Basic Tutorial

for *ConText – A Text/Choice Adventure Game Framework*

This tutorial will guide you through the interface of the framework and in the process you will create a basic text adventure game.

Part 1 covers the setup of the game engine the framework is using and the importing of the framework files.

Part 2 covers the steps to creating a simple small game with the framework, guiding you through the creation of characters, modules, linking of modules and changing settings regarding the visual appearance of your game.

Parts 3 and 4 contain troubleshooting suggestions and some information about importing (media) files into Unity.

Any further or more detailed topics are covered in the Documentation (Documentation_UserStudy2_English.pdf).

Part 1 – Setup (7 steps)

1. Before anything else, you will need to download and install Unity Personal Edition from here: https://unity3d.com/get-unity/download?ref=personal

   Info: Unity is the game engine this framework was developed in and is to be used in. The first time you start Unity, you will need to create/login with an account. This account is necessary to verify your license information. The account can be created and used free of charge.

   *Note: during installation you will be asked which components are to be installed. If you don't intend to do any custom programming, you can deselect **Visual Studio 2015 (Community)** and you may also deselect any **compilers** for platforms you don't intend to put your game on (e.g. Android/iOS/Windows Phone/WebGL compiler etc.), likely you will only need the Android compiler for initial testing. Especially deselecting Visual Studio will increase installation speed a lot due to reduced download size.*
2. Once you have installed Unity, you will need to either use the ConText package (ConText_UserStudy2.unitypackage) provided in your initial invitation email or download it from here: https://www.dropbox.com/sh/vrao65c3ym3nkhc/AAC_WXLxHcPu04Gkaxne7wh9a?dl=0.
   You may download the entire folder which contains the UnityPackage, a *'ConText Project.wlt'* file as well as all PDFs also attached to the email.
   A Unity Package is a collection of files and objects that can be integrated into Unity. Think of them as a sort of special zip/container file for Unity.

3. Start Unity, log in with your account and select **NEW**. On the following screen, enter a name for your project and select **2D**.

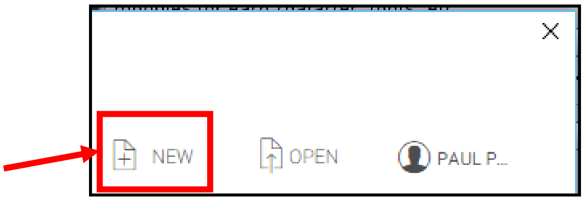
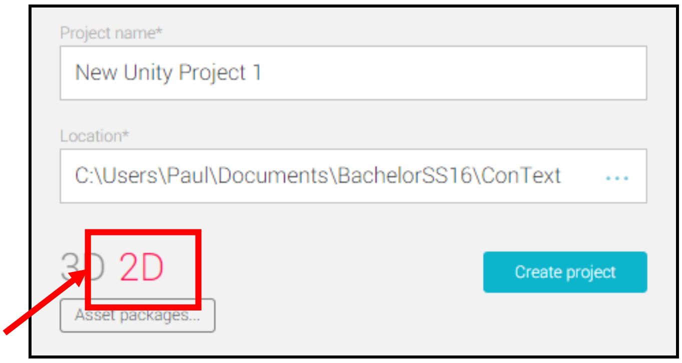


Figure 1: NEW

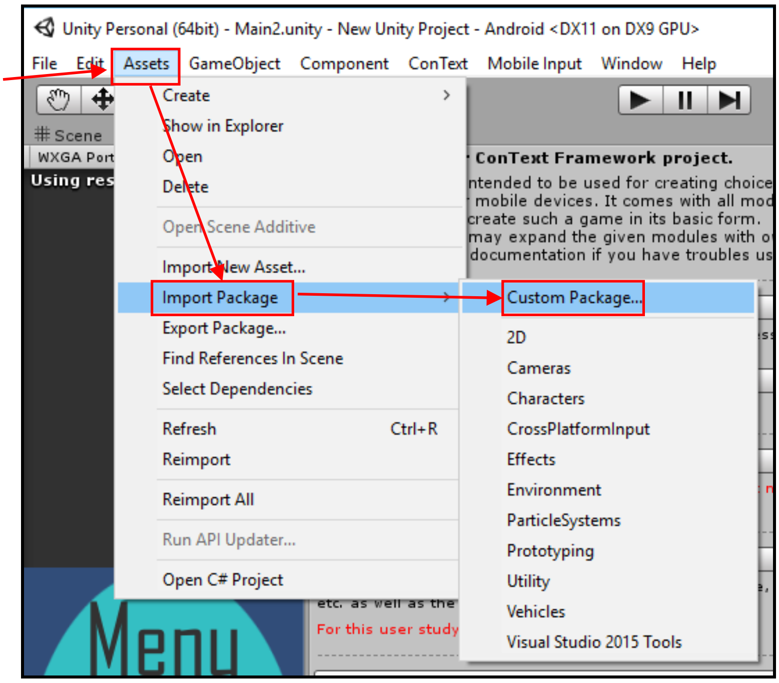


Project name*
New Unity Project 1

Location*
C:\Users\Paul\Documents\BachelorSS16\ConText

Figure 2: Create

4. Import the UnityPackage into Unity through **Assets – Import Package – Custom Package…** . A prompt will ask you to select the file location on your computer. Do so and confirm the choice. A Unity window will pop up listing all items that can be imported from that package. At the bottom of that window, select **All** and then click **Import**.



Unity Personal (64bit) - Main2.unity - New Unity Project - Android <DX11 on DX9 GPU>

Figure 3: Import package

5. Next, you will need *to close Unity again* to add in the last part, the default framework layout. For this, the *'ConText Project.wlt'* file from the original package needs to be copied to the Unity Editor Layout folder. In Windows, that is **%APPDATA%\Unity\Editor-5.x\Preferences\Layouts\**, in Mac OS, it is **~/Library/Preferences/Unity/Editor-5.x/Layouts**

(Copy & paste these directory paths to the top address bar in Windows Explorer/Mac Finder and press Enter to access the respective folder)

Info: This step only needs to be done once per Unity installation (i.e. usually once per computer). Technically you may also manually create the above listed folder and put the .wlt file in it.

6. Almost done setting up the framework. Start Unity again, open your now existing project in the initial screen. Once it is open, in the upper right, select **ConText Project** as the Layout. Ignore the red warning you will now see in the center window.

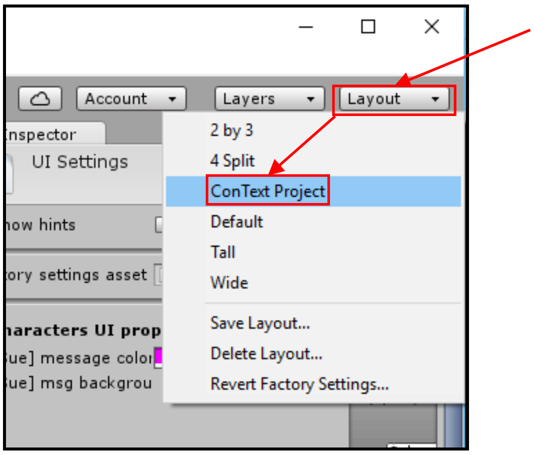


Figure 4: Layout

7. In the bottom Project/Asset folder window, select the top most folder **Assets** and double click **Main** to open the main ConText/Unity scene.

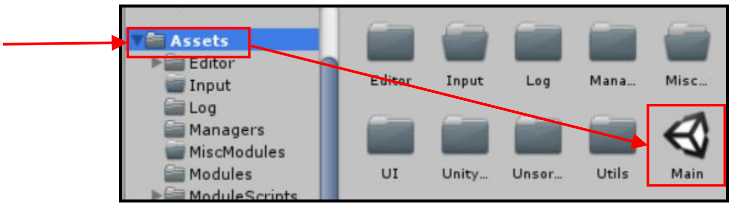
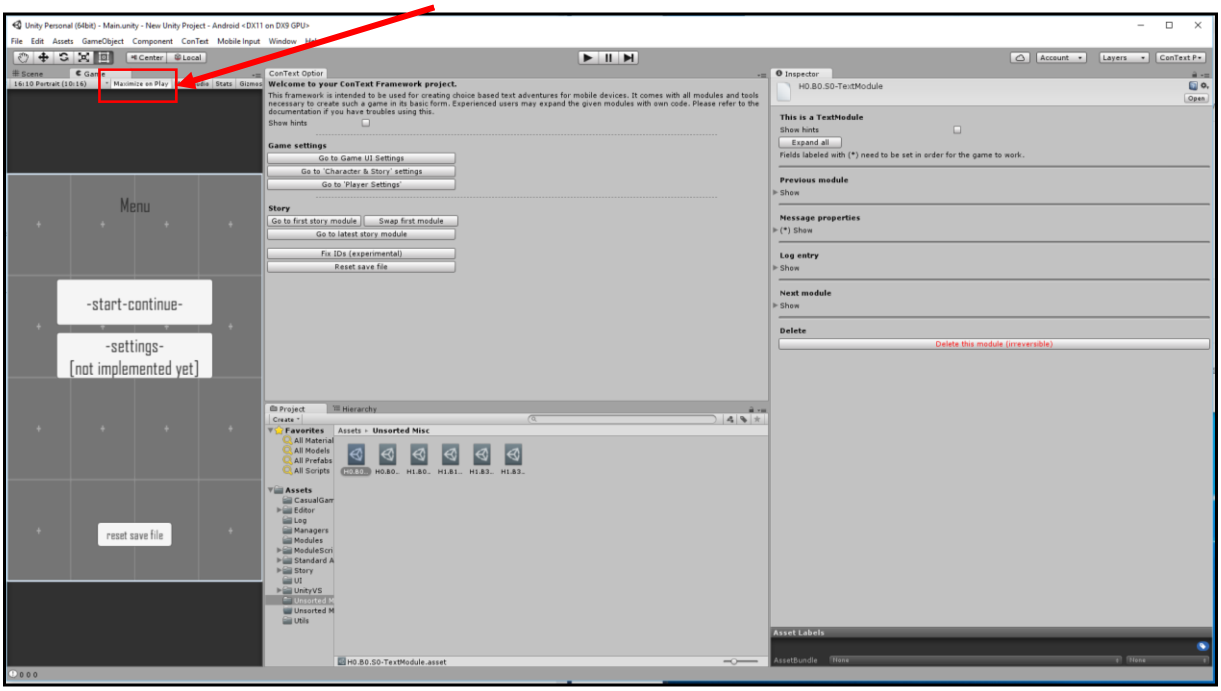


Figure 5: Open Main

Now your Unity editor should look like this:



Unity Personal (64bit) - Main.unity - New Unity Project - Android <DX11 on DX9 GPU>

Welcome to your ConText Framework project.

This framework is intended to be used for creating choice based text adventures for mobile devices. It comes with all modules and tools necessary to create such a game in its basic form. Experienced users may expand the given modules with own code. Please refer to the documentation if you have troubles using this.

Show hints □

H0.B0.S0-TextModule

This is a TextModule

Show hints □

Fields labeled with (*) need to be set in order for the game to work.

H0.B0.S0-TextModule.asset

Figure 6: Overview

If so, the last thing to do is deselect **Maximize on Play** at the top of the left Game preview window, then you are done setting up the framework and can continue with Part 2 of the tutorial. If not, make sure you did all steps as described. If you are sure you did not miss a step and the overview is still wrong, feel free to contact me at paul@preissner-muc.de.

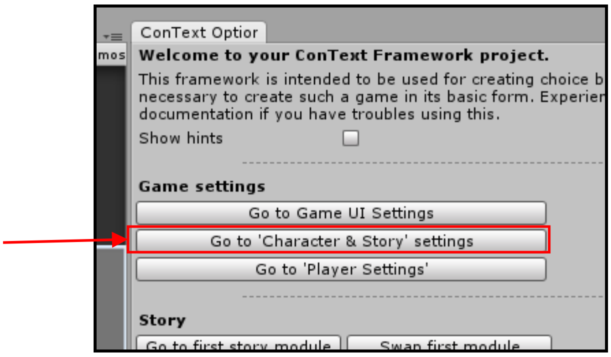
Part 2 – Sample game (10 steps)

In this part you will use the framework to create a simple text adventure game with a few different modules with the framework.

Note: the next steps mention the term "asset" multiple times. Assets in Unity are essentially all actual individual files representing objects. These objects can be or contain images, sounds, program code, GameObjects (i.e. objects that can be put into the actual game world/level), Prefabs (i.e. multiple GameObjects grouped together into one), Fonts, etc. Essentially everything that is permanently saved in files for the game or editor. "Asset" is thus an overarching term for all such files.

1. Given the layout as in Figure 6, the left part shows your *game preview window*. When you test/play your creation, it will be displayed in that window.
   The center part has the *ConText Overview* window in the top half, intended as a sort of hub from which you can go straight to the most important settings and windows. The bottom half is the *Project window*, which displays the directory structure of your asset folders, through which you may directly access story modules, settings files or ultimately script files.
   The right part is the *Inspector window*, which will always display the properties/details of the currently selected asset.

2. To get started, you will create the characters of your story. For this, click **Go to 'Character & Story' settings** in the *ConText Overview* window. This will open the Story Settings asset in the Inspector window. There, click **Add a character**, which will create a new Character asset and automatically display its details in the Inspector window. You may then give it a name and a unique character ID. Add as many characters as you need.



ConText Optior

Welcome to your ConText Framework project.

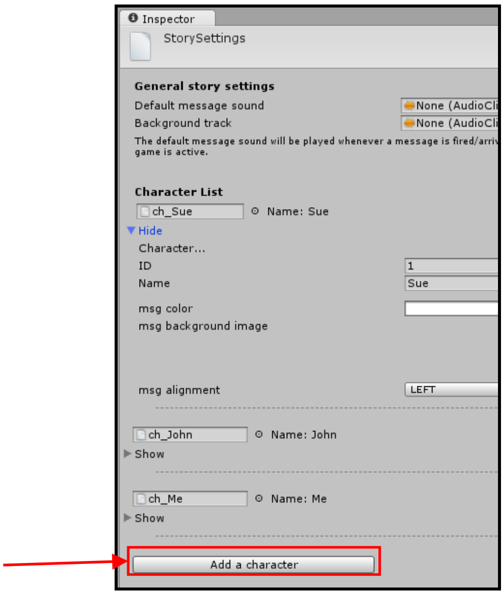
This framework is intended to be used for creating choice b
necessary to create such a game in its basic form. Experier
documentation if you have troubles using this.

Show hints □

Game settings

Story

Figure 7: Go to story settings



Inspector

StorySettings

General story settings

Default message sound: None (AudioCli
Background track: None (AudioCli

The default message sound will be played whenever a message is fired/arriv
game is active.

Character List

ch_Sue ⊙ Name: Sue

▼ Hide

Character...

ID: 1

Name: Sue

msg color

msg background image

msg alignment: LEFT

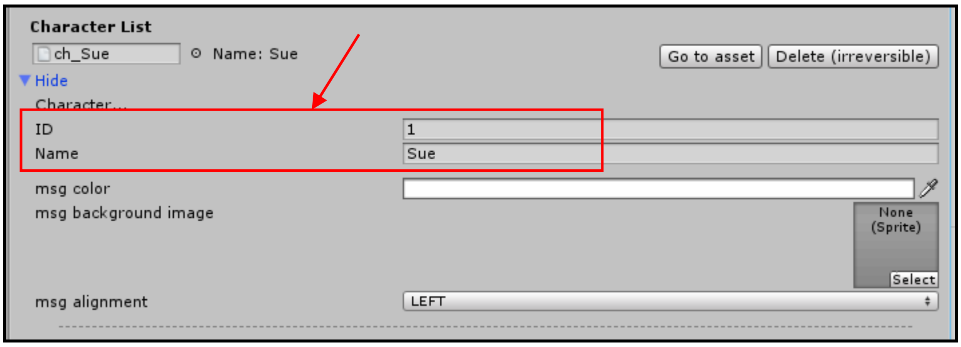
ch_John ⊙ Name: John

▶ Show

ch_Me ⊙ Name: Me

▶ Show

Figure 8: Add character



Character List

ch_Sue ⊙ Name: Sue

▼ Hide

Character...

ID: 1

Name: Sue

msg color

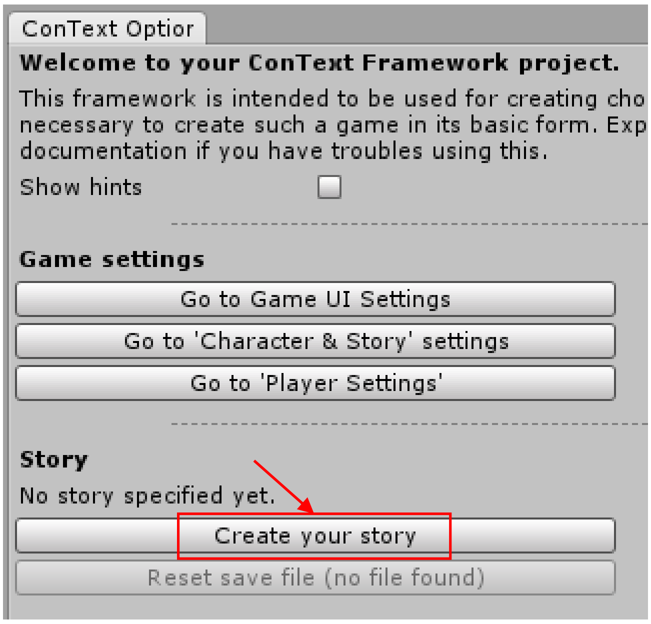
msg background image: None (Sprite)

msg alignment: LEFT

Figure 9: Customize character

3. Next, you will create the start of your story. For this, click **Create your story** in the ConText Overview. This will create the first node of your story, which (as of now) will be a Text module. The now created module will be displayed in the Inspector window. There you will need to choose the Character this message should be sent by clicking on the small circle next to the Character field and then double clicking an entry in the selection window that will pop up.

   Right below that is a text field which will hold the text this message consists of.



ConText Optior

Welcome to your ConText Framework project.

This framework is intended to be used for creating cho necessary to create such a game in its basic form. Exp documentation if you have troubles using this.

Show hints □

Game settings

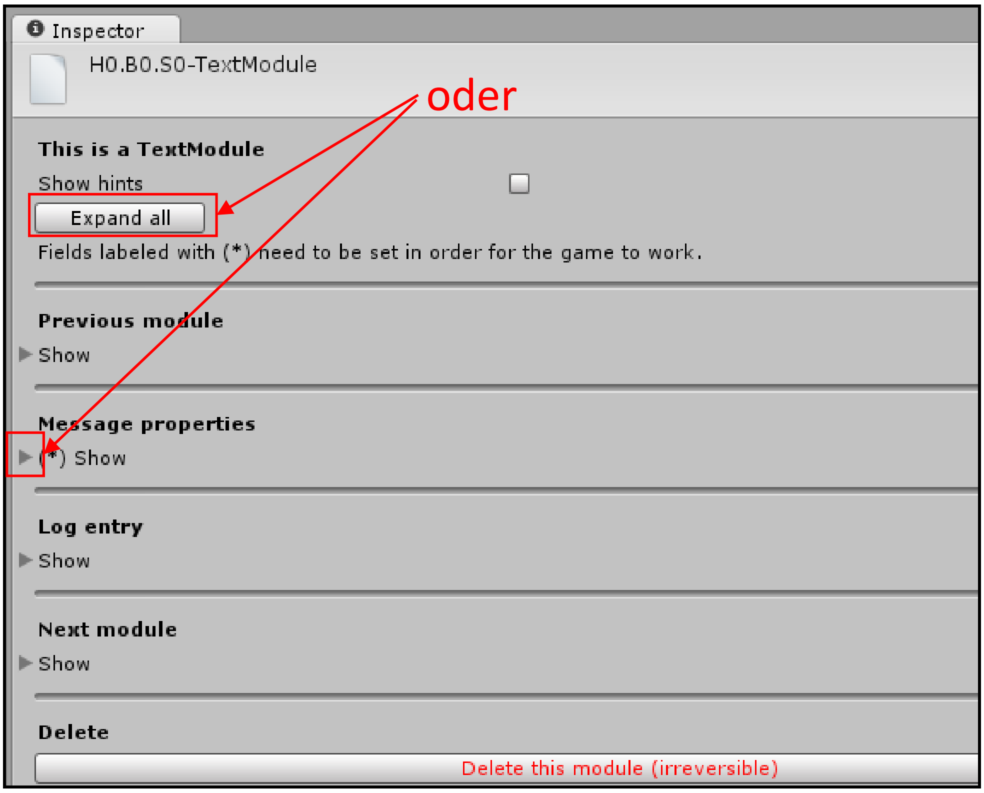
Go to Game UI Settings
Go to 'Character & Story' settings
Go to 'Player Settings'

Story

No story specified yet.

Create your story
Reset save file (no file found)

Figure 10: Create story



Inspector

H0.B0.S0-TextModule

This is a TextModule

Show hints □

Fields labeled with (*) need to be set in order for the game to work.

Previous module

▶ Show

Message properties

▶ (*) Show

Log entry

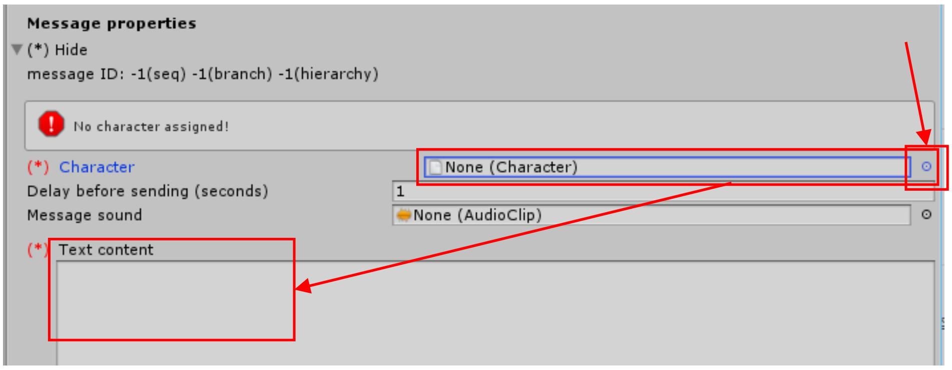
▶ Show

Next module

▶ Show

Delete

Figure 11: Message



Message properties

▼ (*) Hide

message ID: -1(seq) -1(branch) -1(hierarchy)

No character assigned!

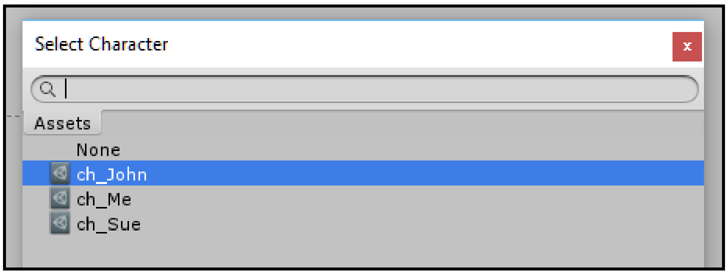
(*) Character: None (Character)

Delay before sending (seconds): 1

Message sound: None (AudioClip)

(*) Text content

Figure 11b: Message



Select Character

Assets

None

ch_John

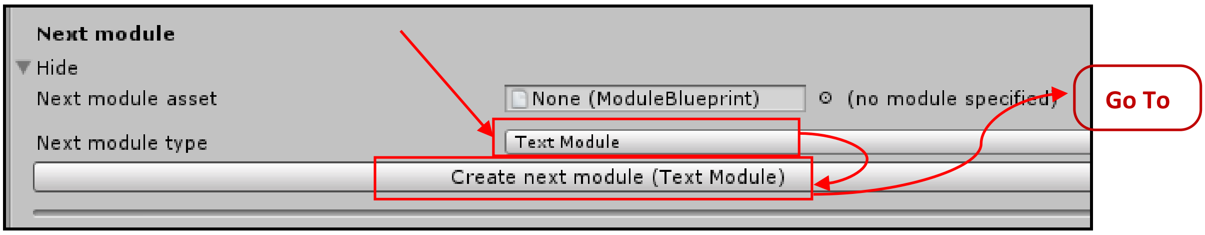
ch_Me

ch_Sue

Figure 12: Select char.

You may also customize the Delay before sending (i.e. how long the game will wait before it virtually "sends" the message) or the specific sound the message should play upon sending *(Note: there is a default sound every message will play upon sending, so this one here should only be used for ones specific to that message)*.

4. Now your story consists of one message. Within each module, you can create the next module to follow after it (sometimes given a certain condition) through the *Next module* segment in its Inspector (or Replies for the Reply Module). For this tutorial, select the **module type as "Reply Module"** and click **Create next module**. This Create button will always reflect what type will be created. To navigate to the created module, click **Go To**.



Next module

▼ Hide

Next module asset: None (ModuleBlueprint) (no module specified)

Next module type: Text Module

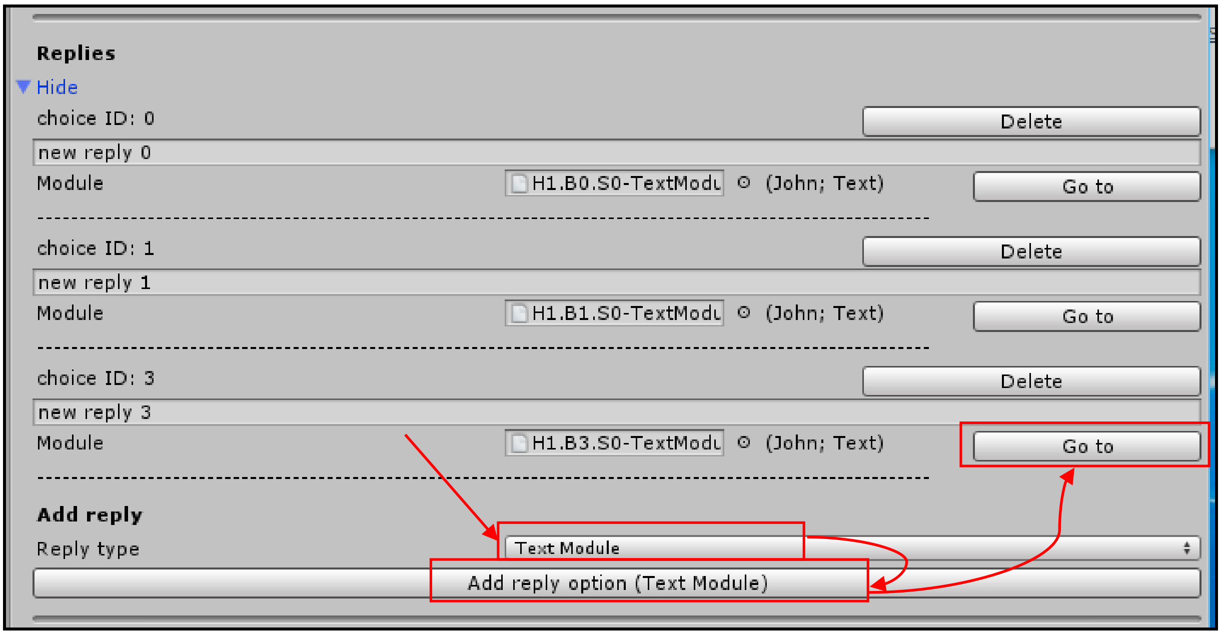
Create next module (Text Module)

Figure 133: Next

5. This will have put that Reply module into focus in the Inspector.

   A specialty about the Reply module is that after displaying its text, it will not automatically send the next message, but leave the player with the respective number of replies to choose from. Each reply may lead to a different branch in the story.

   For this step in the tutorial, set up Character and Text as in step 3, then scroll down to the Replies segment. It works similarly to the Next module segment, with the difference being you can add multiple entries. **Add** at least one **reply option**. If you want to add multiple replies, simply repeat this action. For each reply, you can define the text its button should provide to the user (which is so far "new reply x"; *Note: this is not the text content of that reply, merely the text of the button that will trigger it*). Once you are done, click Go to next to the respective entry you want to configure next.



Replies

Hide

choice ID: 0

new reply 0

Module H1.B0.S0-TextModu (John; Text)

choice ID: 1

new reply 1

Module H1.B1.S0-TextModu (John; Text)

choice ID: 3

new reply 3

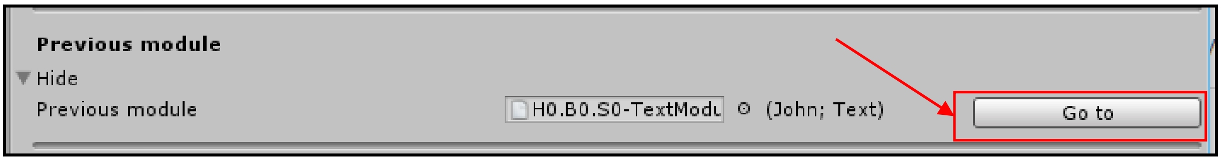
Module H1.B3.S0-TextModu (John; Text)

Add reply

Reply type Text Module

Figure 144: Replies

6. The selected module should be in focus now. For the sake of this tutorial, look at the *Previous module* segment at the top of the Inspector. If the previous module is specified, it will have a **Go to** button next to it. Clicking it will naturally take you to that previous module, in this case the Reply module from step 5. You may have seen by now that each previous or next module has that **Go to** button, which as of now is a simple way to traverse your story modules.



Previous module

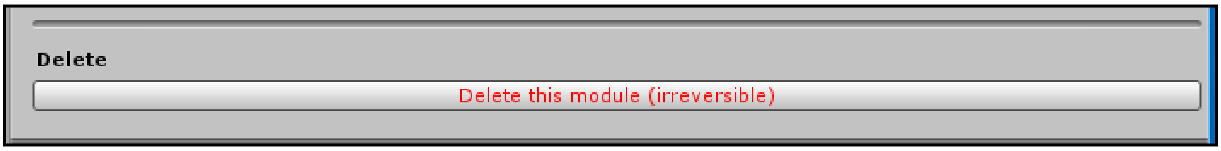
Hide

Previous module H0.B0.S0-TextModu (John; Text)

Figure 15: Go to

7. With this, you should know the basic process of creating modules. If you want to manually reconnect nodes in a different way, you can always click the little circle next to one and choose a different existing module. Be aware though, this can easily screw up the structure of the story and might lead to flawed playback if not done carefully.

8. At the bottom of each module’s Inspector is a *Delete module* button, which will delete that module and attempt to properly connect the surrounding modules. Be aware, however, that when dealing with modules that have multiple next modules (like Reply modules), this may lose the connection to other branches.

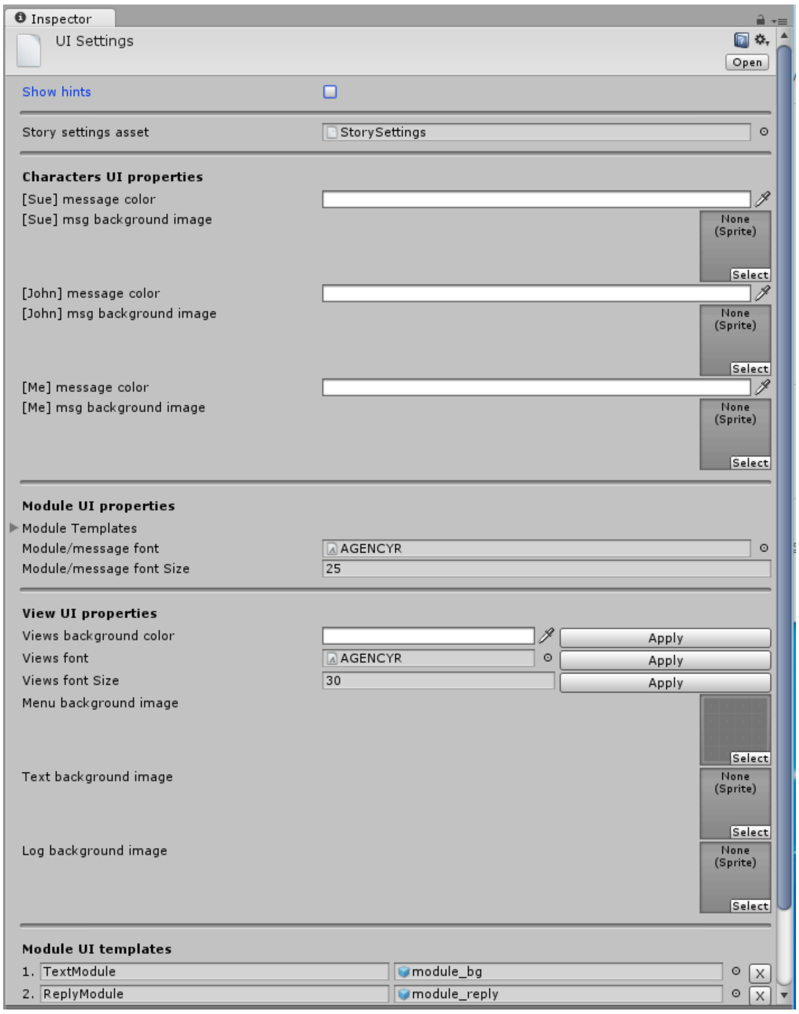


Delete

Delete this module (irreversible)

Figure 16: Delete

9. As the last step of this tutorial, click on **Go to Game UI Settings** in the ConText Overview. This will take you to the visual properties of the messages and modules, where you might change colors and background images for each character’s message bubbles, the font and font size for messages as well as the font, font size, color and background images for each of the three main screens (Main menu, Text view, Log view).



Inspector

UI Settings

Open

Show hints □

Story settings asset: StorySettings

Characters UI properties

[Sue] message color

[Sue] msg background image: None (Sprite) Select

[John] message color

[John] msg background image: None (Sprite) Select

[Me] message color

[Me] msg background image: None (Sprite) Select

Module UI properties

Module Templates

Module/message font: AGENCYR

Module/message font Size: 25

View UI properties

Views background color — Apply

Views font: AGENCYR — Apply

Views font Size: 30 — Apply

Menu background image: Select

Text background image: None (Sprite) Select

Log background image: None (Sprite) Select

Module UI templates

1. TextModule — module_bg — X
2. ReplyModule — module_reply — X

Figure 17: UI Settings

10. Go time! Now that you have set up the framework and created your first basic game with it, it is time to test it and play it. For this, Unity provides the editor play preview. At the top center of the program, you will see **three buttons, looking like typical Play, Pause and Forward** buttons.

They work as follows: Clicking the play button will start the game in editor preview, essentially showing you a close approximation of what the game currently is. Active Play mode is visually recognizable in the Play button being blue, as well as the whole interface becoming a darker gray. Pressing the Pause button will pause/un-pause the preview, again indicated by the button being blue when paused. Pressing the forward button will advance the game by individual frames (which would normally only take a few milliseconds per frame), but has likely no use for your game at this moment.

Important to keep in mind is that all changes made to assets while in Play mode are not saved, they will be reverted once play mode is disabled again.

Now, click on the **Play button** at the top center of the tool and check out what your story looks like in the preview window.

To stop game preview, simply press the (now blue) Play button again.

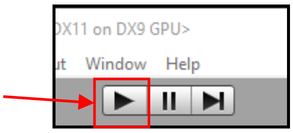
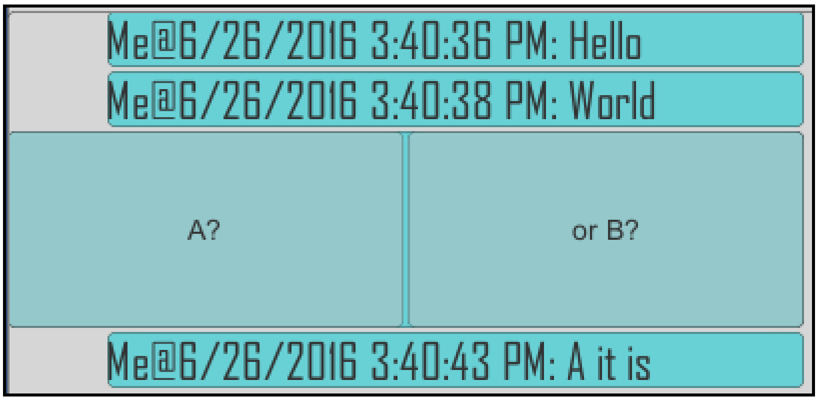


Figure 18: Play!



Me@6/26/2016 3:40:36 PM: Hello

Me@6/26/2016 3:40:38 PM: World

A?

or B?

Me@6/26/2016 3:40:43 PM: A it is

Figure 19: Small beginnings

Part 3 – Troubleshooting

Q: My new story modules are not showing up in the game preview!

A: There are multiple possible causes for this. The most likely are:

1) Play mode is still engaged. Make sure the Play button is not blue.

2) Previously existing story connections changed, as such the existing save file does not fit the storyline anymore. To fix this, click **Delete Save File** in the ConText Overview window.

3) Somehow the IDs of one or more modules in the storyline were corrupted. To fix this, click **Fix IDs** in the ConText Overview window (*Note: Fix IDs is still experimental and might not resolve the issue*).

Part 4 – Additional info about importing files into Unity

It may likely happen that you want to import custom image files to use as background or in an image module or perhaps different fonts.

Files can generally be imported into Unity by simply dragging and dropping them into the Project/Asset folder window. So you can just right click into any folder in that Project window, select Create – Folder to add a separate folder for your files, then drag and drop them in there.

In the case of image files, these will by default be imported as Sprites given you project was set to 2D upon creation. If that is not the case and the image is not available for selection as such an image, select the asset and in its inspector, set Texture Type to “Sprite”.