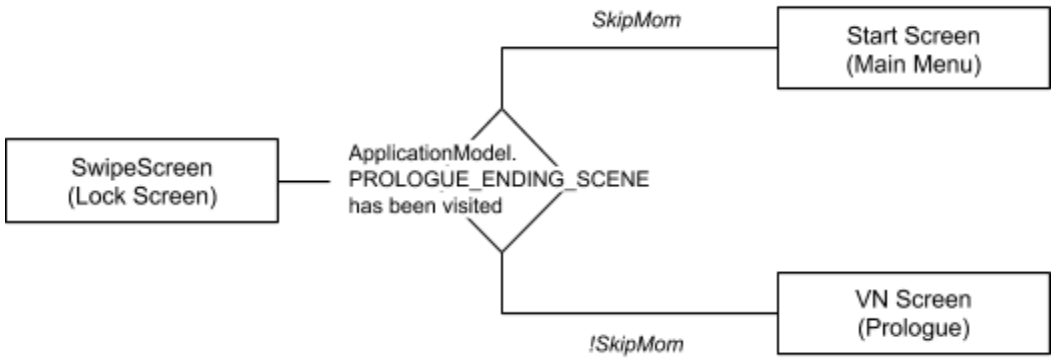
Golden Hour System Architecture

Golden Hour is built with Unity and Yarn Spinner with a custom domain-specific language built on top of Yarn for the dialogue system. Scene management is handled by a number of config files along with Yarn Spinner's built-in variable system.

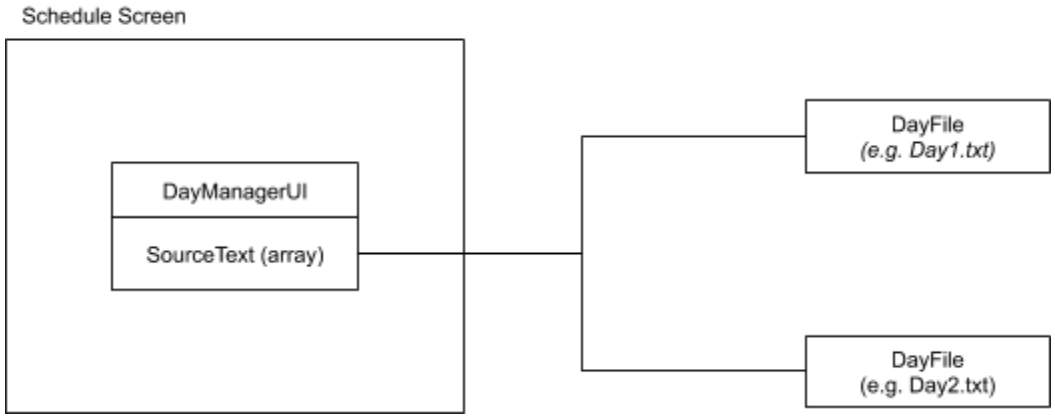
Boot Up Sequence



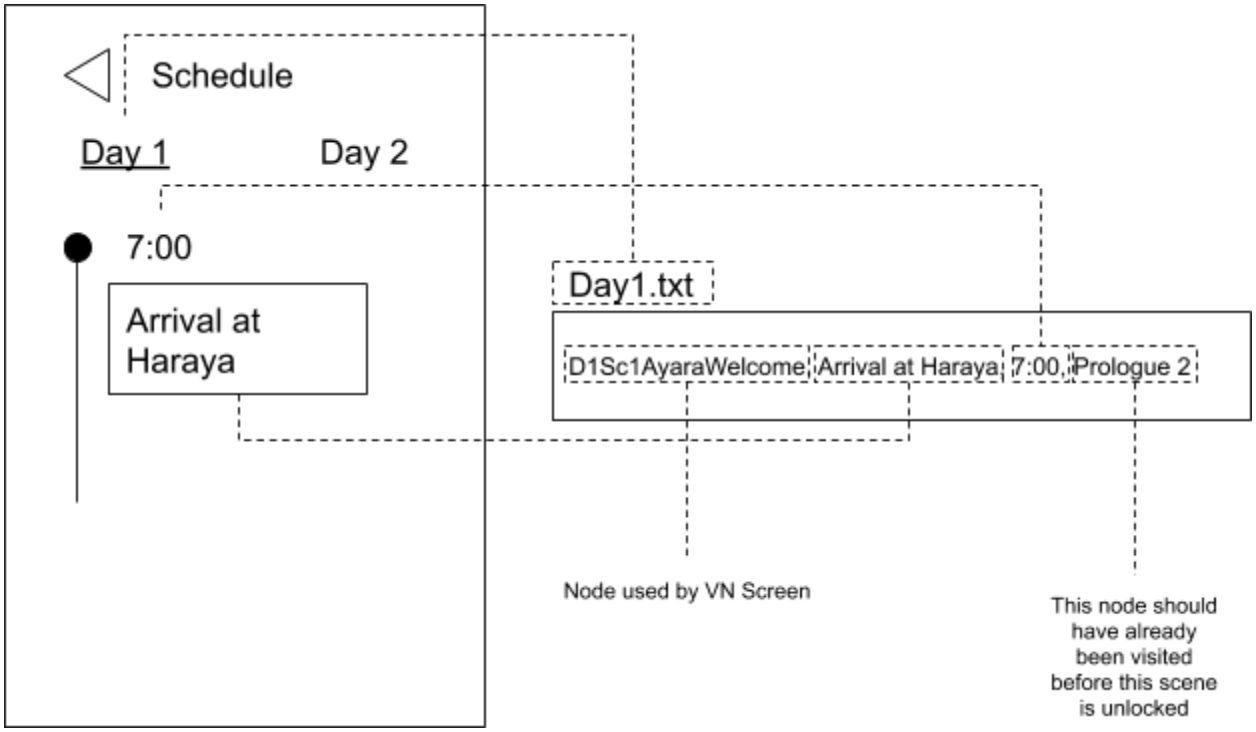
First interactable screen encountered by user after boot up is the Lock Screen (whose Unity scene is named SwipeScreen). The user lands on this screen regardless where they are in the game (at this point in development).

If the user has not gotten past the Prologue then they will be booted to the Mom Calling/Prologue scene (this is determined by the initial node Yarn Spinner variable under the Dialogue object) otherwise they will be led to the main menu.

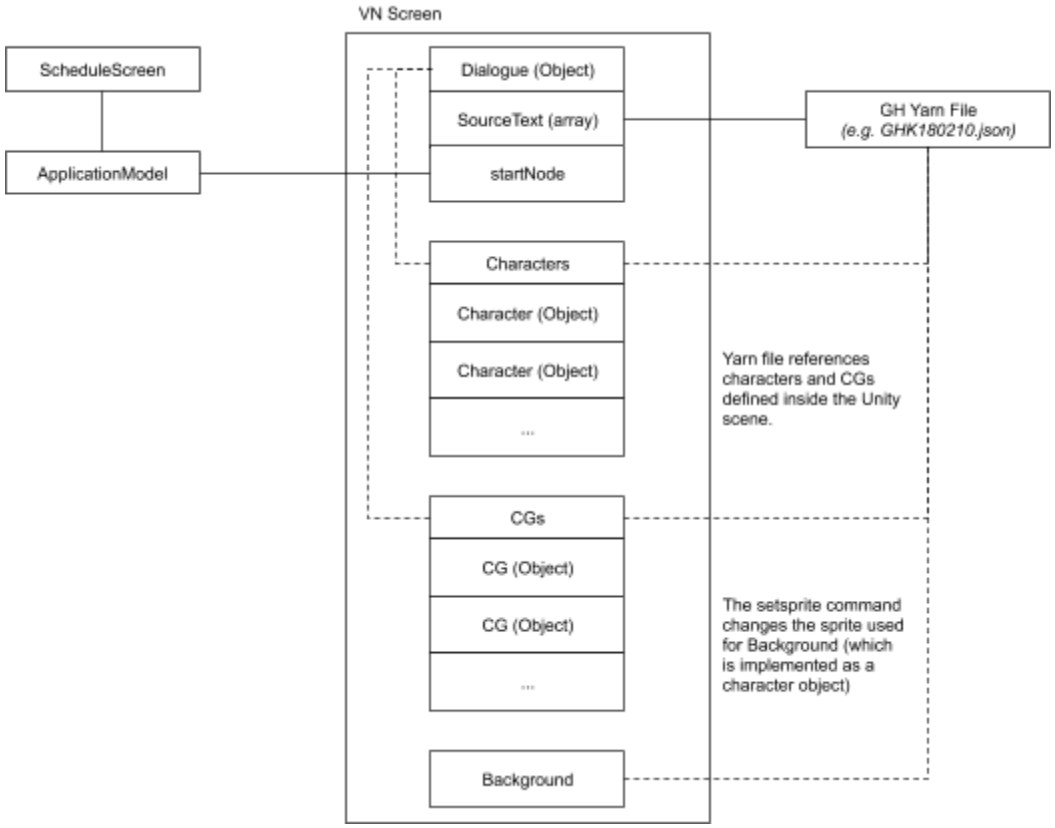
Schedule Screen + Scene Management



The list of available scenes is presented to the user via the Schedule Screen. This screen is mainly driven by the DayManagerUI along with a number of Day files declared via the SourceText array variable.



VN Screen

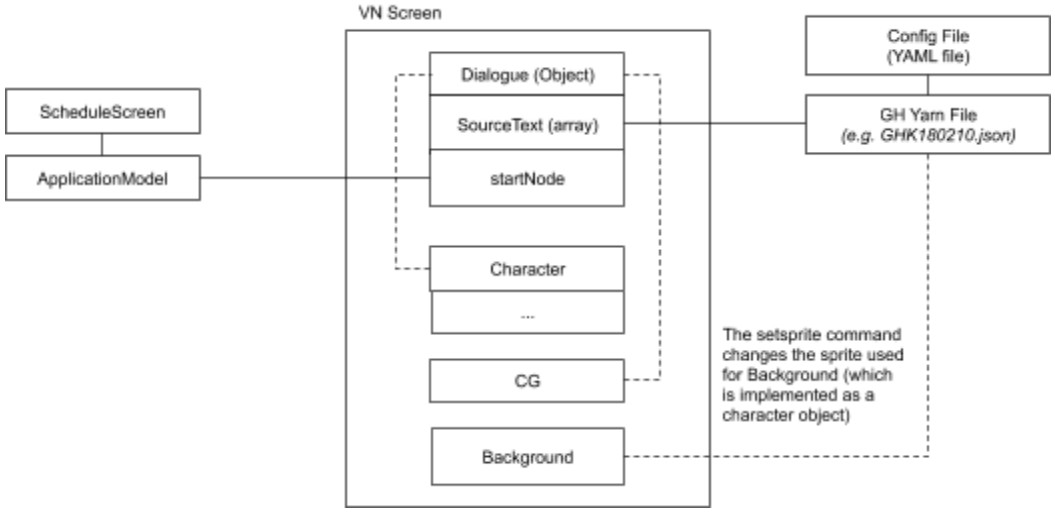


The VN screen is mostly driven by the Yarn Spinner Dialogue object, Characters and CGs defined within the VN Screen Unity Scene, and the linked Yarn Files on the Dialogue object.

If user came from ScheduleScreen, then that sets ApplicationModel.selectedScene to the desired scene based on the DayFile. DialogueRunner then sets that variable as the startNode.

Background is implemented as a Character object with the setsprite command passing the filename for the image to be used *(e.g. "<<setsprite Background Ayara_Library>>")*.

VN Screen (Updated)

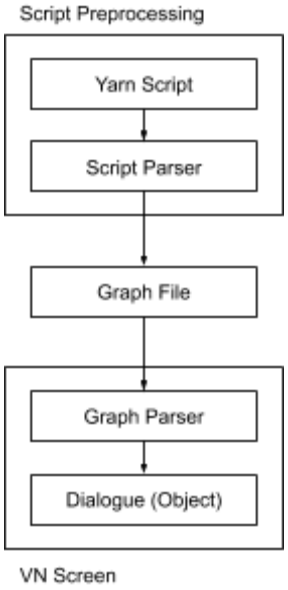


In this proposal, character and CG definition is moved outside of the Unity scene into an external YAML file which serves as a configuration file. Yarn Spinner is then used to dynamically add characters to the scene with the proper sprite instead of toggling visibility of already instantiated objects as currently done.

This should make it easier for non-developers to add and modify characters/CGs since those modifications currently require Unity scene edits. It would also make declarations more visible and interface better with Git.

To prevent faulty configuration files, a pre-commit hook that checks for config file validity is also proposed.

Point System



The Point System is designed to allow for runtime evaluation of player choices while maintaining the integrity of the evaluated score through replays and rollbacks (i.e. choices from inaccessible branches and latter nodes are not evaluated).

The Yarn file is parsed by a Python parser that produces a graph of the current script exported as an image of the graph and serialized in a custom format. The graph image is used only for documentation and serves no functional purpose in the application. The serialized graph, on the other hand, is submitted to the Dialogue object and parsed on app initialization.