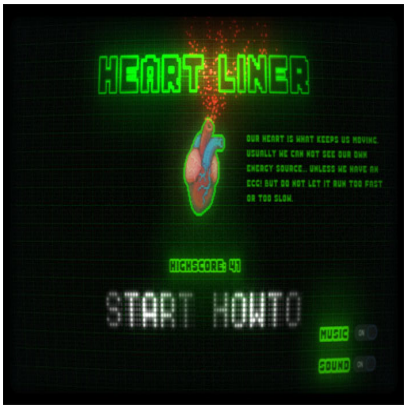


Heart Liner – wenn der Arcade-Puls höher schlägt

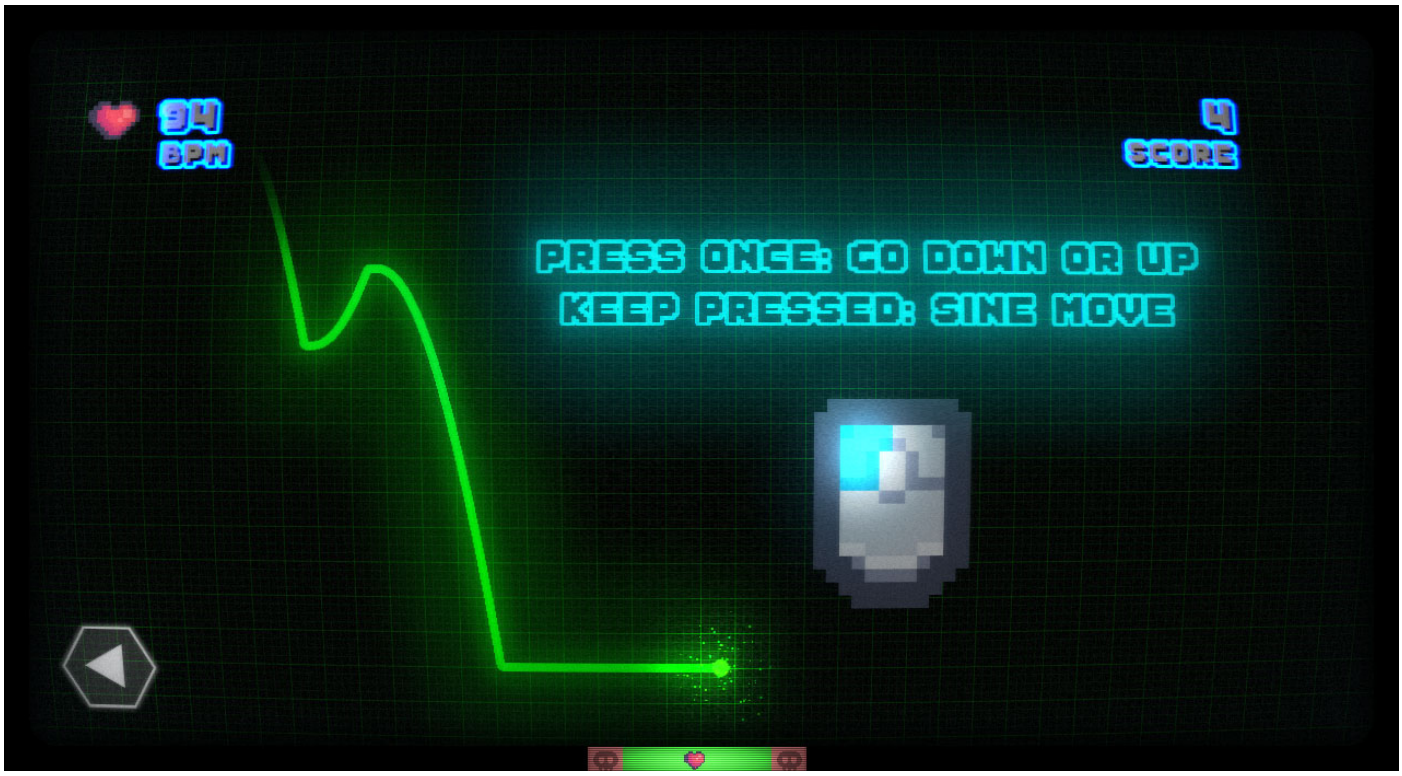
Description

Mein Bezug zu Arcade-Spielen, Reise in die Vergangenheit

Den Ursprung haben Arcade-Spiele in den Spielhallen der 70er und 80er Jahren. Bei den Arcade-Automaten ging es darum, die Spieler möglichst schnell ins Spiel zu bringen. Für lange Tutorials war keine Zeit: Geld einwerfen und loslegen war die Idee. Das hat natürlich auch das Game-Design geprägt. Meist haben Arcade-Spiele ein einfaches Konzept, einen schnellen Einstieg und einen fast exponentiellen Anstieg des Schwierigkeitsgrades.

Zudem startet man das Spiel stets vom Anfang, ohne Speichern zu können. Für mich persönlich macht aber nicht nur das Game-Design ein Arcade-Spiel zu dem was es ist, sondern auch und insbesondere die visuelle Gestaltung. Bevor es LCD-, LED- oder OLED-Fernsehgeräte gab, wurden Videospiele und auch Arcade-Spiele auf sogenannten Röhrengeräten dargestellt.

Im Grunde habe ich meine ganze Jugend vor solchen Flimmerkisten verbracht und damit hat sich auch der Look eingeprägt. Solche Fernseher nutzen Kathodenstrahlröhren oder im Englischen cathode ray tubes, also kurz: CRT. Dies resultiert in ein paar prägenden Effekten für die Bilddarstellung: Zum einen kommt es zu Scanlines, also sichtbaren, horizontalen und sich bewegenden Linien. Zum Anderen gibt es einen RGB-Verschiebungseffekt, der die Farben des Bildes leicht versetzt und damit etwas verzerrt darstellt. Und eine Wölbung des Bildes, welches sich durch die Wölbung der Bildröhre ergibt. Das wird kombiniert mit der geringeren Farbtiefe von damals, also 8-bit oder 16-bit. Das Game-Design aber eben auch diese visuelle Gestaltung machen für mich Arcade-Spiele aus. Das reduzierte, knackige Gameplay und der Charm der 80er, der auch aktuellen Arcade-Spielen innewohnt, faszinieren mich und versetzen mich gerne wieder zurück in die Kindheit.

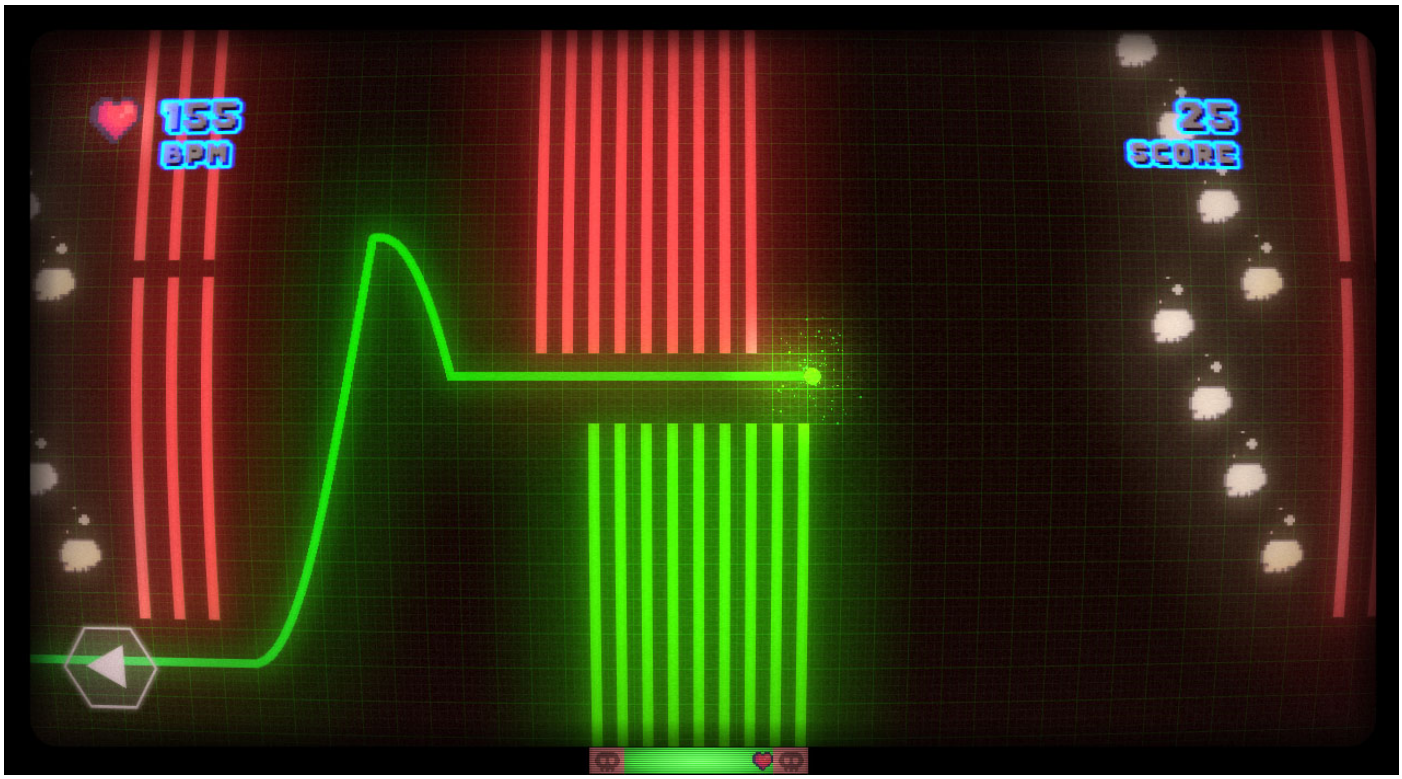


Heart Liner spielt sich mit einer einzigen Taste.

Go Godot Jam 2

Der **Go Godot Jam** ist eigentlich eine Art Festival, welches jährlich über einen Monat veranstaltet wird. Verschiedene Godot Creator veröffentlichen in diesem Zeitraum Content, wie Tutorials oder Interviews. Und es endet dann mit meinem persönlichen Highlight: einem Game Jam. Der wurde diesmal vom 19.11.2021 bis zum 30.11.2021 veranstaltet.

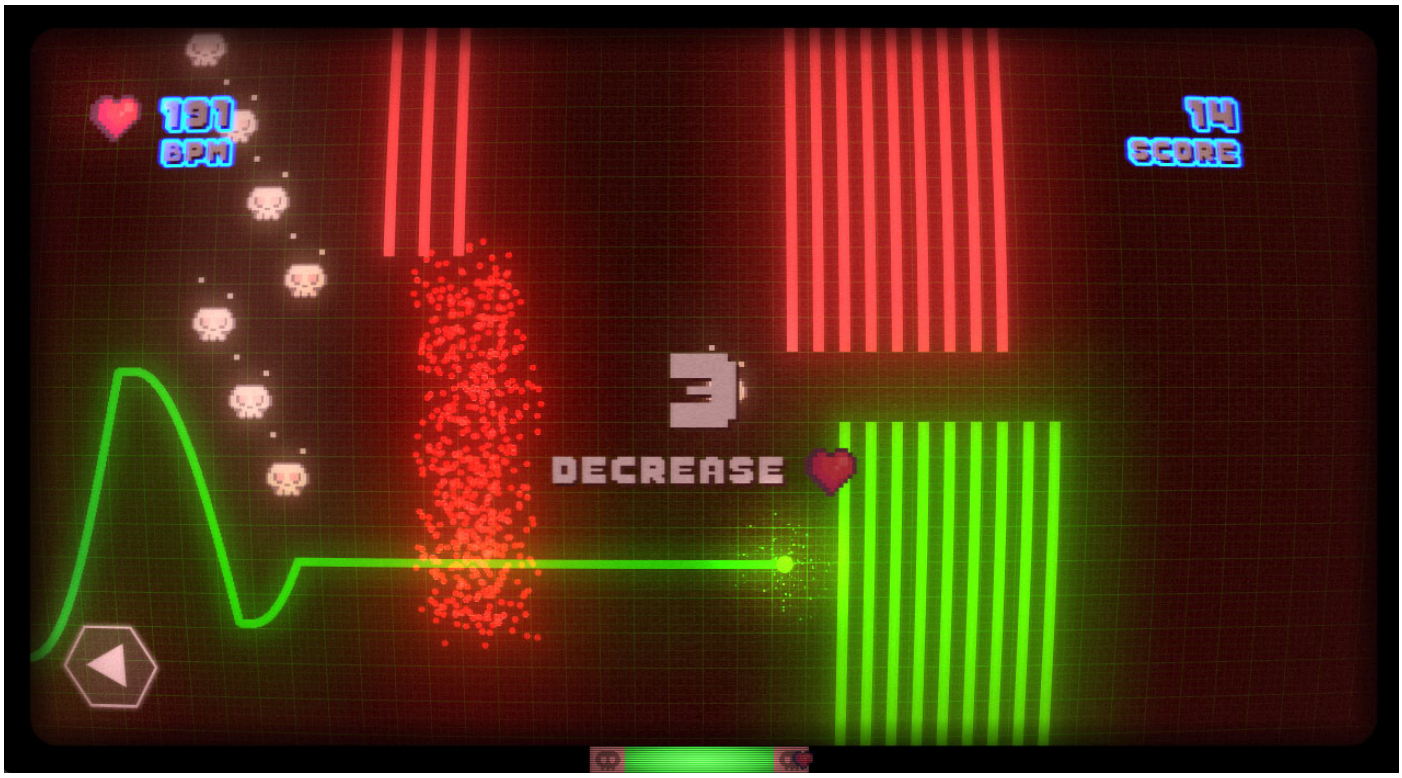
Game Jam bedeutet: In einer vorgegebenen Zeit, zu einem vorgegebenen Thema, ein Spiel entwickeln. Im Vordergrund stehen das Lernen und der Spaß am kreativen Arbeiten. Häufig finden Game Jams in einem Zeitraum von 48 Stunden statt. Beim **Go Godot Jam 2** hatte man stattdessen mehr als 10 Tage Zeit. Eine weitere Besonderheit am **Go Godot Jam 2** war die Förderung des Godot Open-Source Projektes. Godot ist eine Open-Source Spiel-Engine, welche völlig kostenfrei, auch für kommerzielle Zwecke, verwendet werden darf.



Die Steuerung ist simpel – und dennoch tricky.

Seit der ersten verfügbaren Version aus dem Jahr 2014 hat sich sehr viel getan und man kann aufwändige 2D- und 3D-Spiele mit der Engine erstellen. Bei so einem komplexen Open-Source Projekt ist die Finanzierung natürlich immer ein wichtiger Aspekt. Darum wurden beim **Go Godot Jam 2** auch auf unterschiedlichen Wegen Spenden gesammelt wovon 50% direkt an das Godot Team gingen. Weitere 40% gingen an die Content Creator und Organisatoren des Events und weitere 10% wurden als Preise für die Gewinner des Jams verwendet. Gewinnen konnte man den Jam durch die Abstimmung, welche 4 Tage lang nach Abschluss des Game Jams offen war. Jeder Teilnehmer und Organisator konnte die Spiele bewerten.

Bewertet wurden die Spiele in 5 Kategorien: Technische Umsetzung, visuelle Gestaltung, Game-Design, Spielspaß und Umsetzung des Themas. Wie bereits erwähnt, haben Game Jams fast immer ein Thema. Das hat 2 Hauptgründe: zum Einen soll es verhindern, dass die Spiele bereits im Vorfeld fertiggestellt werden, denn das Thema wird erst mit Beginn des Jams verkündet, und zum Anderen soll es den kreativen Prozess anregen, denn es gibt immer viele Arten, wie sich ein Thema interpretieren lässt. Das Thema des **Go Godot Jam 2** war: „Energy Source”, also: Energiequelle.
Links:
https://itch.io/jam/go-godot-jam-2
https://godotengine.org/
https://gogodotjam.com/

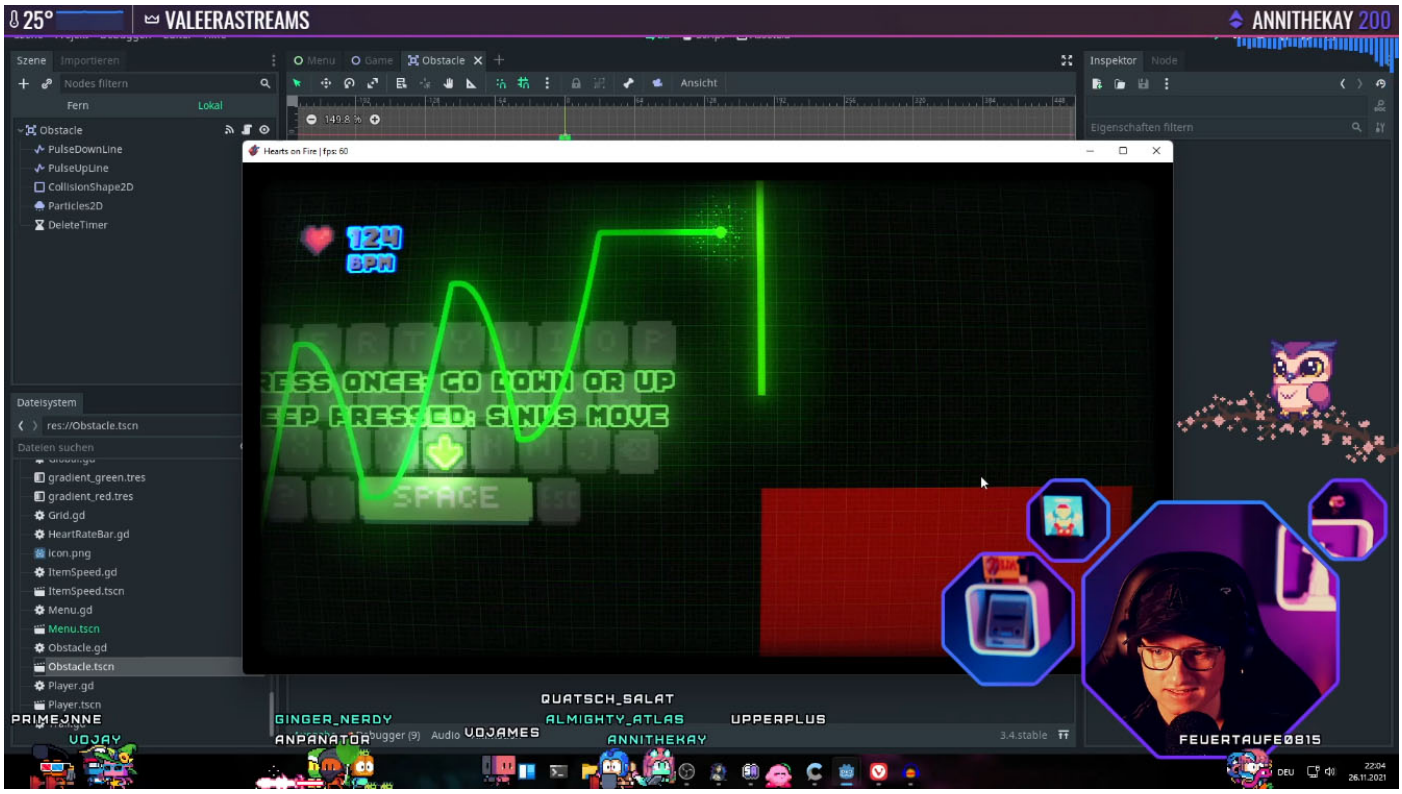


Hindernisse können tödlich sein – oder nützlich – je nach Situation.

Heart Liner – Das Spiel

Mein Projekt für den **Go Godot Jam 2** mit dem Thema „Energy Source" ist ein Arcade-Spiel: **Heart Liner**. Unsere Energiequelle, dass was uns antreibt und Blut durch unseren Körper pumpt und die Versorgung der Organe sichert ist unser Herz. Um die elektrische Aktivität vom Herzen zu messen, verwendet man die Elektrokardiografie, oder kurz: EKG. Die EKG-Kurve wird häufig auf Röhrenbildschirmen dargestellt, erinnert also schon vom Look an den zuvor beschriebenen CRT Effekt. Im Spiel steuert man die EKG-Kurve und muss Hindernissen ausweichen. Jede Aktion, verschnellert den Herzschlag, wohingegen Stillstand auch zur Verlangsamung des Herzschlages führt. Ziel ist es, die Herzfrequenz in der Mitte zwischen zu schnell und zu langsam zu halten.

Besonders dabei ist, dass die Berührung von Hindernissen nicht zwangsläufig negativ ist. Rote Hindernisse erhöhen den Puls und grüne Hindernisse senken ihn. Es kann also Situationen geben indem das gezielte Kollidieren mit einem Hindernis sinnvoll ist. Je länger man durchhält, desto mehr Punkte erreicht man. Die Highscore wird gespeichert und so tritt man stets gegen sich selbst an. Die Steuerung ist denkbar einfach: man verwendet eine Taste um die Richtung der Kurve zu wechseln, entweder es geht hoch oder runter. Die Verwendung von nur einer Taste hat dabei auch den Hintergedanken, dass Spiel möglichst zugänglich und barrierefrei zu machen. Barrierefreiheit in Spielen ist oft ein Thema, welches man in Game Jams vergisst.



Heart Liner wurde im Stream mit der Community erstellt.

Hierzu habe ich eine kleine Anekdote: Bei einem anderen Game Jam an dem ich teilgenommen habe, habe ich ein Spiel entwickelt welches sich, wie für mich gewohnt, mit W, A, S, D und der Maus steuern lässt. Diese Steuerung habe ich intuitiv, ohne große Überlegung festgelegt. In der Voting Phase bekam ich dann die Frage: „Hallo, ich bin Linkshänder, wie kann ich das Spiel steuern?”. Diese einfache Frage hat mir vor Augen gehalten, wie wenig man sich oft damit auseinandersetzt, wie andere Leute das Spiel erleben und wie sehr man mit seinen eigenen Gewohnheiten im Tunnel ist. Deswegen habe ich mir für den **Go Godot Jam 2** selbst auferlegt ein Spiel zu bauen, welches sich mit nur einer Taste steuern lässt.

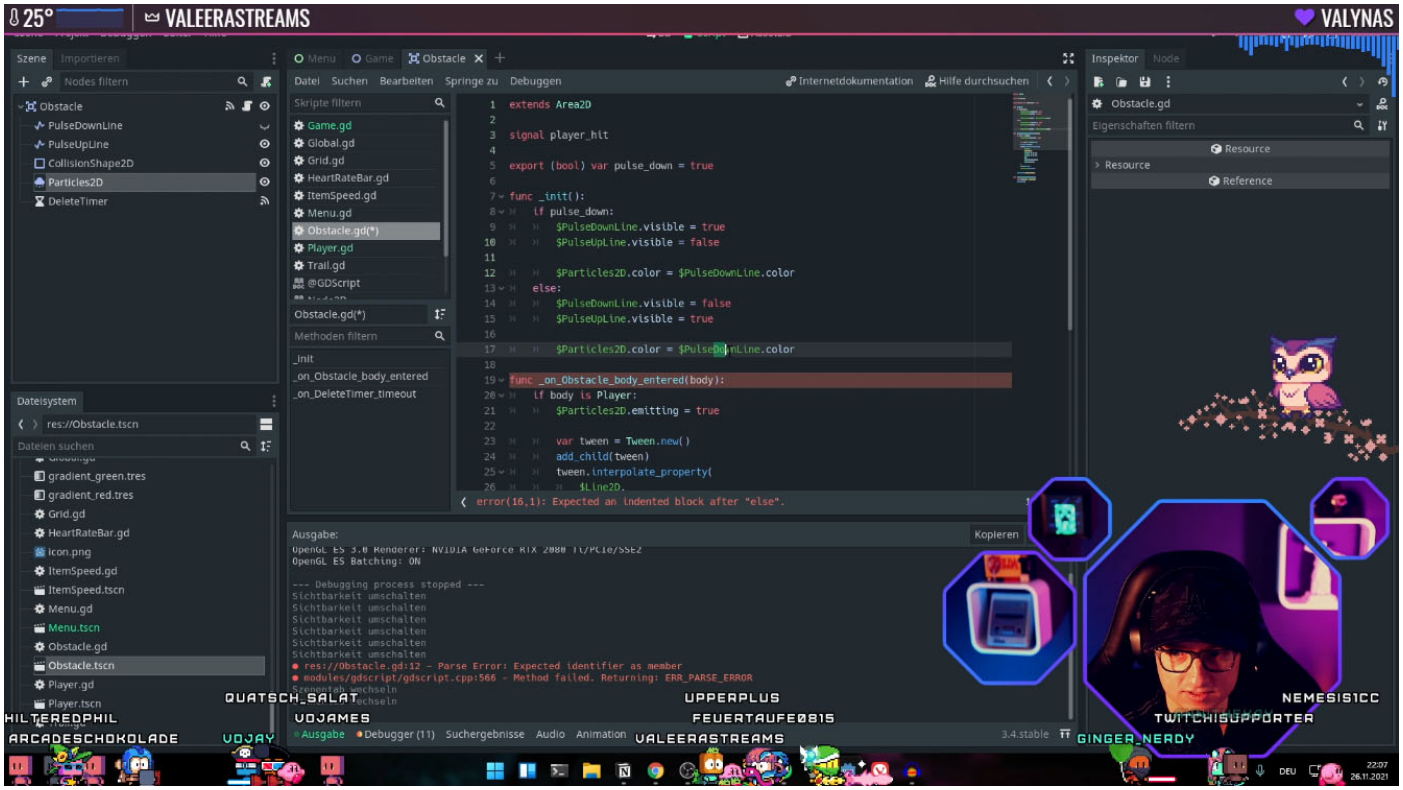
Schaut man nur auf das Game-Design handelt es sich bei **Heart Liner** um ein klassisches Arcade-Spiel. Das Konzept ist einfach, man startet stets von vorne und der Schwierigkeitsgrad steigt knackig schnell. Für die visuelle Umsetzung habe ich versucht das Gefühl zu erzeugen, dass man auf einen alten Röhrenfernseher schaut. Also als ob das Spiel auf dem Bildschirm eines EKG-Gerätes läuft. Zudem habe ich auf einfache Formen, wenig knallige Farben und Neoneffekte gesetzt. Mit dem Synthwave Klängen fühlt man sich so ein wenig in die 80er Jahre zurückversetzt und hat damit ein nostalgisches Arcade-Spiel. Links:
https://vojay.itch.io/heart-liner

Heart Liner – Technik

Das Spiel wurde in der Godot Engine entwickelt. Dabei habe ich GDScript verwendet, welches die eigens für Godot entwickelt Programmiersprache ist. Diese ist ähnlich zu Python und erlaubt es mit relativ einfachen Sprachkonstrukten schnell ans Ziel zu kommen. Man kann Godot auch in der C# Variante verwenden, wodurch man eine mächtigere Sprache erhält, die z.B. auch in der bekannten Unity Engine benutzt wird. Dies ist für größere Projekte definitiv zu empfehlen, aber für Game Jams

setze ich meist auf GDScript.

Das Spiel selbst funktioniert mit prozedural generierten Leveln. Bedeutet: Die Level sind nicht vordefiniert, sondern werden automatisch zur Laufzeit generiert. Das funktioniert wie folgt: Es gibt einzelne Hindernisse. Ein Hindernis kann ein grüner oder roter Balken sein. Diese sind dann in vordefinierten Gruppen zusammengesetzt. Es gibt leichte Gruppen, die z.B. nur einen einzelnen Balken haben oder schwere Gruppen bei denen der Spieler durch ein kompliziertes Konstrukt aus Balken navigieren muss. Zu Beginn des Spiels wird eine Menge von Instanzen jeder dieser Gruppen erzeugt. Dann wird ein Level generiert, indem 6 zufällige Gruppen aus diesen Instanzen aneinandergereiht werden. Ist der Level geschafft, kommen die Gruppen zurück in den Pool und es wird ein neuer Level generiert. Dieses Wiederverwenden von Instanzen wird häufig beim prozeduralen Generieren von Leveln verwendet und nennt sich „Object-Pooling”.



Moderne Engines nehmen einem viel Arbeit ab, programmieren muss man natürlich immer noch!

Der Spieler selbst ist eine Node, welche von der Gravitation beeinflusst wird. Er wird also an den unteren Bildschirmrand gezogen. Mit einer Taste kann der Spieler aber die Gravitation umkehren. Bedeutet: Der Spieler wird nun zum oberen Bildschirmrand gezogen. Um den Arcade-Look zu erzeugen verwende ich 2 Techniken: Zum Einen gibt es einen CRT-Shader, also ein kleines Programm, welches auf der Grafikkarte läuft. Dies erzeugt die Wölbung vom Bild, die Scanlines und die leichte Verzerrung im Bild. Zum Anderen verwende ich eine sogenannte „WorldEnvironment” Node in Godot. Dies ist ein Effekt, der quasi nachträglich auf die gesamte Szene angewendet wird. Hier arbeite ich mit der Eigenschaft „Glow”, welche farbigen Elementen ein Leuchten hinzufügt, je nach dem, wie intensiv die Farbe ist.
Durch diese beiden Techniken, wirkt die Visualisierung einheitlich und hochwertig obwohl im Prinzip nur einfache rechteckige Farbflächen verwendet werden. Der Code vom Spiel umfasst ca. 570 Zeilen.

Dieser teilt sich in 10 Klassen auf. Die größten Teile sind die Spielelogik mit 117 Zeilen und die prozedurale Levelgenerierung mit 103 Zeilen.
Links:
https://gameprogrammingpatterns.com/object-pool.html

Im Vojay-Stream ist GameDev regelmäßig Thema.

Heart Liner – Umsetzung und Stream

Eine weiter Besonderheit an diesem Projekt war, dass es fast komplett live im Stream auf Twitch entwickelt wurde. Insgesamt habe ich 3 Streams mit der Entwicklung des Spiels verbracht. Die Dauer der Streams war ca. 5 – 6 Stunden, also ist das eine Entwicklungszeit von 18 Stunden. Hinzu kommt noch Zeit außerhalb des Streams. Hier habe ich hauptsächlich Fehler behoben und die Levelgenerierung verfeinert. Außerdem wurden noch ein paar Stunden in die Präsentation des Spiels gesteckt, bedeutet Beschreibungstexte verfassen und das Spiel auf itch.io veröffentlichen. Insgesamt wurden also ca. 25-30 Stunden für das Spiel benötigt.
Ein Spiel im Stream live vor Zuschauern zu entwickeln bedeutet vor allem, einen kurzen Feedback-Zyklus zu haben. Das geht schon mit der Idee des Spiel los, die basiert auf einem Brainstorming gemeinsam mit den Zuschauern. Auch das Gameplay basiert auf Ideen und Feedback der Zuschauer. Ein kurzes „hey, kann man das nicht auch so machen?” im Chat reicht um eine neue Richtung einzuschlagen. Die Herausforderung hierbei: konzentriert bleiben, einen kühlen Kopf bewahren und einen guten Mittelweg finden um das Ziel nicht aus den Augen zu verlieren. Insgesamt ist es aber eine große Freude, da ich so Inspiration bekomme und den Zuschauern auch die Spieleentwicklung näher bringen kann.

Natürlich dauert die Entwicklung eines Spiels so zwar länger, aber da ich bereits Erfahrung mit der Godot Engine habe, kann ich dies gut ausgleichen und am Ende hat man ein Spiel, welches bereits durch Feedback verbessert wurde.
Links:
https://www.twitch.tv/videos/1210050047
https://www.twitch.tv/videos/1216498451
https://www.twitch.tv/videos/1216504051

Neben dem Stream gibt es einen Discord-Channel, der Hilfe bei GameDev-Fragen liefert.

Über mich und meinen Stream

Ich bin Volker, oder auch Vojay, Kind der 90er, stolzer Papa und Softwareentwickler in der Gaming Branche. Ich habe Informatik studiert und arbeite seit mehr als 10 Jahren bei InnoGames in Hamburg. Obwohl ich im Stream viel Spieleentwicklung mache, kümmere ich mich bei der Arbeit eher um verteilte Backendsysteme die große Mengen von Daten verarbeiten. Hier kommen Technologien wie Java, SQL, Python und viele Projekte aus dem Bereich Big Data zum Einsatz. In meinem Stream versuche ich zwei Mal pro Woche meinen Zuschauern das Programmieren und die Spieleentwicklung näher zu bringen. Hier entwickeln wir Spiele für Game Jams oder bauen z.B. einen Bot in Java, der den Stream um Features wie eine sprechende Bauchbinde, Lichtsteuerung oder andere Spielereien bereichert. Derzeit nehmen wir an einem Game Jam Teil, bei dem wir ein Spiel für blinde Spieler und Spielerinnen entwickeln. Wer hierzu mehr erfahren möchte, kann uns gerne auf Twitch besuchen.
Links:
https://www.twitch.tv/vojay
https://vojay.stream/
https://www.instagram.com/vojaytogo/

Der Vozone Discord

Neben dem Stream betreibe ich gemeinsam mit meinen Mods einen Discord Server: Die Vozone. Neben Quatsch und Schabernack geht es auch hier häufig um das Thema Programmieren und Spieleentwicklung. Wir haben sogar einen Supportbereich. Hier kann ein Ticket erstellt werden und jedes Mitglied des Servers kann dieses Ticket sehen und helfen.
Die Idee ist eine Plattform zu schaffen, bei der erfahrene Programmiere Neulingen helfen und auch zu motivieren sich mehr mit dem Thema zu beschäftigen.
Links:
https://discord.vojay.stream/

Links (noch mal zusammengefasst)

Go Godt Jam 2
https://itch.io/jam/go-godot-jam-2
Godot-Engine
https://godotengine.org/
Godot-Jam
https://gogodotjam.com/
Heart Liner – Das Spiel
https://vojay.itch.io/heart-liner

Object-Pool-Programming-Pattern
https://gameprogrammingpatterns.com/object-pool.html
Die Streams von der Erstellung von Heart Liner
https://www.twitch.tv/videos/1210050047
https://www.twitch.tv/videos/1216498451
https://www.twitch.tv/videos/1216504051

Der Twitch-Kanal von vojay
https://www.twitch.tv/vojay
https://vojay.stream/
Instagram von vojaytogo
https://www.instagram.com/vojaytogo/
Discord-Channel zum Stream
https://discord.vojay.stream/
Date Created
11. Januar 2022
Author
vojay