Some minimal documentation of the application

Start the application by launching “start.lua”

Remote:

* Navigate between elements using <left>,<right>,<up> and <down> on the remote or keyboard (if running on PC)
* Scroll the synopsis text when focused using <up> and <down> on the remote or the keyboard (if running on PC)
* Press <ok> (or “enter” on the PC) on the “Trailer” button to “start playback”, currently there are no APIs to start the playback nor any URI in the asset data but it should be fairly simple to hook in once there is an API to start playback, additionally the “base” (body) container probably should be hidden at that point and shown once the playback is over or the user aborts.
* Press <ok> on the “Next” or “Previous” buttons to cycle through the movie database

Assetdata:

Asset data is currently hardcoded in the moviepage.lua, it should be fairly simple to load this data dynamically from any source, the important aspect is that the navigation stack is created when the page is loaded thus data can not currently be updated once the page is loaded- it’s not a major task to add support for that but the current version don’t support it.

Asset data is not enumerated, due to the way the data is iterated when constructing the navigation stack there is no guarantee the first object in the table will be movie entry 1, this could be solved easily by enumeration but it was not considered an issue right now the source data is enumerated using the pairs construct which has undefined ordering for unordered entries.

Buttons:

Navigation is handled by the NavigableElement class, which demonstrates how to connect and layout the view to handle navigation. Unfortunately as the STB don’t render RCSS content properly it’s currently impossible to see the focus status on the actual STB. This could be worked around using images as targets instead of arbitrary elements but at the cost as increased application size.

Visual Reference:

The folder Screenshots contains the reference image used as a guide for the example application layout along with 3 screenshots from the libRocket sdl2 build which illustrates the expected render result.