



Projektarbeit

Wibilea Adventure Game

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



Inhaltsverzeichnis

1	Die Idee	5
1.1	Brainstorming	5
1.1.1	Ideensammlung	5
2	Vorgehensweise	5
3	Informieren	5
3.1	Game Engine	5
3.2	Design	6
3.3	Was ist möglich	6
4	Planung	6
4.1	Zeitplan	7
5	Umsetzung	9
5.1	Grafisches	9
5.2	Photoshop einrichten	11
5.3	Photoshop Tools	13
5.3.1	Auswahl Tool [M]	13
5.3.2	Verschieb Werkzeug [V]	14
5.3.3	Buntstift/Pinsel [B]	14
5.3.4	Radiergummi [E]	14
5.3.5	Tastenkürzel	14
5.4	Godot	14
5.4.1	Wie sind die Dateien verknüpft	15
5.4.2	Godot-Script	15
5.5	Eigenes Spiel	16
5.5.1	Erstes Script	19
5.5.2	Animation	20
5.5.3	Tileset	21
5.5.4	Signals	22
5.5.5	Szenenwechsel	23
5.6	Allgemeine Einstellungen	25
5.7	Steuerung	27
5.8	Pausen-menu	27
6	Minigames	28
6.1	Informatikgame	28
6.2	Automatik Minigame	31
6.3	Polymechanik Minigame	34
6.4	Konstruktion Game	36
7	Charakter Auswahl	40

8	Veröffentlichen	42
8.1	Android.....	42
8.2	Windows.....	44
8.2.1	Export	44
8.2.2	Windows Installation.....	44
8.3	IOS.....	44
9	Arbeitsjournal	45
9.1	07.01.2020	45
9.2	08.01.2020	45
9.3	13.01.2020	46
9.4	28.01.2020	46
9.5	29.01.2020	47
9.6	10.02.2020	47
9.7	17.02.2020	47
9.8	03.03.2020	48
9.9	04.03.2020	48
9.10	09.03.2020	48
9.11	25.03.2020	49
9.12	31.03.2020	49
9.13	01.04.2020	49
9.14	06.04.2020	50
9.15	07.04.2020	50
9.16	08.04.2020	50
9.17	13.04.2020	51
9.18	14.04.2020	51
9.19	15.04.2020	51
9.20	16.04.2020	52
9.21	17.04.2020	52
9.22	27.04.2020	52
9.23	28.04.2020	52
9.24	29.04.2020	53
9.25	04.05.2020	53
9.26	05.05.2020	53
9.27	06.05.2020	54
9.28	11.05.2020	54

9.29	12.05.2020	54
9.30	13.05.2020	55
9.31	18.05.2020	55
9.32	19.05.2020	55
9.33	20.05.2020	56
10	Testen	57
10.1	Testfall 1	57
10.2	Testfall 2	58
10.3	Testfall 3	59
10.4	Testfall 4	60
10.5	Testfall 5	61
10.6	Testfall 6	62
11	Tabellen- und Abbildungsverzeichnis	63
12	Glossar	64
13	Stichwortverzeichnis.....	66

1 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.

1.1 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.

1.1.1 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.

2 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.

3 Informieren

3.1 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](#)

3.2 Design

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

Dies ist eine kostenlose Alternative Photoshop:

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

3.3 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.

4 Planung

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

4.1 Zeitplan

Zeitplan Basic Adventure												
Aufgaben	Soll/Ist	04.03.2020	09.03.2020	16.03.2020	23.03.2020	25.03.2020	30.03.2020	31.03.2020	01.04.2020	14.04.2020	15.04.2020	16.04.2020
zeitplan erstellen	Soll Ist											
werkstatt von innen erstellen	Soll Ist											
minigame informatik	Soll Ist											
minigame polymechanik	Soll Ist											
minigame automation	Soll Ist											
minigame konstruktion	Soll Ist											
Minigame Mediamatik	Soll Ist											
Testen	Soll Ist											
Dokumentieren	Soll Ist											
Szenenwechsel	Soll Ist											

[illegible]


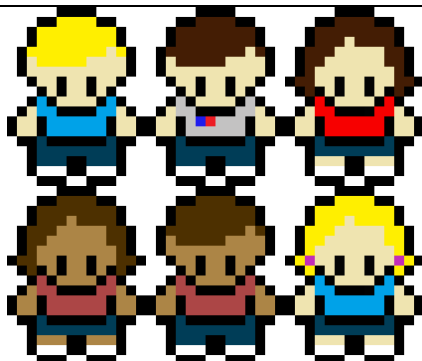
Dies ist unser Zeitplan es wurde bis auf das Mediamatik Minigame alles erreicht zusätzlich haben wir noch einen Charakter Wechsel ins Spiel eingebaut. Wir haben uns Grösstenteils an den Zeitplan gehalten ausser bei dem Szenen Wechsel da dieser vielen Probleme bereitet hat. Ausserdem kamen noch üKs dazwischen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

5 Umsetzung

5.1 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	
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.	

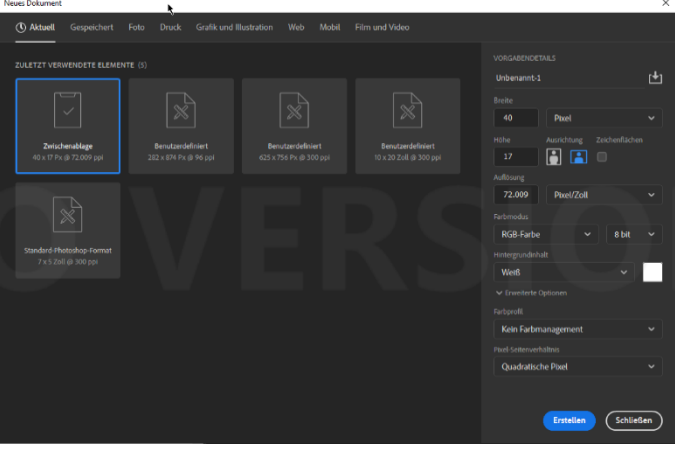
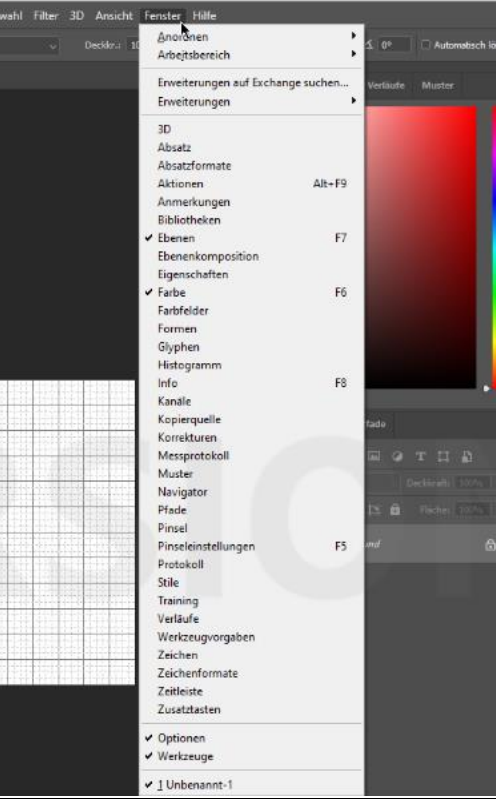
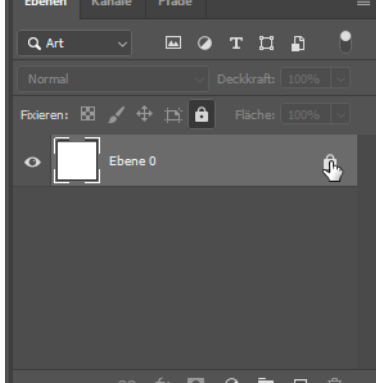
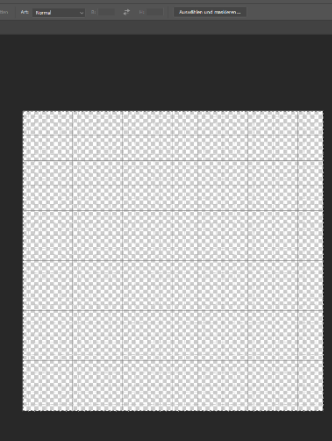
Beschreibung	Bild
<p>Dieses Menu ist im Aktuellen spiel vorhanden.</p>	
<p>Diese Texturen, werden für Wände und Boden, sowie auch Objekte im Spiel verwendet. Sie können alle auf ein Raster gezeichnet werden.</p> <p>Bei den Wänden ist es wichtig, sie so zu zeichnen, dass sie Dreidimensional wirke.</p> <p>Der Hintergrund sollte Transparent sein, damit Objekte auf die Böden gesetzt werden können ohne sie komplett zu verdecken.</p>	

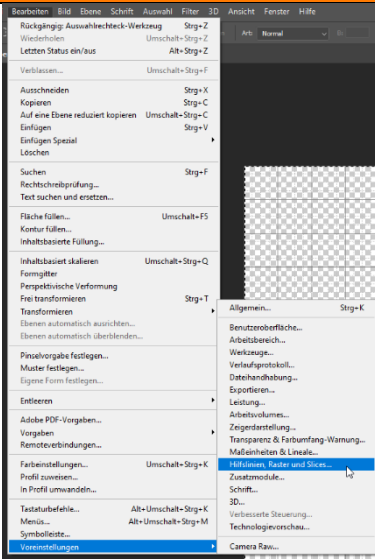
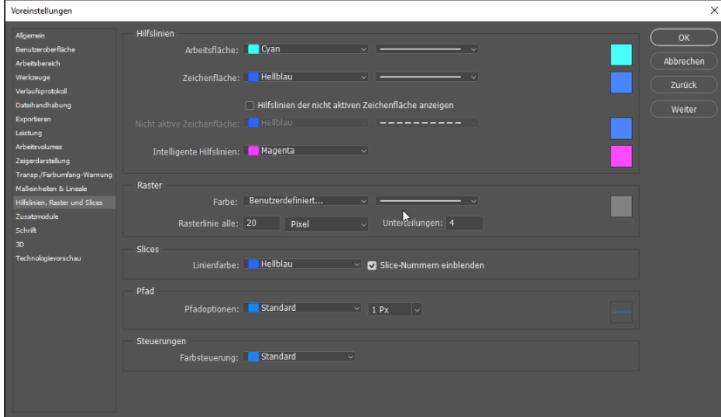
Beschreibung	Bild
<p>Diese Texturen, wurden alle im Spiel verwendet um das Tileset zu gestalten.</p>	 <p>The image displays a comprehensive set of game assets organized into several categories:</p> <ul style="list-style-type: none"> Environment & Nature: Includes various types of trees, grass patches, and stone wall textures. Furniture & Interiors: Features desks, chairs, bookshelves, and other office or room furniture. Vehicles: Shows a blue truck with the 'SIG' logo and a red truck with the 'Coca-Cola' logo. UI & Interface: Contains a health bar with the number '42069', a 'wibilea' logo, and various icons for items and actions. Structures & Obstacles: Includes a large stone wall, a staircase, and a yellow-bordered square. Other Assets: Features a lightbulb, a triangle, a rectangle, a skeleton, and a green machine.

5.2 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.

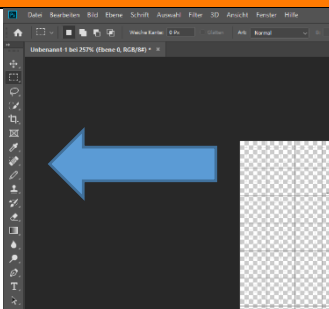
Tabelle 3 Photoshop einrichten

Beschreibung	Bild
<p>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 erstellt.</p>	
<p>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.</p>	
<p>Als nächstes sollte die Hintergrundecken 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.</p>	 

Beschreibung	Bild
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.	

5.3 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.	

5.3.1 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.

5.3.2 Verschieb Werkzeug [V]

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

5.3.3 Buntstift/Pinsel [B]

Standardmässig 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.

5.3.4 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 empfehle ich die Pixel die gelöscht werden sollen auszuwählen mit dem Auswahlwerkzeug [M] und dann mit der [Delete] taste zu löschen.

5.3.5 Tastenkürzel

Tabelle 5 Tastenkürzel Photoshop

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

5.4 Godot

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

Dazu laden wir die [64-bit Version](#) 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.

Name	Anwendung
.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.

5.4.1 Wie sind die Dateien verknüpft

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

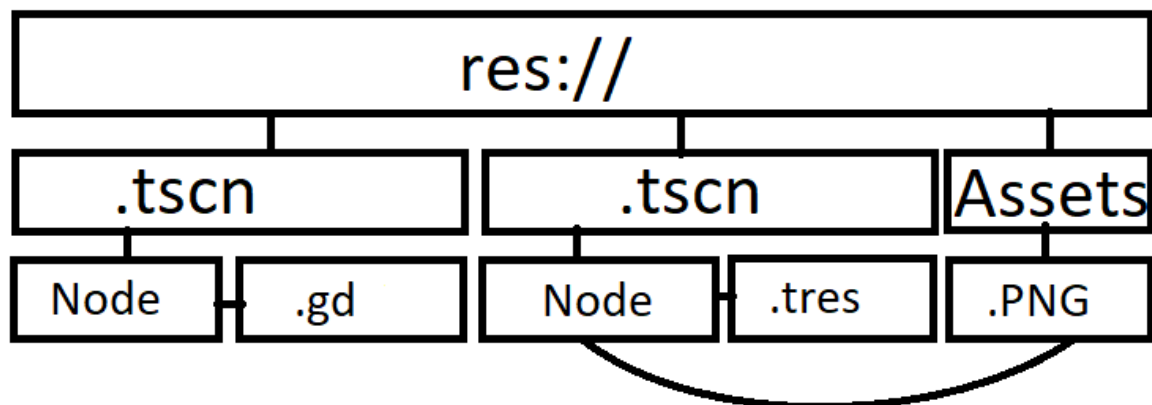


Abbildung 1 Verknüpfung der Dateien

5.4.2 Godot-Script

Tabelle 7 Godot Script

Befehl	Bedeutung
extends + Nodeart	Jedes Script beginnt mit einem extends und der Art der Node z.B. «extends Area2D»

Befehl	Bedeutung
<code>func_trigger</code>	Eine Funktion hat einen Trigger z.B. «_pressed» das heisst sobald diese Node angeklickt wird, wird die Funktion ausgeführt.
<code>If Input.is_action_pressed(«left»):</code>	If stellt eine Bedingung, heisst WENN ein Input kommt, welcher von einer Taste kommt, den Namen left trägt, dann wird das darauffolgende ausgeführt.
<code>If Input.is_action_pressed(«left»):</code> <code>⇒ get_tree().quit</code>	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.
<code>If Input.is_action_pressed(«left»):</code> <code>⇒ get_tree().quit</code> <code>If Input.is_action_pressed(«right»):</code> <code>⇒ get_tree().quit</code>	<p>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.</p> <p>Das Programm geht nun zum ersten If und wenn dies erfüllt ist, beendet es das Spiel.</p> <p>Wenn es nicht erfüllt ist, geht es zum nächsten If und so weiter.</p>
<code>pass</code>	Wenn man pass unter eine Funktion schreibt, wird diese übersprungen
<code>\$Node.befehl</code>	Mit \$ kann eine andere Node in der Szene angesprochen werden und nach dem Punkt kann ein Befehl gegeben werden z.B. <code>\$AnimationPlayer.play(«name_der_animation»)</code>

5.5 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.

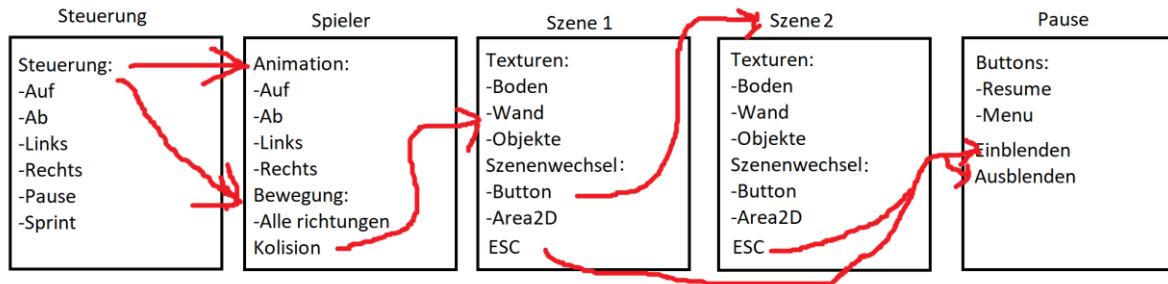
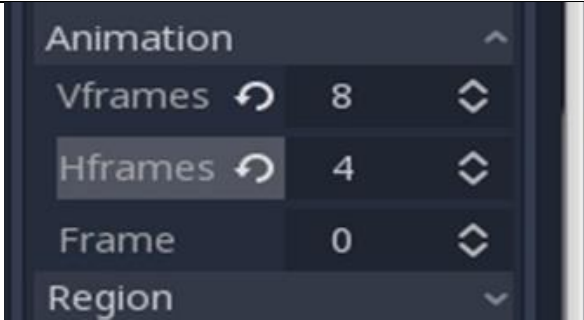

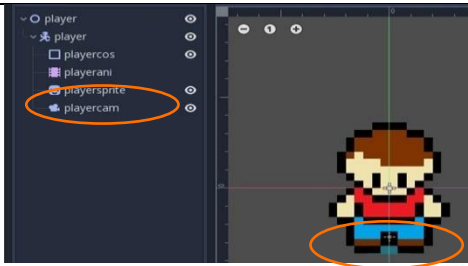
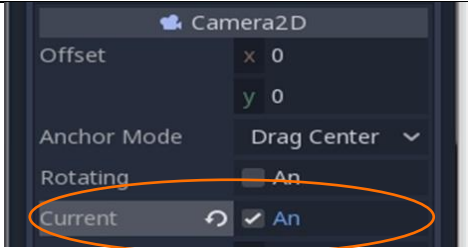




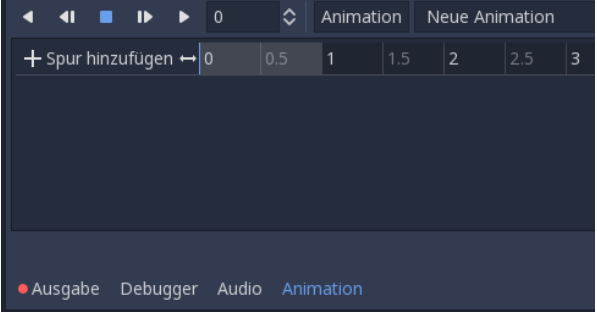
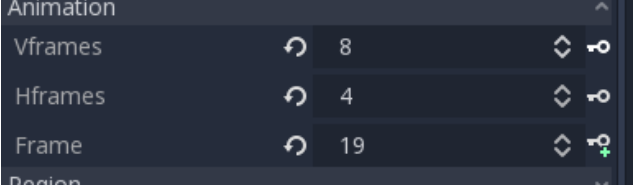
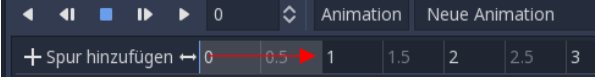
Abbildung 2 Spiel Plan


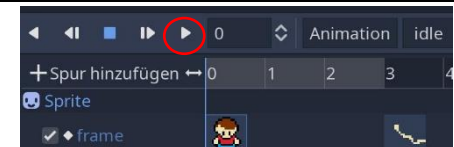
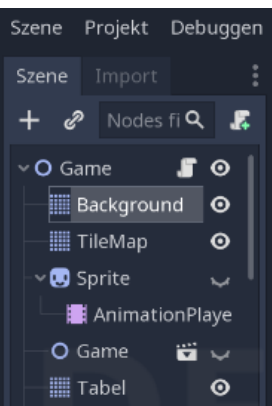
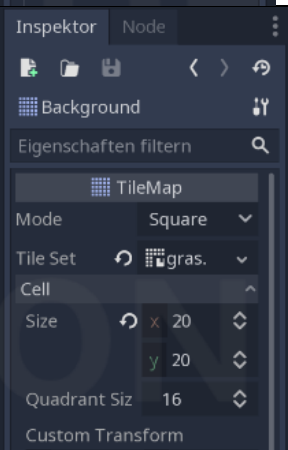
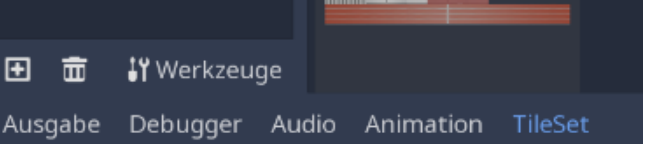
Tabelle 8 Spiel erstellen

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 sehr «verpixelt» und muss deshalb noch verändert werden. Nämlich macht man einen Rechtsklick auf die Textur des Charakters und geht zum Register Import und stellt	

Erklärung	Bild
<p>unter «Voreinstellungen» auf «2D Pixel» um.</p>	
<p>Da die Textur noch aus 32 Bildern besteht, muss sie noch angepasst werden.</p> <p>Dazu geht man auf die rechte Seite des Bildschirms und verändert folgende Optionen.</p>	
<p>Bei der Kategorie «Animation» wird eingegeben, in wie viele Zeilen und Spalten das Bild eingeteilt werden kann, um ein einzelnes zu erstellen.</p>	
<p>Die CollisionShape wird nun an den Charakter angepasst und alle «Nodes» werden so umbenannt, dass man weiss, was sie tun.</p> <p>Ausserdem wird eine «Camera2D» Hinzugefügt.</p>	
<p>Die Kamera besitzt folgende Optionen.</p> <p>Current bedeutet, dass sich die Kamera mit dem Spieler mitbewegt</p>	

Erklärung	Bild
5.5.1 Erstes Script	
<p>Damit man den Spieler bewegen kann, muss man ihm ein «script» hinzufügen.</p> <p>Jedes script wird mit einem «extend+Nodenamen» begonnen.</p> <p>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»</p>	 <pre> 1 extends KinematicBody2D 2 3 func _ready(): 4 pass 5 func _process(delta): 6 var movement = Vector2(0,0) 7 8 #-----Gerade Laufen 9 10 if Input.is_action_pressed("left"): 11 if Input.is_action_pressed("right") == false: 12 if Input.is_action_pressed("down") == false: 13 if Input.is_action_pressed("up") == false: 14 movement.x = -1 15 \$AnimationPlayer.play("walk_left") 16 if Input.is_action_pressed("right") == false: 17 movement.x = 1 18 \$AnimationPlayer.play("walk_right") 19 if Input.is_action_pressed("right"): </pre>
<p>Damit das Spiel weiss, wann es welche Befehle ausführen muss, nimmt man den Befehl «if» (auf Deutsch «wenn»)</p>	 <pre> #-----Gerade Laufen----- if Input.is_action_pressed("left"): if Input.is_action_pressed("right") == false: if Input.is_action_pressed("down") == false: if Input.is_action_pressed("up") == false: </pre>
<p>Input.is_action_pressed(«left»)</p> <p>Mit dem Input Befehl, wird gezeigt, dass es sich um eine Eingabe handelt.</p> <p>Das is_action_pressed, heisst dass es eine Aktion ist.</p> <p>Und in der Klammer ist der Name der Aktion, welche in den Projekteinstellungen eingestellt werden kann.</p> <p>Durch den Doppelpunkt am Ende der Linie wird gezeigt, dass nur wenn diese Bedingung erfüllt wird, die nächste Linie gelesen wird.</p>	 <pre> if Input.is_action_pressed("left"): if Input.is_action_pressed("right") == false: </pre>

Erklärung	Bild
<p>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.</p> <p>movement ist die zuvor definierte variabel, für die aktuelle Position 0,0.</p> <p>Durch drücken, der Taste «left» wird diese dann um 32 Pixel nach links verschoben.</p>	 <pre> var movement = Vector2(0,0) # Gerade Laufen if Input.is_action_pressed("left"): if Input.is_action_pressed("right") == false: if Input.is_action_pressed("down") == false: if Input.is_action_pressed("up") == false: movement.x = -32 \$AnimationPlayer.play("walkleft") if Input.is_action_pressed("sprint"): movement.x = -64 </pre>
5.5.2 <u>Animation</u>	
<p>Um Animationen abzuspielen, müssen diese zuerst erstellt werden, dies funktioniert so:</p>	
<p>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.</p> <p>Nach dem man die «Zeit» auf 0 gestellt hat, kann man Texturen aussuchen.</p>	
<p>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.</p>	
<p>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.</p>	

Erklärung	Bild
<p>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.</p>	
<p>Auf dem kleinen Dreieck kann man die Animation auch laufen lassen und schauen ob sie gut aussieht.</p>	
<h3>5.5.3 <u>Tileset</u></h3>	
<p>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.</p>	
<p>Um ein Tileset zu erstellen muss man erstmal auf eine TileMap klicken diese kann man ganz einfach mit dem kleinen + in sein Projekt einfügen.</p>	
<p>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.</p>	
<p>Auf der unteren seite des Bildschirms öffnen sich ein Fenster, hier muss man auf das Plus drücken um eine PNG datei mit Texturen einzufügen.</p>	

Erklärung	Bild
<p>Es öffnet sich ein neues Fenster hier muss man eine png datei mit texturen aussuchen in unserem fall Gras2.png</p>	
	
5.5.4 <u>Signals</u>	
<p>Für die Kommunikation zwischen Nodes, gibt es «signals» alle Arten von nodes haben verschiedene signals, welche ausgegeben werden, wenn eine bedingung erfüllt wird.</p> <p>Das Signal pressed() wird ausgegeben, wenn diese Node angeklickt wird.</p>	

Erklärung	Bild
<p>Signale können auch selbst definiert werden indem man in das Script der Node, signal name schreibt und dieses durch eine Bedingun ausgegeben wird.</p> <p>Dies geschieht mit <code>emit_signal(«name»)</code></p>	 <pre> 1 extends Button 2 3 signal name 4 5 func _process(delta): 6 emit_signal("name") </pre>
<p>Dieses Signal wird nun bei den Signalen der Node angezeigt.</p> <p>Durch doppelklick kann diese node nun verknüpft werden.</p>	
<p>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.</p>	
<p>Wenn eine Node ausgewählt wurde, kann nun unter die neu entstandene funktion geschrieben werden, was geschieht, wenn das Signal «name» empfangen wird.</p>	 <pre> 1 extends Panel 2 3 4 5 func _on_Button_name(): 6 pass # Replace with function body. 7 </pre>
5.5.5 Szenenwechsel	
<p>Da die meisten Speile aus verschiedenen Szenen bestehen, muss es möglich sein diese zu wechseln.</p> <p>Dies geschieht mit <code>«get_tree().change_scene(«Name der scene»)</code></p>	 <pre> 1 extends Panel 2 3 func _on_goto_scene_pressed(): 4 get_tree().change_scene("res://scene_b.tscn") 5 </pre>

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

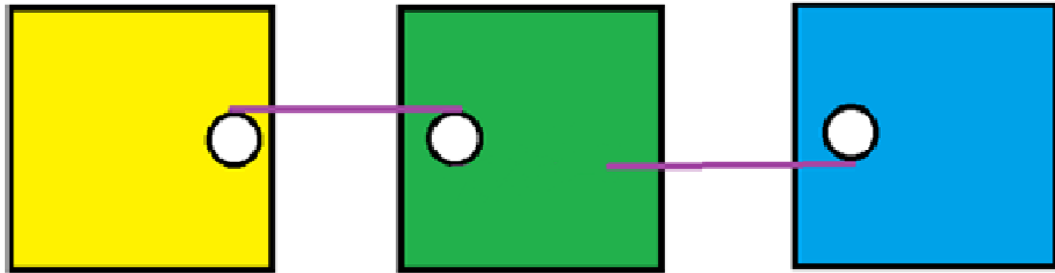


Abbildung 3 Szenen Wechsel einfach

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.

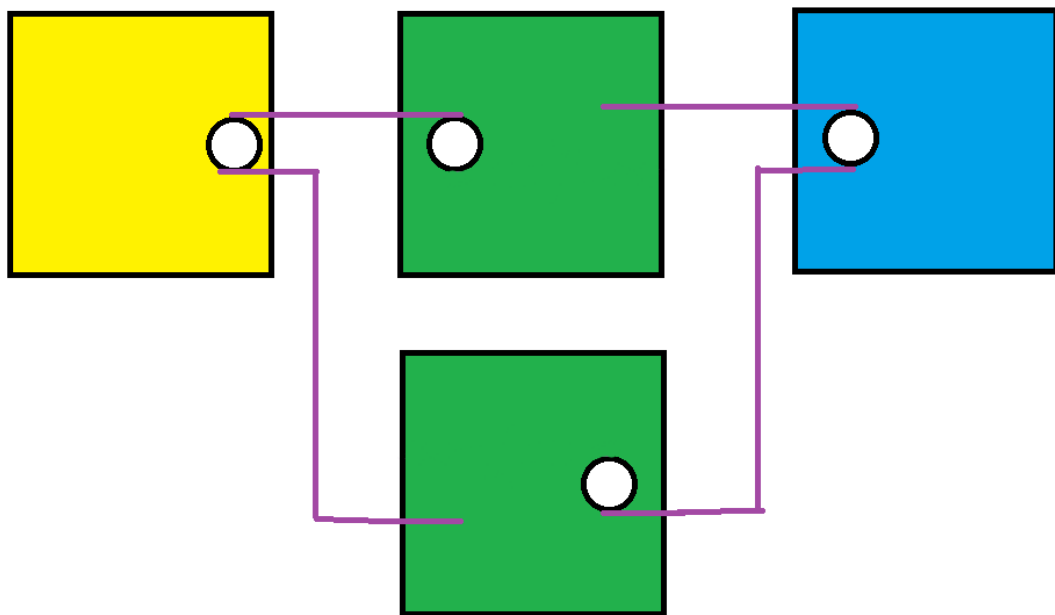


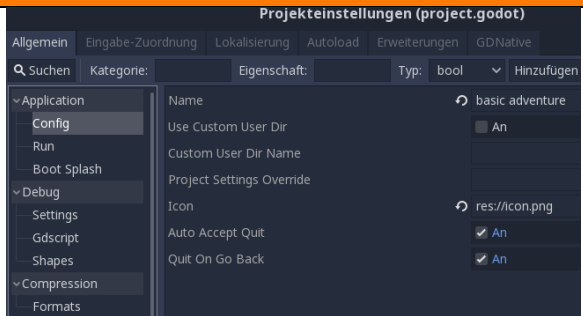
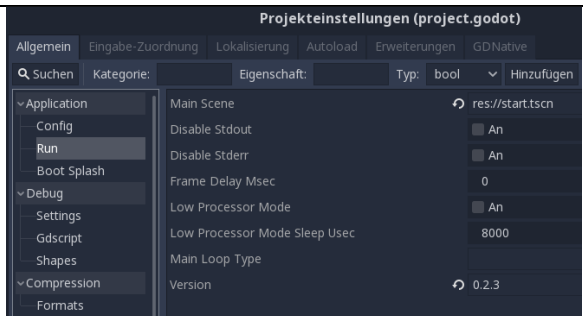
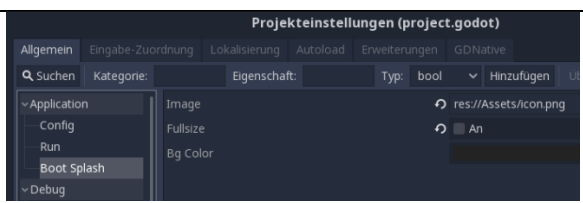

Abbildung 4 Szenen Wechsel erweitert

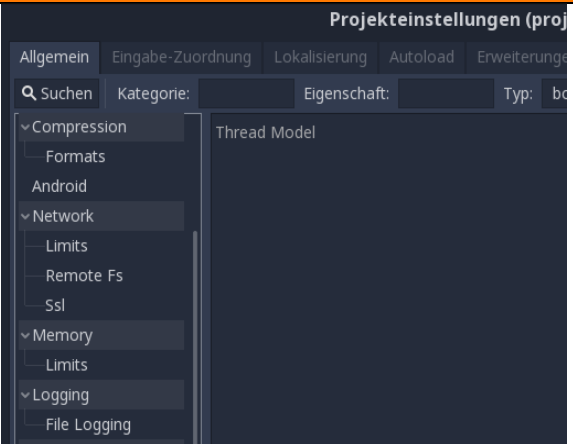
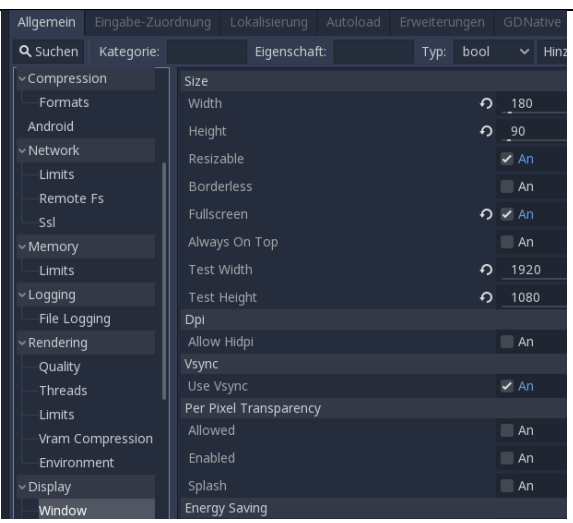
Beim Wechsel von Gelb zu Grün, wird die obere grüne Szene geladen. Beim Wechsel von Blau zu Grün, wird die untere grüne Szene geladen.

5.6 Allgemeine Einstellungen

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

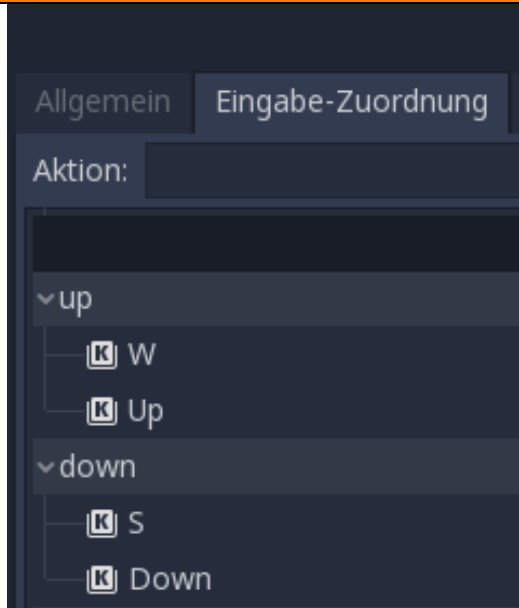
Tabelle 9 Godot Allgemeine Einstellungen

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 Tilessets entstehen.	

Erklärung	Bild
<p>Da heute die meisten Prozessoren Multithreaded sind, sollte man diese Option einschalten.</p>	
<p>Es empfiehlt sich den Vsync zu aktivieren, um Screen tearing zu verhindern.</p> <p>Screen tearing bedeutet, dass sich das Bild verzieht, weil die Wiederholrate nicht synchron ist.</p>	

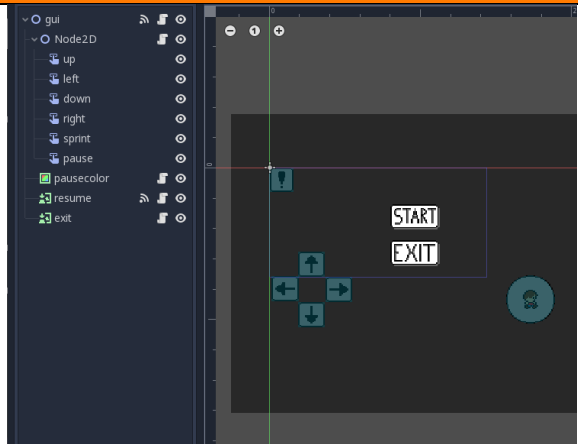
5.7 Steuerung


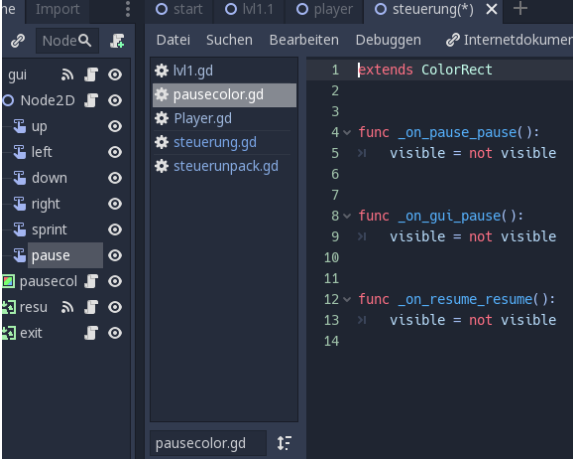
Tabelle 10 Steuerung

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

5.8 Pausen-menu

Tabelle 11 Pausen-menu

Erklärung	Bild
<p>Wichtig für eine Pausen-screen ist, dass er das Spiel überdeckt und man durch ihn das Spiel verlassen oder fortsetzen kann.</p> <p>Dazu integrieren wir in der Steuerung, ein «ColorRect» und 2 «Button», für Exit und Resume.</p>	

Erklärung	Bild
<p>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.</p> <p>Und das Gleiche wird auch ausgelöst, durch den Tasteninput «menu», welcher durch die «esc» Taste getriggert.</p>	 <pre> 1 extends Node2D 2 signal pause 3 func _process(delta): 4 if Input.is_action_just_pressed("menu"): 5 emit_signal("pause") 6 </pre>
<p>Nun sendet die Oberste Node ein Signal an das gesamte GUI, welche ausgeblendet werden «visible = not visible».</p> <p>Das selbe geschieht mit dem zuvor ausgeblendeten Pausenmenu.</p>	 <pre> 1 extends ColorRect 2 3 4 func _on_pause_pause(): 5 visible = not visible 6 7 8 func _on_gui_pause(): 9 visible = not visible 10 11 12 func _on_resume_resume(): 13 visible = not visible 14 </pre>


6 Minigames

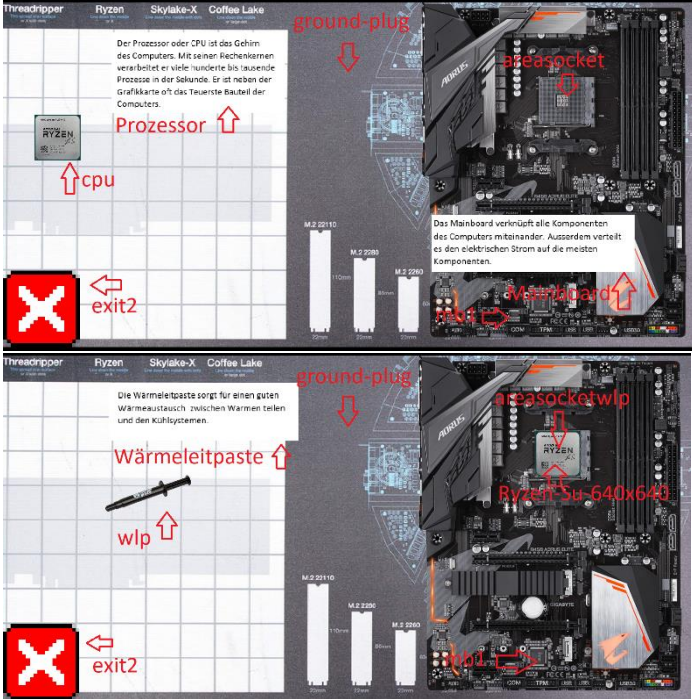
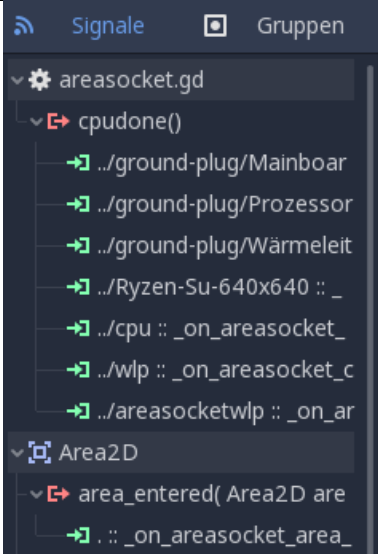
Die Minigames sollen unsere noch leere Map mit Leben füllen, und die Berufe der Wibilea ganz simpel vorstellen. Für die Minigames wird hier auf das bisherige Script verzichtet und es werden neue Szenen erstellt.

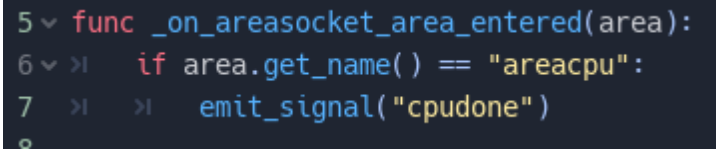
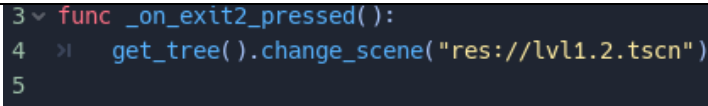
6.1 Informatikgame

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

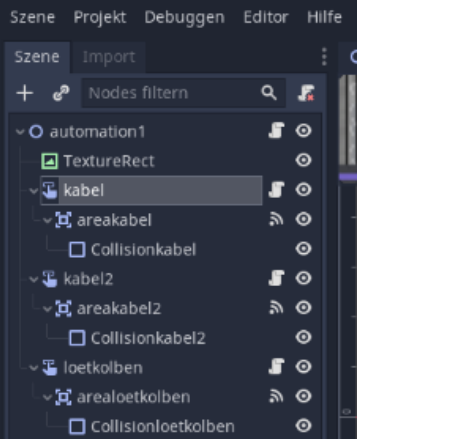
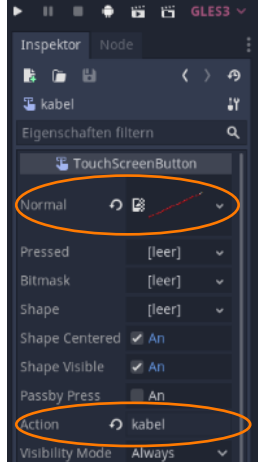
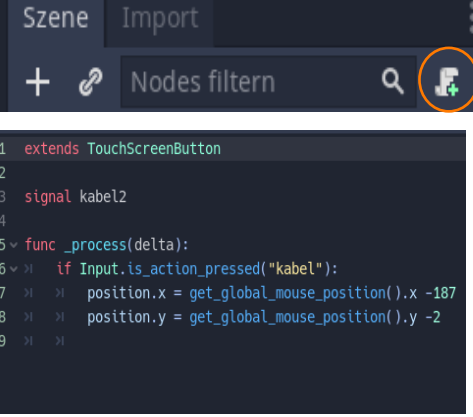
Erklärung	Bild
<p>Hier ist der Fertige Aufbau der Szene. «ground-plug» ist der Hintergrund. Auf ihm befinden sich ausgeblendete Bilder für den Beschrieb von den Teilen.</p> <p>Es folgen weitere Sprites, welche die Teile an ihrem Platz anzeigen, wenn die Aufgabe richtig gelöst wurde.</p> <p>Die TouchscreenButtons sind die «Komponenten», sie haben die deren Textur (CPU, GPU etc.)</p> <p>Die Areas werden genutzt um eine Kollision mit den Teilen zu erkennen. Und dann fest zu stellen, ob es das Richtige Teil ist.</p> <p>Der Button namens exit2 ist für das Vorzeitige Verlassen des Spiels.</p>	

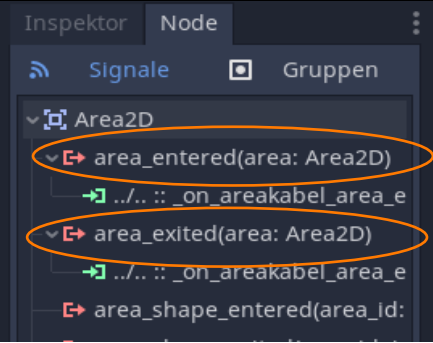
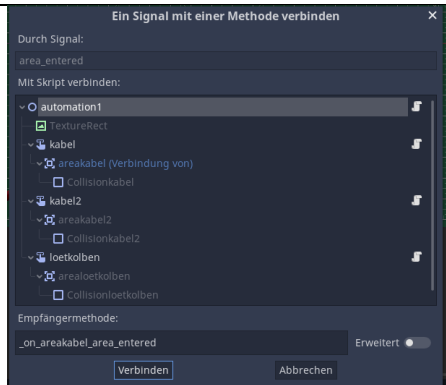
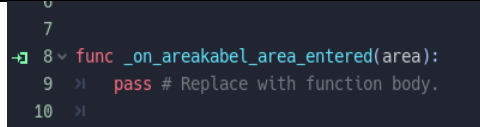
Erklärung	Bild
<p>Die Abfolge in dieser Szene ist folgende: durch klicken auf den Button «cpu», kann man diesen über den Screen bewegen. Sobald dieser auf der Area namens «areasocket» ist, sendet diese ein Signal an sich selber, cpu, Ryzen-Su-640x640, und wlp.</p> <p>Alle bekommen den Befehl (visible = not visible)</p> <p>Nun wird das ganze wiederholt mit dem Button wlp</p>	
<p>Der Komplizierteste des Games ist das «Drag and Drop»</p> <p>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</p>	<pre> 1 extends TouchScreenButton 2 3 func _process(delta): 4 if Input.is_action_pressed("cpuon"): 5 position.x = get_global_mouse_position().x - 6 6 position.y = get_global_mouse_position().y - 6 7 8 func _on_areasocket_cpudone(): 9 visible = not visible </pre>
<p>Der Rest wird mit Signalen verknüpft welche von der Area2D gesendet werden. Diese werden mit allen Objekten verbunden, welche angezeigt oder ausgeblendet werden sollen.</p>	

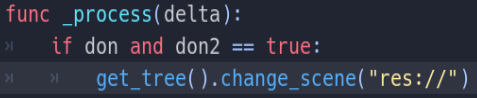
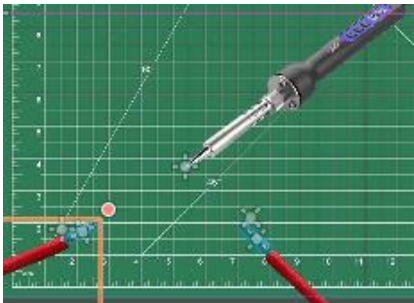
Erklärung	Bild
<p>Damit die Area nur Signale aussendet, wenn der Richtige Button Kollidiert, wird folgendes Script in die Area geschrieben.</p> <p>Das heisst, dass nur dieses als Signal verwendet werden darf, weil die Area sonst bei allen Kollisionen aussenden würden.</p>	 <pre> 5 func _on_areasocket_area_entered(area): 6 if area.get_name() == "areacpu": 7 emit_signal("cpudone") </pre>
<p>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.</p>	 <pre> 3 func _on_exit2_pressed(): 4 get_tree().change_scene("res://lvl1.2.tscn") 5 </pre>

6.2 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.

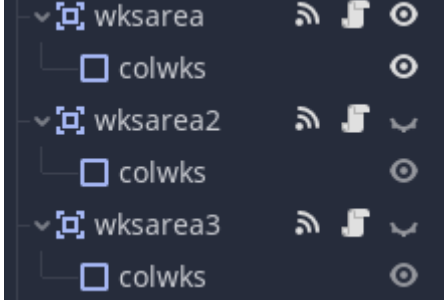
Erklärung	Bild
<p>So Sieht der Aufbau des ersten Levels aus. Man Sieht drei «TouchscreenButtons» (kabel, kabel2 und loetkolben), diese haben je eine Area2D und eine CollisionShape. Zuoberst hat es dann noch ein TextureRect das ist der Hintergrund.</p>	
<p>Alle die TouchscreenButtons haben am Anfang keine 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.</p> <p>Bei uns Hat unser Kabel eine Textur von einem Kabel und eine «Action» die «kabel» heisst diese «Action» wird später wichtig sein.</p>	
<p>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:</p> <pre>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</pre> <p>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.</p>	

Erklärung	Bild
<p>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</p>	
<p>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».</p>	
<p>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.</p>	
<p>Als nächstes erstellen wir 2 variablen Diese schreiben wir ganz oben im Skript. Diese müssen beide den Wert 'false' haben.</p>	<pre>1 extends Node2D 2 3 var don = false 4 var don2 = false</pre>
<p>Unter der Funktion Löschen wir das pass und den Kommentar. Jetzt schreiben Folgendes in der oberen Funktion: <code>if area.get_name() == "arealoetkolben":</code> <code>don = true</code> in der Unteren können wir das Gleiche schreiben nur wird hier das don = false.</p>	<pre>8 func _on_areakabel_area_entered(area): 9 if area.get_name() == "arealoetkolben": 10 don = true 11 func _on_areakabel_area_exited(area): 12 if area.get_name() == "arealoetkolben": 13 don = false</pre>
<p>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.</p>	<pre>8 func _on_areakabel_area_entered(area): 9 if area.get_name() == "arealoetkolben": 10 don = true 11 func _on_areakabel_area_exited(area): 12 if area.get_name() == "arealoetkolben": 13 don = false 14 15 func _on_areakabel2_area_entered(area): 16 if area.get_name() == "arealoetkolben": 17 don2 = true 18 func _on_areakabel2_area_exited(area): 19 if area.get_name() == "arealoetkolben": 20 don2 = false</pre>

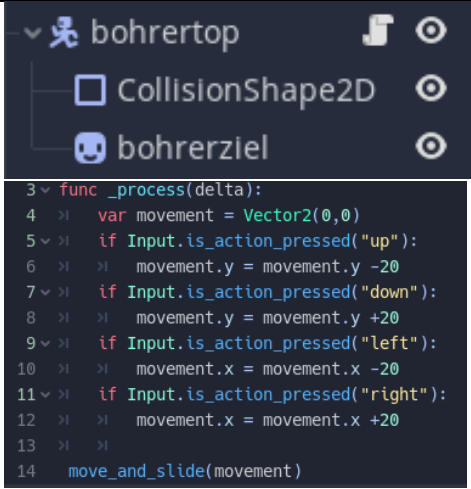
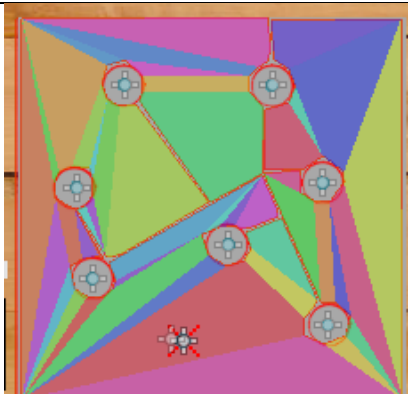
Erklärung	Bild
<p>Als letztes erstellen wir eine neue Funktion und in diese schreiben wir:</p> <pre>func _process(delta): if don and don2 == true: get_tree().change_scene("res://")</pre> <p>Wenn die Area2Ds der Kabel mit dem Lötkolben kollidieren werden don und don2 true und wenn diese true sind wird diese Funktion ausgelöst und man wird in die nächste Szene teleportiert.</p>	
<p>Jetzt sollten noch alle CollisionShapes richtig platziert werden damit es richtig funktioniert.</p>	

6.3 Polymechnik Minigame

Tabelle 14 Polymechnik Game

Erklärung	Bild
<p>Beim Polymechnik Minigame wird hauptsächlich mit Animationen und CollisionShapes gearbeitet.</p>	
<p>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.</p>	<pre>1 extends Sprite 2 var arbeit = false 3 var c1 = false 4 var c2 = false 5 var c3 = false 6 var c4 = false 7 var c5 = false 8 signal fettig 9 func _on_anschalten_pressed(): 10 \$aniwks.play("drehbankidle") 11 yield(get_tree().create_timer(.1), "timeout") 12 arbeit = true 13 14 func _on_wksarea_area_entered(area): 15 if area.name == "werk" and arbeit == true: 16 \$aniwks.play("cutting1") 17 18 c1 = true 19 arbeit = false</pre>


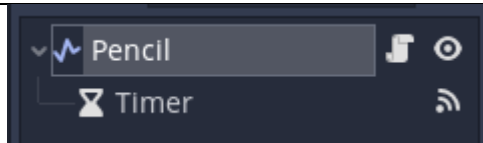
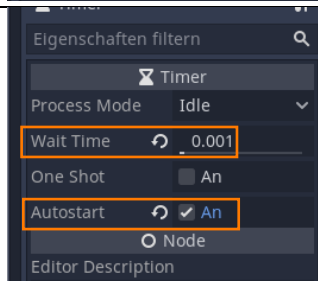
Erklärung	Bild
<p>Wenn alle Areas berührt wurden sind alle variablen auf true und im <code>_process(delta)</code> wird nach ablaufen eines Timers dies Szene gewechselt, mit Hilfe von <code>get_tree().change_scene(«pfad»)</code>.</p>	<pre> 1 extends Node2D 2 var area = false 3 var area2 = false 4 var area3 = false 5 var area4 = false 6 var area5 = false 7 func _on_wksarea_area(): 8 area = true 9 10 func _on_wksarea2_area2(): 11 area2 = true 12 13 func _on_wksarea3_area3(): 14 area3 = true 15 16 func _on_wksarea4_area4(): 17 area4 = true 18 19 func _on_wksarea5_area5(): 20 area5 = true 21 func _process(delta): 22 if area == true and area2 == true and area3 == true and area4 == true and area5 == true: 23 yield(get_tree().create_timer(2), "timeout") 24 get_tree().change_scene("res://polymechnikstart2.tscn") </pre>
<p>Durch die verschiedenen Areas, werden verschiedene Animationen abgespielt.</p>	
<p>Die Animationen zeigen, wie ein Werkstück bearbeitet wird, indem sie je einen Pixel tiefer geschnitten sind.</p>	
<p>Das Werkzeug, welches zur Bearbeitung des Werkstücks genutzt wird, kann per Drag and Drop bewegt werden.</p>	<pre> func _process(delta): if Input.is_action_pressed("werk") and phys == true: position.x = get_global_mouse_position().x - 5 position.y = get_global_mouse_position().y - 10 </pre>
<p>Wenn die unterste Area2D erreicht ist, wird die Szene gewechselt</p>	<pre> if area == true and area2 == true and area3 == true and area4 == true and area5 == true: yield(get_tree().create_timer(2), "timeout") get_tree().change_scene("res://polymechnikstart2.tscn") </pre>
<p>Das nächste Polymechnik Minigame verlangt, mit einem Bohrer (rotes Kreuz) eingezeichnete Löcher in ein Werkstück zu bohren.</p>	

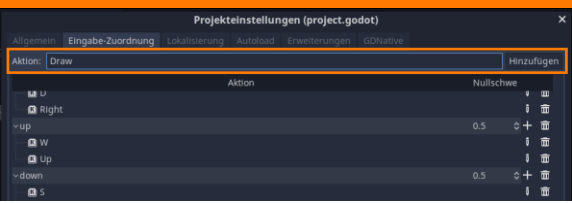
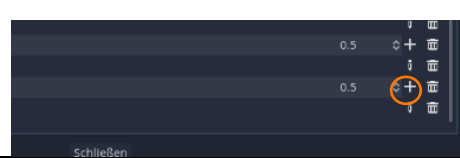
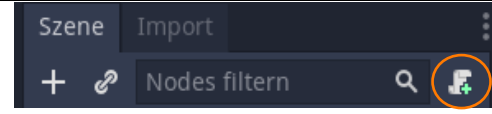
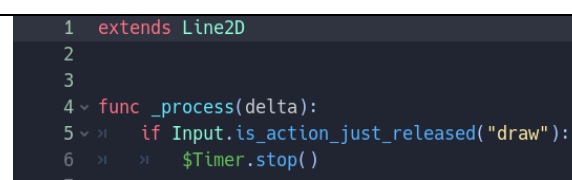
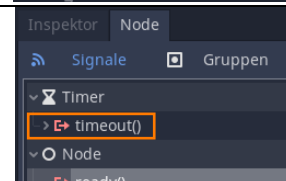
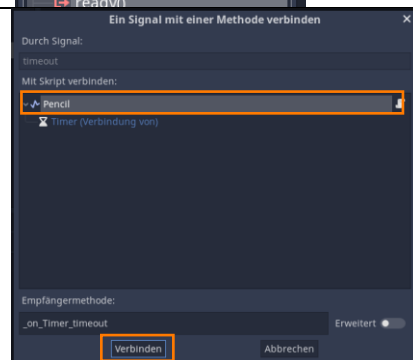
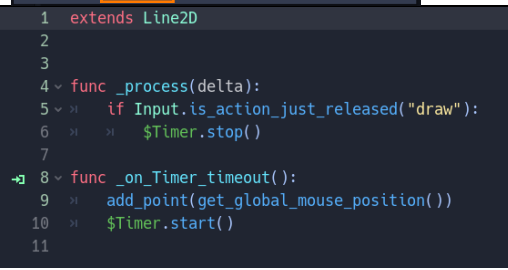
Erklärung	Bild
<p>Der Bohrer wird gleich bedient, wie der Spieler. Er muss eine Kollision haben, damit er nicht das Spielfeld verlässt. Das Spielfeld wird deshalb mit CollisionShapes umkreist. Um den KinematicBody zu steuern, werden 5 Touchbuttons erstellt, welche befehle «up» «down» «left» «right» «bohren»</p> <p>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</p>	 <pre> 3 func _process(delta): 4 var movement = Vector2(0,0) 5 if Input.is_action_pressed("up"): 6 movement.y = movement.y -20 7 if Input.is_action_pressed("down"): 8 movement.y = movement.y +20 9 if Input.is_action_pressed("left"): 10 movement.x = movement.x -20 11 if Input.is_action_pressed("right"): 12 movement.x = movement.x +20 13 14 move_and_slide(movement) </pre>
<p>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»)</p> <p>Wenn alle Löcher getroffen wurden, wird die nächste Szene geladen.</p>	

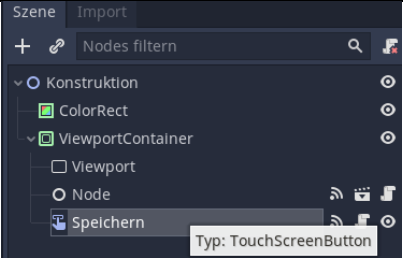
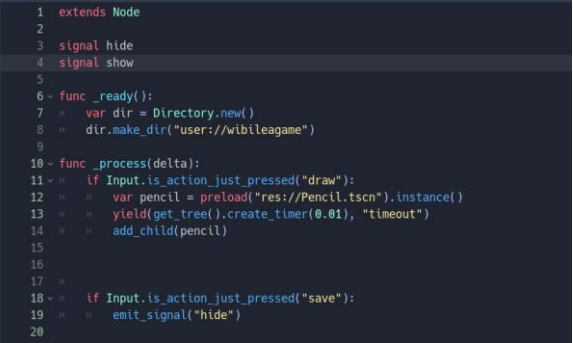

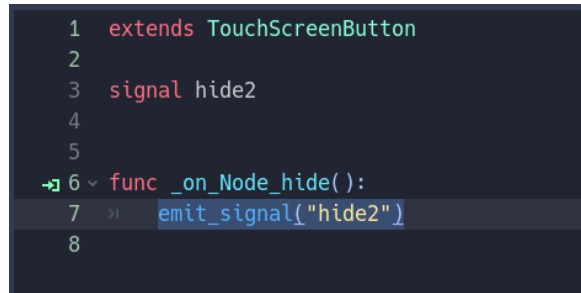
6.4 Konstruktion Game


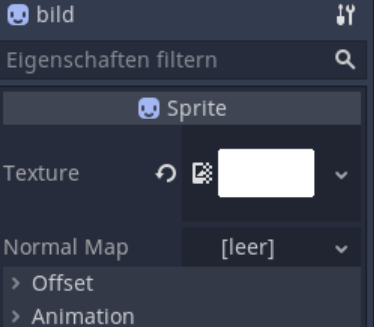
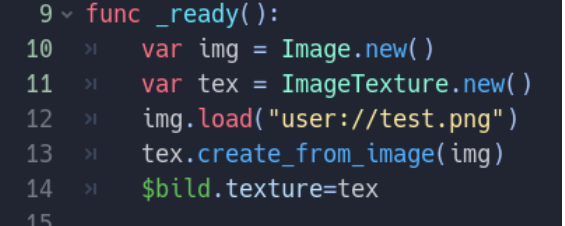
Im Game der Konstruktion kann man etwas zeichnen, diese Zeichnung kann man dann auf der Wand in der Konstruktion sehen.

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.	

Erklärung	Bild
<p>Bevor wir weiter machen können müssen wir oben auf Projekt->Projekteinstellungen->Eingabe-Zuordnung Hier fügen wir die neue aktion Draw ein.</p>	
<p>Danach Scrollen wir runter und klicken auf das + neben Draw und wählen hier Linkemaustaste aus.</p>	
<p>Nun erstellen wir ein leeres Skript für den Pencil indem wir den Pencil auswählen und oben auf die Schriftrolle klicken.</p>	
<p>Ins Skript kommt nun folgender Code:</p> <pre>func _process(delta): if Input.is_action_just_released("draw"): \$Timer.stop()</pre>	
<p>Als nächstes gehen wir wieder in den Timer und erstellen rechts im Tab node ein signal. Dazu doppelklicken wir timeout() an.</p>	
<p>es öffnet sich ein neues Fenster hier wählen wir den Pencil aus und klicken auf verbinden.</p>	
<p>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:</p> <pre>Add_point(get_global_mouse_position()) \$Timer.start()</pre>	
<p>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.</p>	

Erklärung	Bild
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: <code>extends Node</code> <code>signal hide</code> <code>signal show</code> <code>func _ready():</code> <code>var dir = Directory.new()</code> <code>dir.make_dir("user://wibileagame")</code> <code>func _process(delta):</code> <code>if Input.is_action_just_pressed("draw"):</code> <code>var pencil = preload("res://Pencil.tscn").instance()</code> <code>yield(get_tree().create_timer(0.01), "timeout")</code> <code>add_child(pencil)</code> <code>if Input.is_action_just_pressed("save"):</code> <code>emit_signal("hide")</code>	
Als nächstes fügen wir dem TouchscreenButton auch ein Skript hinzu mit folgendem Code: <code>extends TouchScreenButton</code> <code>signal hide2</code>	
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 <code>emit_signal("hide2")</code> ersetzen.	

Erklärung	Bild
<p>Jetzt erstellen wir nochmals ein Signal das Vom TouchscreenButton ausgeht.</p> <ol style="list-style-type: none"> 1. TouchscreenButton Auswählen. 2. Oben rechts auf Node wechseln. 3. Doppelklick auf hide2(). 4. Das Node doppelklicken. <p>pass durch folgenden Code ersetzen:</p> <pre>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")</pre> <p>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 <code>get_viewport().get_texture().get_data()</code> dann müssen wir das Bild noch Flippen damit es richtigerum angezeigt wird und zuletzt wird es mit <code>.save_png('pfad')</code> im erstellten Verzeichnis gespeichert</p>	 <pre>1 extends Node 2 3 signal hide 4 signal show 5 6 func _ready(): 7 var dir = Directory.new() 8 dir.make_dir("user://wibileagame") 9 10 func _process(delta): 11 if Input.is_action_just_pressed("draw"): 12 var pencil = preload("res://Pencil.tscn").instance() 13 yield(get_tree().create_timer(0.01), "timeout") 14 add_child(pencil) 15 16 17 18 if Input.is_action_just_pressed("save"): 19 emit_signal("hide") 20 21 22 23 func _on_Speichern_hide2(): 24 yield(get_tree().create_timer(0.2), "timeout") 25 yield (get_tree(), "idle_frame") 26 yield (get_tree(), "idle_frame") 27 var image = get_viewport().get_texture().get_data() 28 image.flip_y() 29 image.save_png("user://test.png") 30 get_tree().change_scene("res://Konstruktion_ende.tscn") 31</pre>
<p>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.</p>	 <p>The screenshot shows the Godot Inspector for a 'Sprite' node. The 'Texture' property is set to a white square icon. The 'Normal Map' property is set to '[leer]'. There are expandable sections for 'Offset' and 'Animation'.</p>
<p>Jetzt kommt in die Haupt Node noch folgender Code:</p> <pre>func _ready(): var img = Image.new() var tex = ImageTexture.new() img.load("user://test.png") tex.create_from_image(img) \$bild.texture=tex</pre>	 <pre>9 func _ready(): 10 var img = Image.new() 11 var tex = ImageTexture.new() 12 img.load("user://test.png") 13 tex.create_from_image(img) 14 \$bild.texture=tex 15</pre>

Erklärung	Bild
Es ist wichtig dass der Code in <code>func_ready()</code> steht weil Diese Funktion die Textur die wir vorhin gespeichert haben ladet. Deswegen muss diese Funktion immer als erstes Geladen werden.	

7 Charakter Auswahl

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

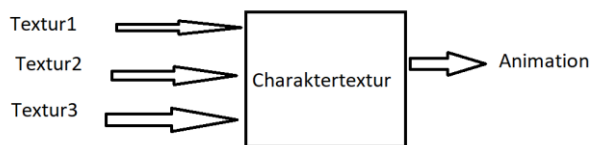
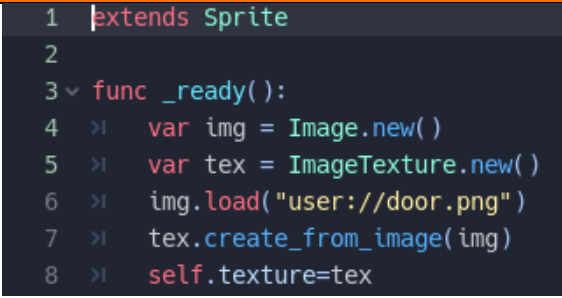



Abbildung 5 Charakterwechsel

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	<pre>func_pressed(): > load("res://ppp1.png").get_data().save_png("user://Charakter.png") > load("res://pp1.png").get_data().save_png("user://door.png") > get_tree().reload_current_scene()</pre>

Erklärung	Bild
<p>Um nun die Vorschau zu ändern wird die Gespeicherte Datei aus dem Userordner genommen.</p> <p>Darum kann die Textur ändern, weil sie immer wieder überschrieben werden kann.</p>	 <pre> 1 extends Sprite 2 3 func _ready(): 4 var img = Image.new() 5 var tex = ImageTexture.new() 6 img.load("user://door.png") 7 tex.create_from_image(img) 8 self.texture=tex </pre>
<p>Dasselbe passiert mit der Animierten Textur des Sprites Ingame. Er nimmt seine Texturen auch aus dem Userordner</p>	 <pre> 1 func _ready(): 2 var img = Image.new() 3 var tex = ImageTexture.new() 4 img.load("user://Charakter.png") 5 tex.create_from_image(img) 6 \$Sprite.texture=tex </pre>

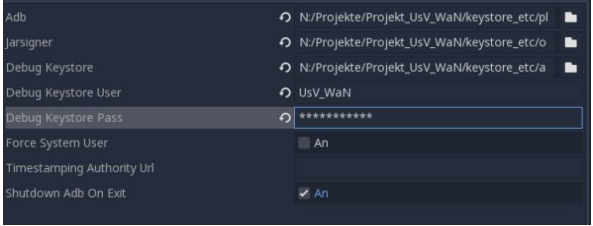
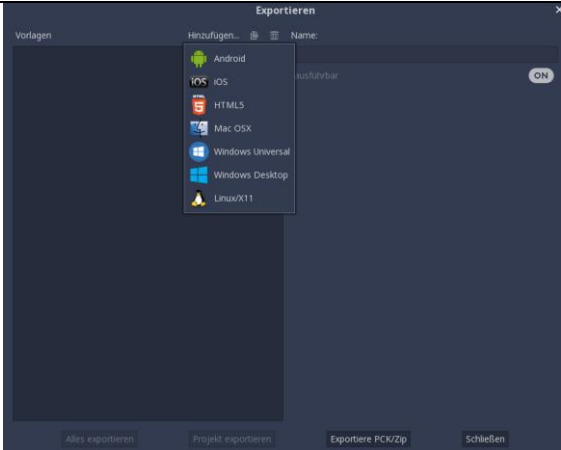
8 Veröffentlichen

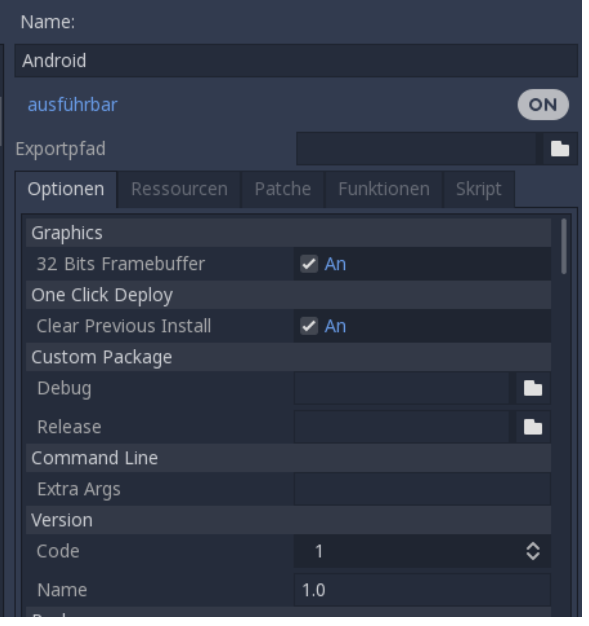
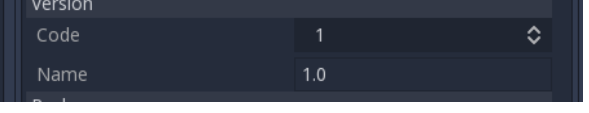
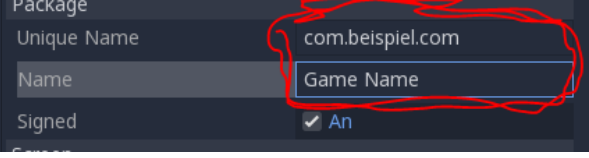
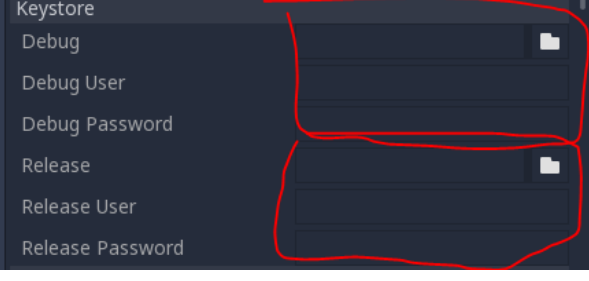
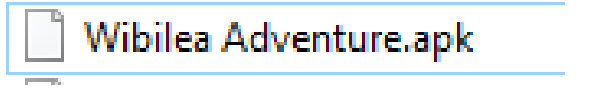
8.1 Android

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.	

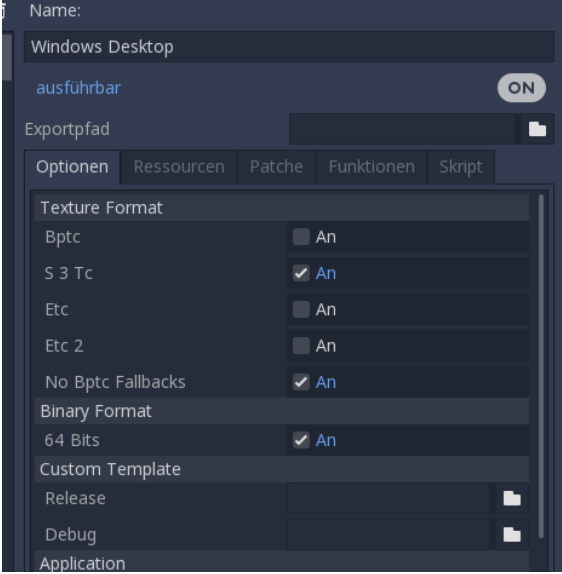

Erklärung	Bild
Nun kommt ein Fenster mit neuen Einstellungen.	
Unser Projekt benötigt eine Version	
<p>Hier müssen wir einen Unique Name geben, welcher wie folgt aussieht com.beispiel.endung.</p> <p>Zusätzlich benötigen wir noch einen Spielnamen.</p>	
Im Debug und Release muss erneut die Keystoredatei 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.	

8.2 Windows

8.2.1 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.	

8.2.2 Windows Installation

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

8.3 IOS

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

9 Arbeitsjournal

9.1 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 usw	Was ist GoDot und was kann es	keine	https://www.youtube.com/watch?v=WRDI2gQObg8&list=PL1td_Fr5vMGOW0hasVEYlvfdm_oYh0xi9
Charakter design usw	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	

9.2 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=WRDI2gQObg8&list=PL1td_Fr5vMGOW0hasVEYlvfdm_oYh0xi9
Texturen erstellen usv	Texturen erstellt für: Explosion Wand	-	keine	programm: GIMP

Tätigkeit	Erledigte Arbeiten / Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
	Boden Gras Stein			
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=6zilyx60N6I

9.3 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 usw	Tutorial 3-8	Teilmap, Scripting	-	https://www.youtube.com/watch?v=WRDI2gQObg8&list=PL1td_Fr5vMGOW0hasVEYlvfdm_oYh0xi9
Speicher Funktion	speichern ermöglicht	wie man ein spiel speichert	funktioniert nicht	https://www.youtube.com/watch?v=ML-hiNytIqE

9.4 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

Tätigkeit	Erledigte Arbeiten / Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Tische, Stühle, PCs etc. design	work in Progress	-	-	Programm: GIMP, Photoshop

9.5 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

9.6 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	Funktionierende Touch steuerung	Wie funktioniert eine Touchsteuerung	Keine	-

9.7 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	-	-	-

9.8 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	-	-	-

9.9 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	szenen wechselt ins nichts	szenen müssen in den preload	-	-

9.10 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	-

9.11 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 Tilesset 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	-	-

9.12 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	-

9.13 01.04.2020

Tätigkeit	Erreichte Arbeit/ Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Werkstatt Wibilea grundriss	der grundriss der werkstatt ist fertig	-	-	-

Tätigkeit	Erreichte Arbeit/ Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Pausenmenü wird neu geschrieben	Pausenmenü mit resume und Menubutton	Pausenmenu darf Process(delta) nicht stoppen	-	-

9.14 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	-	-

9.15 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/

9.16 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		-	-

Tätigkeit	Erreichte Arbeit/ Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
fehlerbehebung Map + bilder hinzufügen	alle gefundenen fehler wurden behoben	Map ist fertig	-	-

9.17 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	-	-

9.18 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	-

9.19 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	<ul style="list-style-type: none"> 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)	-

9.20 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	-	-	-

9.21 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	-	-

9.22 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	-	-	-

9.23 28.04.2020

Tätigkeit	Erreichte Arbeit/ Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
1.versuch Polymechnik minigame	-	-	Polygone colisionshapes können nicht verändert werden	-

Tätigkeit	Erreichte Arbeit/ Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Automation mit Start und end screen ergänzt	Die Automation hat jetzt auch einen start und end screen	-	-	-

9.24 29.04.2020

Tätigkeit	Erreichte Arbeit/ Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
2.versuch polymechnik	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	-	-

9.25 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	-	-

9.26 05.05.2020

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

9.2706.05.2020

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

9.2811.05.2020

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

9.2912.05.2020

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 Unschärf da er nicht als 2D Pixel Importiert wird	-
Kleine Bug Fixes	-	-	-	-

9.3013.05.2020

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	-	-	-	-

9.3118.05.2020

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

9.3219.05.2020

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

9.33 20.05.2020

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

10 Testen

10.1 Testfall 1

Testfallbeschreibung

ID / Bezeichnung	T-001	Download
Beschreibung	Download der APP	
Testvoraussetzung	Android 9 oder 10, Handymarke Samsung Honor oder Xiaomi.	
Testschritte	<ol style="list-style-type: none">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	

10.2 Testfall 2

Testfallbeschreibung

ID / Bezeichnung	T-002	Startup
Beschreibung	Das Spiel startet	
Testvoraussetzung	Erfolgreicher 1. Testfall	
Testschritte	<ol style="list-style-type: none">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	

10.3 Testfall 3

Testfallbeschreibung

ID / Bezeichnung	T-003	EXIT
Beschreibung	Spiel im Hauptmenu verlassen	
Testvoraussetzung	Erfolgreicher 1. Testfall	
Testschritte	<ol style="list-style-type: none"> 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	

10.4 Testfall 4

Testfallbeschreibung

ID / Bezeichnung	T-004	Walk
Beschreibung	Steuern der Spielfigur	
Testvoraussetzung	Erfolgreicher Testfall 2	
Testschritte	<ol style="list-style-type: none">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

10.5 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

10.6 Testfall 6

Testfallbeschreibung

ID / Bezeichnung	T-006	Teleport
Beschreibung	Teleportieren zu den Minigames	
Testvoraussetzung	Erfolgreicher Testfall 2	
Testschritte	<ol style="list-style-type: none"> 1. Nach dem Starten des Spiels, wird auf die Sprechblase der Figur geklickt, welche 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

11 Tabellen- und Abbildungsverzeichnis

Tabelle 1 Spieler Textur	9
Tabelle 2 Andere Texturen	9
Tabelle 3 Photoshop einrichten	12
Tabelle 4 Photoshop Tools	13
Tabelle 5 Tastenkürzel Photoshop	14
Tabelle 6 Godot Basics	14
Tabelle 7 Godot Script	15
Tabelle 8 Spiel erstellen	17
Tabelle 9 Godot Allgemeine Einstellungen	25
Tabelle 10 Steuerung	27
Tabelle 11 Pausen-menu	27
Tabelle 12 Informatik Game	29
Tabelle 13 Automation Game	32
Tabelle 14 Polymechanik Game	34
Tabelle 15 Konstruktion Game	36
Tabelle 16 Exportieren Android	42
Tabelle 17 Exportieren Windows	44
Abbildung 1 Verknüpfung der Dateien	15
Abbildung 2 Spiel Plan	17
Abbildung 3 Szenen Wechsel einfach	24
Abbildung 4 Szenen Wechsel erweitert	24
Abbildung 5 Charakterwechsel	40

12 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 Szenen 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
GL ES 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 Öffentlichen und Privaten APK Key
KinemeticBody	Node Spielerkollisionen in GoDot
Main scene	Hauptszene
Map	Karte
Mini games	Mini Spiele
movement	Bewegung

Abkürzung/Fremdwort	Erklärung
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	Geschriebene 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	Sichtbar
Vsync	Vertikale Synchronisation der Bildschirmwiederholrate
WaN	Abkürzung für Noel Wangler
ZIP	Kompriemierter Ordner

13 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
DeveloperID 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

GL ES 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

Tasteneinput	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
----------	----