Универсальный интерактивный редактор диалогов

(ВИДЕО)

2.2.2

Кристиан Хендерсон

Блог: https://videdialogues.wordpress.com/

Посетите блог для получения новостей, часто задаваемых вопросов, онлайн-документации, отслеживания ошибок и поддержки.

Если вам нравится этот актив и вы хотите поддержать его, рассмотрите возможность пожертвования.

Показатель:

Что такое ВИДЕ?

Что вы будете использовать

Как начать?

Редактор ВИДЕО

- Панель инструментов
- Диалоговый узел
  - Формат
  - Комментарии
    - Данные для каждого комментария
  - Спрайты
    - Дополнительные переменные
- Узел действия
  - Предопределенные действия:
- Назначить вид
- Встроенная локализация
- Мини-карта

ВИДЕО Назначить

использование

Как это устроено

Загрузка/выгрузка диалогов

Справочник по API сценариев

Список изменений

~Благодарности~

Что такое ВИДЕ?

VIDE (Versatile Interactive Dialogue Editor) — это плагин для Unity3D, который упрощает создание сложных интерактивных диалогов, предоставляя пользователю мощный интерфейс узла. VIDE упорядочивает данные узла из созданных диалогов и представляет их кодировщику в удобном, необработанном виде. Он спроектирован так, чтобы его можно было адаптировать к любому пользовательскому диалоговому интерфейсу и взаимодействовать с ним, что позволяет добавлять к нему дополнительные системы и функции.

Некоторые из ключевых функций VIDE включают в себя:

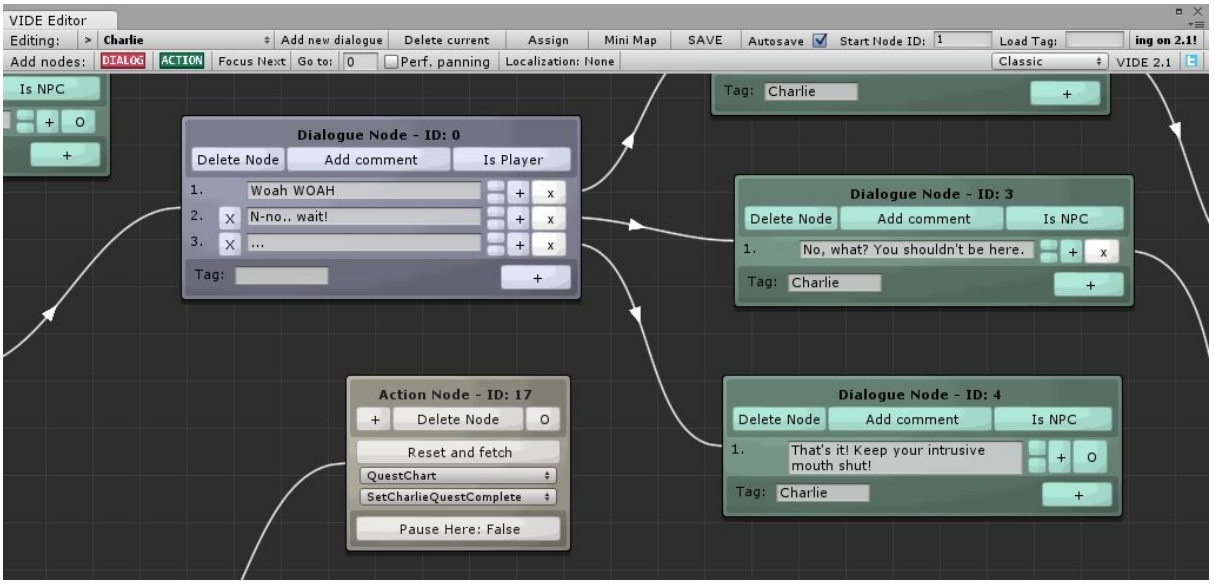
- Редактор VIDE: простой, но мощный интерфейс узлов для создания диалогов.
- Добавляйте спрайты, текст, теги и многое другое в свои узлы.
- Используйте узлы действий для вызова внутриигровых методов.
- Иметь контроль над потоком и началом разговора во время выполнения
- Будь креативным! Система не ограничивается только разговорами
- Переключение между узлами в стиле NPC и Player
- Несколько комментариев на узел
- Относитесь к узлам как к безграничным узлам с множественным выбором для игрока.
- Дополнительные данные и дополнительные переменные на узел, чтобы добавить больше функциональности
- Совместимость с большинством графических интерфейсов и плагинов локализации
- Встроенная система локализации
- Несколько экземпляров диалога
- Сохранить/загрузить состояния данных
- 6 демонстрационных сцен
- Настраиваемые скины редактора
- Аудио и спрайты для каждого комментария
- Выберите и переместите несколько узлов

VIDE не совместим с платформой Unity WebPlayer или любой версией Unity ниже 5.

Что вы будете использовать

Чтобы использовать VIDE Dialogues, вам придется поиграть со следующим:

- **Редактор ВИДЕО:** Здесь вы можете создавать, изменять и назначать новые диалоги. Доступен из Window>VIDE Editor или из VIDE_Assign.



- **Компонент VIDE_Assign:** Компонент, который будет добавлен к игровым объектам, с которыми вы хотите взаимодействовать. Вы назначаете ему диалог и изменяете некоторые свойства.

- **Ваше любимое программное обеспечение для кодирования:** Да, вам все равно придется немного программировать.

Как начать?

Для начала настоятельно рекомендуется сначала ознакомиться с предоставленными демонстрационными версиями. Они расположены в *ВИДЕО/Демо/*
Просто загрузите сцены и нажмите «Воспроизвести», чтобы увидеть все это в действии, а затем взгляните на диалоги и сценарии.

Есть 6 примеров сцен, с которыми вы можете поиграть:

Взаимодействие с игроком: Основная и самая большая сцена в виде от 3-го лица, демонстрирующая различные типы диалогов и взаимодействий.

Очень простой: Охватывает минимальную настройку, необходимую для работы с использованием класса GUI.

Локализация: Простой диспетчер пользовательского интерфейса, показывающий локализованный диалог в действии.

Мультиэкземпляры: Демонстрация, показывающая, что вы можете сделать с классом VD2, чтобы одновременно запускать несколько экземпляров диалогов.

Повествование: Еще один пример простого диалога с использованием системы Navi.

Шаблон: Демонстрация с использованием сценария Template_UIManager, который поможет вам начать работу.

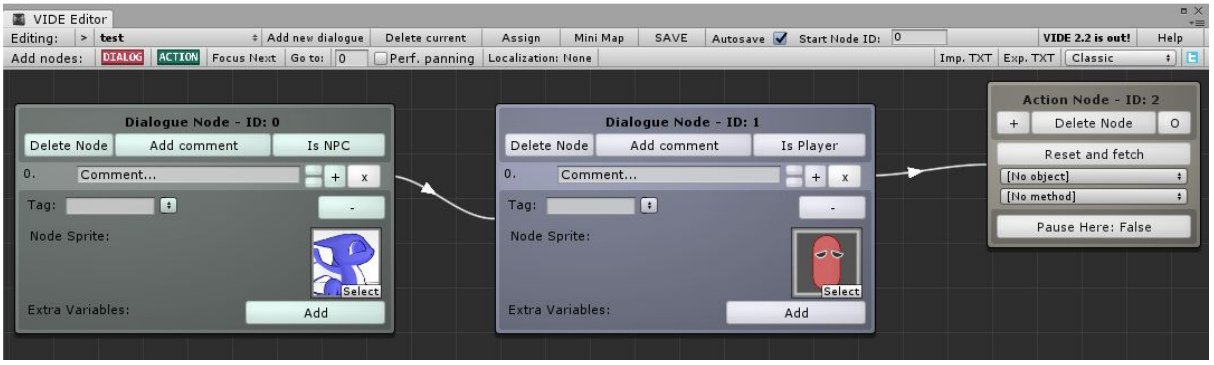
Пока вы проверяете**PlayerInteraction** сцена, убедитесь, что вы проверили **VIDEUIManager1** также скрипт (прикрепленный к игровому объекту UIManager в сцене). Он содержит различные демонстрации того, как использовать данные, которые предлагает VIDE, для изменения хода разговора. Версия шаблона этого скрипта, которую вы можете использовать, называется **Template_UIManager** и вы можете найти его внутри**Шаблон** демо.

Но все начинается с создания собственных диалогов! Вы можете редактировать диалоги в примерах по своему желанию. Просто запустите VIDE Editor и выберите или создайте диалог, который вы собираетесь изменить! См. следующую страницу для справки.

Важная заметка:Настоятельно рекомендуется хранить папку VIDE в корне вашего проекта, чтобы предотвратить любые нежелательные ошибки и поведения!

Редактор ВИДЕО

Здесь вы будете создавать свои диалоги, добавляя**Диалоговые узлы** и соединяя их. Вы можете соединять узлы по своему усмотрению. Кроме того, вы можете создавать узлы действий для вызова своих методов и изменения хода разговора.



1.**Панель инструментов**

Редактирование:Текущий диалог редактируется. Нажмите**«>»** или ударить**Войти** пока ни один узел не будет выделен, чтобы войти в фильтрованный поиск и выбрать другой диалог для редактирования.

Добавить новый диалог:Создайте новый диалог, чтобы начать редактирование.

Удалить текущий:Удалить текущий диалог и загрузить по умолчанию.

Назначать:Введите Assign View, чтобы назначить текущий диалог игровым объектам.

Мини-карта:Войдите в режим просмотра мини-карты.

Сохранять:Сохраните текущие изменения диалога.

Автосохранение:Когда включено, ваш диалог будет сохраняться каждый раз, когда окно теряет фокус.

Идентификатор начального узла:Первый узел, который будет загружен при вызове VD.BeginDialogue().

Область новостей:Кликабельная область новостей. Откроет браузер по ссылке новостей.

Помощь: Меню помощи.

Добавить узлы:Перетащите новые узлы.

Фокус Далее:Сфокусировать представление на следующем узле на основе идентификатора.

Идти к:Сфокусировать представление на узле с идентификатором.

Перф. панорамирование:Включить / отключить перетаскивание производительности.

Локализация:Войдите в меню локализации, чтобы управлять локализацией.

Имп. Эксп. ТЕКСТ:Импорт и экспорт диалогов в текстовые файлы.

2.**Диалоговый узел**

Диалоговые узлы являются основным источником контента. Они содержат текст, спрайты, аудио, теги и дополнительные данные и переменные. Вы можете подключить их к другим узлам диалога или узлам действий.

Каждый узел будет иметь уникальный идентификатор, который можно использовать для установки начальной точки.

Вы можете пойти простым или сложным

Формат

Диалоговые узлы имеют два формата:**NPC** и**Игрок**.

Узлы NPC могут быть подключены только к одному узлу, в то время как комментарий узла Player может быть подключен к другим узлам. Эта разница также будет отражена в переменной:**NodeData.isPlayer**. Это позволит вам легко определить формат для обработки данных по-разному.

```
{
    if (data.isPlayer)
    {
        SetPlayerChoices();
    }
    else
    {
        WipePlayerChoices();
        StartCoroutine(ShowNPCText());
    }
}
```

Комментарии

Каждый узел может иметь бесконечное количество комментариев. Вы сможете получить их через массив строк:**NodeData.comments**.Вы также используете**NodeData.commentIndex** для навигации по массиву или установите его на выбор игрока (см.*ВД.Далее()* в API для получения дополнительной информации о **комментарийИндекс** поведение).

Данные для каждого комментария

Каждый комментарий может иметь спрайт, аудиофайл и дополнительные данные. Вы можете использовать поле дополнительных данных для передачи дополнительной информации. Вы можете получить его как массив строк из **NodeData.extraData**.

Спрайты

Вы можете добавить спрайты к диалоговым спрайтам по умолчанию, узлу диалога и к каждому комментарию.

В Инспекторе вы можете установить спрайты по умолчанию для узлов Player и NPC. Внутри каждого узла и комментария узла вы можете установить еще больше спрайтов.

Спрайты (**и аудио**)должны быть внутри**Ресурсы** папка для загрузки. Помните, что у вас может быть столько папок Resources, сколько вы хотите, и внутри любого каталога, который вы хотите. если ты **не** хотите использовать папку «Ресурсы», вы можете иметь свой собственный массив спрайтов и аудио, на которые ссылаются, и передать его в*База данных VD.sprite* и*VD.audioDatabase* . Вместо этого VIDE будет использовать эти массивы.

Вы можете получить доступ к спрайтам по умолчанию из**VIDE_Assign.defaultPlayerSprite** и **VIDE_Assign.defaultNPCSprite**.

Спрайты будут доступны в NodeData как **спрайт**,и**комментарийСпрайты**.

Взгляните на**NodeChangeAction** метод в**примерUI.cs** для справки об одном способе обработки этих переменных.

```
//Set node sprite if there's any, otherwise try to use default sprite
if (data.sprite != null)
    PlayerSprite.sprite = data.sprite;
else if (VD.assigned.defaultPlayerSprite != null)
    PlayerSprite.sprite = VD.assigned.defaultPlayerSprite;
```

Дополнительные переменные позволяют пользователю хранить**строка, интервал, поплавок,** и **логический** значения в словарь с помощью пользовательских ключей. Эта информация будет храниться в NodeData как**экстравары**.Это даст вам больше контроля над поведением ваших узлов.

В редакторе VIDE вы можете установить ключ и значение для любого количества словарных статей, которое вам нужно.

В зависимости от того, как вы введете значение, оно будет**проанализировано** соответственно в**<строка, объект**>Словарь.

- Bool обнаружение **нет** чувствителен к регистру и может иметь**пустые места**.
- Обнаружение плавающего положения будет распознавать только ' .' .
- Обнаружение Int будет искать ряд целых цифр.
- Строка будет запасным вариантом, если ничего из вышеперечисленного не обнаружено.

Используйте ключ словаря, чтобы получить значение и **В ролях объект** к правильному типу. Если вы не уверены в типе, помните, что вы можете**ПолучитьТип()**


```
if (data.commentIndex == (int)data.extraVars["itemLine"])
```

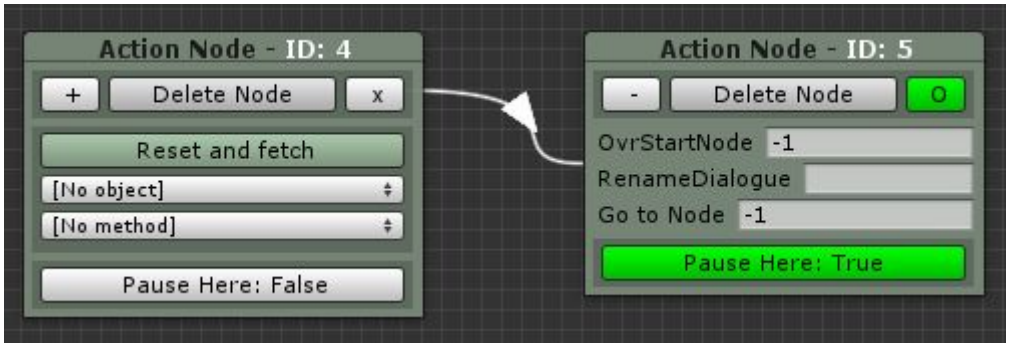
Взгляните на**ExtraVariablesLookUp** метод в**примерUI.cs** для справки о том, как обрабатываются дополнительные переменные.

Вы можете изменить дополнительные переменные во время выполнения! Позвонить**VIDE_Data.SetExtraVariables()** и передайте необходимые параметры для установки новых значений. Вы можете изменять только те диалоги,**загружен**, так что убедитесь, что вы**Загрузить диалоги()** первый.
Ознакомьтесь с Scripting API для получения подробной информации об этом методе.

Будь креативным! Вы можете использовать дополнительные переменные, чтобы добавить динамичности своим диалогам. Например, вы можете использовать их для добавления условий в свои диалоги (посмотрите пример PlayerInteraction и поговорите с парнем условий). И, в некоторых случаях, вы захотите передать ряд чисел или слов, которые позже будут использоваться в качестве массива. В таком случае вы можете воспользоваться**ТоИнтАррай** и **Тострингаррай** методы в**ВД** класс, чтобы быстро разобрать строки для вас. Ознакомьтесь с Scripting API для получения дополнительной информации.

3.**Узел действия**

С помощью этого узла вы сможете вызывать различные методы, прикрепленные к объектам вокруг сцены. Это значительно улучшит возможности добавления действий во время разговора. Вы также можете использовать предопределенные действия.



Как пользоваться:

1. На панели инструментов редактора VIDE создайте узел действия, перетащив его в пустое место.
2. Если вы проверите кнопки раскрывающегося списка, вы увидите, что они пусты. Вам нужно нажать кнопку «Сбросить и получить», чтобы вернуть список объектов для выбора методов. Как только вы нажмете кнопку, она сбросит все переменные и выполнит поиск допустимых игровых объектов и соответствующих им MonoBehaviours и методов.**по определенным правилам (см. ниже)**.Таким образом, вы можете контролировать списки даже при редактировании из другой сцены.
3. После извлечения выберите GameObject из списка.
4. Выберите доступный метод из списка ниже.
5. Если метод имеет допустимый параметр, вы получите дополнительное поле для заполнения нужными данными.
6. При желании вы можете добавить дополнительные предопределенные действия, нажав кнопку [+]. Смотрите ниже для получения дополнительной информации.
7. Наконец, у вас есть возможность **Пауза** поток разговора, как это происходит при достижении узла диалога. Если для параметра «Пауза здесь» установлено значение «Истина», вам потребуется вызвать **Следующий()** снова, чтобы продолжить. NodeData не будет изменен. Обычно узлы действий происходят мгновенно и автоматически перемещаются к следующему узлу.
8. Готово! Не забудьте правильно соединить свои узлы и установить идентификатор начального узла! Кроме того, помните, что вы можете ставить их в ряд!

Поговорите с**ведьма** в**пример1** сцена, чтобы увидеть некоторые узлы действий в действии! Также поговорите с**Мистер Куб** чтобы увидеть, как работают предопределенные действия!

Правила поиска метода:

При поиске доступных методов узел действий будет следовать следующим правилам:

а. Только**включено** Будут найдены игровые объекты с присоединенными MonoBehaviours.

б. MonoBehvaiors в определенных пространствах имен будут игнорироваться. *UnityEngine* по умолчанию занесен в черный список. Проверить**черный список** ниже для получения дополнительной информации.

в. Заявленные методы должны быть**общественные пустоты** и иметь**ноль или один параметр** типа**строка, логическое значение, целое число или число с плавающей запятой**.

В списке появятся только те игровые объекты, которые соответствуют вышеуказанным требованиям.

Черный список узлов действий:

Черный список предназначен для фильтрации спам-методов и объектов, которые мы не хотим видеть в списке. Пространства имен**содержащий** любое из ключевых слов, перечисленных в массиве, будет проигнорировано. UnityEngine по умолчанию занесен в черный список, так как в каждом его компоненте есть миллион «валидных» методов. Вы можете редактировать черный список прямо из скрипта VIDE_Editor.cs:

```
49      //Blacklist for namespaces.
50      //For Action Nodes, add here the namespaces of the scripts you don't wish to see fetched in the list.
51      //Any namespace CONTAINING any of the below strings will be discarded in the search.
52      public string[] namespaceBlackList = new string[]{
53          "UnityEngine",
54          //TMP
55      };
```

Закройте и снова откройте редактор VIDE, если вы вносите изменения в список.

GameObjects с таким же именем:

Объекты сцены с одинаковыми именами будут отображаться в списке объектов только один раз. Тем не менее, при вызове выбранного метода во время игры он попытается вызвать его в каждом объекте с таким же именем. Если вы не хотите, чтобы это произошло, рекомендуется не хранить объекты с одинаковыми именами.

Если вы переименуете объект или метод, вам придется снова выполнять Fetch и Reset!

Событие обратного вызова:

Вы также можете подписаться на***OnActionNode*** событие, чтобы добавить еще больше функциональности. Событие вызывается, когда срабатывает узел действия, отправляя идентификатор узла. Вы можете полностью забыть о вызове методов и предопределенных действиях и вместо этого использовать только события, или вы можете использовать их все в своих интересах.

См. API ниже для получения дополнительной информации.

Предопределенные действия:

Нажав кнопку [+] в узле действий, вы сможете отобразить следующие действия:

Переопределить начальный узел: Измените начальный узел диалога, изменив эту переменную. Используйте идентификатор нового начального узла. Если оставить любое число < 0, действие будет проигнорировано.

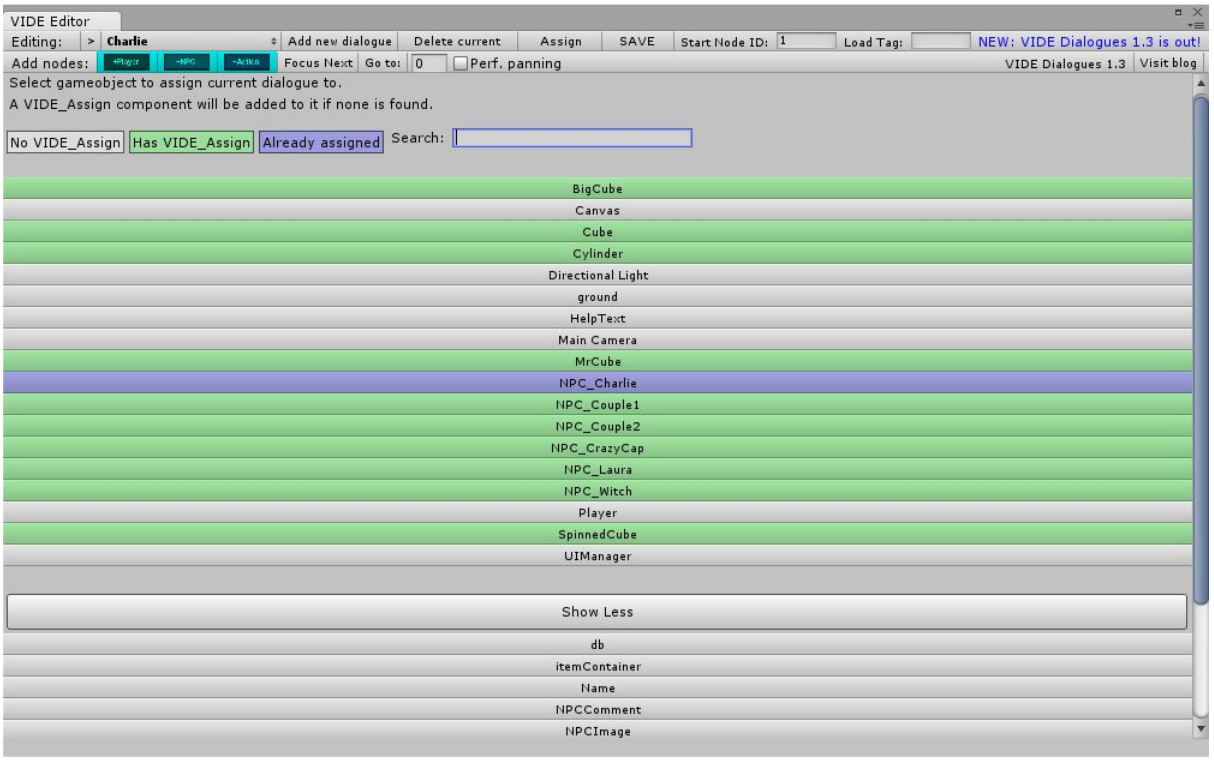
Диалог переименовать: Переименуйте (назначенное) имя диалога. Если оставить пустым, действие будет проигнорировано.

Перейти к узлу: Игнорирует соединение с узлом и переходит к указанному узлу после выполнения действия. Если Pause Here активирована, то она перейдет к узлу после повторного вызова Next() на узле. Оставить на **-1** чтобы не использовать это действие.

Назначить вид

Нажмите кнопку «Назначить», чтобы переключить режим «Назначить вид». Отсюда вы можете назначить текущий диалог игровым объектам, которые вы щелкаете в списке.

Это просто альтернатива ручному присоединению компонента и выбору диалога из выпадающего списка.



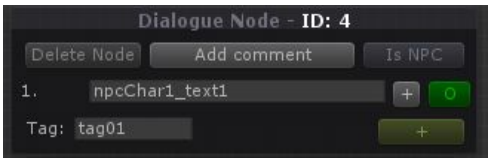
Компонент VIDE_Assign будет создан автоматически, если ни один из них не будет прикреплен к игровому объекту.

Используйте Поиск, чтобы отфильтровать список.

Обычный список будет отображать список активных/включенных объектов в вашей сцене. Появятся только объекты с HideFlags.None.

Нажмите кнопку «Показать больше», чтобы отобразить игровые объекты, которые отключены и/или отсутствуют на сцене.

Встроенная локализация



VIDE уже совместим с системами локализации, работающими с ключами. Вместо того, чтобы вводить фактический текст в узлы, вместо этого вы будете вводить ключи.

Впоследствии, во время игры и перед установкой текста в пользовательский интерфейс, вы должны получить текст, связанный с ключом, с помощью ключа. Большинство плагинов локализации, таких как I2, имеют метод, который возвращает перевод:

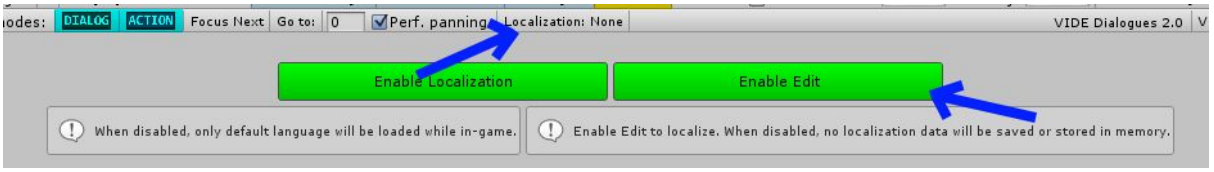
I2.Loc.ScriptLocalization.Get(строковый ключ).

Но, начиная с версии 2.0, VIDE включает встроенную систему локализации, которая может оказаться немного удобной. В отличие от других локализаторов, разница здесь в том, что вам не нужно управлять ключами, и вы можете ввести свой локализованный**текст**,**с прайс**,или**аудио** непосредственно на узлах без создание нового диалога.

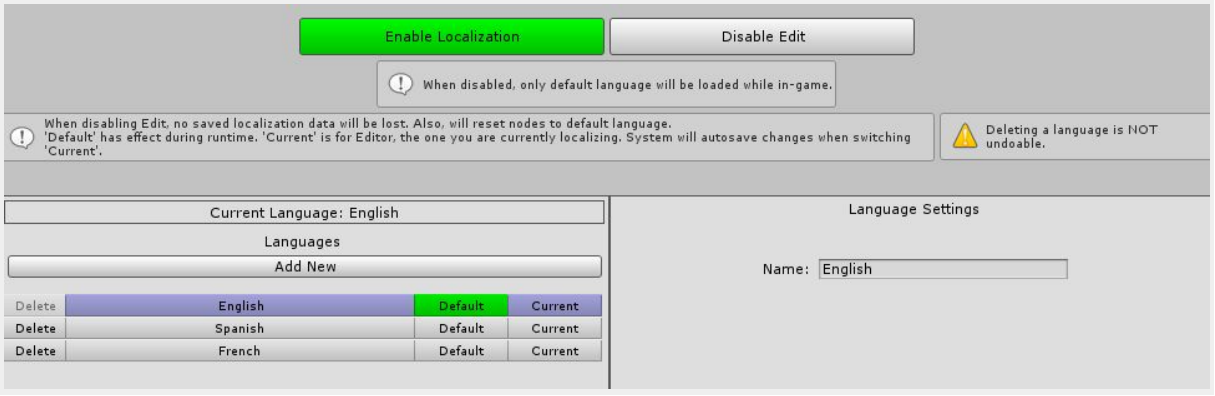
Начиная с 2.0, это первая итерация системы. Улучшения и функции, конечно, будут добавлены в будущем, как только я получу больше отзывов.

Как это работает?

Когда вы активируете «Редактировать» в меню «Локализация», вы сможете добавить столько языков, сколько необходимо:



Пока включен режим редактирования, данные о языке и локализации будут храниться как в памяти, так и в файле .json.**для этого диалога.**



Когда вы создаете и называете новый язык, нажимая «Добавить новый», система сохраняет копию локализованных ключей для этого языка в файл .json, расположенный внутри *ВИДЕО/Ресурсы/Локализовано* . И он будет делать то же самое для каждого языка, все в один и тот же файл.

Убедитесь, что вы правильно называете свои языки!

Что тогда?

Ключи автоматически обрабатываются системой, так как данные уже привязаны к структуре узла. Все, что вам нужно сделать, это выбрать**По умолчанию** и**Текущий** языки.

По умолчанию: Язык по умолчанию, который будет загружен**ВД** во время выполнения. И язык по умолчанию, который будет установлен при отключении редактирования.
Текущий: Текущий язык, который вы редактируете в VIDE Editor.

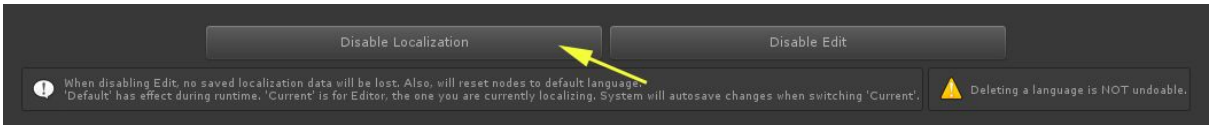
Когда вы нажимаете**Текущий**,VIDE сначала сохранит ваши последние изменения локализации, а затем переключит редактируемый в данный момент язык. Как только вы это сделаете, вы можете закрыть меню локализации и отредактировать свои узлы с локализованными данными.

В то время как вы редактируете другой язык, вы редактируете точно такие же узлы, которые содержат точно такие же дополнительные переменные и настройки, только**комментарии, теги, спрайты,** и **аудио** будет различаться между языками. Вы даже можете продолжать строить свой диалог на другом языке, добавляя узлы, удаляя их и т. д., и другие языки будут синхронизированы.

- При каждом сохранении изменения локализации также сохраняются для текущего языка.
- Когда вы отключите редактирование, текущий язык будет переключен обратно на язык по умолчанию. Никакие новые данные локализации не будут записаны и сохранены.
- Вы будете**нет** потерять все данные при отключении редактирования.
- Единственный способ стереть данные локализации — либо удалить соответствующий диалог, либо удалить язык, а затем сохранить или переключить текущий.
- НЕ удаляйте/не изменяйте вручную файл .json в папке Localized. То же самое касается основных файлов диалогов в папке Dialogues.
- Вы можете скопировать узлы локализации с языка по умолчанию на текущий язык. Просто нажмите «Копировать локализовано по умолчанию» в языковых настройках.

Я задолбался. Что теперь?

Включите локализацию, чтобы она загружала данные в память во время игры. Эта опция не будет иметь никакого эффекта при редактировании, только во время выполнения.



Когда вы запускаете игру и звонит**VD.LoadDialogues**о р**VD.BeginDialogue**,диалоги будут загружаться на языке, установленном по умолчанию, который также является**текущий язык**.

Чтобы изменить язык во время игры, вы звоните**VD.SetCurrentLanguage** и укажите название языка. Это изменит**VD.currentLanguage.** Любые загруженные диалоги будут перезагружены на новом языке. Кроме того, активный VD.nodeData будет обновлен локализованными данными.

- Доступные языки можно узнать, позвонив по телефону**VD.ПолучитьЯзыки().** Он вернет массив строк с именами.

- Используйте**VD.OnLanguageChange** событие для обновления вашего текущего диалога, если это необходимо.

- Для простой демонстрации системы см.**пример3** сцена входит в комплект вместе с **UIManager2.cs**.

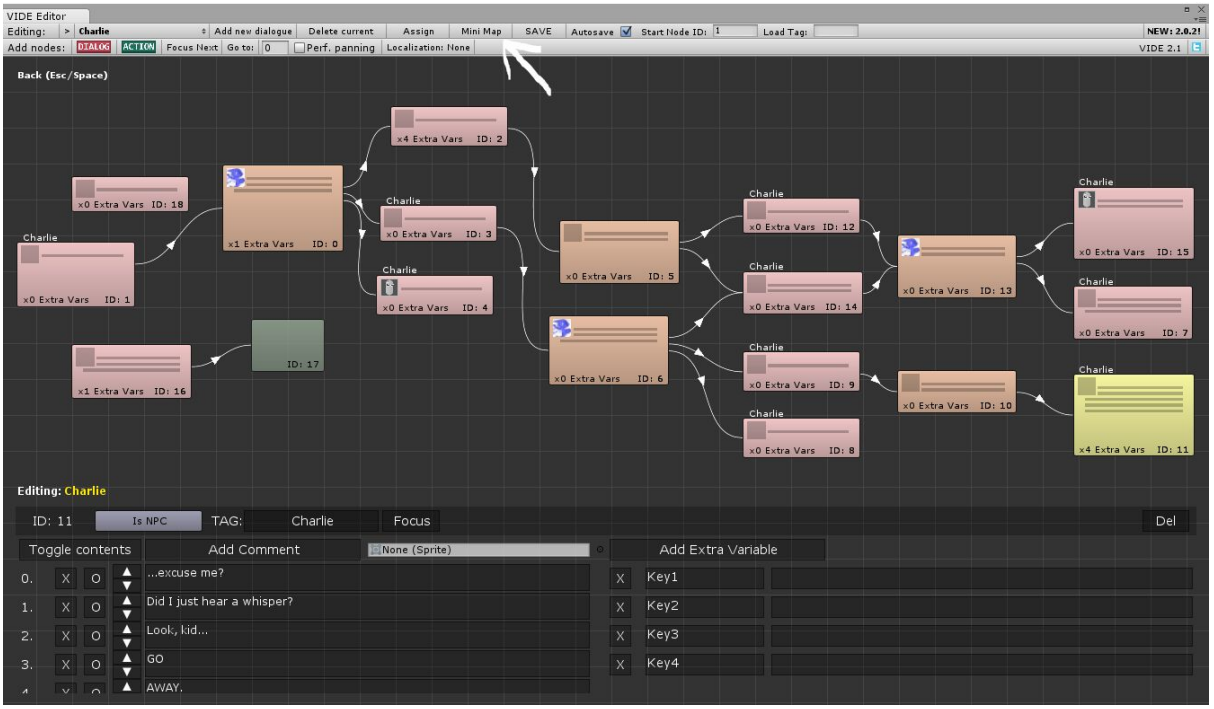
- Для экземпляров VD2 вам придется перезапустить активный диалог, чтобы он обновил язык.

- Убедитесь, что вы также проверили Scripting API.

Если у вас есть какие-либо проблемы, предложения или вопросы, пожалуйста, не стесняйтесь обращаться ко мне.

Мини-карта

Мини-карта — это новая функция версии 2.1. Это улучшает навигацию, видимость и производительность, предоставляя пользователю картографическую версию диалога.



Чтобы войти в мини-карту, нажмите кнопку «Мини-карта» на панели инструментов или нажмите **Космос** пока не внутри текстового поля.

На мини-карте вы можете:

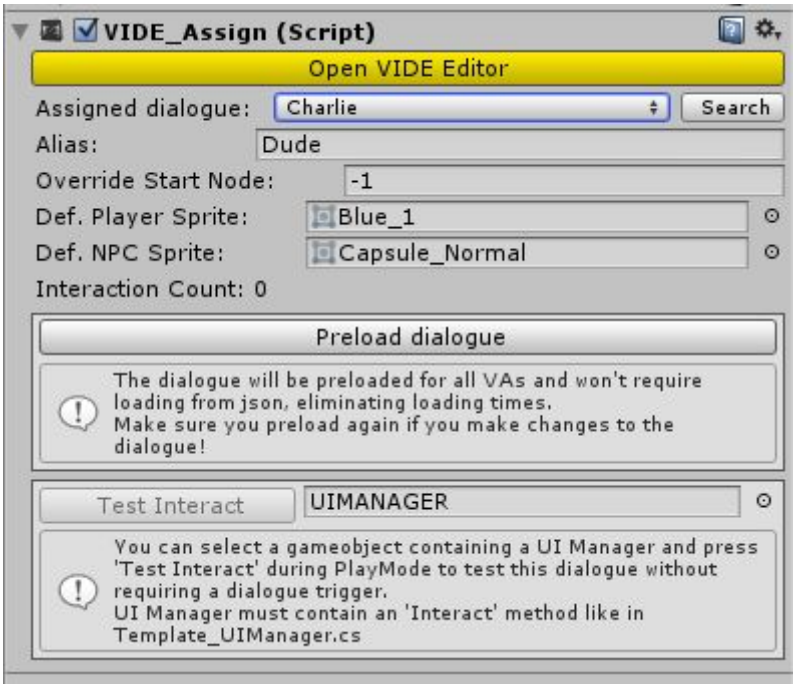
- Получите полную карту вашего диалога и быстро просмотрите содержимое узлов: спрайт, количество комментариев, EV и идентификатор.
- Быстро перейти к другому узлу в диалоге.
- Редактировать содержимое выбранного узла.
- Уничтожить узлы.

На мини-карте нельзя:

- Перемещение, подключение/отключение узлов.
- Создание узлов.
- Перетащите или масштабируйте карту.

ВИДЕО Назначить

Вы прикрепляете этот компонент к игровым объектам, с которыми хотите взаимодействовать. Каждый VIDE_Assign может содержать один диалог.



Откройте редактор видео: Запускает VIDE Editor с назначенным диалогом.
Назначенный диалог: Диалог, назначенный этому компоненту. **Псевдоним:** Альтернативное название диалога.

Переопределить начальный узел: Вместо этого диалог начнется на узле с указанным идентификатором. -1 не использовать.

Спрайт игрока по умолчанию: Спрайт по умолчанию для узлов Player.

Спрайт NPC по умолчанию: Спрайт по умолчанию для узлов NPC.

Количество взаимодействий: Увеличивается каждый раз, когда активный назначенный диалог достигает своего конца.

Предварительная загрузка: Предварительно загрузите данные диалога, чтобы исключить необходимость загрузки из файла json во время выполнения, устраняя любые возможные задержки загрузки больших диалогов.

Тестовый диалог: Запустите диалог, не требуя триггера. Нужен собственный скрипт диспетчера пользовательского интерфейса и метод «Взаимодействия», подобный тому, который находится в Template_UIManager.cs.

Во время активного диалога эти переменные могут быть доступны через **ВД.назначено**.

использование

---

1. В Unity перейдите к*Окно > Редактор видео* чтобы открыть редактор. Вы также можете открыть его из компонента VIDE_Assign.
2. Нажмите «Добавить новый диалог». Назовите новый файл и нажмите «Создать». Допустимые символы: **az, AZ, 0-9, _$#&**
3. Перетащите узел диалога из значков панели инструментов.
4. Нажмите и удерживайте кнопку «Ссылка», затем перетащите курсор в пустое место и отпустите. Это создаст соответствующий новый узел и автоматически подключит к нему текущий узел. Вы также можете отпустить, находясь поверх другого узла, чтобы соединить их. Если вы хотите подключить узел NPC к другому узлу NPC, используйте «Добавить узел NPC», чтобы создать его, а затем соедините узлы. То же самое касается узлов действий.
5. Продолжайте строить диалог. Любой отключенный комментарий или узел действия будут устанавливать конец разговора.
6. Убедитесь, что вы указали идентификатор начального узла на панели инструментов с существующим идентификатором узла.
7. Сохраните диалог, нажав кнопку «Сохранить».
8. Нажмите кнопку «Назначить» в редакторе. Откроется представление «Назначить». Выберите игровые объекты, которым вы хотите назначить диалог. По желанию можно просто прикрепить **VIDE_Assign** компонент вручную и выберите назначенный диалог из раскрывающегося списка в Инспекторе.
9. При желании вы можете настроить псевдоним, overrideStartNode и спрайты по умолчанию для **VIDE_Assign** составная часть.
10. Иметь игровой объект с **VIDE_Data** компонент прилагается. VIDE_Data отвечает за обработку всех диалогов во время выполнения.
11. Наконец, создайте или добавьте скрипт, который будет управлять всеми вещами, связанными с пользовательским интерфейсом, в желаемой игре. объект. Помните, что вы можете использовать**примерUI** сценарий, который поставляется с этим ресурсом в качестве начальной точки. Проверьте**Как это устроено** главу для более подробного объяснения.

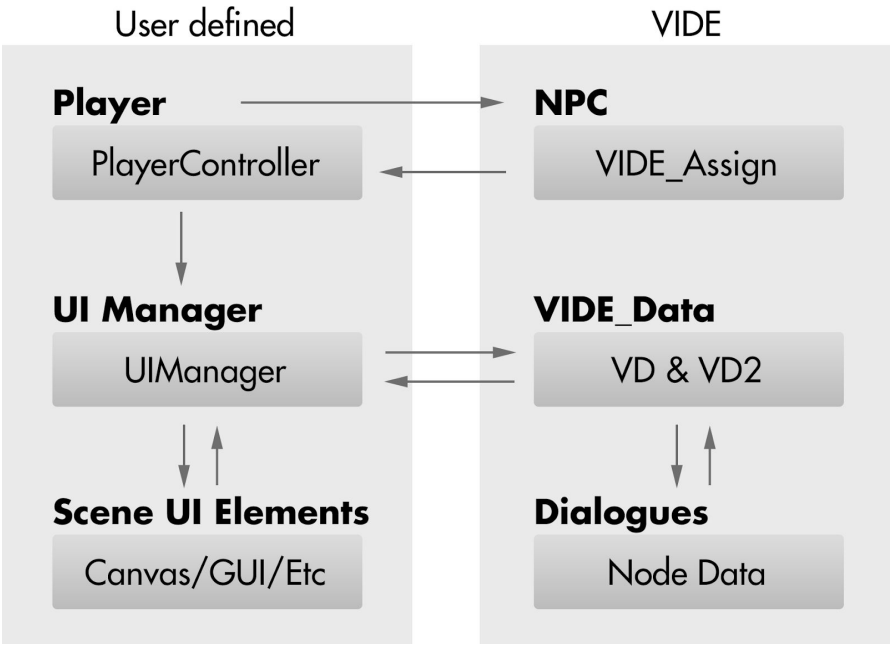
Советы и полезная информация:

- Назовите свои диалоги по определенной номенклатуре, чтобы легко отфильтровать их в диалоговом поиске.
- Хит**Войти** пока ни один узел не будет выделен, чтобы войти в диалоговый поиск и выбрать другой диалог для редактирования. Полезно, когда у вас много диалогов.
- Хит**Esc** для выхода из назначенного представления или диалогового поиска.
- Хит**Космос** для входа на мини-карту.
- Дублировать узлы по **щелчок правой кнопкой мыши** и**перетаскивание**.
- Держи**Сдвиг** нажимая узлы, чтобы выбрать несколько из них, а затем перетащите их вместе.
- Щелкните правой кнопкой мыши пустое место, чтобы создать новый узел диалога.
- Отпустите линию соединения на пустом месте, чтобы создать новый узел и подключиться автоматически.
- Вы можете иметь более одного разговора в одном файле диалога, но помните, что у вас может быть только одна начальная точка по умолчанию. Чтобы начать с другого узла, используйте метод SetNode() в начале диалога или установите**Переопределить начальный узел** поле в Инспекторе для **VIDE_Assign** компонент к идентификатору узла. Проверьте пример 1 **примерUI** script для справки о том, как этого можно достичь.

Как это устроено

Когда вы начинаете разговор с**VD.Начать диалог()**,вы отправляете**VIDE_Assign** к **ВД**. **ВД**'s методы и переменные**публичный статический** и к ним можно получить доступ/вызвать отовсюду.

Вы также можете начать разговор, указав имя. В этом случае временное скрытое VIDE_Assign будет автоматически создано и связано с диалогом, но это оставляет вам очень ограниченные функциональные возможности и не рекомендуется.



Пример блок-схемы общения в игре от третьего лица

ВД Компонент содержит переменную с именем**nodeData**.Эта переменная типа**NodeData** хранит, как вы можете себе представить, все данные текущего узла. В начале разговора,**текущий узел** совпадает с начальным узлом, выбранным вами в редакторе VIDE (если вы не установили переопределение начального узла). Когда вы звоните**Следующий()** метод, разговор пойдет на один шаг вперед в соответствии с вашей структурой узла. Затем nodeData будет заполнен данными из следующего узла, который теперь будет новым текущим узлом. Обратите внимание, что узлы Action не будут изменять nodeData.

По сути, вы постоянно читаете содержимое `VD.nodeData` делать все, что хочешь в вашем диалоговом интерфейсе, используя его данные. Вы можете добиться этого лучше, подписавшись на доступные события, такие как**Оннодечанже**.Вы также можете использовать переменные VIDE_Assign. Во время активного разговора во время выполнения получите доступ к компоненту VIDE_Assign через **ВД.назначено**.

Основной рабочий процесс кодирования:

1. Пользовательские звонки ***VD.Начать диалог()*** **м**етод на**ВД** начать разговор с НПС. Метод требует, чтобы пользователь отправил**VIDE_Assign** компонент, прикрепленный к игровому объекту**или** имя диалога. Это заполнит***VD.nodeData*** с данными с первого узла, с которого начинается диалог.

2. Пользователь использует данные в **VIDE_Data.nodeData** настроить интерфейс внутриигрового диалога. Вы можете сделать это с помощью собственных методов или с помощью событий обратного вызова.

3. Пользовательские звонки ***ВД.Далее()*** на заселение *ВД* .nodeData с данными от следующего узла в разговоре. Для узлов Player система будет считывать **nodeData.commentIndex** переменная, чтобы знать, куда идти.

4. Пользователь использует новые данные в *VD.nodeData* настроить интерфейс внутриигрового диалога.

5. Шаги 3 и 4 повторяются до тех пор, пока пользователь не прочитает ***nodeData.isEnd*** переменная или ***VD.OnEnd*** мероприятие знать, когда звонить ***ВД.Конец Диалога()*** метод и очистите диалоговый интерфейс.

Очень важно, чтобы вы проверили**Справочник по API сценариев** следующую главу, чтобы понять методы и содержание**NodeData** класс. Как только вы познакомитесь с предлагаемыми переменными и методами, вы узнаете, как включить их в свой сценарий пользовательского интерфейса.

Вам не нужно начинать с нуля!

Помните, что VIDE уже поставляется с несколькими демонстрационными примерами и сценарием шаблона с полными комментариями. **Template_UIManager** который взаимодействует с VIDE_Data. Вы можете использовать его в качестве отправной точки и понять принцип работы. Не стесняйтесь изменять его и расширять. VIDE предоставит все данные, необходимые для его работы. Учтите, что для вашей игры могут потребоваться более простые или более сложные менеджеры. Это зависит от вас!

Подписатьсяэто учебник по написанию сценариев, чтобы действительно понять, как настроить базовую систему с нуля!

Загрузка/выгрузка диалогов

Когда ты**Начать диалог()**,**ВД** попытается загрузить данные диалога с диска**если** это не уже загружен.

По мере роста проекта вы можете столкнуться с сотнями диалогов с десятками узлов в каждом. Когда вы видите задержку при начале разговора, это, вероятно, связано с тем, что уже загружено много данных. В этом случае было бы лучше, если бы вы загрузили диалог заранее (например, при загрузке сцены), чтобы при взаимодействии с NPC диалог был уже загружен, без задержки.

Вы можете сделать это, позвонив**Загрузить диалоги()** и**Выгрузить диалоги()** методы на**ВД**.Диалоги останутся загруженными в компоненте. Если он будет уничтожен, вам придется снова загружать диалоги. См. Scripting API ниже для получения дополнительной информации о методах.

Вы вызываете эти методы только на VD. VD2 захватит данные диалога с VD.

*Примечание: диалоги**нет** выгружается при завершении разговора.*

Начиная с версии 2.2, вы также можете предварительно загружать свои диалоги из компонента VIDE_Assign. Это устранит задержки за счет небольшого дополнительного выделения памяти.

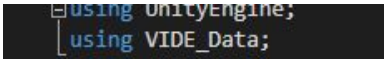
Справочник по API сценариев

---

namespace

Пространство имен ВИДЕО_ Данные

Не забудьте импортировать это, чтобы легко получить доступ к классу VD.



публичный класс VD

Этот класс содержит все статические переменные и методы, необходимые для обработки данных диалога. Ваш сценарий диспетчера пользовательского интерфейса связывается с этим классом для получения данных.

публичный класс VD2

Это то же самое, что и VD, но в нем нет статических членов. В отличие от VD, вы можете использовать его для создания нескольких экземпляров и управления несколькими диалогами одновременно. Некоторые элементы являются эксклюзивными для VD, например методы, связанные с загрузкой/локализацией. Если вы используете VD2, вы все равно можете получить к ним доступ через VD.

Для обоих классов вы можете вызывать и получать доступ к следующим переменным, методам и событиям. Обратите внимание, что статические элементы применяются только к виртуальному диску.

Functions:

общедоступные статические данные BeginDialogue(Name диалога, VIDE_Assign диагностика для загрузки);

Инициирует только что отправленный диалог. Пытается загрузить его с диска, если не загружен. Также заполняет nodeData данные узла переменная с первым узлом на основе начального узла. Также возвращает nodeData. Данные узла пакет. Если первый узел является узлом действия, nodeData будет нулевым, пока не достигнет узла диалога.

(Перегрузка) общедоступные статические NodeData BeginDialogue(Name диалога, VIDE_Assign диагностика для загрузки);

Инициировать диалог, указанный по имени. Временный VIDE_Assign будет связан с диалогом, пока он остается загруженным. У него будет по умолчанию назначенная переменная. Измените те же самые переменные, которые вы изменяете из Инспектора, вызвав SetAssigned. Имейте в виду, что при использовании этой перегрузки функциональность ограничена.

общедоступная статическая строка[] GetDialogues(); получить диалоги(); только ВД. Возвращает список созданных диалогов.

общественное статическое целое GetNodeCount(GetNodeCount(логическийвключить узлы действий); только ВД. Возвращает количество узлов в активном диалоге.

общественная статическая пустота EndDialogue();

Сбрасывает все временные данные. Повышает значение interactionCount количество взаимодействий прежде чем отвязать его. Не вызывайте BeginDialogue() снова, если вы еще этого не вызывали. Это не выгрузит диалог из памяти.

общедоступные статические данные узла Next();

Заполняет данные узла данными от следующего узла на основе текущего данные узла. Если текущий данные узла принадлежит узлу игрока, убедитесь, что nodeData.commentIndex nodeData.commentIndex правильно установлен перед вызовом Next(). Если текущий nodeData принадлежит узлу NPC с несколькими комментариями, вызов Next() будет продвигаться по этим комментариям, прежде чем перейти к следующему узлу.

Будет вызывать OnNodeChange событие.

Вызов Next() для отключенного комментария установит isEnd значение true и не будет получать никаких новых данных nodeData. Прочитайте эту переменную, чтобы вызвать EndDialogue(), или прослушайте событие OnEnd.

Также возвращает текущий Данные узла пакет.

общедоступные статические данные узла SetNode(Установить текущий личности););

Игнорирует текущий данные узла состояние и переходит непосредственно к указанному узлу, будь то диалог или узел действия. Убедитесь, что идентификатор существует.

общедоступные статические данные узла Получить следующий(логическийreturnСледующийКомментарий , больcallAction););

Имитирует метод Next(). Возвращает пакет NodeData следующего узла на основе текущего nodeData. Не будет запускать никаких событий. Не будет изменять текущие данные nodeData. Возвращает текущие данные nodeData, если next имеет значение null или если это ActionNode.

общедоступные статические данные узла Получить данные узла(личности););

Возвращает пакет NodeData указанного узла в активном диалоге. Это не влияет на текущее развитие диалога. nodeData останется прежним.

общедоступные статические данные узла Получить данные узла(логДанные, интудостоверение личности,логическийпринудительная загрузка););

Возвращает пакет NodeData указанного узла в указанном диалоговом окне. Вы можете использовать его с неактивными диалогами, которые загружен. если они не загружены, а вам все еще нужны NodeData, убедитесь, что вы установили для forceLoad значение true.

общедоступная статическая строка Получить первый тег(логическийSearchPlayer););

Возвращает первый найденный тег. Не соответствует структуре узлов. Если searchPlayer имеет значение false, то будут выполняться поиски узлов NPC.

общедоступный статический интервал[] ToIntArray(строкастрока для преобразования) ; **);**

только ВД. Преобразование строки в массив целых чисел. Это полезно при работе с некоторыми электромобилями. Символы, разрешенные для операции разделения: ',', '-', '_', ' '

общедоступный статический интервал[] ToStringArray(строкастрока для преобразования); **);**

только ВД. Преобразование строки в массив строк. Это полезно при работе с некоторыми электромобилями. Символы, разрешенные для операции разделения: ',', '-', '_', ' '

общественная статическая пустота LoadDialogues(); диалоги();

только ВД. Загружает все диалоги в память.

(Перегрузка) общественная статическая пустота oadDialogues(строка **);**

имя_диалога); только ВД. Загружает нужный диалог(и) в память.

общественная статическая пустота UnloadDialogues();

только ВД. Выгружает все диалоги из памяти.

общественная статическая пустота SaveState(строкаимя_файла); **);**

Сохраняет текущее состояние VD, включая измененные EV и видимость комментариев. При желании сохраните каждое состояние VA, найденное в текущей сцене; они будут сохранены под именем*gameObjectName_state* . Сохраненные данные хранятся внутри VIDE/saves. Работает так же для VD2.

общественная статическая пустота oadState(строкаимя_файла); **);**

Загружает состояние виртуального диска. При желании загрузите состояние для виртуальных машин, найденных в текущей сцене, сохраненных под именем:*gameObjectName_state* . Сохраненные данные хранятся внутри VIDE/saves. Работает так же для VD2.

общественная статическая пустота SetExtraVariables (строкадиалогИмя, интидентификатор узла, Словарь<строка, объект> новые переменные); **);**

Обновляет Дополнительные переменные желаемого диалога и узла новым содержимым. Он изменяетзагружен loaded logue, а не NodeData, поэтому изменения сохранятся, пока диалог остается загруженным. В случае успеха и если диалог в данный момент активен, устанавливает для NodeData.dirty значение true. Обратитесь к exampleUI.cs для демонстрации.

(Перегрузка) public static void НаборE xtraVariables (интидентификатор узла,Словарь<строка, объект> новые переменные); **);**

Обновляетактивный active ительные переменные узла диалога. В случае успеха устанавливает для NodeData.dirty значение true. Обратитесь к exampleUI.cs для демонстрации.

общедоступный статический словарь<строка, объект> Получить**GetExtraVariables**ные(строкадиалогНазвание , инт идентификатор узла)

Получите словарь с дополнительными переменными узла в загруженном диалоге. Полезно для слежки за диалогами, которые в данный момент не активны.

(Перегрузка) public static Dictionary<string, object> ПолучитьE xtraVariables(интидентификатор узла) Получите словарь с дополнительными переменными узла в текущем активном диалоге.

общественная статическая пустота **SetComment**(строкадиалогНазвание , интидентификатор узла, инткомментарийИндекс , строкановый комментарий)

Обновите комментарий узла новым комментарием. Это перезапишет локализованное значение.

общедоступный статический VIDE_Assign **GetAssigned**(строка диалоговое имя)

Получает временный VIDE_Assign диалога (не компонент). Используйте VD.assigned, чтобы получить текущий назначенный компонент VIDE_Assign.

общественная статическая пустота **SetAssigned** (строкадиалогНазвание , строкапсевдоним, интовр, Спрайт playerSprite , Спрайтнпссприйт)

Устанавливает временные переменные VIDE_Assign для данного диалога.

общественная статическая пустота **SetVisible**(строка диалогНазвание ,инт идентификатор узла,инт комментарийИндекс ,логический видимый)

Устанавливает видимость комментария узла. Диалог не должен быть активным. Когда комментарий невидим, он не будет включен в Nodedata. этотвключает скрытые **includes hidden comments**, поэтому переданный индекс должен учитывать скрытые комментарии.

(Overload) (Перегрузка) общественная статическая пустота **SetVisible**(инт идентификатор узла,инт комментарийИндекс ,логический видимый)

Устанавливает видимость комментария узла в активном диалоге. Когда комментарий невидим, он не будет включен в Nodedata. Не включает скрытые комментарии. Индекс передан согласно имеющимся комментариям.

общественная статическая пустотаSetCurrentл язык (строкаязык)

только ВД. Переключить текущий язык. Автоматически перезагрузит любые загруженные диалоги. Убедитесь, что имя языка существует.

общедоступная статическая строка[] **GetLanguages**()

только ВД. Получите список доступных созданных языков. Добавьте их из меню «Локализация» в редакторе VIDE.

Е вентиляционные отверстия:

общедоступное статическое событие ActionNode; Узел для чтения

Вызывается при срабатывании узла действия. Отправляет ID узла. Методы подписки с одним параметром типа int.

публичный статический LoadUnload OnLoaded;

Вызывается после завершения загрузки диалогов.

публичный статический LoadUnload OnUnloaded;

Вызывается после окончания выгрузки диалогов.

публичный статический NodeChange OnNodeChange;

Вызывается при изменении каждого узла (только Player и NPC). Обратитесь к событию OnActionNode для узлов действий.

публичный статический NodeChange OnEnd;

Вызывается, когда мы пытались вызвать Next для отключенного узла/комментария.

публичный статический NodeChange OnLanguageChange;

Вызывается после завершения установки текущего языка.

Note: Примечание: события OnEnd и OnNodeChange для класса VD2 будут возвращать значение VD2. объект вместо NodeData. Это полезно, чтобы найти индекс экземпляра внутри список.

Variables:

Переменные

общественное статическое логическое isLoaded;

загружен; Есть ли активный диалог?

общественное статическое int startPoint; (только для чтения)

Идентификатор начального узла по умолчанию, установленный в редакторе VIDE. Возвращает 0, если диалог в данный момент не загружен.

публичный статический VIDE_Assign assigned;

Ссылка на загруженный в данный момент файл VIDE_Assign. Переменная имеет значение null, если диалог в данный момент не загружен.

общедоступные статические данные узла nodeData;

Переменная, содержащая все текущие данные Node, необходимые для настройки диалогового интерфейса. Переменная имеет значение null, если диалог в данный момент не загружен.

публичный статический Sprite[] spriteDatabase;

только ВД. Передайте массив спрайтов в эту переменную, чтобы избавиться от необходимости в папке Resources.

публичный статический AudioClip[] audioDatabase;

только ВД. Передайте массив спрайтов в эту переменную, чтобы избавиться от необходимости в папке Resources.

общественное статическое логическое localizationEnabled; (только для чтения)

только ВД. Локализация включена? Вы можете установить это в меню локализации в VIDE Editor.

общедоступная статическая строка Текущийл язык; (Только для чтения)

только ВД. Текущий набор языков. Установите его, вызвав SetCurrentLanguage().

общедоступная статическая строка По умолчаниюл язык; (Только для чтения)

только ВД. Возвращает имя языка по умолчанию. Вы можете установить это в меню локализации в VIDE Editor.

NodeData

Данные узла класс

Этот класс хранит все соответствующие переменные вашего текущего узла. Обычно это возвращаемое значение многих доступных методов, но к нему также можно получить доступ через VD.nodeData.

общественное int nodeID;

Идентификатор узла; ID текущего узла.

публичный bool isPlayer;

Является ли текущий узел узлом игрока?

публичный bool pausedAction;

Мы сейчас находимся на приостановленном узле действий?

публичный bool isEnd;

Это конец разговора?

общедоступная строка[] comments;

Массив строк со всеми комментариями узла.

общедоступная строка[] extraData;

Дополнительные данные для каждого комментария. Используйте вместе с commentIndex.

публичный спрайт[] sprites;

Спрайт для каждого комментария. Используйте вместе с commentIndex.

публичный аудиоклип[] audios;

Звук для каждого комментария. Используйте вместе с commentIndex.

общественное int commentIndex;

Начинается всегда с 0. Для узлов NPC увеличивается с каждым Next() до достижения последнего комментария. Для узлов Player он действует как выбранный вариант. Установите эту переменную вручную перед вызовом Next() на узлах Player.

публичная строка тег;

Тег, который вы установили для узла NPC в редакторе VIDE.

публичный спрайт спрайт;

Набор спрайтов для этого узла в VIDE Editor.

общедоступный словарь<строка, объект> extraVars;

Дополнительные переменные устанавливаются в редакторе VIDE.

публичный булевый грязный;

Становится истинным, когда вы изменяете дополнительные переменные активного диалога. При следующем взаимодействии он снова будет ложным.

видE_ Назначать

Класс, содержащий информацию о взаимодействии. Компонент для игрового объекта или диалога, в зависимости от того, как вы его используете.

Functions:

публичная строка GetAssigned();

Возвращает строку с именем текущего назначенного диалога.

публичный булевый AssignNew(строкадиалогимя);

Назначьте этому VIDE_Assign другой диалог (только как компонент). Диалог, который вы собираетесь назначить, должен существовать, иначе метод вернет false. Это то же самое, что выбрать его в Инспекторе. Не включайте расширение файла для имени диалога.

общественная пустота SaveState(строкаимя_файла);

Сохраняет текущее состояние VA. Сохраненные данные хранятся внутри VIDE/saves.

общественная пустотал oadState(строкаимя_файла););

Загружает состояние VA. Можно загрузить в любой VA, но будьте осторожны.

Variables:

Переменные:

общественное место количество взаимодействий; **interactionCount;**

Эта переменная начинается с нуля. Каждый раз, когда вы вызываете EndDialogue() для VIDE_Data, когда этот VIDE_Assign в данный момент загружен, InteractionCount будет увеличиваться на 1. В конце концов, он отслеживает, сколько раз вы взаимодействовали с этим игровым объектом, в частности.

публичная строкапсевдоним; **alias;**

Пользовательское имя для этого диалога. Можно установить из Инспектора. (ранее известный как dialogName)

общественное место переопределитьStartNode; **overrideStartNode;**

По умолчанию -1. При изменении начальный узел назначенного диалога будет игнорироваться и вместо этого будет использоваться этот узел. Убедитесь, что идентификатор существует. Это только внутриигровое изменение, оно не изменяет начальный узел фактического диалога. Установите его обратно на -1, чтобы использовать исходный начальный узел. Вы также можете установить его из Инспектора.

публичный спрайт по умолчанию NPCSprite; **defaultNPCSprite;** Спрайт по умолчанию, назначенный для узла NPC.

публичный спрайт по умолчанию PlayerSprite; **defaultPlayerSprite;** Спрайт по умолчанию, назначенный для Playernode.

Список изменений

Версия 2.2.2

- Исправлены ошибки компиляции из-за устаревшего API.
- Исправлено: Редактор не запоминает скин и цвета.
- Исправлено: Пустой редактор при запуске Unity (после выхода с открытым).
- Добавлен VD.GetNodeCount()
- Исправлено: неправильные цвета для полей int и float узла действия.
- Черный список Action Node теперь принимает имена классов, чтобы исключить их из поиска.

Версия 2.2

- Сильно оптимизированный exampleUI.cs (теперь VIDEUIManager1)
- Добавлен чистый новый скрипт Template_UIManager, который можно использовать в качестве отправной точки и для прототипирования.
- Добавлена демонстрационная сцена шаблона для тестирования диалогов с использованием скрипта Template_UIManager.
- Добавлена кнопка справки в VIDE Editor.
- Переименованы некоторые папки и демонстрационные скрипты.
- Добавлен экспорт/импорт файла TXT в VIDE Editor.
- Добавлена опция диалога предварительной загрузки в VIDE_Assign, чтобы исключить необходимость загрузки диалога из json (более высокая скорость загрузки).
- Удалена функция LoadTag. Функция предварительной загрузки заменяет это.
- В VIDE_Assign добавлена функция «Тестовый диалог» для проверки диалогов без необходимости внешнего триггера.
- Добавлена опция «Поиск» в VIDE Assign для назначения нового диалога.
— Добавлен выпадающий список в поле «Тег» в VEditor.
- В основную демонстрационную сцену добавлен демонстрационный NPC TriggerDude.
- Щелкните правой кнопкой мыши, чтобы удалить новый узел диалога.
- Исправлен заголовок новости
- Исправлены неправильные цвета классического скина по умолчанию.
- Исправлено исключение демо-версии локализации.
- Исправлен редкий аргумент вне диапазона при переключении диалогов в VEditor
- Исправлено исключение при перекомпиляции во время режима игры в основной демоверсии.
- Исправлен масштаб трансформации выбора игрока в основной демоверсии.
- Исправлен цвет текста узла действия для раскрывающихся списков.
- Оптимизация
- Обновлены документы

Версия 2.1.1

- Исправлена ошибка, из-за которой SetVisible(int, int, bool) не игнорировал скрытые комментарии при передаче индекса, что приводило к обработке неправильного комментария.
- Исправлено Next() для скрытых комментариев. Не получал правильный индекс комментариев, когда были скрытые комментарии.
- Изменен пользовательский интерфейс индекса комментариев узлов, чтобы он начинался с нуля, чтобы избежать путаницы.
- Обновлена документация для обоих методов SetVisible.

Версия 2.1

- Исправлен доступ к переменным локализации (вы больше не используете «_» перед именами переменных локализации).
- Добавлена мини-карта в редактор VIDE.
- Добавлены методы SaveState() и LoadState() для VD и VA для сохранения и загрузки состояний. Это легко поможет вам восстановить изменения, внесенные в ExtraVariables, перезаписать StartNodes и т. д. между игровыми сеансами.
- В класс VD добавлены методы ToIntArray и ToStringArray. Избавляет вас от необходимости разбивать и анализировать отдельные элементы.
- Добавлено сохранение в основную демку.
- В основную демо-сцену добавлен персонаж «Кондиционер». В нем есть диалог, который работает с условиями.
- Исправлено перетаскивание выбранных узлов при перетаскивании новых.
- Исправлены одиночные числа, позволяющие использовать тысячи (запятые) при анализе дополнительных переменных. Строки, содержащие запятые, останутся строками.
- Добавлены настраиваемые скины редактора.
- Улучшен безье в редакторе VIDE.
- Исправлены утечки памяти Texture2D при использовании VIDE Editor.
- Мелкие исправления.
- Обновлена документация

Версия 2.0.2

- ПЕРЕИМЕНОВАНЫ UpdateComment и UpdateExtraVariables в SetComment и Установить дополнительные переменные.
- Изменить порядок комментариев в редакторе VIDE.
- Выберите и переместите несколько узлов
- Исправлена ошибка, из-за которой GetNodeData(string, int, bool) аннулировал currentPlayerStep во время диалога, что могло возвращать ошибки.
- Новый флаг видимости комментариев. Установите его из инспектора и через метод SetVisible.
— Новые переменные spriteDatabase и audioDatabase для устранения необходимости в папке «Ресурсы».
- Добавлен пример сцены JS.
- Добавлен демонстрационный объект Quest Chart в сцену example1.
- Локализация: исправлены исключения с новыми языками.

- Локализация: добавлено «Копировать локализацию по умолчанию». Скопируйте локализованные данные узла с языка по умолчанию на текущий.
- Обновлен документ

Версия 2.0.1

- Нулевая проверка для метода выборки. Узлы действий будут возвращать исключения при нажатии «Сбросить и получить», если у вас отсутствовали монноповедения, прикрепленные к игровым объектам.
- Исправлена невозможность пристыковать VIDE Editor.
- Теперь вы можете безопасно добавлять подпапки в папку «Диалоги», чтобы упорядочить диалоги.
- Удален автофокус
- Локализация и загрузка/выгрузка переменных и методов удалены из VD2. Используйте VD, чтобы справиться с ними.
- Экземпляр VD2 больше не будет отдельно загружать диалоги в память. Теперь он синхронизирован.
- Добавлена перегрузка GetNodeData для доступа к данным узла из неактивных диалогов.
- Переименован isLoaded в isActive.
- Исправлено дублирование узлов, ссылающихся на дополнительные переменные.
- Пример капитального ремонта сцены 1

Версия 2.0

- Первая версия встроенной локализации.
- Удалены узлы NPC. Теперь это просто узлы диалога и узлы действия. Вы можете переключаться между форматами NPC и Player для диалогового узла.
- VIDE_Data теперь является пространством имен для доступа к классам VD и VD2.
- Теперь вы можете использовать класс VD2 для создания нескольких экземпляров и управления несколькими диалогами одновременно.
- Компонент VIDE_Data в сцене теперь необязателен.
- Компонент VIDE_Assign в сцене теперь необязателен.
- Исправлено множество проблем с системой отмены. Теперь вы можете безопасно отменить удаление и загрузку диалогов.
- Слоты аудио и спрайтов для каждого комментария.
- Теперь вы можете дублировать узлы, щелкнув их правой кнопкой мыши.
- Добавлена сетка и привязка к редактору
- Перемещение узла теперь помечает диалог как грязный.
- Добавлено предопределенное действие «Перейти к узлу» в узел действия.
- Добавлен метод SetComment для VIDE_Data.
- Добавлена перегрузка GetExtraVariables для VIDE_Data.
- Добавлены методы GetNodeData для VD.
- Добавлена перегрузка «BeginDialogues».
- Добавлены методы GetLanguages и SetCurrentLanguage.
- Добавлены «GetAssigned» и «SetAssigned» для управления некомпонентным VIDE_Assign.
- В сцену example1 добавлен еще один демо-персонаж.

- Добавлена сцена примера 3, которая охватывает новый менеджер пользовательского интерфейса и локализацию.
- Изменен режим просмотра редактора для лучшей производительности. Холст больше не бесконечен.
- Исправлена ошибка дублирования идентификатора при ручном дублировании файлов диалога .json.
- Улучшена фокусировка вида узла.
- Автофокус на startNode при загрузке диалога.
- Исправлено некоторое неожиданное поведение при использовании метода SetNode.
- Исправлено некоторое неожиданное поведение при использовании Action Node.
- Исправлен пример exampleUI.cs, позволяющий прокручивать комментарии к узлам NPC и, таким образом, возвращать исключения.
- Добавлена проверка новостей на панель инструментов.
- Добавлен диалоговый поиск в редактор.
- В редактор добавлено меню «Назначить вид». Позволит вам легко проверять и назначать диалоги игровым объектам.
- Исправлен неперетаскиваемый редактор после выпуска узла за пределы окна.
- Полированное обновление инспектора VIDE_Assign.
- PathToVide устарел. VIDE теперь автоматически определяет расположение новой папки в проекте.
- Изменения в переменных NodeData.
- Мелкие исправления и полировка.
- Обновлена документация.

Версия 1.2

- Исправлена загрузка VIDE_Data неправильных диалогов (со смещением) после создания или удаления новых диалогов.
- Новая панель инструментов, которая заменяет Инструменты редактора.
- Добавляйте спрайты в свои диалоги и узлы! Выберите спрайты Player/NPC по умолчанию из компонента VIDE_Assign и/или установите определенный спрайт для каждого узла. Получите доступ к спрайтам через VIDE_Data.assigned.defaultPlayerSprite и NodeData.nodeSprite. Спрайты должны находиться внутри Resources.
- Совершенно новая система дополнительных переменных. Сохраняйте строки, целые числа, числа с плавающей запятой или логические значения в качестве значений. Доступ к объектам из словаря extraVars с помощью пользовательских ключей. Устаревшие дополнительные данные все еще доступны, хотя они будут удалены в более поздней версии.
— Добавлен метод SetExtraVariables для изменения дополнительных переменных во время выполнения.
— В NodeData добавлены переменные «extraVars», «nodeSprite» и «dirty».
- Добавлены кнопки, чтобы развернуть/свернуть содержимое окна на свободное место.
- Добавлен флажок, чтобы отключить представление производительности при панорамировании в редакторе VIDE (пустые узлы при панорамировании).
- Редактирование нескольких объектов Inspector теперь поддерживается для компонента VIDE_Assign.
- Обновлена сцена exampleUI.cs и example1 со спрайтами и дополнительными переменными.
- Обновлена документация.

- Методы и переменные VIDE_Data теперь статичны и доступны из любого места; экземпляр больше не нужен. Вам придется обновить свои скрипты.

— Переменная VIDE_Assign.dialogueName переименована в VIDE_Assign.alias. Возможно, вам придется снова установить имена для каждого VIDE_Assign и обновить свои сценарии. Это было необходимым изменением, поскольку «dialogueName» вводило в заблуждение.

Версия 1.1.3f2

- Добавлены события OnNodeChange и OnEnd в VIDE_Data.
- Полностью рефакторинг exampleUI.cs. Теперь это лучшая отправная точка для новых пользователей.
- Исправлена ошибка, из-за которой редактор VIDE автоматически не открывал вновь созданный диалог.
- Исправлен случай, когда VIDE Editor не сохранялся при закрытии или потере фокуса.
- Исправлены узлы действий для методов без параметров.
- Исправлены новые диалоги, не имеющие начального узла по умолчанию 0
- Оптимизированы узлы, находящиеся за пределами видимого холста, для повышения производительности VIDE Editor.
- Теперь вы можете угрожать Чарли с помощью Таинственной ракетницы (example1.scene)

Версия 1.1.3

- Теперь вы можете загружать и выгружать диалоги в память в любое время, вызывая LoadDialogues() и UnloadDialogues().
- Добавлено событие OnActionNode в узлы действий.
- Добавлены предопределенные действия для узлов действий.
- Обновлено содержимое инспектора VIDE_Data.
- Добавлено *Загрузить тек*ст инструментам редактирования в редакторе VIDE.
- Исправлена ошибка, из-за которой Редактор VIDE не сортировал список после создания нового диалога.
- Мелкие исправления и улучшения.
- Обновлены example1.scene и документация.

Версия 1.1.2

- (Linux) Теперь нажатие кнопки «Открыть редактор VIDE» откроет нужный диалог.
- (Linux) VIDE Editor больше не будет каждый раз открываться в полноэкранном режиме.

Версия 1.1.1

- Дополнительные данные пустого игрока теперь также будут добавлены в массив, чтобы сохранить соответствие индексу.
- Сохраните дополнительную защиту системы от ошибок, возникающих, когда диалоги не найдены.
- Исправлено "ВИДЕО/Ресурсы/Диалоги" на "ВИДЕО/Ресурсы/Диалоги" для Linux.
- Исправлена ошибка, из-за которой exampleUI.cs считывал предыдущие дополнительные данные, находясь на приостановленном узле действия.
- Теперь редактор запоминает последний диалог, который вы редактировали при открытии из редактора Window/VIDE.

Версия 1.1

- Реализованы узлы Action для вызова методов внутри скриптов, включая черный список.
- Добавлены дополнительные поля данных в комментарии игроков.
- Добавлено *playerCommentExtraData* и *pausedAction* в ariables к данным узла.
- Удалены кнопки «Конец здесь». Теперь, *конец* становится истинным при вызове Next() на отключенном узле любого типа.
- Новый метод для VIDE_Data. *ПолучитьПервыйТэг()*
- Теперь вы можете перетаскивать новые узлы на холст.
- Добавлены умные стрелки к линиям соединения.
- Убрано автосохранение и отполирована система сохранения.
- Теперь VIDE будет проверять допустимые символы при именовании диалогов (az, AZ, 0-9, _$#&), чтобы предотвратить ошибки.
- Добавлен новый пример персонажа ведьмы в сцену примера 1.
- Добавлен поиск элемента с заменой имени в сцену example1.
- Оптимизирован и улучшен пример UI.cs (пример UI Manager)
- Исправлена ошибка, из-за которой линии соединения не обновлялись должным образом после подключения узла.
- Исправлена обработка пустых комментариев NPC, например.UI и VIDE_Data.
- Исправлена ошибка, из-за которой окна не перерисовывались при отпускании перетаскивания вне редактора.
- Исправлена ошибка, из-за которой Редактор VIDE не открывался корректно после закрытия Unity, когда он был открыт.
- Исправлены исключения, которые происходили, когда не было создано диалогов
- Улучшен отлов ошибок в VIDE_Data и VIDE_Editor.
- Обновлены некоторые арты.
- Обновлена документация.

Примечание. Не забывайте делать резервные копии текущих диалогов и измененных основных сценариев, если они у вас есть.

Версия 1.0.3

- Исправлена серьезная ошибка, из-за которой сцена не обнаруживала и не сохраняла изменения, внесенные в компонент VIDE_Assign. Таким образом, назначенный диалог не запоминался и возвращал ошибки.

  Спасибо RedDeer и Грегу Мичу за помощь.

Версия 1.0.2

- Теперь файлы будут идентифицироваться по их идентификатору в компоненте VIDE_Assign, чтобы предотвратить смещение индекса файла при создании нового диалога, которое затрагивало каждый созданный диалог.
- Папка диалогов теперь будет обновляться при создании и удалении диалогов.
- Добавлено **метка игрока** переменная в NodeData. Теперь вы также можете назначить тег узлу проигрывателя в редакторе VIDE.
- Обновлена документация.

Примечание. При обновлении до этой версии убедитесь, что в редакторе VIDE вы сохранили все диалоги, которые вы уже создали. Это создаст их соответствующие идентификаторы, чтобы предотвратить автоматическое назначение неправильного диалога.

Версия 1.0.1

- Исправлена проблема с titleContent для добавления поддержки Unity 5.0.
- Исправлена ошибка, из-за которой компонент VIDE_Assign неправильно загружал диалоги при импорте актива.
- Исправлена ошибка, из-за которой тексты NPC не очищались должным образом (ExampleUI.cs).
- Изменен режим масштабирования холста на постоянный размер пикселя для лучшей согласованности между соотношениями сторон.
- Переименованы все классы/скрипты, чтобы предотвратить дублирование определений.
- Убраны некоторые скрипты.
- Обновлена документация

Примечание. При обновлении исходной версии сделайте резервную копию папки диалогов (VIDE/Resources/Dialogues), обновите и замените папку диалогов. Возможно, вам придется снова установить компоненты VIDE_Assign (DialogueAssign).

Версия 1.0

Первый выпуск.