

Delphi Programmierpraktikum

FloodPipe

Nima Mohammadimohammadi

Softwareentwicklung

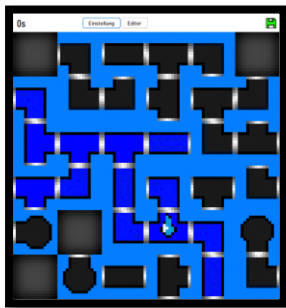
10544

Inhaltsverzeichnis

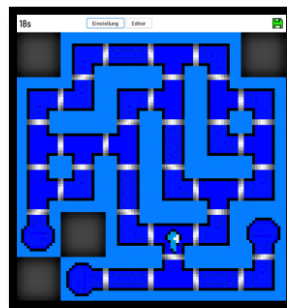
1. Allgemeine Problemstellung	3
2. Benutzerhandbuch	4
2.1. Ablaufbedingungen	
2.2. Programminstallation	
2.3. Programmstart	5
2.3.1. Verknüpfung von der Exe	
2.4. Bedienungsanleitung	
2.4.1. Hauptbildschirm	
2.4.2. Einstellung Menu	
2.4.3. Bedienung	6
2.4.4. Spielfeld	
2.4.5. Feld Platzierungen	
2.4.6. Editor Modus	7
2.4.6.1. Rohr drehen	
2.4.6.2. Rohr ändern	
2.4.6.3. Aktives Rohr	
2.4.6.4. Speicherung.....	8
2.5. Fehlermeldungen.....	
2.6. Wiederanlaufbedingungen.....	9
3. Programmierhandbuch	10
3.1. Entwicklungskonfiguration.....	
3.2. Problemanalyse und Realisation	11
3.2.1. Problemanalyse	
3.2.2. Realisation	12
3.3. Beschreibung grundlegender Datenstrukturen	23
3.4. Programmorganisationsplan	24
3.5. Programmtests	25

1 Allgemeine Problemstellung

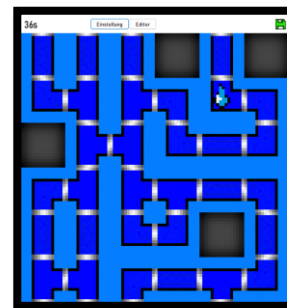
In dem Spiel **FloodPipe** wird dem Spieler ein Spielfeld präsentiert mit lauter Rohren, die man richtig aufstellen muss. Der Spieler kann durch das mehrfache Anklicken eines Rohrs es drehen, um mit dem Nachbar Rohren die Verbindung herstellen. Dabei ist es wichtig, dass am Ende alle Rohre mit Wasser gefüllt wurden und dabei kein offenes Ende entsteht.



1.1 präsentiertes Spielfeld



1.2 gelöstes Spielfeld



1.3 Überlaufmodus

Dabei kann der Spieler das Spielfeld beliebig groß einstellen, bedeutet mehr Spalten und Zeilen (max. 15x15) oder weniger (min. 2x2). Es darf auch eine nicht quadratische Anordnung sein z. B. (8x5).

Wie schon in der Abbildungen 1.1 und 1.2 werden auch unbewegbaren Objekte (Steine) auf dem Spielfeld platziert. Die Häufigkeit der Steine können auch beliebig eingestellt werden (1% bis 10%).

Der Überlaufmodus ist eine Einstellung Möglichkeit, die nach Bedarf eingeschaltet werden kann , bei den Modus handelt es sich um die direkte Verbindung von den Spielfeldränder (von oberem Rand zum unterem bzw. linkem und rechtem Rand).

Diese Einstellungen ermöglichen ein schweres oder leichtes Spielerlebnis je nach Bedarf.

Die Rohre werden von der Quelle aus aufgefüllt ,falls eine Verbindung besteht. Dabei kann auch die Animationsgeschwindigkeit beliebig eingestellt werden.

Der Spielstand kann jederzeit gespeichert werden und in einen späteren Zeitpunkt geladen werden, dabei muss das aktuelle Feld in eine Datei gespeichert werden.

Außerdem noch eine Stoppuhr, um die vergangene Zeit zu messen und dabei kann der jeweilige Spielstand geteilt werden und dabei sich mit Freunden zu messen ob und wie schnell man war.

Zusätzlich gibt es noch ein Editor , worin man Levels für FloodPipe erstellen kann.

Das Spiel FloodPipe kann von jeder Alters Gruppe gespielt werden und möchte den Spieler unterhalten und durch die Einstellung Möglichkeiten kann das Spiel sehr schwierig eingestellt werden und damit auch eine Tatsächliche Herausforderung werden. Um diese Herausforderung zu bewältigen , kann sich die Fähigkeit im Bereich räumliches denken verbessern.

2 Benutzerhandbuch

2.1 Ablaufbedingungen

Betriebssystem	Windows 32-bit
----------------	----------------

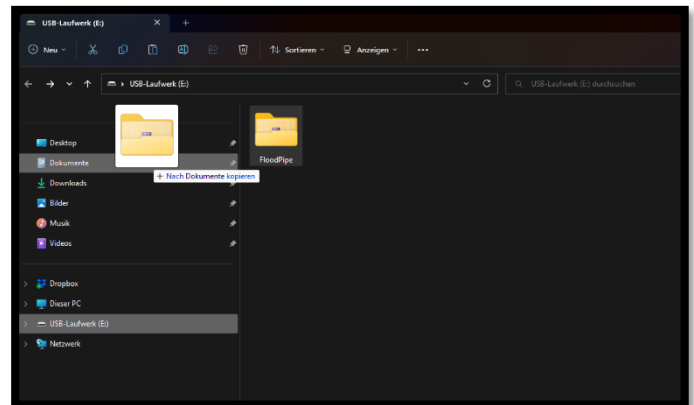
2.2 Programminstallation

Die Installation : Zuerst wird FloodPipe.zip entpackt und dann erhält man ein Ordner namens „FloodPipe“.

Danach wird der Ordner „FloodPipe“ an den gewünschten Ort verschoben .

(z. B. Desktop, Dokumente etc.)

Dann wird der jeweilige Ordner geöffnet und dementsprechend in den Unterverzeichnissen navigiert: FloodPipe -> Game



2.2.1 Geöffneter Ordner und Verschiebung in Dokumente Ordner

Hier ist der Inhalt im Game Ordner aufgelistet:

Bilder:

gespeicherten Rohr Bilder

Levels:

Da sind Spielstände hinterlegt, die mit dem Editor gemacht wurden und einerseits auch Spieltest Dateien.

config.dat:

In der Datei config.dat sind gespeicherte Einstellungen enthalten. Um die Einstellung bei jedem Start nicht neu einzustellen.

Bilder	19.01.2023 19:33	Dateiordner	
Levels	19.01.2023 19:30	Dateiordner	
config.dat	19.01.2023 19:29	DAT-Datei	1 KB
FloodPipe.exe	19.01.2023 19:14	Anwendung	16.552 KB

2.2.2 Game Ordner

2.3 Programmstart

Die FloodPipe.exe startet das Programm mit einem Doppelklick und gegeben falls kann da eine Verknüpfung erstellt werden und an einen anderen Ort verschoben werden. Damit man nicht immer wieder von neu dahin navigieren muss.

Name	Änderungsdatum	Typ	Größe
Bilder	19.01.2023 19:33	Dateiordner	
Levels	19.01.2023 19:30	Dateiordner	
config.dat	19.01.2023 19:29	DAT-Datei	1 KB
FloodPipe.exe	19.01.2023 19:14	Anwendung	16.552 KB
FloodPipe.exe - Verknüpfung	19.01.2023 20:11	Verknüpfung	2 KB

2.3.1 Verknüpfung von der Exe

2.4 Bedienungsanleitung

Neues Spiel :

Öffnet ein Separates Fenster, worauf man spielen kann (2.4.4).

Laden :

Öffnet ein Dialog, womit man im Ordnerpfad navigieren kann und um den gespeicherten Spielständen zu laden.

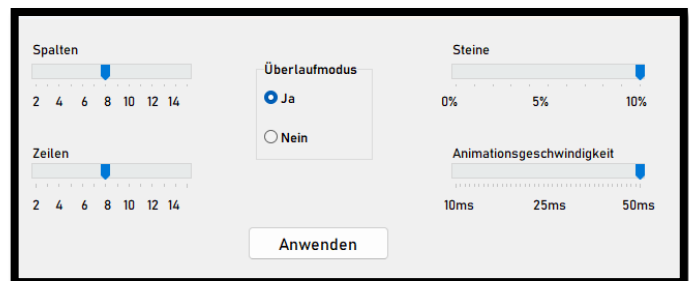
Einstellung (2.4.2) :

Öffnet ein Separates Fenster, um die Attribute vom Spiel zu ändern :

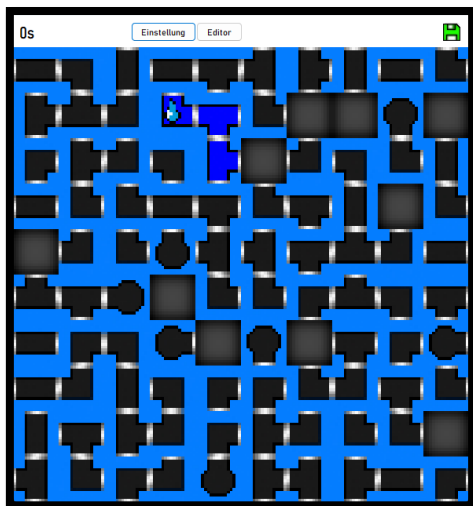
- Überlaufmodus: bei dem Modus handelt es sich um die direkte Verbindung von den Spielfeldränder (von oberem Rand zum unterem bzw. linkem und rechtem Rand).
- Spalten und Zeilen:
Um die Größe des Feldes zu verändern. Die Spaltengröße ändert die Breite des Spiels und die Zeilengröße die Höhe.
- Wahrscheinlichkeiten von den Steinen:
Je nach Wahrscheinlichkeit wird die Steine Anzahl im Feld variieren.
- Animationsgeschwindigkeit:
Dabei wird eingestellt ,wie schnell sich das Wasser von der Quelle aus verbreitet.



2.4.1 Hauptbildschirm beim öffnen



2.4.2 Einstellung Menu






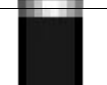
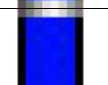

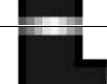
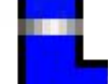

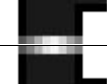



2.4.4 Präsenziertes Spielfeld

Auf der Abbildung (2.4.3) gibt es auch ein Einstellung Knopf, womit das Fenster (2.4.2) geöffnet wird und dann noch den Editor Modus , indem man eigene Levels designen kann.

2.4.3 Bedienung

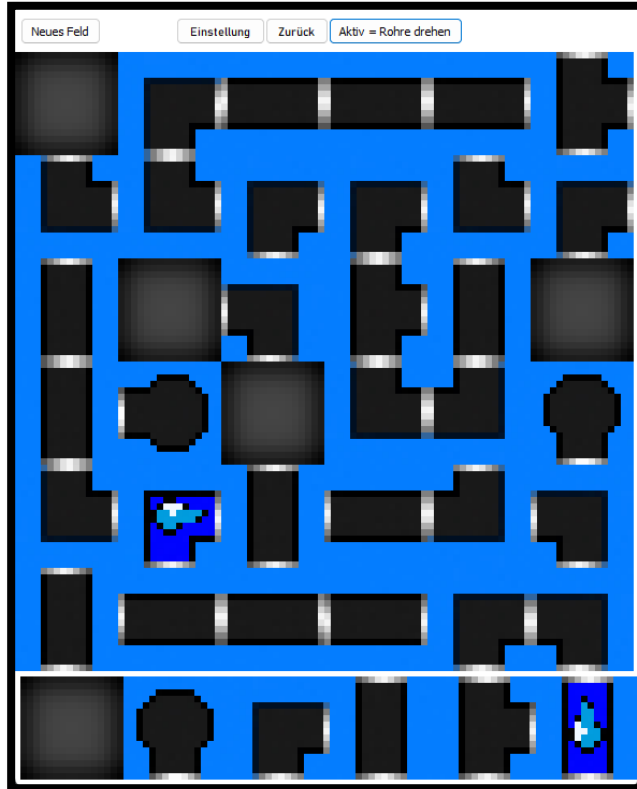
Mit der Linken Maustaste auf ein Rohr wird es 90° nach links gedreht und mit der rechten Maustaste wird das Rohr 90° nach rechts gedreht.

2.4.5 Feld Platzierungen

Leer	Aufgefüllt	Quelle	Bezeichner
			Endstück
			Gerade
			Kurve
			T-Form
			Stein

2.4.6 Editor

Nachdem man im Spielfeldmodus auf dem Editor gedrückt hat , wird das Layout sich verändern:



2.4.6.1 Rohr drehen



2.4.6.2 Rohr ersetzen



2.4.6.3 aktives Rohr auf das Quell-Rohr
geändert

Neues Feld: Ersetzt alle Felder mit Steinen

Einstellung: Öffnet das Einstellung Menu
(2.4.2)

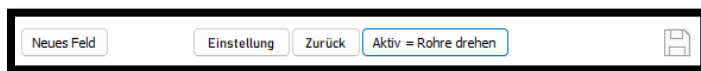
Zurück: Verlässt man den Editor Modus und
das aktuelle Feld wird verworfen und ein
neues wird generiert

Der nächste Knopf (Aktiv = Rohre drehen)
beschreibt den aktuellen Modus des Klicks
auf dem Feld.

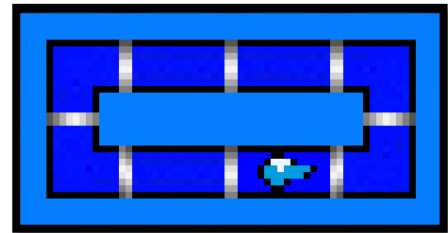
Wenn der Knopf betätigt wird , wird sich der
Modus ändern in dem „Rohr ersetzen“
Modus. Dabei erscheint ein roter Strich, der
das aktive Rohr/Stein beschreibt und mit dem
aktiven Rohr/Stein kann sich mit einem Klick
auf dem Feld das jeweilige Rohr ersetzen.

Mit einem anderen Klick auf den
nebenständigen Rohren wird das aktive
Rohr/Stein geändert (2.4.6.3).

Mit dem Quell-Rohr lässt sich die Quelle
verschieben auf ein anderes **Rohr** und die alte
Quelle wird gelöscht.



2.4.6.4 nicht speicherbar



2.4.6.5 speicherbar

Um im Editor Modus zu Speichern benötigt das Spielfeld eine Lösung , sonst kann es nicht gespeichert werden. Damit das Spielfeld eine Lösung bekommt, müssen alle Rohre mit einander Verbunden werden und somit hat man eine Lösung die dann gespeichert werden kann.

2.5 Fehlermeldung

Es gibt in dem Spiel FloodPipe nur Fehler bei Speicherung und beim Laden .

Fehlertyp	Ursachen	Behebung
Falscher Dateityp	Ungültige Dateinamen	Notation : „IrendEinName“+“.dat“ Beispiel : Feld1.dat
Öffnen	Datei nicht im Ordner Pfad	Erneuter Laden Vorgang , Neuinstallation des Spiels
Schließen	Datei nicht Ordner Pfad	Erneuter Laden Vorgang , Neuinstallation des Spiels
Lesen	Keine FloodPipe Feld Datei, Fehlerhafte Datei, unrealistische Werte	Erneuter Versuch
Schreiben	Keine FloodPipe Feld Datei, Fehlerhafte Datei, unrealistische Werte	

2.6 Wiederanlaufbedingung

Falls das Programm zur Laufzeit abgebrochen wird :

- Das aktuelle Feld sowohl Editor oder als auch Spielfeld Modus sind weg
- die „config.dat“ könnte nicht auf den neusten Stand sein
- Der Spielstand könnten nun fehlerhaft sein

Behebung :

- Falls beim nächsten Starten der Exe Fehlermeldung auftauchen sollten sie das Programm neuinstallieren.

3 Programmierhandbuch

Dieser Abschnitt informiert sie wie dieses Projekt geplant wurde , realisiert und am Ende programmiert wurde.

3.1 Entwicklungskonfiguration

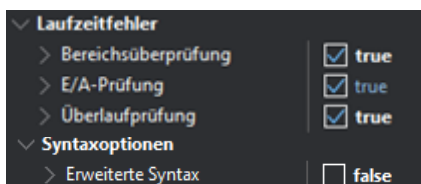
Die Entwicklungskonfiguration beschreibt welche Programme und Betriebssysteme genutzt wurden bei der Programmierung von FloodPipe.

Betriebssysteme:	Bits:	Version:
Windows 11 Pro	64	22H2
Windows 11 Home		
Windows 10 Home		

Entwicklungsoberflächen & Compiler:	Version:
Delphi Community Edition Embarcadero Technologies, Inc.	10.4.2
Delphi Community Edition Embarcadero Embarcadero Technologies, Inc.	11.1.1

Compiler Einstellung :

Projekt->Optionen->Delphi Compiler ->Compileren



Text Editor:	Version:
Visual Studio Code	1.73.3

Visual Studio Code Extension:	Version:
Pascal	v9.5.1
Delphi Extension Pack	v2.0.1

3.2 Problemanalyse und Realisation

3.2.1 Problemanalyse

1. Spielfeld erstellen
 - a. Datenstruktur initialisieren
 - b. Darstellung von Bildern
 - c. Speicherung der Bilder
 - d. Platzierung von Steinen und Rohren
 - e. Quelle hinzufügen und die Animation
 - f. Hat das Feld eine Lösung
 - g. Alle Rohre verdrehen
 - h. Feld zeichnen
2. Steuerung
 - a. Bilder drehen
 - b. 90° Rechtsdrehung
 - c. 90° Linksdrehung
3. Einstellung Möglichkeiten
4. Speicherung des Spielfelds
 - a. Dateiformat
 - b. Lesen und Schreiben
5. Editor
 - a. Auswahlmöglichkeiten für das Ersetzen darstellen
 - b. Modi für die Klicks
 - c. Neue Quelle hinzufügen und alte Quelle entfernen

3.2.2 Realisationsanalyse

1 Spielfeld erstellen

a Datenstruktur Initialisieren

Möglichen Lösungsansätzen :

Verkettete Liste:

Pro:	Contra:
<ul style="list-style-type: none">• Verkettete Listen haben eine bessere Performance als dynamischen Arrays	<ul style="list-style-type: none">• Verkettete Listen haben einen höheren Speicherbedarf, da sie interne „next“ Zeiger benötigen, um die Elemente miteinander zu verketten.• Verkettete Listen können auch schwerer zu debuggen sein, da der Code schwerer zu verfolgen ist.

Dynamisches Zweidimensionales Array

Pro:	Contra:
<ul style="list-style-type: none">• Sie verwalten Speicher besser als verkettete Listen, da sie nicht mehr Platz als nötig beanspruchen.• Kann leicht dynamisch verändert werden	<ul style="list-style-type: none">• Dynamische Arrays in der Länge zu verändern, führt zu Performenzverlust

Das Arbeiten mit dem Arrays ist viel einfacher und haben gegenüber verketteten Listen eine deutlich einfache Handhabung durch die Index Angabe . Damit der Array mehr Werte pro Index speichern soll muss noch zusätzlich ein Record deklariert werden. In dem Record sind dann die Attribute hinterlegt, die eine Zelle braucht (**→3.3 Datenstrukturen**).

b Darstellung von Bildern

Möglichen Lösungsansätzen:

Array of TImages

Pro:	Contra:
<ul style="list-style-type: none">• Man kann das zugehörige Bild leicht zuweisen• Mit OnClick Events arbeiten	<ul style="list-style-type: none">• Laufzeit mangelnde Lösung viele Objekte• Größer kleiner ziehen von Bildern ist sehr aufwendig

TDrawGrid / TStringGrid

Pro:	Contra:
<ul style="list-style-type: none"> • Einfache Bearbeitung von mit einzelnen Zellen • Größer kleiner machen von Zellen ist sehr Leicht 	<ul style="list-style-type: none"> • Laufzeit mangelnde Lösung • Delphi Lösung

TCanvas auf dem Formular

Pro:	Contra:
<ul style="list-style-type: none"> • Ist kein Objekt ,also Laufzeitschonender • Das Ersetzen von Bildern ist sehr einfach • Kann auch mit anderen Programmiersprachen umgesetzt werden 	<ul style="list-style-type: none"> • Es gibt kein Ereignis • Man muss rechnen, wohin man das Bild haben möchte • FormResize wird aufwendig

Letztendlich werden die Bilder mit TCanvas Funktionen auf dem Formular angemalt, um die Laufzeit zu schonen . Durch das zweidimensionale Array kann der Index gleichzeitig auch als Koordinate für das Bild gelten.

Beispiel :

Wie haben ein 10x10 Feld und wir wollen an der [4,5] ein Bild malen.

$X = 4 * \text{Zellbreite}$

$Y = 5 * \text{Zellhöhe}$

c Speicherung der Bilder

Möglichen Lösungsansätzen:

Imagelist

Pro:	Contra:
<ul style="list-style-type: none"> • Einfache und kompakte Speicherbedarf • Kontextsensitives Einbinden von Bildern in Anwendungen • Kann in einer Vielzahl von Formaten gespeichert werden • Verschiedene Filtermöglichkeiten, um Bildkontraste zu verändern 	<ul style="list-style-type: none"> • Keine größeren Bilder sind als 16px • Änderungen der Bilder müssen manuell vorgenommen werden • Bei einer großen Anzahl von Bildern ist es schwer das bestimmte Bild zu finden

mit Dateipfad speichern:

Pro:	Contra:
<ul style="list-style-type: none">• Das Bild kann beliebig groß sein• Änderung der Bilder sind direkt im Spiel	<ul style="list-style-type: none">• Zu viel Aufwand für ein Bild

Die Rohr- und Stein Bilder sind in eine „ImageList“ gespeichert worden und haben auch ein Index in der „ImageList“ und durch den Index kann jedes Bild von der „ImageList“ leicht erhalten werden. Dann folgt der Aufruf „GetBitmap“ und das Bild ist in der jeweilige Bitmap gespeichert und kann danach mit Canvas gezeichnet werden.

d Platzierung der Steine und Rohre

Mögliche Lösungsansätze:

Vorher Steine platzieren:

Pro:	Contra:
<ul style="list-style-type: none">• Die Steine Anzahl, die eingestellt wurde, ist immer vertreten• Leichtere Implementierung	<ul style="list-style-type: none">• Es entstehen selten „Steingruppen“<ul style="list-style-type: none">- Mehrere Steine auf einen Fleck

Nachher Steine platzieren:

Pro:	Contra:
<ul style="list-style-type: none">• Es können Mehrere Steine auf derselben Stelle platziert werden	<ul style="list-style-type: none">• Es können weniger Steine auf dem Feld sein• Die Implementierung wird aufwendiger mit

Die Steine werden vorher zufällig auf das Spielfeld platziert, um die Rohr Generierung leichter zu implementieren.

Möglichen Lösungsansätzen:

Rohre platzieren:

Von [0,0] reihenweise bis Ende

Pro:	Contra:
<ul style="list-style-type: none">• Laufzeiteffizienter• Einfache Umsetzung	<ul style="list-style-type: none">• Nicht zufällig [0,0] ist immer eine Kurve, wenn daneben keine Steine sind

Random irgendein Rohr platzieren, bis alle aufgefüllt wurden

Pro:	Contra:
<ul style="list-style-type: none">• Wirklich Random	<ul style="list-style-type: none">• mangelnde Laufzeit• große Vorarbeit

Durch die einfache Umsetzung wird das Feld von [0,0] reihenweise bis zum Ende generiert und das Argument, das es wirklich Random ist, spielt hier keine Rolle, weil das Ergebnis meist identisch ist und lieber ist das Programm schneller als das der Random Faktor größer ist.

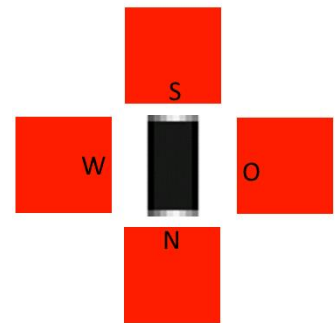
Erklärung mit Beispiel:

10x10

Erstmals fangen wir bei [0,0] an beim Feld und gehen reihenweise bis zum Ende hin also [9,9]. Dann findet eine Überprüfung statt und wenn diese Überprüfung fehlt, schlägt wird das Feld neu generiert.

Die Überprüfung von den Rohren passiert wie gefolgt :

Es wird ein Random Rohr (T-Form , Gerade, Kurve) ausgewählt und dann werden die Nachbar Rohre überprüft, ob sie eine Verbindung mit dem Rohr haben. Bedeutet die müssen die entgegen gesetzter Richtung auf True haben z. B. das überprüfte Rohr hat im Norden eine Verbindung also muss das Rohr darüber eine Verbindung Richtung Süden haben(Osten-> Westen ,Süden->Norden usw.) .Falls nicht wird das Rohr 3-mal gedreht und wenn diese nicht passt, wird das nächste Rohr ausgewählt und der Gleiche Vorgang passiert nochmal. Wenn die Rohre (T-Form , Gerade, Kurve) nicht passen wird ein Endstück platziert und das wird auch 3-mal rotiert und wird überprüft, falls das Endstück auch nicht passt, schlägt die Überprüfung fehl. Die Generierung des Feldes wird verworfen und es wird von neu aus gestartet.



Rotation :

Es ist keine echte Rotation, sondern die Richtung Attribute werden um eine Richtung weitergegeben.

Beispiel:

Norden := Westen

Osten := Norden

Süden := Osten

Westen := Süden



Random Rohr :

Es wird ein statisches Array deklariert mit der Länge drei und auf jeder Position des Arrays wird ein Random Rohr deklariert. Dann wird aufsteigend nach Index durch das Array gegangen.

e Quelle einfügen und die Animation

Mögliche Lösungsansätze für die Darstellung der Quelle:

Mit vier verschiedene Quell-Bilder:

Pro:	Contra:
<ul style="list-style-type: none">• Leichte Implementierung mit der ImageList• Jede Quelle hat ihr eigenes Design	<ul style="list-style-type: none">• Speicherintensiv

Eine Quell Bild auf andere Rohre überlappen:

Pro:	Contra:
<ul style="list-style-type: none">• Änderung einer Rohr Datei ist einfacher• Speicherschonender	<ul style="list-style-type: none">• Benötigt die Funktion, die das macht• Jede Quelle würde gleich aussehen

Es gibt vier verschiedene Quell-Bilder, die für jede Variation der Rohre entsprechen und durch eine ImageList Struktur ist die Lösung mit mehr Bilder deutlich angenehmer für die Programmierung.

Mögliche Lösungsansätze um Felder zu Speichern die angemalt werden müssen.

Speicherung in eine einfach verkettete Liste

Pro	Contra
<ul style="list-style-type: none">• Direkter Zugriff von auf X und Y Komponenten	<ul style="list-style-type: none">• schwierige Implementierung in der Rekursion

Speicherung von Rohren in String

Pro	Contra
<ul style="list-style-type: none">• Leichte Implementierung mit einer Rekursion	<ul style="list-style-type: none">• Keine Direkter Zugriff von X und Y• Verwendung von strtoint() und inttostr()

Nach der Feld Generierung wird eine Quelle zufällig platziert und dann beginnt die Animation:

Animation:

Von der Quelle aus werden alle Anbindung durchgegangen und falls eine Anbindung besteht, wird die Rekursion (fillRekursiv(...)) mit dem Rohr nochmal aufgerufen die zuvor mit der Quelle verbunden war und durch jeden neuen Aufruf der Rekursion wird das Rohr in einem String gespeichert die letztlich die Animationsreihenfolge bestimmt.

Jetzt ist die Animationsreihenfolge als String gespeichert und nun wird eine Stoppuhr gestartet, um den String von vorne nach hinten durchzugehen. Die Stoppuhr wird so lange aufgerufen, bis sie ein leerer String hat.

f Hat das generierte Feld eine Lösung?

Bei dem Test, ob das generierte Feld lösbar ist, wird eine Fake „Quelle“ hinzugefügt und die Rekursion (fillRekursiv(...)) wird gestartet danach wird gezählt wie viele Rohre im String vorhanden sind, falls weniger als erwartet drin sind, wird das Feld neugeneriert. Falls alles gut ablief, wird anschließend die „Quelle“ entfernt.

g Alle Rohre verdrehen

Um das Feld richtig für den Spieler vorzubereiten , werden mit random(4) durch alle Zellen durchgegangen und die Rotationanzahl von der Zelle werden erhöht .Um im späteren Zustand es zu malen.

h Feld zeichnen

Nach dem die Rotation jetzt zufällig ist , werden alle Zellen nach ihren Attributen angemalt und die Bilder dementsprechend auf dem Feld mit Canvas aufgemalt.

2. Steuerung

Das Feld wurde mit Canvas auf das Formular gemalt und da kein Objekt erstellt wurde hat es auch kein Event für diese Anwendung und um das Problem zu lösen, wird die „FormMouseDown“ Event verwendet, um Links- Rechtsklicks zu erkennen. In der Funktion muss außerdem noch ausgerechnet werden welches Rohr dabei angeklickt wurde und das wird mithilfe der Maus Position auf das Feld ermittelt.

a Bilder drehen

Mögliche Lösungsansätze:

Es gibt von jedes Rohr 4 Bilder mit verschiedener Rotation:

Pro:	Contra:
<ul style="list-style-type: none">• Ermöglich unterschiedliche Bilder zu verwenden	<ul style="list-style-type: none">• Man braucht die vierfache Speicherung

Funktion mit Rotation, um Bilder zu drehen:

Pro:	Contra:
<ul style="list-style-type: none">• Kompakte Speicherung	<ul style="list-style-type: none">• Einbindung von Bibliotheken

Das Programm verwendet eine Funktion die Bilder drehen kann und ist dadurch viel kompakter als die anderen Lösungsansatz . Die Funktion dreht die übergebene Bitmap um 90° nach rechts und somit ist die Steuerung der rechten Maustaste gesichert und um 90° nach links zu drehen wird die Funktion 3-mal aufgerufen ist praktisch die Links Drehung.

3. Einstellung Möglichkeiten

Mögliche Lösungsansätze:

Mit TEdits (Text Felder):

Pro:	Contra:
<ul style="list-style-type: none">• Der Benutzer kann direkte Eingaben vornehmen• Keine interaktive Lösung	<ul style="list-style-type: none">• Zusätzliche Fehlerüberprüfung auf Bereichs Überschreitungen

Mit eine Radiogroup (Häkchen Felder):

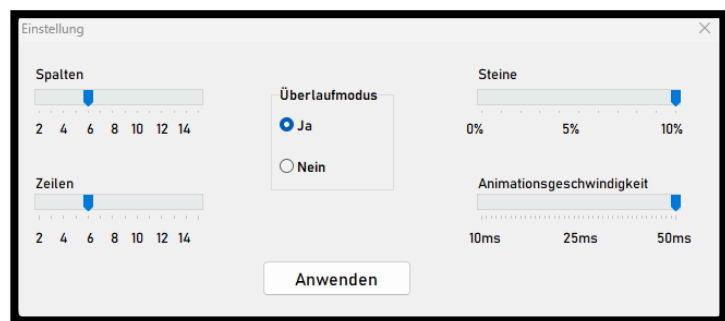
Pro:	Contra:
<ul style="list-style-type: none">• Interaktive Lösung	<ul style="list-style-type: none">• Zu viele Felder für den Benutzer• Unschönes Layout

Mit Track Bars (Regler)

Pro:	Contra:
<ul style="list-style-type: none">• Keine Bereichs Überprüfung• Interaktive Lösung	<ul style="list-style-type: none">• Beschriftung fehlt

Bei den Einstellung Möglichkeiten werden hauptsächlich TrackBars verwendet, um den Benutzer zu entlasten und damit der Benutzer eine bessere Übersicht hat , wie groß er einzelne Attribute einstellen lassen kann.

Für den Überlaufmodus wurde eine Radiogroup ausgewählt ,weil da keine Bereichs Überprüfung stattfinden soll.



4. Spielfeld Speicherung

a Datei Format

Mögliche Lösungsansätze:

Textdateien:

Pro:	Contra:
<ul style="list-style-type: none">• Textdateien ermöglichen es Benutzern, Änderungen an der Datei vorzunehmen	<ul style="list-style-type: none">• Textdateien haben eine größere Größe als Dat-Dateien, was die Speicheranforderungen erhöht.• Manipulation von Daten

Dat-Dateien:

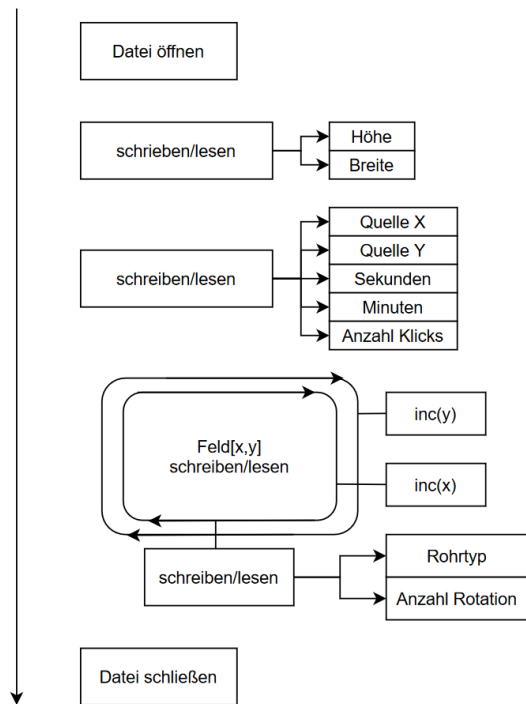
Pro:	Contra:
<ul style="list-style-type: none">• Dat-Dateien ermöglichen eine strukturierte Speicherung von Informationen.• Manipulation von Daten ist erschwert• Dat-Dateien sind leistungsfähiger, da sie mehr Informationen in einem kleineren Dateiformat speichern. Alle	<ul style="list-style-type: none">• Um Dat-Dateien zu ändern, braucht man ein extra hex Editor

Durch die effiziente Speicherung der Dat-Dateien arbeitet das Programm mit Dat-Dateien und um auch damit Spieldatei nicht manipuliert werden können.

Erklärung von den Funktionen (file2Field,field2file):

1. Erst wird die Datei geöffnet, um das Feld zu speichern/laden.
2. Dann werden die Attribute vom Feld gespeichert/gelesen.

Wenn das Feld gelesen wird, wird die Länge des Dynamische Array gesetzt.
3. Dann werden Attribute gespeichert/gelesen die wichtig für den Spielstand sind. Mit Quell Koordinaten.
4. Dann wird durch das Feld gegangen von [0,0] bis zum Ende [Breite, Höhe] dabei werden Rohrtyp des Feldes geschrieben/gelesen und mit der Anzahl der Rotation des Feldes.
5. Datei wird geschlossen.



2. Editor

a Auswahl für das Ersetzen der Rohre einfügen



Mögliche Lösungsansätze:

Array of TImage:

Pro:	Contra:
<ul style="list-style-type: none">• Man kann das zugehörige Bild zuweisen• Mit OnClick Events arbeiten	<ul style="list-style-type: none">• Laufzeit mangelnde Lösung• Größer kleiner ziehen von Bildern ist sehr aufwendig

TCanvas auf dem Formular:

Pro:	Contra:
<ul style="list-style-type: none">• Ist kein Objekt ,also Laufzeitschonender• Das Ersetzen von Bildern ist sehr einfach• Kann auch mit anderen Programmiersprachen umgesetzt werden	<ul style="list-style-type: none">• Es gibt kein Ereignis• Man muss rechnen, wohin man das Bild haben möchte• FormResize wird aufwendig

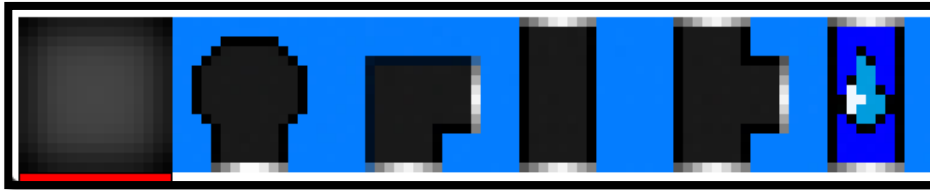
Bei der kleinen Anzahl von Bildern, verwendet das Programm ein „Array of TImage“, um ein Event zu haben, der Unabhängig von „FormMouseDown“ arbeitet. Bei den „OnClick“ Event wird „MoveShape“ aufgerufen, um das unten liegen Rechteck bei jedem Klick auf die Rohrbilder zu bewegen ,was das aktuelle Rohr beschreibt ,was ersetzt würde, wenn auf das Feld geklickt wurde.

b Modi für den Editor

Grundlengen gibt es zwei Modi:

Aktiv = Rohre drehen

Aktiv = Rohre ersetzen



Das Rohr drehen ist schon bekannt durch den Spielmodus aber beim Editor wird geschaut welche

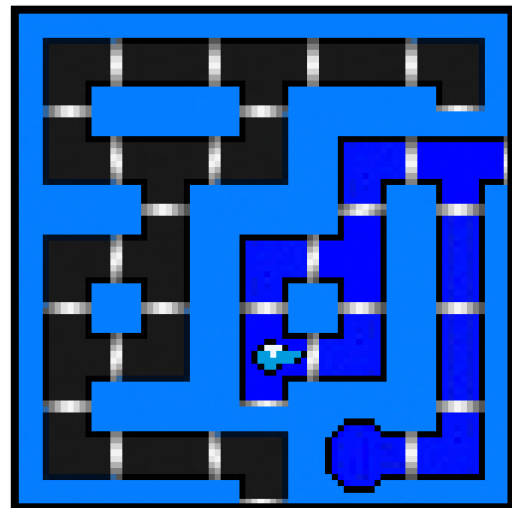
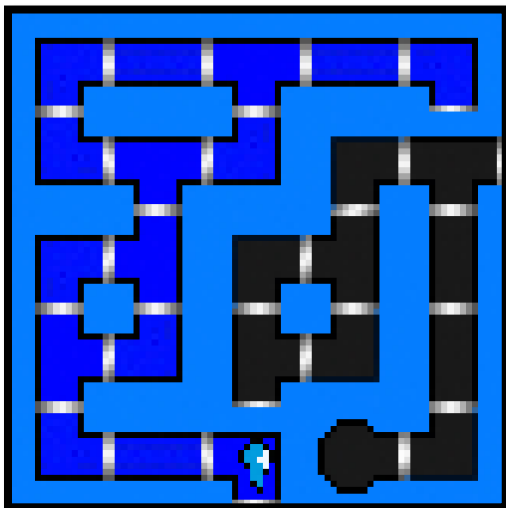
von den nun Aktiv ist. Wenn „Rohr ersetzen“ Aktiv ist wird geschaut wo das Rechteck sich befindet.

Mit einem Klick auf ein anderes Rohr bewegt sich das Rechteck mit und wenn jetzt auf dem Feld Rohre oder Steine angeklickt werden wird das Rohr ersetzt.



Nach jedem Klick des Modi-Knopf wird die Überschrift geändert auf das gegenüber und ein Attribut wird deklariert ,was den Zustand des Modus entscheidet. Dann nach jedem Klick auf das Feld wird geschaut, ob das Attribut gesetzt wurde.

c Neue Quelle hinzufügen und alte Quelle entfernen



Falls beim „Rohr ersetzen“ Modus das Quell-Rohr ausgewählt wurde ersetzt man das Rohr nicht sondern wird das angeklickte Rohr wird zu Quelle umgewandelt und die alte Quelle verschwindet.

Nach Programm Logik wird das Bild ersetzt und das angeklickte Rohr wird bekommt das jeweilige Quell-Bild und die Rekursion wird von neu gestartet.

3.3 Datenstrukturen

TIndex: Aufzählungstyp für den Array Index, um die Lesbarkeit zu verstärken und wird vom „direction“ Array verwendet.

TFieldrecord: Ein Record um eine Zelle zu präsentieren.

direction: Ein Array mit 4 Himmelsrichtungen um die Rohre nach ihre Richtungen zu platzieren.

occupied: Ein Boolean um zu überprüfen ob das Eine Zelle initialisiert wurde.

filled: Ist ein Boolean attribut was auf true gesetzt wird wenn ein Feld aufgefüllt wurde.

bitMapIdx: Ein Index der für die ImageList gedacht ist um das Bild zu malen.

rotNum : Anzahl der Rotation von dem Rohr

TField : Ein zweidimensionales dynamisches Array was ein Feld mit Zellen beschreibt. Was auch der Datentyp des Spielfeldes entspricht.

```
type
  TIndex = (N, O, S, W);

  TFieldrecord = record
    direction: array [TIndex] of boolean;
    occupied, filled: boolean;
    bitMapIdx: byte;
    rotNum: byte;
  end;

  TField = array of array of TFieldrecord;
```

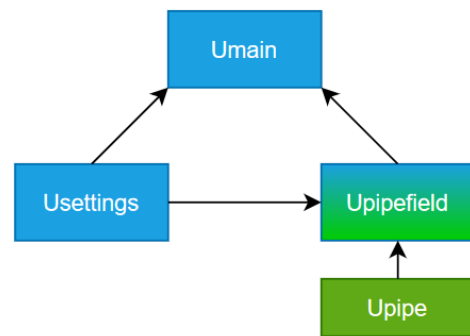

3.4 Programmorganisationsplan

Umain:

Implementation

Verwendet: Upipefield , Usettings

Umain verwendet Upipefield und Usettings, um beiden Formulare aufzurufen. Einmal wenn man die Einstellungen ändern will und wenn ein Spiel gestartet wird und geladen wird.



Grün : Logik, Blau : Forms

Upipefield:

Interface

Verwendet: Upipe, Usettings

In Upipefield ist das Formular mit dem Spiel ,worin auch die Darstellung ,Speicherung, Laden und der Editor Modus gelagert ist. Hier passiert so möglichst alles, was für den Spieler sichtbar ist und was mit Komponenten und Ereignissen arbeitet.


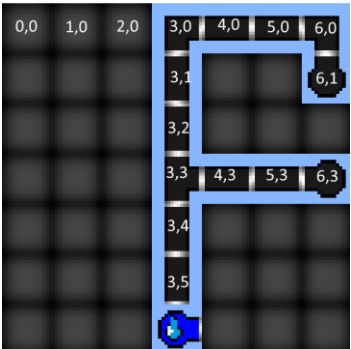

Usettings:

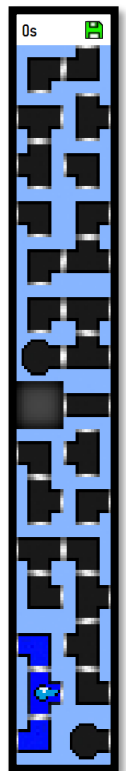
Im der Einstellung Formular werden die Einstellung Möglichkeiten aufgezählt und das Speichern der config Datei wird hier auch vorgenommen. Andere Units/Forms verwenden die Unit um die eingestellte Attribute zu erhalten.

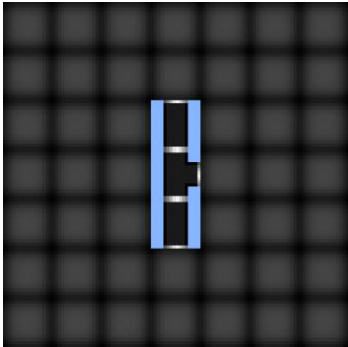
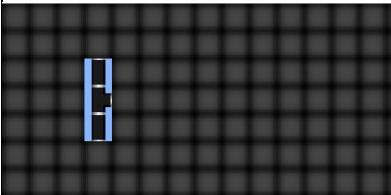
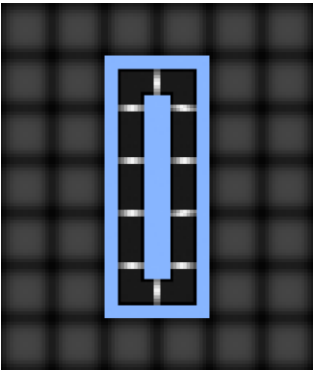
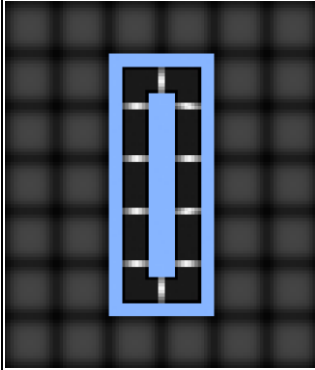

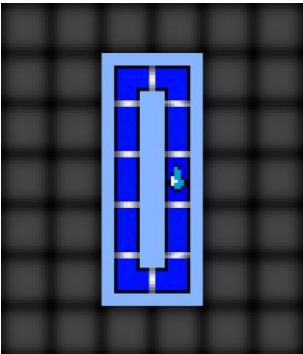
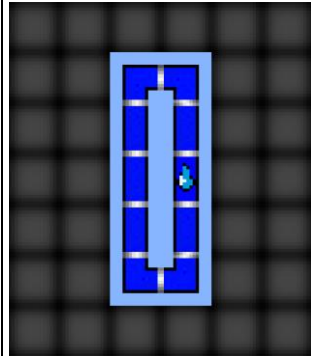

Upipe:

Upipe ist die einzige Unit im Projekt . In der Unit werden die Feld Datentypen erstellt um sie im Upipefield zu verwenden und es werden Funktionen deklariert die mit keine Objekt arbeiten und alle Funktionen die wenig bis keine Attribute von der Upipefield verwenden.

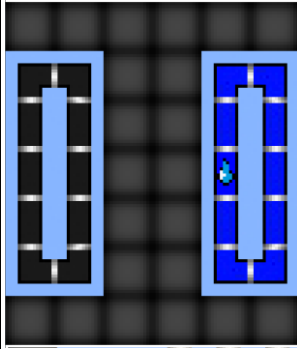
3.5 Programmtests:

Testfälle:	Erwartung:	Erzielte Lösung:
<p>Wenn ein Feld gelöst wurde und dann auf die Showmessage bestätigt, wurde sollte ein neues Feld generiert werden.</p> <p>\FloodPipe\Game \Levels\ NeueGenerierung.dat</p>	<p>Es wird ein Feld generiert ,wo alle Rohre verdreht wurden und es eine Quelle wurde auf dem Feld random hinzugefügt. Die Sekunden Anzeige ist auf 0</p>	
<p>Animation: Die Datei Animation.dat wurde geladen und die Animation wurde getestet.</p> <p>Drehung der Quelle nach links:</p>  <p>\FloodPipe\Game \Levels\ Animation.dat</p>	<p>Gleichzeitige Animation von dem T-Form[3,3]</p> <p>Animation ->3,5 -> 3,4 -> 3,3 -></p> <p>Gleichzeitig: ([3,2],[4,3])-> ([3,1],[5,3])-> ([3,0],[6,3])-></p> <p>->4,0->5,0->6,0->6,1</p>	<p>Keine Gleichzeitige Animation :</p> <p>Animation: ->3,5->3,4->3,2->3,1->3,0 ->4,0->5,0->6,0->6,1 ->4,3 ->5,3->6,3</p> <p>Die Animation wird bei mehr zwerigige nach der Reinforme durchgeführt:</p> <p>Norden->Osten->Süden->Westen</p> <p>Erst wird er Norden komplett durchgeführt dann der Osten usw.</p>
<p>Spielfeld wird Größe (2x15) wurde im Hauptmenu eingestellt und ein neues Spiel wurde erstellt.</p> <p>\FloodPipe\Game \Levels\ 15x2.dat</p>	<p>Alle Sichtbaren Komponenten sollen sichtbar sein.</p>	<p>Die Buttons für die Einstellung und Editor sind nicht sichtbar und die Zeitangabe könnte nach zweistellige Minuten Anzahl könnte die Darstellung darunter leiden. →</p>
<p>Spielfeld wird Größe (15x2) wurde im Hauptmenu eingestellt und ein neues Spiel wurde erstellt.</p> <p>\FloodPipe\Game \Levels\ 2x15.dat</p>	<p>Alle Sichtbaren Komponenten sollen sichtbar sein</p>	<p>Alle Sichtbaren Komponenten sind sichtbar.</p> 
<p>Wenn im „Game“ Ordner die Datei config.dat gelöscht wird.</p>	<p>Wird beim nächsten Aufruf von der Einstellung ,wird dann die config.dat wieder erstellt.</p>	<p>Die Datei wird erstellt beim nächsten Aufruf von der Einstellung Form</p>



<p>Speicherung: Als Name anderer Datei Format Eingabe. Beispiel: Hallo.txt</p>	<p>Abbruch, erneuter Aufruf des saveDialog</p>	<p>Abbruch, erneuter Aufruf des saveDialog</p>
<p>Beim Editor Größe des Spielfeldes vergrößern. Spalten von 7 zu 15.</p> 	<p>Das Feld wird mit Steinen bis 15 Spalten voll aufgefüllt aber das alte Feld bleibt bestehen.</p>	<p>Das Feld wird mit Steinen bis 15 Spalten voll aufgefüllt aber das alte Feld bleibt bestehen.</p> 
<p>Editor: Quelle auf Steine platzieren</p>	<p>Sollte nicht möglich sein</p>	<p>Es passiert nichts, also nicht möglich</p>
<p>Editor: Ohne Quelle abspeichern</p> 	<p>Sollte nicht möglich sein, Der Knopf wird gar nicht erscheinen.</p>	<p>Der Knopf ist deaktiviert ohne Quelle</p>  
<p>Editor: Mit Quelle abspeichern</p> 	<p>Ist möglich und der Knopf wird erscheinen.</p>	  <p>FloodPipe\Game\Levels\ SteineUmzingelt.dat</p>

Editor: Quelle verschieben



Die Alte Quelle wird gelöscht und die neue Quelle wird verschoben und Animation fängt von vorne an.

