Functional Specification

Group 3

June 7, 2022

Contents

Revision History

Revision	Date	Author(s)	Description
0.1.0	15.02.22		Doc created
0.1.5	03.03.22	DM1306	Remove WebP
0.2.0	13.03.22	DM1306	Action AEW's recommendations
0.2.0	12.05.22	DM1306	Revise Specification

1 Product Overview

We are building an integrated environment for aiding the teaching of Computing in schools. It shall allow Users to create Multimedia User Interfaces - attaching scripts to each element on screen which are triggered on User actions, allowing for intuitive control and design of graphical applications -without concern for complicated toolkits - allowing Educators to focus on Principals whilst allowing students to create interesting and engaging projects.

2 UI

2.1 Base Behaviour

When the User opens the application, the main User Interface will be loaded and a default presentation will be displayed; the User will be presented with top menu-bar containing entries such as 'File', 'Edit', 'View'; a menu containing tools and object properties will be in the left of the window, and the presentation's initial view will be displayed in a larger section in the right hand side; at the bottom of the screen the user will be presented with an horizontal list containing all the Cards/Slides within the presentation, and below this with a bar for the display of non-blocking information messages; messages requiring User action, such as fatal unrecoverable errors, the user will be displayed in a console. Non-blocking messages should fade away after between 5 and 15 seconds.

A User may open a Presentation from disk by selecting the 'Open' option from the 'File' menu; this will open a file-chooser dialogue displaying directories and presentation files (if present), permitting the User to open the desired presentation. A User shall be able to save the displayed presentation to a file different to the initial one through a 'Save-As' entry in the 'File' menu, which will display a file-chooser dialogue on click permitting the User to choose the output file; a 'Save' entry will save the currently open file - if it is new, a file-chooser shall be displayed.

2.2 Elements

To add an Element to a presentation, be it a Card/Slide or any other Visual Element, the User shall select the desired Element from the Tools menu and either click on the desired location, or it shall appear in the relevant location on-click of the tool (dependant on the type of element or tool).

To edit an element's properties the User shall select the Editing tool, and then select the Element with her System's pointing device, and the Element's basic properties shall appear below the tool menu.

The User may select a button reading 'Edit Script' in the Element properties list - this shall open a simple text editor window to permit him to attach Code snippets to handle user actions on the Element.

Each element shall be automatically given a unique identifier; this shall allow for identification within the Application and Scripting environement. The User may change this Identifier manually to a more memorable name, however the user must then manually ensure it is unique.

2.2.1 Media

A Creator-User may decide if Playable Media Elements should display player controls and if they should automatically play on load, in the Element's properties. If the Player is enabled, simple Play/Pause, volume, and location controls will be drawn below the Element. Whilst a piece of image media is loading, the user should be presented with an indicator image; if the specified image is not found, there should be a "not found" indicator image loaded in its' place.

2.2.2 Console

A console window should be available for textually interfacing with the application. A User should be able to access this from the 'View' menu, or through the Scripting Engine when a User script triggers a console I/O event - at which point, it should become visible as a 'pop-up'.
It should contain a line for User input, and a large box containing previous input lines and outputs.

2.3 Navigation

When a User opens the presentation the UI shall display a list of Slides/Cards as stated above in §2.1. A Creator-User may also provide her own buttons for a custom navigation flow or appearance.

3 Files

3.1 User Files Provision

When the User requests that the Application load a given Presentation Stack as defined in §2.1, the Application shall load the Stack and its' assets from a *Zip Archive* with the Stack Description File placed at the top level. The Application shall place all media in sub-directories based on media type, and will ensure that no items have conflicting names or identifiers.

3.1.1 Stack Description File

A User's graphical Presentation Stack shall be transformed by the Application into XML, as specified by the Project Wide Standards (PWS) XML Schema Document.
The Application may include domain-specific XML Attributes or Elements that are not described in the PWS - these will be documented in the Application's Documentation.

3.2 Importable Media Formats

The Application shall be able to read and "work with" the following external data formats, upon request by the User from within the interface of the Application, or when specified in the Stack Description upon loading the Stack.
The User may request media be loaded from the network by specifying a non-local URI - the process shall be the same as for loading local media.
The User may import media by dragging the media-type's Element from the Tools box - once placed the user will be able to open a dialogue from the Element's basic properties to select the required file or specify a network URI; when adding local media presentation, the selected file is added to the currently open archive or directory, unless otherwise noted.

3.2.1 Video

When the User requests that a video be loaded, the application must immediately begin loading the video, provided it is contained in one of the following formats:

- MPEG-2

3.2.2 Audio

When the User requests that a piece of audio be loaded, the application must immediately begin loading the audio provided it provided it is contained in one of the following formats:

- Opus
- MP3

The User shall also be able to request that web audio streams, using an M3U8 stream description, be loaded.

3.2.3 Images

When the User requests that an image be loaded, the application must the application must immediately load the image provided it is of one of the following formats:

- JPEG
- PNG
- BMP
- GIF

3.2.4 Text

The User will enter text graphically within the application, via the Element properties, which will place the entered text directly into the relevant part of the Stack Description XML document.

For completeness however, the User may also include text from an external file containing XML formatted text following the inner format of the 'text' element of the Stack Description File (as defined in the PWS) placed inside of a 'stub' element. If the User provides text in such a "source file", it must be UTF-8 encoded with UNIX-style line endings. Files with DOS-style line endings shall not be guaranteed to work.

4 Scripting

As noted by Eric S. Raymond in "The Art Of Unix Programming", It is common for Users to expect to be able to extend the usefulness of programs through scripts. As such, the User will have the ability to associate a Script with the Stack, a Card, or an Element on a Card to run when a User makes certain actions, such as clicking an object on-screen. This shall allow the User to create more dynamic Card Stacks or even small visual applications, with complete access to the Stack data and Application functions from within the interpreter.

4.1 Language

A User may expect that the language used to extend the Application will be either already familiar or at least well-known, and would expect good documentation on the use of that language. As such, the User will be able to write scripts using Python.
JPython will interpret user Python scripts within the JVM giving the user access to typical Python functions, as well as functions defined in Java within the Application, providing domain-specific functionality and methods to easily operate on the presentation Stack.

4.2 Access

As previously mentioned, the Scripting Environment will have full access to the presentation Stack and also to public Methods that are available in the Application's 'Engine' (the main Controller, in User Interface design parlance). This will allow Scripts to have full control over the display of a presentation - at the most extreme, a user would have the ability to write a presentation entirely from this environment.
Less extreme is controlling animations, or interactive views of data, or adding new interactive elements directly on the presentation - like a microstrip-line calculator for an introductory EM-waves lecture.

4.3 Environment Scope

The Environment will have an hierarchical scope, with the first precedence of variables and functions given to the current Element's environment, then to the current Card's, and then finally to the Global scope; actions, such as clicks, can be thought of as 'bubbling' through from the clicked element to the top-level document.

5 Security

Users often expect their software to not contain interesting exploits, so we will put effort into using good secure design principles to ensure that the possibility and potential impact of any interesting vulnerability is limited.

5.1 Remote Control

To secure the Remote Control interface, all messages 'over-the-wire' shall be optionally signed using a PGP scheme using MIT's public key distribution system. This enables the Viewer-User's software to verify that the message is from the real, hopefully trusted, Presenter-User.

5.2 Script Engine

Embedding an unrestricted scripting engine means malicious activity is a possibility - although this is the case when running any third-party program.
Reducing risk from running arbitrary software on a machine is the subject of much

ongoing research and many PHDs, as such we are not yet going to attempt to tackle this in our product.

A Draft UI Layout

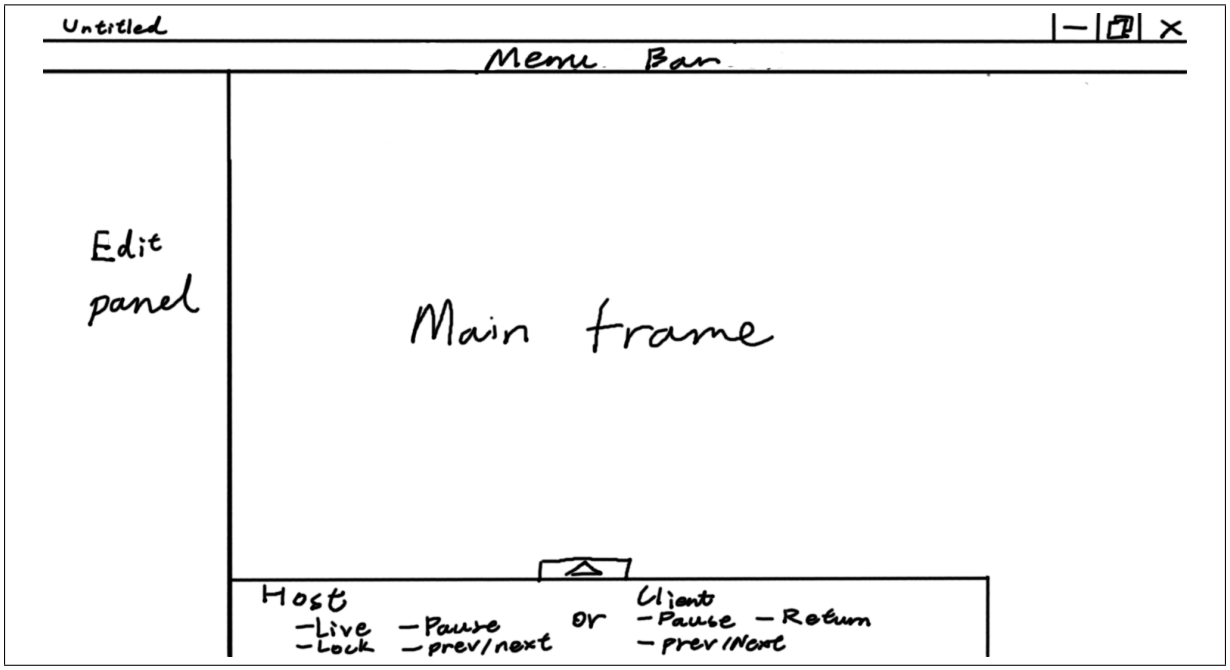

Figure 1: Draft Main UI Layout

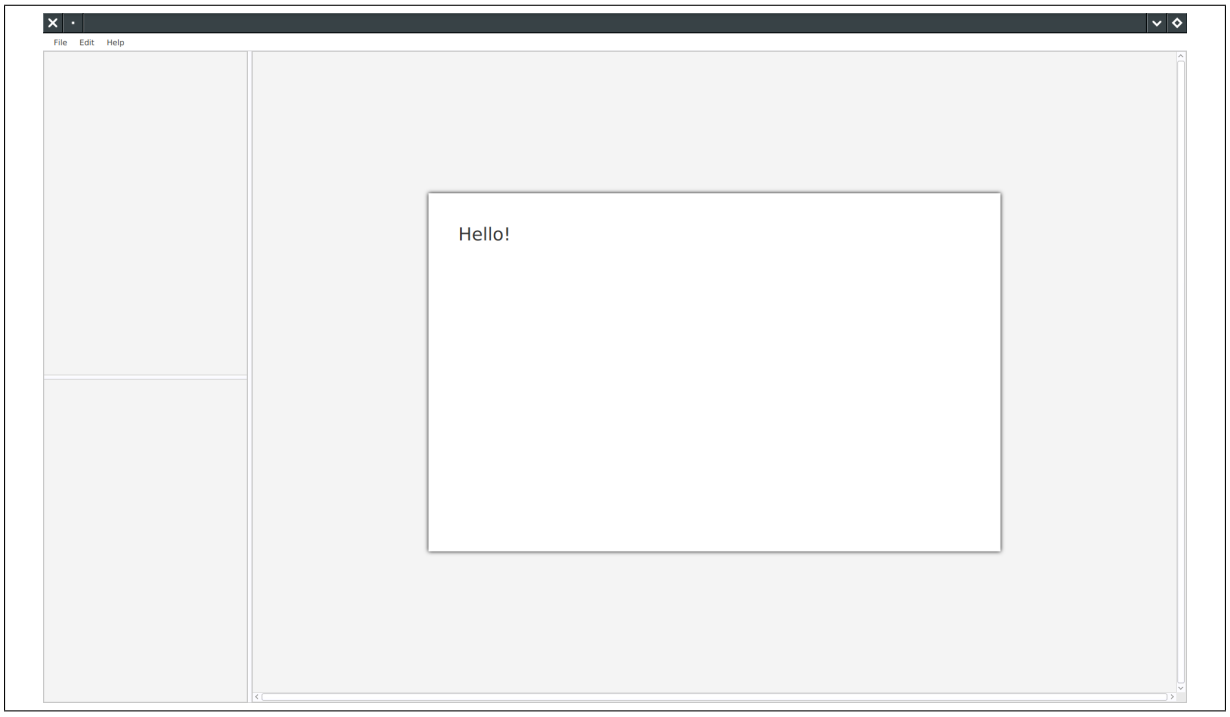

Figure 2: Simple Mock UI Layout