INTERNATIONALE HOCHSCHULE

IU DISCUSSION PAPERS

Design, Architektur & Bau

BROWSER – IM GRUNDE EINE GAME ENGINE.

JÖRG BURBACH

IU Internationale Hochschule

Main Campus: Erfurt

Juri-Gagarin-Ring 152

99084 Erfurt

Telefon: +49 421.166985.23

Fax: +49 2224.9605.115

Kontakt/Contact: kerstin.janson@iu.org

Autorenkontakt/Contact to the author(s):
Prof. Jörg Burbach

ORCID 0000-0003-3100-8008

IU Internationale Hochschule - Campus Köln

Bonner Straße 271

50968 Köln

Telefon: +49-1793269461

Email: joerg.burbach@iu.org

IU Discussion Papers, Reihe: Design, Architektur & Bau, Vol. 2, No. 1 (Februar 2023)

ISSN-Nummer: **2750-6266**

Website: https://www.iu.de/forschung/publikationen/

BROWSER – IM GRUNDE EINE GAME ENGINE.

Moderne Browser bringen alle Funktionen mit, um mit ihnen Spiele zu spielen. Aber sind sie deswegen eine Game Engine?

Jörg Burbach

ABSTRACT:

Game Engines gibt es bereits seit den 1980er Jahren. Eine der ersten Game Engines wurde vom Team um Shigeru Miyamoto in ihrem Motorrad-Rennen "Excitebike" (NES, 1984) eingesetzt. Später nutzen sie die Engine, um Mario in "Super Mario Bros" (NES, 1985) auf die Sprünge zu helfen. Auf dem Markt gibt es viele Game Engines, jede mit ihren Vor- und Nachteilen. Und jedes aktuelle Spiel benötigt eine Art Game Engine, aber kann das auch der Browser sein?

Game engines have been around since the 1980s. One of the first game engines was used by Shigeru Miyamoto's team in their motorcycle racing game Excitebike (NES, 1984). Later, they used the engine to give Mario a run for his money in Super Mario Bros (NES, 1985). There are many game engines on the market, each with its advantages and disadvantages. And every current game needs some kind of game engine, but can it be the browser?

KEYWORDS:

Game Studies, Game Design, Technologie, Game Engine, Browser

AUTOR

***Jörg Burbach** ist Professor für Game Design an der IU Internationalen Hochschule mit Schwerpunkt Entwicklung. Er studierte Allgemeine Verfahrenstechnik auf Diplom in Köln und Game Development and Research am Cologne Game Lab der TH-Köln. Sein Masterthema „The Future Perspectives of Point & Click Adventures“ sah das erneute Aufflammen des Genres vorher. In Vorträgen erzählt er seinem Publikum von den Vorzügen von Retro-Games und was von ihnen für die moderne Spieleentwicklung gelernt werden kann.*

Was ist eine Engine?

Eine Engine ist ein Computerprodukt, das Rahmenbedingungen und Programmroutinen für Spiele und Anwendungen zur Verfügung stellt (dthg, 2021). Aber kann ein Browser auch eine Game Engine sein? Schauen wir uns zunächst einmal die Definition der drei aufeinander aufbauenden Komponenten für die Programmierung an: Bibliothek, Framework und Engine. Diese sind wie folgt definiert.

„Bei einer Bibliothek … handelt es sich um eine Sammlung von Klassen und Funktionen. Einige arbeiten dabei im Hintergrund und bleiben nach außen hin verborgen. … Der Entwickler ruft diese Funktionen nun direkt auf und behält somit die Kontrolle, zu welchem Zeitpunkt und in welchem Umfang er diese Aufrufe tätigt" (Domin, 2018).

Die Bibliothek umfasst also lediglich einzelne Routinen, um Standardbefehle anbieten und ausführen zu können. Die entwickelnden Personen sind daher gefragt, die Routinen in der entsprechenden Art und Weise anzuwenden, um mithilfe der Bibliothek ein Spiel oder eine App herzustellen. Die Bibliothek stellt somit eine Sammlung von mehr oder weniger mächtigen Befehlen zur Verfügung. Das können etwa Befehle sein, um ein MP3 zu laden, es zu analysieren und die Datenhappen in abspielbare Wellenformen umzuwandeln. Ein weiterer Befehl könnte sein, diese Wellenformen dann an eine Soundkarte zu schicken, um sie dem Ohr zur Verfügung zu stellen.

„Das Framework stellt eine wiederverwendbare, gemeinsame Struktur für Anwendungen zur Verfügung. Entwickler binden das Framework in ihre eigenen Anwendungen ein und erweitern es so, dass es ihre bestimmten Anforderungen erfüllt" (Johnson, 1988, zit. nach Dormin, 2018).

Ein Framework hält somit spezialisierte Funktionen vor, die festgelegten (Mindest-)Anforderungen genügen müssen. Dazu gehören etwa das Abspielen von Animationen oder Musik oder auch die Interaktion mit spielenden oder nutzenden Personen. Ein Framework kann daher zum Beispiel auch andere Frameworks oder aber Bibliotheken verwenden. Um im Bild der MP3-Datei zu bleiben, würde das Framework entweder eine Bibliothek oder eigene Funktionen verwenden und die entwickelnden Personen benötigten im Bestfall nur einen einzigen Befehl, um das MP3 zu laden und automatisch abzuspielen. Den Rest - Kommunikation mit dem System zum Laden des MP3, Initialisierung der Soundkarte, Übergeben der Daten an die Soundkarte, usw. erledigt das Framework - oder die passende Routine in der Bibliothek.

Im Allgemeinen sind Bibliotheken eher auf spezielle Aufgaben fokussiert, z. B. alles, was mit der Tonerstellung und -ausgabe zu tun hat. Das Framework umfasst dahingehend weitere Routinen, etwa über Audio hinaus.

Eine Engine schließlich ist ein Softwareprodukt, mit oder ohne grafische Benutzeroberfläche, mit der eigene, vollständige und von der ursprünglichen grafischen Benutzeroberfläche unabhängig Softwareprodukte erstellt werden können. Dabei ist unerheblich, ob die Engine programmiert, also mit Quellcode gefüttert werden muss, oder ob sie grafisch bedient werden kann, etwa über das WYSIWYG-Verfahren. Die wichtigste Funktion, die die Game Engine mitbringt, ist die so genannte Game Loop. Die

Game Loop bezeichnet eine Schleife, die in der Engine immer wieder abläuft: Abfrage von Benutzereingaben, Auswerten und Umsetzen der Eingaben und interne Abläufe, Ausgabe der berechneten Grafik, Musik, etc.

Nystrom beschreibt den Unterschied wie folgt:

„Hier zeigt sich meiner Ansicht nach der Unterschied zwischen einer »Engine« und einer »Bibliothek«: Beim Einsatz einer Bibliothek stammt die Game Loop aus Ihrer Feder und Sie rufen die Bibliotheksfunktionen auf. Eine Engine hingegen ruft in ihrer Game Loop ihren Code auf" (Nystrom, 2015, S. 165).

Dazu kommt noch die immer stärker werdende Abstraktion von der Bibliothek über das Framework hin zur Engine. Entwickelnde entfernen sich dabei im Bestfall immer weiter vom Programmieren hin zum Anwenden und Umsetzen - das Wissen um die Programmierung von Computern, Mobilgeräten oder Konsolen rückt in den Hintergrund und wird somit spezialisiertem Fachpersonal überlassen, die diese auf Spezialfälle optimieren oder für die allgemeine Verwendung aufbereiten können. Dieser abstrahierte Teilbereich ist immens wichtig für die Demokratisierung der Spieleentwicklung, denn so können auch nicht-Programmierende eigene Spiele oder Anwendungen umsetzen. Zur Demokratisierung gehören zudem für Einsteiger kostenlose Game Engines, etwa Unity, Unreal oder Godot. Der Einstieg ist somit sehr leicht, niedrigschwellig und einfach möglich (Anthropy, 2012)

Der Browser als Game Engine

Nachdem wir nun wissen, was eine Game Engine (aus)macht, schauen wir uns moderne Browser an.

Moderne Browser können Musik und Videos abspielen, sie ermöglichen Interaktion, nehmen Eingabebefehle per Gamepad oder Tastatur entgegen. Sie stellen Text und Bilder dar und zeigen mit den passenden Frameworks oder Bibliotheken etwa auch 3D-Objekte. Außerdem ermöglichen sie durch vielfältige offizielle Schnittstellen auch Augmented Reality, Datenbankanwendungen oder auch das Orten per GPS. Die Plattform Caniuse zeigt, welche Funktion von welchem Browser unterstützt wird (Caniuse, 2022).

Da es sich in weiten Teilen um Standards handelt, kümmert sich das W3C-Konsortium um die Weiterentwicklung und Standardisierung des HTML-Befehlssatzes, der seit 1993 immer wieder verändert wird (w3c, 2021a). Auf diesem Standard basieren alle Webseiten, die mit einem Browser aufrufbar sind. Damit jeder Browser eine ähnliche Benutzererfahrung bietet, arbeiten die derzeit größten Browser-Hersteller (Apple, Google, Mozilla) und viele weitere Unternehmen aus der Technologiebranche, die HTML nutzen, am Standard mit (w3c, 2021b).

Wir wissen nun, dass es Standards gibt, die von allen Browserherstellern unterstützt und weiterentwickelt werden. Außerdem kennen wir die Möglichkeit, über caniuse, die Funktion zu überprüfen, die unser Produkt oder Spiel unterstützen soll. Außerdem enthält der aktuelle HTML5-Standard viele Funktionen, die umzusetzen Pflicht ist. Das führt dazu, dass im Rahmen des HTML5-Standards und der auf Webseiten nutzbaren, ebenfalls standardisierten Skriptsprache JavaScript, Funktionsbibliotheken und Frameworks programmiert wurden und auf eben diesen Webseiten nutzbar

sind. Im Falle der MP3-Datei genügt eine Zeile JavaScript, um eine MP3-Datei zu laden und abzuspielen, ganz so, wie in einer Game Engine:

new Audio('<url>').play();

MP3-Dateien sind dabei der kleinste gemeinsame (Audio)-Nenner, denn das Format wurde im Jahr 1991 von Mitarbeitenden der Fraunhofer Gesellschaft – genauer Fraunhofer IIS Audio and Media Technologies – entwickelt (fraunhofer, 2022). Im Mai 2017 liefen die Patente auf das MP3-Format vollständig aus. Seitdem ist es in den ISO-Standards ISO/IEC 11172-3 und ISO/IEC 13818-3 beschrieben und als frei und kostenlos verfüg- und nutzbar ausgezeichnet.

Diese einfache Verfügbarkeit von Funktionen durch HTML5 setzt sich in vielen Bereichen durch. Neben MP3-Dateien können Browser Videos abspielen, nehmen Benutzereingaben per Tastatur, Maus oder Gamepad entgegen und ermöglichen mit wenigen Zeilen Quellcode aufwändige interaktive Erlebnisse.

Wie oben bereits erwähnt, muss eine klassische Game Engine eine Game-Loop unterstützen. Im Grunde enthält der Browser auch eine Game-Loop, denn Animationen können z. B. mit CSS – CSS oder Cascading Style Sheets sind Textschnipsel, die Designelemente auf Webseiten beschreiben – gestartet werden und laufen dann selbstständig ab, ganz wie in der Game-Loop. Oder ein MP3 wird gestartet und erst durch einen weiteren Befehl gestoppt.

Auch wartet der Browser ganz wie eine Game Engine auf Nutzereingaben, um dann auf so genannte Events, etwa einen Klick auf einen Knopf, zu reagieren. Events nutzt etwa auch Unity, um UI-Abfragen wie Mausklicks auf Knöpfe zu erkennen und dann programmierte Routinen ablaufen zu lassen. Aber auch auf Events wie das Drehen eines Mobilgerätes oder das Ende einer Animation reagieren Browser und Game Engines gleichermaßen.

Wir können also schlussfolgern, dass moderne Browser tatsächlich auch als Game Engine bezeichnet werden können, wenn die aktuell laufende Webseite ein Spiel enthält. Sie erfüllen zumindest alle Bedingungen einer Game Engine.

Der Browser als technologie-agnostische Lösung

Also, warum werden nicht viel mehr Spiele im Browser gespielt? Liegt es an der Performance der Browser? Oder fehlen Funktionen?

Wir haben gesehen, dass Browser im Prinzip jegliche Funktionen einer Game Engine abbilden oder durch Bibliotheken abbilden können. Bei der auf Webseiten vorherrschenden Sprache JavaScript handelt es sich um eine interpretierte Sprache. Das bedeutet, dass die Browser die Befehle nach und nach aus dem Quelltext lesen und dann interpretieren, also die Funktion aus dem Text abbilden und ausführen. Moderne Systeme übersetzen beim ersten Ausführen eines Befehls per JIT (Just in time) aus JavaScript in Maschinencode (oder Bytecode). Das führt dazu, dass die Befehle bei Folgeaufrufen wesentlich schneller ausgeführt werden können, Maschinencode kann der Computer in höchstmöglicher Geschwindigkeit ausführen - auch wenn er nicht so optimiert sein wird, wie ein ausführbares Programm (heise, 2019). Moderne Browser übersetzen JavaScript meist in Bytecode, der dann von einer entsprechenden Engine ausgeführt wird. Technische Informationen dazu bietet ein

Blogeintrag aus der Entwicklung des Webkit-Frameworks, das Apples Safari und andere Browser antreibt (webkit, 2019).

Mit Technologien wie WebAssembly, das ebenfalls vom w3c standardisiert wird, können JavaScript und HTML5 basierte Webseiten mit sehr performantem Code versorgt werden, der beinahe die Geschwindigkeit eines programmierten Produkts erreicht. WebAssembly und die per JIT übersetzten JavaScript-Befehle können in der gleichen Engine im Browser ablaufen und erreichen somit auch eine ähnliche Geschwindigkeit oder auch Komplexität (webassembly, 2022).

Unabhängig davon, dass bei normalem JavaScript-Code die Programmierung einsehbar ist – schließlich handelt es sich um eine einfache Text-Datei – gibt es für Spiele-Studios noch eine andere Problematik: Konsolen etwa unterstützen von Haus aus keine Browser-basierten Spiele, denn es gibt spezialisierte Software Development Kits der Hersteller. Außerdem wollen und müssen die Entwicklerteams ihre Spiele auf die gewünschte Konsole oder den Computer optimieren, da sie ihren Spielern die bestmögliche Erfahrung bieten wollen.

Wir fassen also zusammen: Browser können durchaus als Game Engine bezeichnet bzw. genutzt werden und können in gewissen Grenzen sicherlich mit einer Engine wie Unity oder Unreal mithalten. Sie bieten alle notwendigen Funktionen, werden aber im Allgemeinen nur für im technischen Sinne wenig anspruchsvolle Spiele verwendet. Dabei wären sie eine hervorragende Möglichkeit, größere Käufer- und Spielerschichten anzusprechen: Denn im Browser laufende Webseiten sind per se Technologie-agnostisch – ihnen ist egal, auf welcher Plattform oder in welchem Browser sie laufen, solange die für ihre eigene Funktion notwendigen Bestandteile, also etwa das Abspielen des MP3, zur Verfügung stehen. Und damit sind sie grundsätzlich von der Plattform unabhängig und überall einsetzbar.

Durch geschickte Programmierung würde der Browser als überall verfügbare, oft vorinstallierte Game Engine eine große Verbreitung für Spiele und Apps ermöglichen und macht sich dadurch unabhängig von Gatekeepern wie Apple und Google, die den mobilen Markt quasi unter sich aufgeteilt haben. Android hält im November 2022 72,96%, iOS 27,48% des Marktes. (statista, 2022)

Einzig die möglicherweise notwendigen Anpassungen an verschiedene Browser sind es, die den Programmierenden Sorgen machen und sie gleichermaßen fordern – speziell, wenn die Anforderung an das Spiel ist, überall identisch zu sein (Mishra, 2019; Unadkat, 2022; Nield, 2022). Funktionsgleichheit ist mit den aktuellen Mechanismen, den w3c-Standards, JavaScript und dem Nutzen ausschließlich standardisierter Funktionen und Inhalte kein Problem.

Aber die Probleme betreffen in ähnlichem Maße das Portieren eines Windows-Spiels auf eine andere Plattform, wie etwa MacOS oder auf eine Konsole wie Switch oder Playstation. Microsoft hat das Problem mit dem eigenen Game Development Kit etwas entschärft, indem die Xbox sich immer weiter Windows annähert (Microsoft, 2022). Auf diese Weise sind viele Funktionen auf beiden Plattformen identisch.

Seit wann wird über den Browser als Basis nachgedacht?

Dass diese Idee nicht neu ist, erstaunt nicht: Bereits 2008 stellten Antero Taivalsaari et. al. von Sun Microsystems die These auf, dass Apps mit Browsern als Basis hervorragend funktionieren sollten:

“Applications are no longer written for a specific type of computer, operating system or device. Rather, they will be written for the Web, to be used via a web browser from anywhere, anytime.“ (Taivalsaari, 2008)

Somit haben sie etwa auch ChromeOS von Google vorhergesagt, das genau das ist: ein Betriebssystem, das im Prinzip nur aus einem Browser besteht, in dem Apps, also Webseiten, laufen. Und genauso sagten sie damit auch Teile von iOS vorher, denn auf Apple-Geräten werden Progressive Web Apps – ebenfalls Webseiten – gerne in iOS-Apps untergebracht und über den App Store verteilt. (PWA, 2021)

Taivalsaari fasst zusammen – und das gilt grundsätzlich genauso für Videospiele:

“Like it or not, the Web is increasingly the platform of choice for advanced software applications. Web-based applications have major benefits: no installation or upgrades needed, instant worldwide deployment, the ability to “own” the users' data. Web-based applications will dramatically change the way people develop, deploy and use software -> paradigm shift!“ (Taivalsaari, 2008)

Mit einem Web-Browser ist also im Prinzip schon eine mächtige Game-Engine auf jedem Computer oder Mobilgerät vorinstalliert oder sehr leicht zu installieren. Der nächste Schritt wäre also eine flächendeckende Nutzung der Web-Technologien für neue Spiele und Apps.

Literaturverzeichnis

Anthropy, A. (2012): Rise of the Videogame Zinesters: How Freaks, Normals, Amateurs, Artists, Dreamers, Drop-outs, Queers, Housewives, and People Like You Are Taking Back an Art. Seven Stories Press, illustrated Edition

Caniuse (2022): Support tables for HTML5, CSS3, etc. Website. https://caniuse.com

Chrome OS (2022): ChromeOS. Webseite. https://www.google.com/chromebook/chrome-os/

Domin, A (2018). Library vs. Framework: Das sind die Unterschiede. Webseite. https://t3n.de/news/library-vs-framework-unterschiede-1022753/

dthg (2021). Kurz erklärt: Was ist eine Spiele-Engine und wozu brauche ich sie? Webseite. https://digital.dthg.de/kurz-erklaert-spiele-engine/

fraunhofer (2022). Was ist MP3? Webseite. https://www.mp3-history.com/de/whatismp3.html

heise (2019). JavaScript Engines: Performance-Steigerung der Browser. Webseite. https://www.heise.de/hintergrund/JavaScript-Engines-Performance-Steigerung-der-Browser-4264792.html?seite=4

Microsoft (2022). Xbox Game Development. Webseite. https://learn.microsoft.com/en-us/gaming/xbox/

Mishra, P. (2019). Browser Engines: The Crux Of Cross Browser Compatibility. Webseite. https://www.lambdatest.com/blog/browser-engines-the-crux-of-cross-browser-compatibility/

Nield, D. (2022). Which Browser Engine Powers Your Web Browsing and Why Does It Matter? Webseite. https://www.gizmodo.com.au/2022/08/which-browser-engine-powers-your-web-browsingand-why-does-it-matter/

Nystrom, R. (2015). Design Patterns für die Spieleprogrammierung (mitp Professional) (1. Aufl.). mitp., S. 165

PAW (2021). Publish your PWA to the iOS App Store. Webseite. https://blog.pwabuilder.com/posts/publish-your-pwa-to-the-ios-app-store/

Statista (2022). Marktanteile der führenden mobilen Betriebssysteme an der Internetnutzung mit Mobiltelefonen weltweit von Januar 2011 bis November 2022. Webseite. https://de.statista.com/statistik/daten/studie/184335/umfrage/marktanteil-der-mobilen-betriebssysteme-weltweit-seit-2009/

Taivalsaari, A., Mikkonen, T., Ingalls, D., Palacz K. (2008). Web Browser as an Application Platform. Conference Proceedings of the EUROMICRO. http://doi.org/10.1109/SEAA.2008.17

Unadkat, J. (2022). Understanding the Role of Rendering Engine in Browsers . Webseite. https://www.browserstack.com/guide/browser-rendering-engine

w3c (2021a). Standards. Webseite. https://www.w3.org/standards/

w3c (2021b). Current Members & Testimonials. Webseite. https://www.w3.org/Consortium/Member/List

webassembly (2022). WebAssembly 1.0 has shipped in 4 major browser engines. Webseite. https://webassembly.org

webkit (2019). A New Bytecode Format for JavaScriptCore. Webseite. https://webkit.org/blog/9329/a-new-bytecode-format-for-javascriptcore/

Williams, Andrew (2017). History of Digital Games: Developments in Art, Design and Interaction. CRC Press. S. 152 – 154.

Alle Webseiten abgerufen am 12. Januar 2022.