Overview

PowerQuest is a unity tool for creating 2d point and click adventure games (but you knew that already).

If you've used AGS before some things will seem familiar, but, being Unity, things are pretty complex, so reading this is probably necessary to get anywhere. If you haven't used Unity, then probably do a bunch of tutorials to get your bearings first.

Other documentation / video series

This documentation roughly covers the Editor side of things. Also check out the online documentation for scripting, and more recent updates. Eventually this information will be migrated to the online one:

View the PowerQuest scripting documentation here!

There's also a **Quick Tutorial Video** to get you started. (Sorry for the terrible audio mix!)

About PowerQuest Development

This really started as a tool for myself, after having used Adventure Game Studio a way back, and Unity more recently, I wanted something that was more like working in AGS, but with the flexibility of Unity. I'm making a game with this tool, and developing the tool as I go.

For help and feedback-

- Discord channel- Join the powerhoof discord, and post in the *#powerquest* channel. It's the quickest way to get help.
  https://discord.gg/powerhoof
- Forums: **https://powerhoof.itch.io/powerquest/community**
- Email me dave@powerhoof.com
- Tweet me @duzzondrums

WHAT IT DOES

(Main features)

- Streamlines unity for making 2d point-and-click adventure games
- Similar features, workflow and scripting to Adventure Game Studio
- All the advantages of Unity (portability, fancy shaders, no real engine limitations basically)
- Quest Editor window to set up your characters/rooms/inventory, etc.
- Script Editor to quickly edit interactions in simplified dialog style language.
- Script hot-loading- edit scripts while in game for rapid-fire testing
- Scripts save to native unity c# so you're able to do anything unity allows you to
- Tailored to Lucas arts style dialog with one/two click interface (for now).
    - Sierra style portrait gui is available, though not fully featured yet
- Dialog tree system, for branching dialog (similar to AGS)
- Smooth scrolling camera + parallax
- Custom 2d audio system
- Powerful sprite animation system (PowerSprite) included. Easy directional animation handling
- Options for both pixel art or high-res games
- Text export to script file for dialog recording. Speech/voice support and automatic lip-sync generation
- Text export to csv, for translation/localization
- Automatic Save/Restore system that won't break user's saves when you change things

WHAT IT DOESN'T

(Caveats, things that aren't planned to be included)

- No 3D – I guess you could use some systems and make a 3d game, but it's tailored for 2d.
- Not friendly to non-coders– If you don't want to do any coding at all, try AdventureCreator. Though it's a good place to start learning!
- No Huge Support Community - Honestly, I'm not going to be able to compete with the amazing support that AdventureCreator and the AGS community gives. Features and fixes won't be coming thick and fast. But the code is all there for people to edit themselves, so you won't get stuck with no way to make something work.
- Not so simple - Unity's a huge beast to tame (compared to AGS), no getting around that.

WHAT IS PLANNED

(Main TODO features, things AGS has that this doesn't yet)

- Improved font support and TextMeshPro integration
- Integrated "text input" GUI control
- More editor customization for verbs (currently optimized for just Look and Use), though you can script your own style.
- Further support for characters following other characters (if requested)
- Anti-glide walking style from AGS (if requested)
- Templates for other GUI interface types (currently there's just the single/two-click and SCUMM templates), and high-res game template
- A continuous stream of tweaks and improvements as I work on my own project

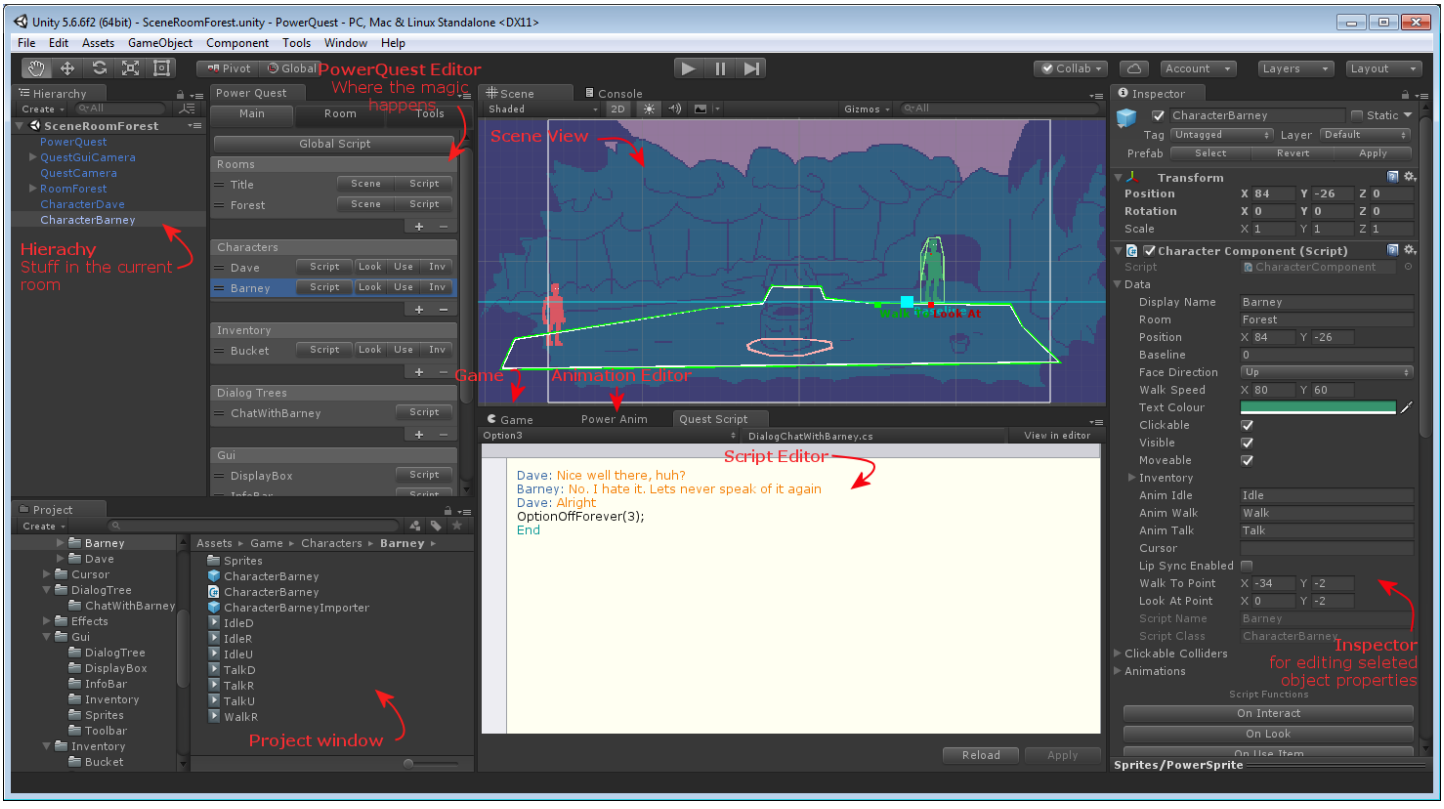
GETTING STARTED

CREATING A NEW POWERQUEST PROJECT

- **Install Unity:**
  - The latest unity is probably the least tested. I'm still a using Unity 2017.4! But later versions work.
  - I recommend using LTS versions of unity for stability. https://unity3d.com/unity/qa/lts-releases
- **Create a new project**
  - Double click the PowerQuest unitypackage
  - On the unity project window- Click **New,** select the name/location, set it to **2D,** then click **Create Project**
  - Once it loads, click **Import** on the popup that appears with all the files to import.
  - In the file/edit toolbar at the top, click **Window** -> **PowerQuest**, and dock it
  - Hit the **Set it up!** Button
- **Run the demo game:**
  - Hit play in the editor to test the demo game!
  - Hit the scene button next to the **Forest** room
  - Hit play to test from that scene
- Now just jump in and start changing stuff, and adding your own

UNITY LAYOUT OVERVIEW

Here's the basic layout I use (although I change it up constantly)



There's lots going on here. I'll focus on the PowerQuest panel since that's the main thing you'll navigate by.

POWERQUEST PANEL

This is your hub for your adventure game objects. I tend to call them **Quest Objects**.

The PowerQuest panel has 3 tabs:

- Main:
  - View/Select Quest Objects. (Saves you having to find them in the scene or Project Panel)
  - Quickly jump to editing interaction scripts on quest objects
  - Quickly jump to different Room Scenes
  - Add or remove Quest objects (with the +/- buttons)
- Room:
  - View/Select Hotspots/Props/Regions/Points/Walkable areas in the open room.
  - Quickly access scripts
  - Add or remove the Room's Quest Objects (with the +/- buttons)
- Tools: Miscellaneous Adventure gamey things
  - PowerQuest is split up into different game objects to control different things, such as:
  - Project settings: The PowerQuest prefab has a lot of game wide settings to play with
  - Game Text tools:
    - Process scripts to extract text, then generate a script for voice actors, and generate lip sync data, if you want
    - Export game text to csv for translation, and import it again
  - Mouse Cursor: You can change how your cursor behaves here
  - Game Camera: Control how the camera moves
  - Audio Settings: Some fall-off settings you probably don't have to worry about
  - Dialog Text object: The Text Object that displays above characters heads when they talk.

QUEST OBJECTS

The different Quest Objects are:

- **Characters**: Anything that can talk, walk, move between rooms, etc.
- **Rooms**: Each scene in your game (it might be a forest or a cave but it's still conceptually a room). Rooms also have:
    - **Hotspots**: Clickable areas in your room
    - **Props**: Visuals in the room, including background/foreground. They can be intractable just like hotspots, but don't have to be.
    - **Regions:** Areas that can trigger events when the player walks on them, and they can tint/scale the character.
    - **Points**: Named positions that can be easily accessed in scripts
    - **Walkable Areas & Holes**: Defines where characters can and can't walk
- **Inventory Items**: Each character can collect these items and use them on things, they can also be looked at, used on things, etc
- **Dialog Trees:** For branching dialog, each has a number of dialog options that trigger a script
- **Guis**: Guis that can be controlled simply with the script (eg: so you can turn them on/off). See http://powerquest.powerhoof.com/manual_gui.html

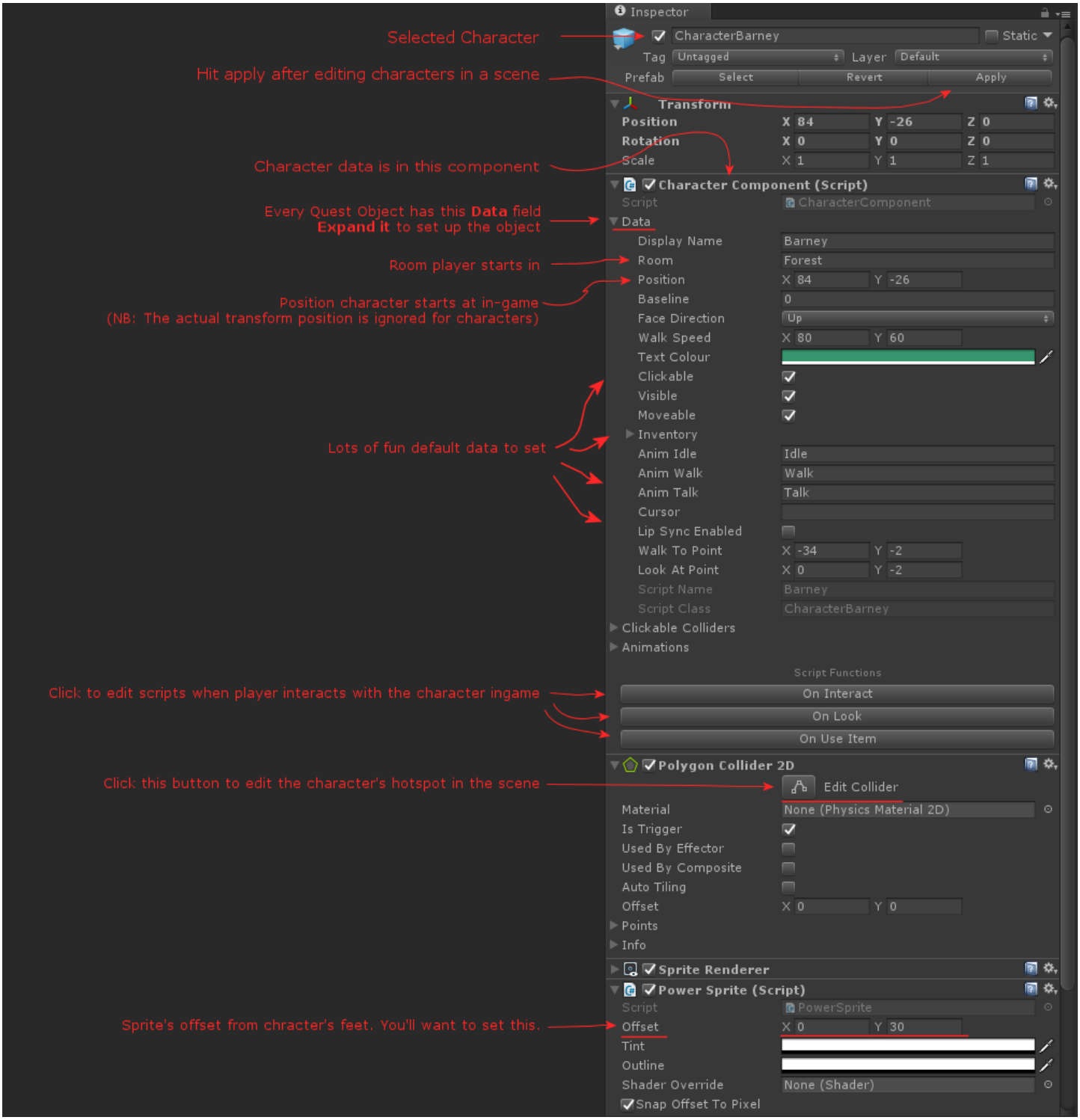
Inspector Panel: Editing Quest objects

When you select a Quest Object, it'll either select an instance in the current scene, or select it's prefab in the project. Either way, it'll open it in the *inspector*.

Quest Objects all have a "*Data*" field under their component which you *need to expand* to edit their default data.

Quest Objects are all have different data to set up. Eg. Characters, Props, and Hotspots will all have a polygon collider, where you edit the clickable region.

Here's the character inspector with some specifics for setting up a character



IMPORTING SPRITES / CREATING ANIMATIONS

This gets its own section because there's a bunch of ways to do it

BASIC:

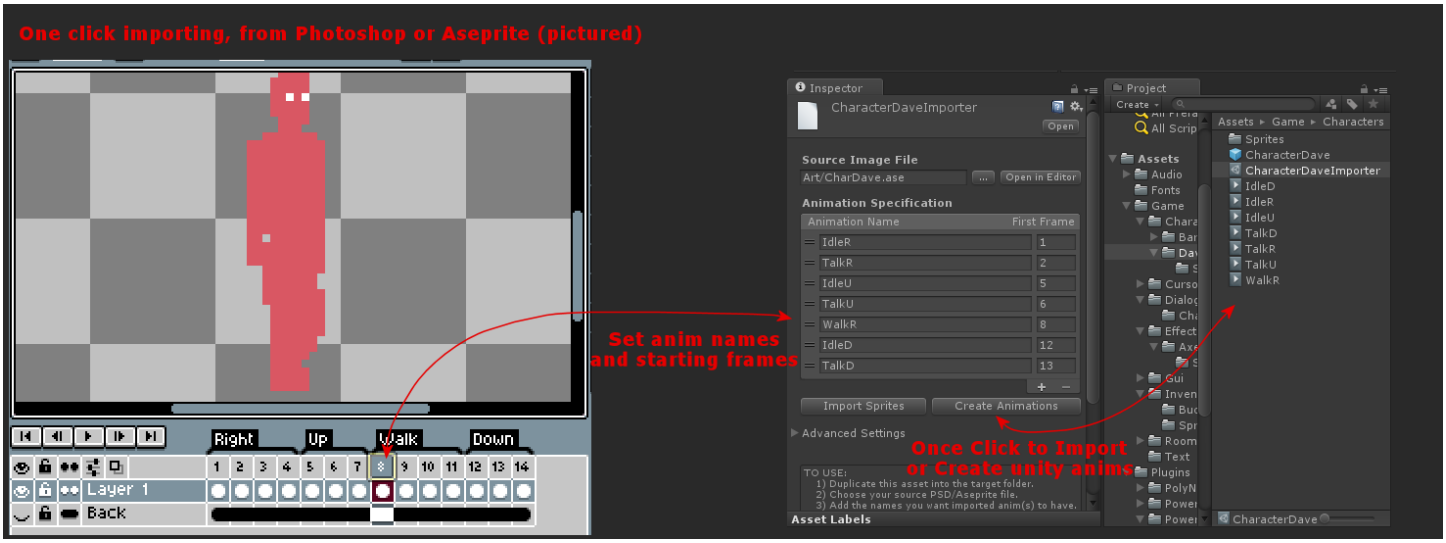
- To Import Sprites
  - When you create a Room or Character, a new folder will be created for it.
    Eg. *Assets**Game**Characters**Dave*\
  - Navigate to that in the *Project* window (click on the character/room in the Power Quest Panel)
  - For Photoshop users:
    - Open photoshop file to import
    - Right click in the appropriate **\Sprites** folder in unity, select "***Import Sprites from Photoshop***"
  - Or: Save/Drag the sprites or spritesheets you want into the appropriate ***Sprites*** folder
- Create animations
  - Navigate to the folder for the Character or Room in the Project Window
  - Click "Create" -> Animation and name it (See "*Animation Names*" below for how to name animations)
  - Open PowerSprite Animator (see below for details) and drag the sprites into the timeline.
  - Set up timing/looping options how you'd like.

ADVANCED - USING THE SPRITE IMPORTER:
(windows + photoshop/Aseprite only)

The sprite importer is a quicker way of importing a bunch of sprites or animations from a single file.

It's used to map in-game sprite files to art source files. Once set up it makes editing and re-importing really quick.

For scenes I find it easiest to have all the props on different layers in my photoshop/aseprite file, and then have each prop layer turned on for one frame. Then I can edit and export the whole scene really quickly whenever I want to make tweaks.



- To Import Sprites:
  - When you create a Room or Character, a new folder will be created for it.
    Eg. *Assets\Game\Characters\Dave*
  - Navigate to that in the *Project* window (click on the character/room in the Power Quest Panel)
  - Select the Importer file in the Character or Room folder. Eg. *CharacterDaveImporter*
  - Set the source PSD or Aseprite file (You can duplicate the importer for each additional source file)
  - Add a row for each animation you want to output, and the first frame of each animation. If it's just 1 anim or 1 frame that's fine. (See "*Animation Names*" below for how to name animations)
- Create Animations:
  - Click "Import Sprites" to have sprites imported (you can always delete them and redo if you got something wrong). They'll be copied to the sprites folder, and be set up and named nicely
  - Click "Create Animations" to create animations
  - Open PowerSprite Animator (see below for details) and set up timing/looping how you'd like.

PowerSprite Animator

This is another tool I made which comes bundled with this. It makes editing frame by frame animations much easier than using Unity's default tools. There's a separate manual pdf for it in the project, but it's pretty self-explanatory.

Open it by clicking *Window->PowerSprite Animator* in the taskbar

Animation Names

It's also recommended to create an animation even if an object only has a single frame or sprite. It just keeps things consistent, and allows more things to be automatic. Of course you can just add a default sprite to a prop's sprite renderer (for example) if you don't want to do this.

Props, Inventory: Name the animation the same as the prop or item (eg: Door, Key) and it'll automatically be hooked up for you.

Characters: Use *Idle, Talk, Walk* by default, they can be overridden in data, and in scripts

Directional animations for characters should end with L, R, U, D, UL, UR, DL, DR (for left, right, up down, up-left, up-right, down-left, down-right).

Eg: WalkR, IdleDL, OpenDoorU.

If there's only a Left, or Right, then the sprite will be automatically flipped when going the other direction. You can include as few or many directions as you want.

PROJECT SET UP

Start with the demo project and edit the stuff that's there, delete things you don't want.

There's a million settings in unity for project setup, I won't document everything, but google's your friend if you're searching for a setting.

GAME RESOLUTION: PIXEL ART OR HIGH RES

To change your intended game resolution:

- Under *Tools -> PowerQuest Project Settings…*
- Set the **Snap to Pixel** option to **on** if your game is pixel art, otherwise **off**
- Set the **Vertical Resolution** to be what you want your background height to be (eg: 180, 720, 1080)
    - You can also override this per-room too in the Room Inspector
    - Also set the default camera size for previewing the game in the Camera prefab
      Go to *Tools -> Game Camera-> Select Prefab button,* and *s*et the *size* to HALF your vertical resolution
    - (Alternatively you could stick with the 320x180 unit size, and just adjust fonts and the *Pixels Per Unit* setting in sprites you import. You can change the default import setting in SpriteImportSettings.cs)
- Note on fonts: If you want to use high-res fonts (not pixel art):
    - Edit the TextMesh component on text. Increase the Font Size (eg: from 10 to 100) and decrease the Character Size (eg, from 10 to 1) to make them not blurry. It takes a bit of fiddling to get the right balance. In future I'll add support for TextMesh Pro to help with this.
    - Text is used in a few places in PowerQuest, so you'll need to edit it in the following prefabs:
        - DialogText
        - GuiDialogTree
        - GuiDisplayBox
        - GuiInfoBar
        - GuiStatus
        - GuiSpeechBox (if using portrait style dialog)
    - For Unity 2018.2 or EARLIER, drag them into the scene/hierachy to preview/edit. Then hit apply in the inspector and delete them from the scene/hierarchy.
    - For Unity 2018.3 or LATER, double click them to edit.

INTERFACE STYLES

Although PowerQuest editor is tailored towards a Single or Two click interface (BASS, Wadjet eye style). You can customize your in-game interface by adding your own guis and changing the way mouse clicks are processed in your Global Script. (If you know what you're doing)

More built-in customization would come in future. Let me know what you want!

Dialog Style

In PowerQuest general settings there's an option for different dialog styles:

- Above Character: LucasArts style dialog visuals, text displayed above the character in the game.
    - This is the main one supported, (cause that's what my games use).
- Portrait: Displays GuiTextBox at the top of the screen and the Talk anim for the character next to it
- Caption: For displaying dialog text at the bottom of the screen. (ie: not above character's head)
    - You can add a ScreenAlign component to the DialogText prefab to make it align to the bottom of the screen or wherever you want to put it

I'll add more stuff based on requests, so let me know.

Room Set Up

Basic New Room Setup

- Click the + under rooms in Power Quest Main tab
- Give your room a name
- Select the Room tab in Power Quest
- Add a prop for the background (Call it Back or BG or something)
    - Either import a sprite and drag it into the SpriteRenderer's Sprite field,
    - Or import sprites and create an animation for it (instructions above). Give the animation the same name as your prop, and it will appear automatically.
- Setup a walkable area
    - Click Show Polygon Editor button -> Edit Collider, drag out the "walkable area"
    - To add holes/obstacles to walk around, add a hotspot with both "Walkable" and "Clickable" set to false.
- Hit Play to test your character walking around
    - Or untick "Player Visible" if the player shouldn't be seen in this room (like the title screen)
- Add more Props and Hotspots, add interactions, etc etc

Scrolling rooms

- Edit the room bounds (or drag out the yellow box in the scene) to match the width of your background sprite, then the camera will follow your player around
- Also set the Scroll bounds to control how quickly the room scrolls right to the edge
- There are more settings for controlling camera behavior in the QuestCamera object
- You can add Parallax to Props by setting the Parallax Depth in their data (hover for tooltips) and the Parallax Alignment to help line it up on left/right of screen (-1 is left, 1 is right)
- If rooms are different sizes, you can override the project's "Vertical Resolution" (camera height) in each room.
- You can override the camera from scripts too to position it to different places manually, and zoom in on things.

To view characters in the room:

- Drag your Character from the project window into the scene
- They're only visible in game if they're the player character, you've set them up to appear in that room in their data, or you've set them to move there in your OnEnterRoom function

Particle Systems, Text Mesh, Other Flashy Unity Things

- Props (and characters, etc) are all just Game Objects, so you can do all the normal unity things you want.
- Adding Particle Systems or Text under a prop will work fine, and Visible propery will control whether you can see them or not.
- Here's an example of getting your fancy custom unity component that's on a prop from a PowerQuest script:
  MyComponent myComponent = Prop("Door").Instance.GetComponentInChildren<MyComponent>();
  myComponent.DoAmazingThing();

Video Playback

- Props can have a video play on them instead of a regular animation
- Add a prop and in the inspector, add a VideoPlayer component.
- Set the video component up with a video file, and set desire Render Mode (Camera Near, Camera Far, and Material Override can all work)
- To play video in script use the PlayVideo functions on the prop. eg: Prop.MyVideoProp.PlayVideo();

Character Set Up

Basically like setting up a room, look at the "Inspector Panel" image that shows the key things you need to set.

Here's some caveats:

- Don't forget to set the Sprite Offset so your character's feet are at his pivot
- There's currently no way to set the Player Character, it's always just the first one in the list. (Though that's easy enough to add later)
- If you're editing a character in the scene, you'll want to hit Apply so it's changes will be remembered across scenes
- Moving a character instance around in the scene won't change their position in game (like it will for a prop). You'll have to set the Room and Position in the editor.
- Don't forget the animation naming conventions in that section above

Other Setup

Inventory

- Same deal, pretty easy.
- Inventory sprites/animations go in a shared folder under Game/Inventory/.
- (The prefabs/scripts are in separate folders for now, but later they might move to shared folders/file too)
- The Cursor object can be set to give an "outline" to inventory items when they're hovered over something clickable (Under Tools->MouseCursor)

Dialog Trees

- These are done in a similar way to Adventure Game Studio if you've used that.
- Create a dialog tree, then add the "options" players can select.
- Click an option to edit it's script. You can call functions to enable/disable options within dialog trees to create branching dialog. See the scripting API for more info

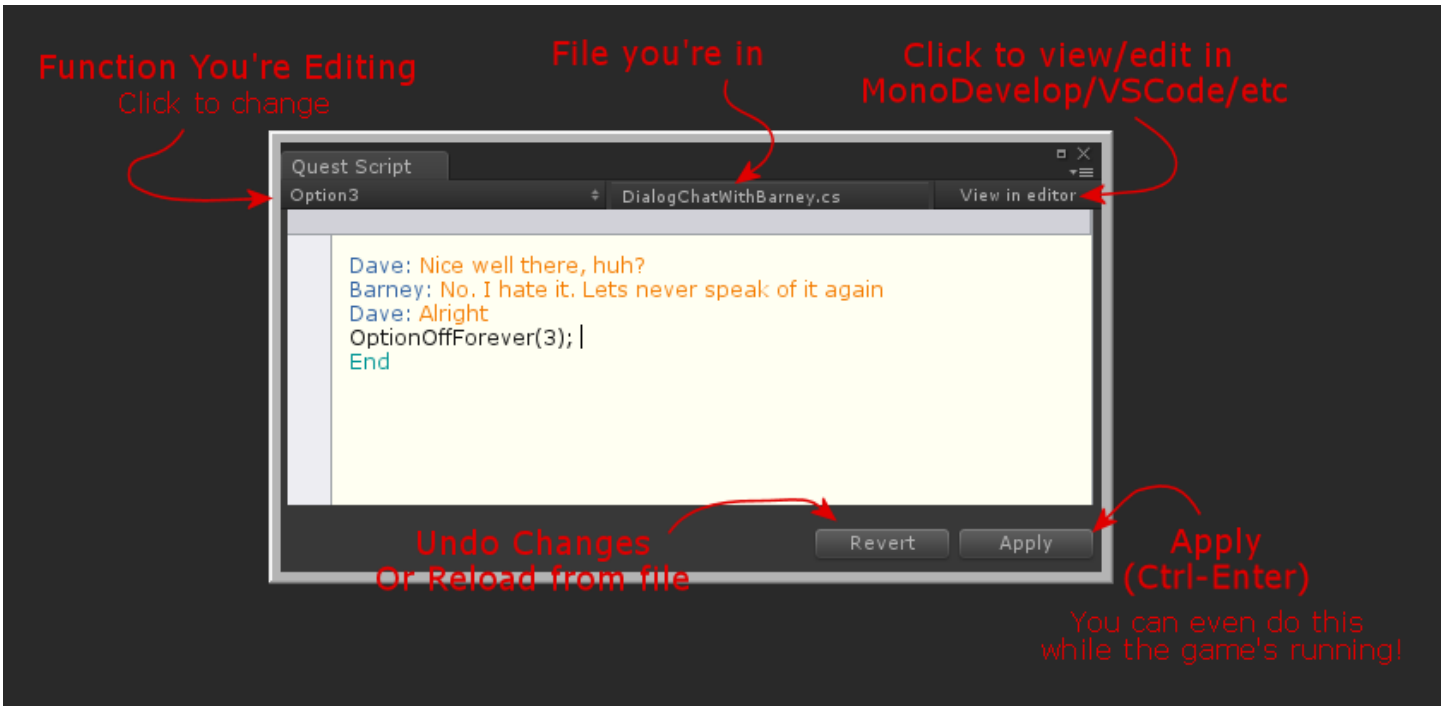
GUIs

- PowerQuest has its own Gui system you can use, which functions similarly to PowerQuest rooms, with hotspots and props.
- The full documentation for that is here: http://powerquest.powerhoof.com/manual_gui.html
- Alternatively you can use one of Unity's built in GUI systems, if you're already used to using one of them

Scripting

Ok, here's the fun bit! But I'll direct you to the **PowerQuest Scripting Documentation** cause that's got all the API details.

But basically, you click the "Script" or "Look"/"Use"/"Inv" button, and start typing away in the nifty little script editor:



You'll have to jump into the c# file sometime, so don't be shy to click "View in editor" and get used to how the files look. There's lots of ugly "yield return" everywhere, but you get used to it. In MonoDevelop, etc, you have nice autocomplete there so it mitigates

NB: If you break the code and fix it but unity doesn't see the fix, press Ctrl-R. (it forces a manual refresh). Its' my bug, not unity's ;)

Save/Restore Games

Hopefully this will mostly *Just Work* ™ , and you won't have to think about it

All the data in the Quest Objects is saved automatically, along with the basic variables you add to your scripts. However, references to game objects and more complicated things you add in your scripts aren't saved automatically.

If there's some reason you need this just ask me for now, and I'll figure out how to document it later. But hopefully you don't have to worry too much ;)

Also there's no nice gui for it yet, you can hit F5 to save, and F7 to restore, and I have a "save and exit" feature, and a "continue" game on the demo title screen. Full save/restore gui will have to come later.

GAME TEXT, TRANSLATION, LOCALISATION

Under the tools tab, there's a Game Text Tools dropdown.

Process Text From Scripts: This goes through all scripts in the game, and optionally all descriptions and Quest Text components, and assigns them an ID.

- **Preserve Ids** should be ticked if you've already got some speech files, or translation, and don't want to mess them up. If it's unticked, your dialog will be re-numbered sequentially, which is nice to have when editing your VO.
- **Dialog only** should be ticked if you're not translating the game, but are using V.O

Once this has been done, you can:

- **Generate Script**: This spites out an html script you can give to voice actors (see Audio Section)
- **Export to CSV**: This creates a spreadsheet of all the game text to open in excel or google sheets. Useful for spell checking, but also used for translation.
- **From Characters**, **From Rooms** fields are useful if you just want dialog from a particular scene or set of characters. Only Sections with specified characters will be included. (just put space between each character or room for multiple)
- **Languages**:
    - At the bottom of Game Text Tools, there's a "Languages" array.
    - Add an element to that for each language you're supporting
    - The "Code" is used in-game to specify in-game which language is selected. Settings.Language = "ES";

When you export to CSV, there'll be a column for each language, once filled in, import the CSV again to get the languages.

AUDIO

SFX/MUSIC

The audio system uses a concept of an AudioCue. The cue is an object where you set up how you'd like a sound to play. Then you can just "play" the cue from a script.

To create a SFX/Music cue and play it:

- Stick you sound files in the Audio/SFX folder, right click it and select -> Create -> Audio Cue. (or duplicate an existing cue and drag in your audio files)
- Set the volume, and other settings in the Cue, I like to add a bit of random pitch variation for sounds you hear a lot. And you can include multiple sounds in a single cue to have them play randomly.
- In a script call *Audio.Play("nameOfTheSFXCue");*
- Looping sounds:
    - Tick the "Loop" tickbox for looping sounds
    - Play it like normal. eg. *Audio.Play("FireCrackle");*
    - Call *Audio.Stop("FireCrackle");* to stop it again.
    - You can also call Audio.PlayMusic(…) and *Audio.PlayAmbientSound(…);* which handle crossfading
- Character Footsteps
    - Set C.Player.Footstep = "MyFootstepSoundName";
    - Add events to the animation timeline called "Footstep" (ensure the "Add anim prefix" box is ticked)
    - There's also a FootstepAlt incase you want to have separate left/right steps
- When you play a sound, an AudioHandle is returned, you can use this to mess with the sound how you want, I often do *Audio.Play("FireCrackle").FadeIn(2);* for example
- If you want a looping sound to get louder as you get closer, Play it, Then in your room's update function add an Audio.UpdateCustomFalloff(…) function.

VOICE/SPOKEN DIALOG

- Follow the instructions in the **Game Text** section above to Process your text, and generate your script
- Now you have the HTML of the script for the game (you can print this, save as pdf, whatever)
- Record your lines, and save the files with the filenames written next to the lines (eg: DAVE1.wav, DAVE2.wav)
- Add the file to the Audio/Resources/Voice directory. There's some notes in there about import settings for clips
- Done!

LIP SYNC

- To use this you'll have to install Rhubarb (It's free!). Get it here: https://github.com/DanielSWolf/rhubarb-lip-sync
- Put it under your project folder in folder called 'Rhubarb'. Eg: *\MyProject\Rhubarb\rhubarb.exe*
- Then, once you have dialog recorded you can hit the Tools->Game Text Tools -> "Process Lip Sync Data" button, that will run an automated tool called Rhubarb to work out which "Frame" of talking animation to play at what time. It's quick and dirty but kinda cool.
- Tick "Lip Sync Enabled" in characters you want to use it
- Those characters need talk anims with frames ABCDEFX in that order, from- https://github.com/DanielSWolf/rhubarb-lip-sync
- If you want to use the G,H frames, go to Tools->Game Text Tools -> and edit the "Lip Sync Extended Shapes". GHX for all three, GX for just those two, etc

Separate Mouth Animation

- In addition, or alongside lip sync, you animate your character's mouths separate to their body
- This means you can change your characters pose and not have to make another set of talk animations where the mouth wobbles
- To use it:
    - Create an animation for the mouth (eg: named MouthR) with your mouth frames
        - Use the ones mentioned in the LipSync section above for example
        - You can change the name of the mouth animation just like you do for talk animations
    - Set the "Anim Mouth" in your character's inspector to the name of your anim (eg. "Mouth")
    - Open your Talk animations in PowerSprite Animator (or your idle if you want to skip adding a talk anim)
    - Add a node by double clicking, and move it where the mouth should be for each frame. (See PowerSpriteAnimator documentation for more details) You can set up a preview sprite for the node to see how it'll look

KEYBOARD SHORTCUTS

Editor shortcuts

- Hold Shift - while clicking something that opens a script to open directly in Visual Studio
- Ctrl+S, or Ctrl+Enter - In the text editor, saves the code (in older versions of unity Ctrl+S)
- Ctrl + M - In the scene view copies the coordinates of the mouse cursor to clipboard to paste into code
- PowerSprite Animator has its own shortcuts, look at it's PDF in the documentation folder for those

In-game debug keys

- F5 – Quick Save
- F7 – Quick Load
- F9 – Restart
- ESC (customizable)– Skip cutscene. Hold to skip text very fast
- Space/Left click – Skip Text
- ~ + I – Give all inventory
- ~ + PageUp – Increase gamespeed
- ~ + PageDown – Decrease game speed
- ~ + End – Reset game speed
- ~ + F7 – Load game from start of room

HELP/SUPPORT/FEEDBACK

- Join our discord – https://discord.gg/powerhoof and post in the *#powerquest* channel. That's probably the quickest way to get help.
- Visit the forums on the itch.io page https://powerhoof.itch.io/powerquest/community
- Email Dave - dave@powerhoof.com

CREDITS

Made by Dave Lloyd @duzzondrums

Check out our website to see the results: http://www.powerhoof.com

PowerSprite Animator (included) is also made by me, check the docs in that folder for info