Makerspace Arcade

Unser Makerspace-Wahlkurs präsentiert stolz ein einzigartiges Projekt: einen selbstgebauten Arcade-Automaten.

Unsere jungen Schülerinnen und Schüler der 5. und 6. Klassen haben mit großer Begeisterung ihre eigenen Computerspiele entwickelt. Um ihnen die Gelegenheit zu geben, ihre individuellen Kreationen auf spielerische Weise der gesamten Schule vorzustellen, haben wir den Spielautomaten konstruiert und mit den entwickelten Spielen ausgestattet.

Wir haben auch dafür gesorgt, dass jeder drankommt, indem die Spielzeit über ein Usermanagement individuell begrenzt wird. Das handgefertigte Holzgehäuse wurde geschickt mit einem alten Schul-PC kombiniert. Durch den Einsatz des hauseigenen Lasercutters wurde das Logo des Automaten in die Realität umgesetzt. Wir haben handwerkliches Geschick und kreative Softwarelösung vereint, um unseren Makerspace-Enthusiasten eine Bühne zu geben.

Inhaltsverzeichnis

1. Fachliche Kurzfassung

Im Folgenden widmen wir uns mit der Entwicklung, wobei der Fokus auf der hardwareseitigen und softwareseitigen Umsetzung liegt. Wir beginnen mit der Wahl eines Schul-PCs als zentrales Element des Automaten, nachdem die ursprüngliche Idee, einen Raspberry Pi zu verwenden, aufgrund der anhaltenden Chip-Krise und der damit verbundenen hohen Kosten verworfen wurde. Das Gehäuse des Automaten besteht aus regionalem Holz, welches aus dem eigenen Wald stammt. Dieses verarbeiteten wir in einem eigenen Design, um die Stabilität und Robustheit des Automaten sicherzustellen.

Der Softwareteil der Arbeit beschäftigt sich mit der Wahl der Spieleentwicklungsplattform Makecode Arcade und den Herausforderungen bei der Integration in das gewählte Betriebssystem Ubuntu. Das Usermanagement erfolgt über die Lernplattform Mebis Bayern, wobei ein QR-Code-basiertes Login-System implementiert wurde. Weiter beschreiben wir dann die Schwierigkeiten bei der Controllerintegration und die Lösung mithilfe von AntimicroX als Übersetzungssoftware. Zusätzlich wurde ein Bewertungssystem für die Spiele integriert, um einen Wettbewerbsanreiz für Spieleentwickler zu schaffen. Die finale Menüsoftware wurde dann letztendlich überarbeitet, um eine verbesserte Performance und ein ansprechendes Nutzungserlebnis zu gewährleisten.

Ein besonderer Fokus liegt auf nachhaltigen Aspekten des Projekts, darunter die Verwendung von regionalem Holz, die automatische Abschaltung des Rechners außerhalb der Nutzungszeiten und das Wiederverwenden eines alten Schul-PCs. Somit wollen wir auch die ökologische und ressourcenschonende Ausrichtung unseres Projekts betonen.

2. Motivation

Die Motivation hinter der Entwicklung unseres Arcade-Automaten entspringt einer tiefen Leidenschaft für Kreativität, Technologie und den Wunsch, innovative Projekte in unserem Makerspace-Wahlkurs zu realisieren. Unser Ziel war es, nicht nur einen bloßen Spielautomaten zu bauen, sondern ein interaktives Erlebnis zu schaffen, das die kreativen Fähigkeiten unserer Wahlkursmitglieder auf einzigartige Weise präsentiert. Dabei kam die Idee, den Automaten in unserem Schulumfeld zu platzieren, um die individuell entwickelten Spiele unserer Mitglieder einem breiteren Publikum zugänglich zu machen.

Die Entscheidung, Joysticks, leuchtende Buttons und ein eigenes Logo einzubauen, zeigt unsere Hingabe für authentische Arcade-Ästhetik. Der Weg zur Realisierung dieses Projekts war geprägt von Herausforderungen, von der Wahl der Hardware bis zur Verwendung einer Lernplattform als Datenbank. Doch diese Hürden haben uns nicht nur technisch, sondern auch teamübergreifend wachsen lassen.

3. Hardware

Erste Ideen für die Umsetzung

Unser erster Ansatz war einen Raspberry Pi, ein kleiner Mini-PC auf einer einzelnen Platine, als Herzstück des Automaten zu verwenden. Dies ist nun einmal eine populäre Wahl und wir hatten auch schon viel Erfahrung mit Raspberry Pis in anderen Projekten gesammelt. Als wir jedoch vorhatten einen zu kaufen, da wir keinen RPI für das Projekt übrighatten, bekamen wir schnell einen Schlag ins Gesicht. Denn als wir mit dem Projekt 2021 starteten, war die Chip-Krise noch groß im Gange und ein Exemplar kostete gut 120€. Wir schauten uns also nach einer besseren Alternative um und wurden auch fündig. Unsere Schule bekam passend zu der Zeit neue PCs für die Computerräume. Die alten hatten also keinen Nutzen mehr. Da kam uns die nachhaltige Idee, einfach einem Rechner mit unserem Automaten einen neuen Zweck zu geben.

Für das Gehäuse haben wir nach ein bisschen Inspiration gesucht. Dabei stießen wir auf ein Design von Rasmus Kønig Sørensen namens MAME Arcade Cabinet. Er zeigte eine mögliche Umsetzung für ein Holzgehäuse. Jedoch konnten wir uns nicht mit der Stabilität und mit dem ein oder anderen Designelement anfreunden, weshalb wir uns selbst ein Design überlegt haben.

Regionales Holz

Das Holz des Automaten stammt nicht von irgendeinem Baum, sondern regional aus unserem eigenen Wald. Die gefällte Fichte haben wir zu der 2 km entfernten, wasserbetriebenen Sägerei gebracht, um sie dort zu den im Automaten verwendeten Brettern zu sägen. Ursprünglich waren diese Bretter nicht für den Automaten vorgesehen, sondern lagen fünf Jahre lang, ohne eine bestimmte Verwendung zu haben, auf dem Heuboden unseres Hofes.

Planung

Zurück in der Gegenwart haben wir uns den Plan von Rasmus Kønig Sørensen noch einmal genauer angeschaut und unsere eigene Skizze erstellt. Da der Automat in unserer Schule aufgestellt werden soll, wo der ein oder andere vielleicht nicht ganz so sanft mit Dingen umgeht, war uns von Anfang an klar, dass wir den Automaten strapazierfähig bauen müssen.

Bedienung des Automaten

Da es sich um einen Arcade-Automaten handelt, stand außer Frage, dass wir Joysticks und leuchtende Buttons benötigen. Also entschieden wir uns, einfach ein Kit aus dem Internet zu besorgen. Somit hatten wir gleich 2 Joysticks, 2 Controller-Boards und 10 Buttons in verschiedenen Farben. Nach einem kleinen, provisorischen Test in einem Karton waren wir voller Zuversicht, dass alles funktioniert und konnten mit dem Bau beginnen.

Grundgerüst

Um die benötigte Stabilität zu erreichen, sind wir von unserer ersten Skizze abgewichen und haben uns entschieden, statt eines einzigen Rahmens im unteren Bereich noch einen zweiten unter der Tastatur und dem Herzstück des Automaten, dem Computer, einzuplanen. Die Wahl des Holzes war kein Hexenwerk, da wir bereits alle Maße hatten. Sie fiel auf das oben beschriebene Fichtenholz. Das Grundgerüst stand nach nicht allzu langer Zeit und war dank unserer Holzwerkstatt recht schnell vollendet.

Tastatur

Wir hatten uns die Buttons und die Joysticks noch nicht besonders genau angeschaut und stellten fest, dass wir durch unsere Wahl des 3,5 cm dicken Holzes zwar die Stabilität garantierten, aber das Gewinde der Buttons zu kurz war und die Joysticks nur ein kleines bisschen herausstehen würden. Somit mussten wir die bereits gebohrten Löchern noch optimieren, indem wir die Buttons und Joysticks hinten eingelassen haben.

Logo aus dem Lasercutter

Um das Gehäuse abzurunden und endlich unseren bisher ungenutzten Lasercutter einzusetzen, verwarfen wir unsere ursprüngliche Idee, das Logo direkt auf den Automaten zu malen, und entschieden uns für die präzise Gravur des Logos mithilfe des Lasercutters.

4. Software

Makecode Arcade

Durch die vielfältigen Möglichkeiten wurde eine Plattform namens Makecode Arcade bei unseren Makern schnell beliebt. Die von Microsoft entwickelte Game Engine ermöglicht auf ähnliche Weise wie auf Scratch, einer Game Engine des MITs, simple Spiele auf intuitive Art mit Code-Blöcken zu

verwirklichen. Das Ergebnis kann dann auf der Website oder kleinen Spielkonsolen gespielt werden. Wir hatten uns es also ganz einfach vorgestellt, die Spiele auf unserem Automaten zum Laufen zu bringen. Dies wird sich schon bald als Trugschluss herausstellen.

Ausprobieren von RetroPie

Nachdem bekannt war, auf welche Weise Spiele unterstützt werden müssen, machten wir uns erste Gedanken für eine mögliche Umsetzung eines Menüs. Die Mindestanforderungen waren eine Auswahl der Spiele und eine Authentifizierung für eine Zeitbegrenzung. Die erste Idee, die uns in den Sinn kam, war die durchaus bekannte Linux-Distribution RetroPie. Es ist ein für Retrospiele spezialisiertes Betriebssystem, dass hauptsächlich für den Raspberry Pi erschaffen wurde. Wir haben es also auf den Rechner installiert und ausprobiert.

Das Menü war akzeptabel, jedoch konnten wir die Spiele nicht vernünftig starten. Nach einer langen Recherche wurde uns dann bewusst, dass Makecode Arcade die Spiele nur für ARM und nicht für x86 kompilieren kann. Dies war der zweite Schlag ins Gesicht. Sowohl besagte Minispielkonsolen als auch der Raspberry Pi basieren auf der ARM-Architektur. Unser Rechner kommt allerdings wie fast alle PCs mit x86 da her. Die Spiele können also nicht ausgeführt werden. Dazu kommt noch, dass es wahrscheinlich keine elegante Lösung gibt, um den Spieler zu identifizieren. Es gäbe zwar die Möglichkeit, ein Python-Skript beim Start ausführen zu lassen, dies würde sich aber als sehr limitierend herausstellen.

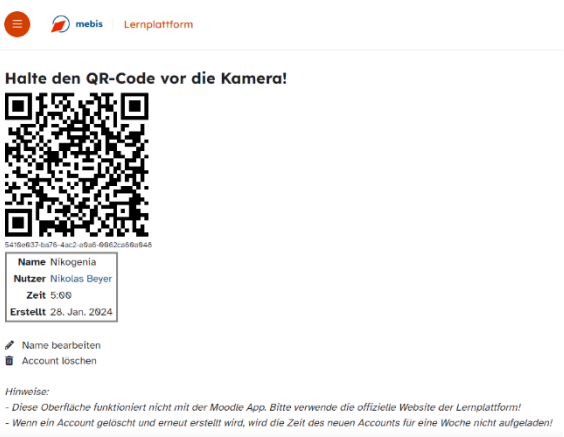
Betriebssystem

Es musste also etwas anderes sein. Die Ideen, dass Spiel auf irgendeine Weise zu dekompilieren und dann für x86 neu zu kompilieren oder einen Simulator laufen zu lassen, waren auch nicht vielversprechend. Also blieb uns nur noch übrig, die Spiele im Browser laufen zu lassen. Dies machte es zwar nicht einfach, jedoch waren wir uns sicher, dass es umsetzbar ist. Die Menüfrage war allerdings immer noch offen. Wir machten uns somit weiter auf die Suche.

Unser Erfolg hielt sich dabei in Grenzen. Schließlich gaben wir die Suche auf und entschieden uns für eine eigene Lösung. Der benötigte Aufwand ist dabei natürlich viel größer, jedoch genießen wir gleichzeitig den Vorteil der Flexibilität. Es galt nun also die genaue Umsetzung zu planen. Schritt 1 war dabei die Wahl des Betriebssystems. Und was eignet sich für solch ein Bastelprojekt besser als ein Linux. Ubuntu war letztendlich der Sieger, da wir bereits ein wenig Erfahrung mit der Distribution hatten. Auch wenn sich diese Wahl später als nicht ganz optimal entpuppte, ermöglichte sie uns fortzufahren.

Datenbank über Lernplattform Mebis Bayern

Der zweite Schritt der Planung war das Usermanagement. Erste Überlegungen basierten auf ausgedruckten QR-Codes, die ausgeteilt werden. Doch wirkte der Ansatz schnell unpraktikabel und wir wollten eine digitale Lösung. Es gab zwei Optionen: Entweder entwickeln wir eine eigene Website und Datenbank für den Login oder wir nutzen die Lernplattform der Schule. Letzteres hat vor allem den Vorteil, dass jeder Schüler nur ein Account besitzen kann. Dies allein hat gereicht, um uns von der Option zu überzeugen. Es galt also die Lernplattform Mebis tiefer zu erforschen.



Die Plattform basiert auf dem Open Source Projekt Moodle und ermöglicht Lehrkräften, auf einfache Art den Unterricht digital zu gestalten. Ein Feature ist zum Beispiel eine Datenbank, die dafür gedacht ist, dass jeder Schüler seine Arbeiten abgibt. Wir bauten dies geschickt um, dass die Abgabe nur aus dem Spielernamen besteht und beim Erstellen eines Eintrags eine Spieldauer und eine UUID, also eine zufällige und eindeutige ID, mitgeneriert wird. Diese wird dann beim Einsehen der Daten als QR-Code gerendert. Noch ein Abgabelimit von 1 für jeden Schüler konfiguriert und voila, jeder Schüler hat genau einen QR-Code.

Jetzt muss das Ganze noch automatisch gelesen werden können. Glücklicherweise kann bei Moodle, der Open-Source-Software, auf der Mebis basiert, alles über eine REST-API durchgeführt werden. Wir haben also aus Sicherheitsgründen einen exklusiven Mebis-Account erstellt und dann mithilfe dessen Tokens auf die API zugegriffen.

Die letzte Frage war noch, was passiert, wenn die Internetverbindung ausfällt. Um einen Fehler auszuschließen, haben wir ein Cache-System eingebaut. Das Ergebnis einer Anfrage wird in einer JSON-Datei abgelegt, sodass das Programm nach einem Neustart direkt schon alle Daten hat, auch wenn die Datenbank nicht erreicht werden kann. Natürlich muss dabei auch noch beachtet werden, was passiert, wenn die Zeit eines Spielers sich lokal verändert hat. Dafür werden nach jedem Request die lokalen Daten mit der Datenbank verglichen und Änderungen hochgeladen. Nach etwas durchfuchsen in der Dokumentation war aber unsere Datenbank nun vollständig einsatzbereit.

Zeitmodell

Beim Zeitmodell haben wir uns überlegt, dass jeder Spieler 5 Minuten die Woche bekommt. Auf diese Weise können Schüler ihre Zeit an unterschiedlichen Tagen nutzen. Makerspace Mitglieder, die ein Spiel entwickelt haben, können ihr eigenes Spiel jedoch unbegrenzt spielen. Wir, Valentin und Nikolas, haben uns natürlich erlaubt, ein dauerhaft unlimitiertes Zeitkonto zu besitzen.

Erster Menü-Entwurf

Die letzte große Frage betrifft die Menüsoftware selbst. Wie und womit soll sie programmiert werden. Wieder einmal wegen einer Menge Erfahrung fiel die Wahl hier auf die Programmiersprache Python in Kombination mit der Grafikbibliothek Pygame. Auch wenn diese wegen ihrer schlechten Performance nicht der ideale Lösungsweg für ein Userinterface ist, ist es doch eine ganz plausible Option. Der Vorteil hierbei gegenüber beispielsweise einer Webanwendung ist eindeutig die größere Kontrolle über das System. So haben wir es schließlich mit simplen Controllern als Eingabequelle zu tun, was im Browser schon einmal Ärger machen könnte. Es ging also mit dem ersten Entwurf los.

Login per QR-Code

Der Rechner hat passenderweise gleich eine Webcam eingebaut. Somit konnten wir mithilfe von den Python-Bibliotheken „OpenCV“ und „PyZBar“ die Kamera einlesen und QR-Codes finden. Um zudem eine Vorschau des Kamerabildes zu ermöglichen, mussten wir ein OpenCV Image zu einem

Pygame Surface umwandeln. Richtig skaliert und noch eine Statusleiste darunter und schon ist ein schöne Login Oberfläche entstanden.

Spielstart im Browser

Eine weitere Challenge war eine Website von Python aus zu starten und verwalten. Der erste Versuch bestand daraus, eine Browser Engine laufen zu lassen und dann das Bild an Pygame weiterzuleiten. Dabei sind wir jedoch nicht weit gekommen. Einerseits hatten die externen Engines teilweise Schwierigkeiten, die Spiele korrekt darzustellen, und andererseits unterstützten sie die Controllerinputs nicht vollständig.

Die nächste Alternative war dann einen richtigen Browser zu starten. Erst wollten wir dafür die Python-Bibliothek „Webbrowser“ verwenden, jedoch stellte sich hierbei heraus, dass es keine Option gibt, dem Browser weitere Details wie den Vollbildmodus mitzugeben. Uns blieb also nichts anderes übrig als manuell einen neuen Prozess zu starten und den Browser selbst zu verwalten. Bei der Browserwahl entschieden wir uns letztendlich für Chromium, da es gute Startargumente zur Verfügung stellt. Und wenn die Zeitabgelaufen ist oder der Spieler keine Lust mehr hat, braucht einfach der Prozess gekillt werden und man befindet sich wieder im Menü.

Zeitanzeige

Der Browseransatz brachte eine neue Challenge mit sich. Wie wird dem Spieler angezeigt, wie viel Zeit er noch übrig hat. Da wir keine Kontrolle über das Bild des Browsers hatten, mussten wir ein weiteres Fenster öffnen. Dieses wird als „Always on Top“ und ohne Rahmen definiert. Dadurch ist es möglich, trotz des Vollbildmodus des Browsers die Zeit auf dem Bildschirm anzuzeigen.

AntimicroX als Übersetzungssoftware

Mit den Controllern lief auch nicht alles glatt. Der Browser konnte mit diesen bereits hervorragend umgehen. Allerdings machte uns es Python nicht so einfach. Wir mussten die Eingaben nämlich systemweit auslesen. Dies liegt daran, dass die Menüsoftware während der Laufzeit der Spiele nicht im Fokus ist und somit normalerweise keine Inputs erhält. Wir mussten jedoch wissen, wann der Spieler das Spiel schließen möchte, und somit ging die Reise los.

Erst wollten wir mithilfe einer kleinen, schlecht dokumentierten Bibliothek namens „Inputs“ die Controller einlesen. Dies war jedoch enorm unzuverlässig und unpraktikabel. Die Bibliothek erforderte auf dem Mainthread von Python zu laufen. Dieser war jedoch bereits für alles andere reserviert. Es dauerte also, bis wir zu der möglichen Lösung kamen, mithilfe von Python Multiprocessing eine weitere Python-Instanz zu starten. Doch auch so war der Ansatz mangelhaft und es musste etwas anderes her.

Da sowieso schon in der Ideenkiste aufzufinden war, weitere Spieltypen zu unterstützen, war eine Übersetzung von Controllereingaben zu Tastatureingaben die beste Option. Es stellte sich heraus, dass es genau für das ein GUI-Programm namens AntimicroX gibt. Wir probierten es also aus und nach etwas Konfiguration lief es genau so, wie wir uns es vorgestellt haben.

Spielebewertungen

Als kleines extra Feature haben wir auch noch ein Bewertungssystem implementiert. So können Spieler, nachdem sie ein Spiel ausprobiert haben, eine Sternebewertung von 1 bis 5 abgeben. Auf diese Weise wollten wir auch ein wenig den Wettbewerb unter den Spieleentwicklern ankurbeln. Um technisch sicherzustellen, dass jeder nur einmal ein Spiel bewerten kann, haben wir die Bewertungen bei den Spielern im Account gespeichert. Somit liegen die Daten auch gleich in der Datenbank und können nicht so schnell verloren gehen.

Finale Umsetzung der Menüsoftware

Mit der Zeit hat sich die erste Version der Menüsoftware als problematisch gezeigt. Wir haben also schweren Herzens die Entscheidung getroffen, noch einmal alles zu löschen und neu anzufangen. Dabei sind wir jedoch in vieler Hinsicht anders vorgegangen. So haben wir den Fehler korrigiert, Variablen, Klassen, Funktionen etc. auf Deutsch zu benennen. Die Idee war ursprünglich, den Code auch für Schüler mit weniger ausgeprägten Englisch-Kenntnissen zugänglich zu machen. Heute ist aber klar, dass es absolut keinen Sinn ergibt, Code in einer anderen Sprache als Englisch zu schreiben. Zudem haben wir durch eine Änderung im Renderprozess die Performance deutlich erhöhen können. Denn Pygame unterstützt eben nun einmal nur CPU-Rendering, was einen dazu zwingt, gut auf die Geschwindigkeit zu achten.

Der Aufwand hat sich jedoch definitiv gelohnt. Durch das Vorschau-Video im Hintergrund und vielen kleinen Animationen ergibt sich ein flüssiges Nutzungserlebnis. Wenn zum Beispiel für eine Weile nichts passiert, wird der Idle-Screen angezeigt, auf dem man das Video optimal sehen kann. Sobald nun eine Taste betätigt wird, kann im Menü das gewünschte Spiel ausgewählt werden. Ist dies getan, wird man aufgefordert, den QR-Code vor die Kamera zu halten. Folglich kann das Spielerlebnis starten.

Um sicher zu stellen, dass auch technisch alles glatt läuft, haben wir noch implementiert, dass bei einem Crash sich das System automatisch neustartet und eine E-Mail mit der Fehlernachricht an uns gesendet wird. Mithilfe eines Remotedesktop- und SSH-Servers kann der Automat auch vollständig aus der Ferne gewartet werden. Wir sind sogar so weit gegangen, ihn in ein eigenes VPN-Netzwerk einzubinden, sodass er aus der ganzen Welt erreichbar ist.

Nachhaltigkeit ist uns auch wichtig. Daher haben wir mithilfe eines CRON-Jobs und einer BIOS-Einstellung dafür gesorgt, dass der Rechner an Wochentagen um 7 Uhr automatisch hochfährt und um 17 Uhr sich auch wieder ausschaltet. Auf diese Weise konnten wir die Zeit, in der Strom verbraucht wird, von 24 * 7 = 168 Stunden die Woche auf 10 * 5 = 50 reduzieren.

5. Ergebnis

3 Jahre später ist es so weit. Der Automat ist einsatzbereit und wird in der Aula unserer Schule aufgestellt. Es ist einfach schön zu sehen, wie wir einen Hingucker geschaffen. Um noch ein

lebendigeres Bild zu erhalten, haben wir einen kleinen Trailer produziert:
https://www.youtube.com/watch?v=8V85Eo9Pjnc

6. Fazit

Das Projekt des selbstgebauten Arcade-Automaten war eine beeindruckende Reise der Kreativität und technischen Umsetzung. Insgesamt spiegelt es nicht nur die technische Expertise unseres Makerspace wider, sondern auch unsere Freude an der Zusammenarbeit und der kreativen Umsetzung von Ideen.

Wir hoffen, dass wir mit diesem Projekt, unserer Schulgemeinschaft etwas Schönes mitgegeben haben und der Welt zeigen konnten, wie ein solches Meisterwerk umgesetzt werden kann.

Nikolas Beyer und Valentin Sutter

7. Quellen

GitHub Repository - https://github.com/Nikogenia/MspArcade

Bodensee-Gymnasium Lindau - https://bodensee-gymnasium.de/

Microsoft Makecode Arcade - https://arcade.makecode.com/

MAME Arcade Cabinet - Rasmus Kønig Sørensen

EG STARTS 2 Arcade Kit - https://www.amazon.de/-/en/Q3-MM7G-FS21/dp/B06WWRKGGD/

AntimicroX - https://github.com/AntiMicroX/antimicrox

Lernplattform Mebis Bayern - https://lernplattform.mebis.bycs.de/

Ubuntu - https://ubuntu.com/

Python - https://www.python.org/

Pygame Community Edition - https://pyga.me/

OpenCV - https://opencv.org/

PyZBar - https://pypi.org/project/pyzbar/

Pynput - https://pypi.org/project/pynput/

Nikocraft - https://pypi.org/project/nikocraft/

ChatGPT - https://chat.openai.com/

OMTech Lasercutter - https://omtechlaser.de/products/50w-co2-laser-graviermaschine-cutter-turbo-535/

Alle Links wurden zuletzt am 27.01.2024 aufgerufen.

Ein Makerspace Projekt von

Nikolas Beyer und Valentin Sutter

Bodensee-Gymnasium Lindau – 2021-2024