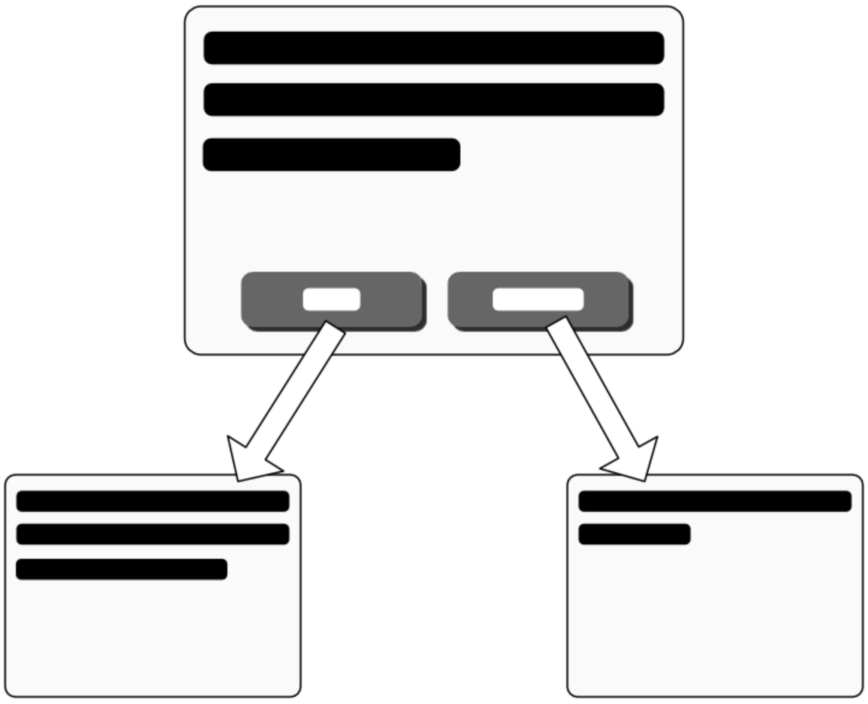
Ecumenical High School
in Bremen
Oberneulander Landstr. 143a
28355 Bremen

Development of an authoring for interactive fiction

computer science, German



Author: Fabian Keßler
teachers: Christian Panse, Katja Eggert

Submitted on: 02/07/2022

1. Introduction

1.1 What is interactive fiction?

Interactive fiction (better known under the English term "interactive fiction") describes a text-based computer game genre in which the player can influence the course of the story through his actions.[1] The best-known representatives of this genre are text adventures, such as “Adventure”, published in 1976. This was the first published game of the genre and already built on the same game mechanics as its successors: A story is told with the help of simple text, in which the player can intervene in the events through written commands. The game also gave its name to the newly created “Adventure” category through its exploratory aspects.[2] However, not all interactive fiction is adventure games. In contrast to the purely text-based text adventures, the Japanese “visual novels”, which are still popular today, for example, do not have any elements of exploration. In addition, these visual novels tell their stories in addition to the text with drawn still images. The interaction with the story does not take place via text input, but via mouse clicks.[3]

Based on the two subgenre examples, text adventures and visual novels, it is easy to see that “interactive fiction” is a broader generic term. Both the nature of the narrative and the nature of the interaction differ greatly between the subgenres.

[1] Cf. Keller, Bernd. “What does interactive fiction mean?”. interactive-fiction.de.
[2] See Wikipedia. "Adventure". en.wikipedia.org/wiki/Adventure.
[3] See Wikipedia. "Visual Novels". en.wikipedia.org/wiki/Visual_Novel.

1.2 What are authoring systems?

Authoring systems are programs used to develop interactive content. The focus is often on the effortless operation of the user without any previous knowledge such as programming.[4] A well-known example would be Microsoft's PowerPoint presentation software. The program offers many opportunities to prepare the presentation content audiovisually and to design slides interactively as well as to define a clear process.[5]

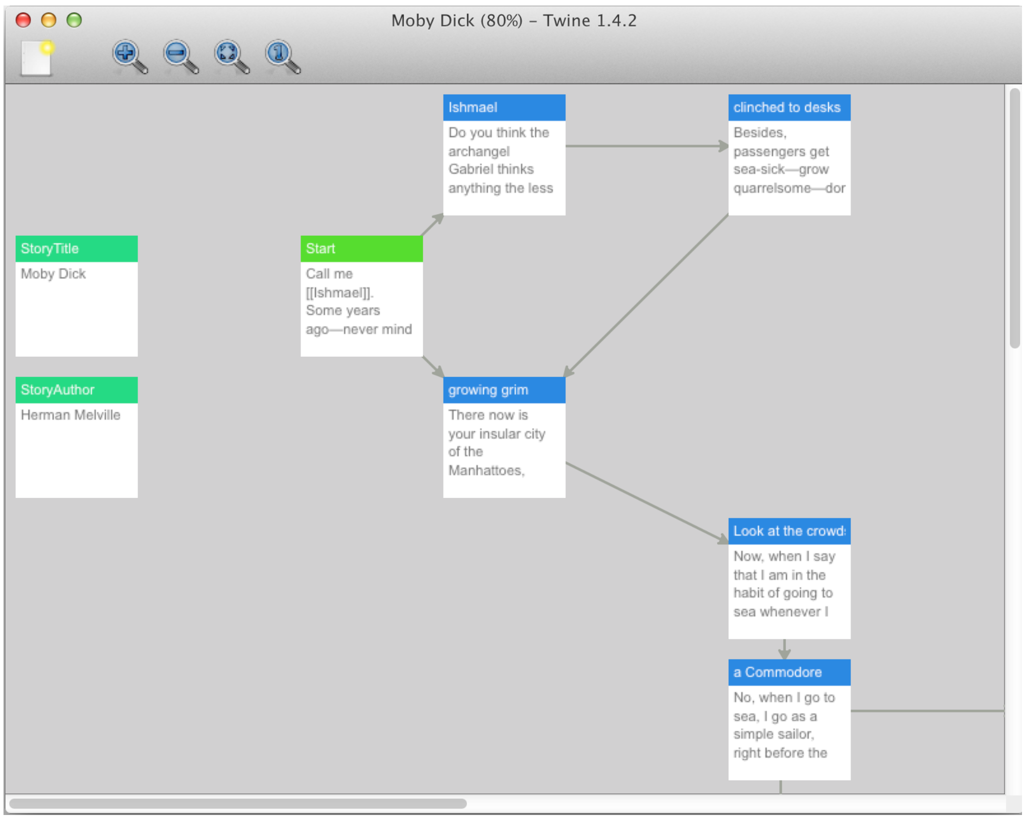
Authoring systems are also often used in the field of e-learning for the development of interactive courses. The teachers design digital learning modules, which are processed by the students or trainees.[6]

But authoring systems are also being developed for interactive fiction. Initially, computer game companies such as Infocom and Telarium only developed the systems for internal use by employed authors. This allowed the focus to be on the game design and the authors did not need any previous knowledge of programming. Finally, in 1983, the first publicly available authoring system, The Quill, was released. It already enabled the development of a game world using a graphical interface.[7]

Today, commercial text-based games are no longer being developed alongside visual novels.[8] In addition to many rather user-unfriendly scripting languages and program libraries for existing programming languages, there are currently only three well-known authoring systems: "Twine", "ADRIFT" and "Quest". All three authoring systems are offered free of charge and rely on a visual user interface designed to make it as easy as possible to develop interactive stories. However, the customization options are rather small. Although all three systems seem to offer functions

[4] See Wikibooks. "Multimedia at a glance/ design/ content/ authoring systems". de.wikibooks.org/wiki/Multimedia_im_%C3%9Cberblick/_Gestaltung/_Contents/ _Author systems.
[5] See Microsoft Office Support. "What is PowerPoint?". support.microsoft.com/de-de/office/was-ist-powerpoint-5f9cc860-d199-4d85-ad1b-4b7401 8acf5b.
[6] See IMAGE. “Authoring systems – the right one for every need?”. bild.de/infos/e-learning/e-learning/autorensysteme-10752772.bild.html.
[7] See Wikipedia. "Interactive Fiction". de.wikipedia.org/wiki/Interactive_Fiction.
[8] See Keller, Bernd. “Are there any new games being developed?”. interactive-fiction.de.

such as image display and sound reproduction, the type of image presentation, animations or mode of interaction are not configurable.[9]



Ishmael
Do you think the archangel Gabriel thinks anything the less

clinched to desks
Besides, passengers get sea-sick—grow quarrelsome—dor

StoryTitle
Moby Dick

Start
Call me [[Ishmael]]. Some years ago—never mind

StoryAuthor
Herman Melville

growing grim
There now is your insular city of the Manhattoes,

Look at the crowd
Now, when I say that I am in the habit of going to sea whenever I

a Commodore
No, when I go to sea, I go as a simple sailor, right before the

Figure 1: "Editing a Story in Twine 1.4" - twinery.org

1.3 The aim of the work

The authoring systems "Twine", "ADRIFT" and "Quest" mentioned in the previous chapter all offer good possibilities for structuring interactive stories, but they have great Deficiencies in the creative freedom of presentation and interaction.[9]

The aim of the work is therefore to develop an authoring system that allows creative freedom in all essential aspects of an interactive story through its own scripts, extensions ("plugins") and designs ("themes"). In order to keep the development of these aspects as simple as possible, an attempt is made to create a development environment that provides tools for developing

[9] See Ephesus Software. “3 tools to create your own text adventure games”. ephesossoftware.com/articles/gaming/3-tools-to-create-your-own-text-adventure-games.html.

scripts, designs, extensions and story structure. All aspects should be able to be developed, exported and imported separately from each other in order to make the adaptation for the final product of an interactive fiction as free as possible. For example, one person can only design the story and its course and use the designs and extensions of other users to adapt the visual and interactive elements.

With such freedom, not only would the development of simple text adventures like “Twine”, “ADRIFT” and “Quest” be possible, but also that of any other kind of interactive fiction (such as visual novels).

2 Choosing the technologies to be

2.1 Tech stacks When

planning the development of software, the choice of the technologies to be used is an essential part. A so-called "tech stack" is often created when a software project such as an authoring system is to be able to process many different types of processes. This indicates which technologies (such as programming languages, data storage technologies or program libraries) a project requires.[10]

In order to be able to decide on a tech stack for the development of the authoring system, it must first be clarified which skills the project requires. To do this, the target software is first divided into small software modules that are only responsible for solving certain problems.

2.2 External modules

I started with the user (here the author), who will only interact with three separate modules from the outside. Mainly, with the help of a program, he must be able to organize his resources such as text, images or sound and to bring them into a format that can be read by another software module. This software module would describe the entire authoring system.

In addition, it is customary to provide the users of a software module with instructions. This so-called "documentation" has the task of explaining the use of the software to the user as comprehensively and comprehensively as possible. Such documentation would also be necessary not only to bring potential authors closer to the authoring system, but above all to inform theme and plugin developers about usable interfaces that they can use to address certain modules.[11]

The third module describes the application with which the players of the interactive fiction ultimately interact. It represents the computer program or mobile application (mobile phone game) in which the stories exported and published by the authors are actually “played”. Additionally, this module can also be used by theme and plugin developers as well as authors during the development process to see their creations in action and to find possible errors ("debugging"). This module is the

[10] See Teufel, Yvonne. mix panel. "What is a technology stack?". https://mixpanel.com/en/blog/what-is-a-technology-stack/.
[11] See Augsten, Stephan. Dev insider. “What is software documentation?”. https://www.dev-insider.de/was-ist-eine-softwaredokumentation-a-744924/.

so-called "game engine", ie the program that ultimately brings all the resources provided by the authoring system into the form of an executable game.

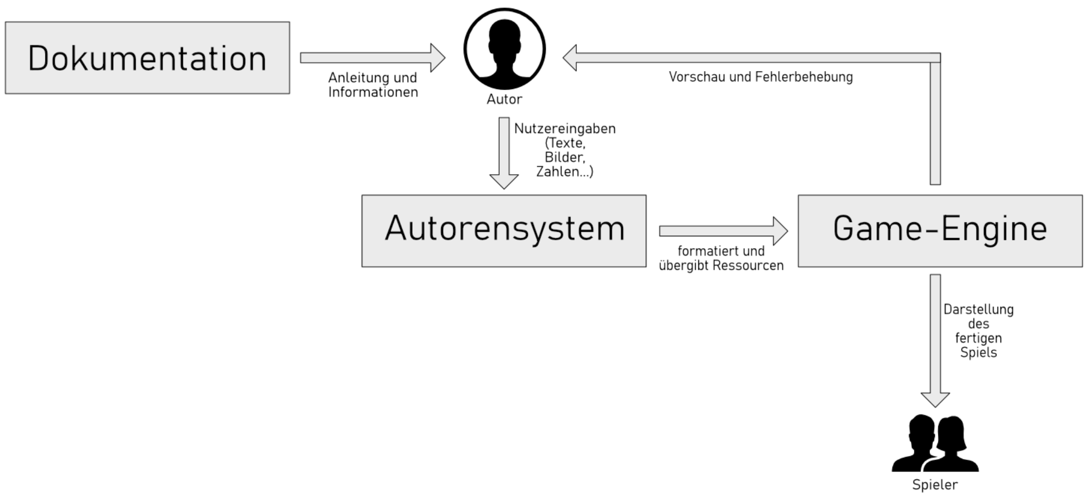

Dokumentation
Anleitung und Informationen
Autor
Vorschau und Fehlerbehebung
Nutzereingaben (Texte, Bilder, Zahlen...)
Autorensystem
formatiert und übergibt Ressourcen
Game-Engine
Darstellung des fertigen Spiels
Spieler

Figure 2: Schematic representation of the three outer modules

2.3 Inner modules

The inner modules describe those that are part of the three outer modules and thus do not interact directly with the author or with the players.

In addition to the comprehensive graphic interface for input by the authors, the authoring system includes a module for converting the project data into a format that can be read by the game engine.

The game engine provides a graphical user interface module for selecting the works to be played and one in the form of a so-called “API” (Application Programming Interface). This is a "programming interface" that allows plugins and themes to communicate with the game engine via program code.[12] However, the essential part of the engine is the interpretation module. Each project file to be executed is first read out and processed before it can be displayed as a game. The data formatted by the authoring system is retrieved in a structured manner and converted into interactive elements bit by bit.

[12] See Talend. “What is an API? Easily explained!". https://www.talend.com/en/resources/what-is-an-api/.

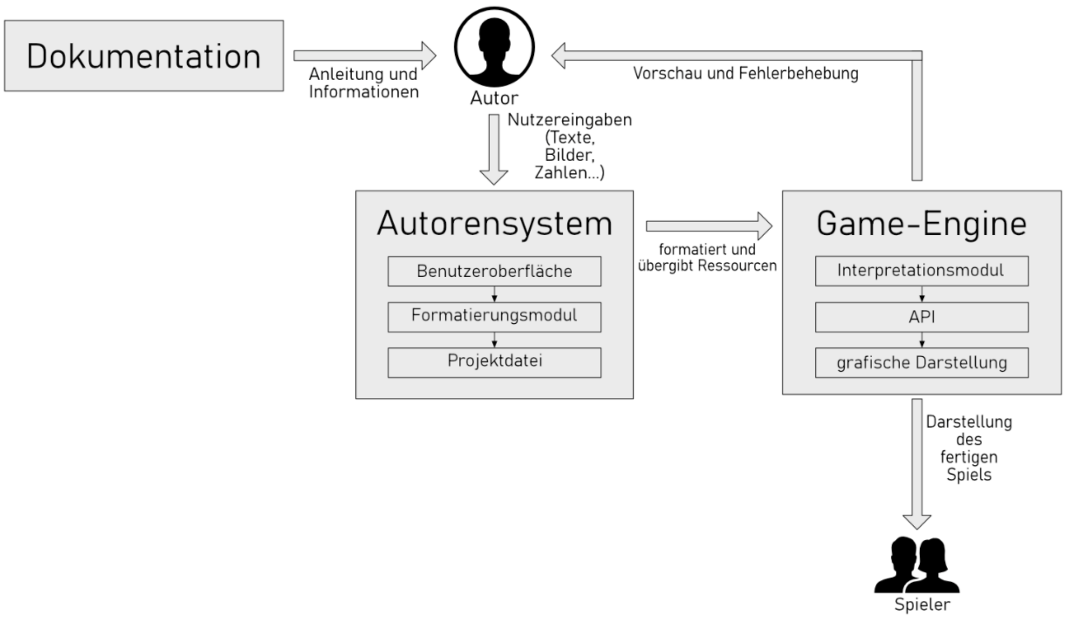

Dokumentation
Anleitung und Informationen
Autor
Vorschau und Fehlerbehebung
Nutzereingaben (Texte, Bilder, Zahlen...)
Autorensystem
Benutzeroberfläche
Formatierungsmodul
Projektdatei
formatiert und übergibt Ressourcen
Game-Engine
Interpretationsmodul
API
grafische Darstellung
Darstellung des fertigen Spiels
Spieler

Figure 3: Schematic representation of the outer and inner modules

2.4 Selected technologies

The term "game engine" is usually understood to mean software that supports the development of games by performing mostly complicated and system-related operations such as rendering[13] images, playing Sounds or the calculation of game physics[14] are anticipated or made easily accessible to the developer through certain program interfaces.[15] However, the game engine developed in this work is only able to interpret the project files of the authoring system. The final display is done by another game engine that was not developed in-house. This game engine had to be able to import various types of resources at runtime and implement them in the game. In addition, it should also be as developer-friendly as possible in order to make the theme and plugin creation as pleasant as possible. Ultimately, the open source[16] game engine "Godot" was used in this work. Above all, the branched scene structure is of great help for the development of plugins and themes (see Figure 4).

[13] Generating an image from raw data
[14] See Wikipedia. "Game physics". https://en.wikipedia.org/wiki/Game_physics.
[15] See Funes, Antonio. PCGames. "Game Engines: What is an Engine?". https://www.pcgames.de/Unreal-Engine-Software-239301/Specials/spiele-games-grafik-3d-historie-allgemein-epic-vorschau-1350751/2/.
[16] Software with freely accessible source code

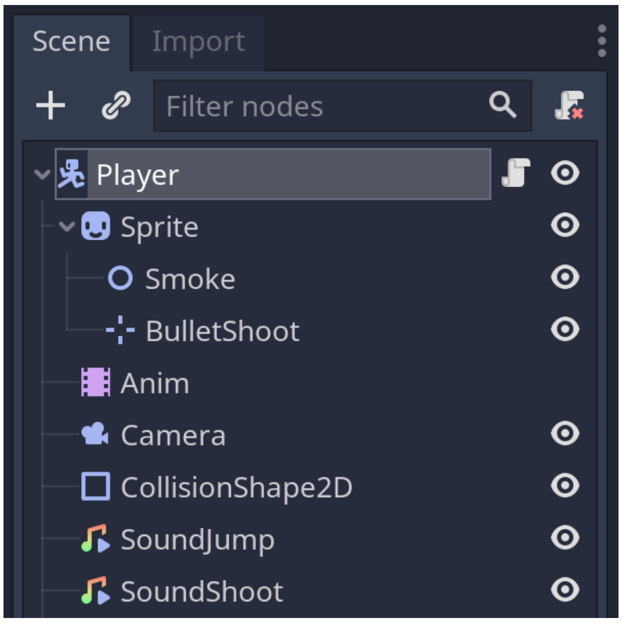
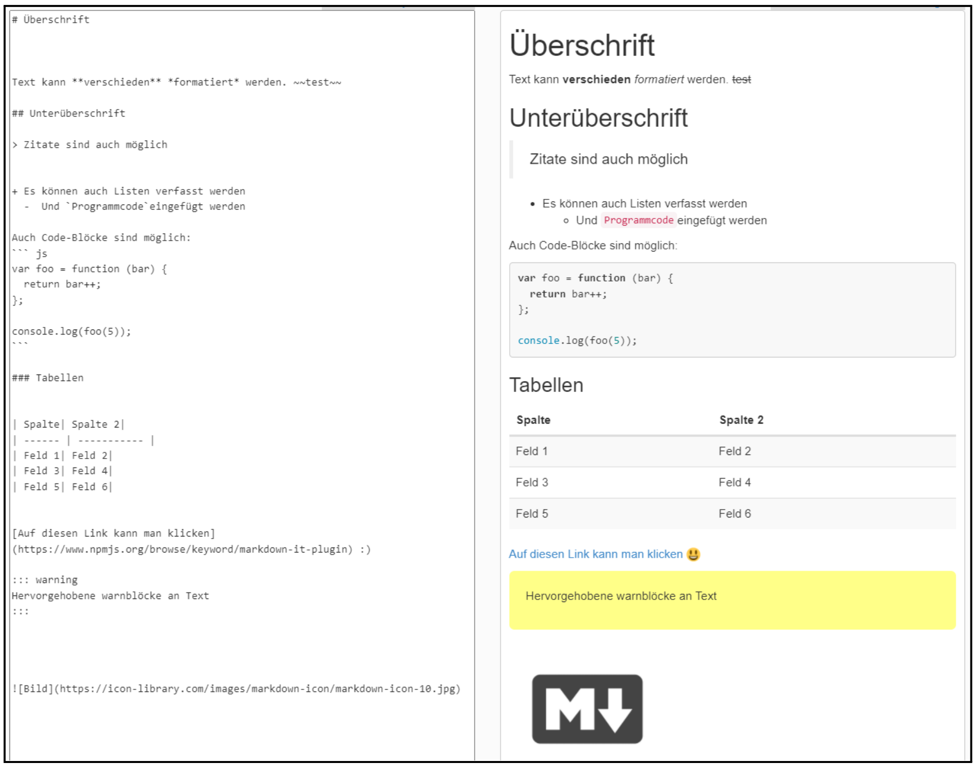


Figure 4: "Big or small ideas adapt seamlessly to Godot's node-based architecture, making your life easier." - godotengine.org

Completely different criteria had to be met for the technology of the authoring system. It should make the creation of the extensive user interface as easy as possible and offer the possibility to process relatively large amounts of data efficiently. It was also important to be able to perform extensive file system operations to create the project files. The programming language "C#" in combination with "Windows Forms" was most familiar to me. Although the latter had the disadvantage of not being able to run on operating systems other than Windows, it offered me a quick and straightforward way to design the user interface of the authoring system.[17]

I also decided on two very helpful technologies for the documentation. In order to be able to offer the documentation as platform-independent and partially automated as possible, I decided to use "Vuepress", which was developed with a focus on software documentation. It allows you to write websites using the easy-to-use[18] markdown language.[19] Markdown is particularly helpful when writing documentation, since a text written in Markdown is still legible in its source code version and the formatting is also recognizable.

[17] See Adegeo. Microsoft Docs. "Desktop Guide (Windows Forms .NET)". https://docs.microsoft.com/de-de/dotnet/desktop/winforms/overview/?view=netdesktop-6.0.
[18] machine-readable language for structuring and formatting text and other data
[19] See Vuepress Guide. "How It Works". https://vuepress.vuejs.org/guide/#how-it-works.



~~~
# Überschrift



Text kann **verschieden** *formatiert* werden. ~~test~~

## Unterüberschrift

> Zitate sind auch möglich


+ Es können auch Listen verfasst werden
  -  Und `Programmcode`eingefügt werden

Auch Code-Blöcke sind möglich:
``` js
var foo = function (bar) {
  return bar++;
};

console.log(foo(5));
```

### Tabellen


| Spalte| Spalte 2|
| ------ | ----------- |
| Feld 1| Feld 2|
| Feld 3| Feld 4|
| Feld 5| Feld 6|


[Auf diesen Link kann man klicken]
(https://www.npmjs.org/browse/keyword/markdown-it-plugin) :)

::: warning
Hervorgehobene warnblöcke an Text
:::




![Bild](https://icon-library.com/images/markdown-icon/markdown-icon-10.jpg)
~~~

Überschrift

Text kann **verschieden** *formatiert* werden. ~~test~~

Unterüberschrift

> Zitate sind auch möglich

- Es können auch Listen verfasst werden
  - Und `Programmcode` eingefügt werden

Auch Code-Blöcke sind möglich:

```
var foo = function (bar) {
  return bar++;
};

console.log(foo(5));
```

Tabellen

Spalte	Spalte 2
Feld 1	Feld 2
Feld 3	Feld 4
Feld 5	Feld 6

Auf diesen Link kann man klicken 😃

Hervorgehobene warnblöcke an Text

Figure 5: Markdown example specially created on the markdown-it.github.io page, the source code can be read on the left, the result on the right

Another advantage of Markdown is that the second technology chosen for the documentation, the also open-source "GDScript-Docs-Maker" can export its results to Markdown.[20] As the name suggests, this tool can be used to automatically generate documentation. However, this part of the documentation then only consists of a so-called "code reference", an overview of all classes and methods generated automatically with the information and comments from the code.[21] This will be especially important for all plugin and theme developers. The second part, i.e. the formulated instructions for the authoring system and the other articles, must be written manually in Markdown.

[20] See Lovato, Nathan. GitHub. "GDScript Docs Maker". https://github.com/GDQuest/gdscript-docs-maker.

[21] See Microsoft Docs. "Reference documentation" https://docs.microsoft.com/en-us/style-guide/developer-content/reference-documentation.

3 Development

3.1 Procedure

I started with the design of the authoring system, which I called "MysteryMaker". An input option had to be created for each of the possible data fields in a project file. So I developed a tree view that gives a good overview of the game elements and allows to add or remove new game elements with a right click (see fig. 6). The respective entry can also be edited by selecting entries.

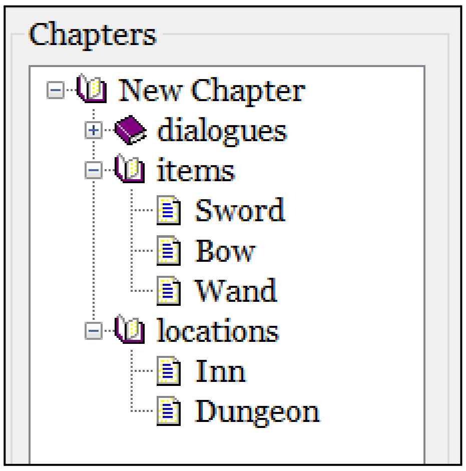
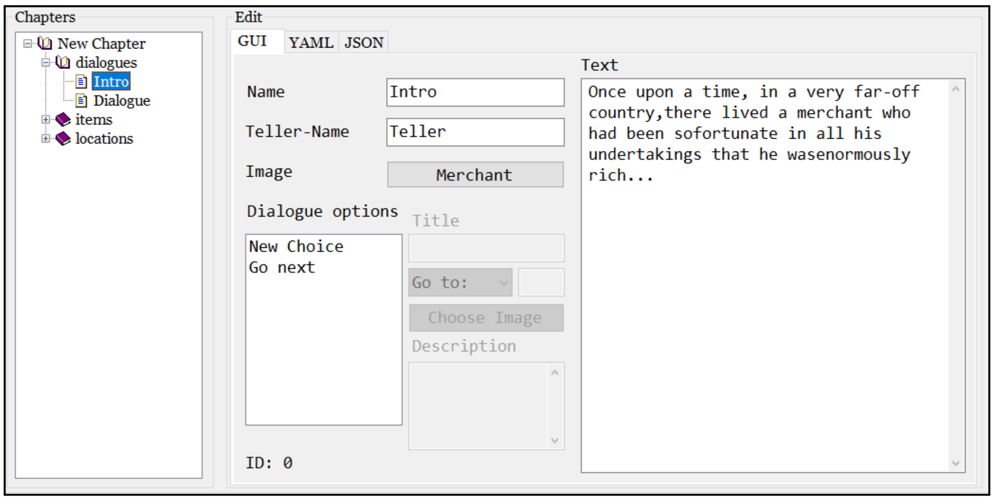


Figure 6: Tree-shaped chapter view in MysteryMaker So

-called "GUI panels"[22] had to be built for each type of entry (dialogue, object or location). These are groupings of interface elements that can be shown and hidden together. A suitable input element was chosen for each of the properties of a game object. The user of the program now always gets a suitable input interface for the element currently being edited.

[22] GUI: Graphical User Interface

Example of using "GUI panels":



- New Chapter
  - dialogues
    - Intro
    - Dialogue
  - items
  - locations

Name: Intro

Teller-Name: Teller

Image: Merchant

Dialogue options:
New Choice
Go next

Title

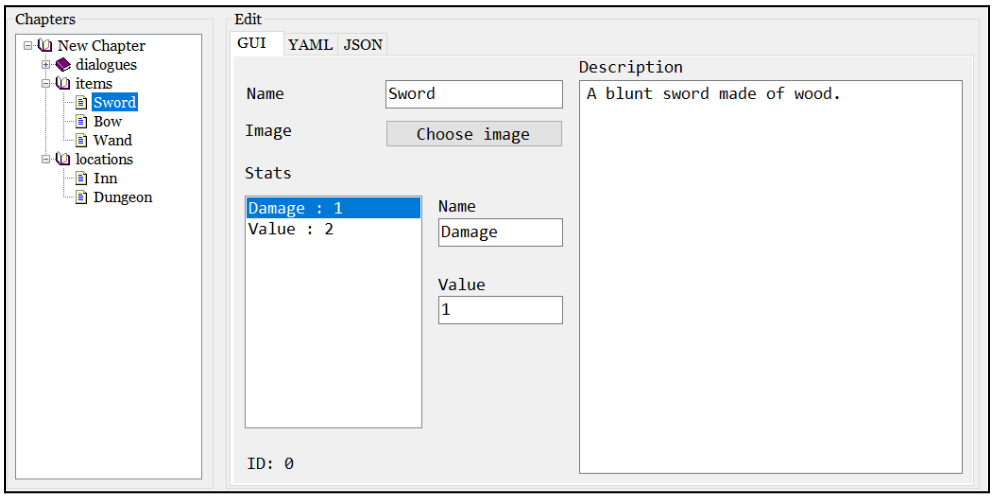
Go to:

Description

ID: 0

Text:
Once upon a time, in a very far-off country,there lived a merchant who had been sofortunate in all his undertakings that he wasenormously rich...

Figure 7: Selecting a dialog element leads to the display of the interface elements that help to enter dialogs.



- New Chapter
  - dialogues
  - items
    - Sword
    - Bow
    - Wand
  - locations
    - Inn
    - Dungeon

Name: Sword

Image

Stats:
Damage : 1
Value : 2

Name: Damage

Value: 1

ID: 0

Description:
A blunt sword made of wood.

Figure 8: Selecting an item element leads to the display of the interface elements that assist in entering items.

In order to enable the user to save his input, I then implemented a project management system. With its help, project folders could be newly created, saved and loaded.

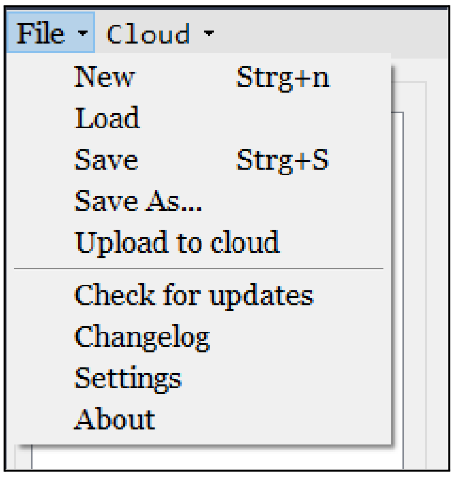


Figure 9: MysteryMaker menu bar

After MysteryMaker projects could be exported, development of the game engine to interpret them began. This engine was named "Velius Engine" and was developed within the Godot game engine. In addition to the interpretation, an important part of the Velius Engine is the management of the imported projects and player-friendly navigation.

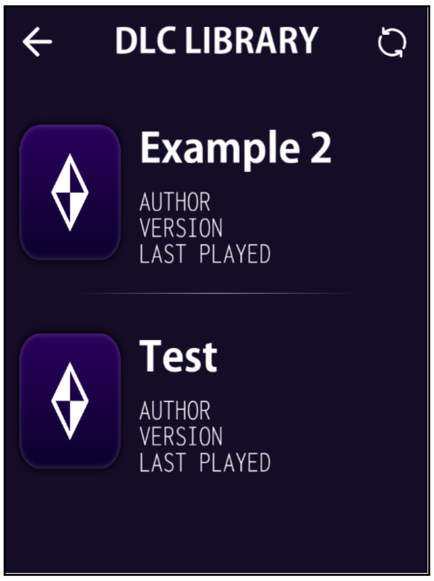
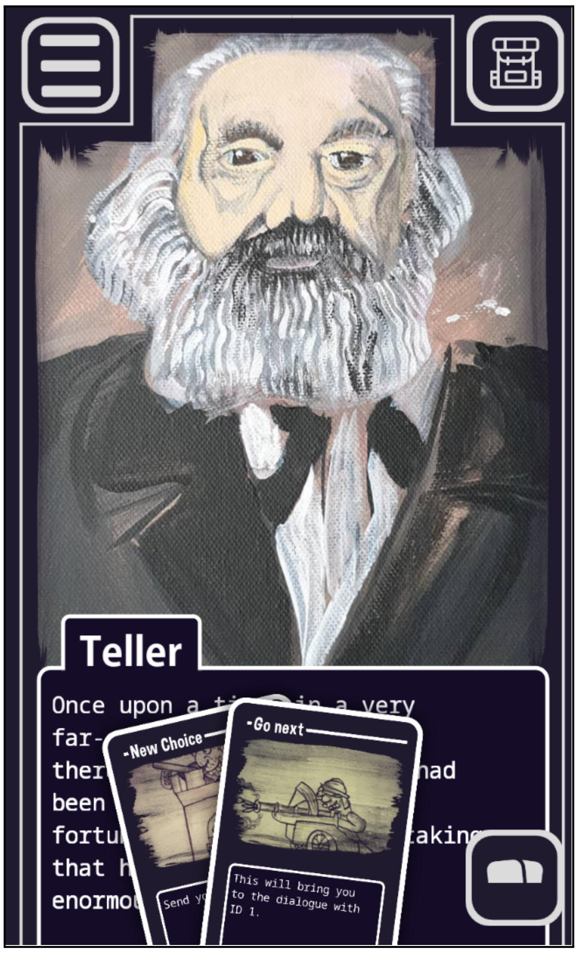


Figure 10: Project selection menu of the Velius engine

In order to be able to offer the smoothest possible work routine, minimalist "debugging tools", i.e. tools for troubleshooting such as a "Play" button and an output console, were then implemented in MysteryMaker. The open project can now be run directly from the Velius Engine at the push of a button and errors and any value outputs can be viewed via the console. This saves a lot of time and work both in the development of the interactive fictions, themes and plugins, as well as in the further development of the Velius Engine and the MysteryMaker itself.

Since a "theme" is always required by the Velius Engine to display the finished game, one more to be developed. I opted for a rather comprehensive presentation in order to emphasize the great freedom in developing themes. A more minimalistic theme, which can be used as a template for user-made themes, is also planned.



Once upon a ti in a very
far-
ther had
been
fortu aking
that h
enormo

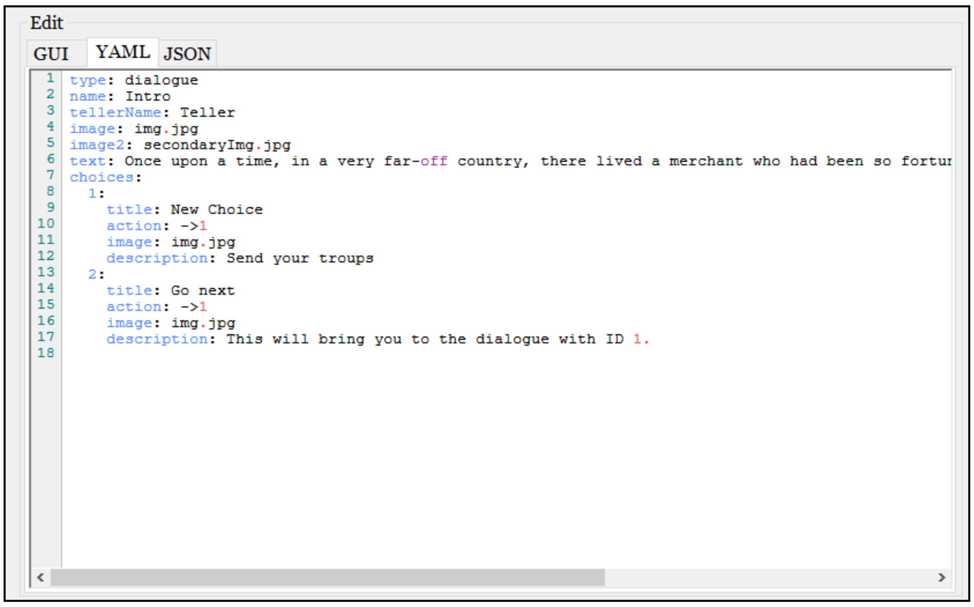
Send yo

This will bring you to the dialogue with ID 1.

Figure 11: The "Card Theme" I created in action, pictures by Milan Ilic

3.2 Problems

During the development of the GUI for the MysteryMaker, I noticed a fundamental disadvantage of graphical interfaces. For example, if themes created by third parties want to accept more values than specified by the program (e.g. a second image for a dialog), this would not be possible via the GUI. For this reason, I added two more ways of editing game elements: YAML and JSON. Like Markdown, both are markup languages, but they are used for data serialization, i.e. for storing data in an organized manner.[23] You could now flexibly switch between different display forms by switching between the respective tabs and also create and fill data fields that are not available in the graphical user interface. The big advantage of YAML is that all information can be displayed compactly in a text field:

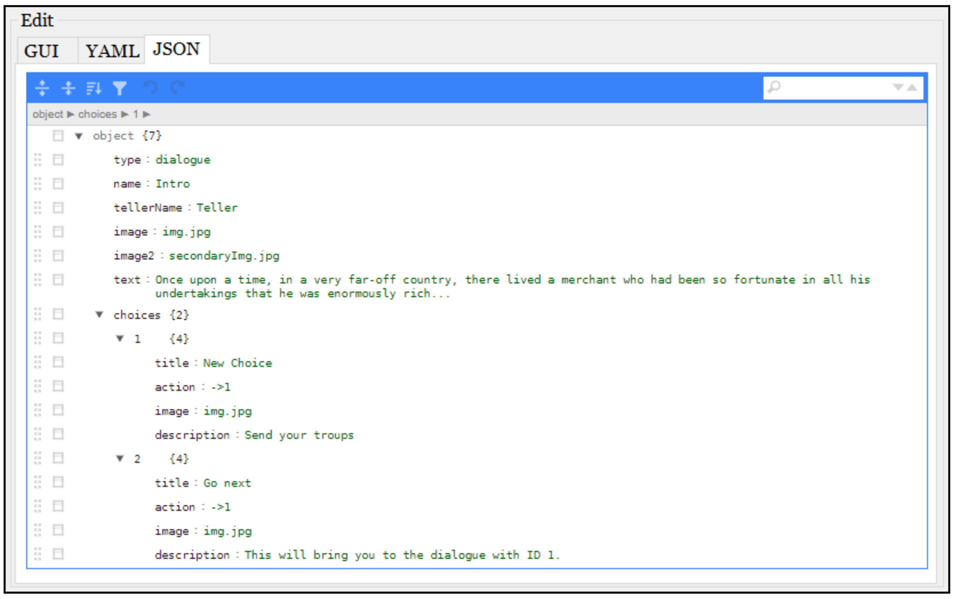


```
type: dialogue
name: Intro
tellerName: Teller
image: img.jpg
image2: secondaryImg.jpg
text: Once upon a time, in a very far-off country, there lived a merchant who had been so fortur
choices:
  1:
    title: New Choice
    action: ->1
    image: img.jpg
    description: Send your troups
  2:
    title: Go next
    action: ->1
    image: img.jpg
    description: This will bring you to the dialogue with ID 1.
```

Figure 12: Editing a dialog in MysteryMaker with YAML

[23] See Keshavarzi, Kaveh and Bayer, Thomas. predic8. "JSON, XML and YAML in comparison". https://www.predic8.de/xml-json-yaml.htm.

However, this type of data entry can seem strange to inexperienced users and quickly lead to formatting errors. Even one space too many or too few leads to the fact that the project cannot even be saved or converted to another form of display. In order to avoid this, no pure JSON source text is displayed with the JSON display form, but is illustrated by an external tool:



```
▼ object {7}
    type : dialogue
    name : Intro
    tellerName : Teller
    image : img.jpg
    image2 : secondaryImg.jpg
    text : Once upon a time, in a very far-off country, there lived a merchant who had been so fortunate in all his
           undertakings that he was enormously rich...
  ▼ choices {2}
    ▼ 1    {4}
        title : New Choice
        action : ->1
        image : img.jpg
        description : Send your troups
    ▼ 2    {4}
        title : Go next
        action : ->1
        image : img.jpg
        description : This will bring you to the dialogue with ID 1.
```

Figure 13: Editing a dialog in MysteryMaker with JSON visualization

The JSON display therefore combines the advantages of both languages: The user can Add, edit and review any new data fields without having to pay attention to the syntax of the language.

Another problem I encountered during development was incorporating interaction. The author of an interactive fiction should be able to determine what should happen once the player chooses a dialogue option. In the beginning I decided to use a specially developed scripting language in order to give the users as many options as possible. However, learning a scripting language is time-consuming and would deter many potential authors because they would not be able to apply what they already learned. For this reason, I switched to the scripting language GDScript, which was already established and, above all, already integrated in the Godot Engine. This also simplified the communication between the script and the Velius engine, which was also written in GDScript. However, while GDScript is said to be very beginner-friendly, it could be very intimidating to non-programmers, as well as being disproportionately complex for simple, maybe even linear, stories. Therefore, I subsequently added the option to enable simple operations such as jumping to other dialogs without writing a script using a drop-down menu:

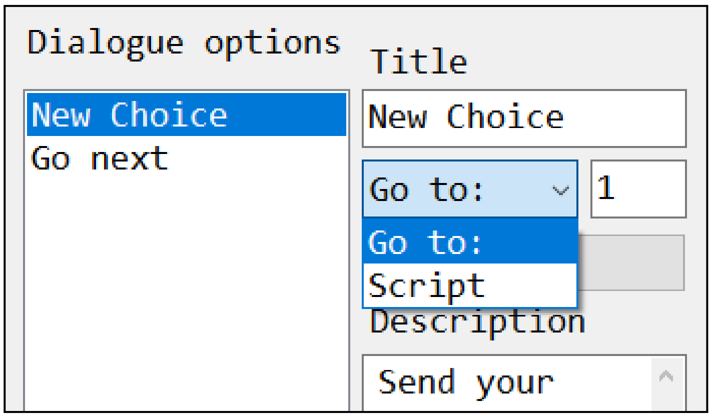


Figure 14: The "Go to" shortcut allows you to switch to another dialog by specifying the dialog ID

3.3 Finaler Status

Even if the MysteryMaker is not yet a fully-fledged authoring system due to a lack of options such as playing sounds or the inventory system that has not yet been implemented, non-linear and therefore interactive fictions can already be created. The theme system also gives theme developers extensive options for shaping the player experience. The system is so open that even entire games can be created within the game itself. In this way, the written stories could be made even more interactive by being able to influence the story not only through dialogue options, but also through smaller games such as "Tic-Tac-Toe" or puzzle games.

A separate plugin system, through which individual functions can be supplied by other developers, is also not yet available.

A major limitation is currently the publication of the projects. While similar authoring systems like Twine offer a website export, through which the game can easily be published as a browser game[24], MysteryMaker projects have to be manually exported as a file, sent and imported by the player, which is laborious.

4 Future plans and conclusion

Since I repeatedly encountered limitations during the development of the authoring system, which arose from the choice of Windows Forms as the basic technology, I plan to redevelop the MysteryMaker based on a newer and more established technology. Although this is very time-consuming, it will lead to a better user experience and make further development considerably easier.

[24] See Official Website. "Twine / An open-source tool for telling interactive, nonlinear stories". https://twinery.org/.

In addition, I will try to solve the problem of publishing MysteryMaker projects with the help of a free marketplace. This would then be integrated into the Velius Engine, allowing developers to publish their projects in a unified and centrally managed manner, and players to conveniently discover, download and play their interactive fictions in-game.

In conclusion, even though MysteryMaker and Velius Engine already offer many possibilities for developing interactive fiction, there is still a lot to be done to release a full version. However, since the project is open source and everyone is free to help develop it, continuous further development is also possible in the future.

References

"3 tools for creating your own text adventure games / gaming". n.d. Ephesos Software Accessed November 30, 2021. https://ephesossoftware.com/articles/gaming/3-tools-to-create-your-own-text-adventure-games.html.

ADRIFT: Create your own Interactive Fiction. n.d. Accessed October 29, 2021. https://www.adrift.co/.

"Adventure (1976)". 2021. At Wikipedia. https://de.wikipedia.org/w/index.php?title=Adventure_(1976)&oldid=213115756.

"Adventure". 2021. At Wikipedia. https://de.wikipedia.org/w/index.php?title=Adventure&oldid=216090578.

"Everything you need to know about game engines". 2020. nobreakpoints® (blog). July 14, 2020. https://blog.nobreakpoints.com/game-engines/.

"Godot Engine - Free and Open Source 2D and 3D Game Engine". Godot engine. Accessed November 29, 2021. https://godotengine.org/.

"How It Works". 2021. Vuepress Guide. Accessed November 29, 2021. https://vuepress.vuejs.org/guide/#how-it-works
"Interactive Fiction". 2021. At Wikipedia. https://de.wikipedia.org/wiki/Interactive_Fiction.

"markdown.de | Markdown Syntax Documentation". n.d. Accessed November 29, 2021. https://markdown.de/.

"Multimedia at a glance / design / content / authoring systems - Wikibooks, collection of free textbooks, non-fiction and specialist books". n.d. Accessed October 29, 2021. https://de.wikibooks.org/wiki/Multimedia_im_%C3%9Cberblick/_Gestaltung/_.

"Quest - Write text adventure games and interactive stories". n.d. Accessed 29 October 2021. http://textadventures.co.uk/quest.

"Reference documentation" 2020. Microsoft Docs. Accessed 01/22/2022. https://docs.microsoft.com/en-us/style-guide/developer-content/reference-documentation.

"Twine / An open-source tool for telling interactive, nonlinear stories". n.d. Accessed October 29, 2021. https://twinery.org/.

"Visual Novels". 2021. At Wikipedia. https://de.wikipedia.org/w/index.php?title=Visual_Novel&oldid=216681037.

"What exactly are authoring systems?" n.d. bild.de. Accessed October 29, 2021. https://www.bild.de/infos/e-learning/e-learning/autorensysteme-10752772.bild.html.

"What is an API? Simply explained!" o.J. Talend - A Leader in Data Integration & Data Integrity. Accessed November 29, 2021. https://www.talend.com/en/resources/was-ist-eine-api/.

adegeo "Desktop Guide (Windows Forms .NET)". 2021. Microsoft Docs. Accessed 01/22/2022.
https://docs.microsoft.com/de-de/dotnet/desktop/winforms/overview/?view=netdesktop-6.0.

Augsten, Stephen. n.d. "What is software documentation?" Accessed November 29, 2021. https://www.dev-insider.de/was-ist-eine-softwaredokumentation-a-744924/.

Funes, Antonio. "Game Engines: What is an Engine?". 2020. PC GAMES. May 24, 2020.
https://www.pcgames.de/Unreal-Engine-Software-239301/Specials/spiele-games-grafik-3d-historie-allgemein-epic-vorschau-1350751/.

Keller, Bernd. o.J. “interactive-fiction.de”. Text. Bernd Keller. Germany. Accessed October 29, 2021. http://www.interactive-fiction.de/.

Keshavarzi, Kaveh and Bayer, Thomas (2011) "XML, JSON and YAML in comparison". n.d. predic8.com. Accessed December 28, 2021. http://www.predic8.de/xml-json-yaml.htm.

lern.link team. o. J. "Author system". lern.link - eLearning with Moodle and PowerPoint (blog). Accessed October 29, 2021. https://lern.link/2020/lexikon/autorensystem/.

Lovato, Nathan. "GDScript DocsMaker". 2021. GitHub. Accessed 01/22/2022. https://github.com/GDQuest/gdscript-docs-maker.

Hell, Yvonne. "What Is a Technology Stack?" 2021. Mixpanel. Accessed July 23, 2021. https://mixpanel.com/de/blog/was-.

Figures

<u>of independent development</u>

I hereby affirm that I have completed the work independently , have not used any other aids than those specified and have clearly identified the passages in the technical work that were taken from other works in terms of wording or essential content with precise references to the source. Information used from the Internet has been made available to the teacher in full as a printout. I have properly saved all the Internet texts used on the enclosed CD (if necessary USB stick). I make the project work available to the school for educational purposes. The project work may be loaned to interested persons.

Place, date: 05.02.2022

Signature: [illegible]