Einführung in Javascript

https://iuk.one/1066-1002

Clemens H. **Cap**
ORCID: 0000-0003-3958-6136

Department of Computer Science
University of **Rostock**
Rostock,Germany
clemens.cap@uni-rostock.de

Version 2

Übersicht

1. Wie entstand Javascript?
2. Wie lernt man Javascript?
3. Das Javascript Eco System
4. Alternativen und Zukunft von Javascript

1. Wie entstand Javascript?

Was waren die **treibenden Kräfte** für die Entwicklung von Javascript?

Die Geschichte

*** 1995:** Erste Gehversuche im Netscape Navigator, motiviert durch Java (* 1995).

2000: Versuche von Microsoft mit **JScript** W

2005: Erste Normierungs-Aktivitäten als **ECMA Script.**
ECMA = **E**uropean **C**omputer **M**anucafturers **A**ssociation
ECMA-Script Standards von 2011, 2015, 2016, 2017, 2018, 2019
Aktuell letzter Standard: ECMA Script 2022 Language Specification vom 6. April 2021

Heute: Je nach Betrachtungsweise eine der am **weitverbreitetsten** Sprachen.

Ausgangspunkt

Anforderungen:

- Webseiten dynamisch machen ohne Server-Interaktion.
- Webseiten mit rascher(er) Reaktion auf Client-Interaktion.
- Nutzung elementarer lokaler Browser-Features (History, Forms).

Frühe Umsetzungsversuche dieser Anforderungen

- **Java Applets:** Bytecode VM, eingebettet oder via PlugIn
- **VBScript:** Universelle MS Skripting-Sprache, auch für Explorer
- **ActiveX:** Kompilierte Programme auf Intel Architektur
- **Javascript:** Interpretierte Sprache

Neue Wünsche

Lokale Hardware nutzen: GPS, Mikrophon & Camera, Sensorik

Bessere Performance erzielen durch

- **Compilation:** Von Interpretation zu Compilation.
- **Parallelität:** Webworker, Serviceworker.
- **Speicherung:** Cache, Cookies, localStorage, IndexedDB.

Stärkere Kontrolle über den User

- Software vom Server: Google Docs, Overleaf, Office 365.
- Binary Only Modules DRM Standards für Video in HTML5 🔗, 🔗
- App Store Struktur iOS, Playstore, MS.

Geringere Entwicklungskosten

- Technologie auf Client und Server gleich.
- UI Entwicklung nur in einer Technologie (HTML–CSS–JS).

Erstes Beispiel

```
<html><head></head>
<body><h1>Beispiel</h1>
Heute ist der <span id="feld"></span>
<script type="text/javascript">
  document.write("<hr>");
  document.write("<b>Hello</b> World");
  var x = document.getElementById("feld");
  x.innerHTML = "<b>" + (new Datum ()) + "</b>";
</script>
</body></html>
```

Src. 1: Ein erstes Javascript Programm.

Heutige Bedeutung

Statista: Most Used Programming Languages

TIOBE-Index der Programmiersprachen

PYPL Popularity-Index of Programming Languages

Northeastern University: Most Popular Programming Languages to Learn

Generelle Situation

Wichtigste Deployment Plattform.
Über 600 Sprachen kompilieren nach Javascript.
Von Typescript bis Python, von Erlang bis Perl, von Ceylon bis F#.

Dutzende Parser, Compiler, Transpiler, Optimierer, Debugger, Interpreter usw.

Quelle: Liste der Sprachen mit JS Verbindung

2. Wie lernt man Javascript?

Learning by Doing ist der ~~beste~~ einzige Weg, eine Programmiersprache zu lernen.

Der Einstieg

Javascript ist ein naher Verwandter von Java und C/C++.

Einige Konzepte sind anders und (meistens) einfacher.

Tip: Durcharbeiten eines guten JS Tutorials im Netz.

Learn JS:	Gut zum ersten Einstieg in die Sprache
Javascript.Info:	Umfassender, auch mit fortschrittlichen Konzepten
W3 Schools:	Übersichtlich mit Online Übungen, gelegentlich kleine Fehler

Referenzen und Quellen

Kangax Kompatibilitäts-Tabellen
Übersicht über die Implementierung von JS Sprachstandards.

Can I Use
Zentrale Quelle zur Verfügbarkeit vieler JS Schnittstellen.

MDN: Mozilla Developers Network
Klassisches Referenzhandbuch für JS Entwicklung.
Tip: Begriff googeln und MDN mit in die Query dazunehmen.

Stackoverflow
Klassische Quelle für Community Hinweise bei JS Problemen.
Vorsicht: Oft falsche weil veraltete Antworten.

Ecma Technical Committee 39
Quelle neuester Sprachvorschläge zu Javascript.

Chrome Developer Blog
Neuigkeiten in Chrome; Entwicklung von Chrome Extensions.

Aufgabe

Schlage alle Elemente des Hello World Programms bei MDN nach.

Der PDF-Player auf iuk.one in der Version März 2021
nutzt(e) das OGG VORBIS Format. Auf iOS zeigt der Player eine Fehlermeldung an.

- Wird OGG VORBIS durch iOS derzeit unterstützt?
- Welcher Prozentsatz aller Desktop Geräte weltweit unterstützt OGG VORBIS?
- Wie würde sich der Prozentsatz ändern, wenn das Format auf MP3 umgestellt würde?
- Der Player nutzt zum Rendern von PDF ein <canvas> Element.
  Könnte auch das zu Kompatibilitätsproblemen führen?

`Static Class Features` sind ein neuer Vorschlag für Javascript.

- Was ist das genau?
- Kann ich das heute in Node.js bereits sinnvoll nutzen? Im Browser?

3. Das Javascript Eco System

Welche Werkzeuge benötigt man zur Entwicklung mit Javascript?

Die Begrifflichkeit des Javascript Ecosystems und ergänzender Elemente wird am besten durch dieses Bild dargestellt: Javascript Eco System

Umgebungen und Werkzeuge

"Engines" werten Javascript aus.

- V8 🔗 W
- Rhino 🔗 W
- Spidermonkey W
- W: Liste von Engines
- W: Vergleich wichtiger Engines

Ausführungsumgebungen von Engines

- **CLI:** Einbettung in ein Shell Programm
- **Browser:** Einbettung in einen Web Browser
- **Programme:** Nutzung in beliebigen Programmen zur Skriptsteuerung
- **IDE:** Als Teil von Entwicklungsumgebungen
- **Sandboxes:** Ausführung in mehr oder weniger sicherer Simulationsumgebung

CLI: Command Line Interface

Erweitern typischerweise um:

- **JS Objects:** Einige grundlegende Objekte (Object, Array, Function)
- **REPL:** Read Eval Print Loop: Auswertung einfacher Ausdrücke
- **Loader:** Laden von Quell-Programmen zur Ausführung
- **OS Interface:** Zugriff auf OS
- **Engine Interface:** Zugriff auf interne Funktionen der Engine

Beispiele: node, Rhino-Shell, Deno

Browser

Erweitern typischerweise um:

- **JS Objects:** Einige rundlegende Objekte (Object, Array, Function)
- **DOM:** Document Object Model: W3C HTML Schnittstelle
- **BOM:** Browser Object Model: Browser-spezifische Schnittstelle
  Typisch: `navigator`, `screen`, `location`
- **Layout Engine:** Visualisierung von DOM + CSS
  Blink (Chrome, Edge), **Webkit** (Safari),
  Gecko (Fox), Trident (IE)
- **Extensions:** Erweiterungsmechanismen

W: List of Web Browsers

Programme

Zunehmend wird JS als Skripting Sprache für beliebige Programme genutzt.

Bsp: TeXStudio **via** QtWebKit Bridge

Bsp: Electron W

- Bündelt einen Chromium Browser mit einem node Server.
- Erlaubt Entwicklung nativer Anwendungen mit Web-Technologien.

Bsp: Atom Editor W

- Klassische Electron Anwendung
- In Javascript geschriebener Editor
  Genauer: In CoffeeScript und Less

IDE: Integrated Development Environment

Persönlich: Erst 15 Jahre Eclipse; dann IDEs zu bloated und opinionated

- **Editor:** Atom nightly, da optimal konfigurierbar.
- **Umgebung:** Neueste Chrome / Node, da Referenzsystem.
- **Hot reload:** Browser-Sync, da Klicks und Keys spart.

Daher kein echter eigener Überblick zu IDEs.

Abhängig von Situation (Team, social coding, ego coding, Firmenprojekt oder Lernen) kann eine angepaßte IDE sehr helfen. (Bsp: Spezifische Angular, React completions)

Übersichten: 🔗, 🔗, 🔗

Sandboxes

Merkmale:

- Oftmals vollständig Web-basierte Portale.
- Mehr oder weniger gut abgeschlossene Umgebungen.
- Bieten Editoren (für JS, auch HTML, CSS, LESS usw.)
- Ausführung via Konsolen-Umleitung und Web-in-iframe Anzeigen
- Anbindung an social und shared coding Mechanismen.

Beispiele:

- jsFiddle
- jsBin
- CodePen
- jsBin
- Plnkr (Plunker)

Persönliche Empfehlung zum Lernen

Grundsätzlich:
Sehr individuell.

Thema 1: Referenzsysteme nutzen
Umgebungen, die etwas nur zu 95% richtig machen.
Die 5% kosten Zeit, Nerven, Energie.

Thema 2: Dogmatischer Stil "opinionated"
Umgebungen, die einen festen Stil vorgeben behindern die Suche nach optimalen Lösungen.

Thema 3: Spezifikation statt "Klickiebuntie"
Bei textueller Eingabe muß man wissen, was man will.
Code Completion kann helfen.
Zu viel Boilerplate und Scaffolding behindern das Lernen und schränken den Stil ein aber beschleunigen die Entwicklung.

Weitere Elemente des Eco Systems

Pedro Rolo: A Javascript Ecosystem Overview
Das einleitende Bild über den Turmbau zu Babel illustriert
warum ein vollständiger Überblick über das JS Ecosystem nicht gegeben werden kann.

Javascript Ecosystem: 38 Tools
Erwähnt (nur) die wichtigsten 38 Tools
Gibt einen sehr guten Querschnitt.

Mirza Leka: Exploring the JavaScript Ecosystem
Gibt einen sehr schönen konzeptuellen Überblick.

4. Alternativen und Zukunft von Javascript

Wo entwickelt sich Javascript hin?

Sprachalternativen

Babel

Modularisierung und Bündelung

Marktbereinigung bei den Frameworks

Typescript

Anhang

Übersicht

Programmquellenverzeichnis

Rechtliche Hinweise (1)

Die hier angebotenen Inhalte unterliegen deutschem Urheberrecht. Inhalte Dritter werden unter Nennung der Rechtsgrundlage ihrer Nutzung und der geltenden Lizenzbestimmungen hier angeführt. Auf das Literaturverzeichnis wird verwiesen. Das Zitatrecht in dem für wissenschaftliche Werke üblichen Ausmaß wird beansprucht. Wenn Sie eine Urheberrechtsverletzung erkennen, so bitten wir um Hinweis an den auf der Titelseite genannten Autor und werden entsprechende Inhalte sofort entfernen oder fehlende Rechtsnennungen nachholen. Bei Produkt- und Firmennamen können Markenrechte Dritter bestehen. Verweise und Verlinkungen wurden zum Zeitpunkt des Setzens der Verweise überprüft; sie dienen der Information des Lesers. Der Autor macht sich die Inhalte, auch in der Form, wie sie zum Zeitpunkt des Setzens des Verweises vorlagen, nicht zu eigen und kann diese nicht laufend auf Veränderungen überprüfen.

Alle sonstigen, hier nicht angeführten Inhalte unterliegen dem Copyright des Autors, Prof. Dr. Clemens Cap, ©2020. Wenn Sie diese Inhalte nützlich finden, können Sie darauf verlinken oder sie zitieren. Jede weitere Verbreitung, Speicherung, Vervielfältigung oder sonstige Verwertung außerhalb der Grenzen des Urheberrechts bedarf der schriftlichen Zustimmung des Rechteinhabers. Dieses dient der Sicherung der Aktualität der Inhalte und soll dem Autor auch die Einhaltung urheberrechtlicher Einschränkungen wie beispielsweise Par 60a UrhG ermöglichen.

Die Bereitstellung der Inhalte erfolgt hier zur persönlichen Information des Lesers. Eine Haftung für mittelbare oder unmittelbare Schäden wird im maximal rechtlich zulässigen Ausmaß ausgeschlossen, mit Ausnahme von Vorsatz und grober Fahrlässigkeit. Eine Garantie für den Fortbestand dieses Informationsangebots wird nicht gegeben.

Die Anfertigung einer persönlichen Sicherungskopie für die private, nicht gewerbliche und nicht öffentliche Nutzung ist zulässig, sofern sie nicht von einer offensichtlich rechtswidrig hergestellten oder zugänglich gemachten Vorlage stammt.

Use of Logos and Trademark Symbols: The logos and trademark symbols used here are the property of their respective owners. The YouTube logo is used according to brand request 2-9753000030769 granted on November 30, 2020. The GitHub logo is property of GitHub Inc. and is used in accordance to the GitHub logo usage conditions https://github.com/logos to link to a GitHub account. The Tweedback logo is property of Tweedback GmbH and here is used in accordance to a cooperation contract.

Rechtliche Hinweise (2)

Disclaimer: Die sich immer wieder ändernde Rechtslage für digitale Urheberrechte erzeugt für mich ein nicht unerhebliches Risiko bei der Einbindung von Materialien, deren Status ich nicht oder nur mit unverhältnismäßig hohem Aufwand abklären kann. Ebenso kann ich den Rechteinhabern nicht auf sinnvolle oder einfache Weise ein Honorar zukommen lassen, obwohl ich – und in letzter Konsequenz Sie als Leser – ihre Leistungen nutzen.

Daher binde ich gelegentlich Inhalte nur als Link und nicht durch Framing ein. Lt EuGH Urteil 13.02.2014, C-466/12 ist das unbedenklich, da die benutzten Links ohne Umgehung technischer Sperren auf im Internet frei verfügbare Inhalte verweisen.

Wenn Sie diese Rechtslage stört, dann setzen Sie sich für eine Modernisierung des völlig veralteten Vergütungssystems für urheberrechtliche Leistungen ein. Bis dahin klicken Sie bitte auf die angegebenen Links und denken Sie darüber nach, warum wir keine für das digitale Zeitalter sinnvoll angepaßte Vergütungssysteme digital erbrachter Leistungen haben.

Zu Risiken und Nebenwirkungen fragen Sie Ihren Rechtsanwalt oder Gesetzgeber.

Weitere Hinweise finden Sie im Netz hier und hier oder hier.

Zitierweise dieses Dokuments

Wenn Sie Inhalte aus diesem Werk nutzen oder darauf verweisen wollen,
zitieren Sie es bitte wie folgt:

Clemens H. Cap: Einführung in Javascript. Electronic document. https://iuk.one/1066-1002
18. 4. 2021.

Bibtex Information: https://iuk.one/1066-1002.bib

```
@misc{doc:1066-1002,
   author       = {Clemens H. Cap},
   title        = {Einführung in Javascript},
   year         = {2021},
   month        = {4},
   howpublished = {Electronic document},
   url          = {https://iuk.one/1066-1002}
}
```

Typographic Information:
Typeset on April 18, 2021
This is pdfTeX, Version 3.14159265-2.6-1.40.21 (TeX Live 2020) kpathsea version 6.3.2
This is pgf in version 3.1.5b
This is preamble-slides.tex myFormat©C.H.Cap

Verzeichnis aller Folien

Legende:
Fortsetzungsseite
Seite ohne Überschrift
Bildseite