

|  |
| --- |
| Projektarbeit |
| Wibilea Adventure Game |

|  |  |
| --- | --- |
| Autoren | Noel Wangler & |
| Version | 0.7 |
| Beruf | Lernende Informatiker |
| Firma | Wibilea |
| Lehrjahr | 1 |
| Semester | 2 |
| Berufsbildner | Rubén Fructuoso |
| Erstelldatum | 13.01.2020 |



Inhaltsverzeichnis

[1 Die Idee 5](#_Toc40881606)

[1.1 Brainstorming 5](#_Toc40881607)

[1.1.1 Ideensammlung 5](#_Toc40881608)

[2 Vorgehensweise 5](#_Toc40881609)

[3 Informieren 5](#_Toc40881610)

[3.1 Game Engine 5](#_Toc40881611)

[3.2 Design 6](#_Toc40881612)

[3.3 Was ist möglich 6](#_Toc40881613)

[4 Planung 6](#_Toc40881614)

[4.1 Zeitplan 7](#_Toc40881615)

[5 Umsetzung 9](#_Toc40881616)

[5.1 Grafisches 9](#_Toc40881617)

[5.2 Photoshop einrichten 11](#_Toc40881618)

[5.3 Photoshop Tools 13](#_Toc40881619)

[5.3.1 Auswahl Tool [M] 13](#_Toc40881620)

[5.3.2 Verschieb Werkzeug [V] 14](#_Toc40881621)

[5.3.3 Buntstift/Pinsel [B] 14](#_Toc40881622)

[5.3.4 Radiergummi [E] 14](#_Toc40881623)

[5.3.5 Tastenkürzel 14](#_Toc40881624)

[5.4 Godot 14](#_Toc40881625)

[5.4.1 Wie sind die Dateien verknüpft 15](#_Toc40881626)

[5.4.2 Godot-Script 15](#_Toc40881627)

[5.5 Eigenes Spiel 16](#_Toc40881628)

[5.5.1 Erstes Script 19](#_Toc40881629)

[5.5.2 Animation 20](#_Toc40881630)

[5.5.3 Tileset 21](#_Toc40881631)

[5.5.4 Signals 22](#_Toc40881632)

[5.5.5 Szenenwechsel 23](#_Toc40881633)

[5.6 Allgemeine Einstellungen 25](#_Toc40881634)

[5.7 Steuerung 27](#_Toc40881635)

[5.8 Pausen-menu 27](#_Toc40881636)

[6 Minigames 28](#_Toc40881637)

[6.1 Informatikgame 28](#_Toc40881638)

[6.2 Automatik Minigame 31](#_Toc40881639)

[6.3 Polymechanik Minigame 34](#_Toc40881640)

[6.4 Konstruktion Game 36](#_Toc40881641)

[7 Charakter Auswahl 40](#_Toc40881642)

[8 Veröffentlichen 42](#_Toc40881643)

[8.1 Android 42](#_Toc40881644)

[8.2 Windows 44](#_Toc40881645)

[8.2.1 Export 44](#_Toc40881646)

[8.2.2 Windows Installation 44](#_Toc40881647)

[8.3 IOS 44](#_Toc40881648)

[9 Arbeitsjournal 45](#_Toc40881649)

[9.1 07.01.2020 45](#_Toc40881650)

[9.2 08.01.2020 45](#_Toc40881651)

[9.3 13.01.2020 46](#_Toc40881652)

[9.4 28.01.2020 46](#_Toc40881653)

[9.5 29.01.2020 47](#_Toc40881654)

[9.6 10.02.2020 47](#_Toc40881655)

[9.7 17.02.2020 47](#_Toc40881656)

[9.8 03.03.2020 48](#_Toc40881657)

[9.9 04.03.2020 48](#_Toc40881658)

[9.10 09.03.2020 48](#_Toc40881659)

[9.11 25.03.2020 49](#_Toc40881660)

[9.12 31.03.2020 49](#_Toc40881661)

[9.13 01.04.2020 49](#_Toc40881662)

[9.14 06.04.2020 50](#_Toc40881663)

[9.15 07.04.2020 50](#_Toc40881664)

[9.16 08.04.2020 50](#_Toc40881665)

[9.17 13.04.2020 51](#_Toc40881666)

[9.18 14.04.2020 51](#_Toc40881667)

[9.19 15.04.2020 51](#_Toc40881668)

[9.20 16.04.2020 52](#_Toc40881669)

[9.21 17.04.2020 52](#_Toc40881670)

[9.22 27.04.2020 52](#_Toc40881671)

[9.23 28.04.2020 52](#_Toc40881672)

[9.24 29.04.2020 53](#_Toc40881673)

[9.25 04.05.2020 53](#_Toc40881674)

[9.26 05.05.2020 53](#_Toc40881675)

[9.27 06.05.2020 54](#_Toc40881676)

[9.28 11.05.2020 54](#_Toc40881677)

[9.29 12.05.2020 54](#_Toc40881678)

[9.30 13.05.2020 55](#_Toc40881679)

[9.31 18.05.2020 55](#_Toc40881680)

[9.32 19.05.2020 55](#_Toc40881681)

[9.33 20.05.2020 56](#_Toc40881682)

[10 Testen 57](#_Toc40881683)

[10.1 Testfall 1 57](#_Toc40881684)

[10.2 Testfall 2 58](#_Toc40881685)

[10.3 Testfall 3 59](#_Toc40881686)

[10.4 Testfall 4 60](#_Toc40881687)

[10.5 Testfall 5 61](#_Toc40881688)

[10.6 Testfall 6 62](#_Toc40881689)

[11 Tabellen- und Abbildungsverzeichnis 63](#_Toc40881690)

[12 Glossar 64](#_Toc40881691)

[13 Stichwortverzeichnis 66](#_Toc40881692)

# Die Idee

Uns war schon Anfangs klar, dass wir ein Game entwickeln werden.  
Nun gab es zu entscheiden in welchem Genre.  
Als erstes dachten wir an einen Shooter, jedoch wurde uns schnell klar, dass dies zu stumpf und viel zu aufwendig war.

Uns packte dann der Charme der 8bit Adventure Games.

## Brainstorming

In welche Richtung unser Abenteuer gehen würde, war uns noch nicht bekannt.

Nach intensivem überlegen und einem Wochenende voller Abenteuer spiele, konnten wir unsere Idee zu Papier bringen.

### Ideensammlung

Das Abenteuerspiel soll das Aufregende und Spannende Leben im Basislehrjahr repräsentieren.

Natürlich werden auch einige erfundene Szenarien und Übertreibungen vorzufinden sein um das ganze unserem Geschmack anzupassen.

# Vorgehensweise

Bevor wir mit dem Projekt anfangen, mussten wir uns für eine Vorgehensweise entscheiden. Unserer Meinung nach, war IPERKA dafür geeignet und wir kannten dieses schon. Ausserdem kann unser Vorhaben gut darin aufgeteilt werden. In IPERKA wird auch der Fokus auf die Vorbereitung gesetzt was uns sehr wichtig war, da wir ohne Erfahrung in die Projektarbeit starteten.

# Informieren

## Game Engine

Da es uns an jeglichem Wissen über die Entstehung eines Spiels fehlte, gab es sich nun im Internet zu erkundigen. Wir kamen auf den Entschluss, das Programm «GoDot» zu verwenden, da dieses sowohl Grafisch als auch per Script genutzt werden kann und Perfekt für 2D Games ist.

Godot Homepage: <https://godotengine.org/>

Auf der Videoplattform YouTube wurden wir über die Grundlagen unseres Programmes aufgeklärt.

Tutorial von [LetsGameDev](https://www.youtube.com/playlist?list=PL1td_Fr5vMGOW0hasVEYlvfdm_oYh0xi9)

## Design

Um unser Spiel Grafisch zu gestalten, nutzten wir das Bildbearbeitungsprogramm, Photoshop.

Dies ist eine kostenlose Alternative Photoshop:

Gimp Homepage: <https://www.gimp.org/>

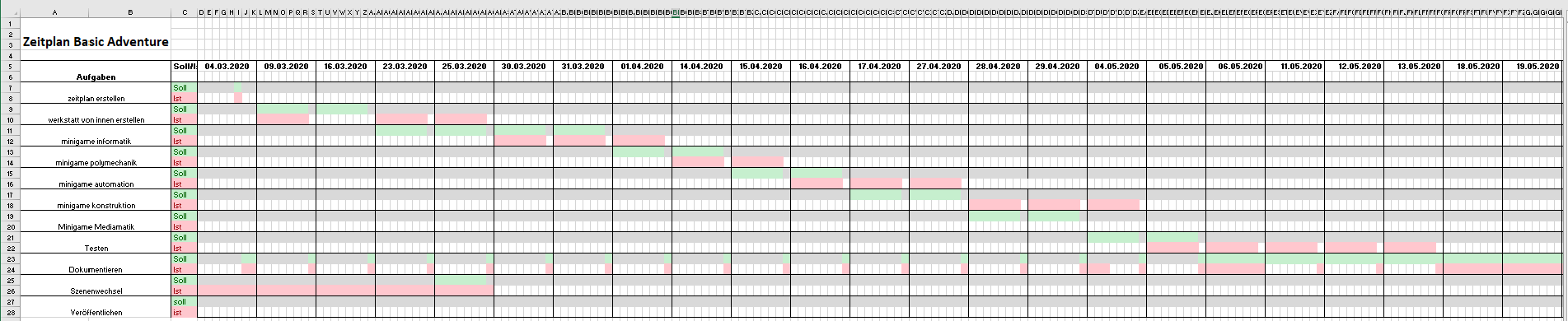
## Was ist möglich

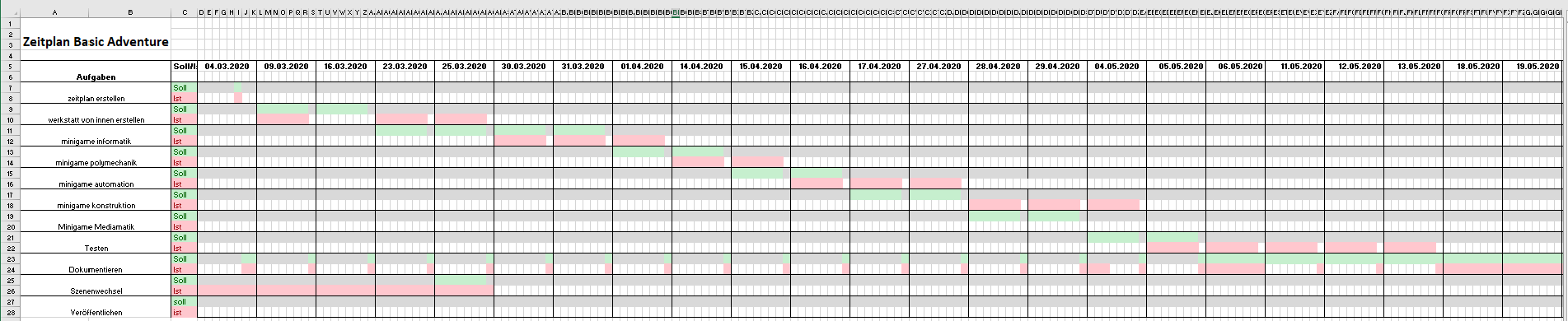
Nach vielen Tutorials, war uns klar, dass die Umsetzung mit diesem Programm gut funktionieren könnte. Da GoDot über ein GUI verfügt, ist es nicht allzu schwer Texturen und Bausteine des Spieles per Drag and Drop einzufügen. Die Scriptsprache ist sehr simpel und zeigt selbst an, was Probleme bereitet.

# Planung

Um ein solches Projekt umzusetzen, muss man Organisiert vorgehen, so sammelten wir Ideen und überlegten, was möglich wäre und was nicht.

## Zeitplan





Dies ist unser Zeitplan es wurde bis auf das Mediamatik Minigame alles erreicht zusätzlich haben wir noch einen Charakter Wechsel ins Spiel eingebaut. Wir haben uns Grösstenteils an den Zeitplan Gehalten ausser bei dem Szenen Wechsel da dieser vielen Probleme bereitet hat. Ausserdem kamen noch üKs dazwischen und am Schluss hatten wir doch noch etwas mehr Zeit.

# Umsetzung

## Grafisches

Damit man ein Spiel spielen kann, muss es Texturen besitzen. Diese Wurden mit Hilfe von Photoshop erstellt. Wichtig hier war es, schon zu wissen, welche Auflösung das Spiel hat. Da wir ein Pixel-Adventure erstellen, entschieden wir uns für ein 20px x 20px Raster, in welches wir zeichneten. Als erstes wurde unser Spiel Charakter erschaffen, für welchen wir 28 Bilder erstellten. Je vier Bilder für nach vorne, hinten links und rechts. Dazu noch zwölf wenn der Charakter nichts tut. Das Resultat sieht wie folgt aus.

Tabelle 1 Spieler Textur

|  |  |
| --- | --- |
| Beschreibung | Bild |
| Die Zusammenhängenden Bilder werden in einer endlosschleife Abgespielt, sobald der Charakter in die passende Richtung geht, oder stehen bleibt.  Um mehr Vielfalt ins Spiel zu bringen, wurden mehrere «Skins» erstellt | C:\Users\noel.wangler\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Charakter_blue.png |
| Dies sind die «Skins» welche es am Ende ins Spiel geschafft haben. |  |

Wir haben auch andere Texturen und Animationen kreiert diese sehen wie folgt aus:

Tabelle 2 Andere Texturen

| Beschreibung | Bild |
| --- | --- |
| Unser Menü besteht aus den folgenden 4 knöpfen einmal der Home Button der Exit Button der Neustart Button und die Einstellungen. Bei all diese knöpfe kann man in GoDot eine zweite und dritte Textur einfügen für den Hover und dem gedrückten zustand. |  |
| Dieses Menu ist im Aktuellen spiel vorhanden. |  |
| Diese Texturen, werden für Wände und Boden, sowie auch Objekte im Spiel verwendet. Sie können alle auf ein Raster gezeichnet werden.  Bei den Wänden ist es wichtig, sie so zu zeichnen, dass sie Dreidimensional wirke.  Der Hintergrund sollte Transparent sein, damit Objekte auf die Böden gesetzt werden können ohne sie komplett zu verdecken. | C:\Users\noel.wangler\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Gras2.png |
| Diese Texturen, wurden alle im Spiel verwendet um das Tileset zu gestallten. |  |

## Photoshop einrichten

Damit man all diese Grafiken in Photoshop erstellen kann, braucht man auch ein wenig Vorkenntnisse. Hier zeigen wir euch einige Basics in Photoshop.

Tabelle 3 Photoshop einrichten

| Beschreibung | Bild |
| --- | --- |
| Im ersten Schritt erstellen wir ein neues Dokument, Hierzu klickt man oben links auf «Datei» und im Anschluss auf «neu» es öffnet sich ein neues Fenster, hier kann man vorlagen von Photoshop nehmen oder einfach auf «Benutzerdefiniert» klicken und eigene Masse angeben. Mit einem Klick auf «Erstellen» wird eine neue Leinwand erstellet. | Ein Bild, das Screenshot, Computer, Monitor, Laptop enthält.  Automatisch generierte Beschreibung |
| Die Leinwand ist jetzt leer, Bevor wir aber etwas zeichnen, müssen wir Photoshop noch ein wenig anpassen, als erstes klicken wir auf «Fenster» und dann muss sichergestellt werden, dass «Ebenen», «Farben», «Optionen» und «Werkzeuge» einen Haken haben. | Ein Bild, das Screenshot, Computer, Monitor, Laptop enthält.  Automatisch generierte Beschreibung |
| Als nächstes sollte die Hintergrundeben transparent werden dazu klickt man als erstes auf das kleine Schloss im Ebenen Fenster, danach wählt man alles mit [CTRL]+[A] aus und löscht es mit der [Delete] Taste. Nun sollte es weiss grau Kariert sein, das heisst das es Transparent ist. |  |
| Ich habe zudem auch ein Raster hinzugefügt damit ich genauer arbeiten kann. Ein Raster ist auch nützlich, wenn man mit Tilesets arbeitet denn dann kann man das Raster so einstellen das man direkt sieht wie gross ein Feld ist. Um ein Raster zu erstellen muss man unter dem Reiter Bearbeiten auf Voreinstellungen und dann auf Hilfslinien, Raster und Slices. Es öffnet sich ein neues Fenster. |  |
| Hier kann nun unter «Raster» das Raster einstellen ich habe mich für ein Raster entschieden das 20 pixel breit ist da alle Texturen in unserem Game 20x20 Pixel gross sind. Ausserdem kann man noch Unterteilungen hinzufügen und auch die Farbe des Rasters ändern. |  |

## Photoshop Tools

Tabelle 4 Photoshop Tools

|  |  |
| --- | --- |
| Beschreibung | Bild |
| Tools in Photoshop befinden sich auf der linken Seite des Bildschirms. Tools haben alle verschiedene Eigenschaften ich zeige euch jetzt nicht alle denn das würde ewig gehen, ich zeige euch die, die ich am meisten gebracht habe bei unserem Game. |  |

### Auswahl Tool [M]

Mit dem Auswahl Tool kann man ein Rechteck auswählen indem man mit gedrückter linker maustaste über dem Bildschirm fährt. Wenn man ein Bereich auswählt dann kann man nur in diesem Bereich zeichnen. Mit Rechtsklick auf das Tool kann man weitere Auswahl Tools sehen wie das Ellipsen Auswahl Tool. Nachdem man einen Bereich ausgewählt hat dann kann man diesen mit [Ctrl]+[C] kopieren und wo anders mit [Ctrl]+[V] einfügen.

### Verschieb Werkzeug [V]

Mit dem Verschieb Werkzeug kann man Objekte in Photoshop verschieben. Man sollte dabei achten, dass man in derselben ebene ist.

### Buntstift/Pinsel [B]

Standartmässig hat man bei diesem Tool den Pinsel ausgewählt dieser ist gut fürs zeichnen da er «weicher» ist als der Buntstift aber für Pixel Texturen muss man den Buntstift benutzen da der klare kanten hat, um diesen zu benutzen muss man ein rechtsklick auf das Tool machen und den Buntstift auswählen. Um Farben zu ändern kann man auf der rechten Seite im Reiter «Farbe», Farben aussuchen. Mit gedrückter [Alt] Taste verwandelt sich der Pinsel/Buntstift in eine Pipette und man kann eine vorhandene Farbe aussuchen.

### Radiergummi [E]

Der Radiergummi wird benutzt um Dinge zu radieren doch dieser ist auch weich wie der Pinsel und ist daher nicht geeignet um Pixel zu löschen daher empfähle ich die Pixel die gelöscht werden sollen auszuwählen mit dem Auswahlwerkzeug [M] und dann mit der [Delete] taste zu löschen.

### Tastenkürzel

Tabelle 5 Tastenkürzel Photoshop

|  |  |
| --- | --- |
| Tastenkürzel | Tool/Werkzeug/Funktion Name |
| [M] | Auswahl Tool |
| [V] | Verschieb Werkzeug |
| [B] | Buntstift/Pinsel |
| [E] | Radiergummi |
| [W] | Schnellauswahl Werkzeug |

## Godot

Nun haben wir alle wichtigen Texturen erstellt und wir beginnen mit dem Erstellen eines Spieles.

Dazu laden wir die [64-bit Version](https://downloads.tuxfamily.org/godotengine/3.2/Godot_v3.2-stable_win64.exe.zip) ohne C# herunter.

Um ein Spiel mit GoDot zu erstellen, sollte man zuerst die Grundlagen kennen.

Tabelle 6 Godot Basics

| Name | Anwendung |
| --- | --- |
| Res:// | Res:// ist das oberste Verzeichnis. In ihm sind alle Szenen, Texturen, Animationen usw. gespeichert |
| .godot | .godot ist die Projektdatei. In ihr sind alle Dateien verknüpft. |
| .gd | .gd Dateien sind scripts. Sie sind in gd-script geschrieben. Das ist eine für Spiele optimierte Variante von Python |
| .tscn | .tscn sind alle Szenen. Sie bestehen aus Nodes, welche miteinander verknüpft werden. |
| Node | Nodes sind Knotenpunkte. Ein GoDot Projekt besteht aus vielen Nodes, welche andere Bedeutungen haben z.B. Eine Node ist ein Button, welcher auf Knopfdruck eine Animation abspielt. |
| .tres | Animationen werden mit .tres gespeichert. Sie speichern das 1.Bild und das Bild, welches auf das Letzte der Animation Folgt und wie der Übergang dazwischen ist. |
| Assets | Es empfiehlt sich einen Ordner namens Assets für alle Bilder und Texturen zu erstellen. |

### Wie sind die Dateien verknüpft

Die .godot Datei, macht aus den einzelnen Dateien folgende Verbindung

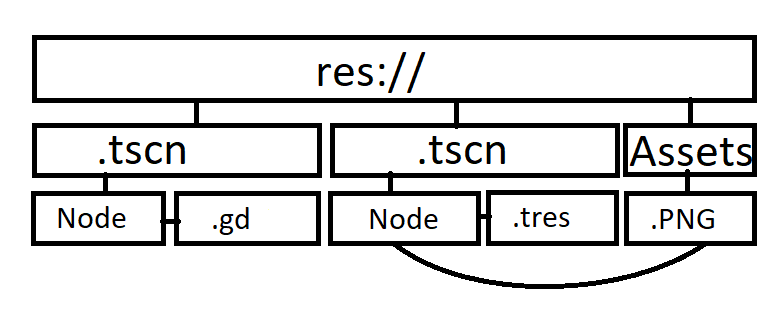


Abbildung 1 Verknüpfung der Dateien

### Godot-Script

Tabelle 7 Godot Script

| Befehl | Bedeutung |
| --- | --- |
| extends + Nodeart | Jedes Script beginnt mit einem extendes und der Art der Node z.B. «extends Area2D» |
| func \_trigger | Eine Funktion hat einen Trigger z.B. «\_pressed» das heisst sobald diese Node angeklickt wird, wird die Funktion ausgeführt. |
| If Input.is\_action\_pressed(«left»): | If stellt eine Bedingung, heisst WENN ein Input kommt, welcher von einer Taste kommt, den Namen left trägt, dann wird das darauffolgende ausgeführt. |
| If Input.is\_action\_pressed(«left»):   * get\_tree().quit | Wenn nun die zuvor gestellte Bedingung erfüllt wurde, wird er denn gesamten «Node-Baum» austauschen durch den Nodebaum von Quit, was heisst, das Spiel wird verlassen. |
| If Input.is\_action\_pressed(«left»):   * get\_tree().quit   If Input.is\_action\_pressed(«right»):   * get\_tree().quit | Wichtig bei GoDot ist, dass es anstatt mit Semikolons, mit Absätzen und Tabulatoren arbeitet. So werden alle befehle auf einer Ebene gleich gewertet, die eingerückten werden erst benutzt, wenn der Befehl oben dran darauf weiterleitet.  Das Programm geht nun zum ersten If und wenn dies erfüllt ist, beendet es das Spiel.  Wenn es nicht erfüllt ist, geht es zum nächsten If und so weiter. |
| pass | Wenn man pass unter eine Funktion schreibt, wird diese übersprungen |
| $Node.befehl | Mit $ kann eine andere Node in der Szene angesprochen werden und nach dem Punkt kann ein Befehl gegeben werden z.B. $AnimationPlayer.play(«name\_der\_animation») |

## Eigenes Spiel

Bevor man mit dem Programmieren beginnt, sollte man sich einen Plan erstellen, auf welchem gezeigt wird, welche Szenen existieren, was man in diesen machen kann und wie sie verknüpft sind.

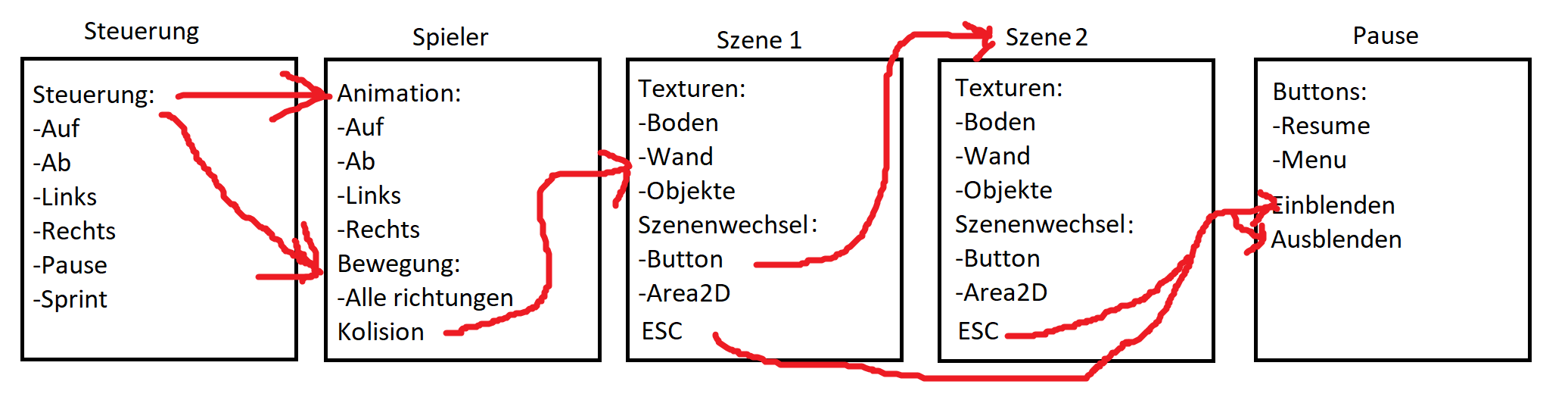


Abbildung 2 Spiel Plan

Tabelle 8 Spiel erstellen

| Erklärung | Bild |
| --- | --- |
| Nach dem Öffnen des Programmes, wird ein neues Projekt erstellt.  Es wird ein Name gegeben und ein Pfad bestimmt. Es Empfiehlt sich GLES 3.0 auszuwählen, da dadurch das Spiel schöner aussieht |  |
| Zu beginne ist GoDot noch auf 3D gestellt. Dies kann aber mit den sich oben in der Mitte befindenden Tasten geändert werden. |  |
| Um einen Charakter zu erstellen, wird mit dem Plus Symbol ein KinematicBody eingefügt. Darunter eine CollisionShape und ein AnimationPlayer. |  |
| Der CollisionShape muss nun noch eine Form gegeben werden. Diese kann auf der rechten Seite ausgewählt werden. |  |
| Jetzt wird die Textur unseres Charakters eingefügt.  Diese ist noch seht «verpixelt» und muss deshalb noch verändert werden. Nämlich macht man einen Rechtsklick auf die Textur des Charakters und geht zum Register Import und stellt unter «Voreinstellungen» auf «2D Pixel» um. |  |
| Da die Textur noch aus 32 Bildern besteht, muss sie noch angepasst werden.  Dazu geht man auf die rechte Seite des Bildschirms und verändert folgende Optionen. |  |
| Bei der Kategorie «Animation» wird eingegeben, in wie viele Zeilen und Spalten das Bild eingeteilt werden kann, um ein einzelnes zu erstellen. |  |
| Die CollisionShape wird nun an den Charakter angepasst und alle «Nodes» werden so umbenannt, dass man weiss, was sie tun.  Ausserdem wird eine «Camera2D» Hinzugefügt. |  |
| Die Kamera besitzt folgende Optionen.  Current bedeutet, dass sich die Kamera mit dem Spieler mitbewegt | C:\Users\noel.wangler\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Godot007.jpg |
| Erstes Script | |
| Damit man den Spier bewegen kann, muss man ihm ein «script» hinzufügen.  Jedes script wird mit einem «extend+Nodenamen» begonnen.  Dann werden Funktionen «func+name():» geöffnet. In unseren Fall wurden \_process(delta): verwendet, da dies der Standard ist.  Damit wir nicht immer die Position angeben müssen, schreiben wir eine Variabel namens «movement» | C:\Users\noel.wangler\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Godot009.jpg |
| Damit das Spiel weiss, wann es welche Befehle ausführen muss, nimmt man den Befehl «if» (auf Deutsch «wenn») |  |
| Input.is\_action\_pressed(«left»)  Mit dem Input Befehl, wird gezeigt, dass es sich um eine Eingabe handelt.  Das is\_action\_pressed, heisst dass es eine Aktion ist.  Und in der Klammer ist der Name der Aktion, welche in den Projekteinstellungen eingestellt werden kann.  Durch den Doppelpunkt am Ende der Linie wird gezeigt, dass nur wenn diese Bedingung erfüllt wird, die nächste Linie gelesen wird. |  |
| Nun folgen die nächsten Bedingungen, welche sagen, dass nur wenn der Input «right», « down» und «up» NICHT gedrückt werden, die Bewegungen «movement.x -32» ausgeführt wird.  movement ist die zuvor definierte variabel, für die aktuelle Position 0,0.  Durch drücken, der Taste «left» wird diese dann um 32 Pixel nach links verschoben. |  |
| Animation | |
| Um Animationen abzuspielen, müssen diese zuerst erstellt werden, dies funktioniert so: |  |
| Durch einen Klick auf das AnimationPlayer Zeichen, öffnet sich im unteren Bereich des Bildschirms dieses Fenster. Hier können unter dem Punkt «Animation» neue Animationen erstellt werden.  Nach dem man die «Zeit» auf 0 gestellt hat, kann man Texturen aussuchen. |  |
| Nun kann die Textur des Charakters angeklickt werden und unter dem Punkt Animation kann ein Frame bestimmt werden bei welchem die Animation startet. Sobald dieser gefunden wurde, wird er mit dem Schlüssel ausgewählt. |  |
| Nun wird im AnimationPlayer der Balken nach rechts geschoben und beim Charakter wird der Frame **nach** dem Letzten Frame der Animation erneut durch einen Klick auf den Schlüssel ausgewählt. |  |
| Wenn man nun den Start und den End Frame ausgewählt hat muss man noch den Aktualisierungsmodus auf Fortlaufend stellen damit alle Bilder dazwischen auch angezeigt werden. |  |
| Auf dem kleinen Dreieck kann man die Animation auch laufen lassen und schauen ob sie gut aussieht. |  |
| Tileset | |
| In einem Tileset befinden sich verschiedene texturen, diese kann man dann in Godot auswählen und ein Tile erstellen. Tiles sind texturblöcke, mit welchen man dann eine Map «Zeichnen» kann. |  |
| Um ein Tileset zu erstellen muss man erstmal auf eine TileMap klicken diese kann man ganz einfach mit dem kleinen + in sein Projekt einfügen. | Ein Bild, das Monitor, Anzeigetafel, Computer, schwarz enthält.  Automatisch generierte Beschreibung |
| Auf der rechten seite hat sich nun ein Fenster geöffnet. Man muss nun auf dem kleinen pfeil neben Tileset klicken und dort «Neues Tileset» auswählen. Anschliessend klickt man jetzt auf das erstellte Tileset. | Ein Bild, das Monitor, Computer, schwarz, drinnen enthält.  Automatisch generierte Beschreibung |
| Auf der unteren seite des Bildschirms öffnen sich ein Fenster, hier muss man auf das Plus drücken um eine PNG datei mit Texturen einzufügen. | Ein Bild, das Monitor, Screenshot, Computer, schwarz enthält.  Automatisch generierte Beschreibung |
| Es öffnet sich ein neues Fenster hier muss man eine png datei mit texturen aussuchen in unserem fall Gras2.png | Ein Bild, das Screenshot, Monitor, Bildschirm, schwarz enthält.  Automatisch generierte Beschreibung |
|  | Ein Bild, das Monitor, Screenshot, Computer, schwarz enthält.  Automatisch generierte Beschreibung |
| Signals | |
| Für die Komunikation zwischen Nodes, gibt es «signals» alle Arten von nodes haben verschiedene signals, welche ausgegeben werden, wenn eine bedingung erfüllt wird.  Das Signal pressed() wird ausgegeben, wenn diese Node ankeklickt wird. | Ein Bild, das Text enthält.  Automatisch generierte Beschreibung |
| Signale können auch selbst definiert werden indem man in das Script der Node, signal name schreibt und dieses durch eine Bedingun ausgegeben wird. Dies geschieht mit emit\_signal(«name») | Ein Bild, das Monitor, Bildschirm, schwarz, befestigt enthält.  Automatisch generierte Beschreibung |
| Dieses Signal wird nun bei den Signalen der Node angezeigt.  Durch doppelklick kann diese node nun verknüpft werden. | Ein Bild, das draußen, Uhr, Straße, Anzeige enthält.  Automatisch generierte Beschreibung |
| Das Signal name, welches von der Node Button kommt, kann nun mit allen Nodes aus der Szene verknüpft werden. Signale können nicht global genutzt werden. | Ein Bild, das Zeichnung enthält.  Automatisch generierte Beschreibung |
| Wenn eine Node ausgewählt wurde, kann nun unter die neu entstandene funktion geschrieben werden, was geschieht, wenn das Signal «name» empfangen wird. | Ein Bild, das Screenshot, Bildschirm, schwarz, Zeichnung enthält.  Automatisch generierte Beschreibung |
| Szenenwechsel | |
| Da die meissten Speile aus verschiedenen Szenen bestehen, muss es möglich sein diese zu wechseln.  Dies geschieht mit «get\_tree().change\_scene(«Name der scene») | Ein Bild, das Bildschirm, Monitor, schwarz, Zeichnung enthält.  Automatisch generierte Beschreibung |

Problematisch bei einem Szenenwechsel ist, dass der Charakter immer den Gleichen Anfangspunkt hat

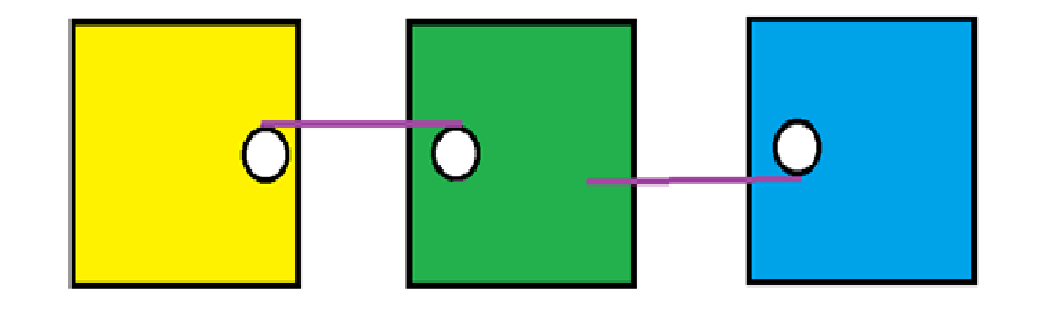


Abbildung 3 Szenen Wechsel einfach

Immer, wenn man in einen Trigger für den Szenenwechsel geht, wird die Nächste Szene geladen, mit dem Startpunkt im Kreis. Das macht Sinn von Gelb zu Grün, aber nicht von Blau zu Grün. Um den Übergang zwischen Allen Szenen Schön zu machen, wird die Grüne Szene Dupliziert und abgeändert.

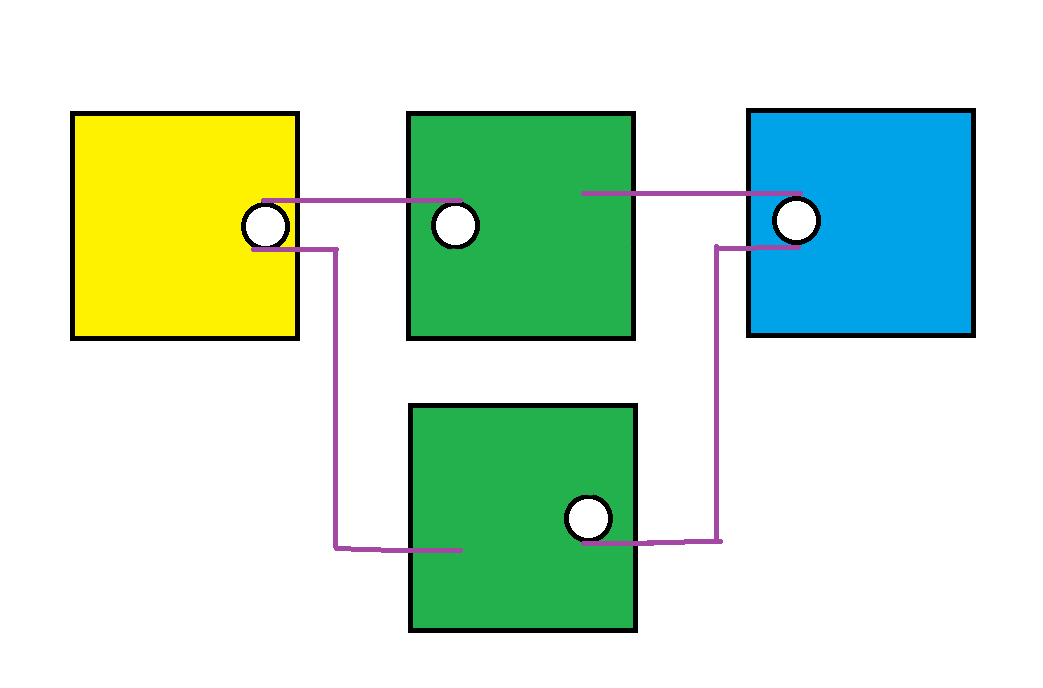


Abbildung 4 Szenen Wechsel erweitert

Beim Wechsel von Gelb zu Grün, wird die obere grüne Szene geladen. Beim Wechsel von Blau zu Grün, wird die untere grüne Szene geladen.

## Allgemeine Einstellungen

Hier werden die Wichtigsten Einstellungen aufgezählt. Geändert werden können sie in den Projekteinstellungen.

Tabelle 9 Godot Allgemeine Einstellungen

| Erklärung | Bild |
| --- | --- |
| Unter Projekt/Allgemein/Application/Config kann der Name des Projekts und das Icon ausgewählt werden. |  |
| Hier kann die Main Scene (Startszene) und die Version ausgewählt werden. |  |
| Um einen eigenen Ladescreen einzubinden, muss er hier angegeben werden. |  |
| Hier kann der Treiber geändert werden  OpenGL ES 3 (für Leistungsstärkere Mobil Geräte)  OpenGL ES 2 (für ältere Geräte)  USE Pixel Snap hilft gegen Grafikfehler, welche durch Tilesets entstehen. |  |
| Da heute die meisten Prozessoren Multithreaded sind, sollte man diese Option einschalten. |  |
| Es empfiehlt sich den Vsync zu aktivieren, um Screen tearing zu verhindern.  Screen tearing bedeutet, dass sich das Bild verzieht, weil die Wiederholrate nicht synchron ist. |  |

## Steuerung

Tabelle 10 Steuerung

|  |  |
| --- | --- |
| Erklärung | Bild |
| In den Einstellungen kann man unter Eingabe-Zuordnung, Tasten mit Aktionen verknüpfen, welche man als Input anwenden kann. |  |

## Pausen-menu

Tabelle 11 Pausen-menu

| Erklärung | Bild |
| --- | --- |
| Wichtig für eine Pausen-screen ist, dass er das Spiel überdeckt und man durch ihn das Spiel verlassen oder fortsetzen kann.  Dazu integrieren wir in der Steuerung, ein «ColorRect» und 2 «Button», für Exit und Resume. |  |
| Um durch das GUI und die Tastatur auf das Pausenmenu zu kommen, wird ein TouchscreenButton erstellt, welcher auf Knopfdruck ein Signal Pause zur Obersten Node Schickt.  Und das Gleiche wird auch ausgelöst, durch den Tasteninput «menu», welcher durch die «esc» Taste getriggert. |  |
| Nun sendet die Oberste Node ein Signal an das gesamte GUI, welche ausgeblendet werden «visible = not visible».  Das selbe geschieht mit dem zuvor ausgeblendeten Pausenmenu. |  |

# Minigames

Die Minigames sollen unsere noch leere Map mit Leben füllen, und die Berufe der Wibilea ganz simpel vorstellen. Für die Minigames wird hier auf das bisherige Script verzichtet und es werden neue Szenen erstellt.

## Informatikgame

Bei dem Informatik Game muss man einen PC zusammenbauen indem man die Komponente in die richtige Position bewegt. Wir haben hier mit «Drag and Drop» gearbeitet d.h. man kann die Komponenten nehmen und wo anders Platzieren.

Tabelle 12 Informatik Game

| Erklärung | Bild |
| --- | --- |
| Hier ist der Fertige Aufbau der Szene. «ground-plug» ist der Hintergrund. Auf ihm befinden sich ausgeblendete Bilder für den Beschrieb von den Teilen.  Es folgen weitere Sprites, welche die Teile an ihrem Platz anzeigen, wenn die Aufgabe richtig gelöst wurde.  Die TouchscreenButtons sind die «Komponenten», sie haben die deren Textur (CPU, GPU etc.)  Die Areas werden genutzt um eine Kollision mit den Teilen zu erkennen. Und dann fest zu stellen, ob es das Richtige Teil ist.  Der Button namens exit2 ist für das Vorzeitige Verlassen des Spiels. |  |
| Die Abfolge in dieser Szene ist folgende: durch klicken auf den Button «cpu», kann man diesen über den Screen bewegen. Sobald dieser auf der Area namens «areasocket» ist, sendet diese ein Signal an sich selber, cpu, Ryzen-Su-640x640, und wlp.  Alle bekommen den Befehl (visible = not visible)  Nun wird das ganze wiederholt mit dem Button wlp |  |
| Der Komplizierteste des Games ist das «Drag and Drop»  Als Physik Prozess wird nun, wenn der Input «cpuon» (wird vom Touchscreen Buttons gesendet) aktiv ist die x,y Position des Buttons zur x,y Position des Mauszeigers | C:\Users\noel.wangler\AppData\Local\Microsoft\Windows\INetCache\Content.Word\infmini4.PNG |
| Der Rest wird mit Signalen verknüpft welche von der Area2D gesendet werden. Diese werden mit allen Objekten verbunden, welche angezeigt oder ausgeblendet werden sollen. |  |
| Damit die Area nur Signale aussendet, wenn der Richtige Button Kollidiert, wird folgendes Script in die Area geschrieben.  Das heisst, dass nur dieses als Signal verwendet werden darf, weil die Area sonst bei allen Kollisionen aussenden würden. |  |
| Durch klicken des exit2 Button, wird ein Signal an die oberste Node geschickt, welche dann die Szene lvl1.2 lädt, welche den Spieler zurück in die Wibilea bringt. |  |

## Automatik Minigame

Bei dem Game der Automation ist das Ziel eine Glühbirne zum Leuchten zu bringen, im ersten Schritt lötet man zwei Kabel zusammen und anschliessend verbindet man eine Glühbirne mit einer Batterie damit diese leuchtet. Das Mini Game verwendet auch Drag and Drop wie das Informatik Game. Hier zeige ich wie Das Erste Level erstellt wurde.

Tabelle 13 Automation Game

| Erklärung | Bild |
| --- | --- |
| So Sieht der Aufbau des ersten Levels aus.  Man Sieht drei «TouchscreenButtons» (kabel, kabel2 und loetkolben), diese haben je eine Area2D und eine CollisionShape. Zuoberst hat es dann noch ein TextureRect das ist der Hintergrund. |  |
| Alle die TouchscreenButtons haben am Anfang keine Textur und keine Action die ausgeführt wird. Diese kann man hinzufügen, wenn man auf ein TouchscreenButton klickt und dann auf der linken Seite auf den Inspektor geht.  Bei uns Hat unser Kabel eine Textur von einem Kabel und eine «Action» die «kabel» heisst diese «Action» wird später wichtig sein. |  |
| Wir möchten, dass man die TouchscreenButtons Frei auf dem Bildschirm Bewegen kann Um dies zu erreichen müssen wir den TouchscreenButtons ein leeres Script hinzufügen (Oben Links im Szenen Tab). In diesem Script schreiben wir dann eine Funktion die Folgendermassen aussieht:  func \_process(delta):  If Input.is\_action\_pressed(‘kabel’):  Position.x = get\_global\_mouse\_position().x -187  Position.y = get\_global\_mouse\_position().y -2  Im Script steht in Worten gesagt einfach das wenn die Aktion «kabel» ausgeführt wird das dann das Node kabel an die Position bewegt wird wo sich die Maus befindet. |  |
| Als Nächstes müssen wir von den Area2Ds ein Signal absenden immer, wenn diese Area2D mit einer anderen Kollidiert. Um das zu erreichen müssen wir zuerst die Area2D auswählen und dann rechts auf das Tab «Node» Hier wählen wir das Signal «area\_entered» aus |  |
| Es öffnet sich ein neues Fenster Hier kann man mit einem Doppelklick auswählen wo das Signal gesendet werden soll Wir wählen hier unsere Haupt Node «Automation» aus.  Danach erstellen wir nochmals ein Signal diesmal aber «area\_exited» und leiten dieses auch in das Haupt Node «automation». |  |
| Wenn Wir jetzt das Script vom Node «automation» öffnen sehen wir Zwei kleinen grünen Pfeil und dahinter je eine Funktion. Diese wurden erstellt als wir die Signale erstellt haben. Alles was unter der Funktion steht, wird ausgeführt sobald unser Kabel eine andere Area2D betritt beziehungsweise verlässt. |  |
| Als nächstes erstellen wir 2 variablen Diese schreiben wir ganz oben im Skript. Diese müssen beide den Wert ‘false’ haben. |  |
| Unter der Funktion Löschen wir das pass und den Kommentar. Jetzt schreiben Folgendes in der oberen Funktion:  if area.get\_name() == "arealoetkolben":  don = true  in der Unteren können wir das Gleiche schreiben nur wird hier das don = false. |  |
| Jetzt erstellen wir die gleichen Signale für das Kabel2 und leiten diese auch in Das Haupt Node «automation». Unter den Funktionen kommt das Gleiche Skript nur das don wird mit don2 ersetzt. |  |
| Als letztes erstellen wir eine neue Funktion und in diese schreiben wir:  func \_process(delta):  if don and don2 == true:  get\_tree().change\_scene("res://")  Wenn die Area2Ds der Kabel mit dem Lötkolben kollidieren werden don und don2 true und wenn diese true sind wird diese Funktion ausgelöst und man wird in die nächste Szene teleportiert. |  |
| Jetzt sollten noch alle CollisionShapes richtig platziert werden damit es richtig funktioniert. |  |

## Polymechanik Minigame

Tabelle 14 Polymechanik Game

| Erklärung | Bild |
| --- | --- |
| Beim Polymechanik Minigame wird hauptsächlich mit Animationen und CollisionShapes gearbeitet. |  |
| Die CollisionShapes müssen in einer gewissen Reihenfolge mit einer Area2D berührt werden.  Dies kann gemacht werden, indem man eine variabel durch eine Berührung true setzt und erst, wenn diese true ist, kann die nächste Area berührt werden. |  |
| Wenn alle Areas berührt wurden sind alle variablen auf true und im \_process(delta) wird nach ablaufen eines Timers dies Szene gewechselt, mit Hilfe von get\_tree().change\_scene(«pfad»). |  |
| Durch die verschiedenen Areas, werden verschiedene Animationen abgespielt. |  |
| Die Animationen zeigen, wie ein Werkstück bearbeitet wird, indem sie je einen Pixel tiefer geschnitten sind. |  |
| Das Werkzeug, welches zur Bearbeitung des Werkstücks genutzt wird, kann per Drag and Drop bewegt werden. |  |
| Wenn die unterste Area2D erreicht ist, wird die Szene gewechselt |  |
| Das nächste Polymechanik Minigame verlangt, mit einem Bohrer (rotes Kreuz) eingezeichnete Löcher in ein Werkstück zu bohren. |  |
| Der Bohrer wird gleich bedient, wie der Spieler.  Er muss eine Kollision haben, damit er nicht das Spielfeld verlässt. Das Spielfeld wird deshalb mit CollisionShapes umkreist. Um den KinematicBody zu steuern, werden 5 Touchbuttons erstellt, welche befehle «up» «down» «left» «right» «bohren»  Die ersten vier sind um den body über das Feld zu bewegen. Der 5. Löst aus, dass überprüft wird, ob der Bohrer über einem eingezeichneten Loch ist |  |
| Wenn der die Taste «bohren» gedrückt wird, wenn der Bohrer nicht über einem eingezeichneten Loch ist, wird die Szene neu geladen. get\_tree().change\_scene(«szenen\_pfad»)  Wenn alle Löcher getroffen wurden, wird die nächste Szene geladen. |  |

## Konstruktion Game

Im Game der Konstruktion kann man etwas zeichnen, diese Zeichnung kann man dann auf der Wand in der Konstruktion sehen.

Tabelle 15 Konstruktion Game

| Erklärung | Bild |
| --- | --- |
| Als erstes erstellen wir eine neue Szene und fügen ein Line2D unter dem kleinem + ein. |  |
| Als nächstes erstellen wir einen Timer als Kind von der Line2D und wir benennen die Line2D in Pencil um. |  |
| Jetzt klicken wir auf den Timer und Stellen Rechts im Inspektor, die Wait Time auf 0.001 und aktivieren Autostart. | v  v |
| Bevor wir weiter machen können müssen wir oben auf  Projekt->Projekteinstellungen->Eingabe-Zuordnung Hier fügen wir die neue aktion Draw ein. |  |
| Danach Scrollen wir runter und klicken auf das + neben Draw und wählen hier Linkemaustaste aus. |  |
| Nun erstellen wir ein leeres Skript für den Pencil indem wir den Pencil auswählen und oben auf die Schriftrolle klicken. |  |
| Ins Skript kommt nun folgender Code:  func \_process(delta):  if Input.is\_action\_just\_released("draw"):  $Timer.stop() |  |
| Als nächstes gehen wir wieder in den Timer und erstellen rechts im Tab node ein signal.  Dazu doppelklicken wir timeout() an. |  |
| es öffnet sich ein neues Fenster hier wählen wir den Pencil aus und klicken auf verbinden. |  |
| Nun öffnet sich das Skript des Pencil, als wir das Signal erstellt haben hat godot hier eine func \_on\_Timer\_timeout(): erstellt. Die Zeile mit dem pass können wir löschen. Anstatt dem pass Schreiben wir folgendes Skript:  Add\_point(get\_global\_mouse\_position())  $Timer.start() |  |
| Das Skript das Wir nun erstellt haben erstellt alle 0.001 Sekunden einen Punkt auf der Maus Position solange Wir die linke maustaste drücken. Dadurch können wir eine Linie zeichnen. |  |
| Als nächstes erstellen wir eine Neue Szene in diese kommt eine Node2D, ColorRect, ViewportContainer, Viewport, node und ein TouchscreenButton. Das Node2D wurde in Konstruktion umbenannt und der TouchscreenButton in Speichern. |  |
| Das Node bekommt ein neues Skript in Dieses Kommt folgender Code:  extends Node  signal hide  signal show  func \_ready():  var dir = Directory.new()  dir.make\_dir("user://wibileagame")  func \_process(delta):  if Input.is\_action\_just\_pressed("draw"):  var pencil = preload("res://Pencil.tscn").instance()  yield(get\_tree().create\_timer(0.01), "timeout")  add\_child(pencil)  if Input.is\_action\_just\_pressed("save"):  emit\_signal("hide") |  |
| Als nächstes fügen wir dem TouchscreenButton auch ein Skript hinzu mit folgendem Code:  extends TouchScreenButton  signal hide2 |  |
| Nun erstellen wir ein signal das vom node kommt.   1. Node Auswählen. 2. Oben rechts auf Node wechseln. 3. Doppelklick auf hide(). 4. Den TouchscreenButton doppelklicken.   pass durch emit\_signal("hide2") ersetzen. |  |
| Jetzt erstellen wir nochmals ein Signal das Vom TouchscreenButton ausgeht.   1. TouchscreenButton Auswählen. 2. Oben rechts auf Node wechseln. 3. Doppelklick auf hide2(). 4. Das Node doppelklicken.   pass durch folgenden Code ersetzen:  yield(get\_tree().create\_timer(0.2), "timeout")  yield (get\_tree(), "idle\_frame")  yield (get\_tree(), "idle\_frame")  var image = get\_viewport().get\_texture().get\_data()  image.flip\_y()  image.save\_png("user://test.png")  get\_tree().change\_scene("res://Konstruktion\_ende.tscn")  In diesem wird immer eine neue Line2D erstellt damit man auch rundungen zeichnen kann ausserdem haben wir im unteren Teil auch das Skript um das Gezeichnete zu speichern, als erstes wird ganz oben im Script ein neues Verzeichnis erstellt wo wir dann das Bild Speichern. Weiter unten machen wir einen Screenshot mithilfe von get\_viewport().get\_texture().get\_data() dann müssen wir das Bild noch Flippen damit es richtigherum angezeigt wird und zuletzt wird es mit .save\_png(‘pfad’) im erstellten Verzeichnis gespeichert |  |
| Zuletzt fügen wir das Bild in die Map ein. dazu fügen wir ein Sprite in die Szene wo es reingeladen werden soll ein. Dann erstellen wir ein weisses png Bild in Photoshop mit dem Namen test.png. In Godot nehmen wir dann das test.png als Textur für den Sprite. |  |
| Jetzt kommt in die Haupt Node noch folgender Code:  func \_ready():  var img = Image.new()  var tex = ImageTexture.new()  img.load("user://test.png")  tex.create\_from\_image(img)  $bild.texture=tex  Es ist wichtig dass der Code in func \_ready() steht weil Diese Funktion die Textur die wir vorhin gespeichert haben ladet. Deswegen muss diese Funktion immer als erstes Geladen werden. |  |

# Charakter Auswahl

Um einen Charakter zu wechseln, werden wir die in der Animation gespeicherten Bilder ersetzen. Das geschieht ähnlich wie bei unserem Konstruktion Game.

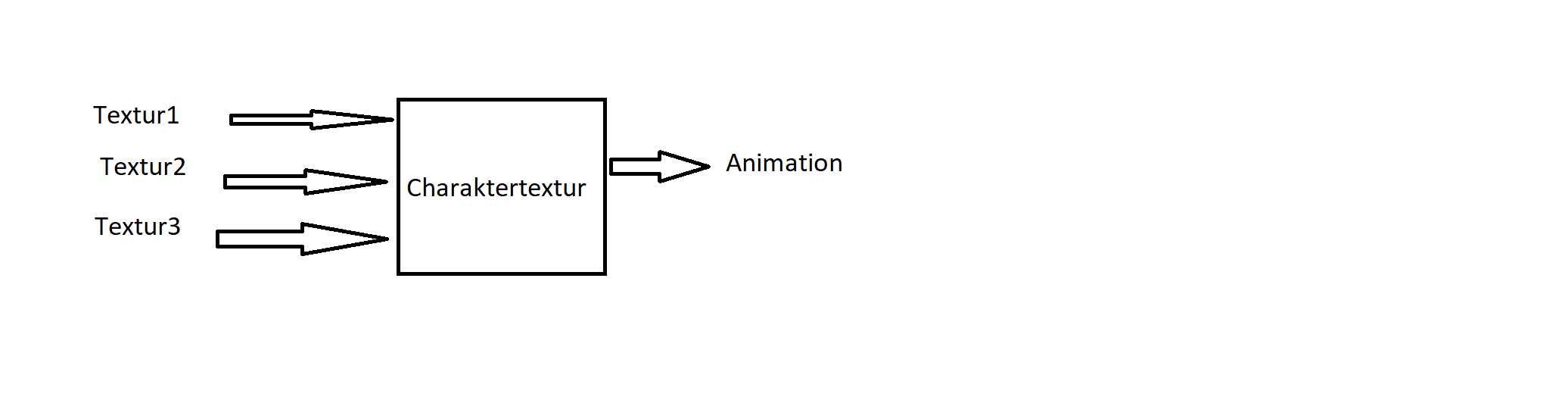


Abbildung 5 Charakterwechsel

| Erklärung | Bild |
| --- | --- |
| Als Erstes Haben wir unsere Start Szene aktualisiert wir haben Sechs neue TextureButtons erstellt. Wenn man diese Drückt sollte der Charakter wechseln und die Spielervorschau wird angezeigt. |  |
| Um die Vorschau des Spielers anzupassen, werden wir einen Sprite auf der Startseite Platzieren, welcher seine Texturen von einem PNG nimmt. Ausserdem Sollten die Aufnahmen eine höhere Auflösung haben damit Sie nicht verschwommen wirken. |  |
| Mit diesem Script wird der Angeklickte Button seine Textur in den Userordner des Geräts kopieren |  |
| Um nun die Vorschau zu ändern wird die Gespeicherte Datei aus dem Userordner genommen.  Darum kann die Textur ändern, weil sie immer wieder überschrieben werden kann. |  |
| Dasselbe passiert mit der Animierten Textur des Sprites Ingame. Er nimmt seine Texturen auch aus dem Userordner |  |

# Veröffentlichen

## Android

Um eine App für Android zu erstellen, muss das Projekt als APK-datei exportiert werden.  
Dazu braucht man einen Keystore. Dieser kann mit Android-Studio erstellt werden.

Ausserdem benötigt man openjdk und platform-tools.

Tabelle 16 Exportieren Android

| Erklärung | Bild |
| --- | --- |
| In den Editoreinstellungen unter Export/Android, müssen Adb, jarsigner und Keystore verlinkt werden. Darunter kommt der Benutzer und Das Passwort des Keystores. |  |
| Danach muss das Projekt exportiert werden. Dazu geht man unter Projekt zu Exportieren und drückt auf Hinzufügen.  Dann kann ein Betriebssystem ausgewählt werden.  Am unteren Rand befinden sich Knöpfe für den Export als apk,exe,ipa,etc. und der Export als zip oder pck |  |
| Da wir für Android exportieren, werden wir Android auswählen. |  |
| Nun kommt ein Fenster mit neuen Einstellungen. |  |
| Unser Projekt benötigt eine Version | C:\Users\noel.wangler\AppData\Local\Microsoft\Windows\INetCache\Content.Word\exportand6.PNG |
| Hier müssen wir einen Unique Name geben, welcher wie folgt aussieht com.beispiel.endung.  Zusätzlich benötigen wir noch einen Spielnamen. |  |
| Im Debug und Release muss erneut die Keystoredatei eingebunden werden und darunter er Benutzername und das Passwort. |  |
| Wenn alle nötigen Felder ausgefüllt werden, kann man das Projekt exportieren, welches als APK am ausgewählten Ort erscheint. |  |

## Windows

### Export

Der Windows Export benötigt weniger Dinge, da man hier keine Signer oder Keys benötigt

Tabelle 17 Exportieren Windows

| Erklärung | Bild |
| --- | --- |
| Gleich wie bei Android, werden rechts neue Optionen angezeigt. Diese müssen hier nicht geändert werden, ausser dem Namen, welcher zu Oberst ist. Zusätzlich können noch Icon, und Beschreibung geändert werden. |  |
| Wenn alle nötigen Angaben angegeben wurden, kann nun das Projekt als PCK exportiert werden (oberer Knopf) und als exe (unterer Knopf). Beide Dateien müssen dann in denselben Ordner gelegt werden, um zu funktionieren. |  |

### Windows Installation

Um das Spiel zu installieren, muss man die exe und pck Datei herunterladen und die exe ausführen.

## IOS

Um eine App für IOS zu erstellen benötigt man eine DevelopperID, diese kostet 109.-Fr für Einzelpersonen.

# Arbeitsjournal

## 07.01.2020

| Tätigkeit | Erledigte Arbeiten / Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Tutorials schauen | Tutorial 1-8 geschaut wan  Tutorial 1-3 usv | Was ist GoDot und was kann es | keine | <https://www.youtube.com/>watch?v=WRDl2gQObg8&list=PL1td\_Fr5vMGOW0hasVEYlvfdm\_oYh0xi9 |
| Charakter design usv | 28 Bilder für Texturen | Wie macht man Animationen in GoDot | keine | Programm : GIMP |
| Eigenen Charakter erstellt  wan | Charakter + Bewegungen erstellt | Wie macht man einen beweglichen Charakter in GoDot | Bei gleichzeitigem Links und Rechts drücken, hatte der Sprite keine Animation |  |

## 08.01.2020

| Tätigkeit | Erledigte Arbeiten / Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Tilemap erstellen  wan | Teilmap erstellt | Wie macht man Hintergründe und wände in GoDot | keine | <https://www.youtube.com/>watch?v=WRDl2gQObg8&list=PL1td\_Fr5vMGOW0hasVEYlvfdm\_oYh0xi9 |
| Texturen erstellen  usv | Texturen erstellt für:  Explosion  Wand  Boden  Gras  Stein | - | keine | programm: GIMP |
| Portale erstellen  wan | Protal erstellt | Wie kann man in godot Level wechseln | die Animation für das ausblenden, konnte nicht ein 2. mal gestartet werden | <https://www.youtube.com/watch?v=6ziIyx60N6I> |

## 13.01.2020

| Tätigkeit | Erledigte Arbeiten / Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Start und Pausen Menu erstellen  wan | Start und Pausen Menu erstellt | wie macht man pausen und Start Menu |  | <https://www.youtube.com/watch?v=5sC96znayHA> |
| Tutorials schauen usv | Tutorial 3-8 | Teilmap, Scripting | - | <https://www.youtube.com/>watch?v=WRDl2gQObg8&list=PL1td\_Fr5vMGOW0hasVEYlvfdm\_oYh0xi9 |
| Speicher Funktion | speichern ermöglicht | wie man ein spiel speichert | funktioniert nicht | <https://www.youtube.com/watch?v=ML-hiNytIqE> |

## 28.01.2020

| Tätigkeit | Erledigte Arbeiten / Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Inventar erstellen | Work in Progress | noch nichts | - | <https://www.youtube.com/watch?v=ec17gemiKpw> |
| Tische, Stühle, PCs etc. design | work in Progress | - | - | Programm: GIMP, Photoshop |

## 29.01.2020

| Tätigkeit | Erledigte Arbeiten / Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Tilemap erstellen | Tilemap erstellt | - | - |  |
| Inventar vervollständigen | work in progress | - | - | <https://www.youtube.com/watch?v=ec17gemiKpw> |

## 10.02.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Für Handy optimiert | Spiel läuft auf Handy | Wie man ein spiel für Android erstellt | keine | - |
| Touch steuerung | Funktonierende Touch steuerung | Wie funktioniert eine Touchsteuerung | Keine | - |

## 17.02.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Szenen Geflickt | Spiel läuft wieder | - | - | - |
| Map erweitert | Map ist grösser | - | - | - |
| touch steuerung erweitert | touch steuerung wurde um 2 tasten erweitert | - | - | - |

## 03.03.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| SSMap erweitert | Werkstatt von aussen | - | - | - |
| Szenenwechsel für PC | Szene wechselt an PC | - | - | - |

## 04.03.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Spiel charakter erstellt | 6 neue charakter | - | - | - |
| Szenen wechsel bei Android versucht | szene wechselt ins nichts | szenen müssen in den preload | - | - |

## 09.03.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Grundriss wibilea gebäude | grundriss vom Wibilea gebäude fertig | - | - | - |
| Neue Szenen Verknüpfung | Szenen werden zusammengefügt | dies ist nicht die lösung | Die Szenen können immer noch nicht gewechselt werden | - |

## 25.03.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Aussenbereich Wibilea grundform und texturen | alle Texturen die im Aussenbereich vorkommen sind gemacht und Grundform vom Aussenbereich ist fertig | -bereich von einem Tileset kopieren und woanders einfügen | -anfags hat das kopieren des tilesets nicht funktioniert | - |
| Szenen wieder auseinandernehmen und neu verknüpfen | szenen können nun auch bei Android gewechselt werden | Szenen müssen mit get\_tree():changescene gewechselt werden | - | - |

## 31.03.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| details vom wibilea gebäude und Aussenbereich | Das Wibilea gebäude und der Aussenbereich  wurden mit reichlich details geschmückt | - | - | - |
| Teleport Punkte in die Map einfügen | Map besitzt nun Teleportpunkte | Wie man mit Area2D und Buttons wechselt | Pausenmenü führt nun zu problemen | - |

## 01.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Werkstatt Wibilea grundriss | der grundriss der werkstatt ist fertig | - | - | - |
| Pausenmenü wird neu geschrieben | Pausenmenü mit resume und Menubutton | Pausenmenu darf Process(delta ) nich stoppen | - | - |

## 06.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| grundriss Konstruktion | der Grundriss der Konstruktion ist fertig | - | - | - |
| Startbildschirm | Startbildschirm ist vorhanden und funktioniert | Wie macht man einen Startbildschirm | - | - |

## 07.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Details Werkstatt | work in progress |  | - | - |
| Bugfix + export und kompilation | Angepasste Szenen einbinden, Texturen anpassen, Einstellungen anpassen | Welche einstellungen sind wichtig | Kein release für IOS möglich | [https://docs.godotengine.org/](https://docs.godotengine.org/en/stable/) |

## 08.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| details werkstatt & konstruktion | die werkstatt und Konstruktion wurde mit details geschmückt |  | - | - |
| fehlerbehebung Map + bilder hinzufügen | alle gefundenen fehler wurden behoben | Map ist fertig | - | - |

## 13.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Fehlersuche Map | die map wurde auf Fehler mit collision shapes geprüft |  | - | - |
| Minigame | Informationssuche + Konzepte | weitere Spiele in einem Spiel einbinden | - | - |

## 14.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Godot basics wiederholt |  |  | - | -<https://docs.godotengine.org/en/stable/> |
| Testen mit Drag und Drop | Grundlage von Drag and Drop in Godot | Drag and drop mit Hilfe von Signalen und GlobalMousePosition | Mausposition ist relativ zu x,y Koordinate, anstatt zum Objekt | - |

## 15.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| automation minigame | work in progress | drag and drop in godot | * zuerst funktionierte das drag and drop | -<https://docs.godotengine.org/en/stable/> |
| Informatik game | Funktionierendes Mini Game + Drag and Drop geflickt | Drag and Drop | - nicht richtig (man konnte nur ein gegenstand bewegen) | - |

## 16.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Start Screen für minigames | ein Start Screen für alle minigames wurde eingebunden |  | - | - |
| Informatik minigame | Überarbeiten, Bilder hinzufügen, Mit anderen Szenen verknüpfen | - | - | - |

## 17.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Automation Grundgerüst | das Grobe Grundgerüst ist fertig |  | - | - |
| Hardware Information | Im Informatik Spiel gibt es nun Hardware Information | Texte in Godot einfügen | - | - |

## 27.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Automation in Hauptprojekt übertragen | Das Automation Game wurde von der Testumgebung in das Hauptspiel übertragen |  | - | - |
| Reparieren von ADB | Problem gefunden | - | - | - |

## 28.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| 1.versuch Polymechanik minigame | - | - | Polygone colisionshapes können nicht verändert werden | - |
| Automation mit Start und end screen ergänzt | Die Automation hat jetzt auch einen start und end screen | - | - | - |

## 29.04.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| 2.versuch polymechanik | versuchen mit colisionshapes tilemaps zuschneiden | geht nicht | tilemaps können nicht zugeschnitten werden | - |
| zeichnen für Konstruktion | wir haben herausgefunden wie man in godot ein “paint” programmiert. | -arbeit mit line2d | - | - |

## 04.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Polymech texturen | Polymechanik texturen erstellt |  | - | - |
| reparieren von Szenenwechsel | spiel stürzt nicht mehr ab | - | - | - |
| Minigame Konstruktion | speichern von gezeichneten Figuren | image.save\_png | - | - |

## 05.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Polymech minigame | Polymech minigame | - | - | - |
| Konstruktion Minigame | Konstruktion Basis fertig | - | - | - |

## 06.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Konstruktion Game | Start und End Screen + Verknüpfung mit der Map | - | - | - |

## 11.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Charakter Wechsel | Ansatz wie der Charakter Wechsel funktionieren könnte | - | - | - |
| Kleine Bug Fixes | - | - | - | - |

## 12.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Charakter Wechsel | Idee von gestern umgesetzt  Funktioniert aber nicht ganz | - | -Durch das Importieren des Neuen Charakter wird dieser Unscharf da er nicht als 2D Pixel Importiert wird | - |
| Kleine Bug Fixes | - | - | - | - |

## 13.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Charakter Wechsel | Lösung für Problem von Gestern und Charakter Wechsle Fertig | -Wir haben aus den 20x20px Bilder grössere Bilder Gemacht damit diese beim Import nicht geglättet werden. | -Beim Samsung Galaxy S8 und Honor 10 lite haben die Importierten Charaktere eine Komische farbe bei dem Xiaomi mi 9T Pro kommt dieses Problem nicht vor. | - |
| Kleine Bug Fixes | - | - | - | - |

## 18.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Dokumentation | Dokumentation wurde ergänzt | - | - | - |

## 19.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Dokumentation | Dokumentation wurde ergänzt | - | - | - |

## 20.05.2020

| Tätigkeit | Erreichte Arbeit/ Erreichte Ziele | Erfolge & neu gelernt | Aufgetretene Probleme | Genutzte Quellen |
| --- | --- | --- | --- | --- |
| Dokumentation | Dokumentation wurde Fertiggestellt | - | - | - |

# Testen

## Testfall 1

Testfallbeschreibung

| ID / Bezeichnung | T-001 | Download |
| --- | --- | --- |
| Beschreibung | Download der APP | |
| Testvoraussetzung | Android 9 oder 10, Handymarke Samsung Honor oder Xiaomi. | |
| Testschritte | 1. Handy entsperren. 2. Die Webseite «blwebsite.cf» aufrufen. 3. Im Burgermenu unter dem Register «Projekte» runterscrollen bis zum Titel «Wibilea Adventure». 4. Den «Download» Link anklicken. 5. Unter Releases, den obersten Eintrag herunterladen (unter Assets, link mit Endung APK). 6. Warten bis der Download abgeschlossen ist. 7. APK mit APK Installer installieren. 8. App öffnen | |
| Erwartetes Ergebnis | App startet auf. | |

Testdurchführung und Testergebnis (Mängelklasse)

|  |  |
| --- | --- |
| Testdatum | 19.05.2020 |
| Tester | Till Gasser |
| Mängelklasse\* | 0 |
| Mangelbeschreibung | - |
| Bemerkungen | -Funktioniert |
| \*Mängelklasse: 0 = mängelfrei; 1 = belangloser Mangel; 2 = leichter Mangel; 3 = schwerer Mangel; 4 = kritischer Mangel | |

## Testfall 2

Testfallbeschreibung

| ID / Bezeichnung | T-002 | Startup |
| --- | --- | --- |
| Beschreibung | Das Spiel startet | |
| Testvoraussetzung | Erfolgreicher 1.Testfall | |
| Testschritte | 1. Spiel wird durch ein kurzes Tippen auf das Icon gestartet. 2. Nach dem Ladescreen, werden 6 Spielfiguren angezeigt. 3. Nach dem Anklicken einer Figur wird auf Start geklickt. | |
| Erwartetes Ergebnis | Man kann nun die ausgewählte Figur steuern | |

Testdurchführung und Testergebnis (Mängelklasse)

|  |  |
| --- | --- |
| Testdatum | 19.05.2020 |
| Tester | Till Gasser |
| Mängelklasse | 1 |
| Mangelbeschreibung | Die Figur im Spiel ist etwas dunkler als die, welche man ausgewählt hat. |
| Bemerkungen |  |
| \*Mängelklasse: 0 = mängelfrei; 1 = belangloser Mangel; 2 = leichter Mangel; 3 = schwerer Mangel; 4 = kritischer Mangel | |

## Testfall 3

Testfallbeschreibung

| ID / Bezeichnung | T-003 | EXIT |
| --- | --- | --- |
| Beschreibung | Spiel im Hauptmenu verlassen | |
| Testvoraussetzung | Erfolgreicher 1.Testfall | |
| Testschritte | 1. Spiel wird durch ein kurzes Tippen auf das Icon gestartet 2. Nun wird der «EXIT» Knopf gedrückt | |
| Erwartetes Ergebnis | Das Spiel wird geschlossen | |

Testdurchführung und Testergebnis (Mängelklasse)

|  |  |
| --- | --- |
| Testdatum | 19.05.2020 |
| Tester | Ian Hild |
| Mängelklasse | 0 |
| Mangelbeschreibung | - |
| Bemerkungen | Funktioniert |
| \*Mängelklasse: 0 = mängelfrei; 1 = belangloser Mangel; 2 = leichter Mangel; 3 = schwerer Mangel; 4 = kritischer Mangel | |

## Testfall 4

Testfallbeschreibung

| ID / Bezeichnung | T-004 | Walk |
| --- | --- | --- |
| Beschreibung | Steuern der Spielfigur | |
| Testvoraussetzung | Erfolgreicher Testfall 2 | |
| Testschritte | 1. Es werden alle Tasten in der Linken unteren Ecke nacheinander gedrückt. 2. Danach wird das Ganze wiederholt und zusätzlich die Taste auf der rechten Seite gedrückt | |
| Erwartetes Ergebnis | Die Spielfigur bewegt sich in alle Himmelsrichtungen 1 mal, wenn die rechte Taste gleichzeitig gedrückt wird bewegt sich der Charakter schneller. | |

Testdurchführung und Testergebnis (Mängelklasse)

|  |  |
| --- | --- |
| Testdatum | 20.05.2020 |
| Tester | Ian Hild |
| Mängelklasse\* | 0 |
| Mangelbeschreibung | - |
| Bemerkungen | Funktioniert |

## Testfall 5

Testfallbeschreibung

| ID / Bezeichnung | T-005 | Pause |
| --- | --- | --- |
| Beschreibung | Öffnen des Pausenmenus | |
| Testvoraussetzung | Erfolgreicher Testfall 2 | |
| Testschritte | 1. Während dem Spielen, wird auf die Pausentaste gedrückt (Obere linke Ecke) | |
| Erwartetes Ergebnis | Es wird ein Pausenoverlay geöffnet mit den Knöpfen «Resume», «Home» und «Exit». Ausserdem wird die Steuerung ausgeblendet. | |

Testdurchführung und Testergebnis (Mängelklasse)

|  |  |
| --- | --- |
| Testdatum | 20.05.2020 |
| Tester | Ian Hild |
| Mängelklasse\* | 0 |
| Mangelbeschreibung | - |
| Bemerkungen | Funktioniert |

## Testfall 6

Testfallbeschreibung

| ID / Bezeichnung | T-006 | Teleport |
| --- | --- | --- |
| Beschreibung | Teleportieren zu den Minigames | |
| Testvoraussetzung | Erfolgreicher Testfall 2 | |
| Testschritte | 1. Nach dem Starten des Spiels, wird auf die Sprechblase der Figur geklickt, welch gegenüber dem Spieler steht. 2. Unter dem Text sind 4 Knöpfe mit Bildern (ohne den mit dem Kreuz) 3. Es wird der erste Knopf gedrückt. 4. Es wird der Pause Knopf (Links Oben) gedrückt. 5. Jetzt wird der Home Knopf gedrückt. 6. Und schritt eins und zwei werden wiederholt. 7. Nun wird der zweite Knopf gedrückt 8. Jetzt wird schritt vier und fünf wiederholt. 9. Jetzt wird schritt eins und zwei wiederholt. 10. Nun wird der dritte Knopf gedrückt. 11. Jetzt wird schritt vier und fünf wiederholt. 12. Jetzt wird schritt eins und zwei wiederholt. 13. Nun Wird der vierte Knopf gedrückt | |
| Erwartetes Ergebnis | Jeder Button Teleportiert einem zu einem anderen Ort | |

Testdurchführung und Testergebnis (Mängelklasse)

|  |  |
| --- | --- |
| Testdatum | 20.05.2020 |
| Tester | Ian Hild |
| Mängelklasse\* | 0 |
| Mangelbeschreibung | - |
| Bemerkungen | Funktioniert |

# Tabellen- und Abbildungsverzeichnis

[Tabelle 1 Spieler Textur 9](#_Toc40872840)

[Tabelle 2 Andere Texturen 9](#_Toc40872841)

[Tabelle 3 Photoshop einrichten 12](#_Toc40872842)

[Tabelle 4 Photoshop Tools 13](#_Toc40872843)

[Tabelle 5 Tastenkürzel Photoshop 14](#_Toc40872844)

[Tabelle 6 Godot Basics 14](#_Toc40872845)

[Tabelle 7 Godot Script 15](#_Toc40872846)

[Tabelle 8 Spiel erstellen 17](#_Toc40872847)

[Tabelle 9 Godot Allgemeine Einstellungen 25](#_Toc40872848)

[Tabelle 10 Steuerung 27](#_Toc40872849)

[Tabelle 11 Pausen-menu 27](#_Toc40872850)

[Tabelle 12 Informatik Game 29](#_Toc40872851)

[Tabelle 13 Automation Game 32](#_Toc40872852)

[Tabelle 14 Polymechanik Game 34](#_Toc40872853)

[Tabelle 15 Konstruktion Game 36](#_Toc40872854)

[Tabelle 16 Exportieren Android 42](#_Toc40872855)

[Tabelle 17 Exportieren Windows 44](#_Toc40872856)

[Abbildung 1 Verknüpfung der Dateien 15](#_Toc40872857)

[Abbildung 2 Spiel Plan 17](#_Toc40872858)

[Abbildung 3 Szenen Wechsel einfach 24](#_Toc40872859)

[Abbildung 4 Szenen Wechsel erweitert 24](#_Toc40872860)

[Abbildung 5 Charakterwechsel 40](#_Toc40872861)

# Glossar

| Abkürzung/Fremdwort | Erklärung |
| --- | --- |
| .gd | Dateiendung für GoDot Script |
| .godot | Dateiendung der Projektdatei |
| .PNG | Dateiendung für Bilder |
| .tres | Dateiendung für Animationen in GoDot |
| .tscn | Dateiendung für Scenen in GoDot |
| [Ctrl] | Steuerungs Taste |
| 64Bit | Aktuelle Prozessor Architektur |
| 8bit games | Retro Spieldesigne |
| Action | Aktion |
| ADB | Android Debug Bridge, möglichkeit Android Geräte vom PC zu debuggen |
| Android | Linux Kernel basierendes Betriebssystem für Smartphones |
| AnimationPlayer | Node um Animationen Abzuspielen |
| APK | Dateityp für Android Installationen |
| Area2D | Node für eine Fläche |
| Assets | Ordner für spielressourcen |
| Button | Knopf |
| C# | Programmiersprache von Microsoft |
| Camera2D | Ingame Kamera in 2D |
| CollisionShape | Fläche für Kollisionen in GoDot |
| ColorRect | Node für ein Farbbereich |
| CPU | Central Processing Unit / Prozessor |
| DeveloperID | ID für iOS Entwickler |
| Drag and Drop | Anklicken und Verschieben |
| ESC | Escape Taste |
| EXE | Ausführbare dateien für Windows |
| Gimp | Open source Bildmanipulationsprogramm |
| GLES 3.0/2.0 | API für Bildrenderung |
| GoDot | Python basierendes Programm für Spielentwicklung |
| GPU | Grafikprozessor |
| GUI | Grafische Benutzeroberfläche |
| Icon | Logo |
| iOS | Betriebssysteme für Apple Smartphones |
| IPA | Ausführbare Dateien für iOS |
| IPERKA | Projektmanagment Methode |
| Jarsigner | Entwickler Verifizierungs Tool |
| Keystore | Datei für Offentlichen und Privaten APK Key |
| KinemeticBody | Node Spielerkollisionen in GoDot |
| Main scene | Hauptszene |
| Map | Karte |
| Mini games | Mini Spiele |
| movement | Bewegeung |
| Multithreded | Mehrere Prozessorkerne werden genutzt |
| Node | Alle Objekte in GoDot |
| OpenGL ES 3/2 | Weiterentwicklung von OpenGL |
| PCK | Verpackte Spielressourcen |
| Photoshop | Kostenpflichtiges Bildbearbeitungsprogramm |
| Res:// | Projektordner |
| Screen | Bildschirm |
| Screen tearing | Bild Verzerrung |
| Script | Geschrieberne Programmiersprache |
| signals | Signale |
| Skins | Spielertexturen |
| Tilemap | Rasterfläche um in GoDot die Karte zu gestalten. |
| Tileset | Ansammlung mit Texturen für die Tilemap |
| Touch screen | Bildschirm mit Berührungssteuerung |
| Trigger | Auslöser |
| User Ordner | Ablageordner für Benutzerspezifische Dateien |
| UsV | Abkürzung für Valdrin Useini |
| Visible | Sichbar |
| Vsync | Vertikale Synchronisation der Bildschirmwiederholrate |
| WaN | Abkürzung für Noel Wangler |
| ZIP | Kompriemierter Ordner |

# Stichwortverzeichnis

\_

\_process(delta) 19, 32, 34, 35, 37, 38

A

Adb 42

Android 42, 44, 47, 48, 49, 57

AnimationPlayer 16, 17, 20

apk 42, 43, 57

Area 29, 35

area\_exited 33

Area2D 15, 30, 32, 33, 34, 35, 49

B

Betriebssystem 42

Button 9, 15, 23, 27, 29, 30, 31, 41, 49, 62

C

CollisionShape 17, 18, 32

CollisionShapes 34, 36

ColorRect 27, 38

D

Debug 43

DevelopperID 44

Drag and Drop 6, 28, 30, 31, 35, 51, 52

E

emit\_signal() 23

exe 42, 44

Export 42, 44

Exportieren 42, 44

F

func 19, 34, 37, 38, 40

G

get\_global\_mouse\_position() 32, 37

get\_tree().change\_scene() 23, 34, 35, 36, 39

get\_viewport().get\_texture().get\_data() 39

GLES 3.0 17

GoDot 5, 6, 9, 14, 15, 16, 17, 45

GUI 6, 28

I

Icon 25, 44

Input 16, 19, 20, 27, 30, 32, 37, 38

Input.is\_action\_pressed() 19

IOS 44, 50

ipa 42

IPERKA 5

J

jarsigner 42

K

Keystore 42, 43

KinematicBody 17

L

Ladescreen 25, 58

Line2D 36, 39

M

Map 21, 28, 40, 48, 49, 51, 54

Menü 9

Minigame 28, 31, 34, 35, 51, 54, 62

movement 20

movement.x 20

Multithreaded 26

N

Node 15, 16, 18, 22, 23, 28, 31, 32, 33, 37, 38, 39, 40

O

OpenGL ES 2 25

OpenGL ES 3 25

P

pass 16, 33, 37, 39

Pausenmenu 28, 50

Pausen-screen 27

pck 42, 44

Photoshop 6, 9, 11, 12, 13, 14, 40, 47

Pixel Snap 25

PNG 22, 41

pressed() 22

Projekt Einstellungen 9, 25, 27, 43, 50

R

Raster 13

Release 43

Res:// 14

S

save\_png() 39

Signal 23, 30, 37, 38, 39, 51

Spiel Charakter 9

Sprite 29, 40, 41, 45

Startszene 25

Steuerung 27, 61

Szenenwechsel 23, 24, 48, 54

T

Tasteninput 28

Tastenkürzel 14

TextureButton 41

Texturen 10, 11

TextureRect 32

Tileset 11, 13, 21, 25, 49

timeout() 37, 38, 39

Timer 36, 37

TouchscreenButton 28, 29, 32, 38, 39

Treiber 25

U

Unique Name 43

V

Version 1, 14, 25, 43

Viewport 38

ViewportContainer 38

visible = not visible 28, 30

Vsync 26

Z

zip 42