");
var zahl_1 = Number(window.prompt("Erste Zahl eingeben!"));
var zahl_2 = Number(window.prompt("Zweite Zahl eingeben!"));
if(zahl_1 > zahl_2) {var ausgabe = "Erste Zahl ist größer";}
else {var ausgabe = "Zweite Zahl ist größer";}
if(zahl_1 == zahl_2) {var ausgabe = "Beide Zahlen sind gleich";}
window.alert(ausgabe);
```



Es können auch mehrere Bedingungen auf einmal definiert werden. Dazu verbindet man die Bedingungen mit den logischen Operatoren && oder ||
&& bedeutet UND also alle Bedingungen müssen erfüllt werden
|| bedeutet ODER also eine einzige Bedingung reicht aus.

```
if (benutzer == "User" && zahl_1 == zahl_2)
{window.alert("User hat zwei gleiche Zahlen eingegeben!");}
if (benutzer == "Admin" || benutzer == "Root")
{window.alert("Egal, Root bzw. Admin darf sowieso alles!");}
```

Übungsblatt 7.0 Verzweigung

Referenzcode JSU070

Technologien HTML, CSS, JavaScript

Dateinamen U070_Verzweigung.odt, U070_Verzweigung.pdf

Konvergenz HCL127, HCL121 bis HCL126

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben ein Script, welches zwischen den Geschlechtern unterscheidet und dem gemäß eine passende Anrede generiert.	I	IV
... nutzen HTML Radio-Buttons und werten diese mit JavaScript aus.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... scripten ein Webdokument, welches Bestandteil eines Webshop ist und einen Rabatt bei einer bestimmten Abnahmemenge ermittelt.	I	IV
... orientieren sich an einer bebilderten Vorlage.	K	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... scripten ein Webdokument, welches unterschiedliche Systemvoraussetzungen einer Software ermittelt.	I	IV
... beschäftigen sich mit Hardware-Spezifikationen.	I	I
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: Frau oder Mann

In einem Formular wird das Geschlecht (männlich oder weiblich) über Radio-Buttons abgefragt. Zusätzlich noch der Nachname. Bei weiblich wird "Sehr geehrte Frau" und bei männlich "Sehr geehrter Herr" mit dem Nachnamen ausgegeben.

- HTML** zwei Radiobuttons, ein Eingabefeld, einen Button, ein Ausgabefeld
- JS** eine Funktion mit Verzweigung
- CSS** Ansprechende Gestaltung (Freies Design)

HTML

```
<input type="radio" id="mann" name="geschlecht" >
<input type="radio" id="frau" name="geschlecht" checked >
```

JS

```
var Mann = document.getElementById("mann").checked;
```



*Auswertung eines Radiobuttons. **Mann** ist eine boolesche Variable. Ist ein Radiobutton aktiviert, wird **true** an die Variable übergeben.*



Übung B: Rabattberechnung

Ein Olivenöl-Hersteller vermarktet sein Öl über das Internet. Die 1-Liter Flasche kostet € 12,40. Bestellt man 8 oder mehr Flaschen, bekommt man 5 % Rabatt. Bestellt man 16 oder mehr Flaschen, bekommt man 12 % Rabatt auf die Gesamtsumme.

- HTML** ein Eingabefeld, einen Button, ein Ausgabefeld
- JS** eine Funktion mit Verzweigung, Berechnung der Gesamtsumme und des Rabatts.
- CSS** Ansprechende Gestaltung (Freies Design)

Rabattberechnung

Wie viele Flaschen á € 12,40 möchten Sie bestellen?

Anzahl Flaschen

Sie haben 11 Flaschen bestellt!
 Sie bekommen 5 % Rabatt.
 Brutto: € 136,4 - Rabatt: € 6,82 = € 129,58



Übung C: Systemvoraussetzungen

Ein Softwarehersteller hat drei Versionen der 3D Anwendung "Threedim" auf den Markt gebracht. Jede Version hat unterschiedliche Systemvoraussetzungen.

- Threedim Lite (RAM: 2 GiB, Prozessor: 1.8 GHz, Festplatte: 30 GiB)
- Threedim Full (RAM: 4 GiB, Prozessor: 2 GHz, Festplatte: 40 GiB)
- Threedim Prof (RAM: 16 GiB, Prozessor: 3 GHz, Festplatte: 120 GiB)
- Erstelle eine Webseite mit der Eingabe für die Größe von RAM, Prozessor und Festplatte. Ein Script soll dann die möglichen Software-Versionen ermitteln und ausgeben.
- HTML** Drei Eingabefelder, ein Button
- JS** eine Funktion mit Verzweigungen
- CSS** Ansprechende Gestaltung (Freies Design)

Lernhandout 7.1 switch case

Referenzcode **JSL071**

Technologien JavaScript

Dateinamen L071_switch.odt, L071_switch.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... erkennen eine Switch Case Verzweigung als Alternative zu if-else	KI	I
... kennen die Syntax einer switch case Anweisung. (switch, case, default und break;)	I	II
... schreiben das Beispiel: "Produktionsstätten von deutschen Automarken".	I	II
... unterscheiden zwischen lokalen und globalen Variablen.	I	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Im Beispiel stehen die Automarken unter Anführungszeichen, weil es sich um eine String-Abfrage handelt. Wäre es ein Integer-Wert (der zu erwarten ist), dann wäre das case selbstverständlich ohne Anführungszeichen. Die Sonderzeichen sind immer das dramatische Moment in der Programmierung mit JavaScript.
- Bei Vorzeigen und Nachmachen ist zu beachten, dass das Beispiel case-sensitive ist. Mit anderen Worten: `bmw` ist nicht gleich `BMW`.

Alternativ zu einer IF-Else Verzweigung gibt es die Switch Case Verzweigung. Dabei wird mit **switch** eine Variable ausgewählt und über **case** die Bedingung definiert. Die **case** Anweisungen werden mit **break;** beendet.

JS

```
switch (var) {  
  case 1: Anweisung;  
          Anweisung;  
          break;  
  case 2: Anweisung;  
          Anweisung;  
          break;  
  default: Anweisung;  
          break; }
```



Nach **case** wird der Wert zur Bedingung angeführt und mit einem Doppelpunkt abgeschlossen. Mit **break;** beendet man den Anweisungsblock. **default:** wird dann ausgeführt, wenn keine andere **case**-Bedingung übereinstimmt.



Beispiel: Produktionsstätten von deutschen Automarken

```
<script>  
var prodOrt = "";  
function meineFunktion() {  
  var dasAuto = document.getElementById("deAuto").value;  
  
  switch(dasAuto) {  
    case "BMW":  
      prodOrt = "München"; break;  
    case "VW":  
      prodOrt = "Wolfburg"; break;  
    case "Opel":  
      prodOrt = "Rüsselsheim am Main"; break;  
    case "Porsche":  
      prodOrt = "Stuttgart"; break;  
    default: prodOrt = "Gültige Automarke eingeben!"; break;  
  }  
  document.getElementById("ausgabe").innerHTML = prodOrt;  
}  
</script>  
  
<h3>Geben Sie bitte eine deutsche Automarken ein!</h3>  
<input type="text" id="deAuto" onKeyUp="meineFunktion();" >  
<p id="ausgabe"></p>
```



Variablen die in einer Funktion vereinbart werden, sind auch nur innerhalb der Funktion gültig. Variablen außerhalb, sind globale Variablen und stehen jeder Funktion zur Verfügung.

Im Beispiel oben ist: var prodOrt = ""; eine globale Variable.

Übungsblatt 7.1 switch case

Referenzcode JSU071

Technologien HTML, CSS, JavaScript

Dateinamen U071_switch.odt, U071_switch.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben ein Script zur Auflösung einer Postleitzahl in die Leitzone (Bundesland).	I	IV
... verwenden dafür eine switch case Verzweigung.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... schreiben eine Gegenüberstellung von if-else und switch-case	I	IV
... reflektieren auf die Vor- und Nachteile beider Verzweigungen.	KI	III
... erarbeiten ein vollständiges Verständnis für die Thematik und vertiefen selbstständig durch Internetrecherche.	ASI	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: Postleitzahl

Noch während der Eingabe einer Postleitzahl soll das dazupassende Bundesland angezeigt werden. Die erste Ziffer einer Postleitzahl steht für das Bundesland bzw. die Leitzone. **Quelle:**
[https://de.wikipedia.org/wiki/Postleitzahl_\(Österreich\)](https://de.wikipedia.org/wiki/Postleitzahl_(Österreich))

- HTML ein Eingabefeld (passendes Ereignisattribut), ein Ausgabefeld
- JS **eine Funktion mit switch case**
- css **Ansprechende Gestaltung (Freies Design)**

1xxx	Wien (auch teilweise für benachbarte Orte in Niederösterreich)
2xxx	östliches und südliches Niederösterreich, auch teilweise für Nordburgenland
3xxx	westliches Niederösterreich und kleine Teile südöstliches Oberösterreich
4xxx	Oberösterreich und kleine Teile westliches Niederösterreich
5xxx	Salzburg und westliches Oberösterreich
6xxx	Nordtirol und Vorarlberg
7xxx	Burgenland, soweit es nicht unter 2xxx oder 8xxx fällt
8xxx	Steiermark, Teile des Bezirks Bezirk Jennersdorf im Südburgenland
9xxx	9xxx – Kärnten und Osttirol



Übung B: Gegenüberstellung if vs. switch

Erstelle eine Webseite mit einer Gegenüberstellung (z. B. Vor- und Nachteile, Syntax, Schreibarbeit, Übersichtlichkeit usw) von **IF ELSE** vs. **SWITCH CASE**

Recheriere im Internet und baue eigene Überlegungen mit ein.

- HTML **HTML Dokument**
- JS **Kein Code notwendig**
- css **Ansprechende Gestaltung (Freies Design)**

IF ELSE

VS.

SWITCH CASE

--	--

Lernhandout 8.0 Schleifen

Referenzcode **JSL080**

Technologien JavaScript

Dateinamen Blatt.odt, Blatt.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen die Funktion einer Schleife.	K	I
... kennen die Syntax und Funktion einer kopfgesteuerten Schleife.	KI	III
... nutzen die Kurzschrift eines Zählers in JavaScript (++)	I	II
... kennen die Syntax und Funktion einer fussgesteuerten Schleife.	KI	III
... erkennen das Problem einer Endlosschleife.	KI	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Keine

Mit einer Schleife wiederholt man einen Anweisungsblock. **while** Schleifen eignen sich für Durchläufe, deren Anzahl noch nicht bekannt ist.

Kopfgesteuerte Schleife

JS



```
while (Bedingung) {Anweisung; Anweisung; usw.}
```

Für die Bedingungen gelten die selben Regeln wie bei einer IF-ELSE Verzweigung. Siehe 7.0 Verzweigung (Vergleichsoperatoren). Die Anweisungen werden solange wiederholt, solange auch die Bedingung erfüllt ist (true).

```
<p>Du musstest <span id="ausgabe"></span> mal raten!</p>
<script>
  var eingabe = "";
  var versuch = 0;
  while (eingabe != "Innsbruck") {
    eingabe = window.prompt("Hauptstadt von Tirol?");
    versuch++;
    document.getElementById("ausgabe").innerHTML = versuch;
  }
</script>
```



versuch++; ist ein Zähler. Dabei wird pro Durchgang eines hinzuaddiert. Es entspricht einem **versuch = versuch + 1**;

Fussgesteuerte Schleife

JS



```
do {Anweisung; Anweisung; usw.}
while (Bedingung);
```

Der Anweisungsblock wird einmal sicher durchlaufen. Am Ende werden die Bedingungen mit `while ()`; definiert, die ein weiteres durchlaufen bestimmen sollen. Wird die Bedingung erfüllt (true), dann wird der Anweisungsblock nochmals ausgeführt.

```
var eingabe = "";
var zaehler = 0;
do {eingabe = window.prompt("Mathequiz: 4 x 3 = ?");
  zaehler ++;
  if (eingabe == 12) {
    window.alert("Bravo! " + zaehler + " Versuche");
    break; }
  }
while (eingabe != 12);
```



Mit **break**; verlässt man sofort eine Schleife!



!!! ACHTUNG !!! Wenn die Bedingungen falsch gesetzt sind, kann eine Endlosschleife entstehen. Hier ein Beispiel für eine Endlosschleife:

```
var zaehler = 5;
do {zaehler++; window.alert(zaehler);}
while (zaehler > 5);
```

Lernhandout 8.1 for Schleifen

Referenzcode **JSL081**

Technologien JavaScript

Dateinamen L081_Schleifen.odt, L081_Schleifen.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen die Funktion von Zählschleifen.	K	I
... verstehen Zählvariablen.	KI	I
... verstehen Bedingungen mit Vergleichsoperatoren in einer Zählschleife.	KI	I
... verstehen die Zählvarianten.	KI	I
... schreiben ein Script, welches die Schriftgröße in Durchläufen verändert.	I	II
... schreiben ein Script, welches Buchstabe für Buchstabe eines Strings ausgibt.	I	II
... verstehen eine <code>for-in</code> Schleife für einen String.	I	I

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Keine

Wenn man abschätzen kann, wie viele Durchgänge (Wiederholungen bzw. Iterationen) ein Anweisungsblock zu durchlaufen hat, dann verwendet man am Besten for Schleifen. Diese haben eine Zählvariable, eine Bedingung für die Zählvariable und eine Zählvariante die entweder höher oder niedriger wird.

JS



```
for (Zählvariable; Bedingung; Zählvariante)
  {Anweisungen;}
```

Zählvariable eine Zählvariable wird mit `var` vereinbart und definiert den Startwert z. B. `var i = 5;`
Die Zählvariable kann man für die Anweisungen verwenden.

Bedingung für die Zählvariable wird eine Bedingung mit Vergleichsoperatoren definiert. z. B. `i <= 50;`

Zählvariante um welchen Betrag sich die Zählvariable verändert. Der Betrag kann größer oder kleiner werden – abhängig von der Zählvariante. z. B. `i++` oder `i = i - 1`

```
var zahl = window.prompt("Bis zu welcher Zahl?");
for (var i = 1; i <= zahl; i++) {
  document.write(i + "<br>"); }
```



Verändert die Schriftgröße um plus 2 Pixel pro Durchlauf. Dazu brauchen wir ein HTML Element welches unser Vorhaben schön darstellt:

```
<p style="font-size: 10px;">Schrift mit 10 Pixel Größe</p>
```

```
const anfang = '<p style="font-size:';
const mitte = 'px">Schrift mit ';
const schluss = ' Pixel Größe</p>';
for (var i = 10; i <= 46; i = i + 2) {
  document.write(anfang + i + mitte + i + schluss); }
```



Beispiel: Buchstabe für Buchstabe

```
<script>
  var meinText = window.prompt("Überschrift?");
  for (var i in meinText) {
    document.write('<p>' + meinText[i] + '</p>');
  }
</script>
```



Durch die Bedingung (`var i in meinText`) wandert die Zählschleife Buchstabe für Buchstabe den String der Variable `meinText` ab (von links nach rechts).

Ausgegeben wird der Buchstabe an der Position `i` → `meinText[i]`

Mehr dazu in der Einheit über Arrays.

Übungsblatt 8.1 Schleifen

Referenzcode **JSU081**

Technologien HTML, CSS, JavaScript

Dateinamen U081_Schleifen.odt, U081_Schleifen.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Abfrage Schleife für die Eingabe eines Geschlechts.	I	II
... nutzen logisches Operatoren für die Abbruchbedingung.	I	II
... verwenden eine kopf- bzw. fussgesteuerte Schleife.	I	II
Übung B		
... schreiben ein Script zur Ausgabe der Zahlen 1 bis 10.	I	II
... verwenden alle drei Schleifen (Kopf- und Fussgesteuert, For Schleifen).	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... schreiben ein Script zur Ausgabe von Primzahlen innerhalb eines Zahlenbereich.	I	IV
... finden einen Algorithmus für die Ermittlung von Primzahlen.	KI	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung D		
... schreiben ein Webdokument zur Ermittlung von Koordinatenpunkten in einer quadratischen Funktion.	KI	IV
... fügen die nötigen Eingabefelder hinzu und nutzen für die Ausgabe eine HTML Tabelle.	I	II
... orientieren sich an den Textvorgaben und der bebilderten Vorlage.	KI	III
... nutzen Variablenkonvertierungen (String in Dezimalzahl mit <code>parseFloat</code>).	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln



Übung A: Geschlecht eingeben

Eine Benutzerin oder ein Benutzer wird nach ihrem bzw. seinem Geschlecht gefragt. Die Frage soll solange wiederholt werden, bis sie oder er ein »w« oder ein »m« eingibt.

- JS **Schleife, Eingabeprompt**



Wenn man mehrere Bedingungen angeben will, dann verknüpft man diese mit &&.



Übung B: Die drei Schleifen

Schreibe ein Script, das alle ganzen Zahlen von 1 bis 10 ausgibt. Verwende dafür alle drei Schleifen (Kopf- und Fussgesteuerte, Zählschleifen).

- HTML **drei Ausgabefelder**
- JS **drei Schleifen für die Ausgabe der Zahlen 1 bis 10**
- CSS **Ansprechende Gestaltung (Freies Design)**



Übung C: Primzahlen

Es soll ein Zahlenbereich (von – bis) definiert werden. Per Mausklick auf einen Button sollen alle Primzahlen innerhalb des Zahlenbereichs ausgegeben werden.

- HTML **Zwei Eingabefelder, ein Button, ein Ausgabefeld**
- JS **Funktion mit Schleife, IF-Verzweigung für die Primzahlenermittlung**
- CSS **Ansprechende Gestaltung (Freies Design)**



Übung D: Quadratische Funktion

Eine quadratische Funktion hat die Form $f(x) = ax^2 + bx + c$.

Schreibe eine Webseite mit der Eingabe für a, b, c. Damit ist die Funktion berechenbar. In Folge kann der User einen Zahlenbereich für x eingeben und eine Schrittweite für x – die dann f(x) bzw. y ausgibt.

- HTML **sechs Eingabefelder, eine Ausgabetablelle**
- JS **Funktion mit Schleife**
- CSS **Ansprechende Gestaltung (Freies Design)**



*Mit `parseFloat()` ; kann man einen String in eine Dezimalzahl umwandeln.
z. B. `var fx = parseFloat(y)` ;*

Übung D

Quadratische Gleichung

$f(x) = ax^2 + bx + c$

a:

X Anfang:

b:

X Ende:

c:

X Schritte:



x	f(x)
1	9
2	18
3	31
4	48
5	69
6	94
7	123
8	166

Lernhandout 9.0 Arrays

Referenzcode **JSL090**

Technologien JavaScript

Dateinamen L090_Arrays.odt, L090_Arrays.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen was ein Array bzw. Feld ist.	IK	I
... kennen die Schreibweise von Arrays.	I	II
... erstellen ein Array mit Wochentagen und rufen einen bestimmten Tag ab.	I	II
... verändern Array-Werte mittels einer Zuweisung.	I	II
... zeigen das komplette Array an.	I	II
... vereinbaren ein leeres (neues) Array.	I	II
... greifen mittels einer Variable auf ein bestimmtes Element im Array zu.	I	II
... nutzen die <code>.length</code> Eigenschaft um die Anzahl der Elemente eines Array zu erheben.	I	II
... greifen mittels der <code>.length</code> Eigenschaft auf das letzte Element eines Array zu.	I	II
... nutzen <code>.length</code> um von einem String die Anzahl der Zeichen zu ermitteln.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Mit der `.length` Eigenschaft auf das letzte Array zugreifen? Insbesondere, wenn man nicht weiß, wieviele Elemente das Array hat - also wenn es dynamisch wächst bzw. schrumpft.

Ein Array (Feld) ist eine Variable, die mehrere Werte speichern kann. Die Positionen der Werte werden von links nach rechts ins Array vergeben. Die erste Position hat die Nummer 0, die zweite die Nummer 1 usw.

Ein leeres Array wird mit `var arrayname = []`; vereinbart. Je nach Typ, muss man die Schreibweise beachten. So wird ein String mit Anführungszeichen und ein Integerwert nur durch einen Beistrich (ohne Anführungszeichen) getrennt.

z. B. `var alter = [32, 28, 12, 54]`;

JS



```
var arrayname = [Wert0, Wert1, Wert2];
```

Die Werte können von jedem Typ sein (z. B.: String, Integer usw.). Den Wert ruft man mit `arrayname[0]` ab. Im Beispiel unten, wird Dienstag ausgegeben.

```
var tage = ["Sonntag", "Montag", "Dienstag",
            "Mittwoch", "Donnerstag", "Freitag", "Samstag"];
window.alert(tage[2]);
```



Einen Wert verändert man gleich, wie eine bekannte Variablenzuweisung. Im Beispiel wird "Montag" durch "Wochenbeginn" ersetzt.

```
tage[1] = "Wochenbeginn";
window.alert(tage[1]);
```



Ohne Nummer, wird der gesamte Inhalt des Array angezeigt.

```
console.log(tage);
```



Natürlich kann die Nummer auch durch eine Variable gesetzt werden. Im Beispiel wird der Wochentag "Donnerstag" angezeigt, weil die Variable `xy` gleich 4 ist.

```
var xy = 4;
window.alert(tage[xy]);
```

Die `.length` Eigenschaft

JS



```
.length;
```

Die `.length` Eigenschaft gibt die Anzahl der Elemente eines Arrays zurück. Mit der `.length` Eigenschaft kann man auf das letzte Element zugreifen.

```
window.alert(tage.length);
window.alert(tage[tage.length - 1]);
```



Die `.length` Eigenschaft kann auch auf eine String Variable angewandt werden. Dabei wird die Anzahl der Zeichen im String zurückgegeben.

```
var meinText = "Wochentage";
window.alert(meinText.length);
```

Übungsblatt 9.0 Arrays

Referenzcode **JSU090**

Technologien HTML, CSS, JavaScript

Dateinamen U090_Arrays.odt, U090_Arrays.pdf, Dienstgrade.zip

Konvergenz HCL022

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen eine Webseite mit der Darstellung der österreichischen Offiziere.	I	II
... erzeugen ein Array mit den Dienstgraden.	I	II
... nutzen Variablen um auf ein Array zuzugreifen.	I	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erweitern die Übung A.	I	II
... fügen ein Range-Slider Element hinzu.	I	II
... greifen mit JavaScript auf das Range-Slider Element (value) zu.	I	II
... implementieren eine Bildanzeige, je nach gewählten "Dienstgrad".	I	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Militäraffektive Schüler_innen sollen über diese Übungen stärker involviert werden. Es sollte eine wertfreie Auseinandersetzung der militärischen Strukturen angestrebt werden.
- Wenn diese Übungen aufgrund einer (berechtigt) patriotischen Grundstimmung nicht möglich ist, dann sind folgende Änderungen denkbar:
 - Hierarchien einer Kirche
 - Kader einer Fussballmannschaft (mit Triko-Nummern und Fotos, Urheberrechte beachten).
 - Länder der Europäischen Union mit Flaggen



Übung A: Österreichische Offiziere

Erstelle eine Webseite, in welcher ein Benutzer die Zahl einer Rangfolge eingibt und dann die Bezeichnung des Offiziersrang angezeigt wird.

Quelle: <http://www.bundesheer.at/abzeichen/dienstgrade.shtml>

- HTML Eingabefeld, Ausgabefeld, Button
- JS Funktion mit einem Array
- css Ansprechende Gestaltung (Freies Design)

0. Fähnrich		6. Oberst	
1. Leutnant		7. Brigadier	
2. Oberleutnant		8. Generalmajor	
3. Hauptmann		9. Generalleutnant	
4. Major		10. General	
5. Oberstleutnant			



Übung B: Österreichische Offiziere II

Erweitere das Beispiel A um die Anzeige der Abzeichen. Verwende für die Eingabe einen Range Slider `<input type = "range" ... >`

- HTML Range Slider, Ausgabefeld, Bilder
- JS Funktion mit Array
- css Ansprechende Gestaltung (Freies Design)

Übung B

Offiziersränge

Niedrigster
|
|
|
Höchster

Brigadier
(7. Rang)

Lernhandout 9.1 Array Methoden

Referenzcode **JSL091**

Technologien JavaScript

Dateinamen L091_Arrays.odt, L091_Arrays.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... nutzen die unterschiedlichen Array-Methoden um effektiv mit Feldern zu arbeiten.	SI	II
... fügen ein neues Element an vorderster Stelle in ein Array ein. <code>unshift()</code> ;	I	II
... löschen das erste Element eines Arrays. <code>shift()</code> ;	I	II
... fügen am Schluss ein neues Element hinzu. <code>push()</code> ;	I	II
... löschen das letzte Element aus einem Array. <code>pop()</code> ;	I	II
... fügen bzw. löschen ein oder mehrere Elemente aus einem Array. <code>splice()</code> ;	I	II
... wandeln ein Array in einen String mit Trennzeichen. <code>join()</code> ;	I	II
... ermitteln den Typ einer Variable mit <code>typeof(var)</code> ;		

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



.unshift() – Vorne an Position [0] ein neues Element hinzufügen

JS

arrayname.unshift()

```
var sport = ["Golf", "Tennis", "Fussball"];
sport.unshift("Skispringen");
window.alert("Die Sportarten: " + sport);
```

.shift() – Das erste Element löschen

JS

arrayname.shift()

```
var wegdamit = sport.shift();
window.alert("Gelöscht wurde: " + wegdamit);
window.alert("Übrig bleibt: " + sport);
```

.push() – Ein Element am Schluss hinzufügen

JS

arrayname.push()

```
var neuerSport = window.prompt("Neue Sportart hinzufügen");
sport.push(neuerSport);
sport.push("Tanzen");
window.alert(sport);
```

.pop() – Löscht das letzte Element aus einem Array

JS

arrayname.pop()

```
sport.pop();
window.alert(sport);
```

.splice() – Ein Element einfügen oder löschen

JS

arrayname.splice(Position, Löschen, neue Elemente)



Position	An welcher Stelle das neue Element eingefügt werden soll.
Löschen	Wie viele Elemente gelöscht werden sollen. Wenn man kein Element löschen will, dann gibt man eine 0 an.
neue Elemente	Die Werte, die hinzugefügt werden sollen.

```
sport.splice(2, 1, "Reiten", "Laufen", "Ice Hockey");
window.alert(sport);
```

.join() – Umwandlung in einen String mit Trennzeichen

JS

arrayname.join(" - ")

```
var meineSportarten = sport.join(" und ");
window.alert(meineSportarten);
console.log(typeof(meineSportarten));
```



Mit `typeof (var)` kann man den Typ einer Variable ermitteln.

Lernhandout 9.2 Arrays sortieren

Referenzcode **JSI092**

Technologien JavaScript

Dateinamen L092_Arrays.odt, L092_Arrays.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... sortieren Arrays alphabetisch mit <code>.sort()</code> ; bzw. der <code>.reverse()</code> ; Methoden.	I	II
... sortieren numerische Arrays aufsteigend bzw. absteigend.	I	III
... nutzen die <code>forEach()</code> Methode um jeden Wert im Array anzusprechen.	I	II
... "durchwandern" mit einer <code>for ... in</code> Schleife ein Array.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Keine

Arrays lassen sich recht einfach sortieren. Dafür gibt es die `.sort()` Methode bzw. die `.reverse()` Methode um die Inhalte umgekehrt aufzulisten.

JS



```
arrayname.sort();
```

Die Arrayelemente werden alphabetisch aufwärts sortiert. Bei dieser Methode werden die Elemente wie ein String behandelt.

```
var studium = ["Philosophie", "Chemie", "Medizin", "Jus"];
studium.sort();
window.alert(studium);
```

JS



```
arrayname.reverse();
```

Dreht die Reihenfolge um, sodaß das letzte Element zum Ersten wird und das Erste zum letzten.

```
var studium = ["Philosophie", "Chemie", "Medizin", "Jus"];
studium.reverse();
window.alert(studium);
```

JS



```
arrayname.sort(function(a, b) {return a - b});
```

Da die `.sort()` Methode nur Strings in eine alphabetische Reihenfolge sortiert und keine Zahlen aufsteigend (z. B. 1, 2, 3, 33, 4, 456, 5 ...) gibt es einen Trick um das Problem zu umgehen. Die Funktion(a, b) übernimmt zwei Werte und subtrahiert sie. Je nachdem wie das Ergebnis ausfällt (Minus, Null, Plus) wird die Zahl auch sortiert.

```
var punkte = [50, 101, 1, 5, 25, 12];
punkte.sort(function(a, b){return a - b});
window.alert(punkte);
```

JS



```
arrayname.forEach(eineFunktion);
```

Startet eine Funktion für jeweils jedes Element im Array. Übergeben wird der Wert und der Index → deshalb sollte die Funktion beide Variablen vereinbaren.

```
var zahlen = [65, 44, 12, 4];
zahlen.forEach(meineFunktion);

function meineFunktion(item, index) {
window.alert("Position: " + index + " ist der Wert " + item);}
```



Selbiges erzielt man auch mit einer `for...in` Schleife. Diese durchwandert jedes Element im Array. `for...in` kann auch auf einen String angewandt werden, dann wandert die Schleife jeden Buchstaben ab.

```
for (var index in zahlen) {
window.alert("Bei: " + index + " steht " + zahlen[index]);}
```

Übungsblatt 9.2 Arrays sortieren

Referenzcode: **JSU092**

Technologien: HTML, CSS, JavaScript

Dateinamen: U092_Arrays.odt, U092_Arrays.pdf

Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... scripten eine Funktion um den Mittelwert aus 100 Zahlen in einem Array zu ermitteln und die Zahlen über den Mittelwert in einem String auszugeben.	I	IV
... erstellen ein Array mit 100 Zufallszahlen.	I	II
... sortieren das Array aufsteigend.	I	II
... ermitteln alle Zahlen die über dem Mittelwert liegen.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... schreiben eine Wettkampf-Protokoll-Webseite.	I	IV
... ermitteln die Zeit in Sekunden zwischen dem Klick auf den Start und dem Klick auf den Stop Button.	I	III
... speichern die Daten in ein Array und geben es in einer Tabelle aus.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erstellen eine Webseite, die nach einer Text(String)eingabe nach einem bestimmten Buchstaben sucht und diesen nach seiner Häufigkeit zählt.	I	IV
... benutzen ein Text-Array.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

■



Übung A: Mittelwert

Fülle ein Array mit 100 Zufallszahlen zwischen 1 und 10. Berechne den Mittelwert und ermittle die Anzahl der Zahlen, die über dem Mittelwert liegen. Sortiere das Array aufsteigend und gib alle Zahlen zusätzlich als String aus.

- HTML** Ausgabefelder und Startbutton
- JS** Funktion mit Array
- css** Ansprechende Gestaltung (Freies Design)



Eine Zufallszahl zwischen 1 und 10 ermittelt man mit:

```
var zufallszahl = Math.round(Math.random() * 9) + 1;
```



Übung B: Wettkampf

Für einen Wettkampf, sollen die Zeiten in Sekunden und der Name des Wettkämpfers protokolliert werden. Der Schnellste soll automatisch ermittelt werden.

- HTML** Eingabefeld für den Namen, Start-Stop Button, Ausgabe in einer Tabelle.
- JS** Funktionen mit Arrays, Berechnungen
- css** Ansprechende Gestaltung (Freies Design)



Die Zeit in Millisekunden ermittelt man mit:

```
var datum = new Date();
var zeit = datum.getTime();
```

Wettkampf

Bitte geben Sie den Namen ein!

Am Besten ist Iris mit 0.711 Sekunden

Peter	4.407 Sekunden
Klaus	1.074 Sekunden
Iris	0.711 Sekunden



Übung C: Buchstaben zählen

In ein Textarea-Feld soll ein beliebig langer Text eingegeben bzw. hineinkopiert werden. Im Anschluss wird ein Buchstabe im Text gesucht und gezählt.

- HTML** Textarea-Feld, Inputfeld, Button
- JS** Funktion mit einem Text-Array
- css** Ansprechende Gestaltung (Freies Design)

Buchstaben zählen

Ein Skript um c zu ermitteln!

Welchen Buchstaben?

Auf der Registerkarte 'Einfügen' enthalten die Kataloge Elemente, die mit dem generellen Layout des Dokuments koordiniert werden sollten. Mithilfe dieser Kataloge können Sie Tabellen, Kopfzeilen, Fußzeilen, Listen, Deckblätter und sonstige Dokumentbausteine einfügen. Wenn Sie Bilder, Tabellen oder Diagramme erstellen, werden diese auch mit dem aktuellen Dokumentlayout koordiniert.

Der Buchstabe m kommt im Text 42 vor!

Lernhandout 10.0 Text Operationen

Referenzcode **JSL100**

Technologien JavaScript

Dateinamen L100_TextOP.odt, L100_TextOP.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... finden in einem Text einen bestimmten String (Textteil).	I	II
... ermitteln in einem Text die letzten Übereinstimmungen mit <code>.lastIndexOf</code>	I	II
... schneiden aus einem Text einen bestimmten String heraus.	I	II
... bewerten den Nutzen von <code>.slice()</code> im Vergleich zu <code>.substr()</code>	KI	III
... ermitteln den UTF-16 Code eines Zeichens.	I	II
... wandeln einen UTF-16 Code zurück in einen String.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Fremdquellen

- In den Beispielen wurden Palindrome verwendet: "O Genie, der Herr ehre dein Ego!" und "Dreh mal am Herd!" kann man auch Rückwärts lesen!

Bisher haben wir Strings mit dem + Operator verbunden. Wir haben auch einen String als Array betrachtet und mit der .length Methode die Anzahl der Zeichen ermittelt.

Positionen von Text in einem String finden

JS



`.indexOf("Textteil", start);`

start ... gibt die Position an, ab welcher die Suche starten soll.
z. B. 4 → also nach dem vierten Zeichen.

Ist der Textteil nicht im String, wird - 1 ausgegeben.

Mit `.lastIndexOf` sucht man nach der letzten Übereinstimmung.

```
var eingabe = "O Genie, der Herr ehre dein Ego!";
var pos = eingabe.indexOf("ehre", 4);
window.alert(pos);
```

Text aus einem String ausschneiden

JS



`.slice(start, ende);`

start ... Startposition im Index z. B. 5 → also das fünfte Zeichen.

ende ... Ende im Index z. B. 12 → bis zum 12 Zeichen. Wird kein Wert für das Ende angegeben, dann gibt die `.slice` Methode den Rest des Strings zurück.

Gibt man eine negative Zahl als start Wert an, dann wird vom Ende des Strings die Startposition ermittelt. z. B. `eingabe.slice(-5);` schneidet die letzten fünf Zeichen ab.

```
var eingabe = "O Genie, der Herr ehre dein Ego!";
var ausgabe = eingabe.slice(23, 31);
window.alert(ausgabe);
```



Die `.substr()` Methode schneidet ebenfalls einen Textteil heraus. Nur wird hier der Startwert und die Anzahl der Zeichen angegeben.

```
var ausgabe = eingabe.substr(23, 8);
```

UTF-16 Code ermitteln

JS



`.charCodeAt(index);`

index ... Die Position des Zeichens. Das erste Zeichen hat die Position 0.

Zurück gegeben wird der UTF-16 Code des Zeichen. z. B. e = 101, D = 68

```
var satz = "Dreh mal am Herd!";
var derUTFcode = satz.charCodeAt(2);
window.alert(derUTFcode);
```



Mit `String.fromCharCode()` dreht man die Sache um.
Man gibt den UTF-16 Code ein und bekommt das Zeichen zurück.

```
var ausgabe = String.fromCharCode(83, 116, 114, 105, 110, 103);
window.alert(ausgabe);
```

Übungsblatt 10.0 Text Operationen

Referenzcode **JSU100**

Technologien HTML, CSS, JavaScript

Dateinamen U100_TextOP.odt, U100_TextOP.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben ein Script, das eine eMail Adresse in seinen local-part und seinen domain-part trennt.	I	IV
... führen eine Zeichen-Positionssuche (das @ Zeichen) aus.	I	II
... führen Textoperationen durch (String herausschneiden).	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen eine UTF-16 Code Tabelle (Zeichen mit Code).	I	IV
... verwenden eine Zählschleife.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... entwickeln einen eigenen Kryptographie-Algorithmus.	IK	IV
... recherchieren im Internet nach schon bestehenden Algorithmen.	IS	III
... nutzen die UTF-16 Codes um einen String zu Ver- und Entschlüsseln.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Der Schlüssel in Übung C kann auch mit einem Salt verglichen werden. Ein Salt ist die kleine besondere Beigabe zu einer Kryptographie.
- Die Cäsar Verschlüsselung ist eine der einfachsten kryptographischen Methoden. Dabei wird einfach z. B. ein a durch ein D ersetzt oder ein k durch ein T. Es firmiert unter den monoalphabetischen Verschlüsselungsverfahren. Schüler_innen können sich hervor tun, wenn sie komplexe Formen der Verschlüsselung entwickeln. Die Kryptographie ist nicht nur seit Turing eine Königsdisziplin der Informatik!



Übung A: E-Mail-Adresse

Eine E-Mail-Adresse besteht aus einem lokalen Teil (local-part, vor dem @-Zeichen) und einem Domänenteil (domain-part, nach dem @-Zeichen). Schreibe ein Script, das den Domänenteil einer eMail-Adresse ermittelt und ausgibt.

z. B. webmail@css4.at → **local-part:** webmail → **domain-part:** css4.at

- HTML **E-Mail-Eingabefeld, Ausgabefeld**
- JS **Funktion mit Textoperationen**
- css **Ansprechende Gestaltung (Freies Design)**



Übung B: UTF-16 Code

Schreibe ein Script, das die ersten 256 Zeichen des UTF-16-Codes ausgibt (Code und Zeichen). Erweitere die Eingabe um einen Zahlenbereich!

- HTML **zwei Eingabefelder, ein Ausgabefeld**
- JS **Funktion mit Zählschleife und CharCode Ausgabe**
- css **Ansprechende Gestaltung (Freies Design)**

UTF-16-Code

Bitte geben Sie einen Zahlenbereich ein.

Von: Bis: Code-Tabelle anzeigen

32 =	65 = A	98 = b	131 =	164 = ¨	197 = Å	230 = æ
33 = !	66 = B	99 = c	132 =	165 = €	198 = Æ	231 = ç
34 = "	67 = C	100 = d	133 =	166 =	199 = Ç	232 = è
35 = #	68 = D	101 = e	134 =	167 = \$	200 = È	233 = é
36 = \$	69 = E	102 = f	135 =	168 = "	201 = É	234 = ê
37 = %	70 = F	103 = g	136 =	169 = ©		
38 = &	71 = G	104 = h	137 =			
	72 = H	105 = i				



Übung C: Kryptographie

Schreibe ein JavaScript, welches eine Eingabe verschlüsselt (also für Menschen unlesbar macht). Der verschlüsselte String soll wieder lesbar gemacht werden. Zusätzlich soll ein Schlüssel Feld eingebaut werden. Der Schlüssel ist notwendig für die Ent- bzw. Verschlüsselung.



Achtung: Wenn man mit dem Webdokument online geht, sollte man bedenken, dass der JavaScript-Code jederzeit über den Quelltext gelesen werden kann und damit auch der kryptographische Algorithmus.

Übung C

Kryptographie

Texteingabe:

Schlüssel (Zahl zwischen 1 und 10):

Lernhandout 10.1 String Methoden

Referenzcode **JSL101**

Technologien JavaScript

Dateinamen L101_StringMethoden.odt, L101_StringMethoden.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... ersetzen Textteile durch einen neuen String.	I	II
... ersetzen alle Textteile durch einen neuen String mit einem /g Flag.	I	II
... heben das Case-Sensitive mit einem /i Flag auf.	I	II
... konvertieren Großbuchstaben in Kleinbuchstaben und umgekehrt.	I	II
... entfernen alle Leerzeichen vor und nach einem String mit <code>.trim()</code> ;	I	II
... teilen einen String über ein Trennzeichen in ein Array mit <code>.split()</code> ;	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die / ... /g bzw. / ... /i Flags sind Konventionen von RegEx. Wer an dieser Stelle gerne die Regular Expressions (RegEx) vertiefen möchte, ist gerne dazu eingeladen.

Textteile ersetzen

JS



```
.replace("old_string", "new_string");
```

Die Methode liefert einen neuen String mit der gewünschten Ersetzung. Es wird ausschließlich die erste Übereinstimmung ersetzt und `.replace` ist Case-Sensitive.

```
var meinText = "Das Gegenteil von umfahren ist umfahren";
var neuerText = meinText.replace("umfahren", "durchfahren");
window.alert(neuerText);
```



Will man alle Übereinstimmungen im String ersetzen, dann muss man den Textteil zwischen `/Text/g` setzen. Ein `/g` Flag ersetzt dann die Anführungszeichen.

```
var meinText = "Das Gegenteil von umfahren ist umfahren";
var neuerText = meinText.replace(/umfahren/g, "anhalten");
window.alert(neuerText);
```



*Ein `/i` Flag hebt Case-Sensitive auf
z. B. `meinText.replace(/UMFAHREN/i, "anhalten")`*

In Groß- und Kleinbuchstaben konvertieren

JS

```
.toUpperCase(); ← In Großbuchstaben
```

JS

```
.toLowerCase(); ← In Kleinbuchstaben
```

```
<p id="gross"></p>
<p id="klein"></p>
<script>
  var derName = window.prompt("Bitte den Namen eingeben");
  var grosserName = derName.toUpperCase();
  var kleinerName = derName.toLowerCase();
  document.getElementById("gross").innerHTML = grosserName;
  document.getElementById("klein").innerHTML = kleinerName;
</script>
```

Leerzeichen (Whitespace) vorne und hinten entfernen

JS

```
.trim();
```

```
var farben = "    Rot und Blau    ";
alert(farben.trim());
```

String in ein Array splitten

JS



```
.split();
```

*Der `.split` Methode wird das Trennzeichen mitgegeben.
z. B. `.split("|")` oder `.split(",")`*

```
var tageszeit = "Morgen, Mittag, Abend, Nacht";
var ausgabe = tageszeit.split(",");
window.alert(ausgabe[2]);
```

Übungsblatt 10.1 String Methoden

Referenzcode **JSU101**

Technologien HTML, CSS, JavaScript

Dateinamen U101_String.odt, U101_String.pdf, Elemente_einf.html

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Webseite, die HTML-eigene Zeichen in HTML Code umwandelt.	I	IV
... scripten eine Funktion zur Zeichenübersetzung.	I	II
Übung B		
... erstellen eine Textsuche mit JavaScript, die einen Suchstring in einem Text hervorhebt.	I	IV
... finden eine Lösung, um mehrere Übereinstimmungen hervorzuheben.	I	III
... arbeiten mit einer Textvorgabe (Elemente_einf.html) und analysieren ein fremdes HTML Dokument.	SI	III
Übung C		
... schreiben ein Script zur Validierung einer IBAN Kontonummer.	IK	IV
... verstehen die Werge der IBAN-Validierung als ein Algorithmus.	IK	III
... führen diverse Textoperationen durch (vier Zeichen nach hinten setzen, Buchstaben in Zahlen umwandeln, usw.).	I	II
... finden eine Lösung um eine Ganzzahldivision in Teilschritte aufzuteilen.	IK	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: HTML Zeichenreferenz

In eine `<textarea>` kann man HTML Code schreiben. Erstelle ein Script, daß die HTML-eigenen Zeichen in HTML Code umwandelt.

z. B. aus `<p id="inhalt">` wird `<p id="inhalt">`

- HTML Zwei `<textarea>` für Ausgabe und Eingabe, Button
- JS Funktion mit Zeichenersetzung

Zeichen	Bezeichnung	Code
"	Anführungszeichen	"
&	kaufmännisches Und	&
<	öffnende spitze Klammer	<
>	schließende spitze Klammer	>
'	einfaches Anführungszeichen	'



Übung B: Suche mit JS

Scripte eine Suche für die Webseite Elemente_einf.html. Das eingegeben Wort (bzw. der String) soll hervorgehoben werden. Sollte das Wort (bzw. der String) öfters vorkommen, dann soll jede Übereinstimmung hervorgehoben werden.

- Öffne die Datei `Elemente_einf.html`



Es gibt mehrere Lösungen. Für eine Lösung mit `.replace` benötigt man `RegExp(suchVariable, 'g')` um die `/g` flags zu setzen.



Übung C: IBAN Validierung

Schreibe ein JavaScript zur Validierung einer IBAN Kontonummer.

Infos: https://de.wikipedia.org/wiki/Internationale_Bankkontonummer

- Teste dein Script mit mind. 3 IBANs aus 3 unterschiedlichen Ländern.

1.	Die ersten zwei Zeichen einer IBAN-Nummer geben an, aus welchem Land sie kommt.	DE68 2105 0170 0012 3456 78
2.	Die ersten vier Zeichen werden an das Ende gesetzt.	2105 0170 0012 3456 78 DE68
3.	Alle Buchstaben werden durch ihre Position im Alphabet + 9 ersetzt. (A = 10, B = 11, C = 12, D = 13, E = 14 ... Z = 35).	210501700012345678131468
4.	Nun wird der Rest berechnet, der sich beim ganzzahligen Teilen der Zahl durch 97 ergibt (Modulo 97). Das Ergebnis muss 1 sein, ansonsten ist die IBAN falsch.	210501700012345678131468 mod 97 = 1



ACHTUNG: Eine Ganzzahldivision (Modulo) wird mit `%` durchgeführt. Nun ist aber JavaScript nicht fähig so große Integer-Zahlen zu berechnen (wegen dem Rundungsfehler, Integer-Zahlen sind nur bis zu 9 Stellen als sicher einzustufen). Überlege dir also eine Funktion, die eine Ganzzahldivision in Teilschritte aufteilt.

Lernhandout 10.2 Datum

Referenzcode: **JS1102**
Technologien: HTML, JavaScript
Dateinamen: L102_Datum.odt, L102_Datum.pdf
Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... vereinbaren ein Datum Objekt mit dem aktuellen (Browser)Datum.	I	II
... nutzen das <time> Element zur besseren Strukturierung.	I	II
... wandeln ein Datum in die deutsche Schreibweise und geben Uhrzeit bzw. das Datum aus.	I	II
... vereinbaren ein bestimmtes Datum mit der Übergabe von Jahr, Monat usw.	I	II
... ermitteln die Millisekunden ab und zu einem bestimmten Zeitpunkt.	I	II
... verstehen die <code>.getTime()</code> Methode auch als Möglichkeit zur Zeitmessung (Stoppuhr).	KI	I

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Der UNIX Timestamp zählt die Sekunden und Millisekunden vom 01. Jänner 1970, 00:00 UTC. Mit diesem Modell lässt sich hervorragend Datumsdifferenzen berechnen.
- Im Jahr 1999 glaubte man an einen totalen (informatischen) Crash wegen dem Millenniumswechsel. Dieser hat sich nicht Bewahrheitet. Bedeutend interessanter ist der 19. Jänner 2038. Ab diesem Datum benötigt der UNIX Timestamp um ein Bit mehr, weil die 32Bit-Grenze überschritten wird. Der Timestamp überschreitet den Wert 2.147.483.647 und weil ein Bit für das Vorzeichen benötigt wird kann es zu einem Zählerüberlauf kommen.

JS



```
var objektname = new Date();
```

Bei der Instanziierung wird dem neuen Objekt das aktuelle Datum des Browsers übergeben.

```
<p>Das aktuelle Datum ist <time id="ausgabe"></time></p>
<script>
  var heute = new Date();
  document.getElementById("ausgabe").innerHTML = heute;
</script>
```



Das `<time>` Element ist nicht unbedingt notwendig für die Ausgabe. Es dient zur besseren Strukturierung.

JS



```
.toLocaleString('de-DE')
```

Die Methode wandelt das Datum in eine deutsche Schreibweise um.

```
.toLocaleTimeString('de-DE'); ← gibt nur die Uhrzeit zurück
.toLocaleDateString('de-DE'); ← gibt nur das Datum zurück
```

```
var heute = new Date();
heute = heute.toLocaleString('de-DE');
window.alert(heute);
```

JS



```
var objekt = new Date(jahr, monat, tag, stunde, minute, sekunde, ms);
```

Ein bestimmtes Datum kann auch bei der Instanziierung definiert werden. Das Jahr, Monat und der Tag sind notwendig, die anderen Angaben sind optional.

```
<p>Friedrich Dürrenmatt starb am <span id="ttag"></span>
  um <span id="tzeit"></span> Uhr an Herzversagen.</p>
<script>
  var gestorben = new Date(1990, 12, 14, 22, 34);
  var todestag = gestorben.toLocaleDateString('de-DE');
  var todeszeit = gestorben.toLocaleTimeString('de-DE');
  document.getElementById("ttag").innerHTML = todestag;
  document.getElementById("tzeit").innerHTML = todeszeit;
</script>
```

JS



```
.getTime()
```

Gibt die Anzahl der Millisekunden vom 01. Jänner 1970, 0:00 UTC bis zum gesetzten Datum zurück.

```
var heute = new Date();
var mondfin = new Date(2022, 4, 16, 4, 12, 42, 0);
mondfin = mondfin.getTime();
heute = heute.getTime();
var zeitbis = (mondfin - heute) / (1000 * 60 * 60 * 24);
window.alert(zeitbis + " Tage bis zur nächsten Mondfinsternis");
```

Lernhandout 11.0 .style

Referenzcode **JSL110**
Technologien HTML, CSS, JavaScript
Dateinamen L110_style.odt, L110_style.pdf
Konvergenz Ab HCL041

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verändern die CSS Eigenschaft eines Elements mit der .style Eigenschaft.	I	II
... kennen die CamelCase Schreibweise.	I	II
... nutzen "Best Practice" Vorschläge um eine Webseite gut zu strukturieren.	IK	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Mit JavaScript lassen sich alle CSS Styles verändern. Dafür wird ein Element ausgewählt, --→ um `.style` erweitert → die CSS Eigenschaft hinzugefügt und schließlich der neue Wert zugewiesen.

JS



```
Element.style.Eigenschaft = 'wert';
```

Das Element wird ausgewählt z. B. mit `document.getElementById("id")` für Eigenschaft wird die Bezeichnung der CSS Eigenschaft angegeben. z. B. `color` oder `margin`

Die Wertzuweisung erfolgt immer mit einfachen oder doppelten Anführungszeichen. z. B. `= "red";` oder `= "3em";`

```
<h1>Lizenzvereinbarung</h1>
<p id="info">... unsere EDV Dienstleistungen unterliegen der
  GNU/GPL und der CommonCreatives.</p>
<button onClick="meineFunktion();" > OK </button>
<script>
  function meineFunktion() {
    document.getElementById("info").style.color = 'gray';
  }
</script>
```



Für CSS Eigenschaften mit einem Bindestrich wird die **CamelCase** Schreibweise angewendet. Aus `font-size` wird `fontSize`. Aus `border-radius` wird `borderRadius`. Aus `background-color` wird `backgroundColor` usw.

```
<input type="text" id="pwd" onKeyUp="hervor();"
  placeholder="Mind. 8 Zeichen">
<script>
  function hervor() {
    if (document.getElementById("pwd").value.length >= 8) {
      document.getElementById("pwd").style.backgroundColor = 'lime';
    }
  }
</script>
```

**Best Practice**

Um eine gut strukturierte Webseite zu scripten sollte man:

- Den HTML Code gut überlegt aufbauen.
- Das Aussehen gänzlich mit CSS gestalten.
- `<noscript>` Elemente verwenden. Mit HTML5 kann das `<noscript>` Element auch im `<head>` eingesetzt werden. Ein `<style>` Element ist dann im `<noscript>` möglich.
- `Element.style` nicht für den Aufbau und das Design, sondern nur für dynamische Veränderungen verwenden!

Lernhandout 11.1 `.style` Objekt

Referenzcode **JSL111**

Technologien HTML, JavaScript

Dateinamen L111_styleObjekt.odt, L111_styleObjekt.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... vereinbaren ein Objekt über einen Selektor.	I	II
... erweitern ein vereinbartes Objekt um die <code>.style</code> Eigenschaft.	I	II
... übergeben ein Objekt mit dem Funktionsaufruf (<code>this</code>).	I	II
... nutzen die Eventattribute <code>onFocus</code> und <code>onBlur</code> .	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Wenn man einen Element-Selektor z. B. `document.getElementById()` öfters benötigt, kann man ihn als Objekt vereinbaren. Damit erspart man sich viel Schreibarbeit und der Code wird übersichtlicher.

JS

```
var Objektname = document.getElementById();
```

```
<h1>Österreich-Quiz</h1>
<p>Wie heißt die Hauptstadt von Österreich?
<input type="text" id="Hauptstadt" ></p>
<p><button onClick="aufloesen();">Auflösen</button></p>

<script>
function aufloesen() {
    var antwort = document.getElementById("Hauptstadt");
    if (antwort.value == "Wien") {
        antwort.style.backgroundColor = 'lime';
        antwort.style.border = '3px solid green';
    } else {
        antwort.style.backgroundColor = 'red';
        antwort.style.color = 'white';
        antwort.style.textDecoration = 'line-through';
    }
}
</script>
```



Das vereinbarte Objekt funktioniert nicht nur mit `.style` sondern auch mit allen anderen JS Befehlen, die auf das Element zugreifen.
Im Beispiel oben: `antwort.value`



Ein im HTML Element gesetztes Eventattribut mit dem Wert "`this`" (z. B. `<input onClick="this" id="dasHier" >`) übergibt das Element als Objekt an eine JavaScript-Funktion (z. B. `function anzahl(eingabe);`)



Im Beispiel wird eine Funktion ausgeführt, sobald der Fokus (`onFocus`) auf das `<input>` Element gesetzt wird. Wenn man das Input-Feld verlässt (`onBlur`) wird eine andere Funktion ausgeführt. Es ändert sich jedes Mal der Rahmen. Die JS Funktionen `hervor()`; `raus()`; übernehmen den Selektor vom HTML Element und greifen auch wieder auf dieses zu.

```
<input type="text" id="ant1" onFocus="hervor(this);"
    onBlur="raus(this);" placeholder="Vorname" ><br>
<input type="text" id="ant2" onFocus="hervor(this);"
    onBlur="raus(this);" placeholder="Nachname" ><br>
<input type="text" id="ant4" onFocus="hervor(this);"
    onBlur="raus(this);" placeholder="eMail" ><br>

<script>
function hervor(eingabefeld) {
    eingabefeld.style.border = '3px solid blue';
}
function raus(dasFeld) {
    dasFeld.style.border = '1px solid gray';
}
</script>
```

Übungsblatt 11.1 .style Objekt

Referenzcode JSU111

Technologien HTML, JavaScript

Dateinamen U111_styleObjekt.odt, U111_styleObjekt.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... ermitteln den aktuellen Wochentag in einem Script und verändern je nach Wochentag die Hintergrundfarbe.	I	IV
... erstellen Style Zuweisungen durch Verzweigungen.	I	II
Übung B		
... schreiben eine Webseite zur Ermittlung der Lesegeschwindigkeit eines Benutzers.	I	IV
... implementieren eine Stoppuhr die über einen Button gesteuert wird.	I	II
... scripten eine Funktion um die Schriftgröße eines Textes zu verändern.	I	II
... schreiben eine Funktion zum Ausgrauen von Absätzen.	I	II
... bewerten die Leseleistung des Benutzers über vorgegebene Bewertungskriterien.	I	II
... reflektieren auf die persönliche Lesegeschwindigkeit.	AKS	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: Heute

Scripte eine Webseite die den heutigen Wochentag ausgibt. Bei Montag, Mittwoch und Freitag färbt sich der Hintergrund dunkelblau. Am Dienstag und Donnerstag wird der Hintergrund gelb. Am Wochenende ist der Hintergrund grün.

- HTML** Ausgabe des Wochentags
- JS** Funktion zur Ermittlung des Wochentages. Style Zuweisungen.



Die `.getDay()` Methode ermittelt den Wochentag zu einem Datum. Ausgegeben wird eine Zahl → Sonntag = 0, Montag = 1, usw.



Übung B: Lesegeschwindigkeit

Erstelle eine Webseite um die Lesegeschwindigkeit zu testen. Suche dafür einen Text mit mind. 570 Wörtern. Verteile über die Absätze `<p>` Tags.

Vor dem Text, ist ein Start-Button. Nach dem Text ein Stop-Button. Bevor der User beginnt zu lesen, drückt er oder sie auf den Start-Button. Wenn der Text zu ende gelesen wurde drückt er oder sie auf den Stop-Button. Ein Script soll dann ermitteln, wie viele Worte pro Minute der_die Leser_in geschafft hat.

Zusätzlich soll der_die Benutzer_in die Schriftgröße des Textes verändern können. Klickt er_sie auf einen Absatz, dann wird dieser Absatz durchgestrichen und ausgegraut.

- Text mit mind. 570 Wörtern.
- HTML** `<p>` Tags mit Event-Attributen, Buttons (Start, Stop, Schriftgröße) Ausgabefeld
- JS** Start-Stop-Funktion (Ermittlung der Worte pro Minute), Schriftgröße-Funktion Funktion zum ausgrauen und durchstreichen von Absätzen.
- css** Ansprechende Gestaltung (Freies Design)
- Teste deine Seite**



Bewertung der Lesegeschwindigkeit

(von Hunziker, H.-W.: Im Auge des Lesers, www.learning-systems.ch)

- weniger als 150 Wörter pro Minute:**
Langsamer Leser.
- 150 bis 200 Wörter pro Minute:**
Durchschnittliche Lesegeschwindigkeit.
- 200 bis 240 Wörter pro Minute:**
Guter Leser
- 240 bis 300 Wörter pro Minute:**
Schneller Leser

Lernhandout 11.2 Selektoren

Referenzcode **JSL112**

Technologien HTML, CSS, JavaScript

Dateinamen L112_Selektoren.odt, L112_Selektoren.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen neben <code>.getElementById()</code> noch die <code>querySelector()</code> Methode um ein HTML Element anzusprechen.	I	II
... nutzen die <code>querySelector()</code> Methode um Elemente, Klassen, IDs und Attribute anzusprechen.	I	II
... selektieren mehrere Elemente mit <code>querySelectorAll()</code> ; und speichern diese in ein Array.	I	II
... iterieren mit einer For-Schleife durch ein Array, welches mit der <code>querySelectorAll()</code> Methode erstellt wurde.	IK	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Bisher haben wir ein Element nur mit `document.getElementById()` angesprochen. Mit der `querySelector()` Methode kann man auch Klassen oder Elemente per Tag-Name ansprechen.

JS

`document.querySelector()`

Die Methode spricht immer das erste Element im HTML Code bzw. DOM an.

- `.querySelector("h1")` → das erste `<h1>` Element
- `.querySelector(".hervor")` → erstes Element mit `class = "hervor"`.
- `.querySelector("#antwort")` → erstes Element mit `id = "antwort"`.
- `.querySelector("[title]")` → erstes Element mit dem Attribut `title`

```
<h1>Knifflig?</h1>
<div class="frage">Was wird beim Trocknen nass?</div>
<p id="antwort"></p>
<button onClick="antworten();" title="Auslösen">Auflösen</button>
<script>
  function antworten() {
    document.querySelector("h1").style.color = "green";
    document.querySelector(".frage").style.fontStyle = "italic";
    document.querySelector("#antwort").innerHTML = "Das Handtuch";
    document.querySelector("[title]").style.display = "none";
  }
</script>
```

JS

`document.querySelectorAll()`

`.querySelectorAll()`; spricht alle Elemente im Dokument an und speichert diese in ein Array. Es können, gleich wie bei `.querySelector()`, Tags, Klassen, ID's, Attribute usw. ausgewählt werden.

```
<p>Was ist bei der Diät erlaubt, und was nicht!</p>
<p class="raus">Konditoreien und Fast-Food-Buden</p>
<p>Meiden Sie <span class="raus">Zucker</span> und
      <span class="raus">Fett</span></p>
<script>
  var r = document.querySelectorAll(".raus");
  r[1].style.color = "red";
</script>
```



Weil die Elemente in ein Array gespeichert wurden, kann man auf alle über eine Zählschleife zugreifen.

```
<script>
  var RausKlasse = document.querySelectorAll(".raus");
  for (var i = 0; i < RausKlasse.length; i++) {
    RausKlasse[i].style.textDecoration = "line-through";
    RausKlasse[i].style.fontVariant = "small-caps";
  }
</script>
```

Lernhandout 11.3 .setAttribute

Referenzcode **JSL113**

Technologien HTML, CSS, JavaScript

Dateinamen L113_setAttribute.odt, L113_setAttribute.pdf

Konvergenz HCL022

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... erweitern ein HTML Element um ein zusätzliches Attribut mit <code>.setAttribute</code> .	I	II
... erweitern ein <code><a></code> Element um ein <code>href</code> Attribut mit einer URL als Wert.	I	II
... fügen einem Element mit <code>.setAttribute</code> eine neue CSS-Klasse zu.	I	II
... ermitteln den Attributwert eines Elements mit <code>.getAttribute()</code> ;	I	II
... löschen ein Attribut mit <code>.removeAttribute()</code> ;	I	II
... ermitteln ob ein Element ein bestimmtes Attribut besitzt.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- In HTML gibt es die Möglichkeit auch eigene Attribute hinzuzufügen, und zwar mit dem Schlüsselwort: `data`.
Z. B.: `data-status="aktiv"` oder `data-meinAttribut="wichtig"`
- Achtung: Im ersten Codebeispiel kommt ein `<style>` Block vor. Dieser muss natürlich in den `<head>` kopiert werden. Es soll gezeigt werden, das eine CSS Klasse mit `.setAttribute` auf ein Element übertragen werden.

Mit der `.setAttribute()` Methode kann man einem HTML Element ein zusätzliches Attribut verleihen. Es benötigt einen Attributnamen und einen Wert! `.getAttribute()` liefert einen Attributwert aus. `.removeAttribute()` entfernt ein Attribut inkl. Wert und mit `.hasAttribute()` kann man prüfen, ob ein Attribut vorhanden ist.

JS



`Element.setAttribute(Attribut, Wert);`

Das Element wird selektiert mit `getElementById()` oder `querySelector()`. Das Attribut wird dann mit Anführungszeichen angegeben z. B. "href". Nach einem Beistrich gibt man den Wert des Attributes an. z. B. "https://www.css4.at".

```
<style>
  .meinLink {text-decoration:none;}
  .meinLink:hover {border-bottom:1px dashed gray;}
</style>
<a id="lern">Lernplattform</a>
<button onClick="nLink();">Link herstellen</button>
<script>
  function nLink() {
    var neuerLink = document.querySelector("#lern");
    neuerLink.setAttribute("href", "https://www.css4.at");
    neuerLink.setAttribute("class", "meinLink");
  }
</script>
```

JS



`Element.getAttribute(Attribut);`

`Element.removeAttribute(Attribut);`

`.getAttribute` gibt den Wert eines selektierten Elements zurück.

`.removeAttribute` löscht ein Attribut aus dem HTML Element.



Das Beispiel übernimmt den Wert des Placeholder Attributes, entfernt das Attribut und fügt den Wert als title wieder ein!

```
<input type="email" id="Mail" onFocus="pruefe(this);"
  placeholder="eMail Adresse eingeben">

<script>
  function pruefe(obj) {
    var titletext = obj.getAttribute("placeholder");
    obj.removeAttribute("placeholder");
    obj.setAttribute("title", titletext);}
</script>
```

JS



`Element.hasAttribute(Attribut);`

Ermittelt, ob es ein bestimmtes Attribut im Element gibt. Zurückgegeben wird `true` oder `false` (also Boolean).

```
var meinElement = document.getElementById("Mail");
console.log(meinElement.hasAttribute("placeholder"));
```


Übungsblatt 11.3 .setAttribute

Referenzcode: **JSU113**

Technologien: HTML, CSS, JavaScript

Dateinamen: U113_setAttribute.odt, U113_setAttribute.pdf,
Registrierung.html

Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben ein Registrierungsformular.	I	IV
... implementieren eine Funktion zur Veränderung der Schriftgröße via Button-Klick.	I	II
Übung B		
... erweitern die Übung A um einen Button, der die Webseite ins Englische übersetzt.	I	II
Übung C		
... erweitern die Übung B und fügen ein zusätzliches Attribut zu allen Eingabefeldern hinzu.	I	II
Übung D		
... erweitern die Übung C um eine Ausblende-Funktion für Eingabefelder mit einem bestimmten Attribut (required).	I	II
Übung E		
... öffnen alle Lösungen zu dieser Übung und erweitern sie um weitere Eingabefelder und sowie zusätzliche Funktionen.	I	II
... verbessern die Webseite selbstständig.	I	IV
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Alle Übungen beziehen sich auf ein vorgefertigtes Webdokument Registrierung.html. Das Formular kann aber auch laut der Bebilderung auf dem Handout neu mit HTML reproduziert werden!

**Übung A: Schriftgröße ändern**

Öffne das Webdokument *Registrierung.html*. Mit den + und – Buttons soll sich die Schriftgröße von den Eingabefeldern und der Beschriftung ändern.

- Bei Klick auf Plus: Schriftgröße
+1pt für Beschriftungen und
+2pt für die Eingabefelder.
- Bei Klick auf Minus: Schriftgröße
-1pt für Beschriftungen und
-2pt für die Eingabefelder.

**Übung B: Englisch**

Öffne das Webdokument *Registrierung.html*. Bei einem Klick auf den Button "Ins Englische übersetzen" soll die englische Beschriftung angezeigt werden.

**Übung C: Englisch II**

Öffne deine Lösung von Übung B (*Registrierung.html*). Zu jedem Eingabefeld soll nach einem Klick auf den Button "Ins Englische übersetzen" ein `title` Attribut mit dem Wert "Only english allowed" hinzugefügt werden.

TIPP: `.setAttribute`

**Übung D: Pflichtfelder anzeigen**

Öffne das Webdokument *Registrierung.html*. Bei einem Klick auf den Button "Nur Pflichtfelder anzeigen" werden alle Eingabefelder ausgeblendet, die nicht als Pflichtfeld (required) gekennzeichnet sind.

**Übung E: Erweiterungen**

Öffne deine Lösung von Übung D (*Registrierung.html*) und ergänze es um die Lösungen von Übung A, B und C.

- HTML** Füge neue Eingabefelder für das Geburtsdatum, Geburtsort und Familienstand hinzu!
- JS** Per Klick soll zwischen Englisch und Deutsch gewechselt werden. Title Attribute sind auf Englisch und Deutsch verfügbar. Alle Felder sollen per Mausklick wieder einblendbar sein!

Lernhandout 12.0 Nodes im DOM

Referenzcode **JSI120**

Technologien HTML, JavaScript

Dateinamen L120_Nodes.odt, L120_Nodes.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen das DOM (Document Object Model) und damit verbunden die Rolle des W3C.	I	I
... kennen die Bestandteile der Baumstruktur des DOM.	I	I
... kennen die Knotentypen "Elementknoten", "Attributknoten" und "Textknoten".	I	I
... unterscheiden zwischen Eltern-, Kind- und Geschwisterelementen.	I	II
... erkennen über eine Graphik die Verbindung zwischen den einzelnen Knoten.	I	III
... verstehen die "Verwandtschaftsverhältnisse" im DOM.	I	I
... können vom <code><body></code> Tag aus alle "Verwandtschaftsverhältnisse" bestimmen.		

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Nachdem ein Webdokument geladen wurde, erstellt der Browser das DOM – Document Object Model. Es ist die Schnittstelle zwischen HTML und JavaScript.

Der DOM Standard wird vom W3C (WWW Consortium) herausgegeben – die Browserhersteller (Mozilla, Microsoft usw.) halten sich in der Regel an diese Standards.

Das DOM wird oft als Baumstruktur dargestellt! Die einzelnen Bestandteile in dieser Baumstruktur werden auch als Knoten (Nodes) bezeichnet. Zu den drei wichtigsten Knotentypen gehören: **Elementknoten**, **Attributknoten** und **Textknoten**.

Wir unterscheiden:

Elternelemente

parentNode

Kindelemente

childNodes

firstChild

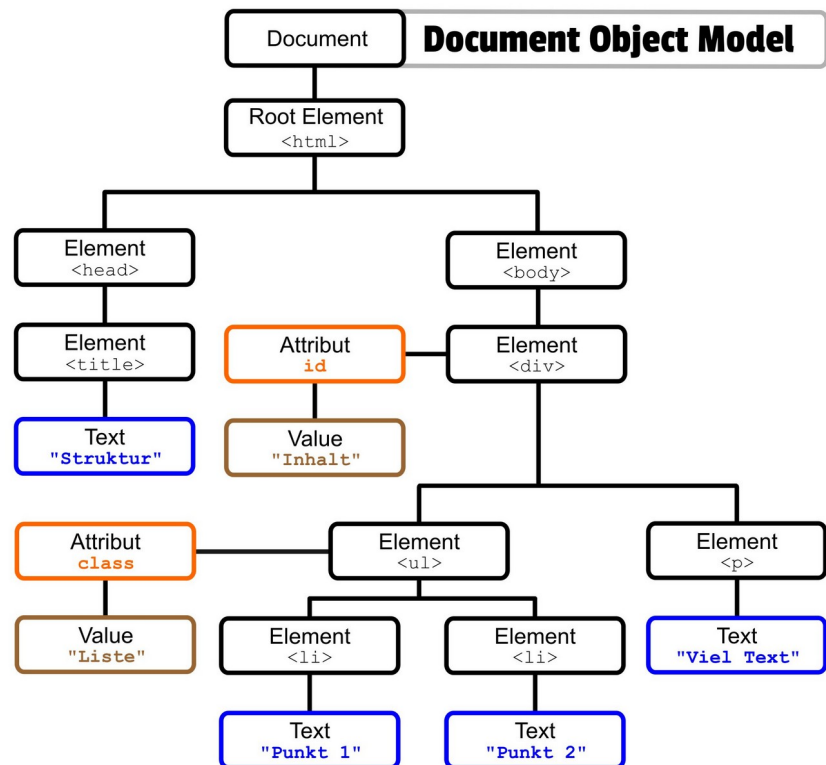
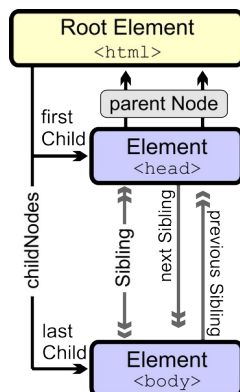
lastChild

Geschwister

sibling

nextSibling

previousSibling



"Verwandtschaftsverhältnisse" wenn der `<body>` Tag als Parent betrachtet wird!

```
<html>
  <head>
    <title>Struktur</title>
  </head>
  <body>
    <div id="Inhalt">
      <ul class="Liste">
        <li>Punkt 1</li>
        <li>Punkt 2</li>
      </ul>
      <p>Viel Text</p>
    </div>
  </body>
</html>
```

- ← Parent (Elternelement)
- ← First Child (Kindelement) von `<body>`
- ← Child von `<div>`, Parent von ``, Sibling von `<p>`
- ← Child von ``
- ← Child von ``, next Sibling von `Punkt 1`
- ← Second Child von `<div>`, Sibling (Geschwister) von `` und last Child von `<div>`

Übungsblatt 12.0 Nodes im DOM

Referenzcode	JSU120
Technologien	HTML, JavaScript
Dateinamen	U120_Nodes.odt, U120_Nodes.pdf
Konvergenz	HCL083

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen eine Graphik eines HTML Code-Ausschnitt.	I	III
... nutzen Desktop-Publishing-Software (z. B. Visio, CorelDraw, odgl).	I	II
Übung B		
... erstellen eine Baumstruktur von einem vorgegebenen HTML Code.	I	III
... erweitern die Baumstruktur-Graphik um die Knotenpunkte (Element, Attribut und Text).	I	II
... nutzen Desktop-Publishing-Software (z. B. Visio, CorelDraw, odgl).	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Ein weiterer wichtiger Knoten wäre der Kommentarknoten.
- Achtung: Whitespace (Leerzeichen, Steuerzeichen usw.) werden als Textknoten interpretiert. So kann z. B. passieren, dass ein `` das Leerzeichen zum nächsten `` als next Sibling interpretiert und nicht wie gewünscht das nächste List-Item-Element. Bei den Übungen sollte diese Besonderheit vernachlässigt werden.
- Zu empfehlen ist die vektorbasierende DP-Software (DP steht für Desktop Publishing) Inkscape. Sie ist frei und quelloffen.



Übung A: Verwandtschaft

Erstelle eine Graphik die alle Verwandtschaftsverhältnisse der Elemente vom HTML Code unten beschreibt.

- Nutze dafür eine DP Anwendung deiner Wahl**

(z. B. LibreOffice, Inkscape, CorelDraw, Illustrator odgl.)

```
<div id="hauptinhalt">
  <h1 class="ueber1">Was bedeutet DP?</h1>
  <p>DP ist die Abkürzung für Desktop Publishing</p>
</div>
```



Übung B: Baumstruktur

Erstelle eine Baumstruktur vom HTML Code unten. Es sollen alle drei wichtigen Knotenpunkte (Elemente, Attribute und Texte) ersichtlich sein!

- Trage bei den Elementen ihre Verwandtschaft ein!

- Nutze dafür eine DP Anwendung deiner Wahl**

(z. B. LibreOffice, Inkscape, CorelDraw, Illustrator odgl.)

```
<html lang="de">           ← _____
<head>                   ← _____
<title>Printmedien</title> ← _____
</head>
<body>                   ← _____
<div id="Inhalt">        ← _____
<h1>Zeitungen</h1>      ← _____
<ol id="Liste">          ← _____
<li>Standard</li>       ← _____
<li>Die Presse</li>     ← _____
</ol>
</div>
</body>
</html>
```

Lernhandout 12.1 Nodes einfügen

Referenzcode: **JS1121**
Technologien: HTML, JavaScript
Dateinamen: L121_Nodes.odt, L121_Nodes.pdf
Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... selektieren Elternelemente um einen neuen Knoten hinzuzufügen.	I	II
... erzeugen ein neues Element mit <code>.createElement()</code> ;	I	II
... erzeugen einen Text mit <code>createTextNode()</code> ;	I	II
... verbinden mit <code>.appendChild()</code> ; den neuen Text mit dem neuen Element.	I	II
... fügen den neuen Knoten in das selektierte Elternelement mit <code>.insertBefore()</code> ; ein.	I	II
... erweitern einen Knoten um ein Attribut mit einem Wert für das Attribut.	I	II
... hängen einen neuen Attributknoten an das selektierte Elternelement.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Im Beispiel wurden Fussballclubs aus Österreich angeführt. Sollte der_die Vortragende selbst ein passionierter Fussballfan sein, dann sollte er_sie unbedingt die Reihenfolge ändern und mit Kommentaren menschliche Authentizität in den pädagogischen Alltag bringen. Zu beachten ist, dass der hinzugefügte Knotenpunkt dann die Tabelle anführen wird, also: Sturm Graz wird die Nr. Eins in der Tabelle und bekommt das Titleattribut: "Die beste Fussballmannschaft".

Mit den Nodes (Knoten) können HTML Elemente hinzugefügt, ersetzt und gelöscht werden. Dafür wird das neue Element erzeugt und das Elternelement selektiert. In das Elternelement wird dann das neue HTML Element hinzugefügt.

JS	<code>document.createElement();</code>
1	Ein neues HTML Element wird erzeugt! <code>var neuesElement = document.createElement("li");</code>
2	Ein neuer Text wird definiert. <code>var neuerText = document.createTextNode("Das hier ist neu");</code>
3	<code>.appendChild();</code> Der Text wird an das neue Element angehängt (hinzugefügt). <code>neuesElement.appendChild(neuerText);</code>
4	<code>.insertBefore();</code> 1. Das Elternelement wird ausgewählt 2. Jenes Kindelement, vor dem das neue Element eingefügt werden soll, wird ausgewählt. 3. Mit <code>Elternelement.insertBefore(neuesElement, Kindelement)</code> wird das neue Element hinzugefügt.



Beispiel: Ein neuer Eintrag `` in einer Nummerierung ``

```
<ol id="meineListe">
  <li id="Eintrag1">Redbull Salzburg</li>
  <li id="Eintrag2">Rapid Wien</li>
  <li id="Eintrag3">LASK</li>
</ol>

<script>
  var neuesElement = document.createElement("li");
  var neuerText = document.createTextNode("Sturm Graz");
  neuesElement.appendChild(neuerText);

  var elternElement = document.getElementById("meineListe");
  var kindElement = document.getElementById("Eintrag1");
  elternElement.insertBefore(neuesElement, kindElement);
</script>
```

Ein Attribut hinzufügen (z. B. title, id, class usw.)

JS 1:	<code>document.createAttribute();</code>	← Attribut wird erstellt
JS 2:	<code>.value = "";</code>	← Wert des Attribut wird definiert
JS 3:	<code>.setAttributeNode();</code>	← Attribut wird zugewiesen

```
var neuesAttribut = document.createAttribute("title");
neuesAttribut.value = "Die beste Fussballmannschaft";
neuesElement.setAttributeNode(neuesAttribut);
```


Lernhandout 12.2 Nodes ersetzen

Referenzcode **JSI122**

Technologien HTML, CSS, JavaScript

Dateinamen L122_Nodes.odt, L122_Nodes.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... ersetzen ein Element durch ein anderes Element.	I	II
... verwenden die Methode <code>.replaceChild()</code> ;	I	II
... schreiben ein Script, welches eine Überschrift durch ein <code><p></code> Tag ersetzt.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Um ein HTML Element zu ersetzen, wird zuerst das neue Element erstellt (`.createElement()`). Dem neuen Element wird ein Text zugewiesen (`.createTextNode()`). Zusätzlich kann man dem neuen Element auch ein Attribut zuweisen (`.createAttribute()`).

Das Elternelement vom HTML Tag, welcher ersetzt werden soll, wird selektiert und dimensioniert. Ebenso das HTML Element selbst. Im Anschluss wird mit `.replaceChild()` das HTML Element ersetzt.

JS

`.replaceChild()`

Wird als Methode an das Elternelement angehängt. Übergeben wird dann, das neue Element und das alte Element.

```
elternElement.replaceChild(neuesElement, altesElement);
```



Beispiel: Ersetzt die Überschrift mit der `id="ue1"` durch einen `<p>` Tag mit der Klasse `class="gelesen"`.

1

```
<style>
  .gelesen {color: red;}
</style>
```

2

```
<div id="inhalte">
  <h1 id="ue1">Datenschutz</h1>
  <h1 id="ue2">Allgemeine Bestimmungen</h1>
  <button onClick="meineFunktion();">Gelesen</button>
</div>
```

3

```
<script>
  function meineFunktion() {
    var neuesElement = document.createElement("p");
    var neuerText = document.createTextNode("Gelesen");
    neuesElement.appendChild(neuerText);
```

4

```
    var neuesAtt = document.createAttribute("class");
    neuesAtt.value = "gelesen";
    neuesElement.setAttributeNode(neuesAtt);
```

5

```
    var elternElement = document.getElementById("inhalte");
    var altesElement = document.getElementById("ue1");
    elternElement.replaceChild(neuesElement, altesElement);
  }
</script>
```

**Erklärung zum Beispiel!**

1. CSS Eigenschaften im `<head>`
2. HTML mit der `<h1 id="ue1">` Überschrift
3. Neues Element `<p>` mit dem Text "Gelesen".
4. Neues Attribut für das Element (die CSS Klasse `.gelesen` wird zugewiesen).
5. Eltern- und Kindelement werden selektiert und ersetzt.

Lernhandout 12.3 Nodes entfernen

Referenzcode **JSL123**

Technologien HTML, JavaScript

Dateinamen L123_Nodes.odt, L123_Nodes.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... löschen Konten aus dem DOM.	I	II
... verwenden zum Löschen eines Knotens die <code>.removeChild()</code> Methode.	I	II
... unterscheiden und erkennen einen gelöschten Knoten im Vergleich zu einem ausgeblendeten Objekt mit der CSS Eigenschaft: <code>display: none;</code>	I	III
... hinterfragen, warum man zum Löschen das Elternelement selektieren muss.	I	III
... selektieren das Elternelement mit <code>.parentNode</code> .	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Um einen Knoten (bzw. ein HTML Element) zu entfernen, muss man das Elternelement und das zu löschende Element kennen und selektieren.

JS



.removeChild() ← Löscht ein HTML Element.

Im Gegensatz zu der CSS Anweisung `display: none;` wird das HTML Dokument gänzlich gelöscht und nicht nur ausgeblendet.

```
<ol id="einkaufsListe">
  <li id="e1">Milch</li>
  <li id="e2">Schokolade</li>
  <li id="e3">Äpfel</li>
</ol>
<button onClick="Entfernen();">Löschen</button>
<script>
  function Entfernen () {
    var elternElement = document.getElementById("einkaufsListe");
    var kindElement = document.getElementById("e2");
    elternElement.removeChild(kindElement);
  }
</script>
```



Warum braucht man das Elternelement? Diese Frage ist berechtigt, da das zu löschende Element sowieso direkt selektiert wird. Die Antwort ist: Der DOM benötigt es!



Die Lösung jedoch: mit `.parentNode` wird automatisch das Elternelement selektiert. Das Script zum Beispiel oben, sieht dann wie folgt aus:

```
<script>
  function Entfernen() {
    var elternElement = document.getElementById("e2").parentNode;
    var kindElement = document.getElementById("e2");
    elternElement.removeChild(kindElement);
  }
</script>
```

JS



.parentNode ← Selektiert ein Elternelement!

Mit `.parentNode` wird das Elternelement selektiert. Im Beispiel ist das `<div>` das Elternelement von `<button>` - ausgeblendet wird damit das `<div>` und damit auch das `<h1>` und das `<p>` Element.

```
<div>
  <h1>Informationen</h1>
  <p>Vieles zu lesen und noch mehr</p>
  <button onClick="ausblenden(this);">Ausblenden</button>
</div>
<script>
  function ausblenden(kindObjekt) {
    kindObjekt.parentNode.style.display = 'none';
  }
</script>
```

Lernhandout 12.4 Nodelist

Referenzcode **JSL124**
Technologien HTML, JavaScript
Dateinamen L124_Nodes.odt, L124_Nodes.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... erstellen eine Nodelist und greifen mit <code>.childNodes[]</code> und <code>.childNodes.item()</code> auf diese zu.	I	II
... erkennen Whitespace und Kommentare als Knotenpunkte.	KI	I
... vereinbaren ein Array mit den Kindelementen als Nodelist.	I	III
... fügen mit <code>.appendChild()</code> einen neuen Knoten hinzu.	I	II
... selektieren den ersten Kindknoten mit <code>.firstChild</code>	I	II
... selektieren den letzten Kindknoten mit <code>.lastChild</code>	I	II
... selektieren den unmittelbar nächsten Knoten mit <code>.nextSibling</code>	I	II
... selektieren den unmittelbar vorherigen Knoten mit <code>.previousSibling</code> .	I	II
... selektieren das unmittelbar nächste Element mit <code>.nextElementSibling</code> .	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Für `.firstChild` gäbe es folgendes Beispiel:

Beispiel:

```
var dListe = document.getElementById("meineListe");  
var ersterPunkt = dListe.firstChild;  
console.log(ersterPunkt);
```

Auf die einzelnen Kindknoten kann man wie in einer Liste mit `.childNodes[]` bzw. mit `.childNodes.item()` zugreifen. `.childNodes.length` gibt die Anzahl der Knoten eines Elternelements zurück. Diese Befehle geben aber auch den Whitespace (Leerzeichen, Zeilenschaltungen usw.) und Kommentare als Knotenpunkt zurück. Alternativ kann ein Array mit den Kindelementen als Nodelist vereinbart werden.

JS

`.appendChild()`

Mit `.appendChild()` kann man auch einen neuen Knoten (neues Element) am Ende innerhalb eines Elternelements hinzufügen bzw. anhängen!

```
<ul id="meineListe">
  <li>Physik</li>
  <li>Chemie</li>
  <li>Mathe</li>
</ul>
<input id="neuesFach" type="text" placeholder="Neues Fach">
<button onClick="einesMehr();">Fach hinzufügen</button>
<script>
  function einesMehr() {
    var neuerEintrag = document.getElementById("neuesFach").value;
    var neuesElement = document.createElement("li");
    var neuerText = document.createTextNode(neuerEintrag);
    neuesElement.appendChild(neuerText);
    var elternElement = document.getElementById("meineListe");
    elternElement.appendChild(neuesElement); // ← Fügt neues Element hinzu!
  }
</script>
```




Erweiterung des Beispiel: Eine Nodelist in Form eines Arrays wird erstellt. Der `.querySelectorAll()` durchsucht nur das Elternelement.

```
<button onClick="listeAnzeigen();">Nr. 2 anzeigen</button>
```

```
function listeAnzeigen() {
  var dListe = document.getElementById("meineListe");
  var mListe = dListe.querySelectorAll("li");
  alert("Anzahl " + mListe.length);
  alert("Nr. 2 ist " + mListe[1].innerHTML); }

```

Weitere Knoteneigenschaften

JS	<code>.firstChild</code>	← Selektiert den ersten Kindknoten.
	<code>.lastChild</code>	← Selektiert den letzten Kindknoten.
	<code>.previousSibling</code>	← Selektiert den unmittelbar vorherigen Knoten.
	<code>.nextSibling</code>	← Selektiert den unmittelbar nächsten Knoten.
	<i>Diese Eigenschaften durchsuchen nur die Struktur des Elternelements. Es werden keine Kindes-Kinder gesucht! Beachte dass der Whitespace auch selektiert wird! Ausnahme <code>.nextElementSibling</code></i>	
JS	<code>.nextElementSibling</code>	← Selektiert das unmittelbar nächste Element.

Übungsblatt 12.4 Nodelist

Referenzcode: **JSU124**
 Technologien: HTML, CSS, JavaScript
 Dateinamen: U124_Nodes.odt, U124_Nodes.pdf
 Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen eine "online Einkaufsliste".	I	IV
... verwenden dafür Nodes (Knotenoperationen).	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen eine Personenliste die durchnummeriert und zugleich durch einen Klick auf ein + Symbol erweiterbar ist.	I	IV
... verwenden dafür Nodes (Knotenoperationen).	I	II
... orientieren sich an der Text und Bildvorlage.	SK	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erweitern die Lösung von Übung B um eine Funktion, die Einträge wieder aus der Liste löscht.	I	IV
... transponieren die Eingabeergebnisse in eine HTML Tabelle.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: Einkaufsliste

Erstelle eine online Einkaufsliste. Die Produkte werden in ein Eingabefeld eingegeben und in einer Liste aufgeschrieben.

- HTML** Überschrift und kurzer erklärender Text, Eingabefeld, 2 Buttons, Ungeordnete Liste
- JS** Funktion mit Nodes
- css** Ansprechende Gestaltung

Einkaufsliste

Bitte geben Sie das Produkt ein!

- Brot
- Milch
- Salz



Übung B: Personenliste Plus

Gestalte eine Webseite mit Eingabefeldern für eine Personenliste. Die Einträge sollen durchnummeriert sein. Es sollen zwei Eingabefelder für Vor- und Nachnamen verfügbar sein. Am Ende von jedem neuen Eintrag soll ein + Symbol sein, das einen neuen Eintrag ermöglicht. Es darf nur ein + Symbol geben und dieses muss auf der gleichen Höhe sein, wie der letzte Eintrag.

- HTML** **Überschrift, Text, Eingabefelder, Button, Nummerierung**
- JS** Mind. eine Knotenoperation
- css** Ansprechende Gestaltung (Freies Design)

Personenliste

Bitte geben Sie Vor- und Nachnamen ein!
Möchten Sie eine weitere Person eintragen? Dann klicken Sie bitte auf das Plus-Zeichen

1.	<input type="text" value="Peter"/>	<input type="text" value="Müller"/>	
2.	<input type="text" value="Claudia"/>	<input type="text" value="Weisner"/>	
3.	<input type="text" value="Vorname"/>	<input type="text" value="Nachname"/>	<input type="button" value="+"/>



Übung C: Personenliste Minus

Öffne deine Lösung von Übung B. Bei jedem Eintrag soll ein Minus Symbol sein. Klickt man darauf, wird der Eintrag aus der Personenliste gelöscht. Zusätzlich soll noch ein Fertig Button hinzugefügt werden. Klickt man darauf, dann werden alle Einträge in einer HTML Tabelle zusammengefasst!

Ausgabe in einer Tabelle

1.	<input type="text" value="Karl"/>	<input type="text" value="Königsberger"/>	<input type="button" value="-"/>
2.	<input type="text" value="Sabine"/>	<input type="text" value="Rainer"/>	<input type="button" value="-"/>
3.	<input type="text" value="Franz"/>	<input type="text" value="Schuhmacher"/>	<input type="button" value="+"/>

Minus Symbole um einen Eintrag zu löschen

Nr.	Vorname	Nachname
1	Karl	Königsberger
2	Sabine	Rainer
3	Franz	Schuhmacher

Lernhandout 13.1 document

Referenzcode **JSL131**

Technologien HTML, JavaScript

Dateinamen L131_document.odt, L131_document.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... repassieren das bisherig-gelernte über das <code>document</code> Objekt.	IK	I
... ermitteln die URL eines Webdokuments.	I	II
... ermitteln die Domain eines Webdokuments.	I	II
... ermitteln den Zeichensatz eines Webdokuments.	I	II
... ermitteln den Titel eines Webdokuments.	I	II
... ermitteln ob ein Dokument fokussiert ist.	I	II
... ermitteln ob ein Webdokument vollständig geladen wurde.	I	II
... verändern die Editierbarkeit eines Dokuments mit <code>.designMode</code> .	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Bisher haben wir das `document` Objekt zum Selektieren von Elementen im DOM verwendet. Dafür verwendeten wir `.getElement...` und `.querySelector...` Wir nutzen das `document` Objekt auch für Nodes. Es kann aber mehr. Hier eine kleine Auswahl von Eigenschaften und Methoden des `document` Objekts ...

JS

<code>document.URL;</code>	← Gibt die Internetadresse URL des Dokuments zurück.
<code>document.domain;</code>	← Gibt die Domain des Dokuments zurück.
<code>document.characterSet;</code>	← Gibt den Zeichensatz des Dokuments zurück.
<code>document.title;</code>	← Gibt den Titel des Dokuments zurück.



Ein neuer Titel wird mit `document.title = "Neuer Titel";` gesetzt!

```
<!doctype html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <title>Mein Dokument</title>
  </head>

  <body>
    <div class="inhalt">
      <h1>Dokumenteigenschaften</h1>
      <p id="ausgabe"></p>
    </div>
    <script>
      var ausgabe = "Hier die Eigenschaften: <br>";
      ausgabe = ausgabe + "<br>URL: " + document.URL;
      ausgabe = ausgabe + "<br>Domain: " + document.domain;
      ausgabe = ausgabe + "<br>Zeichensatz: " + document.characterSet;
      ausgabe = ausgabe + "<br>Titel: " + document.title;
      document.getElementById("ausgabe").innerHTML = ausgabe;
      document.title = "Dokument geladen!";
    </script>

  </body>
</html>
```

JS

<code>document.hasFocus();</code>	← Ermittelt ob das Dokument fokussiert ist (true/false).
<code>document.readyState;</code>	← Ermittelt ob ein Dokument vollständig geladen wurde!

```
<script>
  alert(document.readyState);
  function meineFunktion() {alert(document.readyState);}
</script>
<button onClick="meineFunktion();">Fertig</button>
```

JS

`document.designMode = "on";` ← Macht das Dokument editierbar.



Mit `document.designMode = "on";` kann man das gesamte Webdokument im Browser editieren (zum Beispiel einen Text hinzufügen oder löschen)!
Mit `= "off"` wird das Editieren deaktiviert.

Lernhandout 13.2 Timer

Referenzcode **JSL132**
Technologien JavaScript
Dateinamen L132_Timer.odt, L132_Timer.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen die Wirkweise eines Timers (Ticker, Intervall).	K	I
... nutzen <code>setInterval()</code> um einen Timer zu implementieren.	I	II
... kennen die Syntax von <code>setInterval()</code> und verstehen das Ticken in Millisekunden.	KI	II
... schreiben ein Script, in dem die aktuelle Uhrzeit via Timer abläuft.	I	II
... stoppen einen Timer mit <code>clearInterval()</code> ;	I	II
... nutzen <code>setTimeout()</code> um eine Funktion nach einer bestimmten Anzahl von Millisekunden auszuführen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Die `setInterval()` Methode ruft eine Funktion in bestimmten Intervallen auf. Jedes Intervall wird in Millisekunden angegeben. Lässt man den Timer alle 1000 ms ticken, dann wird zu jeder Sekunde die Funktion aufgerufen. 1000 ms = 1 Sekunde!

JS

**setInterval(Funktion, Millisekunden)**

Der Timer kann als Objekt dimensioniert werden. Mit der Vereinbarung wird der Timer auch gestartet. Will man der Funktion Werte übergeben, dann werden die Parameter nach den Millisekunden hinzugefügt.

Z. B.: `setInterval(Funktion, Millisekunden, param1, param2, ...)` ;

```
<script>
  var meinTimer = setInterval(Uhrzeit, 1000);
  function Uhrzeit() {
    var JetztZeit = new Date();
    JetztZeit = JetztZeit.toLocaleTimeString('de-DE');
    document.getElementById("ausgabe").innerHTML = JetztZeit;
  }
</script>
<p id="ausgabe"></p>
```

JS

**clearInterval()** ;

Stoppt den Timer. Der Methode wird das Timer Objekt übergeben.

Z. B. `clearInterval(meinTimer)` ;

```
<script>
  var lauf = 10;
  var meinTimer = setInterval(meinCountdown, 500);

  function meinCountdown() {
    lauf = lauf - 1;
    document.getElementById("ausgabe").innerHTML = lauf;
    if (lauf == 0) {clearInterval(meinTimer);}
  }
</script>
<p id="ausgabe"></p>
```



Im Beispiel wird von 10 bis 0 hinunter gezählt. Bei 0 wird der Timer gestoppt. Die Variable `lauf` und das Timer-Objekt `meinTimer` sind global und stehen damit jeder Funktion zur Verfügung.

JS

**setTimeout(Funktion, Millisekunden)**

Die Methode ruft eine Funktion nach einer bestimmten Anzahl von Millisekunden auf. Die Funktion wird nur einmal aufgerufen.

Mit `clearTimeout()` kann man das Aufrufen verhindern.

```
<script>
  var meinTimer = setTimeout(meineFunktion, 5000);
  function meineFunktion() {
    alert("Ich habe 5 Sekunden gewartet!");
  }
</script>
```

Übungsblatt 13.2 Timer

Referenzcode: **JSU132**

Technologien: HTML, CSS, JavaScript

Dateinamen: U132_Timer.odt, U132_Timer.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben ein Script um die Ladezeit eines Webdokuments zu ermitteln.	I	IV
... finden im Internet nach einem großen Bild und binden es in ein Webdokument ein.	I	II
... schreiben eine Timer-Funktion mit einer readyState Abfrage.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
... erweitern das Webdokument um weiteren Content und analysieren die Ladezeit um ein "schlankes" Internet zu verstehen.	I	III
... schreiben eine Timeout Funktion als Abbruchbedingung.	I	II
Übung B		
... scripten eine Countdown Funktion (Herunterzählen in Sekunden).	I	IV
... nutzen eine Timer-Funktion für den Countdown.	I	II
... erweitern den Countdown um eine graphische Progress-Bar.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... schreiben ein Script, um einen Text mit Schreibmaschinen-Effekt darzustellen.	I	IV
... nutzen eine Timer-Funktion mit diversen String-Text-Operationen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Übung A: Urheberrechte für das Bild beachten!
- Übung B: Beim Countdown muss das Problem der "60 Sekunden ist eine Minute" gelöst werden und Sekunden und Minuten unter 10 sollten mit einer Null voran dargestellt werden.



Übung A: Ladezeit

In einer Webseite soll ein wirklich großes Bild (mind. > 40 MiB) aus dem Internet dargestellt werden. Ermittle die Ladezeit des gesamten Dokuments.

- Suche im Internet nach einem richtig großen Bild.
- HTML** Binde das Bild mit einem `` Tag ins Dokument ein.
- JS** **Timer und readyState Abfrage, getTime()**
- css** Ansprechende Gestaltung (Freies Design)
- Teste deine Seite und erweitere die Seite mit Content um die Ladezeit zu erhöhen!
- Setze ein Timeout, welches nach 20 Sekunden die Meldung "Zeit abgelaufen" ausgibt.**

Ladezeit: 6.864 Sekunden



Übung B: Countdown

Schreibe eine Webseite mit einer Countdown-Funktion. Über Eingabefelder kann man Minuten und Sekunden eingeben. Das Skript soll dann die Gesamtanzahl der Sekunden herunterzählen.

- HTML** **2 Eingabefelder, Button, Ausgabefeld**
- JS** **Timer Funktionen**
- css** Ansprechende Gestaltung (Freies Design)
- Erweitere das Beispiel um eine optische Rückmeldung.**
(z. B. eine Progress-Bar)




Das `<progress>` Element kann für die optische Rückmeldung verwendet werden. Es kennt die Attribute `value` und `max`.

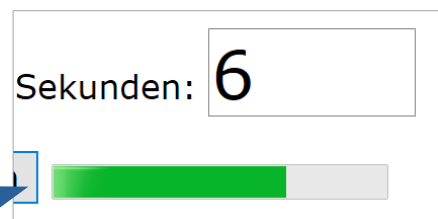
Übung B

Countdown 1 min 15 sek

Bitte geben Sie Minuten und Sekunden ein!

Minuten: Sekunden:





Übung C: Schreibmaschinen-Effekt

Ein Text soll Buchstabe für Buchstabe (so als würde man auf einer Schreibmaschine schreiben) auf dem Bildschirm ausgegeben werden!

- HTML** **Ausgabe**
- JS** **Timer-Funktion**

Lernhandout 14.0 Window Objekt

Referenzcode **JSL140**

Technologien JavaScript

Dateinamen L140_Window.odt, L140_Window.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... verstehen das Window Objekt als offenes Fenster.	KI	I
... kennen die untergeordneten Objekte (Eigenschaften) des Window Objekt, wie z. B. <code>document</code> , <code>history</code> , <code>location</code> oder <code>navigator</code> .	I	III
... ermitteln die Höhe und Breite des Viewports (Fenstergrößen).	I	II
... arbeiten mit der <code>confirm()</code> Methode des Window Objekts um ein OK bzw. Abbrechen zu ermitteln.	I	II
... arbeiten mit der <code>prompt()</code> Methode des Window Objekts als Eingabemöglichkeit.	I	II
... öffnen ein neues Fenster mittels <code>open()</code> und einer URL.	I	II
... starten den Print-Dialog mit <code>print()</code> ;	I	II
... de- und enkodieren einen String mit Base-64 des Window Objekts.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Base-64 Kodierung ist nicht besonders sicher. Jeder kann eine Base-64 Kodierung wieder enkodieren – dennoch kann damit schnell ein Text "unleserlich" gemacht werden, um z. B. ein Passwort über eine URL (get) zu übertragen.
- Die Base-64 Kodierung passt so nicht zu 100 % zum Thema - ausgenommen von der `window.confirm()` Ausgabe. Es kann als Offtopic betrachtet werden.

Das Window Objekt ist das globale Objekt, welches ein offenes Browser-Fenster definiert. Untergeordnete Objekte sind z. B. das document, history, location oder navigator Objekt. Sie werden auch als Eigenschaften des window Objekts betrachtet. z. B. `window.document.querySelectorAll("p");`

Fenstergrößen

JS	<code>.innerHeight</code>	← Die Höhe des Viewports inkl. Scrollbars in Pixel
	<code>.innerWidth</code>	← Die Breite des Viewports inkl. Scrollbars in Pixel

```

<body onResize="fensterGR()">
  <h1>Der Viewport inkl. Scrolbars</h1>
  <p>Breite: <b id="Breit"></b> Pixel</p>
  <p>Höhe: <b id="Hoch"></b> Pixel</p>
  <script>
    function fensterGR() {
      document.getElementById("Breit").innerHTML = window.innerWidth;
      document.getElementById("Hoch").innerHTML = window.innerHeight;
    }
  </script>
</body>

```

Window Methoden

JS	<code>confirm()</code>	← Öffnet ein Dialog-Fenster mit OK oder Abbrechen. Rückgabewert ist Boolean: true oder false
----	------------------------	--

```

function meineFunktion() {
  var ausgabe = window.confirm("Sind Sie sicher?");
  alert(ausgabe);
}

```

JS	<code>prompt()</code>	← Öffnet ein Dialog-Fenster mit Eingabemöglichkeit
----	-----------------------	--

```

function meineFunktion() {
  var antwort = window.prompt("Ihr Name bitte");
  alert(antwort);
}

```

JS	<code>open()</code>	← Öffnet ein Fenster <code>window.open(https://www.css4.at);</code>
	<code>print()</code>	← Öffnet den Druck-Dialog <code>window.print();</code>

Base-64-Kodierung

JS	<code>.btoa()</code>	← Dekodiert einen String mit Base-64
	<code>.atob()</code>	← Enkodiert einen Base-64-Code

```

function kodierFunktion() {
  var str = "Streng Geheim, naja!";
  var enc = window.btoa(str);
  var dec = window.atob(enc);
  var ausgabe = "Dekodiert: " + enc + "\nEnkodiert: " + dec;
  window.confirm(ausgabe);
}

```


Übungsblatt 14.0 Window Objekt

Referenzcode: **JSU140**

Technologien: HTML, CSS, JavaScript

Dateinamen: U140_Window.odt, U140_Window.pdf

Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen ein Seitenlayout nur mit HTML und JavaScript (ohne CSS).	I	IV
... orientieren sich an einer graphischen Vorgabe.	I	II
... schreiben das Seitenlayout so, dass es das gesamte Browserfenster ausfüllt und sich bei einem <code>onResize</code> neu orientiert.	I	II
Übung B		
... erweitern die Lösung von Übung A um zwei Schieberegler, die jeweils Höhe bzw. Breite eines Divs verändern.	I	IV
... antizipieren ein komplexes Seitenlayout.	KI	III
Übung C		
... schreiben ein Script zum en- bzw. dekodieren einer Eingabe.	I	II
... nutzen die <code>window.prompt()</code> Methode für die Eingabe.	I	II
... nutzen die <code>window.confirm()</code> Methode für die Ausgabe.	I	II
Übung D		
... scripten eine Online-Suche für Duden.de.	I	IV
... verstehen den Aufbau einer URL und nutzen die <code>window.open()</code> Methode um eine externe Website aufzurufen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Übung D: Natürlich kann sich die URL Struktur von Duden.de im Lauf der Zeit ändern. Alternativ kann auch eine Google-Suche automatisiert werden.

<https://www.google.at/search?q=suchbegriff>



Übung A: Seitenlayout

Erstelle rein mit HTML und JavaScript (ohne CSS) ein Seitenlayout wie unten dargestellt. Das Layout soll den Viewport voll ausfüllen und es dürfen keine Scrollbalken erscheinen. Wenn man die Größe des Browserfensters (onResize) verändert, wird das Layout angepasst.

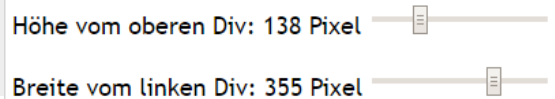
- css **Ohne CSS!**



Übung B: dynamisches Seitenlayout

Öffne deine Lösung von Übung A: Seitenlayout und erweitere sie um zwei Schieberegler (`input type="range"`). Einer verändert die Höhe des div oben und der andere die Breite von div links – natürlich dynamisch!

- JS Dynamische Veränderung
- css **Für die Schieberegler!**



Übung C: En- und Decodieren

Scripte eine Webseite zum En- und Dekodieren mit Base-64. Die Eingabe soll über ein `prompt()` Fenster passieren – die Ausgabe in einem `confirm()` Fenster.



Übung D: Duden.de

Auf duden.de findet man zu beinahe allen deutschen Wörtern einen Eintrag. Die Duden-Site ist so aufgebaut, dass über die URL auch nach einem Begriff gesucht werden kann. z. B. Eine Suche nach dem Wort Mahlzeit ist über `https://www.duden.de/suchen/dudenonline/Mahlzeit` möglich!

Scripte eine Webseite die nach einer Eingabe eines Begriffs die dazu passende Seite auf duden.de automatisch öffnet!

Lernhandout 14.1 Pop-Up Window

Referenzcode **JSL141**

Technologien JavaScript

Dateinamen L141_popup.odt, L141_popup.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... erkennen die Probleme von Pop-Up's für die Usability einer Website.	KAI	III
... öffnen ein leeres Fenster bzw. eine bestimmte URL mit <code>window.open()</code> .	I	II
... übergeben <code>target</code> Attribute an die <code>window.open()</code> Methode.	I	II
... verfeinern die Darstellung eines neuen Fenster durch die Übergabe von Features an die <code>window.open()</code> Methode.	I	II
... erstellen ein Pop-Up Fenster.	I	II
... schließen ein Fenster mit der <code>.close()</code> Methode.	I	II
... überprüfen die Instanziierung eines eigenen Pop-Up Fensters bzw. überprüfen ob ein Pop-Up Blocker das neue Fenster verhindert hat.	I	III


Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Zusätzlich gibt es noch die `.closed` Eigenschaft für das Fenster bzw. das `window` Objekt (Rückgabe: `true` oder `false`)

Mit der `open()` Methode des `window` Objekts kann man auch Pop-Up Fenster erstellen. Diese Pop-Up Fenster sind für Benutzer meist eine "nervige" Sache (besonders Werbe-Pop-Up) und sollten deshalb eher vermieden werden. Viele Browser bieten deshalb Pop-Up Blocker an, also sollte man Pop-Up's gut überlegt einsetzen.

JS 	<code>window.open (URL, Name, Feature) ;</code> Die Werte werden mit Hochkommas geschrieben. Die Features werden durch Beistriche getrennt. Z. B.: <code>window.open ("http://www.css4.at", "css4", "menubar=no") ;</code>
URL	Hier wird eine URL zu einer Webseite festgelegt. Gibt es keine URL, so wird ein leeres Fenster mit <code>about:blank</code> geöffnet!
Name	Über den Namen wird das <code>target</code> Attribut definiert, bzw. dem Fenster ein Name zugewiesen (vgl. <code><iframe></code>). Mögliche Werte sind: <code>_blank</code> Wird in ein neues Fenster geladen! <code>_parent</code> Wird in ein übergeordnetes Fenster geladen. <code>_self</code> Ersetzt die aktuelle Seite. <code>_top</code> Ersetzt ein Frameset.
Feature	<code>height=pixels</code> Die Höhe eines Fenster in Pixel. Mindestens 100 <code>width=pixels</code> Die Breite eines Fenster in Pixel. Mindestens 100 <code>menubar=yes no</code> Definiert, ob das Menü des Browserfensters angezeigt werden soll. <code>scrollbars=yes no</code> Anzeige des scrollbar (Bildlaufleiste). <code>status=yes no</code> Darstellung der Statusanzeige. <code>titlebar=yes no</code> Titelbar Anzeige. <code>toolbar=yes no</code> Anzeige der Browser-Toolbar. <code>left=pixels</code> Position des neuen Fensters, gemessen links vom Bildschirmrand in Pixel. <code>top=pixels</code> Position des neuen Fensters, gemessen vom oberen Anfang des Bildschirms in Pixel. <code>location=yes no</code> Zeigt das Adressfeld (nicht in allen Browser). <code>resizable=yes no</code> Erlaubt die Veränderung der Fenstergröße (nicht in allen Browser) Statt <code>yes</code> oder <code>no</code> kann auch <code>1</code> oder <code>0</code> eingetragen werden!

```
var radioFenster = window.open("", "Radiostream", "width=250,
    height=400, menubar=no,
    toolbar=no, left=200, top=300");
radioFenster.focus();
radioFenster.document.write("<h1>Jazzradio.com</h1>");
```

JS

`.close()` ← Schließt das Fenster wieder!

```
<button onClick="radioFenster.close();">SchlieÙe Fenster</button>
```



Mit `if(!radioFenster){...} else {...}` kann man überprüfen, ob das neue Fenster instanziiert wurde oder ob ein Pop-Up Blocker es verhindert hat.

Übungsblatt 14.1 Pop-Up Window

Referenzcode **JSU141**

Technologien HTML, CSS, JavaScript

Dateinamen U141_popup.odt, U141_popup.pdf, aboutcss.html

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Webseite, die via einem Klick ein Pop-Up öffnet und einen Kartenausschnitt von openstreetmap.org anzeigt.	I	IV
... nutzen die Benutzerschnittstellen einer fremden Website (openstreetmap.org):	I	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen eine Linkliste von einer vorgegebenen Webseite und stellen diese in einem Pop-Up Fenster dar.	I	IV
... ermitteln alle Hyperlinks des Dokuments (aboutcss.html).	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erweitern die Lösung von Übung B.	I	III
... ermitteln die Bildschirmbreite.	I	II
... positionieren ein Pop-Up Fenster neu.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

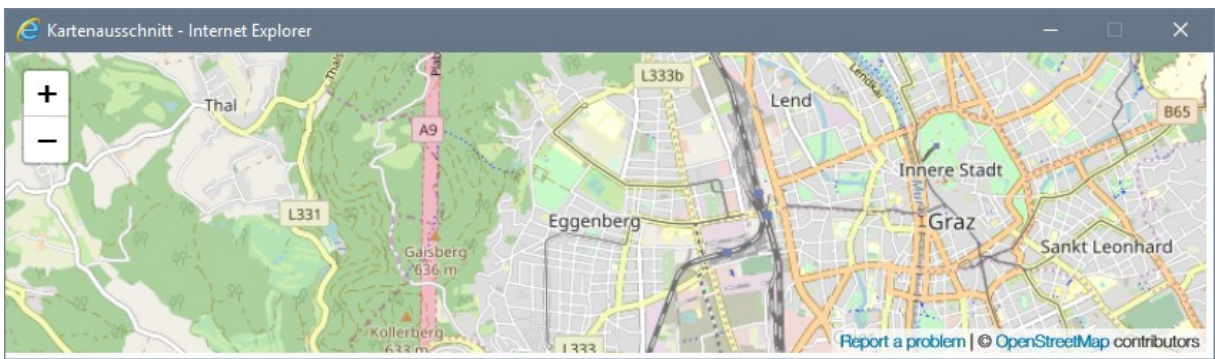
- Auf openstreetmap.org gibt es die Option zum Teilen eines Kartenausschnitts in einem Iframe. Natürlich kann sich diese Option auch im Laufe der Zeit ändern.



Übung A: Open Street Map

Auf openstreetmap.org findet man freie Straßenkarten. Suche dir einen Punkt auf der Landkarte und teile diese in Form von einem iframe. Auf der Website von openstreetmap.org gibt es einen Teilen-Button der den HTML Code ausgibt. Schreibe eine Webseite, die den Kartenausschnitt als Pop-Up öffnet.

- Suche auf openstreetmap.org nach dem Teilen-Button mit dem HTML Code.
- HTML** Ein Button zum Öffnen des Pop-Ups.
- JS** Funktion zum Öffnen des Pop-Ups, blende so viel aus, wie möglich.
- css** Ansprechende Gestaltung (Freies Design)
- Verpacke alles in einem einzigen HTML Dokument!**



Übung B: Linkliste

Öffne die Webseite aboutcss.html. Über einen Button soll sich ein PopUp Fenster mit allen Hyperlinks `<a href ...>` aus dem Dokument öffnen.

- HTML** Button
- JS** Alle Links ermitteln, Pop Up Fenster öffnen
- css** Freie Gestaltung



Lösungsansatz: Mit `.outerHTML` wird der HTML Code (Tag, Attribute, Text usw.) eines Elements ausgegeben.



Übung C: Linkliste

Öffne deine Lösung von **Übung B: Linkliste**.

Erweitere das JavaScript um eine Prüfung, ob das Fenster geöffnet wurde mit einer passenden Rückmeldung (z. B. Button verschwindet, odgl) für den User.

Zusätzlich soll das PopUp Fenster am rechten Bildschirmrand erscheinen.



*`var breite = screen.width;` ermittelt die Bildschirmbreite in Pixel.
`.moveTo(x, y)` bewegt ein Fenster zu den Koordinaten am Bildschirm.*

Lernhandout 14.2 Window Methoden

Referenzcode **JSL142**

Technologien HTML, CSS, JavaScript

Dateinamen L142_Window.odt, L142_Window.pdf, aboutcss.html

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... bewegen ein Fenster zu einer bestimmten Position am Bildschirm.	I	II
... bewegen das Fenster relativ zur eigenen Position.	I	II
... verändern dynamisch die Fenstergröße (Breite und Höhe).	I	II
... ermitteln dynamisch CSS Eigenschaften eines HTML Elements.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Webseite aboutcss.html steht auf der Lernplattform www.css4.at zum Download bereit.

Ein mit `open()` geöffnetes Fenster lässt sich noch weiter verändern, bzw. modifizieren.

JS

`.moveTo(x, y)` ← bewegt das Fenster zu einer bestimmten Position.



`.moveBy(x, y)`

Bewegt das Fenster um eine bestimmte Anzahl von Pixel, relativ zu seiner Position. x-Werte bewegen das Fenster rauf und runter, y-Werte links und rechts. Negative Werte sind möglich!

```
<button onclick="fensterAuf()">Neues Fenster öffnen</button>
<button onclick="fensterMove()">Bewege das Fenster</button>
<script>
  var fenster;
  function fensterAuf() {
    fenster = window.open("", "Info",
                          "width=200, height=100, top=200");
    fenster.document.write("<p>Wichtige Infos</p>");
  }
  function fensterMove() {
    fenster.moveBy(10, -10);
    fenster.focus();
  }
</script>
```

JS

`.resizeBy(Breite, Höhe)` ← relativ zu seiner eignen Größe in Pixel.

`.resizeTo(Breite, Höhe)` ← Vergrößert oder verkleinert ein Fenster in Pixel.

```
<button onclick="fensterAuf()">Create window</button>
<button onclick="fensterSize()">Resize window</button>
<script>
  var meinFenster;
  function fensterAuf() {
    meinFenster = window.open("", "", "width=100, height=100");
  }
  function fensterSize() {
    meinFenster.resizeTo(450, 650);
    meinFenster.focus();
  }
</script>
```

JS



`window.getComputedStyle(Element).getPropertyValue(CSS-Eigenschaft)` ;

Ermittelt das tatsächliche Aussehen eines Elements und gibt den Wert einer CSS-Eigenschaft zurück! z. B. bei einem `<div>` mit einer Breite von 100 % wird die Breite in Pixel ermittelt und zurückgegeben.

```
<div id="rotesDiv" style="height: 50px; background: red;"></div>
<script>
  var element = document.getElementById("rotesDiv");
  var dieCSSeigenschaft = window.getComputedStyle(element);
  var ausgabe = dieCSSeigenschaft.getPropertyValue("width");
  alert("Breite in Pixel: " + ausgabe);
</script>
```


Lernhandout 14.3 pageOffset

Referenzcode **JSL143**

Technologien HTML, CSS, JavaScript

Dateinamen L143_pageOffset.odt, L143_pageOffset.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... ermitteln mit <code>window.pageYOffset</code> die vertikale Scroll-Position innerhalb eines Fensters.	I	II
... ermitteln mit <code>window.pageXOffset</code> die horizontale Scroll-Position innerhalb eines Fensters.	I	II
... scrollen mit JavaScript in einem Dokument relativ zur aktuellen Position.	I	II
... scrollen mit JavaScript zu einer bestimmten Position im Dokument.	I	II
... ermitteln die Y-Position eines Elements, gemessen vom Dokumentanfang.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Mit der `window.pageYOffset` Eigenschaft ermittelt man, wie weit ein Benutzer nach unten gescrollt hat (z. B: durch drehen am Mausrad). Gemessen wird der Wert in Pixel vom Dokumentanfang. `window.pageXOffset` gibt die gescrollte Entfernung in Pixel über die Horizontale (von links nach rechts) zurück.

JS

`window.pageYOffset;`

← Gescrollt über die Y-Achse (Vertikal)

`window.pageXOffset;`

← Gescrollt über die X-Achse (Horizontal)



Im Beispiel werden 200 durchnummerierte Zeilen eingefügt. Im `<body>` Tag ist das Attribut `onscroll`. Sobald das Mausrad gedreht wird, feuert die Funktion die Y-Position in das fixierte `<div id="pos">`.

```
<body onscroll="runterScrollen();" >
  <div style="position:fixed; top:10px; right:10px;"></div>

  <script>
    for (var i = 0; i <= 200; i++) {
      document.write("<p>Zeile:" + i + "</p>");
    }

    function runterScrollen() {
      document.querySelector('div').innerHTML = window.pageYOffset;
    }
  </script>
</body>
```

JS

`window.scrollBy(x, y);`← Scrollt das Dokument um einen xy-Wert`window.scrollTo(x, y);`← Scrollt das Dokument zu einem xy-Wert

Sobald um 500 Pixel nach unten gescrollt wird, springt (scrollt) das Dokument zurück zum Anfang `window.scrollTo(0, 10);`

```
if(window.pageYOffset >= 500) {
  window.scrollTo(0, 10);
}
```

JS

`.offsetTop;`

← Ermittelt die Y-Position eines Elements



Per Click auf ein `<p>` wird seine Position in Pixel, gemessen vom Dokumentanfang ermittelt und in einem alert Dialog ausgegeben.

```
<p onClick="posY(this);" style="height:300px; background:red;"></p>
<p onClick="posY(this);" style="height:300px; background:blue;"></p>
<p onClick="posY(this);" style="height:300px; background:green;"></p>

<script>
  function posY(meinEl) {
    alert(meinEl.offsetTop);
  }
</script>
```

Übungsblatt 14.3 pageOffset

Referenzcode **JSU143**

Technologien HTML, CSS, JavaScript

Dateinamen U143_pageOffset.odt, U143_pageOffset.pdf, aboutcss.html

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erweitern eine fremde Webseite um einen "Scroll-nach-oben-Button".	I	IV
... scripten eine pageOffset Lösung.	I	II
... nutzen die CSS-Eigenschaft <code>scroll-behavior: smooth;</code>	I	II
Übung B		
... erweitern die Lösung von Übung A um ein automatisch generiertes Inhaltsverzeichnis alle <code><h2></code> Überschriften.	I	IV
... scripten eine pageOffset Lösung um über das Inhaltsverzeichnis zum passenden Text zu gelangen.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erweitern die Lösung von Übung B und fügen zwei "Rauf-Runter" Buttons hinzu, die von <code><h2></code> zur nächsten <code><h2></code> Überschrift scrollt.	I	IV
... verwenden Positionsabfragen von <code><h2></code> Überschriften und pageOffset Funktionen.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Webseite aboutcss.html steht auf der Lernplattform www.css4.at zum Download bereit.



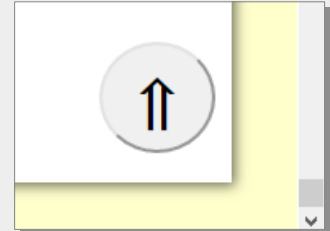
Übung A: Scrollen

Öffne die Webseite `aboutcss.html` bzw. deine Lösung von **14.1 C: Linkliste**.

Sobald um 300 Pixel nach unten gescrollt wurde, soll ein Button erscheinen, der per Klick das Dokument wieder nach oben zum Anfang scrollt.

Arbeite hier **nicht** mit Anker sondern finde eine `pageOffset` Lösung.

- HTML** Button
- JS** Funktion zur Ermittlung des `pageOffset`, Funktion für das Scrollen
- css** **Button ansprechend gestalten, unten rechts am Viewport fixieren!**



Mit der CSS Eigenschaft `html {scroll-behavior: smooth;} scrollt das Dokument geschmeidig und springt nicht nur zu seiner neuen Position!`



Übung B: Inhaltsverzeichnis

Öffne das Webdokument `aboutcss.html` bzw. deine Lösung von **Übung A: Scrollen**.

Füge unter der Hauptüberschrift `<h1>` ein automatisch generiertes Inhaltsverzeichnis mit allen `<h2>` Elementen hinzu.

Per Klick auf ein Element im Inhaltsverzeichnis soll automatisch zu diesem Punkt im Dokument gescrollt werden. Arbeite hier **nicht** mit Anker sondern finde eine `pageOffset` Lösung.

- HTML** Ausgabeelement (z. B. `<u1>`)
- JS** Funktion zur Erstellung des Inhaltsverzeichnis, Verlinkung der Verzeichnispunkte mit den `<h2>` im Dokument
- css** Ansprechende Gestaltung (Freies Design)

Inhaltsverzeichnis

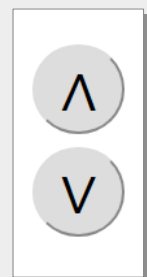
- Inhaltsverzeichnis
- Beschreibung
- Geschichte
- Grundlagen
- Kaskadierung
- Numerische Angabe
- Farben
- CSS: Hintergrund
- Box-Modell
- Schriftart
- Schriftgröße
- Schriftgewicht
- Schriftstil
- Schriftvarianten
- Schriftweite
- Korrektur der Schriftgröße
- Schrift (allgemein)
- Zeichenabstand
- Zeilenhöhe
- Textausrichtung
- Textdekoration
- Texteinrückung
- Textschatten
- Texttransformation
- vertikale Ausrichtung
- Textumbruch
- Wortabstand
- Schreibricht
- Cursor
- Tabell



Übung C: Rauf und runter

Öffne das Webdokument `aboutcss.html` bzw. deine Lösung von **Übung B: Inhaltsverzeichnis**. Erweitere die Webseite um zwei Pfeilbuttons für Rauf und Runter. Per Klick soll zur nächsten `<h2>` Überschrift gescrollt werden (bzw. zur vorherigen zurück).

- HTML** Zwei Buttons
- JS** Entsprechende Funktionen
- css** **Ansprechende Gestaltung**



Lernhandout 15.1 Navigator

Referenzcode **JSL151**
Technologien JavaScript
Dateinamen L151_Navigator.odt, L151_Navigator.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... ermitteln Browserinformationen über das <code>.navigator</code> Objekt.	I	II
... greifen auf die Eigenschaften des <code>window.navigator</code> Objekts zu, wie z. B.: Name des Browsers, Version, ob Cookies erlaubt sind, ob eine Internetverbindung besteht und auf welcher Plattform (Betriebssystem) der Browser läuft.	I	II
... ermitteln die Browser-Sprache und analysieren die unterschiedlichen Sprachcodes.	I	III
... nutzen <code>.indexOf()</code> um eine bestimmte Sprache (<code>de</code> = Deutsch) oder ein bestimmtes Land (<code>DE</code> = Deutschland) zu erkennen.	I	II
... ermitteln installierte Plug-Ins des Browsers.	I	II
... reflektieren auf mögliche Sicherheitsrisiken in Verbindung mit dem <code>.navigator</code> Objekt.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen


- Insbesondere auf alten Windows-Browsern (Edge, Explorer) gab es Plug-Ins die einen Zugriff auf das Dateisystem zuließen. Da war plötzlich Speichern und Löschen von Dateien möglich oder es gab einen Zugriff auf die Zwischenablage. Man sollte also schon gut aufpassen mit welchen Plug-Ins man im www surft.

Das Navigator Objekt gibt Informationen über den Web-Browser des Users aus. Es ist ein unmittelbares Unterobjekt von window → `var navObj = window.navigator;`

Eigenschaften von window.navigator

JS	<code>.appName</code>	← Spitzname des Browsers
	<code>.appVersion</code>	← offizieller Name des Browsers
	<code>.cookieEnabled</code>	← Browser-Version
	<code>.onLine</code>	← Cookies erlaubt (true false)
	<code>.platform</code>	← besteht eine Internetverbindung (true false)
		← Plattform, bzw. Betriebssystem

```
<p id="info"></p>
<script>
  var navObj = window.navigator;
  var ausgabe = "Browser:  " + navObj.appName + "<br>";
  ausgabe += "Spitzname: " + navObj.appCodeName + "<br>";
  ausgabe += "Version:   " + navObj.appVersion + "<br>";
  ausgabe += "Cookies:   " + navObj.cookieEnabled + "<br>";
  ausgabe += "Online:    " + navObj.onLine + "<br>";
  ausgabe += "Plattform:  " + navObj.platform + "<br>";
  document.getElementById("info").innerHTML = ausgabe;
</script>
```

JS 	<code>.language</code>	← Browser-Sprache
	<code>.language</code> liefert den Sprachcode (en = Englisch, de = Deutsch usw.) und Browserabhängig das Land (AT = Österreich, DE = Deutschland usw), dann wird z. B. de-DE oder de-AT ausgegeben. Mit <code>.indexOf()</code> durchsucht man den String ob ein bestimmter Sprachcode vorkommt. Ist der Teilstring vorhanden, dann gibt die Methode die Position im String zurück, also: größer als -1.	

```
if (navigator.language.indexOf("de") > -1) {alert("Guten Tag");}
if (navigator.language.indexOf("fr") > -1) {alert("Bonjour");}
if (navigator.language.indexOf("AT") > -1) {alert("Servus");}
```



Um die exakte geographische Position eines Device (z. B. Smartphone mit GPS) zu ermitteln, bietet `.navigator` das Objekt `.geolocation` an.

JS	<code>.plugins</code>	← Ermittelt installierte Plug-Ins im Browser
----	-----------------------	--

```
function instPlugIns(){
  var ausgabe = "";
  for (var i = 0; i < navigator.plugins.length; i++) {
    ausgabe += " NAME:  " + navigator.plugins[i].name;
    ausgabe += " ART:   " + navigator.plugins[i].description;
    ausgabe += " FILE:  " + navigator.plugins[i].filename + "<br>";
  }
  return ausgabe;
}
```

Lernhandout 15.2 Location

Referenzcode **JSL152**

Technologien JavaScript

Dateinamen L152_Location.odt, L152_Location.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... nutzen das <code>.location</code> Objekt um auf die URL (Uniform Resource Locator) zuzugreifen.	I	II
... analysieren die Bestandteile einer URL.	I	III
... verwenden <code>.location</code> Eigenschaften um auf die Bestandteile einer URL zuzugreifen.	I	II
... benutzen <code>.location</code> Methoden um eine Seite neu zu laden (<code>.reload</code>) oder eine Seite durch eine andere zu ersetzen (<code>.replace</code> , <code>.assign</code>).	I	II
... kennen die <code>history</code> Methoden <code>.back</code> , <code>.forward</code> und <code>.go</code> .	I	II

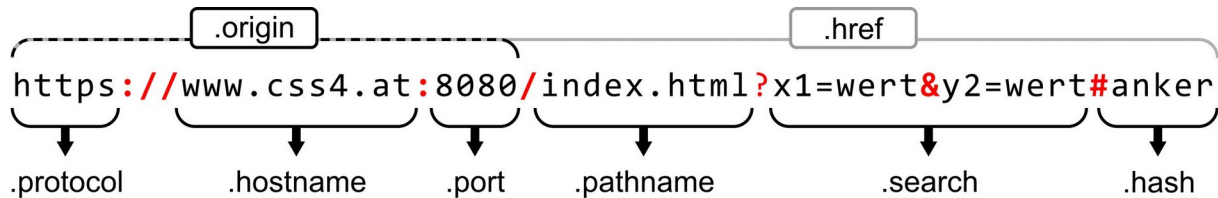
Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene




Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Geolocation wird im Lernhandout 19.1 Geolocation vertiefend erklärt.

Mit dem `Location` Objekt (Unterobjekt von `window`) hat man Zugriff auf die vollständige URL der aktuellen Webseite. Alle Eigenschaften des `location` Objekts können sowohl ausgegeben als auch gesetzt werden! URL steht für eng. Uniform Resource Locator.



JS 	<code>location.href</code> ← Die gesamte URL Definiert die gesamte URL (vom Protokoll bis zum Anker). Mit <code>location.origin</code> bekommt man das Protokoll, den Hostnamen und den Port geliefert.
JS 	<code>location.protocol</code> ← Protokoll bzw. Schema Liefert bzw. setzt das Protokoll (auch Schema genannt), einer URL. Das kann HTTP, HTTPS, FTP aber auch mailto, file, news usw. sein.
JS	<code>location.hostname</code> ← Definiert den Hostnamen (z. B. <code>www.css4.at</code>)
	<code>location.port</code> ← Port Die Angabe des Ports erlaubt die Ansteuerung eines TCP-Ports. Wird kein Port angegeben, so wird der Standard-Port des jeweiligen Protokolls verwendet – zum Beispiel bei HTTP 80, bei HTTPS 443 und bei FTP 21. Mit <code>location.host</code> wird Hostname und Port definiert.
JS	<code>location.pathname</code> ← Definiert den Pfad mit Datei (z. B. <code>/html/index.html</code>)
	<code>location.search</code> ← Abfragestring z. B. Übergabewerte Liefert bzw. setzt Übergabewerte in Form eines "Query-String". Mit einem Fragezeichen (?) beginnt der Query-String. Das Fragezeichen wird ebenfalls ausgegeben. Mehrere Übergaben sollten durch ein kaufmännisches Und (&) getrennt werden.
JS	<code>location.hash</code> ← Definiert den Anker (z. B. <code>#anker</code>)

Location Methoden

JS	<code>location.reload()</code> ← Ladet eine Seite neu
	<code>location.replace()</code> ← Ersetzt eine Webseite durch eine andere.
	<code>location.assign()</code> ← Gleich wie <code>.replace()</code> nur ohne 'Zurück-Button'.

History Methoden

JS	<code>history.back()</code> ← Öffnet die vorherige Seite nochmals (Zurück-Button).
	<code>history.forward()</code> ← Ladet die nächste Seite aus der <code>history</code> Liste.
	<code>history.go()</code> ← Springt innerhalb der <code>history</code> Liste. z. B. <code>history.go(-2)</code> ; springt zwei Seiten zurück.

Übungsblatt 15.2 Location

Referenzcode: **JSU152**

Technologien: HTML, CSS, JavaScript

Dateinamen: U152_Location.odt, U152_Location.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Webseite, die Browserinformationen ausgibt und die URL analysiert.	I	IV
... nutzen <code>.navigator</code> und <code>.location</code> Eigenschaften und Methoden.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen ein Error Doc um Fehlermeldungen des Servers darzustellen.	I	IV
... analysieren die Fehlerstruktur von Webservern über die Referenznummer und der Beschreibung.	IK	III
... nutzen <code>.navigator</code> und <code>.location</code> Eigenschaften und Methoden.	I	II
... scripten eine Funktion, die jeweils die Fehlerbeschreibung in Deutsch bzw. Englisch ausgibt – je nachdem, welche Sprache im Browser eingestellt ist.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomie: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Fremdquellen

- Auf https://offtopic.css4.at/Error_Docs.php findet man eine Tabelle über die Error-Docs. Dort findet man ein ZIP-File mit vorgefertigten Error-Docs. Die Studierenden sollen aber nicht um eine Lernerfahrung beraubt werden!

Anmerkungen

- Individuell angefertigte serverseitige Error Docs sind ein Statement für gutes Webdevelopment - auch wenn dieser der die User_in nie sehen sollte! Am prominentesten ist ein 404 - Seite nicht gefunden (Page Not Found). Es entsteht z. B. wenn ein User eine falsche URL auffordert. Man könnte hier gleich mit JavaScript eine Umleitung einrichten oder ein Feedback an den User senden. Gute Webhoster protokollieren diese Zugriffe akribisch.



Übung A: Browser Informationen

Schreibe eine Webseite die alle Informationen über den Browser ermittelt und in einer Tabelle ausgibt. Zusätzlich soll die URL ausgewertet werden.

- HTML** **Tabelle**
- JS** **Funktion, .navigator Objekt, .location Objekt,**
- CSS** **Ansprechende Gestaltung (Freies Design)**
- Teste deine Seite mit unterschiedlichen Browsern, wenn möglich online!**



Im Internet gibt es zahlreiche gratis Anbieter von Webspaces.

window.navigator		
Browser	Netscape	.appName
Spitzname	Mozilla	.appName
Version	5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36	.appVersion
Cookies	true	.cookieEnabled
Online	true	.onLine
Platform	Win32	.platform
Sprache Land	de-AT	.language
Installierte Plug-Ins		
Plug-In	Chrome PDF Plugin	
Plug-In	Chrome PDF Viewer	



Übung B: Error Docs

Die meisten Webhoster (Domain, Webspaces usw.) stellen den Webdevoloper_innen ein Verzeichnis für Error Docs zur Verfügung. Error Docs werden angezeigt, wenn ein Server-Fehler auftritt z. B. 404 – Seite nicht gefunden. So können "Server-Fehlermeldungen" dem Design der Domain angepasst und technische Informationen mit JavaScript und PHP gesammelt werden.

Scripte ein Error Doc deiner Wahl!

- Mindestanforderungen:** ein Zurück Button und eine Fehlerbeschreibung in englischer und deutscher Sprache (abhängig von den Spracheinstellungen des Browsers).

- 400 - Bad Request (Ungültige Anforderung)
- 401 - Authorization Required (Erlaubnis benötigt)
- 403 - Forbidden (Verboten)
- 404 - Page Not Found (Seite nicht gefunden)
- 405 - Method Not Allowed (Methode nicht erlaubt)
- 406 - Not Acceptable (Inakzeptabel)
- 407 - Proxy Authentication Required (Proxy-Authentifizierung erforderlich)
- 412 - Precondition Failed (Vorbedingung fehlgeschlagen)
- 414 - Request-URI Too Long (Anforderungs-URI zu lang)
- 415 - Unsupported Media Type (Nicht unterstützter Medientyp)
- 500 - Internal Server Error (Interner Serverfehler)
- 501 - Not Implemented (Nicht implementiert)
- 502 - Bad Gateway (Gateway Fehler)
- 503 - Service Temporarily Unavailable (Dienst vorübergehend nicht verfügbar)

Lernhandout 15.3 Wertübergabe GET

Referenzcode **JS1153**

Technologien HTML, JavaScript

Dateinamen L153_WertuebergabeGET.odt, L153_WertuebergabeGET.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... übergeben Werte mittels der GET Methode eines Formulars an andere Webseiten, bzw. PHP Skripte usw.	I	II
... werten eine GET-Übergabe (über die URL) mit <code>window.location.search</code> aus.	I	III
... verwenden Textoperationen um einen Übergabe-String in ein Array zu splitten.	I	II
... schreiben eine Webseite, die Übergabe-Werte auswertet.	I	II
... erkennen das Problem von Sonderzeichen im Übergabe-String und wandeln diese mit <code>unescape ()</code> ;	I	II
... nutzen die weitaus einfachere Auswertung mit <code>URLSearchParams</code> .	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Bestimmte Sonderzeichen sind in einer URL nicht erlaubt. Z. B. das Leerzeichen - deshalb wird es als Zeichenfolge %20 hinzugefügt oder ein Anführungszeichen " als %22 dargestellt.
- Streng genommen ist der Begriff: "Wertübergabe" nicht richtig. Man spricht eher von einer Parameterdeklaration bzw. Parameterdefinition. Weil aber die Werte weiter gegeben werden ist der Begriff umgänglicher. Es gibt dennoch ein Key-Value Prinzip.

Bei einer Wertübergabe eines Formulars mit der Methode GET (`method="GET"`), werden die Namen und die Werte in die URL geschrieben. Die Übergabe (Querystring) erfolgt mit einem Fragezeichen zu Beginn, Name und Wert werden durch ein Ist-Gleich-Zeichen getrennt, und zwischen den Übergabeparametern steht ein Kaufmännisches Und. `?name1=wert1&name2=wert2&name3=wert3&...`

Mittels `window.location.search` wird der Querystring aus der URL ermittelt.



Formular mit einer GET Übergabe.
name und value werden mit der URL übergeben!

```
<form method="get" action="auslesen.html">
  <p>Benutzer: <input type="text" name="Benutzer" ></p>
  <p>eMail: <input type="text" name="eMail" ></p>
  <p>Passwort: <input type="password" name="Passwort" ></p>
  <p><button type="submit" >Absenden</button></p>
</form>
```



Auswertung der URL
Hier der Code im Webdokument `auslesen.html`
Werte und Namen der Variablen werden in einer Tabelle ausgegeben.

```
<table>
  <thead><tr><th>Variablenname</th><th>Wert</th></tr></thead>
  <tbody id="ausgabeTab"></tbody>
</table>
<script>
  var abfrageURL = window.location.search; // Einlesen aus der URL
  abfrageURL = abfrageURL.slice(1); // Fragezeichen löschen
  var werteGesamt = abfrageURL.split("&"); // Array erzeugen
  var paar, ausgabe = "";
  for (var i = 0; i < werteGesamt.length; i++) {
    paar = werteGesamt[i].split("="); // Name und Wert trennen
    paar[0] = unescape(paar[0]); // Sonderzeichen umwandeln
    paar[1] = unescape(paar[1]);
    ausgabe += "<tr><td>" + paar[0] + "</td>"
    ausgabe += "<td>" + paar[1] + "</td></tr>"; }
  document.getElementById("ausgabeTab").innerHTML = ausgabe;
</script>
```



`unescape()` ; verwandelt Sonderzeichen wieder in einen lesbaren String.



Auswertung mit `URLSearchParams` Achtung: `URLSearchParams` ist zurzeit noch nicht in allen Browsern verfügbar! Mit `.get` wird der Wert einer Variable ausgegeben. Weitere Methoden wären `.getAll()`, `.has()`, `.forEach()`, `.sort()` und viele mehr.

```
var abfrageURL = window.location.search;
var sucheURL = new URLSearchParams(abfrageURL);
alert(sucheURL.get("Passwort"));
```

Übungsblatt 15.3 Wertübergabe GET

Referenzcode **JSU153**

Technologien HTML, CSS, JavaScript

Dateinamen U153_Wertuebergabe.odt, U153_Wertuebergabe.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... analysieren die Google-URL.	KI	III
... schreiben eine Webseite mit einem eigenen Google Suchfeld.	I	IV
... nutzen Textoperationen um eine sinnvolle Wertübergabe mittels GET an die Google-Suche zu realisieren.	I	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... schreiben eine Website um URL's zu analysieren.	I	IV
... verwenden dafür keine <code>URLSearchParams</code> , sondern realisieren es mit <code>window.location.search</code> .	I	II
Übung C		
... erstellen eine "Bücher-Tauschbörse" mit zwei HTML Dokumenten.	I	IV
... implementieren eine GET Übergabe, um einen Link mit allen Informationen zum Buch-Tausch zu teilen.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: Google Suche

Öffne `www.google.at` in einem Browser und Suche nach einem beliebigen Begriff. Analysiere danach die URL im Adressfeld `https://www.google.com/...`

Betrachte die übergebenen Schlüsselpaare `?q=...` und überlege was sie bedeuten können. Schreibe danach eine Webseite, mit einem eigenen Google Suchfeld und einer passenden Wertübergabe.

- HTML** Form mit der Methode GET und der Action GoogleURL.
- css** Ansprechende Gestaltung (Freies Design)



Übung B: URL Analyse

Scripte eine Get Funktion, die aus einer URL einen bestimmten Schlüsselwert ausgibt. Die URL soll über ein Textfeld ermittelt werden!

Beispiel:

URL: `...com/?ben=han&pass=solo&r2=d2#xwing`

Funktion: `zeigeWert("pass")` ; Ausgabe: `solo`

- JS** Funktion mit Return Value, Auswertung einer URL
- HTML** Eingabefeld für die URL
- Verwende keine URLSearchParams**

URL eingeben
`https://www.sternenkriegerinnen.com/mitglieder.thml?ben=han&pass=solo&r2=d2&basis=tatooine&jedi=true#xwing`

Schlüsselname

ben	han
pass	solo
r2	d2
basis	tatooine
jedi	true



Übung C: Bücher Tauschbörse

Erstelle eine einfache Bücher Tauschbörse mit zwei Webdokumenten. Im ersten wird das Buch durch Felder beschrieben (Titel, Autor, Erscheinungsjahr, ISBN, Auflage, Verlag) – die Eingaben werden in einer URL zusammengefasst. Im zweiten Webdokument wird die URL ausgelesen und optisch ansprechend ausgewertet. Die Idee: Man braucht einfach nur einen Link versenden bzw. in sozialen Netzwerken posten!

- HTML** **Zwei HTML Dokumente,**
HTML 1: Eingabeformular, HTML 2: Darstellung
- JS** Funktion zur Generierung einer URL, Auswertung der URL
- css** **Ansprechende Gestaltung (Freies Design)**

Titel des Buches

Autor|in

Erscheinungsjahr

ISBN

Verlag



Ich tausche ...

Hallo, ich tausche das Buch **Die Physiker** von **Friedrich Dürrenmatt**.
 Erschienen ist das Werk **1998** im **Diogenes** Verlag.
 Die ISBN lautet **978-3-257-23047-5**.
 Mach' mir ein Angebot. Mein eMail lautet: **ich@meineSeite.at**

Lernhandout 15.4 localStorage

Referenzcode **JSL154**

Technologien JavaScript

Dateinamen L154_localStorage.odt, L154_localStorage.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... nutzen die Web Storage API um Daten lokal auf dem Rechner des Benutzers zu speichern.	I	III
... unterscheiden zwischen <code>localStorage</code> und <code>sessionStorage</code> .	I	III
... speichern einen Schlüsselwert mit <code>setItem</code> .	I	II
... laden einen Wert mit <code>getItem</code> .	I	II
... löschen einen Schlüsselwert mit <code>removeItem</code> .	I	II
... löschen alle Einträge mit <code>clear()</code> ;	I	II
... speichern ein Array im <code>localStorage</code> und wandeln dieses in ein JSON Format.	I	II
... rufen ein gespeichertes Array wieder ab und parsen es von der JSON-Notation wieder in ein Array.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Auch für localStorage gelten die verschärften Datenschutzbestimmungen wie sie für Cookies angedacht sind.
- Im Kapitel 20.4 – JSON wird noch detaillierter auf JSON eingegangen!

Mit dem Web Storage API kann man Daten lokal auf dem Rechner des Users speichern. Es werden nur der Schlüsselname und der Schlüsselwert in Form eines String gespeichert. Die lokal gespeicherten Daten können dann von allen Skripten der Domain abgerufen werden. Das Web Storage API beinhaltet zwei Objekte: `window.localStorage` (speichert unbegrenzt und ohne Ablaufdatum) und `window.sessionStorage` (nur für eine Sitzung, nachdem das Browser-Tab geschlossen wurde, werden auch die Daten gelöscht).



Für `sessionStorage` gelten dieselben Methoden wie für `localStorage`:
`.setItem()`; `.getItem()`; `removeItem()`; und `.clear()`;

window.localStorage Methoden

JS


`window.localStorage.setItem(Name, Wert)`

← Speichern

[Name] Der Name des Eintrags

[Wert] Der String zum gespeicherten Eintrag

`.setItem()` speichert den Eintrag lokal im Cache des Browsers. Wenn ein Eintrag schon vorhanden ist, wird dieser ohne Rückmeldung überschrieben.

```
var eMailAdresse = prompt("Ihre eMail Adresse");
localStorage.setItem("benutzerMail", eMailAdresse);
localStorage.setItem("Vorname", "Mario");
```

JS

`localStorage.getItem(Name)`

← Abrufen der Speicherung

```
var neueMail = localStorage.getItem("benutzerMail");
alert(neueMail);
```

JS

`localStorage.removeItem(Name)`

← Löscht den Eintrag

```
localStorage.removeItem("benutzerMail");
```

JS

`localStorage.clear()`

← Löscht alle Einträge zur Domain

```
localStorage.clear();
```



Speicherung eines Array

`JSON.stringify()`; wandelt ein Array (bzw. ein Objekt) in eine JSON-Notation. Die JavaScript Objekt Notation (JSON) ist ein Dateiformat für den Datenaustausch zwischen Anwendungen und erlaubt das Speichern eines Arrays in einen String.

```
var meinFeld = ["Blumen", "Pflanzen", "Bäume", "Gräser"];
var meinJSON = JSON.stringify(meinFeld);
localStorage.setItem("nimmDas", meinJSON);
```



Das Array abrufen

`JSON.parse()`; wandelt die JSON-Notation wieder in ein Array um.

```
var dasFeld = localStorage.getItem("nimmDas");
var meinFeld = JSON.parse(dasFeld);
for(var i =0; i < meinFeld.length; i++) {alert(meinFeld[i]);}
```


Übungsblatt 15.4 localStorage

Referenzcode **JSU154**

Technologien HTML, CSS, JavaScript

Dateinamen U154_localStorage.odt, U154_localStorage.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... scripten einen Zähler, der jeden Besuch auf einer Webseite mitprotokoliert.	I	IV
... verwenden <code>localStorage</code> um den Zähler lokal am Rechner zu speichern.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... scripten eine Funktion, die dem Benutzer anzeigt, wann sein letzter Besuch war.	I	II
... verwenden <code>localStorage</code> um das Datum und die Uhrzeit lokal am Rechner zu speichern.	I	II
... verwenden Datumsverarbeitungen um das Script zu vervollständigen.	I	II
... erweitern die Webseite um eine Löschfunktion des <code>localStorage</code> .	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erstellen eine online "Einkaufliste".	I	IV
... verarbeiten Preis und Bezeichnung der Ware in einem Array.	I	II
... speichern das Array im <code>localStorage</code> .	I	II
... implementieren eine Löschfunktion von einzelnen Positionen.	I	II
... laden die Werte aus dem <code>localStorage</code> und parsen es wieder in ein Array.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln



Übung A: Besuchzähler

Jedes mal wenn das Webdokument geöffnet wird, soll ein Besuchzähler mitprotokollieren und die Anzahl der Besuche auf der Webseite zurückgeben.

- HTML **Ausgabe in einem HTML Element**
- JS **localStorage des Zählers**
- CSS **Ansprechende Gestaltung (Freies Design)**



Übung B: Datum

Öffne deine Lösung von **Übung A: Besuchzähler** und erweitere sie um eine Datumsausgabe des letzten Besuch. (Datum und Uhrzeit).

Füge zusätzlich noch einen Button hinzu, der den localStorage wieder löscht.

- HTML **Ausgabeelement, Button**
- JS **localStorage des Datums, Datumsverarbeitung, Löschfunktion von localStorage**
- CSS **Ansprechende Gestaltung (Freies Design)**



Übung C: Einkaufsplan

Erstelle eine Art "Einkaufsliste" in einem Webdokument. In ein Input Feld soll die Bezeichnung der Ware eingegeben werden. In einem weiteren Input Feld der dazu passende Preis.

Alle Preise sollen addiert und als Summe ausgegeben werden. Per Mausklick kann man ein Element (Bezeichnung und Preis) von der Liste löschen.

Der Einkaufsplan soll lokal gespeichert werden und wird bei einem nochmaligen Aufruf automatisch aus dem lokalen Speicher geladen.

- Arbeite mit Arrays
- HTML **Zwei Eingabefelder, Ausgabe (z. B. Tabelle), Buttons**
- JS **Arrays verarbeiten, localStorage**
- CSS **Ansprechende Gestaltung (Freies Design)**



Lernhandout 15.5 Cookies

Referenzcode **JS1155**

Technologien JavaScript

Dateinamen L155_Cookies.odt, L155_Cookies.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... wissen, was Cookies sind und wie diese aufgebaut sind.	IK	III
... kennen die Datenschutzrichtlinien für Cookies.	IK	II
... verstehen das Key-Value Prinzip (Schlüsselname und Schlüsselwert).	IK	I
... erstellen Cookies mit Name, Wert und Lebenszeit.	I	II
... unterscheiden bei der Lebenszeit zwischen max-age und expire .	I	II
... lesen gesetzte Cookies.	I	II
... löschen Cookies durch Überschreiben mit einem Ablaufdatum von 0 Sekunden.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Cookies sind kleine Textfiles die auf dem Device (Smartphone, Computer usw.) des Users lokal gespeichert werden. Aus Datenschutzgründen muss die Speicherung von Cookies explizit erlaubt werden. Ein Cookie besteht mind. aus dem Schlüsselnamen, dem Schlüsselwert, und einer Lebenszeit. Die Cookies sind nur innerhalb einer Domain gültig – also, nur jener der das Cookie setzt kann es auch wieder auslesen!

`navigator.cookieEnabled`; ermittelt ob die Browsereinstellungen überhaupt Cookies erlauben – die Rückgabe ist ein Boolean Wert, also `true` oder `false`!

Cookie erstellen

JS



```
document.cookie = "name=wert; max-age=sekunden"
```

Achtung: Wenn ein Cookie bereits besteht, und ein neues mit dem gleichen Schlüsselname gespeichert wird, dann wird das alte Cookie überschrieben.

name Schlüsselname bzw. Bezeichner

wert Schlüsselwert z. B. `farbe=red`;

max-age Lebenszeit des Cookie in Sekunden, ohne Angabe verfällt das Cookie nachdem der Browser geschlossen wurde (Session-Cookie)

sekunden 86 400 Sekunden sind ein Tag

```
document.cookie = "benutzer=herbert; max-age=86400";
```



Statt `max-age` kann das Ablaufdatum mit `expire` gesetzt werden, also ein bestimmtes Datum. Damit die Datumsübergabe gut funktioniert, sollte man ein Datumsobjekt vereinbaren `new Date()` und das Objekt dann mit `.toGMTString()`; übergeben.

Im Beispiel werden zum heutigen Datum 86 400 000 Millisekunden hinzuaddiert – also das Ablaufdatum wird auf Morgen gesetzt.

```
var jetzt = new Date();
jetzt.setTime(jetzt.getTime() + 86400000);
document.cookie = "ablauf=morgen; expires=" + jetzt.toGMTString();
```

Cookies lesen

JS



```
var xyz = document.cookie;
```

Alle Cookies werden wie in einem String zurückgegeben und können dann durch Textoperationen ausgewertet werden.

Die Rückgabe hat das Schema: `name1=wert1; name2=wert2; name3=wert3`

```
var meineKekse = document.cookie;
alert(meineKekse);
```

Cookies löschen



Da Cookies überschrieben werden, braucht man nur das zu löschende Cookie nochmals mit einem Ablaufdatum von 0 Sekunden setzen.

```
document.cookie = "benutzer=; max-age=0";
```

Übungsblatt 15.5 Cookies

Referenzcode **JSU155**

Technologien HTML, CSS, JavaScript

Dateinamen U155_Cookies.odt, U155_Cookies.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... recherchieren die rechtlichen Fragen zu Cookies und localStorage.	SI	III
... beachten die rechtlichen Bestimmungen zur lokalen Speicherung als Teil des professionellen Webdesigns.	AS	II
... erstellen ein Webdokument aus dem Recherche-Ergebnis.	I	II
Übung B		
... erweitern die Übung A um eine Schaltfläche mit der Erlaubnis zum Speichern von Cookies.	I	IV
... speichern die Erlaubnis Cookies zu verwenden als Cookie ab!	I	II
... nutzen den Erlaubnis Text auch für andere Webprojekte.	SI	II
... testen das Dokument auf einem Webserver bzw. in einer Virtualisierung.	I	II
Übung C		
... erweitern die Lösung von Übung B und ändern mit CSS das Aussehen so, dass ein vertikaler Scrollbalken erscheint.	I	IV
... ermitteln ob das Speichern von Cookies erlaubt und möglich ist und speichern die Scrollposition als Cookie.	I	II
... schreiben eine Funktion, die bei einem neuen Öffnen bzw. Neuladen, automatisch zur letzten Position scrollt.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Übungen sind aufbauend auf der Lösung der vorherigen.



Übung A: Datenschutzverordnung

Recherchiere im Internet die aktuellen Bestimmungen zur Datenschutzverordnung, dem Telekommunikationsgesetz bzw. dem E-DSVO oder der E-Privacy-Verordnung der EU hinsichtlich der lokalen Speicherung von Daten (personenbezogene, notwendige, sensible usw.).

Mit anderen Worten: "Was ist rechtlich zur Speicherung von Cookies (bzw. localStorage) zu beachten!".

- Erstelle aus deinem Rechercheergebnis eine Webseite.**



Übung B: Cookie OK

Öffne dein Webdokument aus der **Übung A: Datenschutzverordnung** und erweitere es um eine Schaltfläche zur ausdrücklichen Erlaubnis zum Speichern von Cookies.

Klickt der User auf die Schaltfläche, dann wird die Erlaubnis als Cookie gespeichert. Erlaubt der User die Verwendung von Cookies, dann soll bei einem weiteren Öffnen der Webseite, die Erlaubnis nicht wieder erscheinen.

- JS** **Abfragen und setzen von Cookies**
- css** Ansprechende Gestaltung (Freies Design)
- Teste deine Seite**



Cookies sind nur innerhalb einer Domain gültig. Localhost bzw. 127.0.0.1 werden von den Browsern als Domain interpretiert.

Wir verwenden Cookie-Technologie und speichern einige technische Informationen um die Webseite optimiert darzustellen. Es werden keine personenbezogenen Daten gespeichert. Bitte klicken Sie auf den Button um die Speicherung von Cookies zu erlauben.
Cookie-Status: Speicherung verboten!

Cookies speichern erlauben



Übung C: Gescrollt

Öffne deine Lösung von **Übung B: Cookie OK** und ändere das Aussehen mit CSS. Man soll im Dokument vertikal scrollen können. Um den Bildlauf zu ändern können z. B. die Schriftgröße geändert, das Padding erhöht oder der Zeilenabstand vergrößert werden. Das Dokument kann auch um zusätzlichen Text erweitert werden.

Die vertikale Scrollposition soll lokal gespeichert werden. Bei einem erneuten Öffnen des Webdokuments scrollt ein JavaScript automatisch zur letzten Position – aber nur, wenn die Speicherung von Cookies (Übung B) erlaubt wurde!

- JS** `.pageYOffset` und `.scrollTo()`; Lokal speichern wahlweise mit **localStorage** oder als **Cookie**.
- css** Text so verändern, dass vertikal gescrollt werden kann.

Lernhandout 16.1 EventListener

Referenzcode **JSL161**

Technologien HTML, JavaScript

Dateinamen L161_Eventlistener.odt, L161_Eventlistener.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... setzen dynamisch Event Listener an HTML Elemente um ein Event auszulösen.	I	II
... entfernen Event Listener von einem HTML Element.	I	II
... kennen das "Best Practice" einen Event Listener am Ende des Bodys zu schreiben.	I	II
... kennen Druck und Zwischenablagen Events, wie z.B. afterprint , beforeprint , copy , cut und paste .	I	II
... schützen Webseiten vor "copy&paste" mit einem copy Event.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Ereignisse bzw. Events können mit `addEventListener()` an jedes HTML Element angehängt werden. Dabei können beliebig viele Events auf ein HTML Element gelegt werden (sogar vom gleichen Typ). Die Events können dynamisch vergeben werden und erleichtern damit auch die Lesbarkeit vom HTML-Code.

```
HTML <button id="Schalter" onClick="starte();">Drück mich</button>
JS   document.getElementById("Schalter").addEventListener("click", starte);
```

JS



`.addEventListener(Event, Funktion);`

Der Event wird mit Anführungszeichen definiert.

```
<button id="Sender">Absenden</button>

<script>
  document.getElementById("Sender").addEventListener("click", Sende);
  function Sende() {alert("Absenden"); }
</script>
```

JS



`.removeEventListener(Event, Funktion);`

Ein dynamisch hinzugefügter Event kann mit `.removeEventListener` wieder entfernt werden!

```
document.getElementById("Sender").removeEventListener("click", Sende);
```



Am besten schreibt man den JavaScript-Code am Schluss des `<body>` Elements. Dann ist der DOM eingelesen und die Eventlistener können definiert werden!

Druck und Zwischenablagen Events

Die Events für Druck und Zwischenablage dürfen in allen Elementen vorkommen außer im `<head>`. Die Events können direkt auf das globale `window` Objekt gelegt werden!

z. B. `window.addEventListener("afterprint", meldung);`

Druck	afterprint	Nach dem Druckauftrag, bzw. nach geschlossener Druckvorschau
	beforeprint	Vor dem Druck-Dialog
Zwischenablage	copy	Wenn eine Auswahl in die Zwischenablage kopiert wird
	cut	Wenn eine Auswahl in die Zwischenablage ausgeschnitten wird.
	paste	Beim Einfügen aus der Zwischenablage.

```
window.addEventListener("copy", meldung);
function meldung() {alert("Urheberrechte beachten"); }
```


Übungsblatt 16.1 EventListener

Referenzcode **JSU161**

Technologien HTML, CSS, JavaScript

Dateinamen U161_Eventlistener.odt, U161_Eventlistener.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen eine Webseite mit "Energieeffizienzklassen".	I	IV
... fügen einen Button für den Druck hinzu.	I	II
... setzen einen Eventlistener für den Druck.	I	II
... scripten eine CSS Medienabfrage für den Druck.	I	II
... orientieren sich an der Text- und Bildvorgabe.	KI	III
... hinterfragen den Sinn von Energieklassen für den Umweltschutz.	AS	III
Übung B		
... erweitern die Lösung von Übung A.	IS	II
... verwenden Eventlistener mit einem this Argument.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Übung A: Um die Pfeile zu gestalten gibt es selbstverständlich mehrere Lösungswege. Am einfachsten ist es, am Ende des Balken ein Graphikelement (z. B. svg) einzublenden.
Mit CSS, wird ein inline-block Element gedreht:

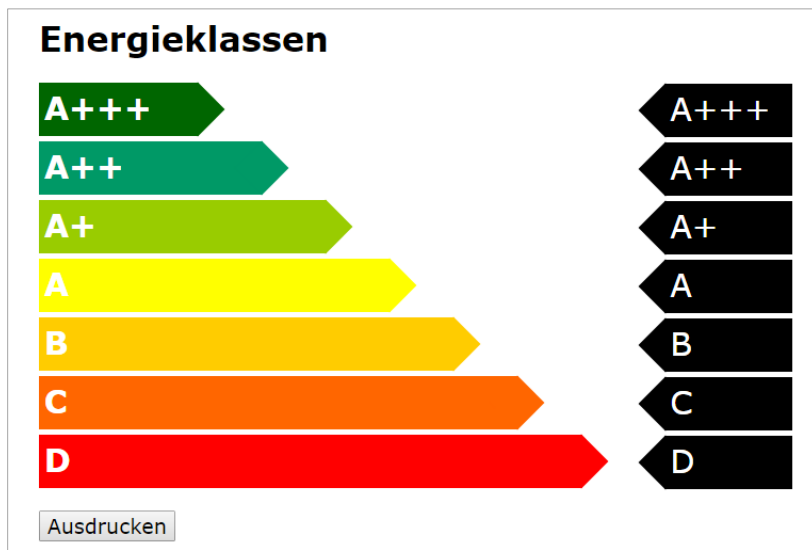
```
<span class="pfeil">&nbsp;&nbsp;&nbsp;</span>
.pfeil {transform: rotate(45deg); height:36px; width:36px;
position:relative; left: -19px; bottom: 12px;
display:inline-block;}
```
- Übung B: Man kann sich vorstellen in einem Elektronik-Fachhandel zu arbeiten. Mit dieser Lösung kann für jedes Elektrogerät eine Energieeffizienzgraphik ausgedruckt werden.



Übung A: Energieklassen

Erstelle eine Webseite mit einer Aufstellung von Energieeffizienzklassen (siehe unten). Füge noch einen Button für den Druck hinzu.
Sobald der User einen Druckbefehl erteilt, soll das aktuelle Datum mit Uhrzeit und einem Copyright Vermerk hinzugefügt werden.

- HTML Ein Webdokument mit Energieklassen Darstellung (siehe unten)
- JS Eventlistener (beforeprint und afterprint)
- css Medienabfrage für den Druck



Übung B: Energieklassen II

Öffne dein Lösung von **Übung A: Energieklassen**. Alle schwarzen Pfeile (rechts) sollen ausgeblendet werden. Per Klick auf eine farbige Energieklasse soll der dazupassende schwarze Pfeil wieder erscheinen.

Am Ausdruck darf auch nur **ein** schwarzer Pfeil sein, natürlich jener der angeklickt wurde. Die farbigen Pfeile bleiben wie sie sind.

- Löse dieses Beispiel mit Eventlistener
- JS Eventlistener und .style Anweisungen
- css Ansprechende Gestaltung (Freies Design)



Code-Tipp: Hier ein Eventlistener, der eine Funktion mit dem this Argument startet:
`.addEventListener("click", function(){einblenden(this);})`

Land	Schweiz	Österreich	Deutschland
Amtssprache	Deutsch	Deutsch	Deutsch
Hauptstadt	Bern	Wien	Berlin
Fläche	41 285 km ²	83 878 km ²	357 385 km ²
Einwohner	8,4 Mio	8,7 Mio	82,5 Mio
Staatsform	föderale Republik	Bundesrepublik	Bundesrepublik
Währung	Schweizer Franken	Euro	Euro
Internet-TLD	.ch, .swiss	.at	
Telefon-Vorwahl	+41		

Lernhandout 16.2 Keyboard Events

Referenzcode **JSL162**

Technologien HTML, JavaScript

Dateinamen L162_Keyboardevents.odt, L162_Keyboardevents.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... setzen Keyboard-Events, die über die Tastatureingabe gefeuert werden.	I	II
... kennen die Keyboard-Events: onkeydown , onkeyup , onkeypress .	I	II
... arbeiten mit Event-Wertübergaben.	I	II
... lesen den KeyCode einer Event-Übergabe aus.	I	II
... kennen beide Eigenschaften (.keyCode und .which) zum Auswerten einer Event-Übergabe und formulieren eine Cross-Browser-Solution.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Mit einem Keyboard Event kann ein Tastendruck ermittelt werden. Die Events können als Attribut in einem HTML Element definiert werden bzw. dynamisch mit einem Eventlistener. Ein Objekt wäre z. B. `var objekt = document.querySelector("input");`

onkeydown feuert sobald eine Taste heruntergedrückt wird

JS `objekt.addEventListener("keydown", meineFunktion);`

HTML `<div onkeydown="meineFunktion()">`

onkeyup feuert sobald eine Taste losgelassen wird

JS `objekt.addEventListener("keyup", meineFunktion);`

HTML `<div onkeyup="meineFunktion()">`

onkeypress wenn Taste heruntergedrückt und festgehalten wird

JS `objekt.addEventListener("keypress", meineFunktion);`

HTML `<div onkeypress="meineFunktion()">`

Objektübergabe mit event

Startet man eine Funktion mit der Objektübergabe `event`, dann wird der Tastendruck an die Funktion übergeben. Mit `event.key` wird der Tastendruck ausgewertet.

JS `var meineTaste = event.key;`

```
<input type="text" onKeyDown="welcheTaste(event);" >
<script>
  function welcheTaste(event) {
    var meineTaste = event.key;
    alert(meineTaste);
  }
</script>
```

event.keyCode Eigenschaft

JS `event.keyCode` ← Gibt den Code der Taste zurück.
z. B.: a = 97, b = 98, Enter = 13

```
function welcheTaste(event) {
  if(event.keyCode == 13) {alert("Enter wurde gedrückt");} }
}
```



Weitere Eigenschaften die einen Unicode zurückgeben sind:

`event.charCode` oder `event.which`.

Beachte: `event.which` wird nicht vom Internet Explorer unterstützt und `event.keyCode` funktioniert nicht als `onkeypress` Event im Firefox.

Deshalb gibt es eine Cross-Browser-Solution in der einfach beide Eigenschaften vereinbart werden (abhängig vom Browser):

```
var meineTaste = event.which || event.keyCode;
```

Übungsblatt 16.2 Keyboard Events

Referenzcode: **JSU162**

Technologien: HTML, CSS, JavaScript

Dateinamen: U162_Keyboardevents.odt, U162_Keyboardevents.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Webseite mit einer Such-Übergabe für wordpress.com	I	IV
... feuern ein Event durch den Tastendruck auf Return (bzw. Enter).	I	II
... analysieren die Suche und die URL von wordpress.com.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen eine Webseite mit beliebigen Fülltext, die eine Schriftgrößennavigation besitzt.	I	IV
... setzen für die Tasten + und – ein Keyboard-Event.	I	II
... schreiben eine Funktion, um die Schriftgröße um 1 pt zu vergrößern und zu verkleinern.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... schreiben einen Tetris-Teil, mit Animation und Pfeiltastennavigation.	I	IV
... erstellen mit HTML und CSS ein L-förmiges Tetris Element.	I	II
... scripten das herunterfallen mit einem <code>setInterval</code> .	I	II
... definieren Eventhandler für die Pfeiltastennavigation.	I	II
... nutzen CSS Eigenschaften für die Animation des L-förmigen Tetris Element.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Übung B: Für den Fülltext können auch die Files `aboutcss.html` und `Elemente_einf.html` herangezogen werden. Für einen anderen Fülltext sind die Urheberrechte zu beachten!



Übung A: Wordpress Suche

Wordpress.com ist einer der größten Blog Anbieter im Netz. Demgemäß gibt es auch eine Vielzahl an Blogs mit Blogbeiträgen (auch zum Thema JavaScript).

Schreibe eine Webseite mit einem Suchfeld ohne Buttons. Sobald ein Suchbegriff eingegeben und die Enter (bzw. Return) Taste gedrückt wurde, soll sich wordpress.com öffnen und die Suche automatisch ausführen.

- HTML** **Ein Suchfeld**
- JS** **Keyboard Event (Enter- bzw. Return Taste)**
- css** **Ansprechende Gestaltung (Freies Design)**
- Alternativ kann auch eine Suche auf www.post.at gescriptet werden!**



Gib auf wordpress.com ins Suchfeld einen Begriff ein und analysiere dann die URL.

Wordpress-Suche



Übung B: Schriftgröße

Erstelle eine Webseite mit einem beliebigen Fülltext. Mit einem Tastendruck auf die Minustaste (-) wird die Schriftgröße um 1pt kleiner. Mit einem Tastendruck auf die Plusstaste (+) wird die Schriftgröße um 1pt größer.

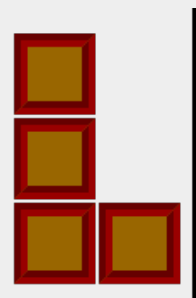
- HTML** **Ein <p> mit einem coolen Text. Thema: frei wählbar.**
- JS** **Keyboard Event mit .style Anweisung**
- css** **Ansprechende Gestaltung (Freies Design)**



Übung C: Tetris

Lasse ein L-förmiges Tetriselement langsam vom Bildschirmrand oben herunterfallen. Mit den Pfeiltasten kann das Element nach links oder rechts navigiert werden. Mit der Pfeiltaste hinauf und hinunter wird das Element gedreht.

- HTML** Ein L-Förmiges Tetriselement
- JS** Alle notwendigen Funktionen
- css** Ansprechende Gestaltung (Freies Design)
- Automatisches Herunterfallen
- ← Pfeil Taste: Bewegung nach Links
- Pfeil Taste: Bewegung nach Rechts
- ↑ Pfeil Taste: Drehung gegen den Uhrzeigersinn
- ↓ Pfeil Taste: Drehung im Uhrzeigersinn



Lernhandout 16.3 Mouse Event Typen

Referenzcode **JS1163**

Technologien HTML, JavaScript

Dateinamen L163_Mouseevents.odt, L163_Mouseevents.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... nutzen Mouse Events um Funktionen mittels Mausaktivität zu feuern.	I	II
... arbeiten mit dem <code>click</code> bzw. <code>onclick</code> Event.	I	II
... kennen das <code>dblclick</code> (Doppelklick mit der linken Maustaste) Event.	I	II
... feuern Funktionen bei einem Klick mit der rechten Maustaste.	I	II
... starten ein Funktion bei Mausbewegungen wie z. B.: <code>mouseenter</code> , <code>mouseleave</code> und <code>mousemove</code> .	I	II
... scripten eine JavaScript Lösung mit Mouse Events die der CSS Pseudoklasse <code>.hover</code> entspricht.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Die Mouse Events starten Funktionen sobald der Mauszeiger bewegt, bzw. wenn ein Mousebutton gedrückt wird.

click	Wird ausgeführt, sobald auf das Element mit der linken Maustaste gedrückt wird
JS	<code>objekt.addEventListener("click", meineFunktion);</code>
HTML	<code><div onclick="meineFunktion();"></code>
<pre> <h2 onClick="textAn()">Großer Flohmarkt</h2> <h2 id="h2u">Kleiner Megastore</h2> <script> document.getElementById("h2u").addEventListener("click", textAn); function textAn() { alert("Auch Sie sind herzlich eingeladen!"); } </script> </pre>	
dblclick	Feuert nach einem Doppelklick mit der linken Maustaste
JS	<code>objekt.addEventListener("dblclick", meineFunktion);</code>
HTML	<code><div ondblclick="meineFunktion();"></code>
contextmenu	Startet eine Funktion, wenn die rechte Maustaste gedrückt wird
JS	<code>objekt.addEventListener("contextmenu", meineFunktion);</code>
HTML	<code><div oncontextmenu="meineFunktion();"></code>
mouseenter	sobald der Mauszeiger auf ein Element bewegt wird.
JS	<code>objekt.addEventListener("mouseenter", meineFunktion);</code>
HTML	<code><div onmouseenter="meineFunktion();"></code>
mouseleave	sobald der Mauszeiger das Element verlässt.
JS	<code>objekt.addEventListener("mouseleave", meineFunktion);</code>
HTML	<code><div onmouseleave="meineFunktion();"></code>
<pre> <div id="inhalt" style="height:200px; width:200px; background:blue; margin:auto;"></div> <script> var mausEle = document.getElementById("inhalt"); mausEle.addEventListener("mouseenter", rein); mausEle.addEventListener("mouseleave", raus); function rein() {mausEle.style.backgroundColor = "green";} function raus() {mausEle.style.backgroundColor = "red";} </script> </pre>	
mousemove	feuert jedes mal, wenn die Mouse in einem Element bewegt wird.
JS	<code>objekt.addEventListener("mousemove", meineFunktion);</code>
HTML	<code><div onmousemove="meineFunktion();"></code>

Lernhandout 16.4 Mouse Eigenschaften

Referenzcode **JSL164**

Technologien HTML, JavaScript

Dateinamen L164_Mouseeigenschaften.odt, L164_Mouseeigenschaften.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... übergeben Mouse Eigenschaften mit der Event Übergabe.	I	II
... ermitteln mit der <code>.buttons</code> Eigenschaft, welche Maustaste gedrückt wurde.	I	II
... erkennen, dass mit der geometrischen Folge von 2, Mehrfachselektionen durch Addition möglich sind.	IK	III
... ermitteln mit <code>.offsetX</code> und <code>.offsetY</code> die Position des Mauszeigers.	I	II
... ermitteln die Position des Mauszeigers relativ zum Dokument.	I	II
... ermitteln die Position des Mauszeigers relativ zum Bildschirm.	I	II
... ermitteln die Position des Mauszeigers relativ zum Viewport.	I	II
... kombinieren Mausevents mit Funktionstasten (z. B. <code>.altKey</code>)	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- 1, 2, 4, 8, 16, 32, 64 usw. Addiert man bei Mehrfachselektion, so ergibt sich eine Summe mit eindeutigen Werten:
z. B. $9 = 8 + 1$ oder $14 = 8 + 4 + 2$ oder $68 = 64 + 4$

Das *Mouse Event* Objekt ermittelt die Maustasten bzw. die Position des Mauszeigers. z. B.: `onClick="meineFunktion(event)"` Zusätzlich kann abgefragt werden, ob eine Funktionstaste (Shift, Alt, Ctrl) gedrückt wurde.

JS

**.buttons**

Gibt eine Zahl zurück die einer Maustaste entspricht. Wenn zwei oder mehr Tasten gedrückt wurden, wird die Zahl addiert: Linke Maustaste (1) plus rechte Maustaste (2) ergibt (3). Alternative Eigenschaften sind: `.button` und `.which`

1.....	Linke Maustaste	2.....	rechte Maustaste
4.....	Mausrad-Button	8.....	Vierte Maustaste
16.....	Fünfte Maustaste		

```
<div style="background: green; height: 5em;"
  onMouseDown="mausTaste(event);" > </div>
<script>
  function mausTaste(event) {
    var taste = event.buttons;
    switch(taste) {
      case 1: console.log("Linke Taste"); break;
      case 2: console.log("Rechte Taste"); break;
      case 3: console.log("Linke und Rechte Taste"); break;
      case 4: console.log("Mittlere Taste"); break;}}
</script>
```



Das `onMouseDown` Attribut startet eine Funktion sobald eine Maustaste gedrückt wird – egal ob linke oder rechte Maustaste.

JS



.offsetX ← Position des Mauszeiger über die X-Achse eines Elements.

.offsetX ← Position des Mauszeiger über die X-Achse eines Elements.

Die Eventeigenschaften `.offsetX` und `.offsetY` geben die Position des Mauszeigers innerhalb eines HTML Elements zurück.

```
<textarea onMouseMove="mausPosition(event);" > </textarea>
<p>Koordinaten X: <span id="xAchse"></span></p>
<p>Koordinaten Y: <span id="yAchse"></span></p>
<script>
  function mausPosition(event) {
    document.getElementById("xAchse").innerHTML = event.offsetX;
    document.getElementById("yAchse").innerHTML = event.offsetY; }
</script>
```



`.pageX` und `.pageY` ermitteln die Position relativ zum Dokument.
`.screenX` und `.screenY` ermitteln die Position relativ zum Bildschirm.
`.clientX` und `.clientY` ermitteln die Position relativ zum Viewport.

Funktionstasten in Verbindung mit einem `onMouseDown` Event.

`.altKey` Bei gleichzeitig gedrückter Alt Taste und Maustaste.
`.ctrlKey` Bei gleichzeitig gedrückter CTRL Taste und Maustaste.
`.shiftKey` Bei gleichzeitig gedrückter Shift Taste und Maustaste.

Übungsblatt 16.4 Mouse Eigenschaften

Referenzcode **JSU164**

Technologien HTML, CSS, JavaScript

Dateinamen U164_Mouseeigenschaften.odt, U164_Mouseeigenschaften.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen eine horizontale Webseiten-Navigation.	I	IV
... scripten einen Hover Effekt mit Mouse Events in JavaScript, ohne CSS.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... schreiben ein Script, welches über den gesamten Viewport ein Fadekreuz legt.	I	IV
... erzeugen ein Fadekreuz mit durchgezogenen Linien, die sich am Mauszeiger orientieren.	I	II
... nutzen Mouse Events und Positionsermittlungen des Mauszeiger.	I	II
... ändern mit CSS den Mauszeiger (cursor).	I	II
Übung C		
... entwickeln einen Color Picker mit variabler Farbpalette.	I	IV
... erzeugen eine Farbpalette mit Schleifen, inline HTML Elementen und Farbzweisungen.	I	IV
... erweitern die Farbpalette um Schieberegler der den Rot-Wert ändert.	I	II
... verwenden EventListener mit Mouse Events.	I	II
... scripten einen Tool-Tip neben dem Mauszeiger.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

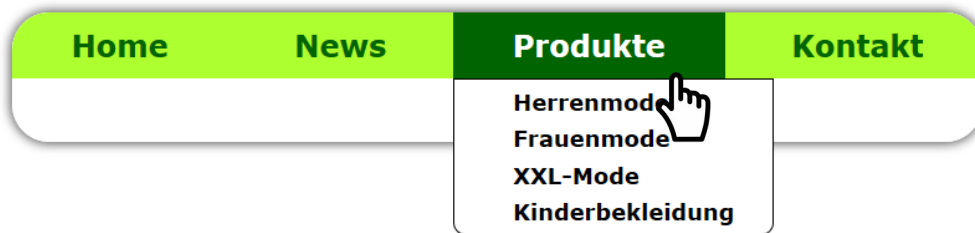
Anmerkungen



Übung A: Navigation

Erstelle eine horizontale Navigation mit den Buttons für Home, News, Produkte und Kontakt. Wenn die Maus über den Navigationsbutton Produkte fährt, erscheint ein Drop-Down Menü mit den Unterpunkten: Herrenmode, Frauenmode, XXL-Mode und Kinderbekleidung.

- HTML **Navigation mit HTML Elementen**
- JS **Hover-Effekt mit JavaScript**
- CSS **Ansprechende Gestaltung (Freies Design)**



Übung B: Fadenkreuz

Schreibe ein Script, das über den gesamten Viewport ein Fadenkreuz legt. Also eine durchgezogene Linie vom Bildschirmrand links nach rechts und von oben nach unten. Das Fadenkreuz verändert sich mit der Position der Maus – der Schnittpunkt des Fadenkreuz ist also die Mausposition.

- JS **Entsprechende Mouseevents und Eigenschaften**



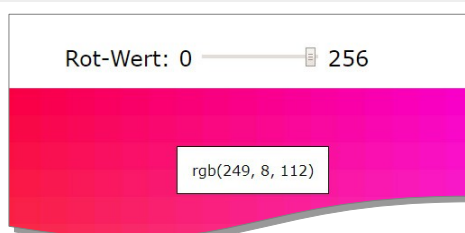
Die CSS Eigenschaft `cursor:crosshair;` verwandelt den Mauszeiger in ein Fadenkreuz. `cursor:none;` blendet den Mauszeiger komplett aus.



Übung C: Color Picker

Füge eine Farbpalette für einen bestimmten Rot-Wert ein. Sobald sich der Rot-Wert verändert, wird die Farbpalette (Grün und Blau-Werte) aktualisiert. Wenn man mit dem Mauszeiger über eine Farbe fährt, soll der RGB-Wert in einem Tool-Tip neben dem Mauszeiger angezeigt werden.

- HTML **Schieberegler für den Rot-Wert, Ausgabeelement für die Farbpalette.**
- JS **Farbpalette erzeugt durch Zählschleifen, Mouseevents und Eigenschaften für den Tool-Tip**
- CSS **Ansprechende Gestaltung (Freies Design)**



Lernhandout 16.5 Touch Events

Referenzcode **JSL165**

Technologien HTML, JavaScript

Dateinamen L165_Touchevents.odt, L165_Touchevents.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen Touch Events für berührungssensitive Devices (z. B. Tablets, Smartphone usw.).	I	I
... arbeite mit Touch-Events für den Beginn der Berührung, mit touchstart .	I	II
... arbeite mit Touch-Events für das Wischen, mit touchmove .	I	II
... arbeite mit Touch-Events für das Ende der Berührung, mit touchend .	I	II
... arbeite mit Touch-Events für den Abbruch einer Berührung, mit touchcancel .	I	II
... ermitteln mit der .targetTouches Eigenschaft eine TouchList.	I	II
... ermitteln mit wie vielen Fingern (oder auch Eingabegeräte wie Stiften) das TouchScreen berührt wird.	I	II
... ermitteln mit .touches die Koordinaten einer Berührung.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Natürlich sollen unsere Webseiten auch auf mobilen Geräten gut funktionieren. Dafür gibt es Touch Events für berührungssensitive Touchscreens. Sie ähneln den Mouse-Events.

touchstart	Beginn der Berührung ⇔ mousedown
JS	<code>objekt.addEventListener("touchstart", meineFunktion);</code>
HTML	<code><div ontouchstart="meineFunktion();"></code>
touchmove	Wischen ⇔ mousemove, pointermove
JS	<code>objekt.addEventListener("touchmove", meineFunktion);</code>
HTML	<code><div ontouchmove="meineFunktion();"></code>
touchend	Ende der Berührung ⇔ mouseup, pointerup
JS	<code>objekt.addEventListener("touchend", meineFunktion);</code>
HTML	<code><div ontouchend="meineFunktion();"></code>
touchcancel	Abbruch der Berührung ⇔ mouseup, pointerup
JS	<code>objekt.addEventListener("touchcancel", meineFunktion);</code>
HTML	<code><div ontouchcancel="meineFunktion();"></code>

JS**.targetTouches**

Die `.targetTouches` Eigenschaft liefert eine TouchList mit Touch Objekten für jeden Finger der das aktuelle Element berührt.

z. B. `.targetTouches[0]` ← für den ersten Finger

Das Beispiel ermittelt, wieviel Finger ein `<div>` berühren.

```
<div ontouchstart="meinTouch(event);">
<p id="ausgabe" style="height: 500px; background: green;">
</p></div>
<script>
  function meinTouch(event) {
    var wieViele = "Anzahl der Finger: ";
    wieViele += event.targetTouches.length;
    document.getElementById("ausgabe").innerHTML = wieViele;
  }
</script>
```

JS**.touches**

← Liefert eine TouchList mit aktuellen Berührungen am Screen

Das Beispiel ermittelt die Koordinaten des Touch Events.

```
<body ontouchstart="zeigeXY(event);" ontouchmove="zeigeXY(event);">
<div id="anzeige" style="height:400px; background:red"></div>
<script>
  function zeigeXY(event) {
    var x = event.touches[0].clientX;
    var y = event.touches[0].clientY;
    document.getElementById("anzeige").innerHTML = x + ", " + y;
  }
</script></body>
```

Lernhandout 17.1 Audio

Referenzcode **JSL171**

Technologien HTML, JavaScript

Dateinamen L171_Audio.odt, L171_Audio.pdf, Europahymne.mp3

Konvergenz HCL103, HCU103

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... fügen Audio Elemente in den HTML Code (DOM) hinzu.	I	II
... bearbeiten Audio Elemente als Knotenpunkte und definieren ein Soundfile.	I	II
... starten ein Audio Element mit JavaScript.	I	II
... pausieren Audio Elemente mit JavaScript.	I	II
... nutzen Audio Eigenschaften um die URL eines Soundfiles zu ermitteln.	I	II
... nutzen Audio Eigenschaften um ein Soundfile in einer Endlosschleife abzuspielen.	I	II
... schalten ein Audio Element lautlos bzw. ermitteln den Status mit der <code>.muted</code> Eigenschaft.	I	II
... verändern oder ermitteln die Lautstärke mit <code>.volume</code> .	I	II
... ermitteln oder bestimmen die Position in Sekunden mit <code>.currentTime</code> .	I	II
... greifen mit der <code>.controls</code> Eigenschaft auf den Status des Kontrollpanel zu.	I	II
... ermitteln mit <code>.duration</code> die Länge eines Soundfiles.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Mit HTML5 ist es möglich, ganz einfach ein Audioelement mit `<audio>` hinzuzufügen.

```
<audio src="Europahymne.mp3" controls>.
```

Hier wird gezeigt, wie ein Audiofile mit JavaScript geladen und gespielt wird.



Audio Element als Knoten hinzufügen.

Das Audioelement wird zum `<body>` hinzugefügt und mit der `.src` Eigenschaft wird die Quelle ausgewählt (diese kann absolut oder relativ sein).

```
var audioSpieler = document.createElement("audio");
document.body.appendChild(audioSpieler);
audioSpieler.src = "Europahymne.mp3";
```



audioSpieler wurde als globales Objekt vereinbart. Damit kann es über eine Funktion abgespielt werden.

Die Funktion kann z. B. von einem Button gestartet werden:

```
<button onClick="abspielen() ;">Abspielen</button>
```

oder als `onLoad` Attribut im Body Tag `<body onLoad="abspielen() ;">` dann wird das Soundfile sofort gestartet.

```
function abspielen() {audioSpieler.play();}
```



Mit einer weiteren Funktion lässt sich das Audiofile pausieren.

```
<button onClick="pause() ;">Abspielen</button>
```

```
function pause() {audioSpieler.pause();}
```

Audio Eigenschaften

Alle Eigenschaften sind sowohl als `get` wie auch `set` verfügbar. Sie können also ausgelesen bzw. gesetzt werden. (`.duration` ist natürlich nur read-only).

JS

`.currentSrc` ← Die URL des aktuellen Soundfile.

```
console.log(audioSpieler.currentSrc);
```

JS

`.loop` ← Boolesch (true|false), Bei true als Endlosschleife.

`.muted` ← Boolesch (true|false), Bei true wird es auf Lautlos gestellt.

```
audioSpieler.loop = true;
audioSpieler.muted = false;
```

JS

`.volume` ← Liest oder setzt die Lautstärke. 0.3 = 30 % Lautstärke.

`.currentTime` ← Liest oder setzt die Position in Sekunden.

```
audioSpieler.volume = 0.3;
audioSpieler.currentTime = 3.43;
```

JS

`.controls` ← (true|false) Anzeige mit oder ohne Kontrollpaneel

`.duration` ← Read-only, gibt die Länge des Soundfile zurück.

Übungsblatt 17.1 Audio

Referenzcode	JSU171
Technologien	HTML, CSS, JavaScript
Dateinamen	U171_Audio.odt, U171_Audio.pdf
Konvergenz	HCL103, HCU103

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... erstellen eine Webseite mit Bildern, Hintergrundmusik und einer Funktion zum Lautlos-Schalten der Hintergrundmusik.	I	IV
... deaktivieren die Audio-Control-Elemente (<code>.controls</code>).	I	II
... bearbeiten die Audio Eigenschaften mit JavaScript (automatisches Abspielen udgl.).	I	II
... fügen einen Button zum Lautlos (bzw. Pausieren) hinzu.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... entwerfen ein Sound-Design für eine Navigation (für Buttons, Musik-Bett für den Hintergrund).	I	IV
... nutzen das Internet um Sound-Files zu finden und beachten dabei die Urheberrechte.	I	III
... verbinden Mouse-Events (z. B. <code>mouseenter</code>) mit Funktionen zum Abspielen von Sound-Files.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung C		
... erstellen einen eigenen MP3 Player.	I	IV
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Übung B: Die Übung A von **16.4 – Mouse Eigenschaften** kann als Navigation verwendet und um das Sound-Design ergänzt werden.

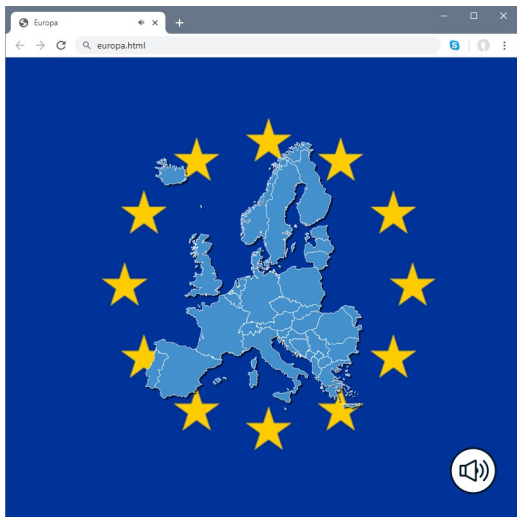


Übung A: Europa

Erstelle eine Webseite mit der EU-Flagge im Hintergrund und einem transparenten PNG von Europa (EU-Mitgliedstaaten) im Vordergrund. Beide Bilder findest du im Internet.

Sobald die Webseite im Browser geöffnet wird, soll automatisch die Europahymne abgespielt werden. Die Control-Elemente des Audio-Elements dürfen dabei nicht aufscheinen – jedoch soll ein Button zum Lautlos schalten bzw. Pausieren hinzugefügt werden.

- Sichtbar ist: Die EU-Flagge, Europa PNG und ein "Lautlos-Button".
- JS **Automatisches Abspielen eines MP3
Funktion zum Lautlos-Schalten bzw. Pause.**
- css **Ansprechende Gestaltung (Freies Design)**



Übung B: Sound-Design

Scripte ein Sound-Design für Buttons in einer Navigation. Soll heißen: Bei einem MouseOver, bei einem MouseClick usw. wird ein Geräusch abgespielt. Im Hintergrund soll ein Musik-Bett laufen! Suche im Internet nach passenden Sounds oder erstelle sie selbst (z. B. mit Audacity).

- HTML **mind. 3 Navigationsbuttons**
- JS **Diverse Events, Abspielen von Sounds**
- css **Ansprechende Gestaltung (Freies Design)**



Übung C: MP3 Player

Erstelle einen eigenen online MP3 Player.

- Einzige Vorgabe: keine Controls im <audio> Element.
- HTML **Freies Design**
- css **Freies Design**

Lernhandout 17.2 Video

Referenzcode **JSL172**

Technologien HTML, JavaScript

Dateinamen L172_Video.odt, L172_Video.pdf, croissants.mp4

Konvergenz HCL104, HCU104

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... übertragen die Erkenntnisse des Audio Element auf das Video Element.	KIS	III
... laden ein bestimmtes Video mit der <code>.load()</code> Methode.	I	II
... kennen des Sinn eines Fallbacks in Verbindung mit dem <code><source></code> Element innerhalb des <code><video></code> Elements.	I	II
... prüfen die Browserunterstützung für Video Formate und Codecs.	I	III
... kennen typische Werte (mit Codecs) für Video Formate.	I	I
... nutzen die <code>.preload</code> Eigenschaft um das Ladeverhalten eines Video Files zu definieren.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Für das `<video>` Element gibt es zahlreiche JavaScript Methoden und Eigenschaften. Die JavaScript Eigenschaften und Methoden vom `<audio>` Element (siehe 17.1 Audio) sind auch für das `<video>` Element anwendbar.

JS



`.load()`; ← Ladet ein neues Video.

Die Wenn im `<video>` Element mit `<source>` mehrere Videofiles definiert wurden (z. B. auch bei einem Fallback), kann mit `.load()` und der ID ein bestimmtes Video geladen werden.

```
<button onclick="meinVideo();">Anderes Video</button>
<video id="meinVideoElement" controls autoplay>
  <source id="mp4Video" src="clip.mp4" type="video/mp4">
  <source id="oggVideo" src="clip.ogg" type="video/ogg">
  Dein Browser unterstützt kein HTML5 video.
</video>
<script>
  function meinVideo() {
    document.getElementById("mp4Video").src = "movie.mp4";
    document.getElementById("oggVideo").src = "movie.ogg";
    document.getElementById("meinVideoElement").load();
  }
</script>
```

JS



`.canPlayType (Wert)` ;

Prüft ob ein bestimmtes Video-Format (bzw. ein Codec) vom Browser unterstützt wird. Die Methode gibt "probably" → der Browser unterstützt den Video-Typ, "maybe" → der Browser unterstützt den Video-Typ aber nicht unbedingt den Codec. oder "" → (leerer String) zurück, in diesem Fall wird der Video-Typ definitiv nicht unterstützt.

Typische Werte

video/ogg
video/mp4
video/webm
audio/ogg
audio/mp4
audio/mpeg

Typische Wert mit Codecs

video/ogg; codecs="theora, vorbis"
video/mp4; codecs="avc1.4D401E, mp4a.40.2"
video/webm; codecs="vp8.0, vorbis"
audio/ogg; codecs="vorbis"
audio/mp4; codecs="mp4a.40.5"

```
function dasVideoStarten() {
  var meinVideo = document.createElement('video');
  var spieltEs = meinVideo.canPlayType('video/mp4');
  if (spieltEs == "") {
    alert("Kein Browser-Support für den Video-Typ");
  }
  else {meinVideo.src = "croissants.mp4";
        meinVideo.play(); }
}
```

JS



`.preload = ""`

Mit der `.preload` Eigenschaft wird ein Video unmittelbar nachdem die Seite geladen wurde ebenfalls geladen. Werte sind: `auto` (sofortiges Laden), `metadata` (nur die Metadaten) bzw. `none` (kein Laden).
z. B. `meinVideo.preload = "auto";`

Lernhandout 17.3 Audio|Video Events

Referenzcode **JSL173**

Technologien HTML, JavaScript

Dateinamen L173_AV-Events.odt, L173_AV-Events.pdf, croissants.mp4

Konvergenz HCL103, HCL104

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... erkennen die Verwendung von AV-Events für Audio und Video Elemente.	I	III
... nutzen das loadstart Event zu Beginn des Ladeprozesses.	I	II
... verwenden das progress Event für den Beginn des Download.	I	II
... verwenden das canplay Event, um eine Funktion zu feuern, wenn das Video-File genug Buffer hat um zu starten.	I	II
... nutzen das waiting Event für den Fall, das ein Video-Frame noch gebuffert werden muss.	I	II
... verwenden das playing Event für wenn ein gestopptes Video-File wieder abgespielt wird.	I	II
... nutzen das seeked Event, um eine Funktion zu einer Positionsveränderung durch den Benutzer zu feuern.	I	II
... schreiben ein Scirpt mit dem seeked Event und einer .currentTime Eigenschaft (Abschnitt 1 oder 2).	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Die Audio bzw. Video Events können als Attribut im `<audio>` und `<video>` Element vorkommen oder als Eventlistener in Form einer Objekt-Vereinbarung. Selbstverständlich feuern auch die meisten anderen Events (z. B. Maus, Keyboard, Touch usw.) mit den AV-Elementen. Hier ein kleiner Auszug der Audio-Video-Events:

loadstart	Zu Beginn des Ladeprozess, wenn der Browser nach dem AV-Element sucht!
JS	<code>objekt.addEventListener("loadstart", meineFunktion);</code>
HTML	<code><video onloadstart="meineFunktion();"></code>
progress	Sobald der Download beginnt
JS	<code>objekt.addEventListener("progress", meineFunktion);</code>
HTML	<code><video onprogress="meineFunktion();"></code>
canplay	Sobald das Video geladen ist und genug Buffer hat, um es zu starten!
JS	<code>objekt.addEventListener("canplay", meineFunktion);</code>
HTML	<code><video onprogress="meineFunktion();"></code>
waiting	Feuert, wenn das Video-Frame gebuffert werden muss.
JS	<code>objekt.addEventListener("waiting", meineFunktion);</code>
HTML	<code><video onwaiting="meineFunktion();"></code>
playing	Wenn nach einer Pause (bzw. wegen Bufferung) wieder abgespielt wird.
JS	<code>objekt.addEventListener("playing", meineFunktion);</code>
HTML	<code><video onplaying="meineFunktion();"></code>
seeked	Wenn der User die Position des Videos verändert hat.
JS	<code>objekt.addEventListener("seeked", meineFunktion);</code>
HTML	<code><video onseeked="meineFunktion();"></code>



Beispiel seeked: Sobald der User die Zeit-Position im Video verändert hat, wird entweder 1. Abschnitt oder 2. Abschnitt im `<h1>` Element ausgegeben.

```
<h1 id="status">Croissants backen!</h1>
<video controls src="croissants.mp4" id="meinVideo"></video>
<script>
  var dasVideoElement = document.getElementById("meinVideo");
  dasVideoElement.addEventListener("seeked", meineFunktion);

  function meineFunktion() {
    if (dasVideoElement.currentTime <= 30) {
      var ausgabeText = "1. Abschnitt";
    }
    else {var ausgabeText = "2. Abschnitt";}
    document.getElementById("status").innerHTML = ausgabeText;
  }
</script>
```

Lernhandout 18.1 Canvas

Referenzcode **JSL181**

Technologien HTML, JavaScript

Dateinamen L181_Canvas.odt, L181_Canvas.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... erkennen das <code><canvas></code> Element als Container für 2D und 3D Graphiken.	KI	III
... nutzen das <code><canvas></code> Element mit seinen Attributen und einem Fallback.	I	II
... vereinbaren ein <code><canvas></code> Element mit <code>.getContext</code> .	I	II
... erzeugen ein Rechteck mit definierter Rahmenfarbe und –breite.	I	II
... bestimmen eine Hintergrundfarbe für ein Rechteck.	I	II
... erzeugen einen Farbverlauf für ein Rechteck.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Das `<canvas>` Element ist ein Container für 2D und 3D Graphiken. Mit JavaScript kann man Pfade, Rechtecke, Kreise, Text und Bilder hinzufügen. Das `<canvas>` Element stellt eine Bitmap-basierte Leinwand mit einem kartesischen Koordinatensystem zur Verfügung. Der Ursprung (0 | 0) ist in der linken oberen Ecke.

HTML



`<canvas> ... </canvas>`

Wenn man die Höhe und Breite des Canvas als Attribute (**height**, **width**) definiert in Pixel und nicht über CSS bestimmt, werden die Standardeinstellungen des Browsers überschrieben.

Text und Content innerhalb des `<canvas>` Tags kann als Fallback verwendet werden falls der Browser keine Canvas unterstützt.

```
<canvas id="meinCanvas" height="300" width="600">
  Dein Browser unterstützt keine Canvas!</canvas>
```

JS



`.getContext('2d');`

Nachdem das `<canvas>` Element selektiert wurde, kann ein 2D Objekt vereinbart werden.

```
var leinwand = document.getElementById("meinCanvas");
var rahmen = leinwand.getContext('2d');
```

JS



`.rect(x, y, Breite, Höhe)`

← Erzeugt ein Rechteck

Im Beispiel wird ein Rechteck an der Position $x = 30$ und $y = 100$ (vom linken oberen Ursprung des canvas) mit einer Breite von 200 Pixel und einer Höhe von 80 Pixel gezeichnet.

```
rahmen.strokeStyle = "green"; // ← Definiert die Rahmenfarbe
rahmen.lineWidth = "5";      // ← Definiert die Rahmenbreite (5 Pixel)
rahmen.rect(30, 100, 200, 80);
rahmen.stroke();             // ← Zeichnet das Rechteck.
```

JS

`.fillStyle`

← Bestimmt eine Hintergrundfarbe für das Rechteck

```
rahmen.fillStyle = "#FF0000"; // ← Roter Hintergrund
rahmen.fill();                // ← Füllmethode
```

JS



`.createLinearGradient(x0, y0, x1, y1);`

← Farbverlauf

Im Beispiel geht der Verlauf von oben ($x_0 = 0$ | $y_0 = 0$) nach unten ($x_1 = 0$ | $y_1 = 300$) mit den Farbverlauf von Blau nach Rot.

```
var verlauf = rahmen.createLinearGradient(0, 0, 0, 300);
verlauf.addColorStop(0, "blue");
verlauf.addColorStop(1, "red");
rahmen.fillStyle = verlauf;
rahmen.fill();
```


Lernhandout 18.2 Canvas Pfade

Referenzcode **JSI182**

Technologien HTML, JavaScript

Dateinamen L182_Canvas.odt, L182_Canvas.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... zeichnen einen Pfad ins <code><canvas></code> Element.	I	II
... setzen einen Start- bzw. Referenzpunkt für den Pfad.	I	II
... zeichnen eine Pfadlinie mit der <code>.lineTo()</code> Methode.	I	II
... schließen einen offenen Pfad und zeichnen diesen mit der <code>.stroke()</code> Methode.	I	II
... erzeugen ein grünes Dreieck in ein <code><canvas></code> Element.	I	II
... nutzen die <code>.save()</code> und <code>.restore()</code> Methode um ein Abbild des Pfades zu speichern bzw. wieder zurückzusetzen.	I	II
... wandeln einen Pfad in einen Container mit der <code>.clip()</code> Methode.	I	II
... prüfen, ob ein bestimmter Punkt innerhalb eines Pfades liegt.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Mit der `.beginPath()` Methode kann ein Pfad in ein Canvas Element gezeichnet werden. Zur Erinnerung: Ein Canvas ist wie ein kartesisches Koordinatensystem mit dem Ursprung (0 | 0) in der linken oberen Ecke. Zuerst wird ein 2D-Objekt vereinbart. Am Objekt wird die `beginPath()` Methode ausgeführt. Mit `.moveTo(x, y)` wird der Startpunkt (Referenzpunkt) definiert und mit `.lineTo(x, y)` das Ziel. `.closePath()` schließt den Pfad – und kehrt wieder zurück zum Referenzpunkt. `.stroke()` zeichnet schlussendlich den Pfad.



Ein grünes Dreieck

```
<canvas id="meinCanvas" width="300" height="300">
  Canvas wird nicht unterstützt!</canvas>
<script>
  var leinwand = document.getElementById("meinCanvas");
  var meinPfad = leinwand.getContext("2d");




  meinPfad.beginPath();           // ← Pfad wird gestartet
  meinPfad.lineWidth = "5";      // ← Breite des Pfad in Pixel
  meinPfad.strokeStyle = "green"; // ← Farbe des Pfad (Grün)
  meinPfad.moveTo(10, 10);       // ← Referenzpunkt/Startpunkt
  meinPfad.lineTo(140, 140);     // ← Erstes Ziel
  meinPfad.lineTo(290, 10);     // ← Zweites Ziel
  meinPfad.closePath();          // ← zurück zum Referenzpunkt
  meinPfad.stroke();             // ← Pfad wird gezeichnet
</script>
```



Sobald der Pfad mit `.closePath()` geschlossen wurde kann das Dreieck mit einer Farbe (bzw. einem Verlauf) gefüllt werden.

```
meinPfad.fillStyle = "red";
meinPfad.fill();
```

Weitere Methoden

JS 	<code>.save();</code> <code>.restore();</code> Speichert das Abbild des Pfades (z. B. Farbe). Mit <code>.restore()</code> wird das Abbild wieder zurückgesetzt.
JS 	<code>.clip();</code> Sobald ein Pfad mit <code>.clip()</code> gestartet wird, werden weitere Zeichnungen nur mehr innerhalb des Pfades dargestellt.
JS 	<code>.isPointInPath(x, y);</code> Prüft ob der Punkt (x, y) innerhalb des Pfades ist. Rückgabe ist boolesch.

```
alert(meinPfad.isPointInPath(50, 60));
```

Lernhandout 18.3 Canvas Text & Bild

Referenzcode **JSL183**

Technologien HTML, JavaScript

Dateinamen L183_Canvas.odt, L183_Canvas.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... fügen Text und Bilder in ein <code><canvas></code> Element.	I	II
... nutzen die <code>.strokeText()</code> Methode um eine Text zu zeichnen.	I	II
... erzeugen eine Textfüllung mit der <code>.fillText()</code> Methode.	I	II
... verwenden die <code>.drawImage()</code> Methode um ein Bild dazustellen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



In Canvas' können auch Texte und Bilder angezeigt werden. Für Texte muss über die `.font` Methode die Schriftart und –größe definiert werden.

Text hinzufügen

JS



`.strokeText(text, x, y);`

Die `.strokeText` Methode zeigt nur den Umriss des Text an. Über die `.font` Methode wird zuerst Schriftgröße und Schriftart definiert.

`text` Text der angezeigt werden soll
`x` x-Position vom linken Rand des Canvas in Pixel
`y` y-Position vom oberen Rand des Canvas in Pixel

```
<canvas id="meinCanvas" width="300" height="200">
  Ihr Browser unterstützt keine Canvas!
</canvas>
<script>
  var dasCanvas = document.getElementById("meinCanvas");
  var ctx = dasCanvas.getContext("2d");
  ctx.font = "40px Arial";
  ctx.strokeText("Willkommen", 40, 100);
</script>
```



Mit `.fillText()` wird Textfüllung angezeigt.

```
ctx.fillStyle = "blue";
ctx.fillText("Willkommen", 40, 100);
```

Ein Bild darstellen

JS



`.drawImage(img, x, y);`

Zuerst wird das Canvas-Objekt vereinbart und ein Image-Objekt dimensioniert. Um die `.drawImage` Methode auszuführen, sollte das Image-Objekt (`src`) vollständig geladen sein. Entweder man selektiert also ein `img`-HTML Element oder man führt die `.drawImage` Methode in einer eigenen Funktion aus, die über ein `.onload` Event gestartet wird. Genauer genommen, das vollständige Laden des Image-Objekts.

```
<canvas id="meinCanvas" width="500" height="500">
  Ihr Browser unterstützt keine Canvas!
</canvas>
<script>
  var dasCanvas = document.getElementById("meinCanvas");
  var ctx = dasCanvas.getContext("2d");
  var meinBild = new Image();

  meinBild.onload = function() {
    ctx.drawImage(meinBild, 50, 50);
  }
  meinBild.src = "pointer.png";
</script>
```

Übungsblatt 18.3 Canvas

Referenzcode: **JSU183**

Technologien: HTML, CSS, JavaScript

Dateinamen: U183_Canvas.odt, U183_Canvas.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Webseite, welche die Olympischen Ringe in einem <code><canvas></code> Element zeichnet.	I	IV
... recherchieren im Internet, wie man Kreise zeichnet.	I	II
... verleihen den Ringen unterschiedliche Farben.	I	II
... erstellen eine einfache Animation der Ringe.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen eine Webseite mit einem kartesischen Koordinatensystem, in welches eine lineare Gleichung dargestellt werden kann.	I	IV
... zeichnen die X und Y Achse eines Koordinatensystem und ergänzen es um eigene Vorstellungen (z. B. ein Raster, Größenveränderungen)	I	II
... schreiben eine Funktion, um in einer Linearen Gleichung x und f(x) zu berechnen.	I	II
... zeichnen die Lineare Gleichung in das Koordinatensystem.	I	II
... erstellen mehrere Ebenen in dem man zwei oder mehr <code><canvas></code> Elemente übereinander legt.	I	III
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen





Übung A: Olympische Ringe

Auf einem Canvas Element sollen die Olympischen Ringe dargestellt werden.

- Recherchiere im Internet, wie man auf einem Canvas Kreise zeichnet
- Die Ringe sollen in unterschiedlichen Farben dargestellt werden.
- Animiere die Darstellung der Ringe**



Übung B: Koordinatensystem

Zeichne ein kartesisches Koordinatensystem in ein Canvas Element. Zusätzlich soll es noch möglich sein, eine lineare Gleichung $f(x) = kx + d$ ins Koordinatensystem zu zeichnen.

- HTML** Canvas Objekt, Buttons, Input-Felder
- JS** Ein Koordinatensystem zeichnen
- css** Ansprechende Gestaltung (Freies Design)
- Erweitere das Koordinatensystem nach eigenen Vorstellungen (z. B. ein Raster).**

Koordinatensystem

Breite: Höhe: Raster: Ja Nein

Lineare Gleichung: $f(x) = kx + d$

k = d =



Tip: Ebenen

Wenn man einmal mehrere Ebenen benötigt (ähnlich wie bei Photoshop) dann kann man zwei oder mehrere `<canvas>` Elemente mit der CSS Eigenschaft `position: absolute;` übereinanderlegen.

```
CSS:
.meinCan {position: absolute;}
```

```
HTML:
<div>
  <canvas id="meinCanvas1"
    class="meinCan"
    height="300"
    width="300">
  </canvas>
  <canvas id="meinCanvas2"
    class="meinCan"
    height="300"
    width="300">
</canvas>
</div>
```

Lernhandout 19.1 Geolocation

Referenzcode **JSL191**

Technologien JavaScript

Dateinamen L191_Geolocation.odt, L191_Geolocation.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen unterschiedliche Formen der geographischen Positionsermittlung wie z. B. WLAN, Funk oder GPS.	I	I
... erkennen Probleme des Datenschutzes in Verbindung mit Geolocation.	SKI	III
... kennen die Objektzugehörigkeit von Geolocation zum <code>.navigator</code> Objekt.	I	II
... starten eine geographische Positionsermittlung mit <code>.getCurrentPosition()</code> mit unterschiedlichen Funktionsprüfungen.	I	II
... ermitteln die Koordinaten eines Device mit der <code>.coords</code> Eigenschaft.	I	II
... ermitteln weitere Eigenschaften die <code>.coords</code> liefert, wie z. B.: Breiten- und Längenangaben, Genauigkeit, Höhenangaben, Richtung, Geschwindigkeiten und Zeit einer Positionsangabe.	I	II
... schreiben einen Fehler-Handler mit Switch-Verzweigung.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

■

Die Geolocation API liefert die geographische Position eines Users bzw. Device. Dabei greift die API auf IP-Basis, WLAN-Netzwerkdaten, Funk-Signale oder GPS zu und ermittelt so die Position des Device (z. B. Tablett, Smartphone). Wegen dem Datenschutz wird der/die User_in oft aufgefordert, den Zugriff auf den Standort zu erlauben. Darüber hinaus erlauben moderne Browser den Zugriff auf den Standort nur in einem "sicheren Kontext" (d. H. HTTPs, lokale Dateien, localhost). Geolocation ist ein Objekt von `window.navigator`. → `navigator.geolocation`.

JS



`.getCurrentPosition(zeigePositionFunktion, zeigeFehlerFunktion) ;`

Im Beispiel wird zuerst geprüft, ob der Browser das `.geolocation` Objekt unterstützt. Ist diese Prüfung `true`, dann ruft die `.getCurrentPosition()` Methode eine Funktion auf welche die Positionsdaten übergibt und eine Funktion die etwaige Fehler auswertet.

```
function ermittlePosition() {
    var meinePos = navigator.geolocation;
    if (meinePos) {
        meinePos.getCurrentPosition(zeigePosition, zeigeFehler);
    }
    else {alert("Ihr Browser unterstützt keine Geolocation.");}
}
```

JS



`.coords`

Die `zeigePosition` Funktion übernimmt die Positionsdaten welche über die `.coords` Eigenschaft ausgelesen werden können.

Folgende Eigenschaften können ausgelesen werden:

`.coords.latitude`.....Breitenangabe als Dezimalzahl
`.coords.longitude`.....Längenangabe als Dezimalzahl
`.coords.accuracy`.....Genauigkeit der Koordinaten (in Meter)
`.coords.altitude`.....Höhenangabe (über dem Meeresspiegel)
`.coords.altitudeAccuracy`.....Genauigkeit der Höhenangabe
`.coords.heading`.....Richtung
`.coords.speed`.....Geschwindigkeit (in m/s)
`.timestamp`.....Zeit der Positionsangabe

```
function zeigePosition(position) {
    alert("Breite: " + position.coords.latitude);
    alert("Länge: " + position.coords.longitude);
    alert("Höhe: " + position.coords.altitude); }

```



Etwaige Fehler werden in einer Funktion abgefangen und über eine Switch-Verzweigung ausgewertet.

```
function zeigeFehler(fehler) {
    switch(fehler.code) {
        case fehler.PERMISSION_DENIED:
            alert("Benutzer lehnte Standortabfrage ab."); break;
        case fehler.POSITION_UNAVAILABLE:
            alert("Standortdaten sind nicht verfügbar."); break;
        case fehler.TIMEOUT:
            alert("Die Standortabfrage dauerte zu lange."); break;
        case fehler.UNKNOWN_ERROR:
            alert("unbekannter Fehler."); break;}}

```


Übungsblatt 19.1 Geolocation

Referenzcode: **JSU191**

Technologien: HTML, CSS, JavaScript

Dateinamen: U191_Geolocation.odt, U191_Geolocation.pdf

Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Website über einen österreichischen Berg mit seinen Attraktionen.	I	IV
... informieren sich über diesen österreichischen Berg.	I	II
... fügen einen Höhenmesser hinzu.	I	II
... fügen eine Entfernungsmessung, von der aktuellen Position bis zur Bergspitzen hinzu.	I	II
... recherchieren im Internet die Funktionsweise der <code>.watchPosition()</code> Methode.	IK	III
... erweitern das JavaScript um einen Error-Handler (Fehler-Handler)	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erweitern ihre Lösung von Übung A um eine Positionsanzeige auf OpenStreetMaps.org oder Google Maps.	I	IV
... recherchieren und entwickeln eine Schnittstelle zum Online Karten Anbieter.	I	III
... stellen die Online Karte in einem <code><iframe></code> Element dar.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Fremdquellen

- OpenStreetMaps.org bzw. Google Maps

Anmerkungen

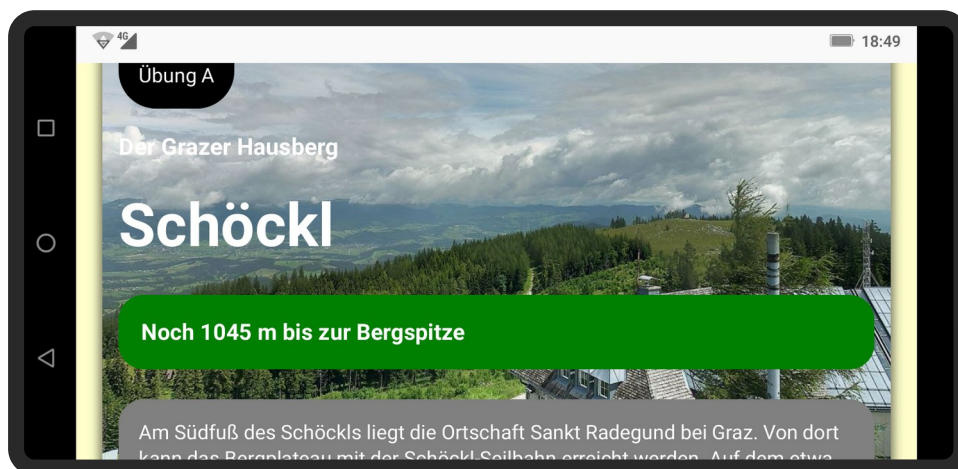
-



Übung A: Berg-App

Schreibe eine Webseite über einen österreichischen Berg mit seinen Attraktionen und Besonderheiten. Zusätzlich soll ein Höhenmesser eingefügt werden, der die Entfernung von der aktuellen Position bis zur Bergspitze anzeigt.

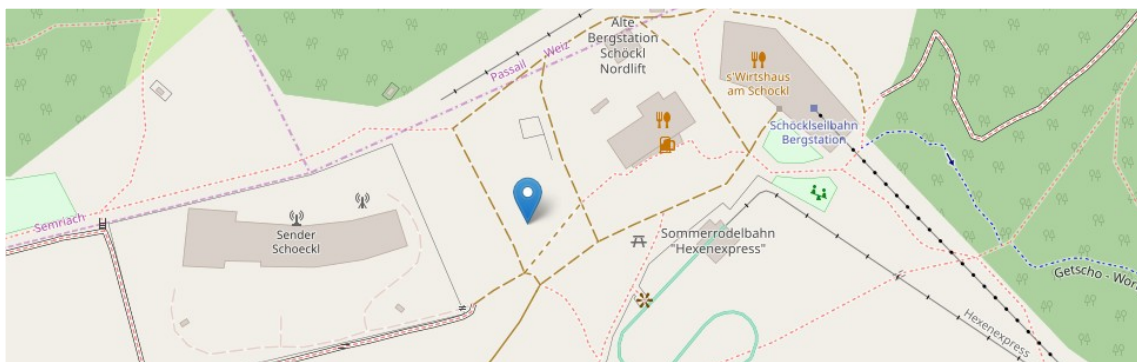
- Recherchiere im Internet die Funktionsweise der `.watchPosition()` Methode und nutze diese in deiner Webseite!
- HTML** **Freie Wahl der Inhalte**
- JS** **Geolocation Script (mit Error-Handler)**
- css** Ansprechende Gestaltung (Freies Design)
- Teste deine Seite bei deiner nächsten Bergwanderung bzw. Ski-Urlaub!**



Übung B: Online Karte auf der Berg-App

Öffne deinen Lösung von Übung A: Berg-App und füge einen Button hinzu, der die aktuelle Position des Device in einer Online-Karte anzeigt.

- Verwende OpenStreetMaps.org oder Google Maps
- HTML** **Iframe oder Link zum Mapanbieter**
- JS** **Geolocation**
- css** Ansprechende Gestaltung (Freies Design)



Lernhandout 20.1 OOP Einführung

Referenzcode **JSL201**
Technologien JavaScript
Dateinamen L201_OOP.odt, L201_OOP.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... arbeiten objektorientiert mit eigenen Objekten, Methoden und Eigenschaften.	I	II
... vereinbaren neue Objekte.	I	II
... definieren Eigenschaften für das neue Objekt.	I	II
... greifen auf Objekteigenschaften zu.	I	II
... definieren Methoden für das neue Objekt.	I	II
... rufen die neuen Objektmethoden auf.	I	II
... schreiben eine Objektmethode zur Errechnung von Zinseszinsen (vgl. Finanzmathematik).	I	II
... geben das Ergebnis einer Objektmethode mittels <code>return</code> zurück.	I	II
... analysieren die Auswirkungen auf Objektmethoden, wenn Objekteigenschaften geändert werden.	I	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Auch mit JavaScript ist **Objektorientiertes Programmieren** möglich, wenn auch nicht in dem Ausmaß wie es in anderen Programmiersprachen möglich ist. Problemlos lassen sich aber eigene Objekte erstellen, die eigene Methoden und Eigenschaften zulassen. Ebenfalls ist auch eine Objekterstellung mit einer Konstrukturfunktion möglich. Am einfachsten wird ein Einzelobjekt mit dem Wort `new` erzeugt.



Zuerst vereinbaren wir ein neues Objekt!

```
var sparbuch = new Object;
```



Dem neuen Objekt werden die Eigenschaften `.name`, `.zinssatz`, `.laufzeit` und `.kapital` zugewiesen.

```
sparbuch.name = "Hans Berger";  
sparbuch.zinssatz = 3.55;  
sparbuch.laufzeit = 10;  
sparbuch.kapital = 10000;
```



Auf die Objekteigenschaften kann problemlos zugegriffen werden:

```
console.log(sparbuch.kapital);
```



Methoden werden mit einer `function()` zugewiesen. Hier im Beispiel wird nach dem Methodenaufruf ein `alert`-Fenster mit den Eigenschaften angezeigt!

```
sparbuch.zinsmeldung = function() {  
    var ausgabe = "Kontoinhaber: " + sparbuch.name;  
    ausgabe += "Zinssatz: " + sparbuch.zinssatz;  
    alert (ausgabe);  
};
```



Der Aufruf der neuen Objektmethode verläuft wie gewohnt.

```
sparbuch.zinsmeldung();
```



Eine Methode zur Errechnung von Zinseszinsen sieht dann so aus – mit `return` wird das Rechenergebnis zurückgegeben!

```
sparbuch.rechne = function() {  
    var zw = Math.pow(1 + sparbuch.zinssatz / 100, sparbuch.laufzeit);  
    return sparbuch.kapital * zw;  
};
```



Die Methode wird mit `sparbuch.rechne()` ausgeführt. Natürlich können den Eigenschaften andere Werte zugewiesen werden!

```
sparbuch.laufzeit = 10;  
alert("Das Endkapital beträgt: " + sparbuch.rechne());
```

Lernhandout 20.2 OOP Literale

Referenzcode **JSL202**
Technologien JavaScript
Dateinamen L202_OOP.odt, L202_OOP.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen weitere Schreibweisen (Literale) in der Objektorientierten Programmierung von JavaScript.	KI	III
... verstehen die alternative Schreibweise (Key-Value-Prinzip mit Doppelpunkten) und der Trennung durch einfache Kommazeichen.	KI	II
... scripten ein Sparbuch-Objekt.	I	II
... verstehen den Zugriff auf eigene Eigenschaften innerhalb eines Objekt mittels <code>this</code> .	I	II
... definieren ein Array innerhalb eines neuen Objekts.	I	II
... greifen auf ein Array innerhalb eines neuen Objekts zu.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Es gibt noch eine zweite Schreibweise, wie man ein neues Objekt definieren kann. Dabei wird einfach eine Objektvariable vereinbart. `var obj = { ... }` Eigenschaften und Methoden werden nach dem Key-Value Prinzip erstellt. Der Key wird mit einem Doppelpunkt vom Value getrennt. Die Zuweisung wird mit einem einfachen Kommazeichen getrennt.



Objektliteral

Es wird eine Eigenschaft (`girokonto.nutzer`) und eine Methode (`girokonto.anzeige`) mit einer alternativen Schreibweise erstellt.

```
var girokonto = {
  nutzer : "Petra Müller",
  anzeige : function() {alert(girokonto.nutzer)}
}
girokonto.anzeige();
```



Unser Sparbuch Objekt kann also auch wie folgt gescrriptet werden:

```
var sparbuch = {
  name      : "Hans Berger",
  zinssatz  : 4,
  laufzeit  : 5,
  kapital   : 1200,
  zinsmeldung : function() {
    var ausgabe = "Kontoinhaber: " + sparbuch.name;
    ausgabe += "Zinssatz: " + sparbuch.zinssatz;
    alert (ausgabe);}
}
```



Wenn in einer Methode auf eine Eigenschaft zugegriffen werden soll, empfiehlt es sich das mit der Anweisung `this` zu tun. Die `rechne()` Methode muss innerhalb des eingezäunten `sparbuch` Objekt sein damit `this` darauf zugreifen kann!

```
rechne : function() {
  var zw = Math.pow(1 + this.zinssatz / 100, this.laufzeit);
  return this.kapital * zw;
}
```



Natürlich kann innerhalb eines Objekts auch ein Array definiert werden!

```
var sparbuch = {auszahlung : [300, 100, 550, 300]}
/* bzw. */
sparbuch.auszahlung = [300, 100, 550, 300];
```



Das Array wird dann wie gewohnt abgerufen:
Hier die Stelle `[2]` also der Wert 550

```
alert(sparbuch.auszahlung[2]);
```

Lernhandout 20.3 OOP Konstruktor

Referenzcode **JSI203**
Technologien JavaScript
Dateinamen L203_OOP.odt, L203_OOP.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... begreifen das Wesen eines Konstruktor und einer Konstruktorfunktion.	I	III
... vergleichen die Konstrukturfunktion von JavaScript mit dem Konstruktorkonzept einer anderen Programmiersprache.	K	III
... instanzieren eine Konstrukturfunktion mit dem Schlüsselwort new .	I	II
... verstehen die Wertübergabe bei Konstrukturfunktionen.	I	II
... nutzen Wertübergaben innerhalb eines neuen Objekts für Eigenschaften und oder Methoden.	I	II
... erweitern einen Konstruktor mit der .prototype Anweisung.	I	II
... erweitern und vererben ganze Konstruktoren um andere Konstrukturfunktionen.	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen



Man kann sich einen Konstruktor als eine Art Container vorstellen, der alle Eigenschaften und Methoden bei der Vereinbarung mit `new` an das neue Objekt übergibt. Im Gegensatz zu anderen Programmiersprachen, wo ein Konstruktor meist über eine Klasse definiert wird, benötigt man in JavaScript eine einfache Funktion → eine Konstruktorfunktion. Erst beim Instanzieren mit `new` wird die normale Funktion zu einer objektorientierten Konstrukturfunktion.

```
function GiroKonto() { this.form = "Giro" }
var meinKonto = new GiroKonto;
```



Konstrukturfunktion: Im Beispiel wird der Funktion ein String-Wert (`besitzer`) übergeben die der Eigenschaft `.name` zugewiesen wird.

```
function meinSparbuch(besitzer) {
  this.name = besitzer;
  this.kapital = 1200;

  this.meldung = function() {alert("Kontoinhaber: " + this.name);}
}
```



Vereinbart wird das neue Objekt (`sBuch1`) mit `new` und dem Namen der Konstrukturfunktion (`meinSparbuch`). Übergeben wurde der Besitzernamen als String. In Folge wird die Eigenschaft `.kapital` in der Console ausgegeben und die Methode `.meldung()` aufgerufen.

```
var sBuch1 = new meinSparbuch("Hans Berger");
console.log(sBuch1.kapital);
sBuch1.meldung();
```



Eine weitere Objekt-Vereinbarung ist problemlos möglich!

```
var sBuch2 = new meinSparbuch("Claudia Klein");
sBuch2.meldung();
```



Mit `.prototype` lässt sich der Konstruktor erweitern! Diese Anweisungen sollten außerhalb der Konstrukturfunktion definiert werden. Hier um die Eigenschaften `.art` und `.dauer`.

```
meinSparbuch.prototype.art = "Prämien sparen";
meinSparbuch.prototype.dauer = 24;
var sBuch3 = new meinSparbuch();
alert("Art des Sparbuch: " + sBuch3.art);
```



Mit `.prototype` lässt sich sogar eine Konstrukturfunktion um eine zusätzliche erweitern. Man spricht dann von Vererbung:
`meinSparbuch.prototype = new GiroKonto;`

Übungsblatt 20.3 Konstruktor

Referenzcode: **JSU203**
 Technologien: HTML, CSS, JavaScript
 Dateinamen: U203_OOP.odt, U203_OOP.pdf
 Konvergenz:

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
Übung A		
... schreiben eine Webseite, die den Anhalteweg (Reaktionsweg + Bremsweg) eines KFZs simuliert.	I	IV
... definieren eine Konstruktorfunktion.	I	II
... erarbeiten über die vorgegeben Formeln die notwendigen Objekteigenschaften und –methoden.	I	II
... stellen die Rechenergebnisse graphisch als Simulation dar.	I	II
... vereinbaren mind. vier KFZ-Simulationen mittels der Konstruktorfunktion.	I	II
... gestalten die Webseite anspruchsvoll und kreativ mit CSS.	IA	II
Übung B		
... erstellen eine Präsentation über weitere Feinheiten der OOP in JavaScript.	KI	IV
... recherchieren im Internet mögliche Themen wie: Verschachtelung von Objekliterals, Zugriffsrechte der Konstruktorfunktion, Anonyme Funktionen, Prototypen (Vererbung, Zugriffsrechte), Aggregation, Closures und Überschreiben ...	IK	III
... wählen ein Präsentationmedium und bearbeiten es.	I	II
... präsentieren ihre "Forschungsergebnisse".	SAI	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Übung A: Im Sinne der Verkehrserziehung sollten die Schüler_innen auch auf den Anhalteweg reflektieren.



Übung A: Anhalteweg

Der Anhalteweg eines KFZ setzt sich zusammen aus dem Reaktionsweg und dem Bremsweg – also vom Erkennen einer Gefahr bis zum Stillstand des KFZ.

$$\text{Reaktionsweg} = \frac{\text{Geschwindigkeit}}{10} \times 3$$

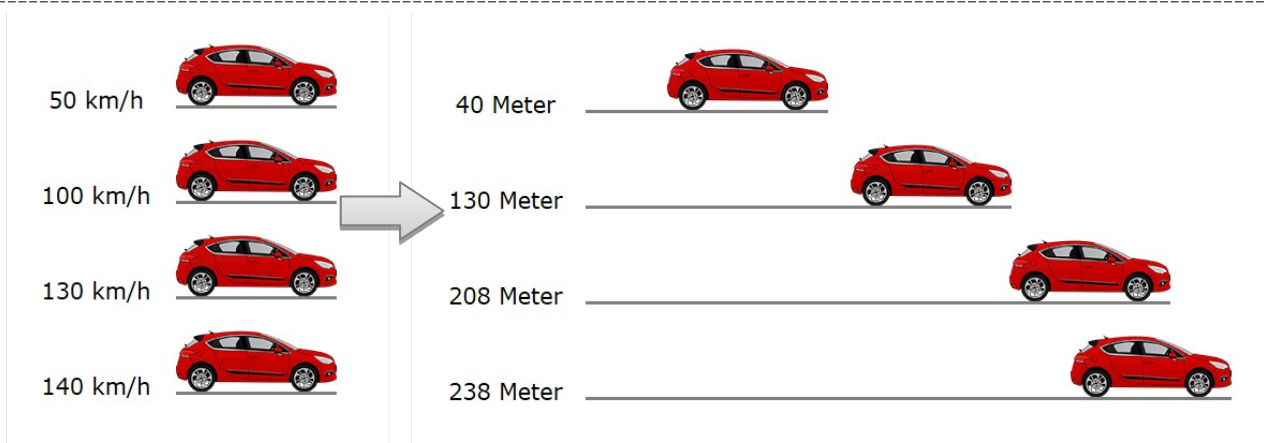
$$\text{Bremsweg} = \frac{\text{Geschwindigkeit}}{10} \times \frac{\text{Geschwindigkeit}}{10}$$

$$\text{Anhalteweg} = \text{Reaktionsweg} + \text{Bremsweg}$$

Scripte eine Webseite, die unterschiedliche Anhaltewege gegenüberstellt. Arbeite hierfür Objektorientiert mit einer Konstrukturfunktion mit dem Ziel, dass beliebig viele Objekte dimensioniert werden. Per Klick auf einen "Abbremsen-Button" soll der Anhalteweg errechnet und dementsprechend dargestellt werden.

- JS **Konstrukturfunktion, Eigenschaften und eine Methode**
- css Ansprechende Gestaltung (Freies Design)
- Vereinbarung/Dimensionierung mit `new` und der Geschwindigkeit als Übergabe. Die Konstrukturfunktion heißt `meinAuto()`**

```
var auto1 = new meinAuto(50); // ← auto1 fährt 50 km/h
var auto2 = new meinAuto(100); // ← auto2 fährt 100 km/h
var auto3 = new meinAuto(130); // ← auto3 fährt 130 km/h
var auto4 = new meinAuto(140); // ← auto4 fährt 140 km/h
```



Übung B: OOP Präsentation

Erstelle eine Präsentation über weitere Feinheit der Objektorientierten Programmierung in JavaScript.

- Themen:** Verschachtelung von Objekliteralen, Zugriffsrechte der Konstrukturfunktion, Anonyme Funktionen, Prototypen (Vererbung, Zugriffsrechte), Aggregation, Closures und Überschreiben ...
- Freie Wahl des Präsentationsmedium (z. B. eine Webseite, PowerPoint, ein Plakat, Mind-Map, am Whiteboard, einen Film usw.)**

Lernhandout 20.4 JSON

Referenzcode **JSI204**
Technologien JavaScript
Dateinamen L204_JSON.odt, L204_JSON.pdf
Konvergenz

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... kennen die Syntax von JSON (JavaScript Objekt Notation):	I	II
... erkennen den Nutzen von JSON zum Speichern bzw. zum Austausch von Objekten, Variablen und Array.	KI	III
... vergleichen JSON mit dem OOP Literal und erkennen die Unterschiede.	KI	III
... kennen die gültigen Datentypen die durch JSON unterstützt werden.	I	II
... verstehen ein Datumsobjekt in JSON als einen einfachen String.	KI	I
... wandeln ein Objekt, ein Array oder eine Variable mittels der <code>.stringify()</code> Methode in eine JSON Notation.	I	II
... parsen eine JSON Notation (JSON Text/String) mittels <code>.parse()</code> wieder zurück in ein JavaScript Objekt (bzw. Array oder Variable).	I	II

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

■

Mit JSON (**JavaScript Object Notation**) verfügt man über einen Syntax um Daten zu speichern, bzw. Daten auszutauschen. Diese Notation wandelt Objekte, Variablen und Arrays in einen reinen Text um, der dann z. B. mit `localStorage` gespeichert oder an ein PHP Skript übergeben werden kann. JSON ist ähnlich aufgebaut wie ein OOP Literal, nur werden hier die Keys immer unter doppelte Anführungszeichen gestellt.

```
{ "Name" : "Karl König", "Alter" : 33 }
```



Folgende Datentypen werden durch JSON unterstützt.

String	{ "wohnort": "Graz" }
Zahlen	{ "PLZ": 8010 }
JSON Objekt	{ "Person": { "name": "Peter", "alter": 25 } }
Arrays	{ "Besucher": ["Karl", "Eva", "Friedrich"] }
Boolean	{ "zugriff": true }
JSON null	{ "akademischerGrad": null }



Gegenwärtig ist es nicht möglich eine Funktion oder ein Datum in eine JSON Notation zu wandeln. Möglich ist es aber, das Datum oder die Funktion als String-Typ anzugeben.

JS



JSON.stringify(Objekt) ;

Die Methode wandelt ein JavaScript Objekt (oder Array bzw. Variable) in die JSON Notation. Das Beispiel hat eine Konstruktorfunktion `mitglied()`. Ein `xxlGruppe` Objekt wird vereinbart – die Eigenschaften `.name` und `.pincode` wurden verändert. Im Anschluss wird das `xxlGruppe` mit `stringify` in einen JSON Text gewandelt.

```
function mitglied() {
    this.name = "",
    this.frei = [3, 6, 4, 22],
    this.pincode = 9999
}
var xxlGruppe = new mitglied;
    xxlGruppe.name = "Herbert Trinker";
    xxlGruppe.pincode = 3532;
var speichern = JSON.stringify(xxlGruppe);
console.log(speichern);
```

JS



JSON.parse(Objekt) ;

In die andere Richtung geht es mit `JSON.parse()` ;
Dieses konvertiert den JSON Text in ein JavaScript Objekt.

```
var eineGruppe = JSON.parse(speichern);
console.log(eineGruppe);
```

Lernhandout 21.1 Projekte

Referenzcode	JSL211
Technologien	HTML, CSS, JavaScript (Projektmanagement)
Dateinamen	L211_Projekte.odt, L211_Projekte.pdf
Konvergenz	

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... analysieren die Vorgaben für die Projektplanung.	SIK	III
... analysieren die Vorgaben für die Prozessdokumentation.	SIK	III
... analysieren die rechtlichen Aspekte für das Projektprodukt.	SIK	III

Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene
Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- *Von projektorientierter Unterricht kann dann gesprochen werden, wenn versucht wird, Probleme und Arbeitsanlässe aus der Lebenswelt der Schüler aufzugreifen, die Schüler an der Planung zur Realisierung und Auswertung zu beteiligen, ... und die Arbeitsergebnisse in geeigneter Weise zu präsentieren.*

Schaub Und Zenke, Wörterbuch Pädagogik, S. 513

Im Projektunterricht wird der Focus auf problemorientiertes Denken, praktisches Tun und eine realitätsbezogenen Erfassung der Wirklichkeit gesetzt *um die Trennung von Leben und Lernen sowie Denken und Handeln aufzuheben.* **Schaub Und Zenke, Wörterbuch Pädagogik, S. 513**

Projektphasen



Für die Dauer von Projekten soll ein Lernen verwirklicht werden ...

... durch das die zeitliche Zerstückelung in einzelne Unterrichtsstunden zugunsten eines verweilenden und konzentriert-intensiven Befassen mit einem Thema einer Sache aufgehoben wird;

... durch das die fachperspektivistische Abschottung zugunsten einer ganzheitlichen Sichtweise aufgehoben werden;

... durch das die in Schulfächern übliche distanzierte Problemlosigkeit der Auseinandersetzung mit der Wirklichkeit zugunsten einer von den Lernenden selbst ausgehenden - problemhaltigen Begegnung ersetzt wird;

... durch das den üblicherweise in ihrem Lernprozessen stark gelenkten Schülern ein möglichst über den gesamten Prozess hin selbstständiges Lernen gewährleistet wird. **Peterßen, Wilhelm H. Kleines Methoden-Lexikon, S. 238**

Projektplanung

Bevor die Programmierarbeit beginnt, soll eine kurze Planung des Projekts erstellt werden. Folgende Inhalte soll die Projektplanung beinhalten:



- Thema und Arbeitstitel des Projekts
- Beschreibung des Projekts
- Ziele des Projekts (aufgeteilt in Meilensteine → Grobziele → Feinziele)
- Aufgaben der Teammitglieder (bei Partner bzw. Gruppenarbeit)
- Beschreibung der Projektphasen (Einsatz der Technologien, Dauer)
- Aufstellung der benötigten Ressourcen (Software, Hardware, geschätzte Kosten in € usw.)
- Verwendung nach Projektfertigstellung (z. B. Veröffentlichung im Web, App, wirtschaftliche Verwertbarkeit, usw.)

Prozessdokumentation

Die Prozessdokumentation beschreibt die realen Gegebenheiten des Projekts. Das "Projekttagbuch" soll folgende Inhalte erfassen:



- Zeiterfassung bzw. Mitschrift der effektiven Arbeitszeit am Projekt. Das schließt jede Projektbesprechung, Schreibarbeit und Recherche sowie Arbeiten an Planung und Dokumentation mit ein.
- Aufstellung der verwendeten Mittel (Ressourcen inkl. Kosten in €)
- Soll-Ist Vergleich (Analyse der Projektplanung)
- Quellenverzeichnis (Literatur, Internetquellen, Bildquellen, usw.)
- Read-Me File mit technischen Details (insbesondere bei Versionsänderungen)
- How-To Dokument (Benutzerhandbuch)

Rechtliches

Für jedes Projekt müssen die rechtlichen Rahmenbedingungen beachtet werden. Insbesondere Datenschutz, Urheberrecht und Kennzeichnungspflicht.










- Sichtbare Kennzeichnung der Projektmitglieder (Name, Kontaktmöglichkeit, Impressum) auf dem Produkt. Zusätzlich auch in den `<meta>` Tags.
- Jedes fremde Werk mit Schöpfungstiefe (Bilder, Videos, Sounds, Text, Skripte usw.) ist durch das Urheberrecht geschützt. Verwende also nur externe Quellen die über die Creative Commons bzw. als 'frei Verwendbar' gekennzeichnet sind. Auf jeden Fall sollen externe Quellen mit Link, Name des Urhebers und einem Datum gekennzeichnet werden.
- Datenschutz: Wenn personenbezogene Daten gespeichert und/oder verarbeitet werden (z. B. in einem Formular) muss eine Datenschutzerklärung vorliegen. Diese beschreibt was mit den Daten passiert und welche Möglichkeiten es zur Einsicht und Löschung der Daten gibt. **!!! Eine Speicherung als Cookie und/oder localStorage benötigt immer die Erlaubnis des User !!!**

Übungsblatt 21.1 Projekte

Referenzcode: **JSU211**

Technologien: HTML, CSS, JavaScript




Dateinamen: U211_Projekte.odt, U211_Projekte.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... beachten die Vorgaben für die Projektplanung, wie z. B. Projektbeschreibung, Ziele, Projektphasen, Ressourcen usw. und erstellen eine Projektplanung.	SIK	IV
... beachten die Vorgaben für die Prozessdokumentation, wie z. B. Zeiterfassung, Soll-Ist Vergleich, Quellenverzeichnis usw. und erstellen eine Projektplanung.	SIK	IV
... berücksichtigen die rechtlichen Aspekte bei der Projektarbeit, insbesondere Datenschutz, Urheberrecht und Kennzeichnungspflicht.	SI	III
... reflektieren auf die rechtlichen Aspekte (Datenschutz, Urheberrecht und Kennzeichnungspflicht) und deduzieren eine Verwendbarkeit für die Praxis.	SI	III
Projekt A: Kryptografie		
Projekt B: Alphanumerisches Sortieren		
Projekt C: Ampelanlage		
Projekt D: Triangulation		
Projekt E: Koordinatensystem und Kurvendiskussion		
Projekt F: Suchfunktion		
Projekt G: Navigation mit Sitemap		
Projekt H: Freies Projekt		




Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Projekte sind einfache und offene Vorgaben die nach Schwierigkeitsgrad der Realisation gekennzeichnet sind.  leicht  mittel  schwer
- Natürlich ist es wünschenswert, wenn ein/e Schüler/in sich für das Projekt H: Freies Projekt entscheidet und eine eigene Projektidee entwickelt.

Für alle Projekte gilt das Prinzip des freien Design für das HTML Layout und die CSS Anweisungen. Erweitere dein JavaScript um Feinheiten, wie z. B. Eingabeüberprüfungen oder Fehlerhandling.

-  Leicht
-  Mittel
-  Schwer



Projekt A: Kryptografie

Schreibe ein Script für eine cleveren Kryptografie. Ein String bzw. Objekt soll verschlüsselt und wieder entschlüsselt werden.



Projekt B: Alphanumerisches Sortieren

Scripte eine Funktion zum alphabetischen Sortieren von Wörtern bzw. einer Tabelle.



Projekt C: Ampelanlage

Simuliere mit JavaScript, HTML und CSS eine Straßenkreuzung mit einer Ampelanlage. Erweitere die Kreuzung um Abbiegestreifen, Fussgängerstreifen, Fahrradstreifen usw.



Projekt D: Triangulation

Erstelle eine umfangreiche Webseite zum Thema Dreiecke und Pyramiden. Scripte dafür alles von Pythagoras bis zu den Winkelfunktionen. Erweitere es um graphische Darstellungen.

- HTML **Canvas bzw. SVG für die graphische Darstellung**
- JS **Objektorientiertes Programmieren**



Projekt E: Koordinatensystem und Kurvendiskussion

Erstelle eine Website für eine Kurvendiskussion. Funktionsgraph, Nullstellen, Wendepunkt und Extremwerte sollen in einem Koordinatensystem dargestellt werden.



Projekt F: Suchfunktion

Entwickle eine clevere Such-Funktion für HTML Seiten.



Projekt G: Navigation mit Sitemap

Scripte ein cooles Navigationslayout für ein Website-Projekt. Überrasche durch smarte Animationen und einer automatisch generierten Sitemap.



Projekt H: Freies Projekt








Hast du eine eigene Idee? Dann mach es zu deinem Projekt – alles ist erlaubt!

Übungsblatt 21.2 Projekte Wirtschaft

Referenzcode JSU212

Technologien HTML, CSS, JavaScript




Dateinamen U212_Projekte.odt, U212_Projekte.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... beachten die Vorgaben für die Projektplanung, wie z. B. Projektbeschreibung, Ziele, Projektphasen, Ressourcen usw. und erstellen eine Projektplanung.	SIK	IV
... beachten die Vorgaben für die Prozessdokumentation, wie z. B. Zeiterfassung, Soll-Ist Vergleich, Quellenverzeichnis usw. und erstellen eine Projektplanung.	SIK	IV
... berücksichtigen die rechtlichen Aspekte bei der Projektarbeit, insbesondere Datenschutz, Urheberrecht und Kennzeichnungspflicht.	SI	III
... reflektieren auf die rechtlichen Aspekte (Datenschutz, Urheberrecht und Kennzeichnungspflicht) und deduzieren eine Verwendbarkeit für die Praxis.	SI	III
Projekt A: Bezugs- und Absatzkalkulation		
Projekt B: Direct Costing		
Projekt C: Anlageverzeichnis		
Projekt D: Rechnungsgenerator		
Projekt E: Einkommensteuer		
Projekt F: Lohn- und Gehaltsrechnung		
Projekt G: Betriebliche Kennzahlen		
Projekt H: Freies Projekt		




Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Projekte sind einfache und offene Vorgaben die nach Schwierigkeitsgrad der Realisation gekennzeichnet sind.  leicht  mittel  schwer
- Natürlich ist es wünschenswert, wenn ein/e Schüler/in sich für das Projekt H: Freies Projekt entscheidet und eine eigene Projektidee entwickelt.
- Die vorgeschlagenen Projekte orientieren sich fächerübergreifend (BWL und Rechnungswesen) am Lehrplan von kaufmännischen Schulen (z. B. HASCH, HAK, HLW usw.).

Für alle Projekte gilt das Prinzip des freien Design für das HTML Layout und die CSS Anweisungen. Erweitere dein JavaScript um Feinheiten, wie z. B. Eingabeüberprüfungen oder Fehlerhandling.

 Leicht
 Mittel
 Schwer



Projekt A: Bezugs- und Absatzkalkulation

Erstelle eine Webseite mit einer progressiven und retrograden Bezugs- und Absatzkalkulation.



Projekt B: Direct Costing

Scripte eine Webseite für eine einstufige Deckungsbeitragsrechnung und kombiniere diese mit einer Break-Even Analyse.



Projekt C: Anlageverzeichnis

Die Anlagegüter sollen erfasst und in Folge soll Buchwert und AfA errechnet werden. Verbinde das Anlageverzeichnis auch mit einem Buchungssatzgenerator, der alle AfA Buchungen ausgibt.



Projekt D: Rechnungsgenerator

Die Rechnungsbestandteile werden über ein Eingabeformular ermittelt und in Folge wird eine automatische USt-Rechnung ausgegeben.



Projekt E: Einkommensteuer

Nach der Eingabe der Einkünfte und aller weiteren steuerrelevanten Posten, soll das JavaScript die Höhe der Einkommenssteuer ermitteln.



Projekt F: Lohn- und Gehaltsrechnung

Scripte ein Tool für die automatisierte Lohn- und Gehaltsrechnung.



Projekt G: Betriebliche Kennzahlen

Erstelle eine Webseite, die nach Eingabe von Unternehmensdaten aus dem Rechnungswesen und der Kostenrechnung unterschiedliche betriebliche Kennzahlen ausrechnet und übersichtlich (wenn möglich sogar graphisch) ausgibt. Beispiele:

- Gesamt- und Eigenkapitalrentabilität, ROI, Cash Flow, Anlagendeckung, Verschuldungsgrad und Kapitalbindung, Kapitalumschlagshäufigkeit uvm.



Projekt H: Freies Projekt








Hast du eine eigene Idee? Dann mach es zu deinem Projekt – alles ist erlaubt!

Übungsblatt 21.3 Projekte Entertainment

Referenzcode JSU213

Technologien HTML, CSS, JavaScript




Dateinamen U213_Projekte.odt, U213_Projekte.pdf

Feinziele	Zielart	Taxonomie
Die Studierenden ...		
... beachten die Vorgaben für die Projektplanung, wie z. B. Projektbeschreibung, Ziele, Projektphasen, Ressourcen usw. und erstellen eine Projektplanung.	SIK	IV
... beachten die Vorgaben für die Prozessdokumentation, wie z. B. Zeiterfassung, Soll-Ist Vergleich, Quellenverzeichnis usw. und erstellen eine Projektplanung.	SIK	IV
... berücksichtigen die rechtlichen Aspekte bei der Projektarbeit, insbesondere Datenschutz, Urheberrecht und Kennzeichnungspflicht.	SI	III
... reflektieren auf die rechtlichen Aspekte (Datenschutz, Urheberrecht und Kennzeichnungspflicht) und deduzieren eine Verwendbarkeit für die Praxis.	SI	III
Projekt A: Medienplayer		
Projekt B: Schiffe versenken		
Projekt C: Mastermind		
Projekt D: Würfelpoker		
Projekt E: Wissens-Quiz		
Projekt F: Ein Brettspiel		
Projekt G: Ein Kartenspiel		
Projekt H: Freies Projekt		




Zielarten: A ... Affektiv, K ... Kognitiv, P ... Psychomotorisch, S ... Selbstkompetenz, I ... Inhaltsebene

Taxonomien: I = verstehen, II = anwenden, III = analysieren, IV = entwickeln

Anmerkungen

- Die Projekte sind einfache und offene Vorgaben die nach Schwierigkeitsgrad der Realisation gekennzeichnet sind.  leicht  mittel  schwer
- Natürlich ist es wünschenswert, wenn ein|e Schüler|in sich für das Projekt H: Freies Projekt entscheidet und eine eigene Projektidee entwickelt.
- Es sollte ein kritischer Umgang mit Glücksspielen (Slot-Machines, Black Jack, usw.) berücksichtigt werden.

Für alle Projekte gilt das Prinzip des freien Design für das HTML Layout und die CSS Anweisungen. Erweitere dein JavaScript um Feinheiten, wie z. B. Eingabeüberprüfungen oder Fehlerhandling.

 Leicht
 Mittel
 Schwer



Projekt A: Medienplayer

Erstelle einen Medienplayer auf einer Webseite. Der Medienplayer soll die gängigen Video und Sound-Formate abspielen können.



Projekt B: Schiffe versenken

Scripte eine Website mit dem beliebten "Koordinatenspiel" Schiffe versenken. Eine KI soll der Computer-Gegner sein – überlege dir auch eine Möglichkeit um Mensch gegen Mensch zu spielen.



Projekt C: Mastermind

Das beliebte "Code-Knacker" Spiel mit den fünf Farben.



Projekt D: Würfelpoker

Würfelpoker bzw. Yatzy – die Herausforderung ist die Punkteliste im digitalen "Spielblock".



Projekt E: Wissens-Quiz

Ein spannendes Wissens-Quiz mit Fragen und Antworten.

- Die Fragen und Antworten sollen als JSON, externes JS oder XML gespeichert werden.



Projekt F: Ein Brettspiel

Checkers, Chess und Mensch-Ärgere-Dich-Nicht. Jedes 2-D-Brettspiel lässt sich im Browser realisieren. Suche dir eines aus, beachte aber die Urheberrechte des Spieles.



Projekt G: Ein Kartenspiel

Erstelle eine eigene Spielkarten API und realisiere sie in einem beliebigen Kartenspiel (z. B. Schnapsen, Black-Jack, Rummy odgl.)



Projekt H: Freies Projekt

Hast du eine eigene Idee für ein spannendes Spiel oder eine coole Entertainment-Anwendung? Dann mach es zu deinem Projekt – alles ist erlaubt!

Pädagogischer Beipackzettel

Die Pädagogik ist wahrscheinlich die kontroversiellste Wissenschaft unter den Geisteswissenschaften. Drei Lehrer_innen habe meist auch drei Didaktiken. Bestandteile von unterschiedlichen Reformpädagogen werden eklektisch zu einem individuellen Gesamtkonzept zusammengeführt – man nimmt sich was man braucht und nutzt was Sinn ergibt. Konsens wird meist von oben befohlen – der Kodex der Schulgesetze 'overruled' die eigene Fachdidaktik. Das Motto soll lauten: "Lasst' uns unsere pädagogische Freiheit bewahren indem wir unser Hirn benutzen und den kritischen Blick auf die vergangene, gegenwärtige und zukünftige Gesellschaft bewahren". Dabei ist die Pädagogik (so denn sie als Erziehungswissenschaft verstanden werden will) eine der ältesten Wissenschaften der Menschheit – neben der Suche nach Gott und Nahrung ist sie jener Beitrag zum Mensch-werden der erst unsere Kultur geschaffen hat.

Als Sokrates im antiken Griechenland zum Tode für "die Verführung der Jugend" gerichtet wurde, ging er nicht nur als Philosoph in die Geschichte ein, sondern vielmehr als der erste Pädagoge der für seine Auffassung starb. Dabei muss aber Pädagogik nicht dramatisch, schrill und laut sein. Das pädagogische Genie kann ebenso introvertiert, wortkarg, zurückhaltend, verschlossen und zugeknöpft sein. Der Genius muss sich nicht immer durch Marktschreierei hervortun – er oder sie kann auch das Höchste in einem Spannungsfeld zwischen Stille und Getöse schaffen und wird vielleicht Zeit seines_ihres Lebens niemals die Würdigung erfahren, die ihm oder ihr zusteht. Die Frage "Was ist ein guter Lehrer?" füllt genauso viele Bücher wie es die Frage "Was ist Philosophie?" tut.

Dieser pädagogische Beipackzettel war ursprünglich als Textbeigabe zu den Risiken und Nebenwirkungen von Methoden gedacht. Ich will ihn aber dazu nutzen, um meine pädagogischen Meinungen der Unterrichtskonzeption beizulegen. Es sind Meinungen die nicht empirisch abgesichert sind, deshalb spannt sich über alle folgenden Worte ein "meines Erachtens" ...

Gedanken zur Leistungsbewertung

Die Bewertung des_der Schüler_in ist keine interpersonelle sondern soll sich, getreu einem kognitivistischen Gesamtkonzept an der Prozessleistung des Lernerfolgs orientieren. Da die Leistungsbewertung im Vergleich zur Beurteilung immanent und ständig erfolgt, ist dem Lernprodukt nicht der gleiche Stellenwert beizumessen, wie es das punktuelle Lernprodukt einer Prüfungssituation nach sich zieht. So gesehen ist es klar, dass die Portfolio-Methode schlussendlich zur Beurteilung herangezogen werden kann – und m. E. auch in der Praxis passieren sollte. Die punktuelle Prüfung des Gelernten mag vielleicht einen Anreiz zur zweiten (bzw. mehrmaligen) Beschäftigung mit der Materie nach sich ziehen – ist aber m. E. nur sinnvoll in einem mündlichen Prüfungsgespräch, wo eigentlich nie die tatsächliche Festigung der Materie als eher die Fähigkeit Antworten auf Fragen zu geben geprüft wird. Ich empfehle, Kommentare im Quellcode des JavaScript Dokuments ebenfalls als prozessorientierte Leistung zu werten, wie das logisch-zielorientierte Denken – welches sich in der Klarheit eines EDV-Codes (in welcher Form auch immer) darstellt. Ein bedeutender prozessorientierter Akt im Webdesign liegt definitiv in der Formulierung von visuellen Vorgaben als auditiven Text (Code) und umgekehrt – die Antizipation des Codes zu seiner visuellen Darstellung. Man merkt, dass hier also linke und rechte Gehirnhälften gleichermaßen gefordert sind und sich wechselseitig bedingen.

Warum Blendet Learning?

Im Informatikunterricht und seinen verwandtschaftlich nahen Teilgebieten – ist der Begriff es blendet Learning wohl eher als ein leerer zu betrachten. Es führt zu einer paradoxen Unterrichtsgestaltung, sich selbst durch die Methode darzustellen – um so im Metabereich der Unterrichtspraxis auf das konzeptionell Höhere zurückzugreifen. Wie soll nun also das Begehren nach digitalen Kompetenzen und der Nutzung selbiger Bezugssysteme erfolgen? Die Loslösung erscheint für mich als das adäquate Mittel um das im System innewohnende durch seine Vielseitigkeit als Prinzip aufzuführen. Die Substanz wird zum erkennenden Moment und entzieht sich einem geblendeten Lernen durch die Vielschichtigkeit ihrer eigenen Möglichkeiten. Weil nun das Rudimentäre zum Absoluten gesetzt wird, kann eine Begleiterscheinung eines Lernens nur durch die methodische Unterschiedlichkeit erklärt werden. Ich will wieder exemplarisch werden: Das Erlernen des Webdesigns kann sowohl über die vergleichbar simple Plattform www.w3schools.com angeeignet werden – als auch über die als höher einzustufende Plattform selhtml.org bzw. den offiziellen Browser-Spezifikationsseiten. Die große (ultimative) Aufgabe dieser didaktischen Konzeption liegt im Mittelweg zwischen dem primitiv Einfachen und dem komplexen Vollständigen. Es liegt im Endeffekt an der Motivation des_der Schüler_in.

Sozialformen in der IKT

Aus pädagogischer Sicht bietet das weiterführende Tutorial keinerlei Anreize zu einem konsequenten Wechsel der Sozialformen. Da nun, wie es in den meisten IT dominierten Fächern der Fall ist, das Lernen von selbigen Inhalten die sich selbst bedingen, ein Lernen im isolierten Zustand – ein Lernen im reinen Ich nach-sich-zieht – kann kein großer Wechsel prognostiziert werden. In allen Fällen, selbst wenn von kollaborativen Lernen gesprochen wird, ist reine Bildschirmarbeit weiterhin eine Isolation vom Wir-Gefühl aber: a) hochgradig potent um kognitive Ziele umzusetzen und b) eine Bereicherung des Ich-Gefühls solange sie als Selbstkompetenz verstanden wird und nicht einem kompetitiv-pädagogischen Szenario ausgesetzt wird. Man möchte meinen, es wird das Ich-Genie im Zusammenhang seines eigenem Denken konstruiert, dass sowohl dem Seienden seine Erwidierung Tribut leistet wie es im gemeinschaftlichen der kollektiven Erkenntnis jenes Moment liefert, dass noch fehlt. Im besten Fall könnte man ein Szenario der Bildschirmknappheit schaffen – man könnte den_die Schüler_in auffordern zu zweit an einer Konsole zu arbeiten – es ist aber nur eine Augenauswischerei. Das Genie bleibt auch in diesem Bereich losgelöst von der Menge und erfährt damit kein besonders pädagogisches Moment – JavaScript ist nun mal nicht die Lösung aller Probleme des pädagogischen Alltags.

Der konsequente Lehrplanbezug

Die genauere Darstellung eines Lehrplanbezuges ist im gegenwärtigen Zustand eher komplex. Natürlich würde ich eine Lehrstoffverteilung auf ein oder das nächste Semester hoch graduieren. Es entzieht sich jeglicher Basis – ein vorurteilendes Argument abzuliefern – weil nun der Lehrplanbezug in seinem Wesen mannigfaltig ist – und somit weitschweifig. Es gibt keine fundamentale Erkenntnis, die den Bezug in seiner endogenen Phase erkennt und so sein Wesen jeglicher esoterischen Basis entzieht. Ich habe schon einige gedankliche Versuchungen angestellt – um genau das nicht Obskure in den Denkprozess als eine Elimination näher einfließen zu lassen. Die ganze Geschichte entbindet sich jeglicher Darstellungs-Objektivierung umso dem Gesamten seine Weisungskraft näher zu legen – wo wir uns fragen, warum auch nicht?

Die kritische Sozialformierung

Der soziale Zusammenhalt im Schulischen ist definitiv nicht nur ein Fremdgesteuerter. Im Rahmen der Bildschirmarbeit, obgleich es in der allgemeinen Informatik oder deduktiv auf ein sich der informatisch gegenstandsbezogenen Unterrichtsdimension wie z. B. der Mathematik mit Hilfe von DO Kopfrechnen oder GeoGera zur Darstellung erkenntnisrelevanter mathematischen und damit in Folge kognitiv/logischen Gedanken nach sich zieht. Tatsache ist, das jegliche Bildschirmarbeit immer eine Arbeit auf isolierter Basis ist. Auch die 'geblendete' Vorstellung einer IT-Tätigkeit über kollaborativen oder sozialen Netzwerken ist definitiv nicht ein Ersatz zur Erlangung von kommunikativen oder sozialen Kompetenzen. Die Übung des logischen Denkens – der kommunikativen Formen und seiner Aktion-Reaktion, mag vielleicht simuliert sein und dem_der Lernenden ein Moment des Selbstbewusstseins verleihen, das sich nicht durch die konkrete als mehr durch die indirekte Konsequenz, abbildet und einstellt. Natürlich lassen sich so in der Sozialform der Einzelarbeit, kognitive Fehler in Form eines Syntax Errors oder eines Syntax Errors in Zeile xy dar niederlegen – sie sind aber noch lange nicht die Fehlleistung des interaktiven in jedem menschlichen Prozess. Das isolierte Lernen zeigt paradoxerweise seine volle Wirksamkeit im perfekt-pädagogischen Wirken, dem geplanten und programmierten Unterricht, der endgültigen Absage von unlogischen Fragen und der klaren Antwort darauf: dem Syntax Error. Seine Antwort ist trivial aber ehrlich.

Jenseits von Gut und Sehr Gut

Gibt sich der Lehrer selbst eine Note, wenn er einen seiner Schüler benotet? Betritt man das unscheinbare Gebäude nah am Stadtrand, möchte man nicht unbedingt davon ausgehen, das hier ein pädagogischer Prozess geschieht. Natürlich, man sieht Lernprodukte - besonders große Plakate, die einen Themenkomplex mit vielen Schlagwörtern in den unterschiedlichen Schriftarten herzeigen. Das Moment der Beachtung liegt aber nicht auf dem Inhalt, sondern eher die originäre Einzigartigkeit der vielen Buchstaben - die noch immer seriell und parallel verlaufen und zusammengenommen einen Sinn ergeben - zumindest auf den ersten Blick hin.

Nach außen gibt sich das Bildungsinstitut zurecht gönnerhaft. Alle Wesensmerkmale einer Heterotopie leuchten auf - Öffnungszeiten, örtliche Begrenzungen, Stockwerke, neue Medien, ein Fahrstuhl, ein Schullogo mit seinen vielen Corporate-Produkten im Anschlag und natürlich eine Schulordnung die nicht veröffentlicht wurde (und wenn, dann gut versteckt). Hat man doch das Schema F immer im Hinterkopf und reproduziert es sich nicht selbst von Schule zu Schule wieder aufs neue obgleich doch tausendmal schon gesehen. Man mag es einfach, wenn die Schuhe in Reih und Glied stehen. Das wusste schon Herbert Grönemeyer zu besingen. Und dass, architektonisch das Lehrerzimmer, das Sekretariat und die Direktion einer Schule doch immer in greifbarer Nähe sein sollten - geclustert auf den wenigen Quadratmetern die noch nicht von den Schüler_innen okupiert wurden, ist doch wunderschön, oder?

Ist es nicht Konvention, dass ein Schüler zuerst den vielbeschäftigten Verwaltungsapparat passieren muss um ein Gespräch mit der Direktion zu führen? Nicht so in dieser Bildungseinrichtung - hier, folgt man den durch die eigene Nase bestimmten logischen Weg muss man zuerst den Prinzipal überwinden um zum Sekretariat zu gelangen - welches sich durch zwei Charakterzüge auszeichnet. 1. Es scheint die gute Seele und 2. der einzige Fixpunkt in einem sich ständig rotierenden Universum zu sein. Alle anderen einfachen Lohnempfänger_innen sitzen und brüten immer an den unterschiedlichsten Stellen im Gebäude welches, wäre es konsequent zu Ende gedacht, ein Labyrinth zwischen Räumen, Gängen und Stiegen und so will es auch der Zeitgeist - so diktiert es die Avantgarde der Wirtschaft.

Besprechungsraum A, B und C - am besten man bucht sie, wenn man sie braucht - das Klassenzimmer on demand. Die Computerräume sind immer offen und immer zugänglich - wieso auch nicht, schließlich verbinden sie Meetingraum II mit einer spartanisch aber als eine auf den Lehrer-

vortrag ausgerichtete Unterweisungsräumeinheit. Die Computerarbeitsplätze sind so angeordnet, dass keiner während der Arbeit am PC zur Beamerleinwand blicken kann - so kann Informatikunterricht einfach nicht funktionieren bzw. pädagogisch gedacht kann nur so Informatikunterricht wirklich funktionieren - wenn der Info-Lehrer ausschließlich Hinterköpfe und Rückenansichten der Schüler_innen sieht, Rektalunterricht/Hektalunterricht in seiner Reinform.

Und im Vergleich zu seinem Pendant, zeichnet sich der gute Info-Unterricht durch die Schweisamkeit des Lehrers aus - Zeiten in denen der Lehrer vorne Linear den einzigen wahren Weg zum Resultat vorzeigt und die Schüler_innen diesen nach-klicken, sollte nur allein wegen einer progressiven und aufgeschlossenen Auseinandersetzung mit der Dynamik der Fachwissenschaft schon nicht gewagt werden. Naja, dieser Appell richtet sich definitiv nur an Medienpädagogen (in welcher Schulstufe und Form auch immer) welche eher auf das Produkt als nach dem Prozessprinzip hinarbeiten.

Die gänzliche Wahrheit des SchülerInnenprodukts ist ihre letzte Erscheinungsform und die dazu (eloquente) Präsentation. Und so muss sich der Medienpädagoge auch zurücklehnen und den Schüler_innen volle Freiheit in den unterschiedlichsten Applikationen gewähren. Der und die Schüler_in wird schon eine Lösung zum Problem finden - und mag die Lösung auch darin liegen, den Medienpädagogen um Hilfe zu bitten (eine ebenso legitime Problemlösung wie die Spracheingabe über die Google-App mit der passenden Tutorenlinkausgabe auf YouTube).

Und so soll auch der Informatikunterricht als Unterrichtsfach gar nicht Einzug finden in die Stundenpläne der Schüler_innen - gänzlich sinnlos (ausgenommen sind höhere Schulformen, aber darauf komme ich noch später zu sprechen). Die Einrichtung eines eigenen Unterrichtsfach "Informatik" in Primär und Sekundarstufe I sollte nicht nur nicht angedacht werden, sondern muss aus pädagogischer Sicht vermieden werden. Der IKT-Bezug zu jedem Kompetenzfeld allein reicht vollständig aus um den jungen Menschen die ach so notwendigen IT-Fähigkeiten mitzugeben - mehr noch, wird der junge Mensch nicht schon in den für seine Entwicklung so wichtigen Jahren affektiv geprägt, die sich durch die Selbstwirksamkeit und der eigenständigen Problemlösungskompetenz ergibt, geformt und kann jener auch im Sinne der klassischen elektronischen Datenverarbeitung sicher nicht sofort die tieferen Grundlagen der Materie erkennen oder vielleicht erfassen, so kann er sich doch als ein IT-Profi fühlen - was mehr Fundament für eine weiterführende Ausbildung bietet, als jedes noch so mühsam erworbene ECDL-Wissen in der Sekundarstufe I tut. Deshalb brauchen wir für den Medienpädagogen in der Sek I ein anderes Berufsleitbild als es die Werteeinheiten-Kolleginnen_en haben. Mag es auch noch so trist und langweilig klingen, aber EDV-Arbeit ist nun mal nicht vergleichbar mit Löwenzähmung oder Extremsportarten.

Der gute Medienpädagoge lässt die Schüler_innen arbeiten - überwacht sie nicht oder quält sie mit Ratschlägen - er_sie greift nicht zur Maus bzw. der Tastatur des Schülers sondern versucht mit der Sprache eine Problemlösungsstrategie mitzugeben. In Folge ist dann der Medienpädagoge (und bitte ich meine beiderlei Geschlecht) nicht mehr nur auf eine einzige Klasse kapriziert sondern hat Kontakt mit eigentlich allen Schülern aus allen Schulstufen. Er_Sie baut die Lehrer-Schüler-Beziehung auf die Affinität des Schülers auf. Fördert jene, die eine besondere IT Neigung besitzen.

Höre mir zu und klicke einfach das nächste mal auf genau jenen Button! Solche Simplifikationen die hinter jedem Tastendruck, jeder Mausbewegung, jedem Wischen am Display und [in ferner Zukunft] jedem Gedanken passiert, ein Grundlagenprozess einer Registersprache ist, in welcher die Information, das Input, die Eingabe einem Verarbeitungsprozess, einer Prozessorzuweisung, einer Kausalität unterworfen ist. Da sich im Kampf der vorherrschenden Logik sich die imperative Logik gegen die prädikative durchgesetzt hat, sind jene Befehlssätzen zum Dogma des Gegenwärtigen erwachsen. Nicht ohne uninteressanten Hintergrund für eine pädagogische Verwertbarkeit, da nun durch das Mittel "Logik" eine Kompetenzvielfalt vermittelt werden kann, die sich zwar am PC üben lässt aber vielmehr Auswirkungen auf eine zukünftige Berufung hat. Scriptet noch der junge Schüler eine Softwarelösung, so schreibt er in Wahrheit eine Konklusion von Befehlssätzen, die angelehnt einer Progressionslogik entsprechen sollen, nein müssen.

Die Dunkelheit ist die Wahrheit - das Licht, ist Multiplikation von 1 und 0. Und will es uns auch noch so romantisch vorkommen, wenn wir den jungen Menschen die Fähigkeit einen Algorithmus zu formulieren vermitteln, so bleibt die Reflexion auf seine Kompromissfähigkeiten nicht unverhüllt - dafür schreibe ich! Es ist aber keine Wahrheit, die an einem Tresen in einer zwielichtigen Bar den Eindruck hinterlässt: "Ich leide an der Erkenntnis der absoluten Wahrheit" - "Ich leide an meinem Wissen!". Es ist keine Wahrheit welche die Seele leiden lässt.

Zu aller Anfang, ermittelt der junge Info-Schüler was als Quelle wahr bzw. fake ist. Das kritische Bewusstsein auf ein all-gegenwärtiges und voll-verfügbares Aussagenkonkluemat und der Versuch die Wahrheit aus den vielen Text, Bildern, bewegten Bildern und gesprochen Text heraus-zu-destillieren. Keine Aufgabe der Informatik! Die Informatik kennt nur die booleschen Begriffe Wahr und Falsch und nicht Wahr und Fake. Da nun aber die vielen Schüler_innen Produkte im Lauf der Zeit darauf ausgelegt sind, der nachfolgenden Schüler_innen Generation eine einfachere Schulzeit zu beschern, wird die kritische Auseinandersetzung mit der Lehrstoffverteilung eines Informatik-Unterrichts immer diffiziler. Klassische und konservative Informatik-Lehrer_innen (wie ich auch einer bin) portionieren den Lehrstoff über drei große Entitäten (den halbherzige Blick auf Lehrpläne des BMUKK bzw. den Deskriptoren von BIFI können der 'Didaktik der Informatik' verpflichtete Lehrerinnen unterlassen, weil ...) Betriebssystem, Softwareentwicklung und Spezifikation die einzig möglichen Ansätze sind. Das Betriebssystem unterteilt sich in Hardware und Software - mit anderen Worten, der schöne Power-Point Vortrag über die Festplatte, den Hauptspeicher über die Peripherie. Gesetze des Betriebssystems sind die Gesetze der plattformunabhängigen Übertragbarkeit auf das RGB-Lichtgeflecht des Monitors, welches die Basis für alles schafft.

Wie schon gehabt, übernehmen Sie einfach die Vorgaben ihres gratis Schulbuchs oder suchen Sie tatsächlich übereinstimmende Wahrheiten der Eingabemöglichkeiten (Maus, Tastatur oder Sprache) - das Betriebssystem entspricht dem Theater des Schauspiels, welches man zu interpretieren und im besten Fall zu beherrschen versucht. Alle großen Betriebssysteme haben ihre Wahrheiten, die Schnittpunkte liefern. Selbst Windows öffnet über Ausführen --> cmd seine Konsole, von den Unix-Derivaten (Linux, OSx) kennen wir es sowieso - damit jedes Betriebssystem weit in seine eigene Vergangenheit greift und API's bereitstellt, die schon immer da waren. Für den Schüler sicher ein Faszinationsmoment. Der zweite Bereich: die Softwareentwicklung oder Programmierung bietet in rein kognitiver Hinsicht so viele Kompetenzen, die niemals nie im Mathematik-Unterricht allein vermittelt werden können. In der Scholastischen Mathematik gibt es nur eine Lösung für ein Problem die brav linear gelernt werden soll. Sie firmieren unter dem Betreff "Lösungsweg" - und kommt ein Schüler zwar zum richtigen Ergebnis, aber nicht unter Einbehaltung des vorgegebenen Lösungsweg, kann dieser schon mal ein "Nicht-Genügend" ernten. In der Softwareentwicklung wäre eine solche Bewertung des Schüler_innen-Produkts eher kompliziert, weil es in der Softwareentwicklung viele Lösungswege gibt. Ist eine iterative Lösung eines Problems gleichwertig zu einer verlangten rekursiven Lösung, nur allein weil die Rekursion als elegant gilt? Ich sage ja, solange der Code geparsed wird. Solange das Ergebnis der Anforderung entspricht - tut sie es nicht, so entsteht das pädagogische Moment des Sinn stiftenden Dialogs, welches zwischen der Trinität Lehrer - System - Schüler auf das Plenum der Schüler erweitert werden sollte. In dritter Ebene findet man die Spezifikation - didaktische vorherrschende Literatur spricht hier oft von einer Bezugswissenschaft, deren die Informatik verpflichtet ist.

In einer Handelsakademie gelten andere Anforderungen wie sie für ein Gymnasium oder eine höhere technische Lehranstalt gelten. So sind in einem, die Nutzung von SAP, im anderen die Kenntnis von Prolog und im nächsten AutoCAD Kenntnisse von Nöten. Die Spezifikation ist immer eine Reduktion auf die intuitive Führung und das korrekte Nutzungspotential reduziert. Sie will definitiv nicht als einfach reduziert verstanden werden - sie ist aber determiniert und kann dadurch nur im Moment seine volle Entfaltung erfahren. So passiert es, das ein Architekt über die Möglichkeiten von AutoCAD höher sprechen kann, als es von einem Informatikprofessors höchster Güte möglich ist. Kein Problem - in einer Handelsakademie kennt der Wirtschaftsinformatiker mehrere werksseitige Funktionen von Excel - der Wirtschaftsinformatiker aber kennt die Möglichkeit eigene Funktionen zu scripten. SAP und Excel - alle Office-Produkte - das ist die Spezifikation der Ausbil-

dung. Informatik dann als Hilfswissenschaft zu verstehen ist die didaktische Aufgabe des Info-Professors der Sekundarstufe II.

Mit Imperativen [programmieren] lernen

Wer die Informatik verstehen möchte, muss sich zu aller erst etwas Bewusst machen. Alles was du weißt, ist Oberfläche - Vergänglichkeit - Flüchtig und immer nur ein Teilbereich vom Ganzen. Im Hintergrund - den man definitiv vertiefen kann - finden unzählige Translationsprozesse statt. Das Eine wird in das komplex-Tiefere übersetzt. Die Tiefe vertieft sich selbst weiter in die Überleitung zum gänzlichen Prozess.

Die Logik des Janus - vielleicht waren da einfach in seiner Jugend zu viele Imperative die er zu befolgen hatte. Jetzt aber, lieber junger Janus darfst du die Befehle geben - jetzt darfst du mit Imperativen programmieren!

Ich begleite den jungen Janus durch die imperative Programmierung und erkläre ihm die Möglichkeiten und Grenzen dieser Fähigkeit. Noch bevor ein paar Befehle an den Prozessor gesendet werden, sollte Janus sich über die Professionalität seines Handelns bewusst werden. Nun ist im-Pro* nicht "Just Cause 4" oder "Hitman 2". Durch das Schreiben von hochwertigen Code wird nicht nur der Nimbus der Arbeit vermittelt, nein nein das Ludische wird zur Gänze vom Screen entfernt und schwarze Pixel werden wieder Schwarz** - subjektiv betrachtet natürlich. Überhaupt, die Pixel kommen erst wirklich wieder zurück - dann wenn Janus ein Point-Set aus programmiert oder gar, Punkte über eine Gleichung der 3. Ordnung wandern lässt. Und dann sind sie alle wieder da - auch wenn es bereits 4000 über die Breite sind - aber mein Gott, der informatorische Pointillismus muss weiterleben - weiterleben durch dich, mein lieber Janus.

Allem Anfang liegt nicht immer nur ein Zauber inne. In der Pädagogik sind oft die Anfänge die größten Anstrengungen für den_die planenden_e Pädagogen_in und die Quelle aller Herausforderungen für den_die spontanen_e Pädagogen_in. So oder so - in der Informationspädagogik macht es keinen großen Unterschied ob nun das Ziel dem Ist-Zustand entspricht oder der Ist-Zustand das explizite Ziel darstellt. Im pädagogischen Prozess zählt wie so oft eigentlich nur das Jetzt-und-Hier. Und genau dieses hic-et-nunc ist auch das stärkste Argument für Lern-Ziel-Kontrollen, Prüfungen, Schularbeiten - alles was dem aufrichtigen Informatiker zu-wider sein sollte, da scripten und coden nichts spontanes oder momentanes ist, sondern auf Gründlichkeit, Treue zur Logik und damit auf Immanenz ausgerichtet ist. Der kontemplative Prozess des Scripten von Code kann nicht in ein billiges 45-Minuten-Prüfungsszenario gepresst werden. So wie jede schriftliche Prüfung (egal welcher Gegenstand) absurd und unnötig ist - da sich Schreib- und Denkarbeit niemals der Zeit beugen darf.

Die mündliche Prüfung jedoch ist umso edler. Sie zeigt tatsächlich im Moment das Können - ob nun in der Fremdsprache der Weg zum Bahnhof gefunden, oder das Wissen über Photosynthese abgefragt wird - es geht immer dabei ums sophisticated-being. Dem jungen Janus ist dieser Umstand aber wohl eher unbedeutend - da er sowohl ein Recht als auch ein Bedürfnis hat, dass seine Leistungen bewertet und beurteilt werden.

Diese seine Arbeit ist nun in diesem Moment produktorientiert obgleich seine Leistung prozessorientiert sein sollte. Der_die Informatik-Lehrer_in muss also beides berücksichtigen - jedoch sollte sich Prozess und Produkt nicht durch Zeit bestimmen - wo dann die große Schwierigkeit entsteht: "Wie kann man einen Prozess im Jetzt bewerten?". Dem jungen Janus sollte im einfachsten Angesicht eines Editors nun Ehrfurcht vermittelt werden. Die Leere der Tabula-Rasa soll sich von Ehrfurcht in Respekt wandeln. Aus einer Welt der Reizüberflutung wird der junge Janus also in eine Welt der Geduld aber Unnachsichtigkeit eingeführt. Die Ästhetik daraus ergibt sich von selbst.

Dieser neue Konstruktivismus ist natürlich durch eine Unmenge an Imperativen determiniert - und nun lernt also Janus tatsächlich die Regeln einer leeren Tafel - denn, das Werk muss einer Logik entsprechen und wenn es auch nur die Logik der Imperative ist. Das unlogische Werk in der Informatik ergibt in einer tieferen Translationsebene wieder Sinn. So soll es auch sein - da der Unsinn

in der Informatik gegenwärtig nur Oberfläche sein kann - dahinter hat der eingeleseene Informatiker nur ein Lächeln für den Versuch "Unsinn zu stiften" übrig. Janus soll zu diesem Lächeln hin-trainiert werden! Allem Anfang nun liegt ein Zauber inne! Und hier blickt der_ die Lehrer_in in Janus Gesicht um zu erfahren ob eine Eingabe die zu einer Ausgabe führt - irgend einen affektiv-pädagogischen Mehrwert generiert.

Für Janus nicht besonders - da seine Eingabe zu einer Ausgabe führt - ist für ihn eigentlich nur die doppelte Darstellung dessen, was er geschrieben hat. Und nun beginnt die imPro. Zwei Dinge lernt nun Janus. Erstens: Wird er einer Geliebten namens EVA vorgestellt, die zwischen der Eingabe und der Ausgabe eine Verarbeitung kennt. Zweitens: "Wird er auf die Möglichkeit, dass ein Anderer die Eingabe tätigt und dieser sowohl rational als auch irrational sein kann, instruiert." EVA soll ihm helfen, diesen Anderen zu verstehen. Die Welt des Janus erweitert sich enorm. Dieser Andere, er will so viel wissen. Er will wissen welcher Tag heute ist, er will wissen welche Verzinsung sich bei einem Eigenkapital von € 5.000,- und einem Prozentsatz von 3 % ergibt. Dieser Andere will Sachen wissen, die Janus unter Umständen interessieren könnte, aber eher im Moment irrelevant sind. Aber unser Janus ist ein guter Schüler. Ein guter Schüler weil er die Imperative und die Logik liebt und jede Möglichkeit als Eventualität betrachtet.

EVA hat also leichtes Spiel mit Ihm - wäre EVA ein Mensch. EVA ist aber die imaginäre Freundin von Janus, die von seinem_ ihrem Lehrer_in zugeteilt wurde. Eine Ehe auf Zeit - bis zur Matura - eine Liebschaft bei jeder Prüfung - je nachdem wie der_ die Lehrer_in Janus zur Traurigkeit oder zum Lachen zwingen will. Die imperative Logik ist in seinem Wesen ästhetisch einer andren Logik gleichzusetzen - sei es nun eine Prädikative oder eine Mathematische oder eine Fuzzylogic. Aus allen Erscheinungen lässt sich eine Informatik bilden - ob sie nun einfach nur auf der Booleschen Algebra basiert ist nebensächlich.

Eine Informatik kann auch rein deskriptiv sein - sie muss nicht nur Nullen und Einsen durch einen Prozessor jagen, der in seinem Wesen doch nur ein Allzu menschliches ist. So ist es ebenfalls denkbar, vom informatischen System eine Antwort auf eine Frage zu bekommen ohne das informatische System deklarativ aufzufordern eine Antwort zu geben - in Form von Imperativen.

Das Design eines Prozessor hängt weniger von seinen möglichen Zuständen als eher von der Art der Verwendung ab - und so bleibt gegenwärtig eine Eingabe nichts anderes als die Aufforderung Billiarden von Elektronen im Prozessor nach links oder rechts zu schupfen um so eine neue Anordnung von Elektronen zu erzeugen - die in ihrer Reihenfolgen Nullen und Einsen geordnet zurückgeben. Janus wird immer dem Ganzen Freude und Langeweile zugleich abgewinnen und die Nullen und Einsen werden zu Zeichen. Wird Janus jedoch in seinem ausgewachsenen Alter ein Offizier einer militärischen Armee, dann wird er Befehle in einer logischen Reihenfolge und in seiner Korrektheit richtig vermitteln können. Das hat ihm die imperative Programmierung gelehrt. Aber das affektive hat er wohl eher von "Just Cause 4", "Hitman 2" oder "Counter Strike" gelernt.

Quellenverzeichnis

Internetquellen und online Tools

Background Image Generator

<http://bg.siteorigin.com/>

css4.at Lernplattform

<https://www.css4.at>

dict.cc Wörterbücher

<https://www.dict.cc/>

Duden online

<https://www.duden.de/woerterbuch>

ExtractMetaData – Meta Daten auslesen

<https://www.extractmetadata.com/de.html>

flickr Fotocommunity

<https://www.flickr.com/>

Fundamentum – pädagogisches Forum

<http://fundamentum.xobor.de/>

Google (nahezu alle Dienste)

<https://www.google.at>

MDN web docs – Mozilla für Entwickler

<https://developer.mozilla.org/de/>

Mediaevent – CSS, HTML und JS mit {stil}

<https://www.mediaevent.de/>

openstreetmap

<https://www.openstreetmap.org>

Peter Kropff Tutorials

<https://www.peterkropff.de/>

pixabay Bilderplattform

<https://pixabay.com/de/>

php.net – Benutzerhandbuch

<https://www.php.net>

PHP-Einfach.de Tutorials

<https://www.php-einfach.de>

selfhtml.org – Die Energie des Verstehens

<https://selfhtml.org/>

stackoverflow – developers empowering

<https://stackoverflow.com/>

W3C CSS Validation Service

<http://jigsaw.w3.org/css-validator>

W3C Markup Validation Service

<https://validator.w3.org>

w3schools.com – Web Developer Site

<https://www.w3schools.com/>

Wikimedia Commons

<https://commons.wikimedia.org>

Wikipedia – die freie Enzyklopädie

<https://de.wikipedia.org>

Literaturverzeichnis

Bauer, Joachim, Lob der Schule, Heyne Verlag, Hamburg 2008

Doralt (Hrsg.), Werner, Kodex Schulgesetze, LexisNexis Verlag, Wien 2010

Esteras, Remacha, Infortech – English for computer users, Cambridge University Press 2008

Fürst, Maria, Trinks, Jürgen, Philosophie, ÖVB Pädagogischer Verlag, SchBNr. 3380, ISBN 3-215-10085-1

Gaisbacher, Johann, Pongratz, Hanns Jörg (Hrsg.), Persönlichkeiten stärken, Leykam Graz, 2012

Glattauer, Niki, Der engagierte Lehrer und seine Feinde, Verlag Carl Ueberreuter, Wien 2010

Goleman, Daniel, Kaufman, Paul, Ray, Michael, Kreativität entdecken, Carl Hanser Verlag, München-Wien 1997

Goleman, Daniel, Soziale Intelligenz, Knauer, München 2008

Greimel, Bettina, Evaluation österreichischer Übungsfirmer, StudienVerlag GesmbH, Wien-Innsbruck 1998

Gruschka, Andreas, Verstehen lehren, Reclam Verlag, Stuttgart 2011

Gudjons, Herbert, Spielbuch Interaktions-Eziehung, Julius Klinkhardt, Bad Heilbrunn 2003

Hauser, Tobias, Wenz, Christian, PHP 7 und MySQL, Rheinwerk Verlag Bonn, 2019

Heckhausen, Heinz, Hoffnung und Furcht in der Leistungsmotivation, Verlag Anton Hain, Meisenheim am Glan, 1963

Hesse, Jürgen, Schrader, Hans Christian Schrader, Testtrainig Logik, Eichborn Verlag, Frankfurt am Main, 1999

Hüffel, Clemens, Reiter (Hrsg.), Anton, Handbuch Neue Medien, CDA Verlag, Perg 2008

Jacobsen, Jens, Website Konzeption, Addison-Wesley Verlag, München 2009

Jarz, Thorsten, Grundlagen der Netzwerktechnik, Servicebetrieb ÖH-Uni Graz GmbH, Graz 2011

Jarz, Thorsten, VB.NET Eine Einführung ins Programmieren, Servicebetrieb ÖH-Uni Graz GmbH, Graz 2010

Jarz, Thorsten, Visual Basic für Applications in Excel, Servicebetrieb ÖH-Uni Graz GmbH, Graz 2009

Jungmayer, J. R., Reproduktions und Druck Technik, Leykam-Verlag, Graz ISBN 3701112444

Klafki, Wolfgang, Neue Studien zur Bildungstheorie und Didktik, Beltz Verlag, Basel 1985

Kölbl, Doris, Hutz, Gabriela, Lehrerinnenarbeit – heute und morgen, StudienVerlag GsmBH, Innsbruck 1997

Kopeinigg, Christine, Textdesign 2007 Informations- und Officemanagement 1/I, Wien 2007

Kopeinigg, Christine, Textdesign und Publishing, Informations- und Officemanagement 1/I, Wien 2011

Laube, Michael, Einstieg in SQL, Rheinwerk Verlag Bonn, 2019

Lauffer, Dora, Der Weg zum erfolgreichen Unterricht, Weishaupt Verlag, Graz 1996

Liessmann, Konrad Paul, Theorie der Unbildung, Piper Verlag, München-Wien 2011,

Mitschka, Ruth, Die Klasse als Team, 1997

Mitschka, Ruth, So lernt man lernen, 1983

Moriz, Werner, Unterrichtswissenschaften, Books on Demand GmbH, Norderstedt 2007

Peterßen, Wilhelm H., Kleines Methoden-Lexikon, Oldenbourg 2009

Pflugfelder, Michael, Normativität und Professionalität in der Übungsfirma, Grin Verlag, Norderstedt 2007

Posch, Peter, 9 x Paritzipation – Praxisbeispiele aus der Schule, Verlag Carl Ueberreuter, Wien 2006

Poser, Hans, Wissenschaftstheorie, Reclam Verlag, Stuttgart 2001

Punkenhofer, Yvonne, Das Funktionieren des Übungsfirmenmarktes, VDM Verlag Dr. Müller, Saarbrücken 2011

Schaub, Horst, Zenke, Karl, Wörterbuch Pädagogik, DTV München 2007

Schischkoff, Georgi, Philosophisches Wörterbuch, Kröner Verlag, Stuttgart 1991.

Schmidt, Siegfried J., Handbuch Werbung, LIT Verlag, Münster 2004

Schneider, Wilfried, u. a., Betriebswirtschaft HAK 1 bis 5, Manz Verlag Schulbuch GmbH, Buch-Nr. 136141, Buch-Nr. 120806, Buch-Nr. 125368, Buch-Nr. 131041, Buch-Nr. 136143, Wien 2008

Schubert, Sigrid, Schwill, Andreas, Didaktik der Informatik, Spektrum Verlag Heidelberg 2011

Schulz-Reiss, Christine, Nachgefragt: Philosophie, Basiswissen zum Mitreden, Loewe Verlag GmbH, Bindlach 2005

Seebohn, Joachim, Kompakt-Lexikon Werbepaxis, Verlag Gabler GmbH, Wiesbaden 2001

Specht, Werner, Thonhauser Josef (Hrsg.), Schulqualität, Innsbruck 1996

Stephan, Ingrid, Büroprozesse in der Übungsfirma, Bildungsverlag EINS, Köln 2011

Stettner, Marko, Manipulation und Pädagogik, Leykam/Pädagogischer Verlag, Graz 1973

Stumpf, Hildegard, Die wichtigsten Pädagogen, Marix Verlag GmbH, Wiesbaden 2007

Theis, Thomas, Einstieg in PHP7 und MySQL 5.6, Rheinwerk Verlag Bonn, 2016

Wabnegger, Erwin, Feindbild Lehrer, EDITION VA BENE, Wien-Klosterneuburg 2002

Werneck, Tom, Die raffiniertesten Denkspiel, VPM Verlagsunion Pabel Moewig KG, Rastatt

Whitehead, Alfred North, Wissenschaft und moderne Welt, Suhrkamp Taschenbuch Verlag, Frankfurt am Main 1984

Winter, Felix, Grundlagen der Schulpädagogik Band 49: Leistungsbewertung, Schneider Verlag Hohengehren, 2012

Winterhoff, Michael, Thielen, Isabel, Persönlichkeiten statt Tyrannen, Wilhelm Goldmann Verlag, München 2011