RuntimeDEbugger

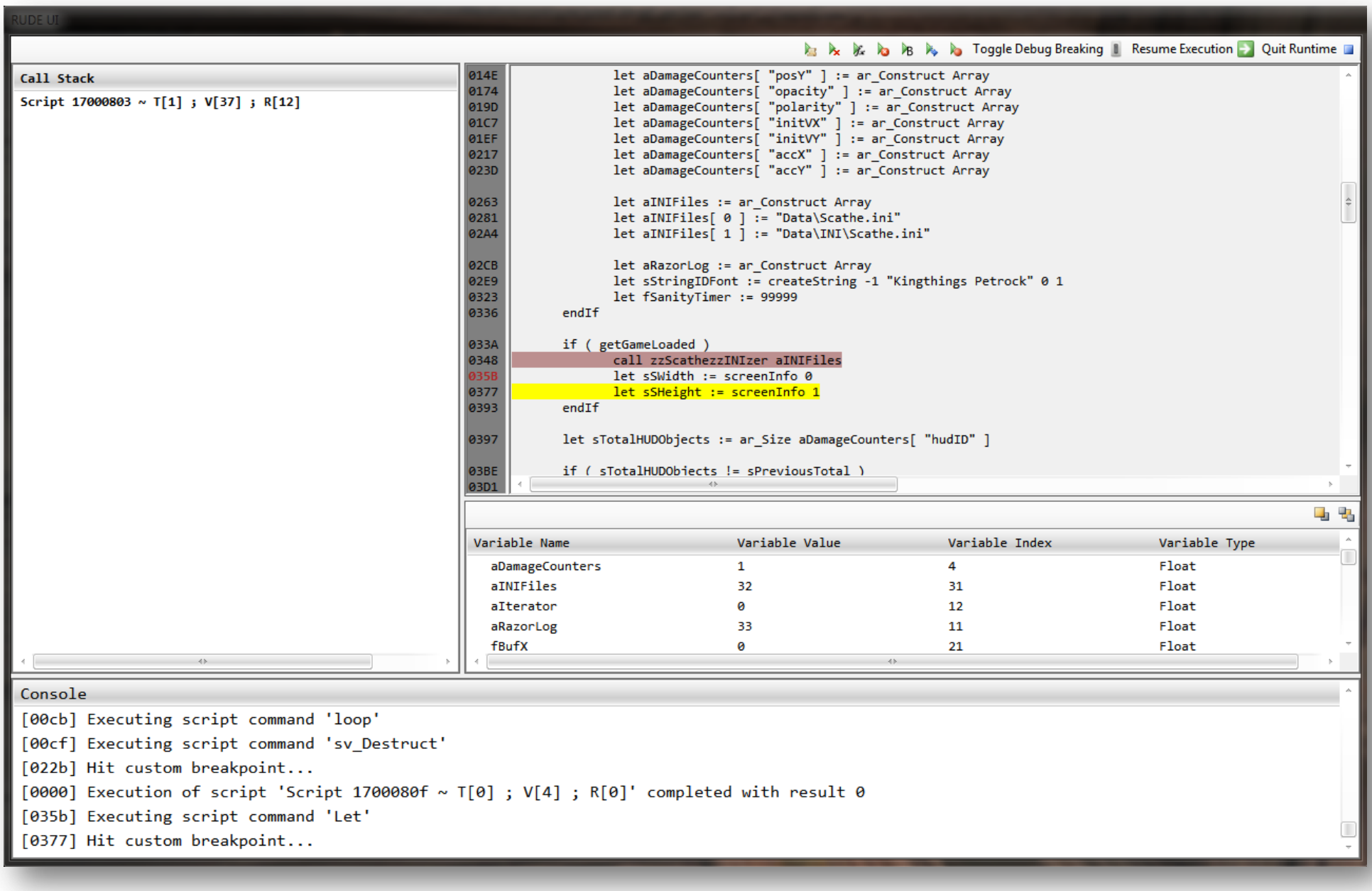
shadeMe

Version 1.0

The runtime debugger is invoked by the **`DebugBreak`** script command. Currently, this command serves as the sole entry point for the debugger (though this detail can change in later versions).

When the script runner encounters the **`DebugBreak`** command, it starts the debugger interface and initializes the executing script for debugging. When running, the debugger prevents the script from executing any further until it's resumed through its UI.

The Interface



The debugger's environment combines a console window, a call stack list, a couple of variable lists and a numbered text viewer. The top toolbar serves as the main toolbar, providing an interface to enable the various execution modes.

- **Console**: Displays standard messages and any script errors that may be encountered during the course of the debugging session. Each message is prefixed with an offset representing where the error was encountered/operation was performed.

- **Call Stack:** Displays a list of executing script contexts in the form of a stack, where the topmost script is the one being currently executed. Double clicking on an item will show the appropriate context's script and variable list. Executing contexts are described in the following format: *`Script <FormID> T[<Script type>] ; V[<Number of variables>] ; R[<Number of references>]`*

- **Variable Lists**: These enumerate the local and global variables used by a context. The toolbar attached to the listview control toggles between local and global script variables. Local variables are those that are declared in the context's script, while global variables are those externally declared but referenced by it. A listed variable's value can be modified by double clicking on its item in the listview control.

- **Script Viewer**: Displays the context's script text. The currently executing line is highlighted in yellow and custom breakpoints are highlighted in light red. Custom breakpoints can be added by clicking on the corresponding line's offset that is displayed to its left.

- **Main Toolbar**: Sports much of the debugger's toolset. Listed left to right:
    - **Execute Till Next Script**: Resumes script execution until a new instance of the script runner is initialized, i.e., the executing script calls another script, directly or otherwise. For instance, when a user-defined function is called or a scripted object is activated.
    - **Execute Till Return**: Resumes script execution until the `Return` command is encountered.
    - **Execute Till Next Command**: Resumes script execution until the next script command is encountered.
    - **Execute Till Next Error**: Resumes script execution until an error is encountered. Only the following errors are handled:
        - *OBSE script errors.*
        - *Vanilla script errors:*
            - Unsuccessful reference evaluation – Causes the delinquent script to be blacklisted from execution for the rest of the game session.
            - Unsuccessful script line execution – Causes the delinquent script's execution to stop prematurely.
    - **Execute Till Next Block**: Resumes script execution until the next script block is encountered.
    - **Execute Till Next Line**: Resumes script execution until the current script line is executed.
    - **Execute Till Next Breakpoint**: Resumes script execution until the next custom breakpoint is hit.
    - **Toggle Debug Breaking**: Prevents the `DebugBreak` command from invoking the debugger/breaking script execution. The `ToggleDebugBreaking` command can be used in the console to achieve the same effect.
    - **Resume Execution**: Deinitializes the debugger and resumes script execution until next execution of the `DebugBreak` command.
    - **Quit Runtime**: Exits the game.