# РЕШЕНИЯ ЗА НЕЛИНЕЕН РАЗКЛОНЕН ДИАЛОГ, ПОДХОДЯЩИ ЗА ИНТЕГРИРАНЕ В ТРИИЗМЕРНА ИГРОВА СРЕДА ЗА ЦЕЛИТЕ НА ОБУЧЕНИЕТО

Ивайло И. Буров

Шуменски университет "Епископ Константин Преславски"

***Резюме**: В тази статия е проведено изследване относно съществуващи решения за създаване на разклонени нелинейни диалози с цел интегрирането им в изграждана 3d игрова среда за целите на обучението посредством игровия двигател Godot. Отчетени са внедрени и липсващи функционалности в тестваните решения, възможностите за интеграция, степента на сложност при внедряването им. Публикуваните резултати могат да бъдат полезни за изследователи, насочили вниманието си към създаване на съвременни 3d игрови решения в обучението.*

***Ключови думи**: интерактивна, виртуална, 3d среда, диалози, диалогова система, интеграция, образование, 3d игри в образованието*

# SOLUTIONS FOR NON-LINEAR BRANCHED DIALOGUE SUITABLE FOR INTEGRATION IN A THREE-DIMENSIONAL GAME ENVIRONMENT FOR EDUCATIONAL PURPOSES

Ivailo I. Burov

Konstantin Preslavsky University of Shumen

***abstract:** In this article, a study was conducted on existing solutions for creating branched nonlinear dialogues in order to integrate them into a built 3d game environment for educational purposes using the Godot game engine. A report includes: implemented and missing functionalities in the tested solutions, the possibilities for integration, the degree of complexity in their implementation. The published results can be useful for researchers who have focused on creating modern 3d game solutions in education.*

***keywords:** interactive, virtual, 3d environment, dialogues, dialog system, integration, education, 3d games in education*

Провеждането на диалози е един от най-популярните интерактивни подходи в съвременното обучение, което означава че и при пренос на обучението в триизмерна виртуална среда, интегрирането на диалогова система е едно от условията за разширена интерактивност. Провеждането на диалози в класната стая не винаги е свързано с последващия ход на представяне на информацията, освен ако тези диалози не са включени в игрови сценарий. Във виртуална среда обаче, провеждането на диалози е свързано с последващия ход не само на представената информация, но и с развитието на хода на играта. Това прави т.н. линейни диалогови системи неподходящи за изискваната степен на интерактивност.

В литературата, киното, театралното изкуство и компютърните игри съществуват различни подходи при поднасяне на информация:

- **Линейният разказ** е една от най-често срещаните форми, като събитията се изобразяват най-общо в хронологичен ред, тоест разказът следва събитията в реда, по който са се случили;

- **Нелинейният** разказ е подход при който събитията не следват реда, по който са се случили и разказът излиза извън директния причинно-следствен модел. При тази техника могат да бъдат развивани паралелно различни страни на една история. Нелинейният разказ се доближава до интерактивния разказ, когато се представя какво би могло да се случи, ако определени персонажи са направили различни избори.

- **Интерактивният разказ** е изграден върху разклонена структура, при която една изходна точка може да доведе до множество развития и резултати. Този подход се използва в компютърните игри като на всяка стъпка се предоставя избор спрямо който се развива ходът на играта, което води до нови серии от избори. Следователно създаването на нелинеен разказ или диалог предполага представянето на неопределен брой паралелни истории.

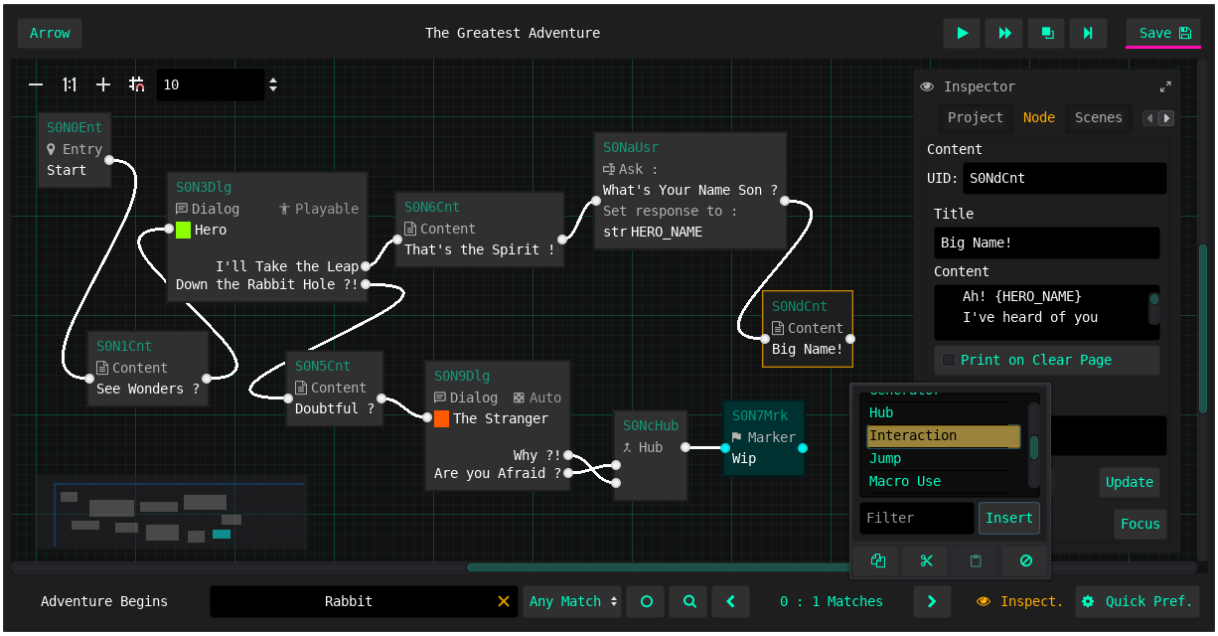
Съществуват различни решения за нелинейни диалогови системи, някои от които са реализирани като отделни апликации, насочени към универсално използване - т.е. създадените с тях диалози да бъдат използвани с различни игрови двигатели, а други - насочени към избран игрови двигател. Въпреки различния избор на визуализацията при създаване на диалозите, като общи между тях се различават следните основни компоненти:

- Система за създаване на диалог;
- Система за интеграция на създадените диалози в игровия двигател.

Понеже в разработвания проект за триизмерна интерактивна виртуална среда за обучението е избран игровия двигател Godot [1], тук ще бъдат визирани някои от съществуващите диалогови решения, насочени към този игрови двигател или към универсално ползване.

**Arrow** (Фигура 1.) е удобен, безплатен и богат на функции инструмент с отворен код за проектиране на разказ на играта, разработване на текстови приключения или

създаване на всякакъв друг вид нелинейни интерактивни документи за разказване на истории [2].




Фигура 1

Визуалният интерфейс е под формата на диалогово дърво, което е удобен избор при създаване на сложни разклонени диалози. В допълнение даден диалог може да бъде продължен посредством създаване на нова сцена, чрез което се избягва претрупване на визуалните елементи при сложни диалози.

**Реализирана функционалност:** Персонализирана система за възли с вградени типове възли като: съдържание; диалогов прозорец; условен възел; модификатор на променлива; потребителско въвеждане; макрос (за многократна употреба) и др. Характеризира се с лесно управление на сцената и навигацията, предварително възпроизвеждане на диалозите с едно щракване, експорт и импорт на диалозите като JSON тип файл. Притежава вградена възможност за експорт на диалозите под формата на интерактивен уеб документ. Функционалността за JSON импорт и експорт предполага универсално ползване от различни игрови двигатели, но на този етап липсата на зареждащ диалозите модул се явява ограничение и се отчита по-долу като липсваща функционалност.

Освен предоставения свободен код, се предлага изтегляне на компилиран вариант за Windows и Linux, а също е достъпна и онлайн версия под формата на уеб апликация.

**Забелязана липсваща функционалност:** Въпреки че този инструмент е създаден с Godot игровия двигател, не е разработена система за въвеждане на създадените диалози от игрови двигатели, вкл. от Godot. Това ограничава приложението му до създаване на текстови новели, освен ако въвеждащата диалозите част не бъде създадена индивидуално от потребителя, ползващ Arrow. Това обаче изисква добро запознаване с дизайна на съществуващия код и време за разработка на допълнителния модул за въвеждане на диалозите.

**Dialogue Designer:** Диалоговият дизайнер и редактор на разклонени сюжетни линии (Фигура 2) е платен инструмент (9.99$), създаден с игровия двигател Godot, насочен към универсална употреба [3].

Фигура 2

**Реализирани функционалности**:

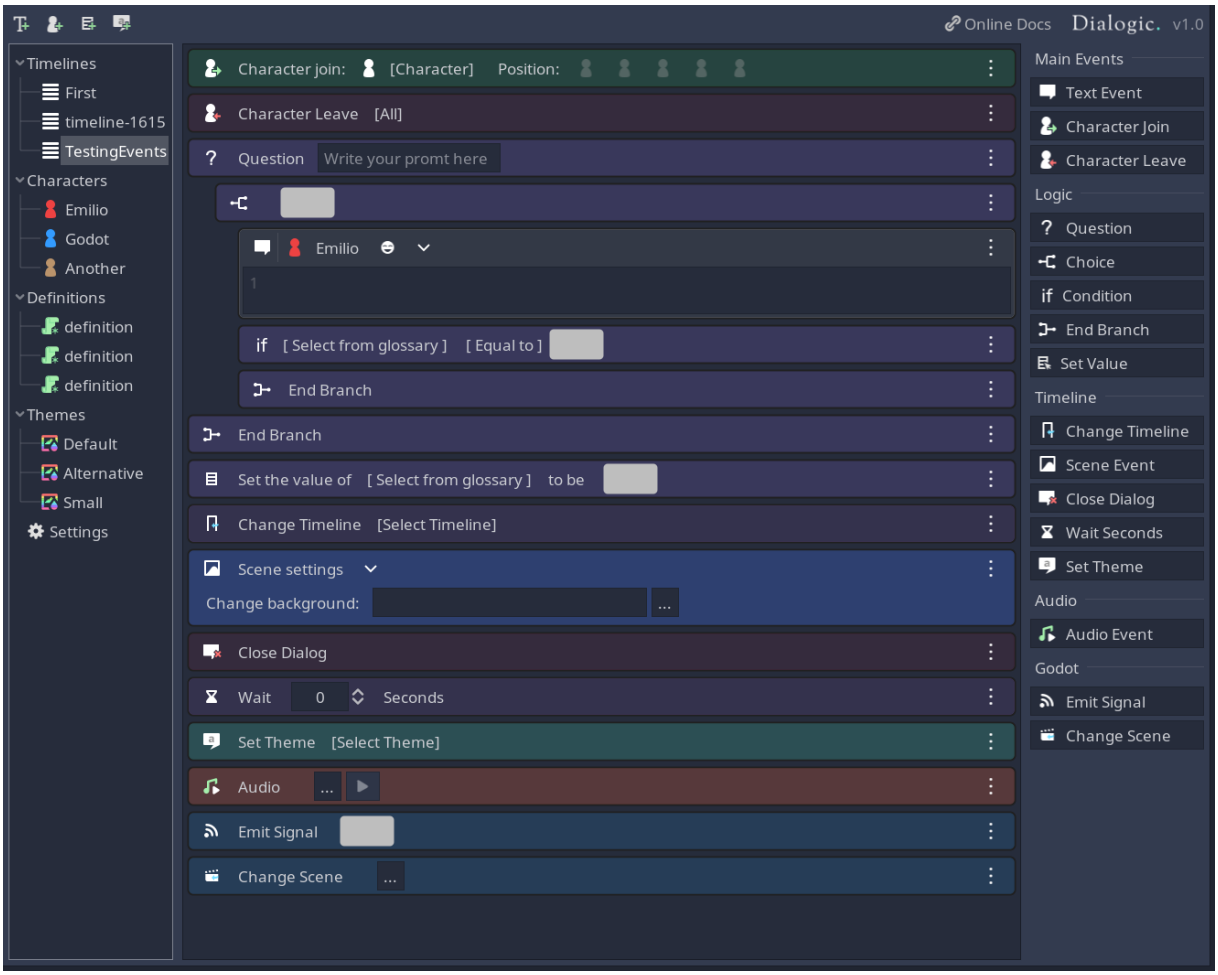
Визуалният интерфейс е под формата на диалогово дърво, с възможност за избор на различни типове възли като: съобщение, избор, повторение, шанс, случайност, изчакване, установяване на променливи и дори изпълнение на функции, извиквани от кода на игровия двигател. Поддържа се многоезиковото създаване на диалози. Диалоговата система за вграждане в игровия двигател е с готов графичен потребителски интерфейс. Авторът предлага безкраен безплатен ъпдейт на всички закупили системата, независимо от версията.

Допълнително към дизайнера на диалози е предоставена и система за въвеждането им в Godot engine Dialogue System 2, 3 (6.99 $), както и описание към структурата на експортирания диалогов JSON файл за улеснено въвеждане с други игрови двигатели [4], което предполага универсално приложение на създадените диалози.

**Отчетени недостатъци**:

Невисоката цена на диалоговия дизайнер няма да бъде отчетен като недостатък, но платената система за въвеждане на диалозите се явява спънка при създаване на проекти с отворен код (като при игри за целите на обучението) и поради тази причина може да се отчете успешно прилагането на този инструмент само при проекти със затворен код.

**Dialogic** [5] е инструмент за създаване на нелинейни диалози в Godot игровия двигател (Фигура 3). Визуалният интерфейс е представен като последователност от диалогови елементи като: въпрос, избор, състояние, текстово събитие, събитие от сцената, присъединяване или напускане на персонаж и др. Предвидена и функционалност за аудио събитие. Преходите при избор се затварят с елемент за край на прехода, което донякъде имитира диалогово дърво. Както и при досега описаните приложения и тук е възможно управлението на променливи и установяване на техните стойности в рамките на диалога. Освен развитата система за събития, е вградена и функционалност за изпращане на сигнали, използвана като основен дизайн в игровия двигател Godot за комуникация между различни части на кода. Данните се съхраняват в json формат, което осигурява преносимост на диалозите, ако е необходимо използването им в друг игрови двигател.

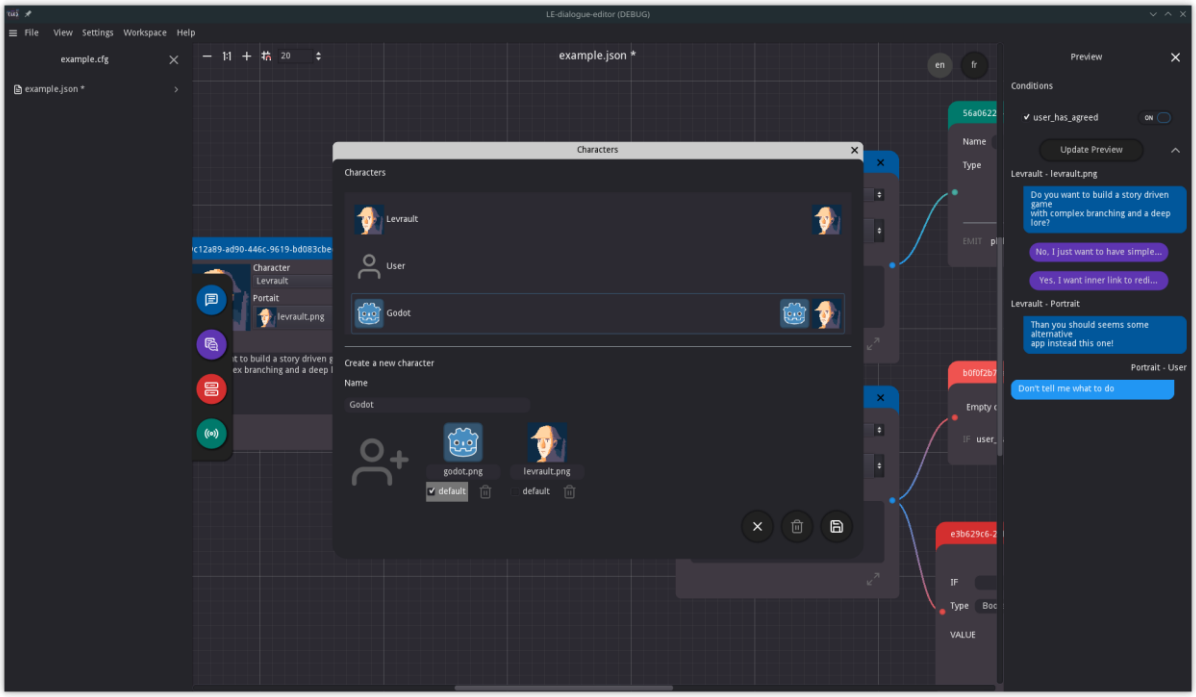



Фигура 3

Според автора на статията, изборът на интерфейс е повече подходящ за създаване на визуални новели, докато използване на системи от тип „диалогово дърво“ са по-удобни за неголеми проекти в триизмерно обкръжение. Въпреки че това е въпрос на предпочитания, при системите от тип „диалогово дърво“ освен представената последователност и проследяване на преходите може да бъде визуализирана и паралелност при отделните линии на развитие на диалозите.

**LE Dialogue Editor** (Фигура 4) е самостоятелно приложение, създадено с игровия двигател Godot, с основна цел за управление на малки, но сложни диалози [6].

Приложението е с отворен код, достъпен за модификация, а също така за изтегляне са налични компилирани версии за Windows и Linux. Авторът на приложението не препоръчва използването му за създаване на визуални новели, като за тази цел определя Dialogic или специализирани за целта апликации като по-подходящи.




Фигура 4

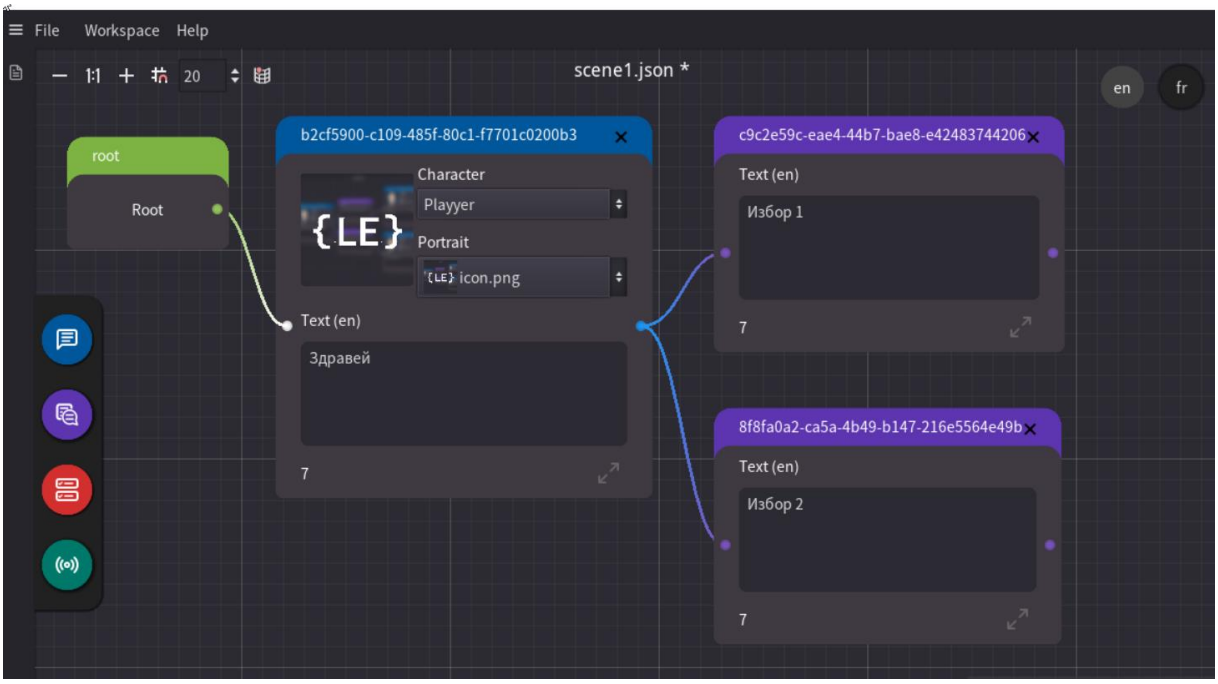
**Реализираните функционалности** са: многоезикова поддръжка посредством динамична локализация; създаване и персонализиране на библиотека с диалогови герои; разнообразни диалогови възли като избор, разклоняване, условно разклоняване както за диалог, така и за избор; визуализация на герои с име и множество портрети; задаване на максимален брой букви на диалог; избор на диалога въз основа на състоянието; излъчване на персонализирани сигнали за диалог; редактиране на няколко JSON файл едновременно, като този формат способства за работа с всеки игрови двигател, който може да чете JSON файл.

**Забелязани липсващи функционалности**

Не се предлага самостоятелен модул или пример за въвеждане на диалоговите json файлове, като това се оставя на потребителя, но за целта може да се използва кода от системата за анализиране (parsing) на json файла. Както в повечето представени дотук диалогови системи, не е описана функционалност за поддръжка на аудио диалози - това трябва да се извършва чрез допълнителна адаптация на кода.

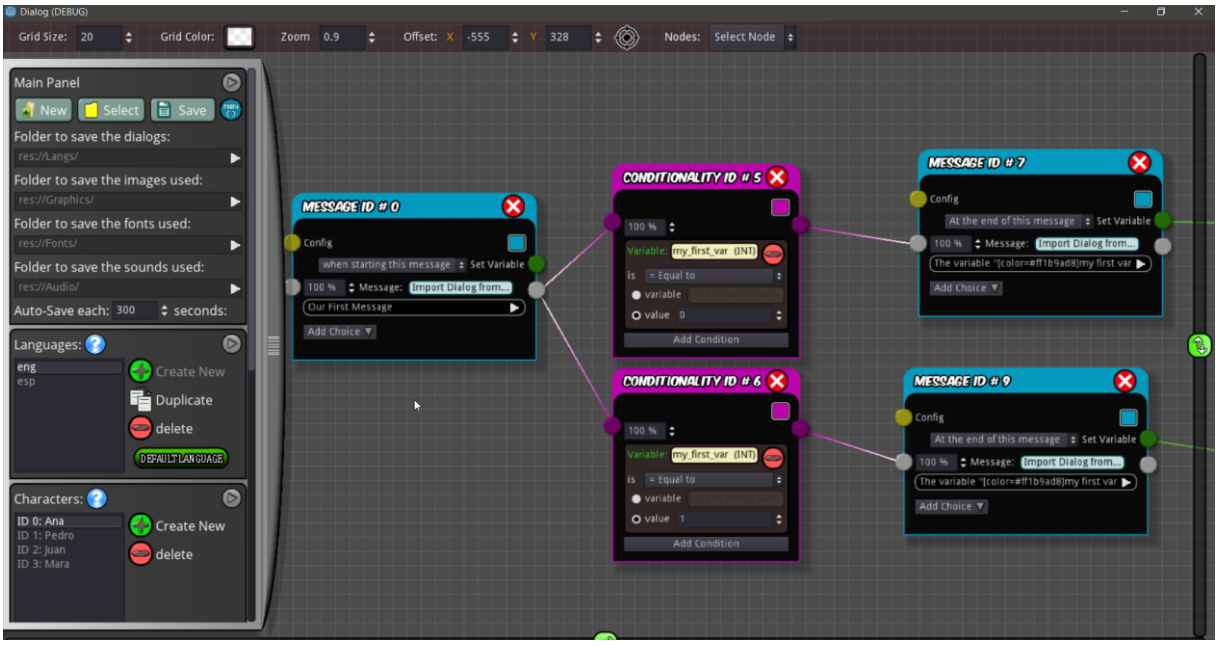
Като недостатък според автора на тази статия се отбелязват все още съществуващите бъгове, понякога водещи до сриване на апликацията, а също и прекомерно големия размер на диалоговите възли, които при намален мащаб стават трудно четими. Диалоговите възли за избор например се реализират от по няколко отделни съставни възела, а не от един с множество изходи за избор. В този случай при

задаване на множество избори се губи голяма част от работното пространство (Фигура 5).




Фигура 5

**Advanced Dialog System** (Фигура 6) е приставка (Plug-in) към игровия двигател Godot с отворен код и MIT лиценз, което прави достъпно използването и модифицирането на съществуващия код в потребителски приложения [7]. Предоставен за изтегляне е и пример за използване на създаден диалог в Godot апликация.




Фигура 6

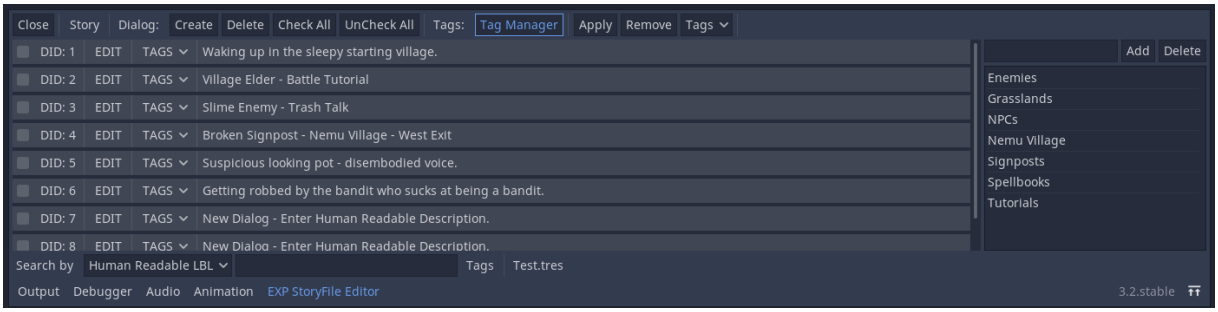
Една от най-отличителните страни на тази диалогова система е вграденият разширен текстови редактор, предлагащ различни форми на модификация на текстовите пасажи - шрифт, цвят, фонт, стил и др., посредством BB код. Реализирана е многоезикова

поддръжка, а съществуващите диалогови възли са: съобщение, условен преход, дефиниране на променливи за условен преход. Може да бъде зададена желана конфигурация за тема, обхващаща различни визуални диалогови елементи и техните стилове. Предвиден е и експорт на диалозите в json формат, но по подразбиране те се съхраняват в собствен двоичен формат с разширение .LANG в зависимост от езика, който е избран да се поддържа. Като недостатък може да се отбележи това, че първоначалното запознаване с диалоговата система може да затрудни потребителя, но за целта авторът прилага обучителни видеоклипове в youtube.

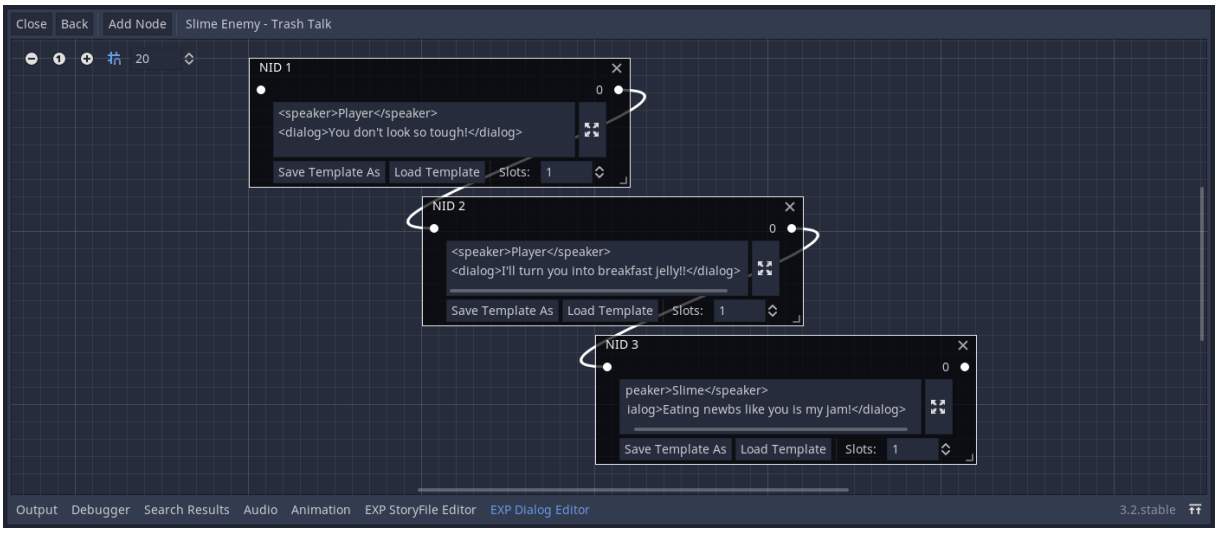
**EXP Godot Dialog System** е добавка, която подпомага генерирането, организацията, съхранението и достъпа на големи количества текстови данни за проекти на Godot Engine. Състои се от:

- редактор на истории, който да създава и управлява списъци с диалогови записи чрез маркери и филтри за търсене (Фигура 7);

- редактор на диалози, чрез който се създават и редактират текстови/диалогови данни чрез графични възли (Фигура 8). Текстовите данни могат да бъдат запазени като ресурсни файлове и да се четат в кода на играта чрез предоставения клас на помощник.




Фигура 7. Редактор на истории

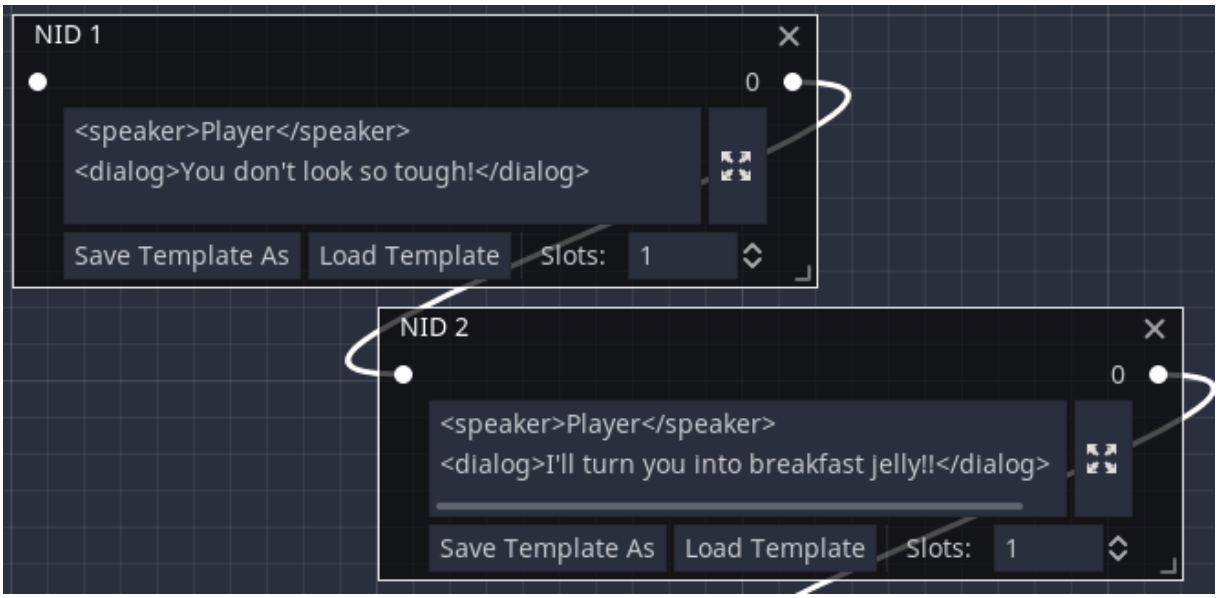



Фигура 8. Редактор на диалози

В редактора на файлове с истории могат да бъдат създавани и изтривани диалогови записи. На всеки нов диалогов запис автоматично се присвоява уникален диалогов идентификатор (DID).

В менюто "Story", чрез Save AS, се запазват текстовите данни и тези за свързване на отделните възли като ресурсен файл на историята, плюс информация, необходима за зареждане на данните обратно в редакторите за по-нататъшно редактиране.

На диалоговите записи може да се присвои име, което е свързано с техния идентификатор (DID), но това име също трябва да бъде уникално. На всеки новосъздаден възел бива назначен уникален идентификатор (NID). Текст може да бъде въведен във всеки възел. Щракването върху бутона за разширяване отваря по-голям прозорец на тестовия редактор, което е удобно при работа с големи пасажи от текст. Всеки възел има един входен слот, към който могат да се свържат множество на брой възли, а също така да се зададат и произволен брой изходни слотове.



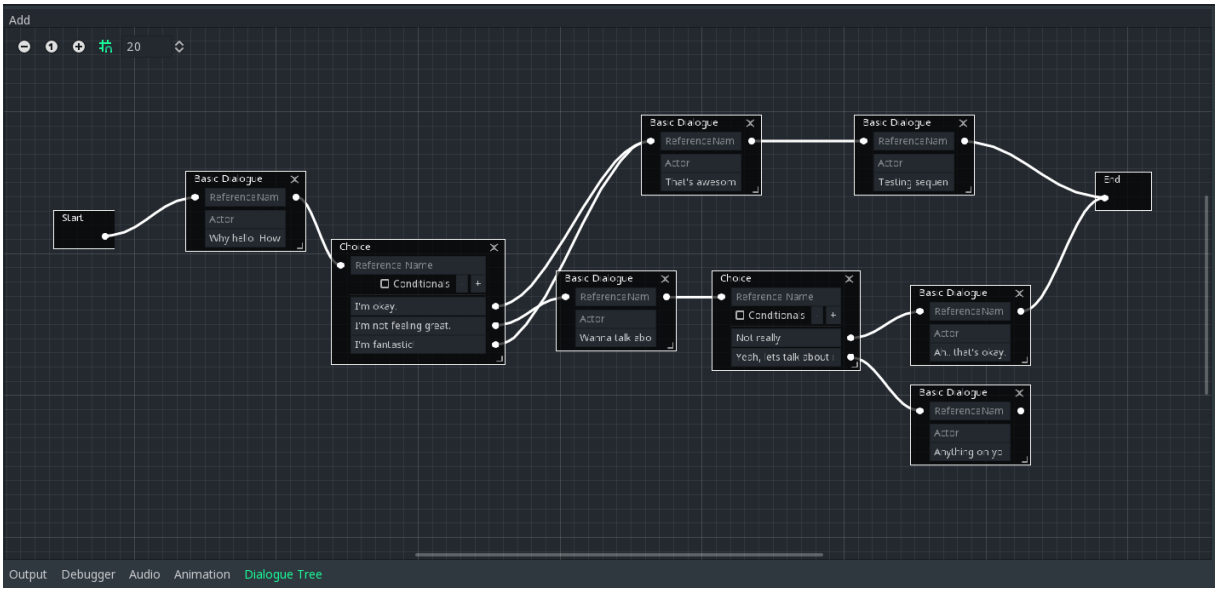

Фигура 9

Предвидена е и възможност за запазване и зареждане на текстови шаблони от възлите в редактора на диалоговия прозорец, което до известна степен улеснява процеса на въвеждане на текст откъм грешки.

Както се вижда от фигура 9, отделните диалогови възли не са специализирани както в предишно представените диалогови системи, а тяхната диференциация се определя от използваните в диалога етикети, ограждащи съдържанието. Чрез този подход се постига по-голяма универсалност на приложение на отделния диалогов възел, за сметка на по-усложнената част, анализираща съдържанието му. На база на този принцип могат да бъдат дефинирани не само типа диалогови възли и персонажите, които ги възпроизвеждат, но и да бъдае добавена информация за аудио диалози, многоезикова поддръжка и др., разбира се за сметка на усложняването на анализатора на диалоговото съдържание. Като недостатък може да се отчете, че писането на подобни тагове в диалоговия възел е досадно и провокира допускане на грешки, но поради тази причина авторът на проекта е предвидил запаметяване и въвеждане на информацията за определен тип възел като шаблон. На по-късен етап тази техника е приложена от автора на статията в диалогова система със специализирани диалогови възли, за разширяване на функционалността им. Предвидена е многоезиковата поддръжка посредством експорт на диалоговите данни и тези на историята в CVS формат, удобен за редактиране.

Импортът на тези данни обаче не генерира диалози, а само подменя текстовото съдържание на вече създадени такива.

**DialogueTreePlugin** [8] е добавка към игровия двигател Godot, посредством която създаването на нелинейни диалози се извършва директно в игровия редактор. Броят на създадените диалози е неограничен. Графичният интерфейс е организиран под формата на диалогово дърво, съдържащо специализирани възли за взаимодействие като: базов диалог, избор, условен преход, случаен преход, посредством които може да бъдат реализирани паралелно развиващи се сценарии (Фигура 10).



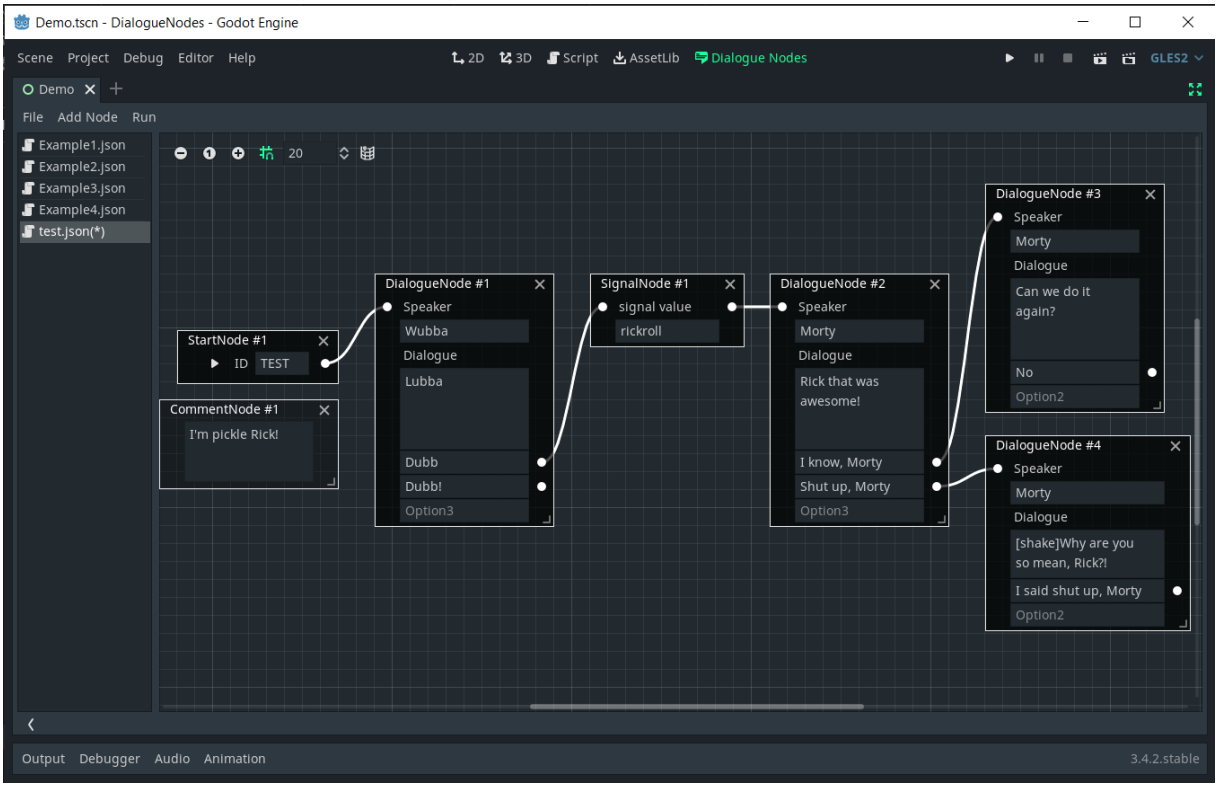

Фигура 10

В базовите диалогови възли са заложени полета като име на възела (за извикване на събития въз основа на конкретни диалогови възли), име на актьора и текст на диалога, като името на актьора може да се прилага при позициониране на диалога спрямо определен персонаж. Възлите за условен преход служат за контролиране на потока на диалога, при което автоматично се извиква функцията за следващ диалог (next_dialogue), в зависимост от изпълненото условие. Полето за име (ReferenceName) във възела за условен преход може да се използва за извикване на събития в Godot спрямо специфични диалогови възли. Системата генерира сигнали за старт и стоп на диалога, следващ диалог и обмен на данни при условен преход, посредством което може да се извършва трансфер на променливи от кода в игровия двигател към диалоговата система. Това може да бъде отделна променлива, но понеже трансферът е реализиран посредством тип Dictionary в gdscript, става възможно прехвърлянето на множество променливи за условия с едно изпращане на данни.

Като основно предимство на този проект може да се отчете лекотата при неговото използване, а като недостатъци това, че данните се съхраняват директно в сцената, в основния проект не е предвидено автоматичното разширяване на диалоговите полета, липса на многоезикова поддръжка и др. Като едно от по-рано появилите се решения за създаване на диалози с интерфейс от тип „диалогово дърво“ на сегашния етап от развитието на проекта по създаване на триизмерна игрова среда за целите на обучението

се използва този диалогов проект. За целта обаче са извършени допълнителни изменения и допълнения като поддръжка на аудио диалози, мащабиране на полетата при по-големи диалогови пасажи, функционалност за създаване на множество полета в базовия диалог с цел постигане на универсалност и многоезикова поддръжка.

**Dialogue Nodes** (Фигура 11) е Godot приставка (Plug-in), която помага при създаването и използването на разклонени диалози в игровия проект. Характеристиките включват: редактор на дървото на диалога; отваряне, редактиране и запазване на множество файлове; множество диалогови дървета в един json файл; вграден диалогов тестер за предварително проиграване на създадените диалози; наличен диалогов интерфейс за стартиране на диалоговите дървета в потребителския интерфейс.

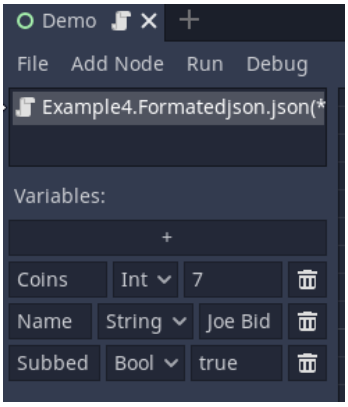



Фигура 11

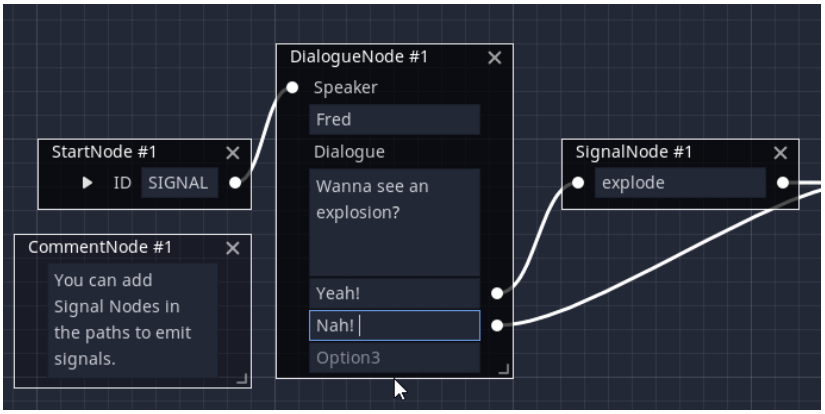
**Налични функционални възможности**

Една от характеристиките, несрещани в представените досега диалогови системи е възможността за съхранение на множество диалогови дървета в един json файл. Вградена е функционалност за поддръжка на BB код в диалозите, а също така и изпълнението на ефекти в тях. Поддържат се променливи в диалоговото дърво, които могат да бъдат модифицирани според условията в самия диалог или чрез потребителски код (Фигура 12). Многоезиковата поддръжка предполага зареждане на json файл, адаптиран към съответния език, въпреки че това не е описано от автора на проекта.

Диалоговите възли са: диалогов; за условен преход; за изпращане на сигнал; за установяване стойността на променлива; за коментар. Не съществува специализиран възел за избор, като тази функционалност е реализирана като динамично добавящи се полета към диалоговия възел (Фигура 13).




Фигура 12




Фигура 13

Проектът се представя с налични четири примера за приложение, разкриващи функционалните му възможности, което улеснява прилагането му в потребителски проекти.

При теста и изследването на проектния код се установи, че функциите, отговорни за комуникация с потребителския код са реализирани като код от диалоговата кутия.

**Забелязани недостатъци:**

Въпреки преобладаващите плюсове в този проект, трябва да се отчетат и някои недостатъци, според автора на статията:

- Слотовете за връзка с други възли са разположени долу вдясно, при което при опит за мащабиране на диалоговия възел (при голям текстов пасаж) се включва функцията за връзка с друг възел, което е повече недостатък на Godot, отколкото на този проект. Същия проблем се наблюдава и при други проекти, където е използван графичния възел в Godot.
- Слотовете за избор, реализирани като полета от диалоговия възел са базирани на LineEdit контрола, вместо TextEdit, което е подходящо за кратки едноредови текстови опции за избор. Това може да бъде коригирано чрез конвертиране на типа на графичния контрол и малка добавка на код, отговорен за мащабирането му.
- Слотовете за избор се генерират динамично и понякога опцията за следващ избор изчезва. За да се появи трябва да се постави например интервал в полето за предишен избор, при което полето с опция за нов избор се появява отново.

**Интернет ресурси, използвани при разработката**

1. GODOT, https://godotengine.org/
2. Arrow, https://github.com/mhgolkar/Arrow
3. Dialogue Designer, https://radmatt.itch.io/dialogue-designer
4. Dialogue System 2, https://radmatt.itch.io/dialoguesystem2
5. Dialogic, https://github.com/coppolaemilio/dialogic
6. LE Dialogue Editor, https://levrault.itch.io/le-dialogue-editor

7. Advanced Dialog System, https://newold3.itch.io/advance-text-editor
8. DialogueTreePlugin, https://github.com/LexiEmory/DialogueTreePlugin

*доц. д-р Ивайло Иванов Буров*
*Шуменски университет "Епископ Константин Преславски"*
*имейл: i.burov@shu.bg*