Image and Sound Programming for Web

Literary Programming for Web
Javascript, HTML, CSS, and Python

Władysław Skarbek

Warsaw University of Technology
Institute of Radioelectronics and Multimedia Technology

Joanna Napieralska [1]

The Fryderyk Chopin University of Music

[1]Co-author of the chapter: "Programming of Web Audio for $\mathbb{DC}^2$ client"

Contents

1 Introduction **4**

1.1 Application designer meets application programmer 6

1.1.1 He outlines the system requirements 6

1.1.2 They are talking on DC^2 layout and basic use cases 8

1.1.3 They discuss structure of DC^2 internal frames 9

1.2 Programmer thinks on DC^2 web application 11

1.3 `DC`^{2} Guide 13

1.3.1 WAP and WAD Reactivation 13

1.3.2 Meeting on Web Audio in $\mathbb{DC}^2$ 29

2 Python programming for $\mathbb{DC}^2$ server **41**

2.1 WAD and WAP debate on $\mathbb{DC}^2$ server 41

2.2 Programmer implements server `dc2.py` 45

2.2.1 Python and CherryPy tools settings 45

2.2.2 Collections of logins and wiki pages 46

2.2.3 HTML control templates 47

2.2.4 Access and saving rights 49

2.2.5 Media URLs and log function 50

2.2.6 Class `DC2` 51

2.2.7 Main Python code 55

3 Programming Web Audio for $\mathbb{DC}^2$ client **57**

3.1 Module `MProcessor` 59
3.2 `Web Audio` basic concepts 64
3.2.1 Audio programming for web browsers – short history 64
3.2.2 `Web Audio` API – an overview 65
3.3 Sound menu programming for AV tags 68
3.4 Synthetic sound creation 91
3.4.1 Oscillator 91
3.4.2 Piano 101

A $\mathbb{DC}^2$ Design and Implementation – main module 103

A.1 Programmer defines main page layout for DC^2 103
A.1.1 Page header 104
A.1.2 $\mathbb{DC}^2$ frames 107
A.2 Programmer designs and implements functionalities for DC^2 109
A.2.1 Overview of DU functionalities 109
A.2.2 Overview of EE functionality 110
A.2.3 Make outline of application `dc2.js` 111
A.2.4 Define global utilities 115
A.2.5 Define unit navigation 117
A.2.6 Implement pop-up menu 124
A.2.7 Event handling 128
A.2.8 Media handlers 135
A.2.9 Key down handler 138
A.2.10 Handling menu path 145
A.2.11 Handling functions for active areas 155
A.2.12 Functions for loading of DC2 resources 157
A.2.13 Media boards manager 165
A.3 Definitions for HTML templates and CSS styles 180
A.3.1 Nesting structure for templates and styles 180

A.3.2 Templates and styles for menus 182

B $\mathbb{D}\mathbb{C}^2$ Design and Implementation – EUniter module 209

B.1 `EUniter` module structure 209

B.2 Utility functions 214

B.2.1 Functions for code and data handling 215

B.2.2 Verification of DU tree structure 221

B.3 Functions for DU creation, insertion, and deletion 224

B.4 Handlers for items of DU menu 228

B.5 UNDO/REDO small engine 235

B.6 Handlers for items of BOX menu 237

B.7 Handlers for list boxes 244

B.8 API for clipboard 248

B.9 API for `Streams of Fragments` 267

B.10 API for `ECELL` 295

Chapter 1

Introduction

This book is the result of interdisciplinary work of people from computing, image and sound design areas. Our basic goal was to present contemporary computing tools for image and sound design using W3C standards.

Another not inferior goal is to show by a case study how such interdisciplinary work can be led by technology and art people in one big project with the emphasis on software design and implementation.

In order to achieve these goals we decided to show the complete process of developing `WEBSA` application.

`WEBSA` is an educational platform which offers for a user (student) access to expandable collection of WEBGL Shaders and Web Audio scripts. In its front-end WEBSA is featured by flexible user interface exclusively based on `HTML5` and its `Javascript` functionalities ready to use in the most popular WEB browsers. Contrary to typical web applications we do not use any additional framework, neither popular Bootstrap or JQuery for web page mastering, nor Three.js for WebGL programming, were chosen.

As the backend platform a lightweight `CherryPy` web server is used which provides direct access to `Python 3` rich set of tools for scientific programming and digital data management.

We present all steps of `WEBSA` creation using the dialog between few persons playing various roles. There are the following roles:

- `WEBSA` creators:

  SED: Sound Effects Designer

  SEP: Sound Effects Programmer

  IED: Image Effects Designer

  IEP: Image Effects Programmer

  WAD: Web Application Designer

  WAP: Web Application Programmer

- `WEBSA` users:

  SAT: Sound Art Teacher

  SAS: Sound Art Student

  IAT: Image Art Teacher

  IAS: Image Art Student

  MTT: Multimedia Technology Teacher

  MTS: Multimedia Technology Student

Yet another feature is worth to mention: the concept of literary programming. Namely, all software is developed using `ILP` (Integrated Literary Programming) – the `Javascript` library for `TexWorks` editor which makes possible to generate the code from the documentation written in `LaTex` and presented using `PDF` format.

We divided our work into four chapters:

1. `WEBSA` **Design and Coding**

2. **Introduction to** `WebGL`

3. **Introduction to** `Web Audio`

4. **Image and Sound Integrated Rendering**

In the context of digital art media projects the chapter could be of interest for people playing the roles of web application designers (WAD) and web application programmers (WAP). The reader follows the road leading from the initial concept of DC^2 web application which is a simple Wikipedia authoring system with facilities to Describe an educational theme, to Configure related applications, to Display their results, and finally to let the wiki readers to make Comments visible for authors (editors[1]) of the wiki notes and visible to other wiki readers.[2]

1.1 Application designer meets application programmer

1.1.1 He outlines the system requirements

WAD: I have arranged this meeting to introduce you to a new project. The goal is to design and implement an authoring web based tool supporting preparing of wiki like notes by academia people. They are going to teach some parts of digital media art which are based on `W3C` Internet standards.

WAP: Do you mean that image, graphics, and sound effects should be combined exclusively within `WebGL` and `WEB Audio` standards?

WAD: This will be the most desirable goal. However, as we know both standards are evolving and slowly converging to their PC application counterparts. Therefore, some effects which are now available in PC workstations, there are either still inferior when presented in web browsers or not available at all. Hence beside on-line effects achieved in the graphics and music cards locally where the browser is installed, it would be nice to present off-line image and sound rendering effects. I imagine that a reader could configure remotely an application which would be

[1]In principle the author and the editor are usually different roles of the system. However, in this project we do not distinguish them.

[2]Resulting $DCDC$ acronym is transformed to $(DC)^2$ and next incorrectly abbreviated to DC^2 to be confused with Einstein's energy formula mc^2.

running on the server side. The result in the form of a short clip could be streamed back to the reader.

WAP: Yes, it seems possible for `OpenGL` applications since `WebGL` is its close cousin. It means that the `WebGL` programmer can easily extend its program to the recent features of `OpenGL`. In case of sound effects I think it is more complex situation.

WAD: OK, let assume that in the versions 1.x of our application we offer only W3C tools. Let me to say something on the main idea of this Wikipedia project. Contrary to the available Wikipedia platforms we want to keep both the reader and the author interfaces as simple as possible. On the other hand on the server side we want to avoid redundancies for the software, image, and sound resources even if the same wiki page can be authored by many people.

WAP: To achieve this we need to constrain somehow the user freedom for creation of various page elements.

WAD: What you mean by constraints? Less freedom for authors or readers?

WAP: I mean here a spatial separation on the single wiki page the areas for description, for commenting, and for results of the related applications including their configuration panels. Free layout, i.e. mixing all of those semantic components is hard in programming and consumes a lot of code. In order to complete the work in a reasonable time we have to use external tools.

WAD: Most of programmers use the web packages like `JQuery`. Why we are going to "open the opened doors"?

WAP: In my opinion there are few good reasons to make it:

1. *Efficiency.* The external software is written for many general use cases – the generalizations takes time for loading (down to CPU memory), and space for data structures.

2. *Future developments.* The dependence on external libraries ties your project to another project which in case of future developments of your application could block some desirable design decisions.
3. *Educational aspect.* Since the planned software is to be used by students for their MSc interdisciplinary projects, having the full control on all its components is very desirable for educational purpose. Students could follow some design and programming patterns and add their own modules in their projects during the study and afterwards.

WAD: OK. The reasons are reasonable provided you can implement on your own the functionalities necessary for our project.

WAP: I hope so. Anyway from my experience, I prefer to look for and correct errors in my own code than in free html editors what once happened to be "the mission impossible."

1.1.2 They are talking on DC^2 layout and basic use cases

WAP: Let us talk now more on basic use cases regarding editors, and readers. It should help me to get the idea on the wiki page layout.

WAD: Yes, the readers are final main users of Wikipedia, but here they get possibility to run applications joined by authors to illustrate the presented theme. Moreover, the reader could become a kind of editor of its own setting and commenting actions, e.g. of the form: *I have found the setting of the parameter x to 1.5 results in a more impressive visual effect.*

WAP: Hence, for readers, we have at least four areas of their activities:

1. Description frame (to read)
2. Configuration frame (to read and write)
3. Display area for application results (to read, watch, and listen)

4. Comment frame (to read and write)

WAD: Very nice, the page layout exhibits the symbolic structure $DCDC = (DC)^2 \mapsto DC^2$. It is also a good name for our system, as well. Do you agree with me?

WAP: Yes, provided you will not add some new functionalities. Then, I will attempt to keep this layout as the page skeleton beside control elements like buttons and drop down lists which will drive the user interaction. More such controls will be needed by editors. Can we keep the same page layout for them?

WAD: Yes, it seems that DC^2 is good for authors as well. They describe the wiki theme in the *Description frame*, define attribute names and default values in the *Configuration frame*, run application, watch and listen their output in the *Display area*, and optionally put their comments in the *Comment frame.* As you have already mentioned the difference will be in greater number of control elements available for authors which should decorate each of DC^2 areas.

1.1.3 They discuss structure of DC^2 internal frames

WAP: In fact editors can put texts of various semantics into three frames: descriptions like in books, application settings like in data driven programs, and comments. It means quite different interaction for each frame, does'nt it?

WAD: Hm, interesting question. It refers to the internal structure of the material we put into the frames. For instance the descriptions can be written in the form of short notes organized hierarchically like in a chapter of the book: we have an introductory note which terminates with links to section notes which in turn terminates by links to subsections. Of course, in the Description frame only one note is presented. Others are available either by links at the end of the current note (to a sub-unit) or in the beginning (to a sup-unit).

WAD: Remember also that teachers like to give students a sort of handouts printed from PDF files. Therefore the conversion to PDF is necessary option.

WAP: It is important information as in this case a document tree could be handled by drop down lists of titles for notes which correspond to LaTex sections, subsections, etc. Latex formatting seems to be the best way to get PDF files for our Wikipedia.In fact we will need three drop down lists, one list of titles per one level of document tree. What about the Comment frame.

WAD: It could be organized by reader login identifiers, e.g. their email addresses. Then we get one drop down list for users, and the second one for the given user sorted by time of the given comment saving.

WAP: You think only about standard commenting. For me it is a sort feedback from users. For instance if the teacher joins in its comment note some questions or problems for his/her students in the context of the wiki page then other comments could be considered as answers to this questions.

WAD: Nice idea. However, we have to personalize the pages to get marking process fair.

WAP: Yes, the idea to be postponed to the DC^2 ver. 2.x.

WAD: I am sure you have document tree approach to the Configuration frame, as well.

WAP: You are right, like in the Description frame we have three levels: application , settings for application, group of parameters for setting. What will distinguish this frame from others is more rigorous way of editing for features and their values. Here we have a kind input fields, but I am not going to implement attribute handling by this kind of HTML elements. I prefer to use HTML tables as they help a lot in nice formatting.

WAD: Let us finish this meeting with a promise. Can you prepare a short demo of DC^2 layout for the next brain-storm meeting with participation of media designers and teachers?

WAP: I will try to do my best! However, I am not sure what kind of content for wiki page I could show to focus only on the general user roles: wiki author[3], wiki active reader[4], and wiki reader?

WAD: I think that the content should be neutral, and useful for participants of the meeting. They will be the potential users of `WEBSA` platform. I understand that DC^2 is planned to be a kind of engine for `WEBSA`?

WAP: Yes, shaders and audiers will be delivered to application scripts via the Configuration frame.

WAP: What do you think on a wiki page where DC^2 is presented and brief user guide is outlined?

WAD: This is perfect idea. This kind of page is necessary anyway. As far as I remember your idea is close to the design paradigm "User Guide Driven Software Development ".

WAO: Yes, perhaps it is a variant of "User Driven Software Design and Implementation".

1.2 Programmer thinks on DC^2 web application

Hm, the very interesting project [5] but with too many degrees of freedom. We get here:

- a number of user roles which are hard to predict now,

[3]He/she can edit the whole content including wiki notes, wiki application attributes, and personal comments.

[4]I mean, the active reader can comment and change values of attributes for applications.

[5]Here he just talks to himself.

- three activity frames (besides application panel),
- arbitrary height for document tree with units as its nodes,
- unlimited number of document units/notes on each level of tree,
- unlimited number of editing units in each document unit,
- flat or tree like organization of document units.

I need to find a unifying mechanisms to make it less complex. In order:

1. To unify roles I could use different sets of CSS values assigned for HTML elements. Then the drawing styles will depend on user roles implying different interface appearance for different user roles and some could be even invisible for one of them.
2. To unify handlers for document units and their editing units I could write a single Javascript class, say `DocUnits` which carries common operations in the same way.
3. To handle arbitrary number of levels, I should stack vertically the navigation elements with names of document units.
4. To copy with unlimited number of units in frame, I could switch style attribute `display` from `block` to `none` value showing only the selected document unit.
5. To handle inserting and removing of editing units, I could consider them as rows in a HTML `table` which are to be added or deleted. The row consists of one data element including an editable content (e.g. paragraph `p` or division `div` or another `table`). This is more robust way than the user defined selection followed by deletion.
6. To unify sequential (flat) and hierarchical (tree) organization for document units, I could consider the former as one level case of the latter structure.

It seems the are feasible concepts to be implemented in a reasonable time. I think that before brainstorm meeting I should prepare `DC`^{2} `Guide` and develop the API for handling editing units as primitive elements within document units which play also the role of visual units.

1.3 DC^2 Guide

1.3.1 WAP and WAD Reactivation

WAP makes excuses

WAP: After few months I have invited you to present the current state of DC^2.

WAD: Thank you for the invitation. As I remember you are supposed to prepare a user guide for wiki editors.

WAP: Excuse me, but my preliminary idea for the guide is different now. The system is now ready in alpha version, and I prefer to show you directly how to prepare a specific wiki page which is representative for multimedia. I will record my remarks, your questions, and my answers.I will make also screenshots on stages for our creative authoring process.

WAD: I understand that the idea of guide driven software development, similarly to test driven approach, is fine but only for standard web applications where work flow is known in advance and experience in the form of patterns, good practices, and supporting tools is widely accepted.

WAP: Exactly, $\mathbb{DC}^2$ system is novel. I decided not use any Javascript frameworks and write its code from scratch in pure Javascript 5 for Document Object Model (DOM) of HTML5. The system is now independent as the graphics user interface for web browsers, resulting in a lightweight authoring tool for wiki pages. Besides the standard multimedia content management, it exhibits a unique facilities for literal programming directly in the browser, and therefore the actual potential for design and programming of applications based on Web standards.

WAD: It sounds very well, but you have to convince me now. May be, let us start from the general concept of the wiki as the web document which implies a basic workflow for any $\mathbb{DC}^2$ editor.

WAP: OK, actually there are two basic concepts of wiki page in $\mathbb{DC}^2$ edition: *document unit* (DU and *edition element* (EE). Briefly $\mathbb{DC}^2$ wiki is an implicit tree of document units, and the document unit is a box-tree of edition elements.

WAD: Do they constrain the workflow for the authors/editors?

WAP: Yes, but to some extent. The work in $\mathbb{DC}^2$ is similar to writing an article. You express your knowledge in sections, subsections, etc. In $\mathbb{DC}^2$ when you open a new document unit you decide whether it is the sub-unit of the current visible unit or its follower/predecessor on the same document level. Once you get an empty unit, you fill it with edition elements, like text paragraphs, media elements (image, sound, movie), element lists, and code fragments. The edition is guided by a nested box layout.

WAD You mean it is also a kind of hierarchical 2d data structure overlaid dynamically over the edited frame?

WAP Yes it is known box layout, for instance Java Swing includes such option. As a matter of fact there are two types of box layouts: vertical (`VBOX`) and horizontal ones (`HBOX`). Each box in vertical layout can be split into boxes organized horizontally, and each box in horizontal layout can be organized into the vertical layout. As expected, the initial (top level) layout of the new unit is vertical. Splitting of boxes is an action selected from a pop-up menu and always results in creating `ECELL` – the empty cell which can be filled by a content or further split either vertically or horizontally.

WAD: Is "boxing" is only way of frame structuring?

WAP: No, in a way lists, and code fragments are additional tools for dividing cells in $\mathbb{DC}^2$. However, they are populated like any sequential data structure. In a sense, they behave like `VBOX` where each its element gets automatically a label. The label is unique wrt the list or the code fragment. While in the list the cell can be any $\mathbb{DC}^2$ object, even the nested list, in case of code fragments its elements are obviously, textual paragraphs, only.

WAD: OK. You propose here the top down strategy for wiki page design. I have to see some examples for better understanding. Firstly, show me how to edit a document tree with structure implied by the following unit titles:

```
TITLE of (COVER) PAGE
- TITLE of SECTION A
  - TITLE of SUBSECTION A.1
  - TITLE SUBSECTION A.2
- TITLE of SECTION B
  - TITLE of SUBSECTION B.1
  - TITLE SUBSECTION B.2
```

By the way. Is the title of $\mathbb{DC}^2$ document units must be unique in the wiki page?

WAP: No, the titles of units are not restricted. Contrary to other wiki editors, like `TiddlyWiki`, each unit is an object getting implicitly a unique identifier. It is more convenient for authors. Imagine a sub-unit title *Input Data Specification*. In $\mathbb{DC}^2$ it can be used as the title in few contexts of the wiki page without the uniqueness problem.

WAD: Nice, having in memory what you have said on box layouts, before you show me the mouse and the keyboard actions, I want to define a related exercise. Namely, after creating the above `DU` tree structure, please select any unit, say `A.1`, and make its "boxing" according to the following layout where the minus `-` is used as the nesting symbol. Read: `ECELL` is nested in `VBOX`, `HBOX` is nested in `VBOX`, `LIST` is nested in `VBOX` which is nested `HBOX`, `(2) ECELL` is the second element of `LIST`, `A. ECELL` is the first element of another `LIST`, etc.

```
VBOX
- ECELL
- HBOX
  - ECELL
  - ECELL
  - ECELL
- ECELL
- HBOX
  - ECELL
  - VBOX
    - LIST
      - (1) ECELL
      - (2) ECELL
      - (3) ECELL
    - ECELL
- LIST
  - A. ECELL
  - B. ECELL
```

[WAP:] OK. Nice exercise for "boxing".

WAP explains interactivity metaphors for HCI

WAP: Yes, before diving into pressing, touching, whatever of computer interactive devices, I should outline metaphors of $\mathbb{DC}^2$ interactivity.

WAD: Do you like to present me an instance of HCI domain (Human Computer Interaction).

WAP: Yes, you are right HCI is almost forgotten in Computer Engineering curricula, like Philosophy disappears from high schools, as well. What I mean by interactivity metaphor is simply expressed by three actions, being sequential or concurrent in time:

- `TOUCH`
- `EMIT SYMBOL`
- `EMIT MODIFIER`

WAD: Touching is obvious for touch-screen. What about regular screens?

WAP: It is simulated by combination of mouse device cursor, which points an object on the screen, and mouse button press. In $\mathbb{DC}^2$ if the mouse device has more than one button, the left one is used only. In this guide when the mouse button is to be pressed, I write `TOUCH`. It is optionally preceded by a modifier, e.g. the actions `ALT-TOUCH` mean for the regular screens: *if you press (left) mouse button and* `ALT` *key on the keyboard then the pop-up menu is displayed near the mouse cursor with menu items relevant to the pointed cell(s).*

WAD: I understand that `ALT-TOUCH` for touch-screen means parallel touching of screen and an `ALT` control screen area.

WAP: Exactly! I will decipher this for `SHIFT-TOUCH`: *if you press (left) mouse button and* `SHIFT` *key on the keyboard then the pop-up menu is displayed near the mouse cursor with menu items relevant to the pointed media object.*

WAD: `TOUCH` without a modifier means that the pointed (touched) object has only a single option to be selected, and then we do need a pop-up menu.

WAP: Again, you are perfectly right. There are few such cases in $\mathbb{DC}^2$. For instance starting edition of unit title is `TOUCH` at its area, while ending this edition is `TOUCH` outside of the title area. It is the special treatment applied only for unit titles as they are used in the navigation areas for `DU`s, and for their update we need to define event of kind *end of title edition.* More about navigation between $\mathbb{DC}^2$ units I will tell you solving exercises, you have given me.

WAD: What about `EMIT SYMBOL`?

WAP: It refers to the focused textual elements, e.g. edited paragraphs of regular text in a cell or code text in a code fragment. After you `TOUCH` such an element it changes its state to focused, and then all pressed (touched) symbols on a keyboard (physical or virtual device). The symbols can be modified, as well, and then the symbols can act as some special actions.

WAD: Be more specific.

WAP: OK. For instance `SHIFT-A` denotes as usually the capital `A` written at the text cursor in the focused element, while `CTRL-A` is the selection of all the textual content in the focused element. By the way, almost all popular key shortcuts used in text editors are valid in $\mathbb{DC}^2$. Namely, the full list follows:

- `CTRL-A` – select all text in the focused paragraph,
- `CTRL-I` – toggle italic font style, [6]
- `CTRL-B` – toggle bold font style,
- `CTRL-U` – toggle text underline,
- `CTRL-Q` – toggle text strike through,
- `CTRL-C` – copy the selected text to clipboard,[7]
- `CTRL-V` – replace the selected text by the content of clipboard (if no selection, the clipboard is inserted at the text cursor position),
- `CTRL-X` – cut the selected text into the the clipboard,
- `CTRL-Z` – undo textual action,
- `CTRL-Y` – redo textual action.

WAD: What about `ALT-A`?

WAP: In $\mathbb{DC}^2$ there is no action assigned to it. However, I have provided mechanism for assigning actions to sequences of symbols, and for instance `ALT-AB` means

[6] Toggling means here setting on/off the given font/text style option.

[7] There are three clipboards used in $\mathbb{DC}^2$: (a) the regular (used here) for texts written in paragraph elements; (b) element cell clipboard devoted for removed cells together with their contents, like image, sound, movie; (c) the document unit clipboard. Except the regular clipboard which is governed by the browser builtin editor, the other clipboards can keep any number of removed elements for possible removal undo and possible removal redo.

addition of a new code paragraph into the code fragment, just below the focused code paragraph.

WAD: I guess `ALT-AA` stands for addition of a code paragraph just above the focused one?

WAP: Perfect. Other examples:

- `TC` – shift the text to the center in the focused paragraph,
- `TL` – shift the text to the left margin of the focused paragraph,
- `TR` – shift the text to the right margin of the focused paragraph,
- `TB` – make the text font size bigger by 5% in the focused paragraph,
- `TS` – make the text font size smaller by 5% in the focused paragraph.

WAP develops the required tree of $\mathbb{DC}^2$ document units

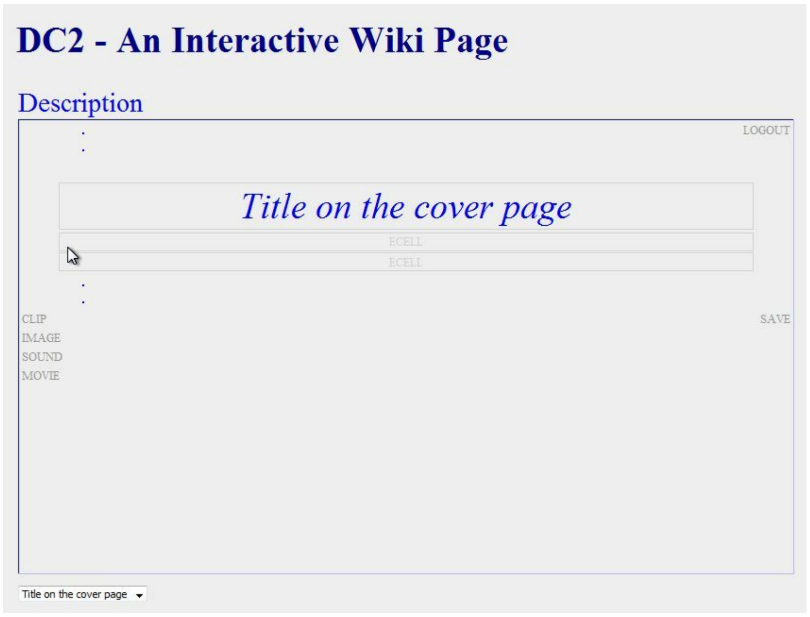
WAD: I assume that the empty Description frame includes some elements for edition and widgets for interaction?

WAP: Yes, it includes a box for title, two empty cells (boxes), and widgets ready for use:

- Left side (middle part):
  1. UNDO (initially hidden)
  2. REDO (hidden)
  3. CLIP
  4. IMAGE
  5. SOUND
  6. MOVIE
- Left bottom corner, just outside the frame:
  SELECTION (for current sibling units)

- Righ side:
  1. LOGOUT (top right corner)
  2. SAVE (middle)
- TOP NAVIGATOR (above the title, for links to ancestors)
- LOW NAVIGATOR (below unit content, for links to children)

WAD: OK, have a look. There is an initial layout:

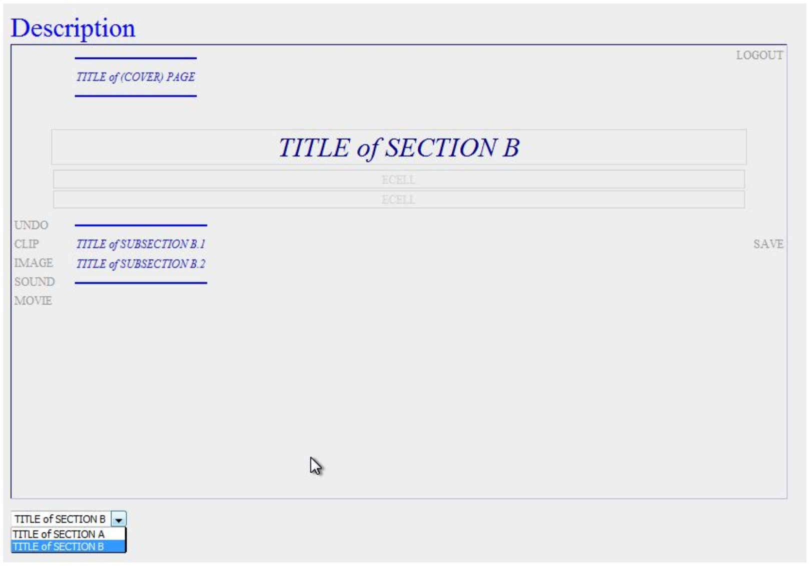


WAP: I perform the following actions:

1. edit title to: `TITLE of (COVER) PAGE`
2. <u>create new sub-unit</u> with title `TITLE of SECTION A`:
   (a) do `ALT-TOUCH` while pointing on the title
   (b) touch item `expand menu for DU`
   (c) touch item `empty first child of this DU`
   (d) edit title to `TITLE of SECTION A`
3. <u>create new sub-unit</u> with title `TITLE of SUBSECTION A.1` (as above for sub-unit)

4. create new sibling unit with title `TITLE of SUBSECTION A.2`:
   (a) do `ALT-TOUCH` while pointing on the title
   (b) touch item `expand menu for DU`
   (c) touch item `insert new DU after this DU`
   (d) edit title to `TITLE of SUBSECTION A.2`
5. using up top navigator, go to unit `TITLE of SECTION A`
6. create new sibling unit with title `TITLE of SECTION B` (as above for sibling unit)
7. create new sub-unit with title `TITLE of SUBSECTION B.1` (as above for sub-unit)
8. create new sibling unit with title `TITLE of SUBSECTION B.2`(as above for sub-unit)
9. using up top navigator go to unit `TITLE of SECTION B`
10. open SELECTION widget

WAD: OK, there is now the top navigator filled the parent unit title while the low navigator shows names of sub-units, and finally I see the titles of all units on `A` level. I see also appearance of UNDO widget:



WAP: OK, press `UNDO` till empty document is restored and next `REDO` till restore completely the document.

WAD: Fine, perfectly the tree structure with unit titles is restored.

WAP: Try now to remove the unit `SUBSECTION A.1` to the clipboard which is called *forest*.

WAD: OK. I go to this unit, and in menu I have `remove to forest`. I did it.But how I can undo it.

WAP: You said – press UNDO. However, if you want to restore your unit after some structuring work making UNDO is not wise as you loose your work. Then the way is somewhat complex:

1. go to the cover (top) unit,
2. open SELECTION widget and you will find the forest element(s) - they are siblings of the root DU (since the root unit has no siblings by the definition, this list can be used for hiding removed units),
3. look on menu on forestry unit - it can be removed to the trash, however then you will not recover DU at all - it is removed from DOM,
4. if you like to move it back to $\mathbb{DC}^2$ document tree then make the removed unit as the target for the next operations (menu item: `set target as this du`),
5. now go to the unit X where you like to insert the target one and do make this insertion, either before or after X unit.

WAD: I understand that such procedure (target and insertion) is used when I like to move any unit to another location in $\mathbb{DC}^2$ tree (not only from forest)?

WAP: Yes, this is the primary use of this mechanism.

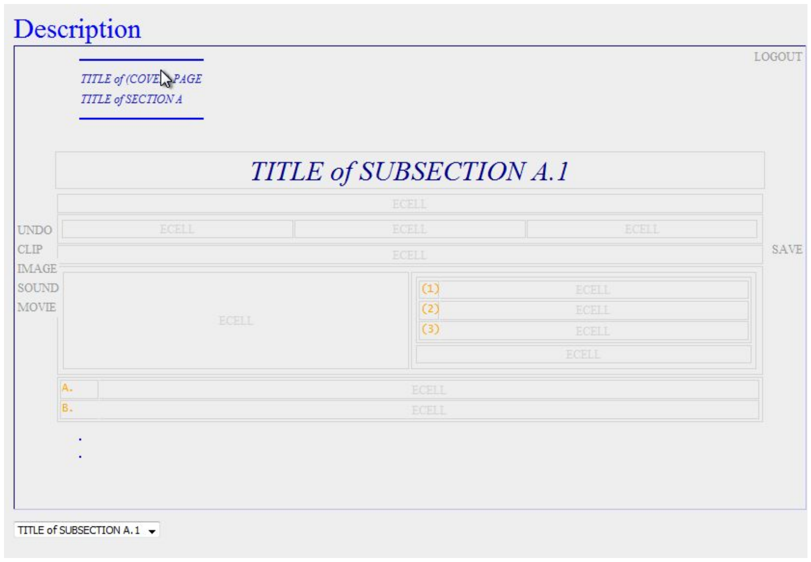
WAP develops the required tree of $\mathbb{DC}^2$ empty boxes

```
VBOX
- ECELL
- HBOX
  - ECELL
  - ECELL
  - ECELL
- ECELL
- HBOX
  - ECELL
  - VBOX
    - LIST
      - (1) ECELL
      - (2) ECELL
      - (3) ECELL
    - ECELL
- LIST
  - A. ECELL
  - B. ECELL
```

WAP: In order to get the above box layout for the unit `A.1`, we observe that in the target top `VBOX` we have 5 boxes. Initial layout of empty wiki contains `VBOX` with two cells B_1, B_2.

1. It means that we have to add three cells B_3, B_4, B_5: press three times menu item `expand menu for vbox` followed by `split cell down` touching any of them.
2. After that we split B_2 horizontally getting cells B_{21}, B_{22}, B_{23}: press menu item `expand menu for vbox` followed by `split cell right` (B_{21}, B_{22} are created) and next for B_{22} press menu item `expand menu for vhox` followed by `split cell right` (B_{23} is added).
3. We proceed to B_4 and split it horizontally getting cells B_{41}, B_{42}: press menu item `expand menu for vbox` followed by `split cell right`.
4. For the created cell B_{42} we apply vertical split to get cells B_{421}, B_{422} : press menu item `expand menu for vbox` followed by `split cell down`.

5. Create the list L_1 in the cell B_{421}:
   (a) We fill the cell B_{421} by the list L_1 object which initially consists of the single empty cell L_{11} labeled by default symbol $1.$: press `expand menu for ecell` followed by `set list box`.
   (b) We add to list L_1 elements L_{12}, L_{13} : press menu item `expand menu lbox` and next `add item after` while pointing on element's label - do this two times.
   (c) We change enumeration style for the list L_1: make key focus to the first element of list, i.e. L_{11} and enter (1) - after blurring this focus all other labels are changed automatically.
6. Create the list L_2 in the cell B_5: do the same actions as for L_1 but: (a) add only one new list element; (b) set the label for L_{21} as $A.$.



WAD: Uff, it is quite tedious clicking process. I have to make training on another example.

WAP: Yes, I do my best to make authors life easier. Pay your attention on box borders when `ALT-TOUCH` action is performed. The color sequence is fixed and it is assigned from the innermost box to uppermost box: red, green, cyan, yellow, magenta. The same colors you can notice assigned to menu items. Then author who likes to split higher level box can select the right menu item being guided by color symbols (small rectangles) standing at menu items. There is also orientation of rectangles saying whether we deal with vertical box layout or with horizontal box layout.

WAP fills empty $\mathbb{DC}^2$ boxes with media content

WAD: So far, we got $\mathbb{DC}^2$ tree of document units we can traverse and fill with and content and one of them containing $\mathbb{DC}^2$ tree of empty cells to be filled something. I am interested now how to transfer my media files (images, movies, and sounds) into those empty cells.

WAP: Yes, it is possible for any media format which is handled by the modern browsers. The lists for supported formats differ slightly between them. However, `jpeg, mp3, h.264` is a must now. There is also for some of them support for access to user media devices. You can listen for instance music on selected sound device or use your camera to record teaching materials based on $\mathbb{DC}^2$ or give a chance to perform face analysis.[8]

WAD: I understand that media files must be delivered to server before I can distribute them in $\mathbb{DC}^2$ units and their empty cells.

WAP: Yes, the procedure of distributing materials is based on catalog structure. Any editor/author registers wiki page he/she likes to create/edit and assigns the wiki to registered before group of wiki pages. The wiki groups are defined hierarchically and actually create a tree of folders. In each folder can be set media folders with fixed names `image, sound, movie.` The author of wiki loads media files

[8]However, the current $\mathbb{DC}^2$ server was not designed for camera related functionalities.

to the folder of the wiki he is authoring. However, he can get access also to media folders created on the path from $\mathbb{DC}^2$ root catalog to the given wiki catalog. For content of media catalogs assigned to groups of wiki materials there is person responsible, which gets the role of wiki group supervisor. For instance authors are students preparing their project reports in $\mathbb{DC}^2$. They have their media files, however they can also use some of files delivered by their project supervisor. Concluding, the mechanism used by $\mathbb{DC}^2$ server to deliver the list of media file urls uses the concept of *closure* used in Javascript compiler to bind value to a variable.

WAD: OK, I roughly know how media are delivered to $\mathbb{DC}^2$ server. However, how wiki authors can retrieve and select them?

WAP: Yes, the mechanisms is based on media boards. Each type of media has its own media board. For the author it looks like another page he/she can switch to by pressing the media button on the left edge of edited page: `IMAGE, SOUND, MOVIE`. There the author finds to areas: `Selected` and `Retrieved`. Browser downloads a part of available media objects into cells of `Retrieved` area. If there is more materials than can be displayed, the editor can scroll media cells using buttons `MORE, LESS`. `MORE` scrolls up and downloads the new row of media objects while `LESS` scrolls down showing objects previously hidden by scrolling up actions.

WAD: You mean that `Retrived` area shows all media available for the author while selected those he/she selects for transferring them into wiki cells. How to transfer objects between those areas and the wiki target cell.

WAP: Firstly, transferring from `Retrieved` and `Selected` areas, in both directions, is achieved by `SHIFT-TOUCH` action. You can make selection and de-selection many times in any time of editorial session. The transfer to wiki cell is preceded by touching an object located in `Selected`. Make `TOUCH` action only. Having source of the transfer, you switch fro media board to wiki page by pressing again media button, and then you go to the target cell of media transfer and use context pop-up menu: `ALT-TOUCH`, `expand menu for ecell`, `set media cell` where `media` is actually `image, sound,` or `movie`.

WAD: It sounds easy. Can I change my mind and put another picture, for instance.

WAP: If you change your mind immediately then press `UNDO` button. If later then use in menu item `move to media board`. If you like to copy/move the same media object into another unit of your wiki use clipboard to this goal. There are menu items `move to clipboard` and `copy to clipboard`.

WAD: OK, quite flexible mechanism of media shuffling. OK. let me show the media manipulation actions in practice.

WAP: OK, then issue commands for me using the cell identifiers I have assigned in the previous example. Start!

WAD: Fill by an image the cell B_{41}.

WAP: `IMAGE`, you like this one?, `SHIFT-TOUCH`, `TOUCH`, `IMAGE`, go to B_{41}, ..., `set image cell`, ready!

WAD: Assign music to list elements L_{21}, L_{22} located in B_5.

WAP: `SOUND`, you like this ones?, `SHIFT-TOUCH`, `TOUCH`, select another `SOUND`, go to L_{21}, ..., `set sound cell`, the same for the second selected, ..., ready!

WAD: Put also an available screen shot for sound analysis as the third item if list L_1, i.e. item L_{13}.

WAP: OK, actions as before for the image in B_{41}

WAD: Transfer a movie to the cell of list item L_{13}.

WAP: OK, actions as before for the images but selection from movie board.

WAD: Assign camera to the cell of list item L_{11}.

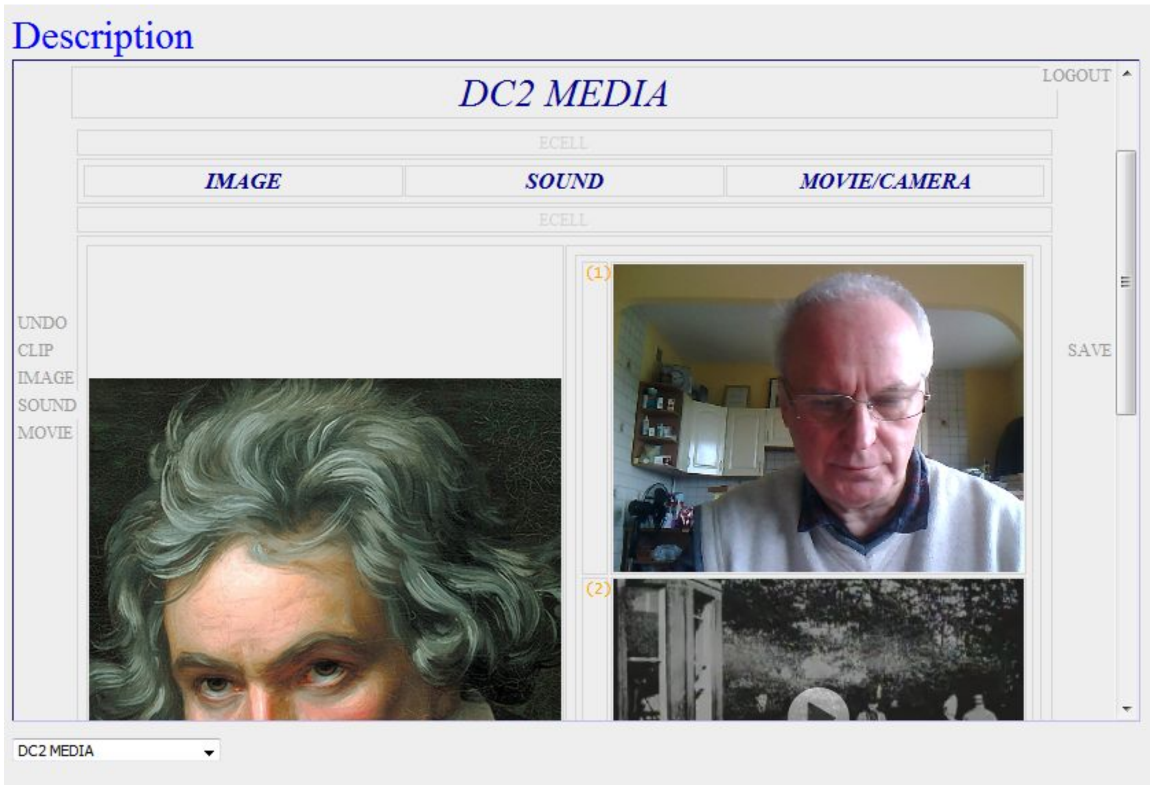
WAP: It is just `ALT-TOUCH`, menu item `expand menu for ecell`, and menu item `set live media.` The Firefox browser gives you dialog window to select video/audio input you like use. You select camera from the list, and finished. Now you are observed by your wiki!

WAD: Finally put centrally enlarged words, in bold and italic style. Let it be `IMAGE` into B_{21}, `SOUND` into B_{22}, `MOVIE/CAMERA` into B_{23}, and `ANALYSIS` into B_{422}.

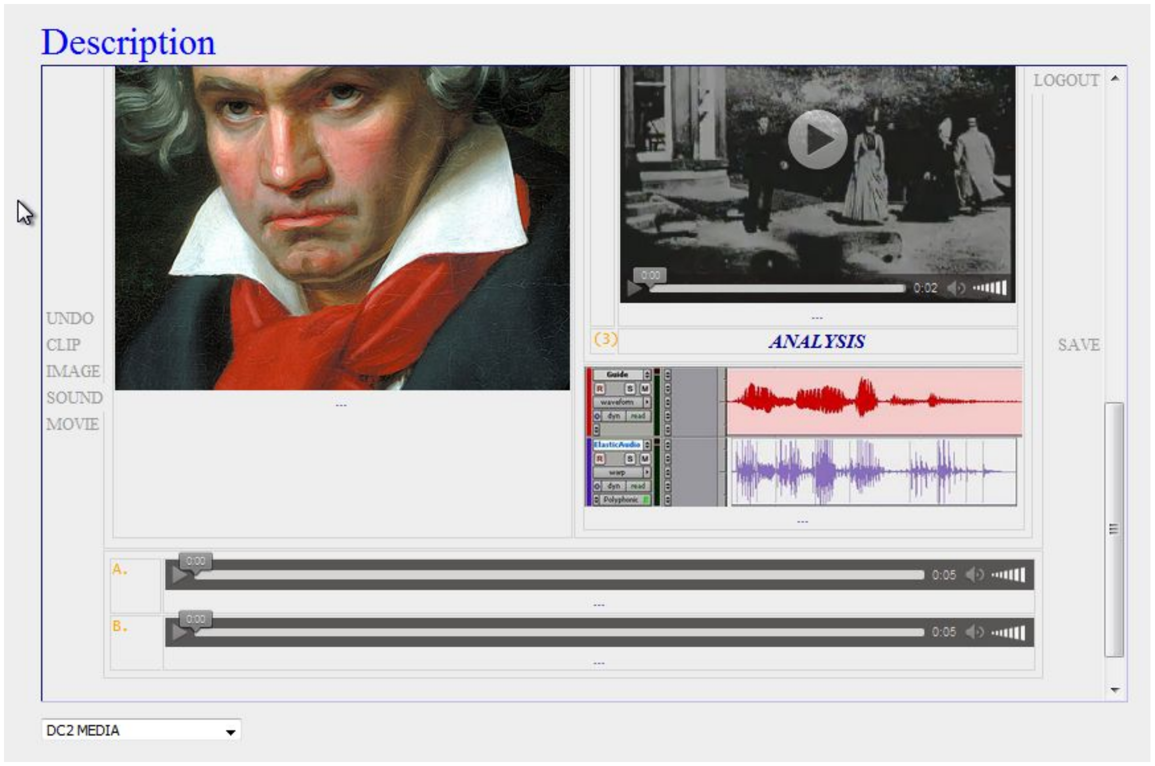
WAP: OK, I will describe actions for word `IMAGE:` focus on B_{21} centering: keys `ALT-TC` enlarging: keys `ALT-TB` few times, by 5% each time italic: keys `CTRL-I` bold: keys `CTRL-B`

WAD: Nice, you have shown me nice rich media editor for wikis!

WAP: Here we have the upper part of *Description* frame:



WAP: Tere we have the lower part of *Description* frame:



1.3.2 Meeting on Web Audio in $\mathbb{DC}^2$

The goal of this meeting is the presentation of $\mathbb{DC}^2$, its general features as interactive wiki editor with the focus on its sound features based on `Audio HTML5` element and `Web Audio` built-in JS library. The participants should get a working knowledge how to prepare simple wiki pages with sound effects.

Therefore the following roles were invited by `WAD`. [9]

WAD: Web Application Designer – he/she chairs the meeting and is responsible for general structure of $\mathbb{DC}^2$.

WAP: Web Application Programmer – he/she is responsible for programming $\mathbb{DC}^2$ and its `Web Audio` interface.

[9]Roles does not mean persons. It could be for instance two persons.

SAT: Sound Art Teacher – he/she likes to prepare a wiki page titled *Introduction to Web Audio*.

SED: Sound Effects Designer – he/she interested in features of the developed `Web Audio` interface to port some effects from sound workstations to web browsers.

SEP: Sound Effects Programmer – he/she supports SED by his/her programming skills

Before the meeting they were acquainted with $\mathbb{DC}^2$ by reading the minutes of the talk between `WAD` and `WAP` registered in the previous section of $\mathbb{DC}^2$ Guide.

WAD: Thank for interest in our $\mathbb{DC}^2$ project on developing of a novel wiki authoring tool. The main added value of$\mathbb{DC}^2$ in my opinion is giving to skilled readers more control on media processing and playing. Comparing with existing tools we offer a simple web platform for experimenting with sound and image effects. I pass my voice to WAP to say more on philosophy and practice of our approach.

WAP: The philosophy is great word but as you know before any complex software is created it is conceived on some paradigms, strategies, good practices, and previous experience of its authors. In this case this something is the concept of literary programming, used for the first time by Donald Knuth in 1980s when he was developing TEX formatting system. Briefly, the literary programmer writes a free style documentation of the developed software together with code fragments interspersed within their descriptions while traditional programming separates system documentation from its code which is extended by sparse coding comments. The obvious benefit of literary programming is natural synchronization of system documentation with its code. Having code requirements coupled with code implementation makes less probable that the former is outdated.

WAD: Obviously, for practical programming in such way we need tools for automatic integration of code fragments immersed in natural language text.

Yes, this is the point. To this goal Knuth had developed a special language, btw he called it WEB. It was an extension of Pascal and despite a web page established in 1990s, the concept is shibboleth. I have encountered the idea more than five years ago and from that time, from time to time, I was developing still new tools for literary programming based on extensions of LaTex. In 2013, I wrote JS library (called `ILP` - Integrated Literary Programming) for `TexWorks` editor and from this time all my educational and research software (mostly in Python) I implement in ILP. Now the $\mathbb{DC}^2$ application was fully written using `ILP`. Moreover, $\mathbb{DC}^2$ became also a web editor for literary programming, not only in Javascript. Today, for the goal of this meeting, I will present just one of possible applications for `ILP` using $\mathbb{DC}^2$, i.e. controlling sound processing in `Web Audio`. However, I would like to get interaction with you.The best if you ask me some detailed questions on $\mathbb{DC}^2$ you are interested, and next on the basis of my answers you could develop together with me small examples on sound processing in $\mathbb{DC}^2$ which will be guiding prospective wiki authors on $\mathbb{DC}^2$ platform.

SAT: As a teacher of sound art I am really excited by web potential for teaching. I have read on `audio` HTML5 tag, as a sound player couples with a playing controller. Can I attach in $\mathbb{DC}^2$ to such controller filtering functions of `Web Audio`?

WAP: Yes, $\mathbb{DC}^2$ offers several sound processing functions available from pop-up menu.Just make `SHIFT-TOUCH` and you can assign `Web Audio Context` what opens full access to `Web Audio` tools.

SED: Does it mean that like in sound workstations panels for audio parameters are displayed if a processing option is chosen?

WAD: The $\mathbb{DC}^2$ approach is different. Instead of GUI for parameters we have code fragments which can be edited even during sound playing with immediate effect without player pausing.

WAP: What is important, the author can write several independent code fragments for the same processing option which can applied in turn by users for any of sound records available in the page. Moreover, readers may modify those parameters and for simplicity code fragments do need to be complete, e.g. it can modify only the filter characteristic frequency, without writing for instance `Q` factor value.

SEP: It is really interesting. Is there any constraint on placement of code fragments wrt to sound controllers.

WAP: Not at all. Code fragments can be written even as the part of another `DU` - document unit which invisible when the sound player is controlled. However, it is more convenient to have parameters along the controller and having a copy of them in any place gives no overhead for $\mathbb{DC}^2$ editor.

SEP: Then how the controller knows which parameter set (you say code fragment) should be selected?

WAP: We use the concept of active or target stream of code fragments. Controller refers to the parameters which are set in the active code fragment. To change the target stream, simply touch the item `assign code stream` and select the new stream from a pop-up selection widget.

SEP: Then what happens if I select from the menu a filter which is not handled by the active code fragment?

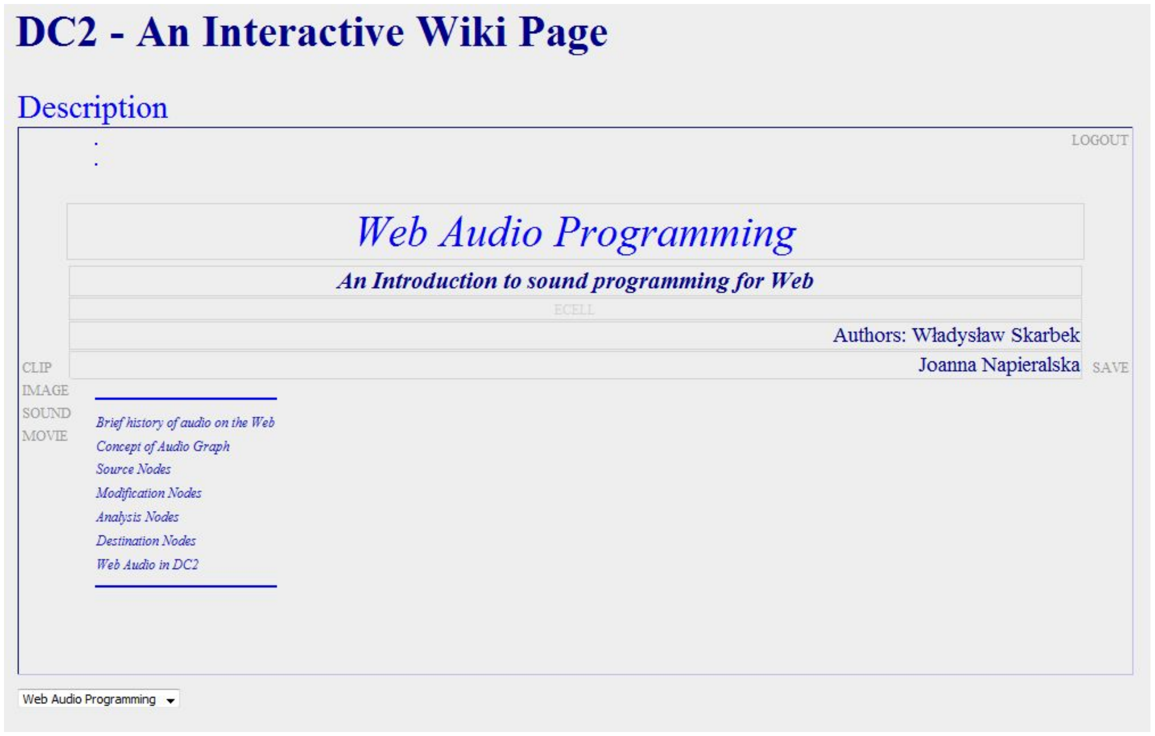
WEP: The matching of filters to parameters is checked by the filter name placed in the first line of code fragment. If no match then the filter selects the default parameters.

SAT: Is there any support for assessment of objective difference between the original and processed sound?

WAD: For the time being we have a visualization of spectrum for both signals: the original along the processed one. By eye view you can observe differences in signal spectrum domain. We are opened for other proposals from your side.

SAT: I think that we have got enough theory to proceed towards some practice. Now, I would like to learn how to:

1. select audio materials,
2. place them in a $\mathbb{DC}^2$ cell,
3. attach `Web Audio Context` to an audio element,
4. apply built-in filters to them,
5. modify parameters for the given filter,
6. detach `Web Audio Context` from an audio element.



DC2 - An Interactive Wiki Page

Description

LOGOUT

Web Audio Programming

An Introduction to sound programming for Web

ECELL

Authors: Władysław Skarbek
Joanna Napieralska

CLIP
IMAGE
SOUND
MOVIE

SAVE

Brief history of audio on the Web
Concept of Audio Graph
Source Nodes
Modification Nodes
Analysis Nodes
Destination Nodes
Web Audio in DC2

Web Audio Programming

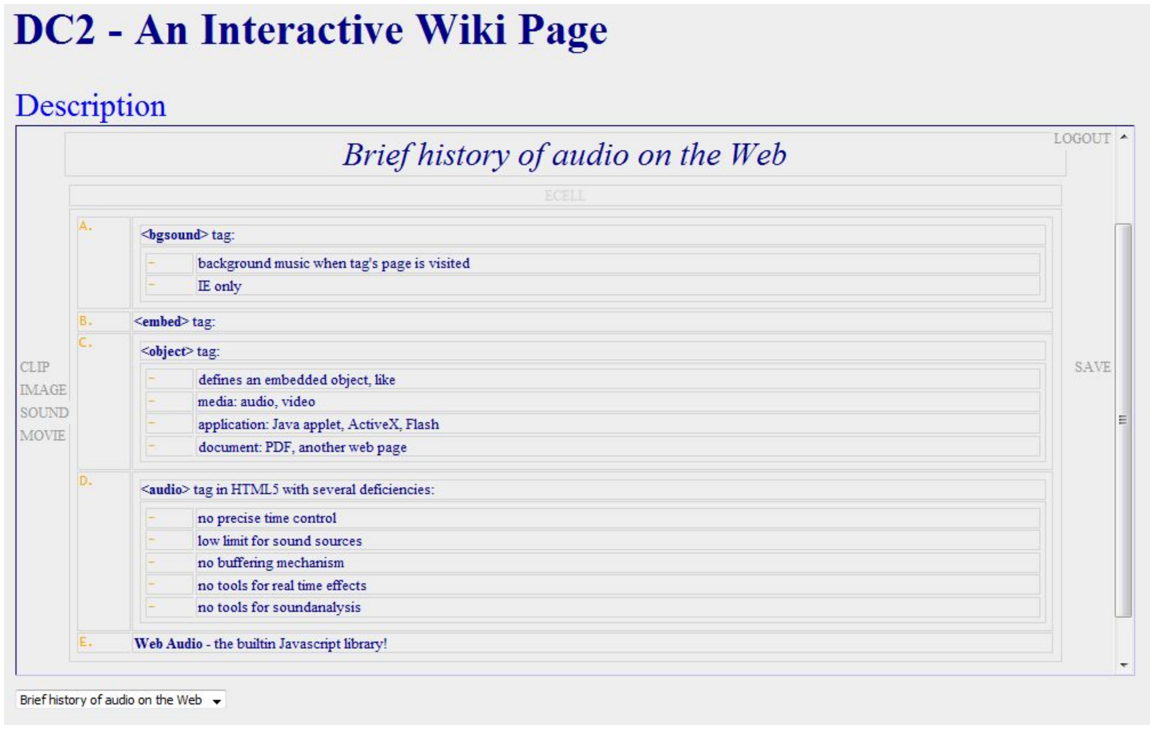
WAD: For the first two edit actions I advice to follow the small guide before the meeting.[10]

SAT: Yes,I have made the lesson and prepared for my students few units of a wiki on `Web Audio`. Don't you mind to help me in preparing programming details for this wiki.

WAP: With pleasure. OK, I see nice cover page with the content of the wiki.

SAT: The contents, as you know, is generated automatically using titles of all sub-units of any unit. For the cover unit, too. Other units need your cooperation.

WAP: Let see the historical note. It seems it needs some media enrichment? OK, I will do this as a final touch.



DC2 - An Interactive Wiki Page

Description

Brief history of audio on the Web

A. <bgsound> tag:
- background music when tag's page is visited
- IE only

B. <embed> tag:

C. <object> tag:
- defines an embedded object, like
- media: audio, video
- application: Java applet, ActiveX, Flash
- document: PDF, another web page

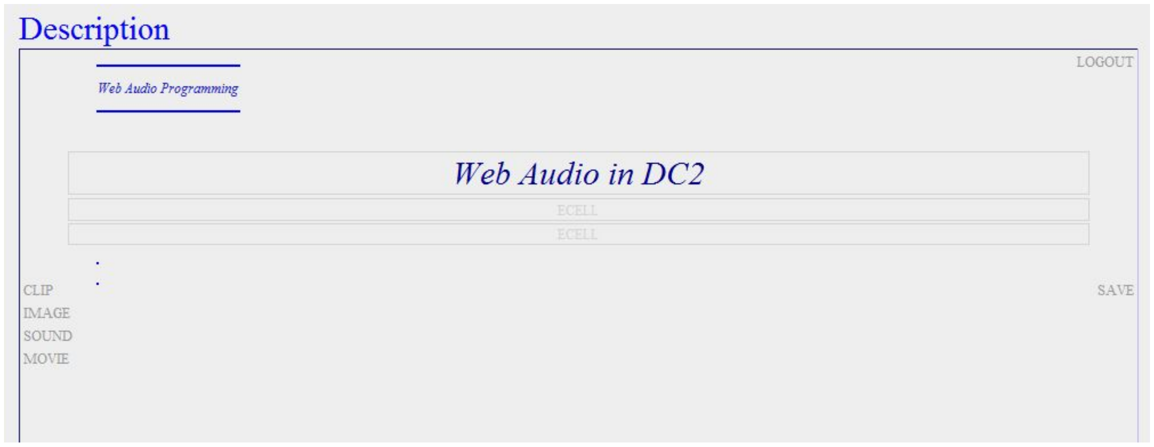
D. <audio> tag in HTML5 with several deficiencies:
- no precise time control
- low limit for sound sources
- no buffering mechanism
- no tools for real time effects
- no tools for soundanalysis

E. Web Audio - the builtin Javascript library!

SAT: Proceed to *Web Audio on DC2* unit. It is empty now. I have an idea to fill it: in the preamble of this unit we will give two examples. One for a short audio sample

[10]Included in the previous section of this guide.

and another one for a longer audio sample. The student could interact with them before reading in the sub-units a description how to prepare such interactivity. Obviously, including the programming aspects in the literary style, as well.



WAP: Good idea, but have you prepared the audio clips in $\mathbb{DC}^2$ image catalog.

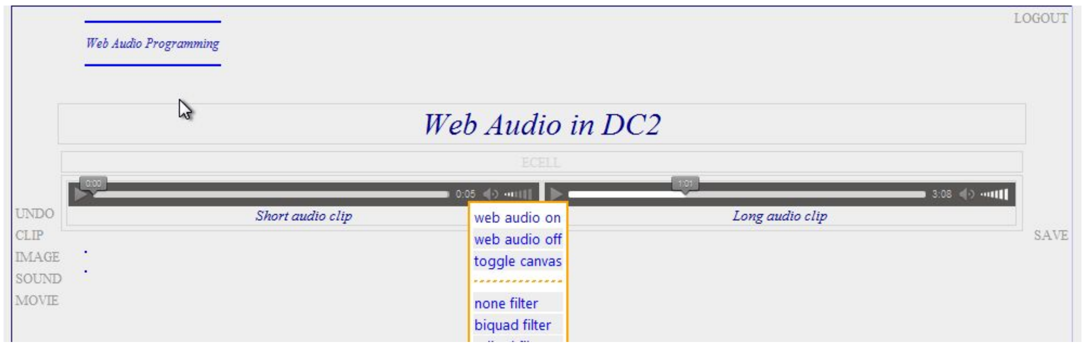
SAT: Yes, but for my `localhost:8080` server.

WAP: OK, for testing it is the good choice. BTW for the future you should know that $\mathbb{DC}^2$ server enables "semi-group editing". Namely, if an editor with rights to the given wiki, makes login before another editor of the same page, the next editor gets wiki with red `SAVE` button what means that saving action is not possible. Getting the rights for saving requires the logout from the page by the current owner of `SAVE` button, and refreshing the wiki page by his/her coworker. It is not "true group editing" but it was compromise between lock mechanisms on the level of the whole page, and lock at each editing elements. The latter mechanism needs completely different design of the whole application and it seems on this stage is superfluous. The current solution makes the second editor the regular reader who can experiment with content, run the wiki scripts. However, the role of reader excludes saving the results of experimenting at $\mathbb{DC}^2$ server side. The only difference is that the regular reader never gets `SAVE` button visible while the inactive red button says to the editor: "OK, you may play with wiki, run its scripts, change parameters, even write code fragments, however it will never be saved. You keep the pen with evaporating ink. Once your coworker makes logging out, you can refresh the page and from then you keep the pen with persistent ink."

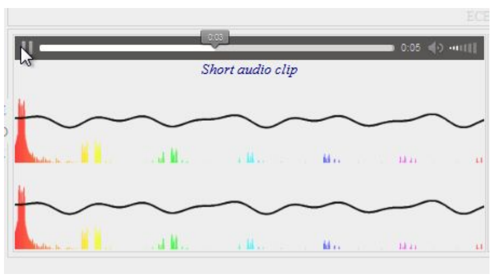
SAT,SED,SEP: It is interesting, but could you help us to fill "Web Audio in DC2".

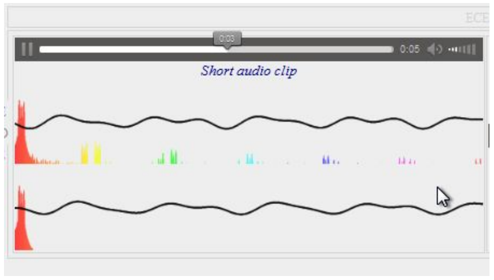
WAP: As you have read the section on media manipulation $\mathbb{DC}^2$ **??**, we will do the first two tasks by yourself under my guidance, only:

- go to sound board: `SOUND-TOUCH`,
- move the clips of your choice from `Retrieved` to `Selected` area: `SHIFT-TOUCH`,
- mark that shorter clip: `TOUCH` (orange frame should appear around the audio controller),
- go back to the unit: `SOUND-TOUCH`,
- split the second cell: ... item `split cell right`,
- insert the marked clip in the left cell: ... item `set audio cell`,
- write any caption under sound controller (in place of dots) for a test play the clip,
- repeat the same for the second clip:
  - go back to the sound board and mark the longer clip,
  - insert the marked clip in the left cell: ... item `set audio cell`,
  - write any caption under sound controller (in place of dots) for a test play the clip.
- assign `Audio Web Context` to the short clip: ...item `web audio on` while touching the controller near timer:

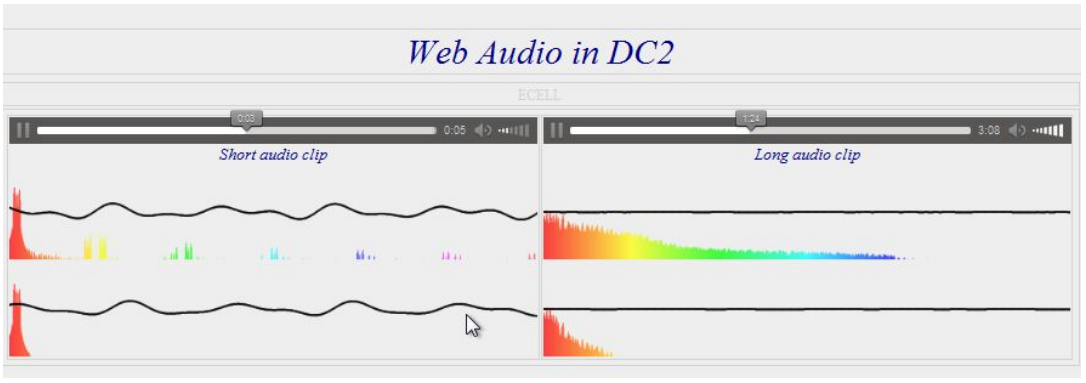


- play with `biquadratic filter` and spectral analysis: ...item `biquad filter` while touching the timer: You could compare the spectral analyser when none filter is used. The upper graph shows the spectrum of the audio source while the lower graph after the filtering. The left picture shows graphs when the web audio is set on and none filter is used. Actually the gain node is defined with `gain=1`:





- repeat the same for the second clip, but here play also with switching off the filters: ...item `none filter`. You can play both clips in parallel:



SAT: What about programming sound playing?

WAP: It is simple – just few lines in Javascript which. For setting parameters you do need to know Javascript. However,you should know:

- How to `start new stream` of code fragments?
- How to join `new code fragment` to the active stream?
- How to link menu item for filter call with its parameter code?
- How to `assign code stream` to switch between different parameter setups?
- What are names of the given filter attributes to be changed?

SED: I guess that most of them are available from pop-up `ALT-TOUCH`?

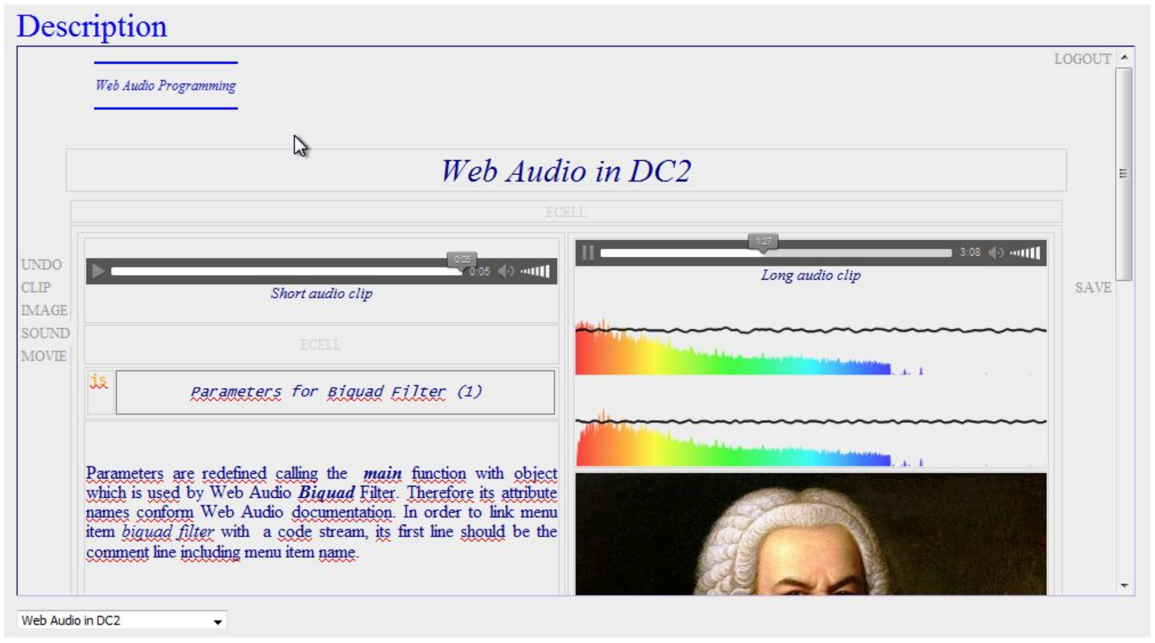
WAP: Yes, besides the knowledge of the filter attributes which needs `Web Audio` documentation)`https://www.w3.org/TR/webaudio/`), we should know that filter parameters are redefined calling the *main* function for object which is used by Web Audio Filter (e.g. `Biquad Filter`). Therefore its attribute names conform Web Audio documentation. In order to link menu item, e.g. `biquad filter` with a code stream, its first line should be the comment line including menu item name – in our example `// biquad filter`.

SAT: Go ahead, perform the clicking.

WAP: BTW, I will show the literary programming in action.

After half an hour of mouse clicking and screen tracking and commenting, the meeting members reached the following form of $\mathbb{DC}^2$ unit on *Web Audio Programming*:

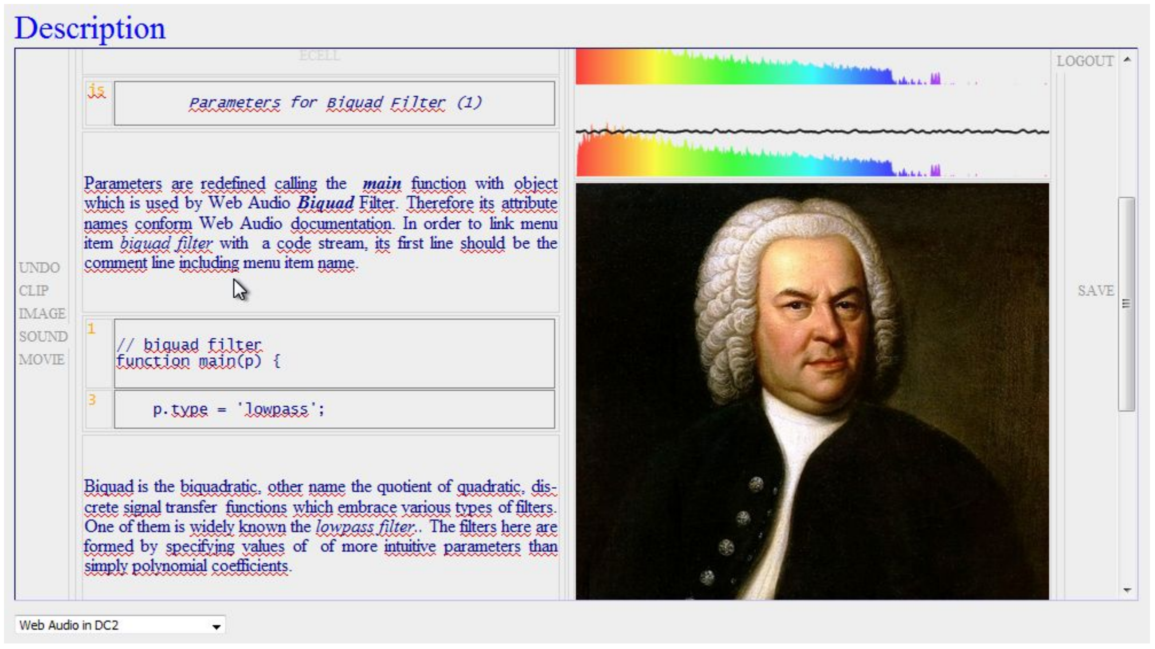
- Frame content upper part:



```
Parameters for Biquad Filter (1)
```

Parameters are redefined calling the ***main*** function with object which is used by Web Audio ***Biquad*** Filter. Therefore its attribute names conform Web Audio documentation. In order to link menu item *biquad filter* with a code stream, its first line should be the comment line including menu item name.

- Frame content middle part:



```
Parameters for Biquad Filter (1)
```

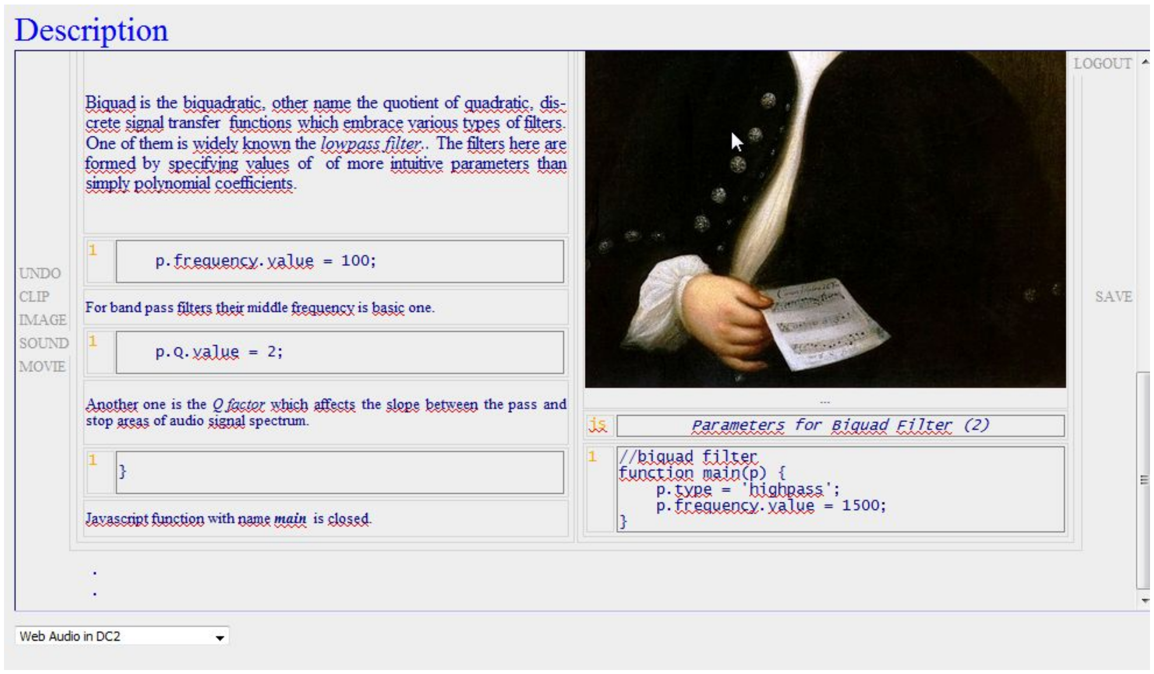
Parameters are redefined calling the ***main*** function with object which is used by Web Audio ***Biquad*** Filter. Therefore its attribute names conform Web Audio documentation. In order to link menu item *biquad filter* with a code stream, its first line should be the comment line including menu item name.

```
// biquad filter
function main(p) {

    p.type = 'lowpass';
```

Biquad is the biquadratic, other name the quotient of quadratic, discrete signal transfer functions which embrace various types of filters. One of them is widely known the *lowpass filter*.. The filters here are formed by specifying values of of more intuitive parameters than simply polynomial coefficients.

- Frame content lower part:



Description

Biquad is the biquadratic, other name the quotient of quadratic, discrete signal transfer functions which embrace various types of filters. One of them is widely known the *lowpass filter*.. The filters here are formed by specifying values of of more intuitive parameters than simply polynomial coefficients.

```
p.frequency.value = 100;
```

For band pass filters their middle frequency is basic one.

```
p.Q.value = 2;
```

Another one is the *Q factor* which affects the slope between the pass and stop areas of audio signal spectrum.

```
}
```

Javascript function with name ***main*** is closed.

js *Parameters for Biquad Filter (2)*

```
//biquad filter
function main(p) {
    p.type = 'highpass';
    p.frequency.value = 1500;
}
```

Web Audio in DC2

Chapter 2

Python programming for $\mathbb{DC}^2$ server

2.1 WAD and WAP debate on $\mathbb{DC}^2$ server

WAD: Today I would like to discuss with you an outline of web server for our $\mathbb{DC}^2$ application. I know that your experience on making web page servers is limited to standard data base applications. In this project we deal with multimedia documents management where their content is created not on the server side, but exclusively on the client side.

WAP: Yes, I agree that it is the traditional web forms downloading, updating, and uploading where the data source and data target is handled by a database management system. I think before we decide how to store $\mathbb{DC}^2$ documents we should define primary users and their roles.

WAD: Firstly, the roles with respect to the given $\mathbb{DC}^2$ page:

1. Editor can register the new wiki page, create its full content and save it.
2. Registered reader:
   (a) can read $\mathbb{DC}^2$ wiki in public domain and in the domains he/she is registered to, use processing elements (like audio filters), and modify parameters for them,
   (b) can save the changes in his/her $\mathbb{DC}^2$ private repository on the server side.

3. Non registered reader:
   (a) can read $\mathbb{DC}^2$ wiki in public domain, use processing elements (like audio filters), and modify parameters for them,
   (b) cannot save the changes.

WAP: I would like to explain saving for registered readers. For the first entry into a $\mathbb{DC}^2$ page, server will make its private copy. What about next entries? Creating each time a new private copy is nonsense. Replacing the previous one is bad as well. I think the registered reader should work on its private, all the time.

WAD: I have been thinking on that and while having the permanent private copy is the reasonable design decision, there is problem of updates for the original $\mathbb{DC}^2$ wiki page. Suppose project supervisor has to change some points in his document and students already have private copies for its previous version. How to follow such updates.

WAP: I think at next visits of the page we should ask the registered reader what is his actual target – the original document or private copy. Moreover, at saving the original document while there exists already created private copy $\mathbb{DC}^2$ server should ask whether really the reader likes to replace his private copy with the currently updated original one.

WAD: It sounds reasonable. Obviously,saving updates of private copy does not imply any question.

WAP: Yes. However, thinking on project use case there are some doubts. In case of school assignments changes are rather against regulations. Students will score it in the course secrete questionnaires. In small companies such changes are normal practice as they have to be flexible.

WAD: Transparently we touch the aspect of target users. I think that $\mathbb{DC}^2$ is potentially flexible. They are three frames for edition. The editor could decide which frame is for exclusive use for registered readers.

WAP: He/she could do this but not in the current $\mathbb{DC}^2$ version where only *Description* frame includes media boards. Other two are only for configurations and comments.

WAD: OK, you are right, the primary mission of $\mathbb{DC}^2$ is education not business projects. The target users are educators as editors and students as reader.

WAP: Having specified roles and potential users we can proceed to the problem of $\mathbb{DC}^2$ server load by wiki materials and their registered users. I mean each editor and each contributes reader to wiki files and wiki media. Average storage requirement is the product of: number of users with rights for saving for the given wiki times the average media storage per wiki times the number of wikis handled by the server. It seems that we have to impose some constraints to make $\mathbb{DC}^2$ affordable for our target users.

WAD: Yes, we should not assume that $\mathbb{DC}^2$ will be widely accepted by an university and will cover all curricula of thei department. I think that we should write a lightweight server which can be run bu users (teacher, student) locally by localhost port with synchronization to remote server. The scaling of capabilities of remote server should be left under supervision of a local $\mathbb{DC}^2$ community (e.g. multimedia teachers and students in the relevant department division).

WAP: I like this idea, as then we can consider platforms written for Python programmers where server configuration tools are compact (just few lines) and interpreting `HTTP` requests is performed by exposed methods.Parameters of the exposed methods directly correspond to `URL` attributes.

WAD: Are there are some which are recommended by programmers?

WAP: `CherryPy` has got good opinion. There are some convincing multimedia applications.

WAD: Then OK let us plan the server's structure.

WAP: Yes,for the time being I would like to postpone the discussion on user logging, choosing his/her role, and selected the wiki page. In the future the complete logging functionality could implemented as the separate *logging server* which transfers the access electronic stamp to worker servers specialized to specific tasks.

WAD: Then we assume that the server of wiki page receives request with a `URL` including attributes like:

- `wiki=` – the short name of wiki page which translated by $\mathbb{DC}^2$ server to its path
- `me=` – the login stamp which embraces user id and its role id
- `kind=` – index of frame content, e.g. 0 for *Description*, 1 for *Configuration*, 2 for *Comments*.

WAP: I think I will need one more `by=` to distinguish the source of `POST` request between `LOGOUT` and `SAVE` buttons.

WAD: What about media files. How `URL` for them looks like?

WAP: I think the static `URL` will simplify the loading problem.

WAD: What about the initial download of wiki page, so called `index` method of access.

WAP: I will send template `HTML` which will be filled by scripts on the client side.

WAD: How the JS scripts will be downloaded?

WAP: There are two ways:

- through the script element located in `dc2-main.html`,
- through static `URL`.

WAD: Then the scripts will request the contents of $\mathbb{DC}^2$ frames and they will post back the updated content.

WAP: Yes exactly, actually we have just outlined main structure of $\mathbb{DC}^2$ server.

WAD: Do not forget about server logs.

WAP: Yes I will use a typical approach with two files `access.log` and `error.log` which will be handled in rotated manner. The server at midnight will close them, rename, and opens the new pair of `.log` files for the next 24 hours.

2.2 Programmer implements server `dc2.py`

2.2.1 Python and CherryPy tools settings

Importing tools

```
#-*-encoding: utf8 -*-

import os, re
import logging
from cherrypy import _cplogging
from logging import handlers
import cherrypy
from host import host,port,pool,log2data,wiki2pages
```

Servers absolute path

```
HERE = os.path.dirname(os.path.abspath(__file__))
```

CherryPy configuration

```
config = {
    'global' : {
    'server.socket_host' : host,
    'server.socket_port' : port,
    'server.thread_pool' : pool,
    },
    '/': {
            'tools.sessions.on': True,
    },
    '/static': {
        'tools.staticdir.on': True,
        'tools.staticdir.dir': os.path.join(HERE),
    },
}
```

2.2.2 Collections of logins and wiki pages

The rights to wiki pages are defined indirectly by sets of labels. Wikis get labels as well, If there is nonempty intersection of wiki labels with user labels then the user has rights to access the page. Login name is the key to other user attributes in `lg2data` Python mapping. Here user set of labels cannot be `None`.

```
fragment_names = ['description.html','configuration.html',
                  'comments.html']
logins = log2data.keys()
```

The access rights do not mean editing rights. The login name must be on the list of editors of the page – then he/she can save their editing work.

The acronym of wiki page is the key to page attributes in `wiki2pages` Python mapping. The attributes used:

1. `path` – relative path to wiki page,
2. `editors` – the list of login names for this page editors,
3. `active` – index (in the above list) of the active editor,
4. `labels` – labels assigned to the page.

```
wikis = wiki2pages.keys()
```

2.2.3 HTML control templates

HTML header

In the HTML header of the $\mathbb{DC}^2$ main page, the language, title, character encoding type, and shortcut icon, are defined. They can be changed before page uploading.

```
mheader = '''
<!DOCTYPE html>
<html lang="{:s}">
<head>
    <title>{:s}</title>
    <meta charset="utf-8"/>
    <link rel="SHORTCUT ICON" HREF="static/cherrypy-32x32.png">
'''
```

HTML trailer

In the HTML trailer of the $\mathbb{DC}^2$ main page, the stamp of user and wiki is registered and edit state, as well. Edit state is computed by `canSave` function.

```
mtrailer = """
<script>
var _wiki = '{:s}';
var _me = '{:s}';
var _editState = {:d};
var _lang = '{:s}';
var mainTitle = document.getElementById('main-title');
mainTitle.textContent = '{:s}';
</script>
</body>
</html>
"""
```

HTML messages

Messages handle various cases:

- No access rights:

```
noAccessMessage = """
<div>
<H3>User '{:s}' has no access rights to read</H3>
<H2>DC2 wiki: '{:s}'</H2>
</div>
"""
```

- Unavailable page:

```
noPageMessage = """
<div>
<H3>User '{:s}' logged to read</H3>
<H2>Unavailable page(s): '{:s}'</H2>
</div>
"""
```

- Unregistered login:

```
noUserMessage = """
<div><H3>User login '{:s}' is not registered yet!</H3></div>"""
```

- No saving rights:

```
noSaveMessage = """
<div>
<H3>User '{:s}' has no saving rights for</H3>
<H2>DC2 wiki: '{:s}'</H2>
</div>
"""
```

2.2.4 Access and saving rights

Finding access rights

The labels are extracted from login and wiki mappings and next intersected. Empty intersection is detected by its cardinality returned by `len` function.

```
def noAccessRights(me,wiki):
    if me not in logins: return True
    if wiki not in wikis: return True

    mlabels = log2data[me]['labels'];
    wlabels = wiki2pages[wiki]['labels'];
    return wlabels and len(wlabels.intersection(mlabels))==0
```

Finding saving rights

There are three states detected in editing context:

0: either wiki is not available or the user is not on the list of editors,

2: if there is another active editor for the page,

1: otherwise.

```
def canSave(me,wiki):
    if wiki not in wikis: return 0
    if me not in wiki2pages[wiki]['editors']: return 0
    indx = wiki2pages[wiki]['editors'].index(me)
    active = wiki2pages[wiki]['active']
    editor = wiki2pages[wiki]['editors'][active]\
             if active!=-1 else 'Mr. Nobody'
    cherrypy.log('active editor at page download: '+editor)
    if active!=-1 and active!=indx: return 2
    return 1
```

2.2.5 Media URLs and log function

Media URLs are build according the rules:

1. three separate lists of media file paths are created according to the types `image`, `sound`, `movie`,
2. in the final list paths are separated by symbol |,
3. the list of media paths combines all files which are located in the folders with name `image`, `sound`,`movie`
4. the media folders are accessed from the folder of wiki, and next by the closure in each ancestor folder on the way to the `wiki` root folder.

```
def generateMediaUrls(rpath):
    cfolders = ['']; cfolders.extend(rpath.split('/'))
    mNames = {'image':[],'sound':[],'movie':[]}
    sfolder = 'static'; hfolder = HERE
    for cfolder in cfolders:
        if len(cfolder)>0:
            sfolder = sfolder+'/'+cfolder
            hfolder = os.path.join(hfolder,cfolder)
        for mfolder in ['image','sound','movie']:
            smfolder = sfolder+'/'+mfolder
            hmfolder = os.path.join(hfolder,mfolder)
            if os.path.exists(hmfolder):
                names = os.listdir(hmfolder)
                pnames = [smfolder+'/'+name for name in names]
                mNames[mfolder].extend(pnames)
    return '|'.join(mNames['image'])+';'+\
           '|'.join(mNames['sound'])+';'+\
           '|'.join(mNames['movie'])
```

The `error.log` is filled by server messages which are send by the application. We assume that `msg` argument contain formatting specifiers in correspondence to the list of the remaining arguments `args`.

```
def logit(msg,*args):
    #print('+'*60,'\n','-'*60);
    cherrypy.log(msg.format(*args))
```

2.2.6 Class `DC2`

`DC2` constructor

When server starts it reads the whole `dc2-mian.html` and `dc2-templates.html` documents.

```
class DC2:
    def __init__(self):
        fp = open('dc2-main.html',encoding='utf-8');
        self.mains = fp.read() ; fp.close()
        fp = open('dc2-templates.html',encoding='utf-8');
        self.templatess = fp.read(); fp.close()
```

Method `index`

Requests from the browser issued by our scripts extend server address by a virtual path followed by a sequence `key=value&` attribute assignments.

The virtual path is linked by CherryPy to an exposed method for `DC2` – the server main object.

Since `index` exposed method is default, request's `URL` does not contain the word `index` as the virtual path.

CherryPy changes multi word attribute values into lists. Therefore here we restore original names. To be sure on `utf-8` encoding we define content type attribute according `HTTP` standard.

```
    @cherrypy.expose
    def index(self,wiki='Test Page',me='anonymous'):
        if type(me)==type([]): me = ' '.join(me)
        cherrypy.response.headers['Content-Type'] =\
                'text/html; charset=utf-8'
```

Login `me` must be in the collection of logins, and `wiki` must be in the collection of wikis. If both do not match `no access message` is returned:

```
        if me in logins:
            if type(wiki)==type([]): wiki = ' '.join(wiki)
            if wiki in wikis:
                if noAccessRights(me,wiki):
                    msg = "User '{:s}' and wiki '{:s}' do not match"
                    logit(msg,me,wiki)
                    return noAccessMessage.format(me,wiki)
```

The header for wiki page is updated using language info and wiki full name.

```
                lang = wiki2pages[wiki].get('lang','en')
                name = wiki2pages[wiki].get('name',wiki)
                header = mheader.format(lang,name)
```

The editing state is found and if editor has been logged in when the other editors is not active, the server register him/her as active editor. Next the trailer of main document is completed and the whole document sent out.

```
                editState =  canSave(me,wiki)
                if editState==1:
                    indx = wiki2pages[wiki]['editors'].index(me)
                    wiki2pages[wiki]['active'] = indx
                trailer = mtrailer.format(wiki,me,editState,lang,name)
                user = 'Editor' if editState>1 else 'Reader'
                msg = user +" '{:s}' gets access to page '{:}'"
                logit(msg,me,wiki)
                return header+self.mains+trailer
```

In case of unavailable wiki page:

```
            else:
                msg = "Logged '{:s}' to read unavailable page(s): {:s}"
                logit(msg,me,wiki)
                return noPageMessage.format(me,wiki)
```

In case of wrong login name:

```
        else:
            msg = "Failed to accept login '{:s}'"
            logit(msg,me)
            return noUserMessage.format(me)
```

Method `getcontent`

Multi word attribute values are restored:

```
    @cherrypy.expose
    def getcontent(self,me='anonymous',wiki='Test Page',kind='0'):
        if type(me)==type([]): me = ' '.join(me)
        if type(wiki)==type([]): wiki = ' '.join(wiki)
        method = cherrypy.request.method
```

Absolute path to wiki frame content is established:

```
        fname = fragment_names[int(kind)]
        path = os.path.join(HERE,wiki2pages[wiki]['path'],fname)
```

In case of no access rights:

```
        if method=='GET':
            cherrypy.response.headers['Content-Type'] =\
                'text/html; charset=utf-8'
            if noAccessRights(me,wiki):
                msg = "Reader '{:s}' and wiki '{:s}' do not match"
                logit(msg,me,wiki)
                return noAccessMessage.format(wiki)
```

Register in log file and return the frame content:

```
            msg = "Reader␣'{:s}'␣is␣getting␣frame␣{:s}␣of␣page␣'{:s}'"
            logit(msg,me,kind,wiki)
            return open(path,encoding='utf-8');
```

Method `postcontent`

In case of logout:

```
    @cherrypy.expose
    def postcontent(self,me='anonymous',
        wiki='Test␣Page',kind='0',by='pressing␣button'):
        if by=='window␣unloading':
            wiki2pages[wiki]['active'] = -1
            msg = "-->Editor␣{:s}␣logged␣out␣of␣page␣'{:s}'<--"
            logit(msg,me,wiki)
            return 'OK'
```

Correct multi word arguments:

```
        if type(me)==type([]): me = '␣'.join(me)
        if type(wiki)==type([]): wiki = '␣'.join(wiki)
        method = cherrypy.request.method
```

Find absolute path to the content file to be saved:

```
        fname = fragment_names[int(kind)]
        path = os.path.join(HERE,wiki2pages[wiki]['path'],fname)
```

In case of `SAVE` button of $\mathbb{DC}^2$ frame the saving is performed in the binary mode (not to spoil unicodes).

```
        if method!='GET':
            indx = wiki2pages[wiki]['editors'].index(me)
            if indx>-1: #and wiki2pages[wiki]['active']==indx:
                txt = cherrypy.request.body.read()
                fp = open(path,'wb'); fp.write(txt); fp.close()
                msg = "-->Editor␣{:s}␣saved␣frame␣{:s}"
                msg += "of␣page␣'{:s}'␣by␣'{:s}'<--"
                logit(msg,me,kind,wiki,by)
                return 'OK'
```

Method `templates`

Templates established in the constructor are returned on browsers requests to initialize frames filling.

```
    @cherrypy.expose
    def templates(self):
        return self.templatess
```

Method `mediaurls`

According the path assigned to wiki acronym all media URLs found on wiki closure are generated as the Python string and returned by CherryPy.

```
    @cherrypy.expose
    def mediaurls(self,wiki='Test␣Page'):
        if type(wiki)==type([]): wiki = '␣'.join(wiki)
        if cherrypy.request.method=='GET':
            return generateMediaUrls(wiki2pages[wiki]['path'])
```

2.2.7 Main Python code

Here log options are set.

```
if __name__ == '__main__':
    cherrypy.config.update({'log.screen': True,})
    appLogHandler = handlers.TimedRotatingFileHandler(
                    'log/access.log', "midnight", 1)
    #appLogHandler.setLevel(logging.DEBUG)
    appLogHandler.setFormatter(_cplogging.logfmt)
    cherrypy.log.access_log.addHandler(appLogHandler)

    appLogHandler = handlers.TimedRotatingFileHandler(
                        'log/error.log', "midnight", 1)
    #appLogHandler.setLevel(logging.DEBUG)
    appLogHandler.setFormatter(_cplogging.logfmt)
    cherrypy.log.error_log.addHandler(appLogHandler)
```

The quick start of server is executed.

```
    cherrypy.quickstart(DC2(),'/',config)
```

Chapter 3

Programming Web Audio for $\mathbb{DC}^2$ client

WAP invites SEP to develop Web Audio tools for $\mathbb{DC}^2$

WAP: As my knowledge on Web Audio is rather shallow, do you agree to help me in developing $\mathbb{DC}^2$ module supporting Web Audio on $\mathbb{DC}^2$?

SEP: With pleasure, learning Web standards, like Web Audio and WebGL, by programming directly on pages is great idea, not only from educational point of view. Users like to have greater control on multimedia effects. Moreover, it enhances user experience, the fundamental feature of any man machine interface.

WAP: Could you outline for me some examples which are primary for Web Audio programmers?

SEP: Yes, I expected such kind of questions and prepared a list of topics:

1. Spectral filtering of sound for various audio sources,including `AUDIO` and `VIDEO` elements of `HTML5`.
2. Sound volume control, i.e. audio signal gain, in various scenarios, including cross-fading effect applied for play-lists.
3. Graphical analysis of the achieved effects with comparison to input signals.
4. Proprietary web page element `tones` built on the basis of `CANVAS` and `Web Audio Oscillator` including "one click" control of all its parameters.

5. Sound spatialization effects when user can experience the virtual motion.

WAP: Is it order you advise to proceed with software production?

SEP: I think that since we have already the multimedia tags on $\mathbb{DC}^2$ page, it is natural to get the sound from them and then stream audio to Web Audio Graphs which will be created according the processing algorithms we are going to design. The "note" element is an independent audio source, and it could be implemented before or after filtering and gaining. However, if we have processors and analyzers for tag sources we can adopt them to any other sound sources. Finally, the sound spatialization will be more convincing if we integrate it with some `WebGL` application. Therefore it could be postponed.

WAP: What about other audio sources and "sinks", like microphone and system `5.1`?

SEP: It is a promising area for programming, but it is more related to `WebRTC` standard. However, both standards are nicely coupled and we can consider the following examples where they cooperate:

1. Synchronized mixing of video, audio effects with out of band audio track.
2. Capturing microphone input, adding effects, and streaming through peer connection including options for spectral analyzer, background music, recording, and possible remote uploading.
3. Receiving audio from peer connection, mixing with spatialization effects, and playing.

WAP: Very interesting! However, referring to peer connections, be aware that it is still experimental technology for W3C, and as far as I know only Firefox browser provides a stable implementation.[1].

[1]`https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection` accessed on February 4th, 2016

3.1 Module `MProcessor`

WAP: We begin the definition of our module for audio processing. In order to reduce intermodule communication I suggest to put all media processing functionality into one module. I will call it `mprocessor.js` as beside some utility functions offered for the main module, it will export groups of functions, one for each kind of media: `audio, image, video, canvas.`

SEP: What about `canvas`? Is it for graphics effects.

WAP: Right. Here, mainly `WebGL` will work.

`MProcessor` setup

Module external and internal variables

WAP: I want to describe you the methodology I develop $\mathbb{DC}^2$ module: each $\mathbb{DC}^2$ module has its quasi-constructor in the form a function warping all the module's code. For instance the module `mprocessor.js` has the function `MProcessor(doc,win,config)` which is called from the `init` function, defined in `DC2` module, which is turn called for each $\mathbb{DC}^2$ frame.

SEP: You mean "quasi" since it is called only once. In fact it is a constructor for a singleton class.

WAP: Yes in case of `mprocessor.js` we get the single instance as we assume that media elements are manged only in the `Description` frame. In case of module `euniter.js` we have three instances – one for each frame. However, the main reason for the prefix "quasi" is no attempt to define functions of the module as members of `prototype` object.

SEP: I think it is no deficiency, at all. It is the advantage of simplicity since the JS closure mechanism does the binding job for you. In case of many instances you can get some memory savings. Writing classes for one or three instances seems unnecessary. Moreover, I can imagine how simpler is `undo/redo` functionality for implementation.

WAP: Yes, I have found this immediately in my programming practice. For further defense of module approach I can tell you that `JQuery`, probably the most popular, and huge, JS package had been written according this methodology. To conclude the software production methodology aspect let me outline the structure of my module quasi-constructor:

1. external declarations (definitions are postponed to `initXXX`) function,
2. global internal declarations,
3. definition of function `initXXX` consisting of:
   - definitions of all externals by extracting values from `config` object (received from the main module),
   - definitions of all or part of internals,
4. for each group of functions:
   - the definition of mapping from function name to function object,
5. return of mapping from group names to group objects.

```
/*
 * DC2 - Interactive Wiki Edit Platform
 * Author: Vwadec Skarbek
 *
 * The MIT License (MIT)
 * Copyright 2015-16, Wladyslaw Skarbek, WUT
 *
 * Licensed under the MIT License
 * http://opensource.org/licenses/mit-license.php
 *
 */
```

```
/* License JS */

function MProcessor(doc,win,config) {
    "use␣strict";
    // declare externals
    var fromStreamCodeToMenuFunction;

    // declare global internals
    var targetAudio =  null, convolverBuffer = null;
```

Constructor for MProcessor

```
    function initMProcessor() {
        fromStreamCodeToMenuFunction = config['code-for-menu'];

        if (doc.id=='Description') {
            win.requestAnimFrame = (function(){
            return  window.requestAnimationFrame        ||
                window.webkitRequestAnimationFrame ||
                window.mozRequestAnimationFrame    ||
                window.oRequestAnimationFrame      ||
                window.msRequestAnimationFrame     ||
                function( callback ){win.setTimeout(callback,50/3);};
            })();
        }
    }
```

Sound menu item handlers

```
    /* Handlers for Items of Sound Menu */

    var menuSoundHandlers = {
        'web␣audio␣on': webAudioOn,
        'web␣audio␣off': webAudioOff,
        'toggle canvas': toggleCanvas,
        'target audio': setTargetAudioElement,
        'none filter': gainFilter,
        'compressor': dynamicCompressor,
        'biquad filter': biquadFilter,
        'convolution': convolvingFilter,
        'convolve target': convolvingTargetFilter,
        'edited filter': editedFilter,
    };
```

Image menu item handlers

```
    /* Handlers for Items of Image Menu */

    var menuImageHandlers = {
    };
```

Movie menu item handlers

```
    /* Handlers for Items of Movie Menu */

    var menuMovieHandlers = {
        'web audio on': webAudioOn,
        'web audio off': webAudioOff,
        'toggle canvas': toggleCanvas,
        'target audio': setTargetAudioElement,
        'none filter': gainFilter,
        'compressor': dynamicCompressor,
        'biquad filter': biquadFilter,
        'convolution': convolvingFilter,
        'convolve target': convolvingTargetFilter,
        'edited filter': editedFilter,
    };
```

Canvas menu item handlers

```
    /* Handlers for Items of Canvas Menu */
    var menuCanvasHandlers = {
    };
```

Menu for synthetic media creators

```
    /* Ecell Media Creators */

    var menuEcellMediaCreators = {
        'oscillator': oscillator,
        'seven␣tones': piano,
    };
```

Module utility functions

```
    /* Sound oriented utility functions */

    /* Image oriented utility functions */


    var utilityFunctions = {
        'init-mprocessor': initMProcessor,
        'activate-oscillator': activateOscillator,
    }
```

Export of module functions (in packets)

```
    return {
      'menu-sound': menuSoundHandlers,

      'menu-image': menuImageHandlers,

        'menu-movie': menuMovieHandlers,

        'menu-canvas': menuCanvasHandlers,

        'menu-ecell-shift': menuEcellMediaCreators,

        utils: utilityFunctions,

    };
}
```

Sound oriented utility functions

Image oriented utility functions

3.2 `Web Audio` basic concepts

3.2.1 Audio programming for web browsers – short history

WAP: Could you briefly tell me how within more than 20 years of Internet sound handling on web pages evolved?

SEP: Yes, here you are, briefly in points:

1. `<bgsound>` tag:
   - background music when tag's page is visited
   - IE only
2. `<embed>` tag: `<object>` tag:
   - defines an embedded object, like
   - media: audio, video
   - application: Java applet, ActiveX, Flash
   - document: PDF, another web page
3. `<audio>` tag in HTML5
   (a) with functionality:
      - properties: controls, duration, loop, muted,. . .
      - methods: load, play, pause,. . .

   (b) with several deficiencies:

- no precise time control
- low limit for sound sources
- no buffering mechanism
- no tools for real time effects
- no tools for sound analysis

4. `Web Audio` – the built-in JS library!

WAP: Does it mean that that `Web Audio` is an extension of `audio` tag functionality.

SEP: No, it is independent library. It can be initialized by any event handler assigned to any HTML element. However, as you know, `audio` element can be the audio source for `Web Audio Graph`. Then both pieces of software cooperate. For instance, the pause on `audio` controller is the pause for `Web Audio Source Node`.

3.2.2 `Web Audio` API – an overview

WAP: Time for the standard overview. I made a brief review of the relevant web pages:

1. W3C official Web Audio API document:
   `http://webaudio.github.io/web-audio-api/`
2. Mozilla Developer Network:
   `https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API`

WAP: What I have understood from this preliminary study, for me as the programmer, it is important to know how to:

1. create programming contexts by `Web Audio` library
2. build `Web Audio` graph of audio nodes, particularly
   (a) define source audio nodes including synthetic sources
   (b) define destination audio nodes

   (c) define nodes for audio processing such as:
      i. sound wave shaping
      ii. spectral band filtering
      iii. convolution based modification
      iv. spatial sound effects
      v. audio gain change
   (d) connect and disconnect audio nodes

3. play and pause such audio graphs

SEP: That is brilliant summary. We see from this that mastering of this application requires from programmers understanding of few classes:

1. `AudioContext` - its instance is a framework for primary methods of `Web Audio`,
2. `AudioNode` - its instances are created by various generators being the methods of `AudioContext` and among them are various audio sources, destinations, and processors,
3. `AudioParam` - its instances are values of audio node properties, like frequency for spectral band filtering and they can be also audio nodes supplied by other audio graph nodes.[2]

WAP: I think, the best way to learn them is by solving the tasks you have specified in the beginning of our today talk. However, before we proceed to API details, I would like to catch the idea of audio signal flow in `Web Audio` graph.

SEP: Understanding the signal flow is not the knowledge necessary for `Web Audio Programmers` but it is good to know that the signal from any audio node, including source nodes, is emitted as *time frames* of *channel frames*. In the standard *time*

[2]For this reason for instance `BiquadFilterNode.frequency` is object, not a number which is hidden in its property of name `value`.

frames are called *buffers*, and *channel frames* are called just *frames*. The channel frame consists of one sample for mono audio, two samples for stereo, and even six samples for `5.1` audio. Each channel of the signal has the same sampling frequency f – generally $44.1kHz$. Therefore if time frame last t, the signal has c channels, the sample has b bytes, the buffer length in bytes should be around $bcft$.

WAP: I have learned that are possible two layouts for channel samples location in buffers: interleaved and planar.

SEP: Yes, in the interleaved layout, the channel samples of the same sampling rate f and time interval t are neighbors and we get ft chunks of bc bytes while in the planar layout the buffer is divided into c chunks each of bft bytes.

SEP: Coming back to the signal flow, the incoming buffers to the given node are processed there and the result is transferred to its output node. If between input buffers there is incompatibility in the sampling frequency then up-sampling to the highest rate is performed. If channel patterns are incompatible then up-mixing or down-mixing is applied - it depends on the processing kind which is working in the given node.

WAP: It means the programmers developing their own processor nodes should be aware of such incompatibilities and undertake the relevant actions?

SEP: Yes, you are right!

WAP: What about timing in `Web Audio`,

SEP: This is the crucial problem for web media applications. Any delays can be audible or visible. Therefore the scheduling of events to start or stop audio source node is controlled by precisely scheduled computer clock events. Note that each context has its local time starting from zero.

3.3 Sound menu programming for AV tags

WAP: Let us begin, according your list of tasks, from enrichment of audio and movie elements by `Web Audio` functionality.

SEP: You mean a pop-up menu will be assigned to their controllers?

WAP: Yes, menu for a complete functionality which can be conceived using the processing pipe-line: source, analyser, processor, analyser, destination.

SEP: I suggest for the simplicity to fix in this graph all nodes beside the processor which can be replaced whenever we select new processor from the menu.

WAP: It really make the code simpler, and the program more efficient, if after the creation of new node we keep this for future use.

SEP: I would also keep the recent audio parameters if, of course, we can change them.

WAP: To this goal I suggest to use code fragments having labels of filters.

SEP: What if there are many such fragments for the same menu item?

WAP: We simply assign the one which belongs to the active CFS stream.

Check for Web Audio Activation

WAP: In my implementation of menu, I get the "freezing" of menu items by calling the function assigned to each item with the second parameter `check=true.` Then the menu item function verifies pre-conditions for its execution.

SEP: Therefore before creation of the audio context assigned to the given audio or video element (with sound track), say `AV` element, all menu items are frozen beside `web audio on`.

WAP: I will simply add the audio context to the classical audio element and then it will be easy to access it.

SEP: Yes, for instance check `isWebAudioActivated(audio)`.

WAP: Where the function `isWebAudioActivated` has the form:

```
function isWebAudioActivated(audio) {
    if (audio.dc2AudioCtx==undefined ||
        audio.dc2AudioCtx==null) return false;
    return true;
}
```

Web Audio On – getting context

WAP: Let us begin our handler for`web audio on`. The first line checks existence of audio context.

SEP: In the next one you could create `AudioContext` object.

WAP: For the compact notation define `audio` variable by the `element` property of the event object `amevt` – a menu event.

SEP: To complete the local variables, let us declare `canvas` variable forfuture use as the area for drawing analysis graphs.

```
function webAudioOn(amevt,check) {
    if (check) return !isWebAudioActivated(amevt.element);

    var audio = amevt.element, audioCtx = new win.AudioContext(),
        canvas = null;
```

Web Audio On – source creation for audio tag

WAP: If context has been created we can bind it to `audio` element as the property `dc2AudioCtx`.

SEP: Remember to pause `audio` before making `audio` as the source node.

WAP: I will also refer to the source node from `audio` and reserve buffer reference, as well.

```
    if (audioCtx) {
        audio.pause();
        audio.dc2AudioCtx = audioCtx;
        audio.dc2AudioBuffer = null;
        audio.dc2AudioSource = audioCtx.createMediaElementSource(audio);
```

WAP: Now, I will create `analysers`. One for the source signal, the second for the processed one.

SEP: Then you can refer from `audio` element to an array of analyzers.

Web Audio On – creation of Web Audio analyzers

```
        audio.dc2AudioAnalysers = [];
        audio.dc2AudioAnalysers.push(audioCtx.createAnalyser());
        audio.dc2AudioAnalysers.push(audioCtx.createAnalyser());
```

Web Audio On – creation and registration the audio gain filter

WAP: I should put something neutral in place of processing node.

SEP: You could create the `gain` filter which by default has the multiplicative gain equal to one.

WAP: There is also problem of keeping created filters for further use.

SEP: Since you plan to modify filters from code fragments going back to the original one either requires memorizing all parameters of the original filter or simply keep the original `ofilter` object separate from the modified `mfilter`.

WAP: I prefer the second option, since some parameters could be hidden, and restoring only the changed ones could miss something.

SEP: Then you could register both filters as joined object in the mapping from filter names (used on menu items) to such joined objects.

WAP: Obviously such "parameter" object should be accessed directly from `audio` object, say as `audio.dc2AudioParams`.

SEP: Having many filters registered we should know which one is currently in the audio graph. It will help to disconnect it when needed.

WAP: Then we need yet another reference: `audio.dc2AudioActiveFilter`.

```
        audio.dc2AudioActiveFilter =  audioCtx.createGain();
        audio.dc2AudioParams =  {};
        audio.dc2AudioParams['gain␣filter'] = {
            ofilter: audio.dc2AudioActiveFilter,
            mfilter: audioCtx.createGain(),
        }
```

Web Audio On – graph construction

WAP: We have all nodes of our audio graph.

SEP: Then let us connect them from the source to the destination

WAP: From where I can get the destination?

SEP: The default audio destination is referred in any web audio context.

WAP: For debugging I will do counting of user actions in `audio.dc2ActionId`.

```
        audio.dc2ActionId = 0;
        audio.dc2AudioSource.connect(audio.dc2AudioAnalysers[0]);
        audio.dc2AudioAnalysers[0].connect(audio.dc2AudioActiveFilter);
        audio.dc2AudioActiveFilter.connect(audio.dc2AudioAnalysers[1]);
        audio.dc2AudioAnalysers[1].connect(audioCtx.destination);
```

Web Audio On – playing and analyzing sound from audio tag

WAP: Now, we can continue playing of audio element.

SEP: Yes, but beside playing we have analyzers which generate signals to be observed.

WAP: Then i should write a function `setAudioAnalysisCanvas` which setups audio analysis canvas including drawing in response to animation events.

SEP: The default audio destination is referred in any web audio context.

WAP: For log messages to browser's console I will make counting of user actions in `audio.dc2ActionId` and messaging by `acl` function (audio console).

```
        audio.play();
        setAudioAnalysisCanvas(audio);

        acl(audio,'none',audio.dc2AudioActiveFilter,'menu');
    }
    return null;
}
```

Replacement of audio filter

WAP: For memory savings we maintain the single audio graph. Therefore the change of a filter means replacement of the current filter by the selected one.

SEP: To this goal there is `disconnect` method in `AudioNode` class. Disconnection refers to graph edge(s) outgoing from the given node x. Disconnection without argument means that all of output links are removed. The given argument y means the removal of the single edge, between x and y.

WAP: Here argument less option is more convenient.

SEP: Yes, since we have a linear form of the graph –single input, single output then this can be used safely.

WAP: Anyway, before disconnection, we should pause `audio` element, and after connection, play again.

SEP: The new filter should be registered at `audio.dc2AudioActiveFilter`.

```
function replaceAudioFilter(audio, filter) {
    audio.pause();

    audio.dc2AudioAnalysers[0].disconnect();
    audio.dc2AudioActiveFilter.disconnect();
    audio.dc2AudioAnalysers[0].connect(filter);
    filter.connect(audio.dc2AudioAnalysers[1]);
    audio.dc2AudioActiveFilter = filter;

    audio.play();
}
```

Web Audio off

WAP: Having `web audio on` we should join to menu the item`web audio off`.

SEP: It is performed by context closing. The `close` method returns a promise, the object of class `Promise` with `then` method having a function as its argument which is called in case of success.

WAP: In theory we should come back to playing the classical `audio`.

SEP: In theory yes, but it practice it may happen that only the controller moves, no sound from speakers.

```
function webAudioOff(amevt,check) {
    if (check) {
        if (amevt.element.tagName.toUpperCase()=='VIDEO') return false;
        return isWebAudioActivated(amevt.element);
    }

    var audio = amevt.element;

    audio.pause();

    if (audio==targetAudio)  targetAudio = null;

    audio.dc2AudioCtx.close().then(function() {

        /* On Web Audio Off */

    });

    return null;
}
```

On Web Audio Off – audio tag recovering

WAP: Yes, I have checked – no sound if we only play after closing web audio context. What to do?

SEP: I suggest to create the tag `audio` again. It is not a smart solution,but we have no other choice.

WAP: I should also remove the canvas including the analysis graphs. The manipulations are easier as both elements, the audio and the canvas are descendants of the same `figure` element.

```
var fig = audio.parentElement,
    canvass = fig.getElementsByTagName('CANVAS');
fig.removeChild(canvass[0]);
audio.dc2AudioCtx = null; audio.dc2Canvas = null;
var url = audio.src, figc = audio.nextElementSibling,
    ext = url.slice(url.lastIndexOf('.')+1);
fig.removeChild(audio);
var audio2 = doc.createElement('AUDIO');
audio2.src = url; audio2.controls = 'controls';
audio2.type = 'audio/'+ext;
audio2.style.width = '100%';
audio2.style.margin = 'auto';
fig.insertBefore(audio2,figc);
```

Audio analysis – audio canvas painter

WAP: It is time to implement `setAudioAnalysisCanvas` – the function used while handling the menu item `web audio on`.

SEP: This kind of function carries two functionalities:

- definition of drawing function for results of analyser(s),
- setting the analyser(s) and start of drawing process,

WAP: How frequently the drawing function is to be called?

SEP: Since for graphics the `animation frame` mechanism is used, We have the refreshing speed determined by `animation frame` frequency (usually 60 frames per second).

WAP: In the documentation `window.requestAnimationFrame` we find the argument `timeStamp` which normally in animation can be used for motion smoothing.

SEP: In audio analysis the observed graphs are very noisy and must be smoothed by a weighted moving average. Therefore exact timing is not important here.

WAP: So drawing function

- adjusts the size of canvas to the audio elemnt width, since it is located under the controller,
- next in the upper part of canvas the analysis graph for the original signal is drawn
- and then the lower part of canvas is filled by the analysis of the processed graph.

```
function setAudioAnalysisCanvas(audio) {

    function drawAudioCanvas(timeStamp) {

        /* Draw Canvas Part */

        canvas.width = audio.offsetWidth;
        canvas.height = HEIGHT;
        drawCanvasPart(0); drawCanvasPart(1);
        if (audio.dc2Canvas && audio.dc2Canvas.style.display=='block') {
            win.requestAnimFrame(drawAudioCanvas);
        }
    }
```

Audio analysis – audio and canvas elements setup

WAP: If the canvas is not present yet then it created and located in the figure below the audio controller.

SEP: Yes, it will performed once just within `web audio on`.

```
    audio.drawAudioCanvas = drawAudioCanvas;

    var fig = audio.parentElement, canvas,
        canvass = fig.getElementsByTagName('CANVAS');

    if (canvass.length>0) {
        canvas = canvass[0];
    } else {
        cl('canvas?');
        canvas = doc.createElement('CANVAS');
        fig.appendChild(canvas);
    }
    audio.dc2Canvas = canvas;
    canvas.style.display ='block';
```

Audio analysis – analyzer setup

WAP: What actually can we get from the analyzer?

SEP: We can obtain smoothed signal data in time or spectral domains.

WAP: How the analyzer is configured?

SEP: There are at least four:

1. smoothing factor σ: $\overline{x}_t = \sigma\overline{x}_{t-1} + (1-\sigma)x_t$,
2. fft size: the number of samples used in spectral analysis (must be the power of two),
3. min decibels: min clip value in decibel domain,
4. max decibels: max clip value in decibel domain.

WAP: In this setup, the buffers are allocated? What size and elemnt type?

SEP: The buffer size is the half of fft size (since the second half is symmetric) while two element types are possible: byte and float.

WAP: We have to start drawing process hereby calling `animation frame`.

```
    var SMOOTHING = 0.8, FFT_SIZE = 2048, HEIGHT = 200;

    var analysers = audio.dc2AudioAnalysers, buffers = [{},{}],
        drawContext = canvas.getContext('2d');

    for (var i=0; i<2; i++) {
        analysers[i].minDecibels = -140;
        analysers[i].maxDecibels = 0;
        analysers[i].smoothingTimeConstant = SMOOTHING;
        analysers[i].fftSize = FFT_SIZE;

        buffers[i].freqs = new Uint8Array(analysers[i].frequencyBinCount);
        buffers[i].times = new Uint8Array(analysers[i].frequencyBinCount);
        //cl('fcount: '+analysers[i].frequencyBinCount);
    }
    if (audio.dc2Canvas.style.display=='block') {
        win.requestAnimFrame(drawAudioCanvas);
    }
}
```

Audio analysis – canvas part painter

WAP: In each part of canvas (upper and lower) I would like to draw both graphs (time and frequency domain).

SEP: As graphs could intersect, we have to discriminate them.

WAP: I think colors can be used for spectral graph where values of power spectrum will be represented as color bars.

SEP: Their colors could be drawn from a rainbow palette.

WAP: Good idea. You mean the rainbow spans along the horizontal axis of the graph. Along the horizontal axis represents frequency bins (in spectral case). Then I will use CSS `hsl` function which converts the triple (hue, saturation, lightness) to RGB value.

SEP: What are the units for arguments of `hsl`?

WAP: The units: for hue are degrees in $[0, 360]$, for saturation and lightness are percents in $[0, 100]$. Let the saturation be 100% and the lightness be 50%. Hue for the bar at the bin i (out of N bins) is located at the angle $i \cdot \frac{360}{N}$.

SEP: I understand now, how you map bins to horizontal coordinates in the graph. What about vertical coordinates.

WAP: Good question. Firstly, I take the bin content value, divide by 256 (the maximum value in byte representation), normalize it to the halved height of canvas (as we have upper and lower parts in the canvas). This could be the vertical coordinate h if the vertical axis of canvas Cartesian coordinate system goes upwards. However, the vertical axis is going downwards with zero (the origin) in the left upper corner of the canvas. Therefore, h is subtracted from the canvas height in the case of frequency ($j = 1$) – the graph is drawn in the lower part. In case of time domain ($j = 0$), h is subtracted from the halved canvas height – the graph is shifted to the upper part of the canvas.

```
    function drawCanvasPart(j) {
        var dw = canvas.width/analysers[j].frequencyBinCount,
            dhue = 360/analysers[j].frequencyBinCount;

        analysers[j].getByteFrequencyData(buffers[j].freqs);
        analysers[j].getByteTimeDomainData(buffers[j].times);
```

```

            for (var i=0; i<analysers[j].frequencyBinCount; i++) {
                var h = canvas.height*buffers[j].freqs[i]/512,
                    y = ((j==0)? canvas.height/2 : canvas.height)-h-1;
                drawContext.fillStyle = 'hsl('+i*dhue+', 100%, 50%)';
                drawContext.fillRect(i*dw,y,dw,h);
            }

            for (var i=0; i<analysers[j].frequencyBinCount; i++) {
                var h = canvas.height*buffers[j].times[i]/512,
                    y = ((j==0)? canvas.height/2: canvas.height)-h-1;
                drawContext.fillStyle = 'black';
                drawContext.fillRect(i*dw,y,1,2);
            }
        }
```

Audio analysis – canvas toggler

WAP: The goal of menu item `toggle canvas` is not only to pause the `Web Audio` play but also to hide canvases. Toggling refers to switching character between states: (audible,visible) and (inaudible, invisible).

SEP: Then, the switching for audio context is performed by `suspend, resume` methods. Since we have cooperation with audio controller, the switching for audio makes the pair `(pause,play)`, both are methods of `Audio` class standing behind `audio` tag.

WAP: In case of`canvas` element, the switching is achieved using its attribute `style.display` with the pair values `('none','block')`.

SEP: It means that visibility condition automatically switches drawing of analysis graphs.

WAP: Yes, this condition is checked at each animation frame.

SEP: Since drawing process after its break must be resumed when the audio is resumed then the request for animation frame must be issued.

```
function toggleCanvas(amevt,check) {
    if (check) return isWebAudioActivated(amevt.element);

    var audio = amevt.element;
    if (audio.dc2Canvas.style.display=='block') {
        audio.dc2Canvas.style.display = 'none';
        audio.dc2AudioCtx.suspend();
        audio.pause();
    } else {
        audio.dc2Canvas.style.display ='block';
        audio.dc2AudioCtx.resume();
        audio.play();
        win.requestAnimFrame(audio.drawAudioCanvas);
    }
    return null;
}
```

Create filter by its name

WAP: In context of bind audio processors to $\mathbb{DC}^2$ manu, it seems that there are mostly common actions. The difference is only in the `create` method used .

SEP: Therefore, let us collect creation activities into one function `createFilter`, and then we can write the common `generalFilter` functionality.

WAP: Yes, `createFilter` returns the audio processor object while `generalFilter` will be called from menu item handlers.

SEP: This possible as a filter name is known from the pressed menu item.

```
function createFilter(ctx,filterName) {
    switch(filterName) {
        case 'biquad␣filter':
            return ctx.createBiquadFilter();
        case 'gain␣filter':
            return ctx.createGain();
        case 'compressor':
            return ctx.createDynamicsCompressor();
        case 'convolution':
            return createConvolvingFilter(ctx,'default');
        case 'convolve␣target':
            return createConvolvingFilter(ctx,'target');
        case 'edited␣filter':
            return createEditedFilter(ctx);
    }
}
```

General filter – activation

WAP: General filter consists of two stages: new filter activation, filter modification by external code, i.e. CFS mechanism.

SEP: In the first stage actually we create two filter objects?

WAP: Yes, `ofilter` – having default parameters of `Web Audio` standard, and `mfilter` – with modified parameters within CFS edited on $\mathbb{DC}^2$ page.

SEP: After their registration, we replace the current filter by `ofilter`?

WAP: Yes, and make entry in `console.log`

```
function generalFilter(amevt,check,filterName) {
    if (check) return isWebAudioActivated(amevt.element);

    var audio = amevt.element, p = audio.dc2AudioParams[filterName];
    audio.dc2ActionId++;
    if (!p) {
        var filter = createFilter(audio.dc2AudioCtx,filterName);
        audio.dc2AudioParams[filterName] = {
            ofilter: filter,
            mfilter: createFilter(audio.dc2AudioCtx,filterName),
        };
        replaceAudioFilter(audio, filter);
        acl(audio,filterName,filter,'menu');
        return null;
    }
```

General filter – modifications by CFS

WAP: If there is the active CFS with the first line `// filter name` then it is interpreted and as the result a function `func` is returned.

SEP: I understand that calling `func` with the `mfilter` as the argument enables the modifications which are in CFS and can be modified in any moment by $\mathbb{DC}^2$ user.

WAP: Yes, however, after any CFS modification the filter must be again assigned from the main.

SEP: The assigned filter replaces the current one even if the current one is the same as the new one.

WAP: Yes, for code simplicity yes.

SEP: What about if the active CFS does not fit to the selected filter?

WAP: Then its original form `ofilter` replaces the current one.

```
    var func = fromStreamCodeToMenuFunction(filterName);
    if (func) {
        func(p.mfilter);
        replaceAudioFilter(audio,p.mfilter);
        acl(audio,filterName,p.mfilter,'cfs');
    } else {
        if (filterName=='convolve␣target') {
            p.ofilter.buffer = targetAudio.dc2AudioBuffer;
            p.mfilter.buffer = targetAudio.dc2AudioBuffer;
        }
        replaceAudioFilter(audio,p.ofilter);
        acl(audio,filterName,p.ofilter,'menu');
    }
    return null;
}
```

Biquad filter by general filter

WAP: The specific filters are obtained from `generalFilter` by calling with the third argument giving the name of the filter.

SEP: While the first two repeat standard arguments for menu item handlers. Could you give me example of CFS for `biquad` if I a user likes to get `highpass` filter instead of default `lowpass`, and additionally change the band transition frequency equal to `1500Hz`?

WAP: Here you are. Firstly, he/she must open a new CFS. Next, add the following code fragment referring to the parameters `type` and `frequency.value`. The users working with many code fragment streams should remember that before the use of the modified filter its CFS should be assigned as target one.

```
//biquad filter
function main(p) {
    p.type = 'highpass';
    p.frequency.value = 1500;
}
```

```
function biquadFilter(amevt,check) {
    return generalFilter(amevt,check,'biquad␣filter');
}
```

Gain filter by general filter

```
function gainFilter(amevt,check) {
    return generalFilter(amevt,check,'gain␣filter');
}
```

Dynamic compressor

```
function dynamicCompressor(amevt,check) {
    return generalFilter(amevt,check,'compressor');
}
```

Edited filter

```
function editedFilter(amevt,check) {
    return generalFilter(amevt,check,'edited␣filter');
}
```

WAP: The edited filter is not built-in `Web Audio` filter. Therefore it needs a generator.

SEP: Yes, such the generator should return the `SriptNode` object.

WAP: How the signal flows through such node?

SEP: In the same way as in any other `AudioNode` object– from input buffer to output buffer. Those buffers are created while audio node generator `context.createScriptProcessor` is executed.

WAP: How the user prepares the audio processing function interfaces with audio context controller.

SEP: When the input buffer of the script node is ready then `AudioProcessingEvent` is emitted. Therefore, the programmer for interfacing prepares its handler which makes processing and puts the result into the output buffer. The event carries references to those buffers.

WAP: I suggest for now to make no processing, so just copy input channel samples to the corresponding output channel samples.

SEP: Yes, this yet another `none filter` will be replaced by the edited one. As a matter of fact audio programmers on our $\mathbb{DC}^2$ platform will write within the `main(scriptNode)` his/her own handler being the value of `scriptNode.onaudioprocess` property.

```
function createEditedFilter(ctx) {
    var scriptNode = ctx.createScriptProcessor(4096,1,1);
    scriptNode.onaudioprocess = function(apevt) {
        var inbuf = apevt.inputBuffer, outbuf = apevt.outputBuffer;
        for (var ch=0; ch<outbuf.numberOfChannels; ch++) {
            var indata = inbuf.getChannelData(ch),
                outdata = outbuf.getChannelData(ch);
            outdata.set(indata);
        }
    }

    return scriptNode;
}
```

Convolution

```
function convolvingFilter(amevt,check) {
    return generalFilter(amevt,check,'convolution');
}
```

SEP: I see from the above code that you like to consider convolutions as other filters we implemented so far. But processing here needs two signals: original input and convolving one which should be stored as `AudioBuffer` object.

WAP: OK, so before pressing `convolution` menu item user must create this buffer.

SEP: Actually, the audio buffer object is referenced from the `ConvolverNode`.

WAP: May be, we could extend menu by, say, `target audio` giving to users the possibility to assign automatically audio buffers in the form required by `Web Audio`. Then also add menu item for convolutions with target audio.

SEP: Moreover, the general filter must now consider this special case for `convolve target`.

```
function convolvingTargetFilter(amevt,check) {
    if (!targetAudio) return false;
    return generalFilter(amevt,check,'convolve␣target');
}
```

Targeting and buffering audio objects

SEP: The buffer should be created and assigned only at first use of this item. Next times, as its name speaks, the audio element is targeted as the `convolver` for the possible convolution with other signal from any other audio element. Moreover, for efficiency we should avoid long convolving signals. Say the duration not more than 15 seconds should be OK.

WAP: Let us look now in the documentation whether we can get `AudioBuffer` from `Audio` object which is created for our audio tag.

SEP: I could not find such function, moreover the incremental building such buffer within `ScriptProcessorNode` is not directly supported. It seems that we have to use `context.decodeAudioData(encodedAudioBuffer)` which returns the promise returning the buffer we need.

WAP: Fine, we will make `ajax` request `req` to the same `url` as registered in `audio.src`. In `req.onload` we register the function which makes decoding of `req.response`. In `promise.then` the buffer will be registered as `audio.dc2AudioBuffer` for future use of convolving, for instance.

```
function setTargetAudioElement(amevt,check) {
    if (check) {
        if (amevt.element.tagName.toUpperCase()=='VIDEO') return false;
        return isWebAudioActivated(amevt.element);
    }
    var audio = amevt.element,
        ctx = audio.dc2AudioCtx,
        abuf = audio.dc2AudioBuffer;
    if (ctx.duration<15 && !abuf) {
        var req = new XMLHttpRequest();
        req.open('GET',audio.src,true);
        req.responseType = 'arraybuffer';
        req.onload = function() {
            ctx.decodeAudioData(req.response).then(function(buffer) {
                audio.dc2AudioBuffer = buffer;
            });
        };
        req.send();
    }
    targetAudio = audio;

    return null;
}
```

Creating convolving filter

WAP: For the creation of convolving filter we could fix a convolution signal which will be used for the original filter without need for selection action of users.

SEP: Yes, we should choose a short signal which clearly affects the input signal. For instance `electro.mp3` works as lowpass convolver still preserving the character of the input audio.

WAP: OK, I will create the audio buffer as for audio elements above, but I will assign to the global variable `convolverBuffer`. Then it is created once and it can be used in any audio context.

```
function createConvolvingFilter(ctx,mode) {

    var convolver = ctx.createConvolver();
    convolver.normalize = true;

    if (mode=='target') {
        convolver.buffer = targetAudio.dc2AudioBuffer;
    } else {
        if (convolverBuffer) {
            convolver.buffer = convolverBuffer;
        } else {
            var req = new XMLHttpRequest(),
                convolverURL = 'static/convolver/electro.mp3';

            req.open('GET',convolverURL,true);
            req.responseType = 'arraybuffer';
            req.onload = function() {
                ctx.decodeAudioData(req.response).then(function(buffer) {
                    convolver.buffer = buffer;
                    convolverBuffer = buffer;
                });
            };
            req.send();
        }
    }
    return convolver;
}
```

Web Audio menu

SEP: I would like to see `HTML` current code of our pop-up menu for audio elements.

WAP: Here you are:

```
<table id="menu-sound" class="pop-menu"
style="display:none;"><tbody>
<tr class="menu-item"
name="web audio on"><td>web audio on</td></tr>
<tr class="menu-item"
name="web audio off"><td>web audio off</td></tr>
<tr class="menu-item"
name="toggle canvas"><td>toggle canvas</td></tr>
<tr class="menu-item"
name="target audio"><td>target audio</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item"
name="none filter"><td>none filter</td></tr>
<tr class="menu-item"
name="compressor"><td>compressor</td></tr>
<tr class="menu-item"
name="biquad filter"><td>biquad filter</td></tr>
<tr class="menu-item"
name="convolution"><td>convolution</td></tr>
<tr class="menu-item"
name="convolve target"><td>convolve target</td></tr>
<tr class="menu-item"
name="edited filter"><td>edited filter</td></tr>
</tbody></table>
```

Console log message on Web Audio filter

WAP: The message leaves the trace for filter assignment to audio track.

SEP: It means, it prints action id, filter name, and filter type,if any.

```
function acl(audio,filterName,filter,from) {
    var atype = '';
    if ('type' in filter) atype = filter.type.toUpperCase();

    cl('('+audio.dc2ActionId+') '+filterName.toUpperCase()+
    ' from '+from+': '+atype);
}
```

3.4 Synthetic sound creation

3.4.1 Oscillator

WAP: So far, we have processed audio tracks. What about sound generators?

SEP: The basic synthesis is performed by the oscillator – the generator of pure tone with the given frequency using one of four wave types extended by a custom buffer of audio:

- sine
- square
- triangle
- sawtooth
- custom

WAP: Pure tone seems is not interesting for aural perception, isn't it?

SEP: Yes, therefore in practice it is modulated by the so called envelope which simply modifies the amplitude within the tone timing interval. Since the envelope can be shaped in a complex way, the final result is acceptable for human ear.

WAP: I have no experience with sound generators, but probably we can assume that similarly to `audio` tag, we have to create the audio graph with the an oscillator node as a source. However, now the tag `TD` representing an empty cell will be the target of menu event `SHIFT TOUCH`.

SEP: Yes, it is quite similar graph and we can follow the similar programming path. Initially for element controls we could use `TD` tag using its background for clicking. The location of such TOUCH could be translated into selected parameters controlling the shape of oscillator's envelope. Others can be modified using ILP code like for filters on audio source.

WAP: Since I need the empty `TD` tag for efficient implementation of clipboard operations then I prefer to represent the oscillator by another tag. What do you think to keep the oscillator within the `FIGURE` element, as for instance `DIV` or `CANVAS` tag, preceding the caption element.

SEP: Good point. It provides the conformance with other media elements in $\mathbb{DC}^2$ platform. Initially we can keep `DIV` with its `CSS` graphics. Later, the `CANVAS` with 2D or 3D contexts will be more flexible for implementing joined sound and graphics effects.

WAP: Let us define `oscillator` function, the handler for menu item of `menu-ecell-shift` which actually creates the `Web Audio` context, and the `OscillatorNode` together with `GainNode` shaping the signal envelope. However, along the audio aspect we have to create the figure with its content elements, the caption and an area for clicking.

```
function oscillator(mevt,check) {
    if (check) {
        if (win.AudioContext) return true;
        return false;
    }

    var ecell = mevt.element,
        audioCtx = null, fig = null, figc, area;

    audioCtx = new win.AudioContext();
    if (audioCtx) {

        fig = doc.createElement('FIGURE'),
        figc = doc.createElement('FIGCAPTION'),
        area = doc.createElement('DIV');

        fig.appendChild(area);
        fig.appendChild(figc);

        fig.style.width = '100%';
        fig.style.margin = 'auto';
        fig.style.resize = 'width';
        fig.title = 'oscillator';

        area.className = 'oscillator';

        area.dc2AudioCtx = audioCtx;
        defineOscillatorParams(area);

        area.addEventListener('mousedown',oscillatorMouseDown,true);
        area.addEventListener('mouseup',oscillatorMouseUp,true);


        figc.setAttribute('contenteditable','true');
        figc.setAttribute('lang',_lang);
        figc.setAttribute('spellcheck','true');
        figc.textContent = 'OSCILLATOR';

        ecell.textContent = ''; ecell.normalize();
        ecell.appendChild(fig);
        cl('after FIGURE link: '+ecell.childElementCount);
        cl('after FIGURE link: '+ecell.tagName);
    }
```

WAP: As for other elements filling cells we need `undo/redo` operations.

SEP: Then whatever is created must be removed within the promise `then` function.

WAP: Actuall, the DOM elements are not removed forever – they are detached from DOM tree and simply labeled to be removed.

```
    function undo() {
        if (audioCtx) {
            audioCtx.close().then(function() {
                area.dc2AudioCtx = null;
                area.removeEventListener('mousedown',oscillatorMouseDown,true);
                area.removeEventListener('mouseup',oscillatorMouseUp,true);
                fig.setAttribute('removed-to','trash');
                ecell.removeChild(fig);
                ecell.textContent = 'ECELL';
            });
        }
    }
```

WAP: The true job for redoing is getting back the closed audio context and its lost oscillator graph.

SEP: Yes, the figure itself can be recovered immediately by removing the attribute `removed-to` and linking back the figure to the emptied cell.

```
    function redo() {
        audioCtx = new win.AudioContext();
        if (audioCtx) {
            area.dc2AudioCtx = audioCtx;
```

```
                defineOscillatorParams(area);
                area.addEventListener('mousedown',oscillatorMouseDown,true);
                area.addEventListener('mouseup',oscillatorMouseUp,true);
                ecell.textContent = ''; ecell.normalize();
                fig.removeAttribute('removed-to');
                ecell.appendChild(fig);
            }
        }
    return [undo,redo];
}
```

WAP: Still we have three functions to finish our `oscillator` widget: `defineOscillatorParams`, `oscillatorMouseDown`, and `oscillatorMouseUp`. Don't we?

SEP: Yes, it is mainly the job for me.

```
function defineOscillatorParams(area) {
    var p = {};
```

WAP: The default parameters for oscillator and the gain nodes could be copied to `area.dc2AudioParams`. However, for the sound envelope we need more.

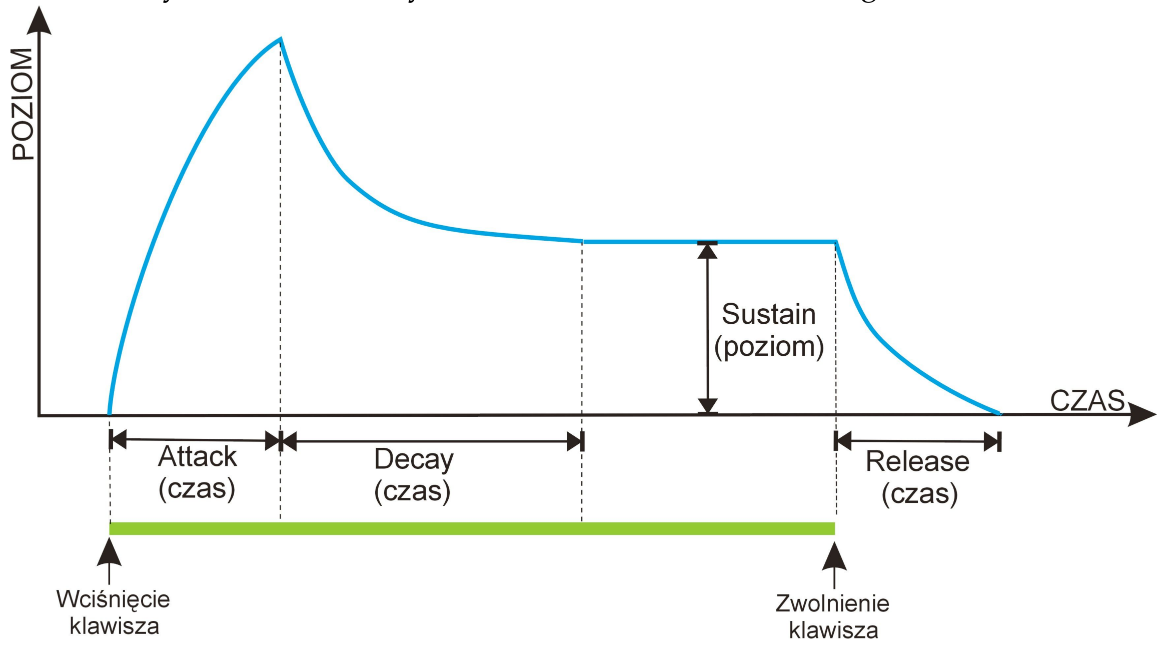
SEP: Yes, it needs careful planning. Since the oscillator is functionally coupled with the gain node, we could put all the parameters, we are going to control into the same parameter object.

WAP: OK, but we need names for them and their initial values.

SEP Yes, for audio node parameters take defaults. It is for `type`, `frequency`, `detune` of the oscillator and`gain` for the gain node. By the way, print them for testing of user scripts.

```
    p.type = 'sine';
    p.frequency = 1000;
    p.detune = 0;
    p.gain = 1.0;
```

SEP: The custom parameters belong to the signal envelope, like on the figure below. They are referred by four letters ADSR, according the flow of time:



WAP: You mean there are three time parameters (attack, decay, release) and the single sound level parameter (sustain). What about the tone timing and attack maximum gain?

SEP: The timing is meaningful if the computer plays tones without man interaction. Our `oscillator` is prompted by mouse clicks. Therefore, here the timing is given instantly by the user. Regarding the maximum gain, it equals to the value of the gain parameter we have already fixed. In the beginning, it is default, later we can modify it in the script labeled `oscillator`.

WAP: Now we need initial values for custom parameters.

SEP: I suggest to keep time parameters in millisecond, what is easier to write in scripts: `attack = 50, decay = 100, release = 100`. The level of `sustain = 0.75*p.gain` as it is relative to the maximum gain achieved.

```
    p.attack = 500;
    p.decay = 500;
    p.release = 500;
    p.sustain = 0.75*p.gain;

    area.dc2AudioParams = [p,simpleAssign({},p)];
    area.dc2AudioParamsUsed = 0;

}
```

WAP: Now, we can proceed to envelop shaping. I understand that in `Web Audio` there are tools for scheduling events which control the change of parameters in time.

SEP: Yes, such control is possible for parameters which are of type `AudioParam`. For instance type of oscillator's wave is of string type and it is not possible to interpolate its values.

WAP: What kind of interpolation in time we can use?

SEP: According of the documentation of `AudioParam` there are several methods to generate values in time:

- `setValueAtTime` – schedules an instant change to the value of the parameter at a precise time, as measured against `ctx.currentTime` given in seconds.
- `linearRampToValueAtTime` – schedules a gradual linear change in the value of the parameter. The change starts at the time specified for the previous event, follows a linear ramp to the new value to be reached at the given time.

- `exponentialRampToValueAtTime` – schedules a gradual exponential change in the value.
- `setTargetAtTime` – schedules the start of a change to the value of the parameter. The change starts at the time specified and exponentially moves towards the value given by the target parameter. The exponential decay rate is defined by the last parameter, which is a time measured in seconds.
- `setValueCurveAtTime` – schedules the values of the parameter to follow a set of values stored in a buffer of type `Float32Array` scaled to fit into the given interval, starting at start time, and having a specific duration.
- `cancelScheduledValues` – cancels all scheduled future changes to the parameter.

WAP: OK, it seems that our custom parameters exclude use of `setTargetAtTime.` Other methods can be used. From the envelope graph, the linear ramp can fit to the attack, other nonlinear parts can be implemented by exponential ramp. Moreover, I see that attack, and decay must be scheduled in `mousedown` event handler while release in `mouseup`.

```
function oscillatorMouseDown(evt) {
    if (evt.shiftKey || evt.altKey) return;
    evt.preventDefault(); evt.stopPropagation();
    var area = evt.target,
          ctx = area.dc2AudioCtx,
          osc = ctx.createOscillator(), gn = ctx.createGain(),
          r = area.getBoundingClientRect(),
            sx = (evt.clientX-r.left)/r.width,
            sy = (evt.clientY-r.top)/r.height,
            p = area.dc2AudioParams[area.dc2AudioParamsUsed];
    cl('DOWN');
    area.dc2GainNode = gn; area.dc2Oscillator = osc;

```

```
  osc.frequency.setValueAtTime(p.frequency*Math.sqrt(sx),
                                                           ctx.currentTime);
    osc.detune.setValueAtTime(p.detune, ctx.currentTime);

    var t_0 = ctx.currentTime, t_1 = t_0+Math.sqrt(sy)*p.attack/1000;
    gn.gain.setValueAtTime(0,t_0);
    gn.gain.linearRampToValueAtTime(p.gain,t_1);
    gn.gain.exponentialRampToValueAtTime(p.sustain,t_1+p.decay/1000);

    osc.connect(gn); gn.connect(ctx.destination); osc.start();
    area.dc2AudioPlaying = true;
}
function oscillatorMouseUp(evt) {
    if (evt.shiftKey || evt.altKey) return;
    evt.preventDefault(); evt.stopPropagation();

    var area = evt.target, ctx = area.dc2AudioCtx,
          gn = area.dc2GainNode, osc = area.dc2Oscillator,
            p = area.dc2AudioParams[area.dc2AudioParamsUsed];

    gn.gain.exponentialRampToValueAtTime(0.01,
                                                      ctx.currentTime+p.release/1000);
    win.setTimeout(function() {
        cl('UP');
        osc.stop(); osc.disconnect(gn);
        gn.gain.cancelScheduledValues(ctx.currentTime);
        gn.disconnect(ctx.destination);
        area.dc2AudioPlaying = false;
    },p.release);
}
```

WAP: I have a concern related to using the oscillator when we create this our custom audio control, save the wiki page, close it, and then visit it again. Since the audio context, and audio graph are JS objects having no representation in html markups they will not recreated automatically. We have to make it in some moment. Don't we?

SEP: You are absolutely right. There are two events we can do it. Just after page loading or at the moment the user likes to play the goven oscillator element for the first time after the page loading.

WAP: I prefer the second approach but as response to `SHIFT TOUCH` user action.

SEP: I agree with you, since then, beside the oscillator activation action we have a chance to add some other actions.

WAP: Yes, right, for instance configuring of the oscillator and its envelope using code fragments with label `oscillator`.

```
function activateOscillator(evt) {
    var area = evt.target;
    if (!area.dc2AudioCtx) {
     cl('A-1');
    var audioCtx = new win.AudioContext();
    if (audioCtx) {
    cl('A-2');
        area.dc2AudioCtx = audioCtx;
        defineOscillatorParams(area);
        area.addEventListener('mousedown',oscillatorMouseDown,true);
        area.addEventListener('mouseup',oscillatorMouseUp,true);
    } else {
        cl('A-3');
        return;
    }
    } else if (area.dc2AudioPlaying) {
        cl('B-1');
        var gn = area.dc2GainNode, osc = area.dc2Oscillator,
                    ctx = area.dc2AudioCtx;
        osc.stop(); osc.disconnect(gn);
        gn.gain.cancelScheduledValues(ctx.currentTime);
        gn.disconnect(ctx.destination);
        area.dc2AudioPlaying = false;
    }
    cl('B-2␣playing?:␣'+area.dc2AudioPlaying);
    var r = area.getBoundingClientRect(),
        sx = (evt.clientX-r.left)/r.width,
        sy = (evt.clientY-r.top)/r.height;

    area.dc2AudioParamsUsed = (sx<0.5)? 0 : 1;
    if (area.dc2AudioParamsUsed) {
        cl('B-3')
        var func = fromStreamCodeToMenuFunction('oscillator');
        if (func) {
          cl('B-4');
            func(simpleAssign(area.dc2AudioParams[1],
                                  area.dc2AudioParams[0]));
            cl(area.dc2AudioParams[1]);
        }
    }
}
```

3.4.2 Piano

```
function piano(mevt,check) {
    if (check) return true;

    function undo() {
    }

    function redo() {
    }

    return [undo,redo]
}
```

Appendix A

$\mathbb{DC}^2$ Design and Implementation – main module

A.1 Programmer defines main page layout for DC^2

Before taking the first layout decisions for DC^2 wiki page, I should state the design assumption: *There is only a unique template for* `dc2-main.html` *file,* i.e. all context dependent attributes and styles are assigned while executing.

```
<!--
 * DC2 - Interactive Wiki Edit Platform
 * Author: Vwadec Skarbek
 *
 * The MIT License (MIT)
 * Copyright 2015-16, Wladyslaw Skarbek, WUT
 *
 * Licensed under the MIT License
 * http://opensource.org/licenses/mit-license.php
 *
-->
```

The layout of HTML document follows from its code subdivision:

```
<!-- License HTML -->


    <!-- Style and Script Elements -->
```

```
</head>
<body>

    <!-- Main Title , User Info , Wiki Theme -->

    <!-- Block for Description Document -->

    <!-- Block for Configuration and Comment Documents -->

    <!-- Block for Application -->
```

Beside the head element and identity info (main title, user id, wiki title) we have three blocks in DC2 main HTML document. The first two are under complete control of `DC2` platform and should get the standard internal layout while the third one depends on the application configured by the user in the Configuration frame.

A.1.1 Page header

The head element beside the main title defines the access to:

1. `CSS`(Cascading Style Sheet) with drawing attributes for all elements of $\mathbb{DC}^2$ except those which are included in the internal frames and the application block.
2. `euniter.js` – `JS` (Javascript) file with tools for organization of DU (Data Units) and EE (Edition Elements)
3. `dc2.js` – main script element defining objects and methods for driving $\mathbb{DC}^2$ platform.

Title, icon, and styles

```
<style>

<!-- CSS for main html -->

<!-- Styles for Description Resources -->
```

```
</style>
```

Script for logout handling

```
<script>
var _logged = true;

window.onbeforeunload = function() {
    if (_editState==1 && _logged) return 'Are␣you␣sure?';
};

function logoutWikiPage() {
    var aurl = 'https://www.google.com';

    function goToPage() {
        cl('history␣length:␣'+window.history.length);
        if (window.history.length>1) {
            window.history.back();
        } else {
            window.location.assign(aurl);
        }
    }

    function logout() {
        var req = new XMLHttpRequest(),
            txt = 'window␣unloading';

        req.onload = function(evt) {_logged = false; goToPage();};

        req.open('POST',getContentURL(-1,txt),true);
        req.send(txt);

    }

    function saveContent(frameName) {
        var req = new XMLHttpRequest(),
            doc = dc2Globals.docs[frameName],
            txt = doc.getElementById('frame-content').innerHTML;

        req.onload = function(evt) {
            switch (frameName) {
                case 'Description': saveContent('Configuration'); break;
                case 'Configuration': saveContent('Comments'); break;
                case 'Comments': logout(); break;
            }
        }

        req.open('POST',getContentURL(frameIndx[frameName],
                'pressing␣button'),true);
        req.send(txt);
    }

    if (_editState==1) {
        var r = confirm('Save␣content?');
        if (r) {
            saveContent('Description');  //nested calls for iframes
        } else {
            logout();
        }
    } else {
        _logged = false; goToPage();
    }
}

</script>
```

Script elements for $\mathbb{DC}^2$ libraries

```
<script type="text/javascript" src="static/euniter.js"></script>
<script type="text/javascript" src="static/mprocessor.js"></script>
<script>
```

```
<!-- DC2 Application Outline -->
```

```
</script>
```

$\mathbb{DC}^2$ title in HTML

While wiki general and theme titles are informative, the user info is just for temporary representation of logging:

```
<H1 id="main-title"></H1>
```

where the font color is set in the CSS element.

```
H1,H2,H3 {color:DarkBlue;}
```

A.1.2 $\mathbb{DC}^2$ frames

Block for `Description` document

In this block the internal frame labeled by `Description` spans the full width of the page and it is rendered from `description.html` file which located in wiki theme folder `DC2 Guide`. After loading the `init('Description')` is called. The definition of the widgets is postponed.

```
<table width=100%><tbody>
<tr><td><label for="Description">Description</label><br>
<iframe id="Description" onload="dc2Globals.init('Description')"
src="/templates" width="100%" height="800px"></iframe>
</td></tr>
<tr><td id="widgets-for-Description"></td></tr>
<tr></tr>
</tbody></table>
```

Block for `Configuration` and `Comment` documents

The configuration parameters and comments are supposed to be more compact and we can align their frames horizontally. In order to keep their widgets aligned vertically we arrange them into `table` element with two columns (column of Configuration and column of Comments) and two rows (row of frames and row of widgets).

```
<br><br>
<table width=100%><tbody>
```

1. Configuration internal frame spans 60% of page width. It is labeled by this name, loaded from `DC2 Guide/configuration.html` file and `init('Configuration')` is called.

   ```
   <tr><td  width=60%>
   <label for="Configuration">Configuration</label><br>
   <iframe id="Configuration" onload="dc2Globals.init('Configuration')"
    src="/templates" width="100%" height="200px"></iframe>
   </td>
   ```

   where the size and color of label is defined by the CSS line:

   ```
    label {color:blue;font-size:150%;}
   ```

2. Comment internal frame gets remaining 40% of the horizontal span, renders by `DC2 Guide/comments.html` file and initializes its DC2 instance by calling `dc2Init('Comments')`.

   ```
   <td width=40%>
   <label for="Comments">Comments</label><br>
   <iframe id="Comments" onload="dc2Globals.init('Comments');"
    src="/templates" width="100%" height="200px"></iframe>
   </td></tr>
   ```

3. The second row includes widgets to be implemented later when the concept of the navigation between document units will be clarified:

   ```
   <tr><td id="widgets-for-Configuration"></td>
   <td id="widgets-for-Comments"></td></tr>
   </tbody></table>
   ```

Block for `Application`

Block for Display area with a red frame around:

```
<br><br>
<label for="Display">Display</label><br>
<div id="Display"
style="height:200px;width:100%;border:1px␣solid␣red" width=100%>
<canvas class="canvas" id="canvas-main" title="Main␣canvas"
width="100%" height="100%"></canvas>
</div>
```

For testing, we could take files `description, configuration, comments.html` with the identical, trivial content:

```
<html><head></head><body></body></html>
```

A.2 Programmer designs and implements functionalities for DC^2

In the previous stage of development the page layout has been defined. It includes several empty internal frames to be filled by a structure of DUs (Document Units) which in turn are populated by EEs (Edition Elements). Before we decide what HTML elements represent DU and EE, the functionalities for handling DU and EE.

A.2.1 Overview of DU functionalities

DU structure could be created according to some user actions which depend on context states:

1. `visible DU` – currently displayed document unit,
2. `target DU` – document unit which could be used in a future operation,

The following list of functions is a compact set of operations which I think satisfies the *Principle of Least Exposure*. It could be used in callbacks providing users simple tools to organize document units in tree and flat structures. Together with its inverse operations for `undo/redo` functionality, they could create a part of the class `Uniter` to be implemented in `euniter.js` file.

1. `duNew()` – the new empty DU becomes visible and located as the right sibling of the currently visible non root DU, otherwise it becomes the first child of the visible root DU (not applicable if the visible root has already children),
2. `duMakeTarget()` – the visible DU is made the target of a future operation,
3. `duAttachAfter()` – the target DU is made the right sibling of the visible DU,
4. `duAttachBefore()` – the target DU is made the left sibling of the visible DU,
5. `duRemove()` – the visible DU is deleted and the relevant links updated (not applicable for root DU),[1]
6. `duAttachFirstChild()` – the target DU is made the first child of the visible DU (applicable if currently there is no children of visible DU),[2]

A.2.2 Overview of EE functionality

Like DU structure, structure of edition elements (EE) is developed (within the visible DU) in response to some user actions which depend on context variables:

1. `visible DU` – currently displayed document unit,
2. `target EE` – edition element (maybe located in another DU) which could be used in a future operation,
3. `pointed EE` – the edition element which is currently somehow pointed in the visible document unit.

[1]Obviously, in order to make undo/redo actions, the deleted units are detached, made invisible, and only marked to be deleted.

[2]It is the only operation which is not applicable for the flat (sequential) document structures.

The underneath set of operations follows the above DU functionality and together with its inverse operations for `undo/redo` functionality, it could create the next part of the class `Uniter`.

1. `eeNew()` – the new empty EE is located after the pointed EE,
2. `eeMakeTarget()` – the pointed EE is made the target for a future operation,
3. `eeAttachAfter()` – the target EE is moved to the location after the pointed EE,
4. `eeAttachBefore()` – the target EE is moved to the location before the pointed EE,
5. `eeRemove()` – the pointed EE is deleted.[3]

A.2.3 Make outline of application `dc2.js`

Global objects of $\mathbb{DC}^2$

```

Array.prototype.extend = function(arr)
{
    this.push.apply(this, arr);
}

var bcolors = ['red','green','cyan','yellow','magenta'];
var frameIndx = {Description:0,Configuration:1,Comments:2};

var dc2Globals = DC2();
```

$\mathbb{DC}^2$ globals initialization

```
function DC2() {
    "use strict";

    var docs = {}, framecontents = {}, posts = {};
```

[3]Like for DU, the deleted elements are made invisible, and only marked to be deleted.

$\mathbb{DC}^2$ constructor – frame object definitions

```
    function init(frameName) {
        var iframe = document.getElementById(frameName);
        var win = iframe.contentWindow, doc = win.document,
                  framecontent, dc2Boards, clipboard, allMediaURLs={},
                  imageboard, soundboard, movieboard, mediaboards = {};
```

$\mathbb{DC}^2$ constructor – frame document attributes

```
        doc.id = frameName; docs[frameName] = doc; posts[frameName] = [];
        doc.indx = frameIndx[frameName]; doc.duRI = 'du-'+(doc.indx+3);
        doc.cfsState = { id: null, toFill: null, sel: null,
                         div: null, nl: null, ref: null};
        doc.cfsLineState = { ref: null, lastId: null, modified: {},
                             sel: null, div: null};
```

$\mathbb{DC}^2$ constructor – edition supporting objects

```
        var inEdition = {};
        inEdition['du-title'] = null; inEdition['lbel'] = null;
        var selector = null, naviUpper = null, naviLower = null,
            cfsRunners = {stopRequest:{}, state:{}};
```

$\mathbb{DC}^2$ constructor – language extensions and comments

```
        var langExts = ['py','js','c','cl','cpp','java',
                        'asy','gsl','html','css','txt','tex',],
            comboListsOfNames = { 'lang-exts': langExts,},
            langComments = {'js':   '/*<␣XXX␣>*/',
                            'gsl':  '/*<␣XXX␣>*/',
                            'cl':   '/*<␣XXX␣>*/',
                            'c':    '/*<␣XXX␣>*/',
                            'cpp':  '/*<␣XXX␣>*/',
                            'py':   '#<␣XXX␣>#',
                            'java':'/*<␣XXX␣>*/',
                                'asy': '/*< XXX >*/',
                                'html':'<!-- XXX -->',
                                'css': '/*< XXX >*/',
                                'txt': '%< XXX >%',
                                'tex': '%< XXX >%',
                                };
```

$\mathbb{DC}^2$ constructor – special symbols

```
        var specialSymbols = {
            link:'&#10154;',
            square:'&#9633;',
            knight: '&#9816;',
        }
```

$\mathbb{DC}^2$ constructor – menu related objects

```
        var menuEvent = {}, menusDiv = null,
            appMenuEvent = {}, appMenusDiv = null;
```

$\mathbb{DC}^2$ constructor – menu state and others

```
        var visibleDU = null,
            mState= {close: mStateClose, outside: mStateOutside,
                     closable: true, on: false,el: null};
        var liveMediaList = [];
```

$\mathbb{DC}^2$ constructor – config for euniter

```
        var config = {
                            'show-du': showDU,
                            'set-focus': setFocus,
                            'undo-visible': undoVisible,
                            'redo-visible': redoVisible,
                            'fill-media-cell': fillMediaCell,
                            'media-boards': mediaboards,
                            'media-within-board': mediaWithinBoard,
                              'in-edition': inEdition,
                              'get-ancestor-by-class-name':
                                              getAncestorByClassName,
                              'get-ancestor-by-tag-name':
                                              getAncestorByTagName,
                              'cfs-runners': cfsRunners,
                              'modal-state': mState,
                              'special-symbols': specialSymbols,
                              'lang-comments': langComments,
                              'live-media-list': liveMediaList,
                          };
```

$\mathbb{DC}^2$ constructor – setting external modules

```
        var callers = {
                              'show␣DU': showDUCaller,
                            };


        var doers = EUniter(doc,win,config);

        var mdoers = null,
            mconfig = {
                              'code-for-menu': doers.utils['code-for-menu'],
                            };
        if (doc.id=='Description') {
            mdoers = MProcessor(doc,win,mconfig);
        }
```

$\mathbb{DC}^2$ constructor – methods definitions

```
        loadFrameResources(frameName);

        /* Fill Media Cell */

        /* Loading Internal Frame Resources */
        /* Getting DU Root */

        /* Show Unit and Navigation Handling */


        /* Menu Handling */

        /* Functions for Active Area Handling */

        /* Combo Texts Handling */


        /* Out Frame Mouse Handling */

    }

    return { init: init, docs: docs, framecontents: framecontents,
             posts: posts};
}

function cl(obj) { console.log(obj); }

/* Getting URL for a resource */

/* Getting Ancestor By Class Name */

/* Simple Merging of Objects */
```

A.2.4 Define global utilities

Getting URL for a resource

```
function getContentURL(indx,by) {
    var url = 'content?me='+_me+'&wiki='+_wiki+'&kind='+indx;
    if (by) {
        url = '/post'+url+"&by="+by;
    } else {
        url = '/get'+url;
        cl(url);
    }
    return url;
}
```

Getting ancestor by class name

```
function getAncestorByClassName(el,className,included) {
    //if (!el) console.log('in getAncestor');

    var cur = el;
    className = className.toUpperCase();
    if (!included) cur = el.parentElement;
    while (cur && cur.className.toUpperCase()!=className) {
        cur = cur.parentElement;
    }
    if (!cur) cur = null;
    return cur;
}
```

Getting ancestor by tag name

```
function getAncestorByTagName(el,tagName,included) {
    //if (!el) console.log('in getAncestor');

    var cur = el;
    tagName = tagName.toUpperCase();
    if (!included) cur = el.parentElement;
    while (cur && cur.tagName.toUpperCase()!=tagName) {
        cur = cur.parentElement;
    }
    if (!cur) cur = null;
    return cur;
}
```

Simple merging of JS objects

```
function simpleAssign(objTo,objFrom) {
    if (!objFrom) return objTo;
    for (x in objFrom) objTo[x] = objFrom[x];
    return objTo;
}
```

A.2.5 Define unit navigation

Show unit

```
function showUnit(duId) {
    var el = doc.getElementById(duId);
    if (!el || !doers) return;
    showDU(el);
}
```

Show DU caller

```
function showDUCaller(args) {
    switch (args[0]) {
        case 'code␣lines':
            var lineGroup = doc.getElementById(args[1]),
                code = lineGroup.lastElementChild;
            showDU(getAncestorByClassName(lineGroup,'du',false));
            code.focus();
            doc.execCommand('selectAll',false,null);
        break;
    }
}
```

Show document unit

```
function showDU(du) {
    visibleDU.style.display = 'none';
    du.style.display = 'block';
    visibleDU = du;
    showNavi(du,'upper');
    showNavi(du,'lower');
    fillSelector(du);
}
```

Show frame content

```
function showFrameContent(mode) {
    framecontent.style.display = mode;
    naviUpper.style.display = mode;
    naviLower.style.display = mode;
}
```

Show boards

```
function showBoard(aboard) {
    //cl('style:'+framecontent.style.display);

    if (aboard.style.display=='block') {
        showFrameContent('block');
        aboard.style.display = 'none';
        return;
    }

    var i = dc2Boards.length;
    while (i--) {
        var board = dc2Boards[i];
        if (board.id==aboard.id) {
            aboard.style.display = 'block';
        } else {
            board.style.display = 'none';
        }
    }
    showFrameContent('none');
}

```

```
function showClipboard()  { showBoard(clipboard); }
function showImageboard() { showBoard(imageboard.board); }
function showSoundboard() { showBoard(soundboard.board); }
function showMovieboard() { showBoard(movieboard.board); }
```

Undo/Redo button visibility

```
function undoVisible(on) {
    var undoArea = doc.getElementById('area-undo');
    if (on) {
        undoArea.style.visibility = 'visible';
    } else {
        undoArea.style.visibility = 'hidden';
    }
}
function redoVisible(on) {
    var redoArea = doc.getElementById('area-redo');
    if (on) {
        redoArea.style.visibility = 'visible';
    } else {
        redoArea.style.visibility = 'hidden';
    }
}
```

Show navigation panels

```
function showNavi(du,what) {
    var naviBlock, whom;
    if (what=='upper') {
        naviBlock = naviUpper;
        whom = 'ancestors';
    } else { // lower
        naviBlock = naviLower;
        whom = 'children';
    }
    var naviTitles = naviBlock.getElementsByClassName('du-navi-title');
    var duIds = doers.utils['get-relatives-id'](du,whom);
    //console.log(whom); console.log(duIds);
    var a = duIds.length, i = a, n = naviTitles.length, j = 0, ii;
    while (i--) {
        if (what=='upper') ii = i;
        else ii = a-i-1;
        if (j==n) {
            naviAppendTitleElement(naviTitles,naviBlock);
            n += 1;
        }
        naviSetVisible(naviTitles[j],true);
        naviTitles[j].setAttribute('link', duIds[ii]);
        var title = doc.getElementById(duIds[ii]).title;
        naviTitles[j].textContent = title;
        //console.log('title['+j+']='+naviTitles[j].textContent);
        j += 1;
    }
    while (j<n) {
        naviSetVisible(naviTitles[j],false);
        j += 1;
    }
}
```

Append title to navigation panel

```
function naviAppendTitleElement(naviTitles,naviBlock) {
    var titleLast = naviTitles[naviTitles.length-1].parentElement
    var newTitle = titleLast.cloneNode(true);
    var tbody = naviBlock.getElementsByTagName('TBODY')[0];
    var ruler = tbody.lastElementChild;
    //console.log('tbody: '+tbody.innerHTML);
    //console.log('ruler: '+ruler.innerHTML);
    tbody.insertBefore(newTitle,ruler);
    naviTitles = naviBlock.getElementsByClassName('du-navi-title');
}
```

Visibility of navigation title

```
function naviSetVisible(naviTitle,beVisible) {
    if (beVisible) {
        naviTitle.parentElement.style.display = 'block';
    } else {
        naviTitle.parentElement.style.display = 'none';
    }
}
```

Set element focus

```
function setFocus(evt) {
    doc.body.scrollLeft = evt.ox;
    doc.body.scrollTop = evt.oy;
    evt.element.focus();
}
```

Fill selector element

```
function fillSelector(du) {
    var selTitles = selector.getElementsByTagName('OPTION');
    var duIds = doers.utils['get-relatives-id'](du,'siblings');
    var a = duIds.length, i = a, n = selTitles.length, j = 0;
    while (i--) {
        var ii = a-i-1;
        if (j==n) {
            selAppendTitleElement(selTitles);
            n += 1;
        }
        selSetVisible(selTitles[j],true);
        selTitles[j].value = duIds[ii];
        if (duIds[ii]==du.id) {
            selTitles[j].selected = true;
        } else if (selTitles[j].selected) {
            selTitles[j].selected = false;
        }
        var title = doc.getElementById(duIds[ii]).title;
        selTitles[j].textContent = title;
        //console.log('title['+j+']='+selTitles[j].textContent);
        j += 1;
    }
    while (j<n) {
        selSetVisible(selTitles[j],false);
        j += 1;
    }

}
```

Append DU title to selector

```
function selAppendTitleElement(selTitles) {
    var titleLast = selTitles[selTitles.length-1]
    var newTitle = titleLast.cloneNode(true);
    selector.appendChild(newTitle);
    selTitles = selector.getElementsByTagName('OPTION');
}
```

Visibility of selector

```
function selSetVisible(selTitle,beVisible) {
    if (beVisible) {
        selTitle.style.display = 'block';
    } else {
        selTitle.style.display = 'none';
    }
}
```

Update for selector

```
function updateSelector(du) {
    var selTitles = selector.getElementsByTagName('OPTION');
    var i = selTitles.length;
    while (i--) {
        var title = selTitles[i];
        if (title.value==du.id) {
            title.textContent = du.title;
            return;
        }
    }
}
```

Getting visible DU

```
function getVisibleDU(dus,rootDU) {
    var i = dus.length, du;
    while (i--) {
        du = dus[i];
        if (du.style.display=='block') return du;
    }
    return rootDU;
}
```

Getting root for DU tree

```
function duGetRoot(div) {
    var dus = div.getElementsByClassName('du'), rootDU;
    if (dus.length==0) {
        rootDU = doers.utils['class-instance']('du');

        var ecells = rootDU.getElementsByClassName('ecell');
        for (var i = 0; i<ecells.length; i++) {
            var ecell = ecells[i];
            ecell.textContent = 'ECELL';
        }
        var tit = rootDU.getElementsByClassName('du-title')[0];
        tit.textContent = 'Title␣on␣the␣cover␣page';
        tit.contentEditable = 'true';
        rootDU.title = tit.textContent;
        rootDU.style.display = 'block';
        visibleDU = rootDU;
        div.appendChild(rootDU);
        return rootDU;
    }
    rootDU = doc.getElementById(doc.duRI);
    if (rootDU) {
        visibleDU = getVisibleDU(dus,rootDU);
        return rootDU;
    }
    msg = 'Missing␣root␣in␣the␣frame:␣'+frameName;
    alert(msg); throw msg;
}
```

A.2.6 Implement pop-up menu

Planning mouse and menu handlers

```
var menuPath, elementPath, activeMenuId,
    isMenuVisible = false, isAppMenuVisible = false;

/* Show/Hide Menu Path */

addMouseDownHandler(); addMouseEnterHandler();
addKeyDownHandler();

/* Handling Mouse Down Event */

/* Assigning Functions to Menu Items */

/* Assigning Functions to Active Areas */
```

Handling menu state

```
function mStateClose() {
    mState.el.style.display = 'none';
    mState.on = false;
    mState.el = null;

}

function mStateOutside(evt) {
    var el = evt.target;

    while (el && el.className!='BODY') {
        if (el.id == mState.el.id) return false;
        el = el.parentElement;
    }
    return true;
}
```

Assigning functions to menu groups

```
function assignMenuFunctions(menu) {
    menu.style.display = 'none';
    var children = menu.getElementsByClassName('menu-item');
    var j = children.length;
    while (j--) {
        children[j].addEventListener('mousedown',menuItemHandler,
                                                  false);
    }
}

function assignAllMenuFunctions() {
    menusDiv = doc.getElementById('pop-menus');
    var menus = menusDiv.getElementsByClassName('pop-menu');
    var i = menus.length;
    while (i--) {
        var menu = menus[i];
        if (/hbox|vbox/.test(menu.id)) continue;
        assignMenuFunctions(menu);
    }
}
```

Assigning functions to menu items

```
function assignAppMenuFunctions(menu) {
    menu.style.display = 'none';
    var children = menu.getElementsByClassName('menu-item');
    var j = children.length;
    while (j--) {
        children[j].addEventListener('mousedown',appMenuItemHandler,
                                        false);
    }
}

function assignAllAppMenuFunctions() {
    appMenusDiv = doc.getElementById('app-menus');
    var menus = appMenusDiv.getElementsByClassName('pop-menu'),
        i = menus.length;
    while (i--) assignAppMenuFunctions(menus[i]);

}
```

Menu group handler

```
function menuItemHandler(evt) {
    if (evt.button!=0) return;
    if (evt.altKey) {
        displayMenuPath(activeMenuId,'none');
        evt.stopPropagation();
        //console.log('in menu item handler');
        return;
    }
    evt.stopPropagation();
    var target = evt.currentTarget;
    if (target.className!='menu-item') return;
    if (target.hasAttribute('frozen')) return;
    var expandTo = target.getAttribute('expand-to');
    //cl('EXPAND TO 1: '+expandTo);
    displayMenuPath(activeMenuId,'none');
    if (!expandTo)  {
        var func = getFunc(target);
        var mevt = simpleAssign({},menuEvent)
        doers.utils['push-undo-redo'](func(mevt,false));
        if (func.figResizable) {
            doers.utils['adjust-size-of-figures']
                           (func.figTargetElement);
        }
    } else {
        //cl('EXPAND TO 2: '+expandTo);
        displayMenuPath(expandTo,'block');
        activeMenuId = expandTo;
    }
}
```

Menu item handler

```
function appMenuItemHandler(evt) {
    if (evt.button!=0) return;
    if (evt.shiftKey) {
        displayAppMenu('none');
        evt.stopPropagation();
        return;
    }
    evt.stopPropagation();
    var target = evt.currentTarget;
    if (target.className!='menu-item') return;
    if (target.hasAttribute('frozen')) return;
    displayAppMenu('none');
    var func = getAppFunc(target);
    var mevt = simpleAssign({},appMenuEvent)
    doers.utils['push-undo-redo'](func(mevt,false));
    if (func.figResizable) {
        doers.utils['adjust-size-of-figures'](func.figTargetElement);
    }
}
```

A.2.7 Event handling

ALT-TOUCH handler – menu state settings

```
function addMouseDownHandler() {
    doc.body.addEventListener('mousedown',function(evt) {
        if (mState.on && mState.outside(evt)) {
            evt.preventDefault(); evt.stopPropagation();
            if (mState.closable) mState.close();
            return;
        }
```

ALT-TOUCH handler – close menu

```
        switch(evt.button) {
            case 0:
                if (evt.altKey) {
                    evt.preventDefault();
                    //evt.stopPropagation();
                    if (isMenuVisible) {
                        displayMenuPath(activeMenuId,'none');
                        return;
                    }
```

ALT-TOUCH handler – expand menu

```
menuPath = [];
var menuId, target = evt.target, nest = 0, cln,
    boxesCount = 0, cellCount = 0;
elementPath = [target];
while (true) {
    menuId = target.getAttribute('menu-id');
    if (menuId) {
        cln = target.className;
        if (cln=='vbox')  {
            menuId = 'menu-vbox-'+nest;
            nest += 1; boxesCount += 1;
        } else if (cln=='hbox') {
            menuId = 'menu-hbox-'+nest;
            nest += 1; boxesCount += 1;
        } else if (cln=='ecell' && cellCount==0) {
            var cs =
            target.getElementsByClassName('ecell');
            if (boxesCount>0 || cs.length>0) {
                menuPath = []; elementPath = [];
                return;
            }
            cellCount += 1;
        }
        menuPath.push(menuId);
        if (cln=='lbel') break;
    }
    if (target.tagName=='BODY') break;
    target = target.parentElement;
    elementPath.push(target);
}
if (boxesCount>0 && cellCount==0) {
    menuPath = []; elementPath = [];
    return;
```

```
                }
                //console.log(elementPath);
                activeMenuId = '';
                bindPopMenus(evt);
                displayMenuPath(activeMenuId,'block');
```

`SHIFT-TOUCH` **handler – media manipulation**

```
            } else if (evt.ctrlKey) {
            } else if (evt.shiftKey) {
                var target = evt.target;
                if (target.className=='ecell' &&
                    target.childElementCount==0) {
                    evt.preventDefault(); evt.stopPropagation();
                    attemptECellFilling(evt);
                }   else if (target.tagName=='IMG' ||
                    target.tagName=='AUDIO' ||
                    target.tagName=='VIDEO') {
                    evt.stopPropagation();
                    if (!attemptMediaBoardOperations(target,
                                                      'shift␣key')) {
                        evt.preventDefault();
                        attemptMediaDocumentOperations(evt);
                    }
```

`SHIFT-TOUCH` **handler – activate OSCILLATOR**

```
                } else if (target.className=='oscillator' &&
                           win.AudioContext) {
                    evt.preventDefault();
                    mdoers.utils['activate-oscillator'](evt);
```

`SHIFT-TOUCH` **handler – hide integrated CFS**

```
                } else if (target.className=='cfs-name') {
                    var tr = target.parentElement.nextElementSibling;
```

```
                        if (tr.style.display == 'none') {
                            tr.style.display = '';
                        } else {
                            tr.style.display = 'none';
                        }
```

`SHIFT-TOUCH` **handler – prompt CFS integration**

```
                    } else if (target.className=='td-editable' &&
                        target.getAttribute('code')=='integrated') {
                        target.parentElement.style.display = 'none';
                        target.textContent = 'Alt+IC␣to␣integrate␣again';
                        target.previousElementSibling.textContent= '';
```

`SHIFT-TOUCH` **handler – CFS exec**

```
                    } else {
                        var cfsRef = getAncestorByClassName(target,
                                                            'cfs-ref',false);
                        if (cfsRef) {
                            evt.preventDefault();
                            evt.stopPropagation();
                            doers.utils['code-data-exec'](evt);
                        }
                    }
```

`TOUCH` **handler – hide menu**

```
                } else if (evt.metaKey) {
                } else {
                    if (isMenuVisible) {
                        displayMenuPath(activeMenuId,'none');
                    }
                    if (isAppMenuVisible) {
                        displayAppMenu('none');
                    }
```

`TOUCH` **handler – focus at media**

```
            var target = evt.target;
            if (target.tagName=='IMG' ||
                target.tagName=='AUDIO' ||
                target.tagName=='VIDEO') {
                //evt.preventDefault();
                evt.stopPropagation();
                attemptMediaBoardOperations(target,'no␣key');
```

`TOUCH` **handler – title edit begin**

```
            } else if (target.className=='du-title') {
                //evt.preventDefault();
                evt.stopPropagation();
                beginTitleEditing(target);
```

`TOUCH` **handler – title edit end**

```
            } else if (inEdition['du-title']) {
                //evt.preventDefault();
                evt.stopPropagation();
                endTitleEditing(target,true);
```

`TOUCH` **handler – list labeling begin**

```
            } else if (target.className=='lbel') {
                //evt.preventDefault();
                evt.stopPropagation();
                beginListLabelEditing(target);
```

`ALT-TOUCH` **handler – list labeling end**

```
            } else if (inEdition['lbel']) {
                //evt.preventDefault();
                evt.stopPropagation();
                endListLabelEditing(target,true);
            }
```

```
                }
                break;
            }
    },false);
}
```

Switch active element

```
function switchActiveElement(target) {
    if (target.hasAttribute('active')) {
        target.removeAttribute('active');
    } else {
        target.setAttribute('active','true');
    }
}
```

Begin DU title editing

```
function beginTitleEditing(title) {
    title.setAttribute('contenteditable','true');
    title.focus();
    var du = getAncestorByClassName(title ,'du',false);
    inEdition['du-title'] = [du, title];
}
```

Begin list label editing

```
function beginListLabelEditing(lbel) {
    if (lbel.parentElement.previousElementSibling) {
        lbel.blur();
        return;
    }
    lbel.setAttribute('contenteditable','true');
    lbel.focus();
    inEdition['lbel'] = [lbel,lbel.textContent];
}
```

End DU title editing

```
function endTitleEditing(target,success) {
    var duTitle = inEdition['du-title'],
        duThen = duTitle[0], title = duTitle[1];
    title.setAttribute('contenteditable','false');
    title.blur();
    inEdition['du-title'] = null;
    var duNow = getAncestorByClassName(target,'du',false);
    if (success && (!duNow || duNow.id==duThen.id)) {
        title.textContent = title.textContent.replace(/\n/g,'');
        duThen.title = title.textContent;
        updateSelector(duThen);
    } else {
        title.textContent = duThen.title;
    }
}
```

End list label editing

```
function endListLabelEditing(target,success) {
    var lbelLabel = inEdition['lbel'],
        lbel = lbelLabel[0], previousLabel = lbelLabel[1];
    lbel.setAttribute('contenteditable','false');
    lbel.blur(); inEdition['lbel'] = null;
    if (success) {
        doers.utils['format-labels-from']
                    (lbel.parentElement,lbel.textContent);
    } else {
        lbel.textContent = previousLabel;
    }
}
```

Add mouse enter handler

```
function addMouseEnterHandler() {
    doc.body.addEventListener('mouseenter',function(evt) {
        var pcommands = dc2Globals.posts[frameName],
            func, args, pc;
        while (pcommands.length>0) {
            pc = pcommands.splice(0,1)[0];
            (callers[pc.funcName])(pc.callerArgs);
        }

    }, false);
}
```

A.2.8 Media handlers

Media element path

```
var mediaTag = { image: 'IMG', sound: 'AUDIO', movie: 'VIDEO'};

function mediaWithinBoard(mtarget) {
    var el = mtarget, mtype, result = [];

    do {
        el = el.parentElement;
        if (el.className=='vbox' &&
            (el.id.endsWith('-selected') || el.id.endsWith('-now'))) {
            result.push(el);
            switch (mtarget.tagName.toUpperCase()) {
                case 'IMG': mtype = 'image'; break;
                case 'AUDIO': mtype = 'sound'; break;
                case 'VIDEO': mtype = 'movie'; break;
            }
            result.push(mediaboards[mtype]);
        } else if (el.tagName=='FIGURE') {
            result.push(el);
        }
    } while (el.tagName!='BODY');
    if (result.length<2) return null;
    return result;
}
```

Focusing on media `Selected` **area**

```
function attemptMediaBoardOperations(target, akey) {
    var result = mediaWithinBoard(target);
    if (!result) return false;
    var fig = result[0], vbox = result[1], mediaboard = result[2];

    switch (akey) {
        case 'no␣key':
            if (vbox.id.endsWith('-selected')>0) {
                if (fig.hasAttribute('active')) {
                    fig.removeAttribute('active');
                    mediaboard.targetFigure = null;
                } else {
                    var prevTargetFigure = mediaboard.targetFigure;
                    if (prevTargetFigure) {
                        prevTargetFigure.removeAttribute('active');
                    }
                    mediaboard.targetFigure = fig;
                    fig.setAttribute('active','true');
                }
            }
            break;
```

Moving media between `Retrieved` **and** `Selected` **areas**

```
        case 'shift␣key':
            if (vbox.id.endsWith('-now')) {
                mediaboard.transfer(fig,'r2s');

            } else if (vbox.id.endsWith('-selected')) {
                if (fig.hasAttribute('active')) {
                    fig.removeAttribute('active');
                    mediaboard.targetFigure = null;
                }
                mediaboard.transfer(fig,'s2r');
            }
            break;
    }
    return true;
}
```

Activating media menu

```
function attemptMediaDocumentOperations(evt) {
    cl('shift␣touch:␣'+evt.target.tagName)
    var mtarget = evt.target, menu = null;
    switch (mtarget.tagName.toUpperCase()) {
        case 'IMG':
            menu = doc.getElementById('menu-image');
            break;
        case 'AUDIO':
            menu = doc.getElementById('menu-sound');
            break;
        case 'VIDEO':
            menu = doc.getElementById('menu-movie');
            break;
    }
    if (!menu) return;
    bindAppMenus(evt);
    freezeAppMenuItems(menu);
    displayAppMenu(menu);
}

function attemptECellFilling(evt) {
    var menu = doc.getElementById('menu-ecell-shift');
    if (menu) {
        bindAppMenus(evt);
        freezeAppMenuItems(menu);
        displayAppMenu(menu);
    }
}
```

A.2.9 Key down handler

Indentation and space constants in `HTML`

```
var _space = '&nbsp;', _indent4 = '&nbsp;&nbsp;&nbsp;&nbsp;';
var regSpace = new RegExp(_space,'g');
```

Trimming front k characters

```
function trimFront(lines,k) {
    var i = lines.length, mx = 1<<30, len, line;
    while (i--) {
        line = lines[i].replace(_space,'␣');
        len = line.length;
        if (len && len<mx) mx = len;
    }
    if (mx<k) return null;
    i = lines.length;
    while (i--) {
        line = lines[i].replace(regSpace,'␣');
        if (line.length) {
            lines[i] = line.slice(k).replace(/ /g,_space);
        }
    }
    return lines;
}
```

Key handler – from code to character

```
function addKeyDownHandler() {
```

```
var nameForKeyCode = {
    27:'Escape', 13:'Enter', 9:'Tab',
    65:'a', 97:'a', 66:'b', 98:'b',
    67:'c', 99:'c', 68:'d', 100:'d', 73:'i', 105:'i',
    76:'l', 108:'l', 78:'n', 110:'n', 79:'o', 111:'o',
    81:'q', 113:'q', 82:'r', 114:'r', 83:'s', 115:'s',
    84:'t', 116:'t',
    85:'u', 117:'u', 86:'v', 118:'v', 88:'x', 120:'x',
    89:'y', 121:'y', 90:'z', 122:'z',
};
```

Key handler – modifier states

```
var altState = null, ctrlState = null, metaState = null;
```

Key handler – key events filter

```
function handledKeyEvent(evt) {
    if (evt.keyCode==27 || evt.keyCode==13 || evt.keyCode==9)
        return true;
    if (evt.altKey && evt.ctrlKey) return  false;
    if (evt.shiftKey && evt.ctrlKey) return  false;
    if (evt.altKey && evt.shiftKey) return  false;
    if (evt.altKey || evt.ctrlKey || evt.metaKey) return true;
    return false;
}
```

Key handler – actions for Ctrl modifier

```
var ctrlActions = {'a':'selectAll', 'i':'italic',
                   'b':'bold', 'c':'copy',
                   'u':'underline', 'q':'strikeThrough',
                   'v':'paste', 'x':'cut',
                   'z':'undo', 'y':'redo',
                  };
var ctrlPrefixes = [];
```

Key handler – actions for Alt modifier

```
    var altActions = {'ic':'Integrate Current stream',
                      'ns':'Nest code Stream',
                      'aa':'Add code lines Above',
                      'ab':'Add code lines Below',
                      'cd':'Code switch with Data',
                      'tc':'Text centered',
                      'tl':'Text left aligned',
                      'tr':'Text right aligned',
                      'tb':'Text bigger',
                      'ts':'Text smaller',
                     };
    var altPrefixes = ['c','i','n','a','t'];
    var metaActions = {}, metaPrefixes = [];
```

Key handler – isolating events

```
    function sp(evt) { evt.preventDefault(); evt.stopPropagation();}
```

Key do action – element path creation

```
    function doAction(action,evt) {
        var target = evt.target;
        elementPath = [target];
        while (target.tagName!='BODY') {
            target = target.parentElement;
            elementPath.push(target);
        }
```

Key do action – storing element path

```
        if (evt.elementPath) {
            cl('element path already defined: '+evt.elementPath)
        }
        evt.elementPath = elementPath;
        if (evt.element) {
            cl('element␣already␣defined:␣'+evt.element)
        }
        evt.element = evt.target;
```

Key do action – function execution

```
        var func =  doers['menu-cfs'][action];
        if (func(evt,true)) {
            var undoRedo = func(evt,false);
            if (undoRedo) {
                doers.utils['push-undo-redo'](undoRedo);
            }
        }
    }
```

Key handler – get HTML selection

```
    function getHTMLSelection() {
        var html = null;
        if (win.getSelection) {
            var sel = win.getSelection();
            if (sel.rangeCount) {
                var container = doc.createElement("div");
                for (var i = 0, len = sel.rangeCount; i < len; ++i) {
                    container.appendChild(
                            sel.getRangeAt(i).cloneContents());
                }
                html = container.innerHTML;
            }
        }
        return html;
    }
```

Key handler – adding event listener

```
    doc.body.addEventListener('keydown',keyHandler,false);
```

Key handler – filtering events

```
    function keyHandler(evt) {
        if (!handledKeyEvent(evt)) {
            ctrlState = null;
            altState = null;
            metaState = null;
            return;
        }
```

Key handler – Escape handling

```
        var kn = nameForKeyCode[evt.keyCode];
        switch (kn) {
            case 'Escape':
                if (isMenuVisible) {
                    displayMenuPath(activeMenuId,'none')
                } else if (evt.target.className=='du-title' &&
                          inEdition['du-title']) {
                    endTitleEditing(evt.target,false);
                } else if (evt.target.className=='lbel' &&
                           inEdition['lbel']) {
                    endListLabelEditing(evt.target,false);
                }
                break;
```

Key handler – Enter handling

```
            case 'Enter':
                if (evt.target.className=='du-title') {
                    sp(evt);
                }
                break;
```

Key handler – Tab handling

```
            case 'Tab':
```

```
            if (evt.target.className=='td-editable' &&
                evt.target.hasAttribute('code')) {
                    sp(evt);
                    var html = getHTMLSelection();
                    if (evt.shiftKey) {
                        if (html) {
                            var lines = html.split('<br>');
                            lines = trimFront(lines,4);
                            if (!lines) return;
                            html = lines.join('<br>');
                            doc.execCommand('insertHTML',false,
                                                            html);
                        } else {
                            doc.execCommand('delete',false,null);
                            doc.execCommand('delete',false,null);
                            doc.execCommand('delete',false,null);
                            doc.execCommand('delete',false,null);
                        }
                    } else {
                        if (html) {
                            html = '<br>'+html;
                            html = html.replace(/<br>/g,
                                                            '<br>'+_indent4);
                            html = html.slice(4);
                            doc.execCommand('insertHTML',false,
                                                            html);
                        } else {
                            doc.execCommand('insertHTML',false,
                                                             _indent4);
                        }
                    }
                }
            break;
```

Key handler – default behavior for Ctrl+V

```
          case 'v':
              if (evt.ctrlKey) return;
              break;
```

Key handler – filtering event receivers

```
          default:
              var cn = evt.target.className;
              if (cn!='p-editable' && cn!='td-editable' &&
                  cn!='cfs-name')
                  return;
```

Key handler – ctrl state update

```
              evt.preventDefault(); evt.stopPropagation();
              if (evt.ctrlKey) {
                  if (evt.shiftKey) cl('shift+ctrl');

                  if (!ctrlState) {
                      ctrlState = '';
                      if (!kn) return;
                  }
                  if (ctrlState.length==0) cl('----------');
                  ctrlState += kn;
                  cl('ctrl␣state:␣'+ctrlState);
```

Key handler – executing builtin commands

```
                  var action = ctrlActions[ctrlState];
                  if (action) {
                      ctrlState = null;
                      cl('action:␣'+action);
                      doc.execCommand(action,false,null);
                  } else {
                      if (ctrlPrefixes.indexOf(ctrlState)<0) {
                            ctrlState = null;
                        }
                    }
```

Key handler – alt state update

```
                } else if (evt.altKey) {
                    if (!altState) {
                        altState = '';
                        if (!kn) return;
                    }
                    if (altState.length==0) cl('————');
                    altState += kn;
                    cl('alt␣state:␣'+altState);
```

Key handler – executing own actions

```
                    var action = altActions[altState];
                    if (action) {
                        altState = null;
                        cl('action:␣'+action);
                        doAction(action,evt);
                    } else {
                        if (altPrefixes.indexOf(altState)<0) {
                            altState = null;
                        }
                    }
                } else if (evt.metaKey) {
                }
        }
    }
}
```

A.2.10 Handling menu path

Menu path identifiers

```
function printMenus() {
    var menus = doc.getElementsByClassName('pop-menu');
    var i = menus.length;
    var s = '';
    while (i--) {
        s += 'MENU with id '+menus[i].id;
        var menuId = menus[i].getAttribute('menu-id');
        if (menuId) {
            s += ' is based on '+menuId;
        }
        s += '\n';
    }
    return s;
}
```

Display menu path

```
function displayMenuPath(menuId,mode) {
    //console.log('MENU ID: '+menuId);
    //console.log('Menu path:'); console.log(menuPath);
    var i = menuPath.length; if (i==0) return;
    var activeBoxes  = getElementsOnPath(elementPath,
                                         ['vbox','hbox','lbox']);
    var cells = getElementsOnPath(elementPath,['ecell','lbel']);
    if (cells.length!=0) {
        var cell = cells[0];
        showCellBorder(cell,mode);
        showBoxBorders(activeBoxes,mode);
    }
    //console.log('path elements:'); console.log(elementPath);
    //console.log('active boxes:'); console.log(activeBoxes);
    while (i--) {
        var menuName = menuPath[i],
            menu = doc.getElementById(menuName);
        if (!menu) {
            var nest = menuName.slice(menuName.lastIndexOf('-')+1);
            //cl('menuName: '+menuName);
            //cl('nest: '+nest);
            appendBoxMenuCopy(nest);
            menu = doc.getElementById(menuName);
            //cl('menu appended: '+menu.id);
        }

        if (menuName!=menuId) {
            menu = doc.getElementById('expand-'+menuName);
            //cl('menu id: '+menu.id)
        } else {
            freezeMenuItems(menu,menuEvent);
        }
        menu.style.display = mode;
        menu.nextElementSibling.style.display = mode;
    }
    menusDiv.style.display = mode;
    isMenuVisible = (mode!='none');
}
```

Display application menu

```
function displayAppMenu(obj) {
    if (obj=='none') {
        var menus = appMenusDiv.getElementsByClassName('pop-menu'),
            i = menus.length;
        while (i--) menus[i].style.display = 'none';
        appMenusDiv.style.display = 'none';
        isAppMenuVisible = false;
    } else {
        appMenusDiv.style.display = 'block';
        obj.style.display = 'block';
        isAppMenuVisible = true
    }
}
```

Element path filter for classes

```
function getElementsOnPath(path,classes) {
    var n = path.length, els = [], i;
    for (i=0;i<n;i++) {
        var el = path[i], cl = el.className;
        if (!cl) continue;
        var k = classes.indexOf(cl);
        if (k<0) continue;
        els.push(el);
    }
    return els;
}
```

Show box borders

```
function showBoxBorders(boxes,mode) {
    var n = boxes.length, i;
    if (n==0) return;
    if (mode=='none') {
        for (i=0;i<n;i++) {
            boxes[i].style.border = ''+(i+3)+'px␣solid␣#eee';
        }
    } else {
        for (i=0;i<n;i++) {
            boxes[i].style.border = ''+(i+3)+'px␣solid␣'+
                                        bcolors[(i+1)%(n+1)];
        }
    }
}
```

Show cell border

```
function showCellBorder(cell,mode) {
    if (mode=='none') {
        cell.style.border = '1px␣solid␣#d4d4d4';
    } else {
        cell.style.border = '1px␣solid␣'+bcolors[0];
    }
}
```

Append copy of box menu

```
function appendBoxMenuCopy(nest) {
    var boxMenuTemplate, menuTemplate, menusTemplate,
        boxMenu, menus, menu, menuItem, parent, ruler;

    boxMenuTemplate = doc.getElementById('menu-box-template');
    boxMenu = boxMenuTemplate.cloneNode(true);
    boxMenu.id = 'menu-box-'+nest;
    boxMenu.style.display = 'block';
    parent = boxMenuTemplate.parentElement;
    parent.insertBefore(boxMenu,boxMenuTemplate);
    menus = boxMenu.children;
    menusTemplate = boxMenuTemplate.children;
    var n = menus.length, i = n;
    while (i--) {
        menu = menus[i];
        if (menu.tagName=='HR') continue;
        menuTemplate = menusTemplate[i];
        menu.id = menuTemplate.id+'-'+nest;
        //cl ('APPENDED MENU ID: '+menu.id);
        menu.style.display = 'none';
        menu.setAttribute('nest',''+nest);
        if (menu.id.startsWith('expand')) {
            menuItem = menu.getElementsByClassName('menu-item')[0];
            menuItem.setAttribute('expand-to',menu.id.slice(7));
        }
        assignMenuFunctions(menu);
        setColorsForBoxSymbols(menu, nest);
    }
}
```

Set colors for box symbols

```
function setColorsForBoxSymbols(menu, nest) {
    var items = menu.getElementsByClassName('menu-item'),
        i = items.length;
    while (i--) {
        var item = items[i];
        var txt = item.innerHTML, c;
        if (menu.id.startsWith('expand')) {
            c = nest-0+1;
        } else {
            c = nest;
        }
        item.innerHTML = txt.replace('c="0"','c="'+c+'"');
        //cl('nest c item innerHTML: '+nest+'|'+c+'|'+item.innerHTML);
    }
}
```

Freeze menu items

```
function freezeMenuItems(menu, menuEvent) {
    var children = menu.getElementsByClassName('menu-item');
    var j = children.length;
    while (j--) {
        var menuItem = children[j];
        var func = getFunc(menuItem);
        if (!func) {
            console.log('menu␣id:'+
            menuItem.parentElement.parentElement.id);
```

```
                console.log('menu item name: '+menuItem.getAttribute('name'));
                console.log('menu item tag: '+menuItem.tagName);
                console.log('menu item class: '+menuItem.className);
            }


            if (func(menuEvent,true)) {
                menuItem.removeAttribute('frozen');
            }
            else {
                menuItem.setAttribute('frozen','true');
            }
        }

}
```

Freeze application menu items

```
function freezeAppMenuItems(menu) {
    var children = menu.getElementsByClassName('menu-item');
    var j = children.length;
    while (j--) {
        var menuItem = children[j];
        var func = getAppFunc(menuItem);
        if (!func) {
            console.log('menu id: '+
            menuItem.parentElement.parentElement.id);
            console.log('menu item name: '+menuItem.getAttribute('name'));
            console.log('menu item tag: '+menuItem.tagName);
            console.log('menu item class: '+menuItem.className);
        }


        if (func(appMenuEvent,true)) {
            menuItem.removeAttribute('frozen');
        }
        else {
            menuItem.setAttribute('frozen','true');
        }
    }

}
```

Bind pop-up menus

```
function bindPopMenus(evt) {
    menuEvent.element = evt.target;
    menuEvent.elementPath = elementPath;

    menuEvent.x = evt.clientX;
    menuEvent.y = evt.clientY;
    menuEvent.ox = doc.body.scrollLeft;
    menuEvent.oy = doc.body.scrollTop;

    menusDiv.style.left = evt.clientX+doc.body.scrollLeft+10;
    menusDiv.style.top = evt.clientY+doc.body.scrollTop+10;
}
```

Bind application pop-up menus

```
function bindAppMenus(evt) {
    appMenuEvent.element = evt.target;
    //appMenuEvent.elementPath = elementPath;

    appMenuEvent.x = evt.clientX;
    appMenuEvent.y = evt.clientY;
    appMenuEvent.ox = doc.body.scrollLeft;
    appMenuEvent.oy = doc.body.scrollTop;

    appMenusDiv.style.left = evt.clientX+doc.body.scrollLeft+10;
    appMenusDiv.style.top = evt.clientY+doc.body.scrollTop+10;
}
```

Retrieval menu item function

```
function getFunc(menuItem) {
    var el = getAncestorByClassName(menuItem,'pop-menu',false);
    var doer = null;
    menuEvent.nest = '0';

    if (el.hasAttribute('menu-id')) {
        doer = doers[el.getAttribute('menu-id')];
        if (el.hasAttribute('nest')) {
            menuEvent.nest = el.getAttribute('nest');
        }
    } else {
        doer = doers[el.id];
    }
    //console.log(menuItem.getAttribute('name'));
    var func = doer[menuItem.getAttribute('name')];
    //console.log(func);
    return func;
}
```

Retrieval application menu item function

```
function getAppFunc(menuItem) {
    var el = getAncestorByClassName(menuItem,'pop-menu',false);
    if (el.id=='menu-ecell-shift') {
        var itemName = menuItem.getAttribute('name'),
            func = (doers['menu-ecell-shift'])[itemName];
        if (!func) {
            func = (mdoers['menu-ecell-shift'])[itemName];
        }
        return func;
    } else {
        return (mdoers[el.id])[menuItem.getAttribute('name')];
    }
}
```

Out Frame Mouse Handling

```
function mouseOutFrameHandler(evt) {
    switch(evt.button) {
        case 0:
            if (evt.altKey) {
            } else if (evt.ctrlKey) {
            } else if (evt.shiftKey) {
            } else if (evt.metaKey) {
            } else {
                var target = evt.target;
                var duTitle = inEdition['du-title'];
                if (duTitle) {
                    endTitleEditing(target,true);
                } else if (target.className=='du-title') {
                    //evt.stopPropagation();
                    beginTitleEditing(target);
                }
                var lbelLabel = inEdition['lbel'];
                if (lbelLabel) {
                    endListLabelEditing(target,true);
                } else if (target.className=='lbel') {
                    //evt.stopPropagation();
                    beginListLabelEditing(target);
                }
                if (isMenuVisible) {
                    displayMenuPath(activeMenuId,'none');
                }
            }
            break;
    }
}
```

A.2.11 Handling functions for active areas

Area actions reserved

```
var areaActions = {};
var htmlUndo = function() {alert('FOR␣FUTURE␣USE!');};
var htmlRedo = function() {alert('FOR␣FUTURE␣USE!');};
```

Assignment of mouse listeners

```
function assignAreaFunctions() {
    var areas = doc.getElementsByClassName('active-area');
    var i = areas.length;
    while (i--) {
        areas[i].addEventListener('mousedown',areaHandler,false);
    }
}
```

Assignment of functions to areas (buttons)

```
function getAreaFunc(areaId) {
    switch (areaId) {
        case 'area-save': return saveFrameContentByButton;
        case 'area-logout': return logoutWikiPage;
        case 'area-undo': return doers.utils.undo;
        case 'area-redo': return doers.utils.redo;
        case 'area-clip': return showClipboard;
        case 'area-image': return showImageboard;
        case 'area-sound': return showSoundboard;
        case 'area-movie': return showMovieboard;
        case 'area-more-image': return imageboard.more;
        case 'area-less-image': return imageboard.less;
```

```
            case 'area-more-sound': return soundboard.more;
            case 'area-less-sound': return soundboard.less;
            case 'area-more-movie': return movieboard.more;
            case 'area-less-movie': return movieboard.less;

            case 'area-SW': return htmlUndo;
            case 'area-SE': return htmlRedo;
            default: return areaActions[areaId];
        }
}
```

Area (button) handler

```
function areaHandler(evt) {
    if (mState.on) {
        if (mState.closable) mState.close();
        return;
    }
    if (evt.button!=0) return;
    evt.stopPropagation();
    var target = evt.currentTarget;
    //console.log("area target class: "+target.className);
    if (target.className!='active-area' &&
        target.className!='active-lm-area') return;
    (getAreaFunc(target.id))();
}
```

Assignment of combo text functions

```
function assignAllComboTextFunctions() {

    var combos = doc.getElementsByClassName('combo-text'),
        i = combos.length;
    while (i--) {
        var combo = combos[i],
            namesId = combo.getAttribute('combo-names'),
            names = comboListsOfNames[namesId];
        combo.addEventListener('mousedown',function(evt) {
            showComboNames(names,evt);
        },false);
    }
}

function showComboNames(names,evt) {
}
```

A.2.12 Functions for loading of DC2 resources

Get media type

```
function getMediaType(mediaSource) {
    var ext = mediaSource.slice(mediaSource.lastIndexOf('.')+1),
        source = mediaSource.slice(0,mediaSource.indexOf('/'));
    if (source=='sound') source = 'audio';
    return source+'/'+ext;
}
```

Fill media cell

```
function fillMediaCell(ecell,mtag,mediaSource) {
    var fig = doc.createElement('FIGURE'),
        figc = doc.createElement('FIGCAPTION'),
        media = doc.createElement(mtag);

    fig.appendChild(media);
    fig.appendChild(figc);


    media.src = mediaSource;

    //var maxWidth =    (!width)? (0.9*ecell.clientWidth) : width;


    //fig.style['max-width'] = ''+maxWidth+'px';
    fig.style.width = '100%';
    fig.style.margin = 'auto';
    fig.style.resize = 'width';
    fig.title = mediaSource.slice(mediaSource.lastIndexOf('/')+1);

    if (mtag!='IMG') {
        media.controls = 'controls';
        media.type = getMediaType(mediaSource);
    }
    media.style.width = '100%';
    //media.style['max-width'] = 'inherit';
    media.style.margin = 'auto';

    figc.setAttribute('contenteditable','true');
    figc.setAttribute('lang',_lang);
    figc.setAttribute('spellcheck','true');
    figc.textContent = '...';

    ecell.textContent = ''; ecell.normalize();
    ecell.appendChild(fig);
    return fig;
}
```

Load frame resources

```
function loadFrameResources(frameName) {
    if (_editState!=1) {
        var saveArea = doc.getElementById('area-save');
        if (_editState<1) {
            saveArea.parentElement.removeChild(saveArea);
        } else {
            saveArea.style.color = 'red';
        }
    }
    assignAllMenuFunctions();
    appendBoxMenuCopy('0');
    assignAllAppMenuFunctions();

    assignAreaFunctions();
    assignAllComboTextFunctions();
    var style = document.head.getElementsByTagName('STYLE')[0];
    doc.head.appendChild(style.cloneNode(true));
    loadFrameContent(frameName);
}
/* Setting Various Elements */
/* Loading and Saving Frame Content */
```

Setup navigation elements

```
function setupNavigation() {
    naviUpper = doc.getElementById('navi-upper');
    naviLower = naviUpper.cloneNode(true);
    naviLower.id = 'navi-lower';
    doc.body.appendChild(naviLower);
    naviUpper.addEventListener('click',naviHandler,false);
    naviLower.addEventListener('click',naviHandler,false);
}
```

Navigation doer

```
function naviHandler(evt) {
    if (evt.target.className=='du-navi-title') {
        var link = evt.target.getAttribute('link');
        var du = doc.getElementById(link);
        showDU(du);
    }
}
```

Unit selector setup

```
function setupSelector(frameName) {

    var cell = document.getElementById('widgets-for-'+frameName);

    var selId = 'select-for-'+frameName;
    cell.innerHTML = '<select id="'+selId+'">'+
                     '<option value="du-0"></option></select>';

    selector = document.getElementById(selId);

    selector.addEventListener('change',function(evt) {
        showUnit(selector.value);
    },false);
}
```

Getter for stream id from its fragment

```
function getStreamIdFromFragment(el) {
    while(el.className!='du') {
        if (el.id.startsWith('cfs-frag-'))
            return el.className.slice(4); //'cfs-\d+'
        el = el.parentElement;
    }
    return null;
}
```

CFS inclusion controller

```
function buildCFSIncluded() {
    var nestLines = doc.getElementsByClassName('nest-line'),
        i = nestLines.length, included = {}, nl, asid, cfsId;
    while (i--) {
        nl = nestLines[i];
        asid = nl.getAttribute('nest-sid');
        if (!asid) continue;
        cfsId = getStreamIdFromFragment(nl);
        if (!cfsId) continue;
        if (included[asid]) {
            included[asid].push(cfsId);
        } else {
            included[asid] = [cfsId];
        }
    }
    return included;
}
```

CFS nesting validator

```
function nestingValid(nsid,sid) {
    var included = buildCFSIncluded(),
        sidInclusion = included[sid],
        asid, ansids;
    if (!sidInclusion) return true;
    if (sidInclusion.indexOf(nsid)!=-1) return false;
    while (sidInclusion.length>0) {
        asid = sidInclusion.pop();
        ansids = included[asid];
        if (!ansids) continue;
        if (ansids.indexOf(nsid)!=-1) return false;
        sidInclusion.extend(ansids);
    }
    return true;
}
```

Setup for CFS selector

```
function setupCFSSelector() {
    doc.cfsState.div =  doc.getElementById('cfs-selector');
    doc.cfsState.sel = doc.cfsState.div.firstElementChild;
    doc.cfsState.toFill =true;

    doc.cfsState.sel.addEventListener('change',function(evt) {
        var val = doc.cfsState.sel.value;
        if (doc.cfsState.nl) {
            var sid = getStreamIdFromFragment(doc.cfsState.nl)
            if (sid!=val && nestingValid(val,sid)) {
                doc.cfsState.nl.setAttribute('nest-sid',val);
                doc.cfsState.nl.title = 'NEST('+val+')';
                doc.cfsState.nl.firstElementChild.innerHTML =
                                                    specialSymbols.link;
                doc.cfsState.nl.lastElementChild.style.color = 'green';
                doc.cfsState.nl = null;
                mState.close();
            } else {
                var msg = 'INVALID␣NEST␣LINK,␣LOOPING␣OCCURS\n';
                msg += 'while␣linking␣id='+sid+'␣to␣id='+val;
                alert(msg);
                doc.cfsState.nl = null;
                mState.close();
            }
        } else if (doc.cfsState.ref) {
            var selIndx = doc.cfsState.sel.selectedIndex,
                text = doc.cfsState.sel.options[selIndx].text;
            doc.cfsState.ref.setAttribute('ref-sid',val);
            doc.cfsState.ref.title = 'REFERENCE('+val+')';
            doc.cfsState.ref.textContent = text;
            doc.cfsState.ref.style.color = 'green';
            doc.cfsState.ref = null;
            mState.close();
        } else {
            doc.cfsState.id = val;
            mState.close();
        }
    },false);
}
```

Setup for CFS line selector

```
function setupCFSLineSelector() {
    doc.cfsLineState.div =  doc.getElementById('cfs-line-selector');
    doc.cfsLineState.sel = doc.cfsLineState.div.firstElementChild;

    doc.cfsLineState.sel.addEventListener('change',function(evt) {
        var val = doc.cfsLineState.sel.value, ok;
        if (doc.cfsLineState.ref) {
            cl('value:␣'+val);
            ok = highlightLineGroup(val);
            if (ok) {
                doc.cfsLineState.ref = null;
                mState.close();
            }
        }
    },false);
}
```

Post command between frames

```
function postCommand(docId,funcName,callerArgs) {
    var post = dc2Globals.posts[docId];

    if (!Array.isArray(callerArgs)) callerArgs = [callerArgs];
    post.push({funcName:funcName,callerArgs:callerArgs});
}
```

Highlight line group

```
function highlightLineGroup(val) {
    var ids = val.split('-'), gid = ids[0], lid = ids[1], docx;
    if (doc.cfsLineState.ref.className=='cfs-ref-data') {
        docx = doc;
    } else {
        docx = dc2Globals.docs['Description'];
    }

    var lineGroup = docx.getElementById('code-lines-'+gid),
        code = lineGroup.lastElementChild;
    if (code) {
        if (doc.id==docx.id) {
            showDU(getAncestorByClassName(lineGroup,'du',false));
            code.focus();
            docx.execCommand('selectAll',false,null);
        } else {
            postCommand(docx.id,'show␣DU',['code␣lines',
                                                   'code-lines-'+gid]);
        }
        return true;
    }
    //specialSymbols.knight
    return false;
}
```

Create clipboard

```
function createClipboard() {
    var cboard = doc.createElement('DIV');
    cboard.id = 'clipboard';
    cboard.spellcheck = "true";
    cboard.title = 'Board␣for␣Clips';
    var ct = doc.createElement('H1');
    ct.id = 'clipboard-title';
    ct.className = 'board-title';
    ct.textContent = cboard.title;
    cboard.appendChild(ct);
    cboard.style.display = 'none';
    doc.body.appendChild(cboard);
    return cboard;
}
```

A.2.13 Media boards manager

Media board module

```
function createMediaboard(boardId,boardTitle,mediaFolder) {
    var board = doc.createElement('DIV');
    board.id = boardId;
    board.title = boardTitle;
    board.className = 'media-board';

    var ct = doc.createElement('H2');
    board.appendChild(ct);
    ct.id = boardId+'-title';
    ct.className = 'board-title';
    ct.textContent = boardTitle;


    var vboxTemplate = doc.getElementById('empty-vbox-template');

    var selectedDiv = doc.createElement('DIV');
    board.appendChild(selectedDiv);
    selectedDiv.id = boardId+'-selected';
    selectedDiv.title = 'Selected';

    ct = doc.createElement('H3');
    selectedDiv.appendChild(ct);
    ct.id = boardId+'-title-selected';
    ct.className = 'board-title';
    ct.textContent = 'Selected';

    var selected = vboxTemplate.cloneNode(true);
    selectedDiv.appendChild(selected);
    selected.id = mediaFolder+'-selected';
    selected.style.display = 'block';


    var retrieved = doc.createElement('DIV');
    board.appendChild(retrieved);
    retrieved.id = boardId+'-retrieved';
    retrieved.title = 'Retrieved';

    ct = doc.createElement('H3');
    retrieved.appendChild(ct);
    ct.id = boardId+'-title-retrieved';
    ct.className = 'board-title';
    ct.textContent = 'Retrieved';


    var shownBefore = vboxTemplate.cloneNode(true);
    retrieved.appendChild(shownBefore);
    shownBefore.id = mediaFolder+'-before';
    shownBefore.style.display = 'none';

    var lessAreaTemplate = doc.getElementById('area-less-template'),
        lessArea = lessAreaTemplate.cloneNode(true);
    lessArea.id = 'area-less-'+mediaFolder;
    lessArea.addEventListener('mousedown',areaHandler,false);

    retrieved.appendChild(lessArea);
    lessArea.style.display = 'block';


    var shownNow = vboxTemplate.cloneNode(true);
    retrieved.appendChild(shownNow);
    shownNow.id = mediaFolder+'-now';
    shownNow.style.display = 'block';

    var moreAreaTemplate = doc.getElementById('area-more-template'),
        moreArea = moreAreaTemplate.cloneNode(true);
    moreArea.id = 'area-more-'+mediaFolder;
    moreArea.addEventListener('mousedown',areaHandler,false);

    retrieved.appendChild(moreArea);
    moreArea.style.display = 'block';

    var shownAfter = vboxTemplate.cloneNode(true);
    retrieved.appendChild(shownAfter);
    shownAfter.id = mediaFolder+'-after';
    shownAfter.style.visibility = 'hidden';

    board.style.display = 'none';
    doc.body.appendChild(board);

    return {board: board, folder: mediaFolder, selected: selected,
            before: shownBefore, now: shownNow, after: shownAfter,
            more: null, less: null, transfer: null, target: null,
            moreArea: moreArea, lessArea: lessArea,
           };
}
```

Media URLs receiver

```
function receiveMediaURLs(folder) {

    var urls = allMediaURLs[folder];
    if (urls.length==1 && urls[0].length==0) return null;
    return urls;
}
```

Media retrieval – receive URLs

```
function setupMediaRetrieval(mediaboard,nRows,nCols) {
    var mediaURLs = receiveMediaURLs(mediaboard.folder);
    if (mediaURLs==null) {
        mediaboard.lessArea.style.display = 'none';
        mediaboard.moreArea.style.display = 'none';
        return null;
    }
```

Media retrieval – define variables

```
    var ecellTemplate = doc.getElementById('empty-ecell-template'),
        trHboxTemplate = doc.getElementById('empty-tr-hbox-template'),
        nFigs = Math.min(mediaURLs.length,nRows*nCols),
        mtag = mediaTag[mediaboard.folder], i, fig, ecell, url,
        trHbox, hbox, boardPart = mediaboard.now,
        cellWidth = 0.9*doc.body.clientWidth/nCols,
        mediaIndex = 2*nFigs, moreArea, lessArea;
```

Media retrieval – fill media cells

```
    ecellTemplate.width = "100%";

    for (i=0;i<(2*nFigs);i++) {
        if (i==mediaURLs.length) {
            mediaIndex = mediaURLs.length;
            var j;
            for (j=i%nCols; j<nCols; j++) {
                ecell = ecellTemplate.cloneNode(true);
                var hf = hbox.firstElementChild;
                hf.firstElementChild.appendChild(ecell);
            }
            break;
        }
        if (i==nFigs) boardPart = mediaboard.after;

        url = mediaURLs[i];
        ecell = ecellTemplate.cloneNode(true);
        ecell.removeAttribute('id'); ecell.normalize();

        fig = fillMediaCell(ecell,mtag,url);

        if (i%nCols==0)  {
            trHbox = trHboxTemplate.cloneNode(true);
            boardPart.firstElementChild.appendChild(trHbox);
            hbox = trHbox.firstElementChild.firstElementChild;
        }
        hbox.firstElementChild.firstElementChild.appendChild(ecell);

    }
```

Media retrieval – MORE, LESS buttons setup

```
    moreArea = doc.getElementById('area-more-'+mediaboard.folder),
    lessArea = doc.getElementById('area-less-'+mediaboard.folder);
    lessArea.style.display = 'none';
    if (mediaboard.after.firstElementChild.childElementCount==0) {
        moreArea.style.display = 'none';
    }
```

Media retrieval – generate selection row

```
    generateSelectionRow();

    function generateSelectionRow() {
        var trHbox = trHboxTemplate.cloneNode(true), i, ecell,
            hbox = trHbox.firstElementChild.firstElementChild,
            row = hbox.firstElementChild.firstElementChild;
        mediaboard.selected.firstElementChild.appendChild(trHbox);
        for (i=0; i<nCols; i++) {
            ecell = ecellTemplate.cloneNode(true);
            row.appendChild(ecell);
        }
        return row;
    }
```

Media retrieval – MORE button service

```
    function more() {
        var firstNow =
        mediaboard.now.firstElementChild.firstElementChild,
            firstAfter =
        mediaboard.after.firstElementChild.firstElementChild;

        mediaboard.now.firstElementChild.removeChild(firstNow);
        mediaboard.before.firstElementChild.appendChild(firstNow);
        lessArea.style.display = 'block';

        mediaboard.after.firstElementChild.removeChild(firstAfter);
        mediaboard.now.firstElementChild.appendChild(firstAfter);
        if (mediaIndex<mediaURLs.length &&
        mediaboard.after.firstElementChild.childElementCount<nRows) {
            trHbox = trHboxTemplate.cloneNode(true);
            trHbox.removeAttribute('id');
            mediaboard.after.firstElementChild.appendChild(trHbox);
            hbox = trHbox.firstElementChild.firstElementChild;
            for (i=0; i<nCols; i++) {
                url = mediaURLs[mediaIndex];
                ecell = ecellTemplate.cloneNode(true);
                ecell.removeAttribute('id'); ecell.normalize();
                fig = fillMediaCell(ecell,mtag,url);
                var hf = hbox.firstElementChild;
                hf.firstElementChild.appendChild(ecell);
                mediaIndex += 1;
                if (mediaIndex==mediaURLs.length) {
                    var j;
                    for (j=i+1; j<nCols; j++) {
                        ecell = ecellTemplate.cloneNode(true);
                        var hf = hbox.firstElementChild;
                        hf.firstElementChild.appendChild(ecell);
                    }
                    break;
                }
            }
        } else if (mediaIndex==mediaURLs.length &&
            mediaboard.after.firstElementChild.childElementCount==0) {
            moreArea.style.display = 'none';
        }

    }
```

Media retrieval – LESS button service

```
    function less() {
        var firstNow =
        mediaboard.now.firstElementChild.firstElementChild,
        lastBefore =
        mediaboard.before.firstElementChild.lastElementChild;

        mediaboard.before.firstElementChild.removeChild(lastBefore);
        mediaboard.now.firstElementChild.insertBefore(
                                                    lastBefore,firstNow);
        if (mediaboard.before.firstElementChild.childElementCount==0) {
            lessArea.style.display = 'none';
        }
        if (mediaboard.now.firstElementChild.childElementCount>nRows) {
            var firstAfter =
            mediaboard.after.firstElementChild.firstElementChild,
                lastNow =
            mediaboard.now.firstElementChild.lastElementChild;
            mediaboard.now.firstElementChild.removeChild(lastNow);
            if (firstAfter) {
                mediaboard.after.firstElementChild.insertBefore(
                                                    lastNow, firstAfter);
            } else {
                mediaboard.after.firstElementChild.appendChild(lastNow);
                moreArea.style.display = 'block';
            }
        }

    }
```

Media retrieval – media cells for transfer

```
    function transfer(fig,where,ecell) {
        function getEmptyCell(row) {
            var cells = row.children;
            for (i=0; i<cells.length; i++)  {
                var cell = cells[i];
                if (!cell.firstElementChild) return cell;
            }
            return null;
        }
        function getNonEmptyCell(row) {
            var cells = row.children;
            for (i=0; i<cells.length; i++)  {
                var cell = cells[i];
                if (cell.firstElementChild) return cell;
            }
            return null;
        }
```

Media retrieval – transfer from `Retrieved` **to** `Selected`

```
        switch (where) {
            case 'r2s':
                var cellFrom = fig.parentElement,
                    tbodyTo = mediaboard.selected.firstElementChild,
                    lastTrHboxTo = tbodyTo.lastElementChild,
                    hboxTo =
                    lastTrHboxTo.firstElementChild.firstElementChild,
                    rowTo = hboxTo.firstElementChild.firstElementChild,
                    cellTo = getEmptyCell(rowTo);
                if (!cellTo) {
                    rowTo = generateSelectionRow();
                    cellTo = rowTo.firstElementChild;
                }
                cellFrom.removeChild(fig);
                cellTo.appendChild(fig);
                mediaboard.cellCount += 1;
                cellFrom.id = mediaboard.folder+'-'+mediaboard.cellCount;
                fig.setAttribute('cell-from',cellFrom.id);
                break;
```

Media retrieval – transfer from `Selected` **to** `Retrieved`

```
            case 's2r':
                var cellFrom = fig.parentElement,
                    rowFrom = cellFrom.parentElement,
                    cellToId = fig.getAttribute('cell-from'),
                    cellTo = doc.getElementById(cellToId);
                cellFrom.removeChild(fig);
                if (!getNonEmptyCell(rowFrom)) {
                    trHbox = rowFrom.parentElement.parentElement;
                    trHbox = trHbox.parentElement.parentElement;
                    var tbody =  trHbox.parentElement;
                    if (tbody.childElementCount>1) {
                        tbody.removeChild(trHbox);
                        trHbox.setAttribute('remove-to','trash');
                    }
                }
                if (cellTo.childElementCount==0) cellTo.appendChild(fig);
                break;
```

Media retrieval – transfer from `Selected` **to** `Document`

```
        case 's2d':
            var cellFrom = fig.parentElement,
                rowFrom = cellFrom.parentElement,
                cellTo = ecell;
            cellFrom.removeChild(fig);
            if (!getNonEmptyCell(rowFrom)) {
                trHbox = rowFrom.parentElement.parentElement;
                trHbox = trHbox.parentElement.parentElement;
                var tbody =  trHbox.parentElement;
                if (tbody.childElementCount>1) {
                    tbody.removeChild(trHbox);
                    trHbox.setAttribute('remove-to','trash');
                }
            }
            cellTo.textContent = ''; cellTo.normalize();
            cellTo.appendChild(fig);
            break;
```

Media retrieval – transfer from `Document` **to** `Selected`

```
        case 'd2s':
```

```
                    var cellFrom = fig.parentElement,
                        tbodyTo = mediaboard.selected.firstElementChild,
                        lastTrHboxTo = tbodyTo.lastElementChild,
                        hboxTo =
                        lastTrHboxTo.firstElementChild.firstElementChild,
                        rowTo = hboxTo.firstElementChild.firstElementChild,
                        cellTo = getEmptyCell(rowTo);
                    if (!cellTo) {
                        rowTo = generateSelectionRow();
                        cellTo = rowTo.firstElementChild;
                    }
                    cellFrom.removeChild(fig);
                    cellFrom.textContent = 'ECELL';
                    cellTo.appendChild(fig);
                    break;
            }
    }
```

Media retrieval – board attributes setup

```
    mediaboard.less = less;
    mediaboard.more = more;
    mediaboard.transfer = transfer;
    mediaboard.cellCount = 0;
    mediaboard.targetFigure = null;
    mediaboard.targetEmptyCell = null;
}
```

Load frame content – parse URL data

```
function loadFrameContent(frameName) {
    function parseUrlText(txt) {
        var mparts = txt.split(';')
        allMediaURLs['image'] = mparts[0].split('|');
        allMediaURLs['sound'] = mparts[1].split('|');
        allMediaURLs['movie'] = mparts[2].split('|');
        //cl(allMediaURLs['sound']);
    }
```

Load frame content – when successful

```
    var req = new XMLHttpRequest();
    req.onload = function(event) {
        var div = doc.createElement('DIV');
        div.id = 'frame-content';
        framecontent = div;
        framecontents[doc.id] = framecontent;

        div.spellcheck = "true";
        div.innerHTML = req.responseText;
        doc.body.appendChild(div);
        cl('Loading␣frame:␣'+doc.id);

        setupNavigation();

        clipboard = createClipboard();
```

Load frame content – media boards setup

```
        if (frameName=='Description') {
            var req2 = new XMLHttpRequest();

            req2.onload = function(event) {
                parseUrlText(req2.responseText);

                var nRows=4, nCols = 15;
                imageboard = createMediaboard('imageboard',
                                 'Board␣for␣Images', 'image');
                mediaboards.image = imageboard;
                setupMediaRetrieval(imageboard,nRows,nCols);

                soundboard = createMediaboard('soundboard',
                                    'Board␣for␣Sounds', 'sound');
                nRows =3; nCols = 6;
                mediaboards.sound = soundboard;
                setupMediaRetrieval(soundboard,nRows,nCols);

                movieboard = createMediaboard('movieboard',
                                    'Board␣for␣Movies', 'movie');
                mediaboards.movie = movieboard;
                setupMediaRetrieval(movieboard,nRows,nCols);

                dc2Boards = [clipboard,imageboard.board,
                             soundboard.board,movieboard.board];
            };

            req2.open('GET','/mediaurls?wiki='+_wiki,true);
            req2.send(null);
```

Load frame content – removal of board related elements

```
        } else {
            dc2Boards = [clipboard];
            var area;
            area = doc.getElementById('area-image');
            area.parentElement.removeChild(area);
            area = doc.getElementById('area-sound');
            area.parentElement.removeChild(area);
            area = doc.getElementById('area-movie');
            area.parentElement.removeChild(area);
            var menu = doc.getElementById('menu-ecell'),
                items = menu.getElementsByClassName('menu-item'),
                i = items.length,
                names = ['set␣image␣cell', 'set␣video␣cell',
                                   'set␣audio␣cell', 'move␣to␣mediaboard'];
            while (i--) {
                var item = items[i], name = item.getAttribute('name');
                if (names.indexOf(name)<0) continue;
                item.parentElement.removeChild(item);
            }
        }
```

Load frame content – other setups

```
        doers.utils['init-euniter']();
        if (doc.id=='Description') mdoers.utils['init-mprocessor']();

        var rootDU = duGetRoot(div);
        doers.utils['verify-du-tree'](div,frameName);



        setupSelector(frameName);
        setupCFSSelector();
        setupCFSLineSelector();

        showDU(visibleDU);

        document.body.addEventListener('mousedown',
                                          mouseOutFrameHandler,false);
    };
    req.open('GET',getContentURL(doc.indx),true);
    req.send(null);
}
```

Remove/insert live media

```
function removeLiveMedia() {
    var i = liveMediaList.length;
    while (i--) {
        var mediaEcell = liveMediaList[i],
            video =  mediaEcell[0], ecell = mediaEcell[1];
            video.pause();
            ecell.removeChild(video);
            ecell.textContent = 'ECELL';
    }
}
function insertLiveMedia() {
    var i = liveMediaList.length;
    while (i--) {
        var mediaEcell = liveMediaList[i],
            video =  mediaEcell[0], ecell = mediaEcell[1];
            ecell.textContent = ''; ecell.normalize();
            ecell.appendChild(video);
            video.play();
    }
}
```

Remove DU thrash

```
function removeDUThrash() {
    var dus = doc.getElementsByClassName('du'),
        i = dus.length;
    while (i--) {
        var du = dus[i];
        if (du.hasAttribute('removed-to')  &&
            du.getAttribute('removed-to')=='trash') {
            du.parentElement.removeChild(du);
        }
    }
}
```

Save frame content

```
function saveFrameContent(by) {
    if (_editState!=1) return;
    removeLiveMedia();
    removeDUThrash();

    var txt = doc.getElementById('frame-content').innerHTML;
    var req = new XMLHttpRequest();
    req.onload = function(event) {
        if (req.responseText!='OK') {
            var msg = doc.createElement('DIV'),
                fchild = doc.body.firstElementChild;
            msg.innerHTML = req.responsText;
            doc.body.insertBefore(msg, fchild);
        } else {
            insertLiveMedia();
        }
    };
    req.open('POST',getContentURL(doc.indx,by),true);
    req.send(txt);
}

function saveFrameContentByButton() {
    saveFrameContent('pressing␣button');
}
```

A.3 Definitions for HTML templates and CSS styles

A.3.1 Nesting structure for templates and styles

```
<!-- License HTML -->

<body><meta charset="UTF-8"/>
<div id="app-menus" style="display:none" width="auto" height="auto">

<!-- App Menu Templates -->
```

```
</div>
<div id="pop-menus" style="display:none" width="auto" height="auto">

<!-- Menu Templates -->

</div>
<div id="active-areas">

<!-- Templates for Responsive Areas -->

</div>

<!-- DU Template -->

<!-- Clip Item Template -->

<div id="navi-upper" class="navi" width="100%">

<!-- Navigation Template -->

</div>
<div id="combo-lists" style="display:none;" width="100%">

<!-- Combo Lists -->

</div>
<div id="cfs-selector" style="display:none" width="auto" height="auto">
<select style="display:block;font-size:90%;">
<option>SELECT a STREAM</option>
</select>
</div>
<div id="cfs-line-selector" style="display:none"
width="auto" height="auto">
<select style="display:block;font-size:70%;">
<option>SELECT a CODE LINE and GO THERE</option>
</select>
</div></body>
```

```
<table width="auto" height="auto"><tbody>
<tr><td><hr class="navi-ruler"></hr></td></tr>
<tr><td  class="du-navi-title" link="du-0">Title</td></tr>
<tr><td><hr class="navi-ruler"></hr></td></tr>
</tbody></table>
```

```
/* Styles for Description Frame */
body {
    font: 150% "Times␣New␣Roman";
    background:#eee; height:100%;
    margin:0px 0px 0px 0px;
    padding:0px 50px 0px 50px;
    color:#1919C0;
}
/* Styles for Menu Elements */
/* Styles for Responsive Areas */
/* Styles for Navigation Elements */
/* Styles for Titles */
/* Styles for Edition Elements */
```

A.3.2 Templates and styles for menus

Application menus

```
<!-- Sound Menu -->
<!-- Movie Menu -->
<!-- Empty Cell Menu by SHIFT TOUCH -->
```

Web Audio Menu

```
<table id="menu-sound" class="pop-menu"
 style="display:none;"><tbody>
<tr class="menu-item"
name="web␣audio␣on"><td>web audio on</td></tr>
<tr class="menu-item"
name="web␣audio␣off"><td>web audio off</td></tr>
<tr class="menu-item"
name="toggle␣canvas"><td>toggle canvas</td></tr>
<tr class="menu-item"
name="target␣audio"><td>target audio</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item"
name="none␣filter"><td>none filter</td></tr>
<tr class="menu-item"
name="compressor"><td>compressor</td></tr>
<tr class="menu-item"
name="biquad␣filter"><td>biquad filter</td></tr>
<tr class="menu-item"
name="convolution"><td>convolution</td></tr>
<tr class="menu-item"
name="convolve␣target"><td>convolve target</td></tr>
<tr class="menu-item"
name="edited␣filter"><td>edited filter</td></tr>
</tbody></table>
```

Movie Menu

```
<table id="menu-movie" class="pop-menu"
 style="display:none;"><tbody>
<tr class="menu-item"
name="web␣audio␣on"><td>web audio on</td></tr>
<tr class="menu-item"
name="web␣audio␣off"><td>web audio off</td></tr>
<tr class="menu-item"
```

```
name="toggle␣canvas"><td>toggle canvas</td></tr>
<tr class="menu–item"
name="target␣audio"><td>target audio</td></tr>
<tr><td><hr class="item−ruler"></hr></td></tr>
<tr class="menu–item"
name="none␣filter"><td>none filter</td></tr>
<tr class="menu–item"
name="compressor"><td>compressor</td></tr>
<tr class="menu–item"
name="biquad␣filter"><td>biquad filter</td></tr>
<tr class="menu–item"
name="convolution"><td>convolution</td></tr>
<tr class="menu–item"
name="convolve␣target"><td>convolve target</td></tr>
<tr class="menu–item"
name="edited␣filter"><td>edited filter</td></tr>
</tbody></table>
```

Menu for empty cell - by SHIFT TOUCH

```
<table id="menu–ecell−shift" class="pop–menu" style="display:none;"><tbody>
<tr class="menu–item"
name="init␣paragraph"><td>init paragraph</td></tr>
<tr class="menu–item"
name="set␣list␣box"><td>set list box</td></tr>
<tr><td><hr class="item−ruler"></hr></td></tr>
<tr class="menu–item"
name="start␣new␣stream"><td>start new stream</td></tr>
<tr class="menu–item"
name="new␣code␣fragment"><td>new code fragment</td></tr>
<tr class="menu–item"
name="resume␣code␣stream"><td>resume code stream</td></tr>
<tr class="menu–item"
name="set␣code␣reference"><td>set code reference</td></tr>
```

```
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item"
name="set␣live␣media␣cell"><td>set live media cell</td></tr>
<tr class="menu-item"
name="oscillator"><td>oscillator</td></tr>
<tr class="menu-item"
name="seven␣tones"><td>seven tones</td></tr>
</tbody></table>
```

Outline of menu templates

```
<!-- Menu for List Box -->

<!-- Menu for ECell -->

<div id="menu-box-template" style="display:none;">

<!-- Menu for Horizontal Box -->

<!-- Menu for Vertical Box -->

</div>

<!-- Menu for Document Units -->

<!-- Menu for Clip Items -->
```

Menu for list box

```
<table id="expand-menu-lbox" class="pop-menu"><tbody>
<tr class="menu-item" expand-to="menu-lbox">
<td>expand menu for lbox</td></tr></tbody></table>
<hr class="menu-ruler"></hr>
<table id="menu-lbox" class="pop-menu"  style="display:none;"><tbody>
<tr class="menu-item" name="add␣item␣before">
<td>add item before</td></tr>
<tr class="menu-item" name="add␣item␣after">
<td>add item after</td></tr>
```

```
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="remove␣empty␣item">
<td>remove empty item</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
```

Menu for empty cell by ALT TOUCH

```
<table id="expand-menu-ecell" class="pop-menu"><tbody>
<tr class="menu-item" expand-to="menu-ecell">
<td><span c="0">&#9633;</span> expand menu for ecell</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
<table id="menu-ecell" class="pop-menu" style="display:none;"><tbody>
<tr class="menu-item" name="assign␣code␣stream">
<td><span c="0">&#9633;</span> assign code stream</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="set␣image␣cell">
<td><span c="0">&#9633;</span> set image cell</td></tr>
<tr class="menu-item" name="set␣video␣cell">
<td><span c="0">&#9633;</span> set video cell</td></tr>
<tr class="menu-item" name="set␣audio␣cell">
<td><span c="0">&#9633;</span> set audio cell</td></tr>
<tr class="menu-item" name="move␣to␣mediaboard">
<td><span c="0">&#9633;</span> move to mediaboard</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="move␣to␣clipboard">
<td><span c="0">&#9633;</span> move to clipboard</td></tr>
<tr class="menu-item" name="copy␣to␣clipboard">
<td><span c="0">&#9633;</span> copy to clipboard</td></tr>
<tr class="menu-item" name="paste␣empty␣cell">
<td><span c="0">&#9633;</span> paste empty cell</td></tr>
<tr class="menu-item" name="remove␣empty␣cell">
<td><span c="0">&#9633;</span> remove empty cell</td></tr>
```

```
</tbody></table>
<hr class="menu-ruler"></hr>
```

Menu for horizontal box

```
<table id="expand-menu-hbox" class="pop-menu"><tbody>
<tr class="menu-item" expand-to="menu-hbox">
<td><span c="0">&#9645;</span> expand menu for hbox</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
<table id="menu-hbox" class="pop-menu" menu-id="menu-box"
 style="display:none;"><tbody>
<tr class="menu-item" name="split␣cell␣up">
<td><span c="0">&#9647;</span> split cell up</td></tr>
<tr class="menu-item" name="split␣cell␣down">
<td><span c="0">&#9647;</span> split cell down</td></tr>
<tr class="menu-item" name="split␣cell␣left">
<td><span c="0">&#9647;</span> split cell left</td></tr>
<tr class="menu-item" name="split␣cell␣right">
<td><span c="0">&#9647;</span> split cell right</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="move␣to␣clipboard">
<td><span c="0">&#9647;</span> move to clipboard</td></tr>
<tr class="menu-item" name="copy␣to␣clipboard">
<td><span c="0">&#9647;</span> copy to clipboard</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
```

```
<table id="expand-menu-vbox" class="pop-menu"><tbody>
<tr class="menu-item" expand-to="menu-vbox">
<td><span c="0">&#9647;</span> expand menu for vbox</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
<table id="menu-vbox" class="pop-menu" menu-id="menu-box"
 style="display:none;"><tbody>
```

```
<tr class="menu–item" name="split␣cell␣up">
<td><span c="0">&#9645;</span> split cell up</td></tr>
<tr class="menu–item" name="split␣cell␣down">
<td><span c="0">&#9645;</span> split cell down</td></tr>
<tr class="menu–item" name="split␣cell␣left">
<td><span c="0">&#9645;</span> split cell left</td></tr>
<tr class="menu–item" name="split␣cell␣right">
<td><span c="0">&#9645;</span> split cell right</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu–item" name="move␣to␣clipboard">
<td><span c="0">&#9645;</span> move to clipboard</td></tr>
<tr class="menu–item" name="copy␣to␣clipboard">
<td><span c="0">&#9645;</span> copy to clipboard</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
```

Menu for clip items

```
<table id="expand–menu–ci" class="pop–menu"><tbody>
<tr class="menu–item" expand-to="menu-ci">
<td>expand menu for clip</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
<table id="menu–ci" class="pop–menu" style="display:none;"><tbody>
<tr class="menu–item" name="switch␣active␣clip">
<td>switch active clip</td></tr>
<tr class="menu–item" name="clip␣to␣trash">
<td>clip to trash</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
```

```
<table id="expand–menu–du" class="pop–menu"><tbody>
<tr class="menu–item" expand-to="menu–du">
<td>expand menu for DU</td></tr>
</tbody></table>
```

```
<hr class="menu-ruler"></hr>
<table id="menu-du" class="pop-menu" style="display:none;"><tbody>
<tr class="menu-item" name="new␣DU">
<td>insert new DU after this DU</td></tr>
<tr class="menu-item" name="empty␣first␣DU␣child">
<td>empty first child of this DU</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="remove␣DU">
<td>remove to forest</td></tr>
<tr class="menu-item" name="delete␣DU">
<td>throw away to trash</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="target␣DU">
<td>set target as this DU</td></tr>
<tr><td><hr class="item-ruler"></hr></td></tr>
<tr class="menu-item" name="attach␣after␣DU">
<td>move target DU after this DU</td></tr>
<tr class="menu-item" name="attach␣before␣DU">
<td>move target DU before this DU</td></tr>
<tr class="menu-item" name="target␣first␣DU␣child">
<td>move target to be the first child</td></tr>
</tbody></table>
<hr class="menu-ruler"></hr>
```

Styles for menu templates

```
table {
    table-layout: fixed;
}

#app-menus,#pop-menus,#cfs-selector,#cfs-line-selector {
    font: menu;  /*6px Verdana, Arial;*/
    position: absolute;
    display: none;
    padding: 2px 2px;
    border: 3px solid orange;
    background-color: white;
    z-index:900;
}
.menu-item:hover:not([frozen]) {
    font-size:110%;
    color: orange;
}
.menu-item[expand-to] {
    font-weight: 600;
    /*background: #eee;*/
    color: #1919C0;
}
.menu-item[frozen] {
    font-weight: 200;
    background: #eee;
    color: blue;
}
hr.item-ruler {
    border: 1px dashed orange;
}

hr.menu-ruler {
    border: 2px solid orange;
    display: none;
}

hr.clip-ruler {
    border:2px solid orange;
}
hr.active-clip-ruler {
    border: 4px solid orange;
}

span[c="0"] {
    color: red;
}
span[c="1"] {
    color: green;
}
span[c="2"] {
    color: cyan;
}
span[c="3"] {
    color: yellow;
}
span[c="4"] {
    color: magenta;
}
```

Templates for responsive areas

```
<table id="area-undo" class="active-area"><tbody>
<tr><td>UNDO</td></tr>
</tbody></table>
<table id="area-redo" class="active-area"><tbody>
<tr><td>REDO</td></tr>
</tbody></table>
<table id="area-clip" class="active-area"><tbody>
<tr><td>CLIP</td></tr>
</tbody></table>
<table id="area-image" class="active-area"><tbody>
```

```
<tr><td>IMAGE</td></tr>
</tbody></table>
<table id="area-sound" class="active-area"><tbody>
<tr><td>SOUND</td></tr>
</tbody></table>
<table id="area-movie" class="active-area"><tbody>
<tr><td>MOVIE</td></tr>
</tbody></table>
<table id="area-more-template" class="active-lm-area"
 style="display:none;"><tbody>
<tr><td>MORE</td></tr>
</tbody></table>
<table id="area-less-template" class="active-lm-area"
 style="display:none;"><tbody>
<tr><td>LESS</td></tr>
</tbody></table>

<table id="area-save" class="active-area"><tbody>
<tr><td>SAVE</td></tr>
</tbody></table>
<table id="area-logout" class="active-area"><tbody>
<tr><td>LOGOUT</td></tr>
</tbody></table>
```

Styles for responsive areas

```
.active-area {
    position: fixed;
    padding: 0;
    background-color: #eee;
    color: #999;
    z-index: 10;
}

```

```
.active-lm-area {
    padding: 0;
    background-color: #eee;
    color: #999;
}

#area-logout {
    right: 0px;
    top: 0px;
    visibility: visible;
}

#area-save {
    right: 0px;
    top: 250px;
    visibility: visible;
}

#area-redo {
    left: 0px;
    top: 200px;
    visibility: hidden;
}

#area-undo {
    left: 0px;
    top: 225px;
    visibility: hidden;
}

#area-clip {
    left: 0px;
    top: 250px;
    visibility: visible;
```

```
}

#area-image {
     left: 0px;
     top: 275px;
     visibility: visible;
}

#area-sound {
     left: 0px;
     top: 300px;
     visibility: visible;
}

#area-movie {
     left: 0px;
     top: 325px;
     visibility: visible;
}
.active-area:hover {
     position: fixed;
     transform: scale(1.5,1.5);
     padding:0;
     background-color: #eee;
     color: orange;
     transition: 1s;
     z-index: 10;
}

.active-lm-area:hover {
     font-size: 200%;
     padding:0;
     background-color: #eee;
```

```
    color: orange;
}

#area-SW {
    left: 8px;
    bottom: 7px;
}
#area-SE {
    right: 8px;
    bottom: 7px;
}

#area-NW {
    left: 8px;
    top: 0px;
}

#area-NE {
    right: 8px;
    top: 0px;
}
```

Styles for navigation elements

```
.navi {
    padding: 5px 5px 5px 30px;
    line-heigth: 50%;
}

.du-navi-title {
    font-size: 90%;
    font-family: Arial Sans;
    font-style: italic;
}
```

```

hr.navi-ruler {
    border: 1px solid blue;
    display: block;
}
```

Define DU/EE templates

```
<div class="du" id="du-template"  counter="0" width="100%"
 title="TITLE␣TO␣BE␣DEFINED" style="display:none;" menu-id="menu-du"
up="du-0" left="du-0" right="du-0" down="du-0">

<table width="100%"><tbody>
<tr><td  contenteditable="false" class="du-title"
width="100%">TITLE TO BE DEFINED</td></tr></tbody></table>

<table class="vbox" width="100%" menu-id="menu-box"><tbody>
<tr width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td></tr>
<tr width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td></tr>
</tbody></table>
</div>


<div style="display:none">

<table id="lbox-template" class="lbox"
width="100%" height="100%" ><tbody>
<tr width="100%">
<td class="lbel" width="5%" menu-id="menu-lbox">1. </td>
<td class="ecell" menu-id="menu-ecell" in-list="true"
width="95%">ECELL</td>
```

```
</tr>
</tbody></table>

<table><tbody>
<tr id="tr-lbox-template" width="100%">
<td class="lbel"  width="5%" menu-id="menu-lbox">1. </td>
<td class="ecell" menu-id="menu-ecell" in-list="true"
width="95%">ECELL</td>
</tr></tbody></table>


<table class="cfs-hdr" id="cfs-hdr-template" counter="0"
width="100%" height="100%" ><tbody>
<tr width="100%">
<td class="cfs-type" combo-names="lang-exts"
 contenteditable="true" width="5%">js</td>
<td class="cfs-name" contenteditable="true"
width="95%">CFS name undefined!</td></tr>
<tr style="display:none" width="100%">
<td class="cfs-line-id" width="5%"></td>
<td class="td-editable" code="integrated"
width="95%">Alt+IC - for stream integration</td></tr>
</tbody></table>

<table class="cfs-frag" id="cfs-frag-template" counter="0"
width="100%" height="100%"><tbody></tbody></table>

<table class="cfs-ref" id="cfs-ref-template" counter="0"
width="100%" height="100%"><tbody>
<tr  width="100%">
<td class="cfs-ref-symbol" width="5%">&#8657;</td>
<td class="cfs-ref-code"
width="95%">Assign main code stream from menu</td>
```

```
</tr>
<tr  width="100%">
<td class="cfs-ref-symbol" width="5%">&#8657;</td>
<td class="cfs-ref-data"
width="95%">Assign data stream from menu</td>
</tr>
<tr  width="100%">
<td class="cfs-ref-start" title="Press␣to␣run␣CFS"
width="5%">&#9633;</td>
<td class="cfs-stdout" state="idle"
width="95%">Output for console.log or stdout</td>
</tr>
<tr  width="100%">
<td class="cfs-ref-stop" title="Press␣to␣stop␣CFS␣(if␣running)"
width="5%">&#9633;</td>
<td class="cfs-stderr" state="idle"
width="95%">Output for throw Exception or stderr</td>
</tr>
<tr  width="100%">
<td class="cfs-ref-symbol" width="5%">&#9633;</td>
<td class="cfs-ref-canvas" canvas-id="0" style="display:none"
width="95%">CANVAS ID: 0</td>
</tr>
</tbody></table>

<canvas class="canvas" id="canvas-template" counter="0"
width="100%" height="100%"
 title="Main␣canvas"></canvas>

<table><tbody>
<tr class="code-lines" id="code-lines-template" counter="0"
width="100%">
<td class="cfs-line-id" style="visibility:hidden"
```

```
width="5%"> </td>
<td class="td-editable" contenteditable="true" code="true"
spellcheck="false" width="95%">CODE/DATA</td></tr>
</tbody></table>

<table><tbody>
<tr class="nest-line" id="nest-line-template" counter="0"
width="100%">
<td class="nest-line-symbol" style="visibility:visible"
width="5%">??</td>
<td class="cfs-nest-name" contenteditable="true"
width="95%">Nested CFS name to define!</td></tr></tbody></table>


<table id="vbox-template" class="vbox" width="100%" height="100%"
menu-id="menu-box"><tbody>
<tr width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td></tr>
</tbody></table>


<table id="hbox-template" class="hbox" width="100%" height="100%"
menu-id="menu-box"><tbody>
<tr width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td></tr>
</tbody></table>

<table><tbody><tr id="ecell-tr-template" width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td></tr>
</tbody></table>

<table>
<tbody>
```

```
<tr>
<td id="vbox-ecell-template" class="ecell" menu-id="menu-ecell">

<table class="vbox" width="100%"  height="100%" menu-id="menu-box">
<tbody>
<tr width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td>
</tr>
</tbody>
</table>

</td>
</tr>
</tbody>
</table>


<table><tbody><tr>
<td id="hbox-ecell-template" class="ecell" menu-id="menu-ecell">
<table class="hbox" width="100%"  height="100%"
menu-id="menu-box"><tbody>
<tr width="100%">
<td class="ecell" menu-id="menu-ecell">ECELL</td>
</tr></tbody></table></td></tr></tbody></table>

<table><tbody><tr>
<td id="ecell-template" class="ecell"
menu-id="menu-ecell" >ECELL</td></tr></tbody></table>

<table id="empty-vbox-template" class="vbox" width="100%">
<tbody></tbody></table>

<table><tbody><tr id="empty-tr-hbox-template" width="100%">
```

```
<td><table class="hbox" width="100%"><tbody>
<tr width="100%">
</tr></tbody></table></td></tr></tbody></table>

<table><tbody><tr><td id="empty-ecell-template"
class="ecell"></td></tr></tbody></table>

</div>
```

Define clip item template

```
<div class="ci" id="ci-template" style="display:none;"
menu-id="menu-ci"><hr class="clip-ruler"></hr></div>
```

Styles for titles

```
.du-title {
    border: 2px solid #d4d4d4;
    font-size: 36px;
    font-family: Arial Sans;
    font-style: italic;
    text-align: center;
    color: DarkBlue;
}

div#du-3 td.du-title, div#du-4 td.du-title,
div#du-5 td.du-title {
    border: 2px solid #d4d4d4;
    font-size: 48px;
    font-family: Arial Sans;
    font-style: italic;
    text-align: center;
    color: blue;
}


.cfs-hdr {
    font-size: 16px;
}

.cfs-nest-name,.cfs-name {
    font-size: inherit;
    font-style: italic;
    font-family: "Lucida␣Console";
    text-align: center;
    border-width: 1px;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}

/*
.cfs-name[data] {
    font-size: inherit;
    font-style: italic;
    font-family: "Lucida Console";
    text-align: center;
    color: red;
    border-width: 1px;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}
```

```


.cfs-name:not([data]) {
    font-size: inherit;
    font-style: italic;
    font-family: "Lucida Console";
    text-align: center;
    color: DarkBlue;
    border-width: 1px;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}
*/


.cfs-ref-code:hover, .cfs-ref-data:hover, .cfs-ref-start:hover,
.cfs-ref-stop:hover {
    font-size:150%;
    color: orange;
}

.cfs-ref-code:not([ref-sid]), .cfs-ref-data:not([ref-sid]) {
    font-size: 90%;
    font-family: "Lucida␣Console";
    color: #d4d4d4;
    text-align: center;
    border-width: 1px;
    border-color: black;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}

.cfs-ref-code[ref-sid], .cfs-ref-data[ref-sid],
.cfs-stdout[state="idle"], .cfs-stderr[state="idle"], .cfs-canvas {
    font-size: 90%;
    font-family: "Lucida␣Console";
    color: green;
    text-align: center;
    border-width: 1px;
    border-color: black;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}

.cfs-stdout[state="running"], .cfs-stderr[state="running"] {
    font-size: 90%;
    font-family: "Lucida␣Console";
    color: green;
    text-align: left;
    border-width: 1px;
    border-color: red;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}

.cfs-stdout[state="finished"], .cfs-stderr[state="finished"] {
    font-size: 90%;
    font-family: "Lucida␣Console";
    color: green;
    text-align: left;
    border-width: 1px;
    border-color: orange;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}

.cfs-nest-name {
    text-align: left;
    font-size: 80%;
    color: green;
}


.board-title {
    border: 1.5px solid #d4d4d4;
    //font: 24px Lucida Sans;
    text-align: center;
}


#active-clip {
    background: #999;
}

figure[active] {
    background: #999;
    border: 3px solid orange;
}

```

```
figcaption {
    font-size: inherit;
    font-style: italic;
    font-family: Arial Sans;
    text-align: center;
    color: DarkBlue;
}
```

Styles for edition elements

```
.ecell {
    border-width: 1px;
    border-color: #d4d4d4;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
    font: 16px "Times␣New␣Roman" Lucida Sans;
    color: #d4d4d4;
    text-align: center;
}

.ecell[in-list] {
    border-left-style: none;
}

.lbel {
    border-width: 1px;
    border-color: #d4d4d4;
    border-top-style: solid;
    border-right-style: none;
    border-bottom-style: solid;
    border-left-style: solid;
    font-family: "Lucida␣Console";
    font-size: 90%;
    font-style: normal;
    color: orange;
    vertical-align: text-top;
}

.cfs-type, .cfs-line-id, .nest-line-symbol {
    border-width: 1px;
    border-color: #d4d4d4;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
    font-family: "Lucida␣Console";
    font-size: 100%;
    line-height: 110%;
    font-style: normal;
    color: orange;
    vertical-align: text-top;
}
.p-editable {
    font-family: inherit;
    font-size: 100%;
    line-height: 110%;
    font-style: normal;
    color: DarkBlue;
    vertical-align: text-top;
    text-align: justify;
    hyphens: auto;
}


.td-editable[code] {
    font-size: 100%;
    font-style: normal;
    font-family: "Lucida␣Console";
    color: DarkBlue;
    line-height: 110%;
    text-align:left;
    border-width: 1px;
    border-top-style: solid;
    border-right-style: solid;
    border-bottom-style: solid;
    border-left-style: solid;
}

.ci {
    height: 200px;
    overflow-y: auto;
}

.oscillator {
    background: linear-gradient(to right top, black, white);
    height: 50px;
}
```

Appendix B

$\mathbb{DC}^2$ Design and Implementation – EUniter module

```
/*
 * DC2 - Interactive Wiki Edit Platform
 * Author: Vwadec Skarbek
 *
 * The MIT License (MIT)
 * Copyright 2015-16, Wladyslaw Skarbek, WUT
 *
 * Licensed under the MIT License
 * http://opensource.org/licenses/mit-license.php
 *
 */
```

B.1 `EUniter` module structure

```
/* License JS */

function EUniter(doc,win,config) {
    "use strict";
    var showDU, setFocus, undoVisible, redoVisible, fillMediaCell,
        clipboard, framecontent, mediaboards, mediaWithinBoard,
        inEdition, getAncestorByTagName, getAncestorByClassName,
        liveMediaList;
    var targetDU, mState, specialSymbols, langComments, cfsRunners;
    var undoStack, redoStack, undoTop, redoTop;
```

`EUniter` **constructor**

```
    function initEUniter() {
        showDU = config['show-du'],
        setFocus = config['set-focus'];
        undoVisible = config['undo-visible'];
        redoVisible = config['redo-visible'];
        fillMediaCell = config['fill-media-cell'];
        mediaboards = config['media-boards'];
        mediaWithinBoard = config['media-within-board'];
        inEdition = config['in-edition'];
        getAncestorByClassName = config['get-ancestor-by-class-name'];
        getAncestorByTagName = config['get-ancestor-by-tag-name'];
        cfsRunners = config['cfs-runners'];
        mState = config['modal-state'];
        liveMediaList = config['live-media-list'];


        specialSymbols = config['special-symbols'];
        langComments = config['lang-comments'];

        clipboard =  doc.getElementById('clipboard');

        framecontent =  doc.getElementById('frame-content');

        targetDU = null;
        undoStack = [], redoStack = [],
            undoTop = -1, redoTop = -1;
    }
```

Utility functions of EUniter module

```
/* Utility Functions of EUniter Module */

var utilities = {
        'init-euniter': initEUniter,
        'verify-du-tree': verifyDUTree,
        'push-undo-redo': pushUndoRedo,
        'get-relatives-id': duGetRelativesId,
        undo: undo,
        redo: redo,
        'adjust-size-of-figures': adjustSizeOfFigures,
        'format-labels-from': formatLabelsFrom,
        'class-instance': classInstance,
        'code-data-exec': codeDataExec,
        'code-for-menu': fromStreamCodeToMenuFunction,
};
```

Handlers for items of DU menu

```
/* Handlers for Items of DU Menu */

var menuDUHandlers = {
    'new␣DU': duNew,
    'empty␣first␣DU␣child': duAttachNewAsFirstChild,
    'remove␣DU': duRemove,
    'delete␣DU': duDelete,
    'target␣DU': duMakeTarget,
    'attach␣before␣DU': duAttachBefore,
    'attach␣after␣DU': duAttachAfter,
    'target␣first␣DU␣child': duAttachTargetAsFirstChild,
};
```

Handlers for items of BOX and list menus

```
/* Handlers for Items of BOX Menu */

var menuBOXHandlers = {
    'split␣cell␣up': splitCellUp,
    'split␣cell␣down': splitCellDown,
    'split␣cell␣left': splitCellLeft,
    'split␣cell␣right': splitCellRight,
    'move␣to␣clipboard': moveContentToClipboard,
    'copy␣to␣clipboard': copyContentToClipboard,
};

var menuListHandlers = {
    'add␣item␣before': addItemBefore,
    'add␣item␣after': addItemAfter,
    'remove␣empty␣item': removeEmptyListItem,
};
```

Handlers for items of CFS menu

```
    /* Handlers for Items of CFS Menu */

    var menuCFSHandlers = {
        'Nest␣code␣Stream': nestCodeStream,
        'Add␣code␣lines␣Above': addCodeLinesAbove,
        'Add␣code␣lines␣Below': addCodeLinesBelow,
        'Integrate␣Current␣stream': integrateCurrentStream,
        'Code␣switch␣with␣Data': codeSwitchWithData,
        'Text␣centered': textCentered,
        'Text␣left␣aligned': textLeftAligned,
        'Text␣right␣aligned': textRightAligned,
        'Text␣bigger': textBigger,
        'Text␣smaller': textSmaller,
    };
```

Handlers for items of clip item menu

```
    /* Handlers for Items of Clip Item Menu */

    var menuCIHandlers = {
        'switch␣active␣clip': switchActiveClipItem,
        'clip to trash': removeClipToTrash,
    };
```

Handlers for items of ECELL menu

```
    /* Handlers for Items of ECELL Menu */

    var menuECELLHandlers = {
        'assign code stream': assignCodeStream,
        'set image cell': setImageCell,
        'set video cell': setMovieCell,
        'set audio cell': setSoundCell,
        'move to mediaboard': moveToMediaboard,
        'move to clipboard': moveContentToClipboard,
        'copy to clipboard': copyContentToClipboard,
        'paste empty cell': pasteEmptyCellFromClipboard,
        'remove empty cell': removeEmptyCell,
    };

    var menuECELLShiftHandlers = {
        'init paragraph': initParagraph,
        'set list box': setListBox,
        'start new stream': startNewCFS,
        'new code fragment': newCodeFragment,
        'resume code stream': resumeCodeStream,
        'set code reference': setCodeReference,
        'set live media cell': setLiveMediaCell,
        //'set math cell' : setMathCell,
        //'set svg cell'  : setSVGCell,
    };
```

UNDO/REDO functions

```
    /* UNDO/REDO Functions */
```

Exporting groups of functions

```
    return {
        'utils': utilities,

      'menu-du': menuDUHandlers,

      'menu-box': menuBOXHandlers,

        'menu-lbox': menuListHandlers,

        'menu-ecell': menuECELLHandlers,

        'menu-ecell-shift': menuECELLShiftHandlers,

        'menu-cfs': menuCFSHandlers,

        'menu-ci': menuCIHandlers,

    };
}
```

B.2 Utility functions

Nested structure

```
var opposite = {'left':'right', 'right':'left', 'up':'down',
                        'root': 'root'};

function duExists(duId) {
    if (duId=='du-0') return false;
    return true;
}
/* Creating Enumerable Class Instances */
/* DU Tree Structure Verification */
/* Functions for DU Inserting and Deleting */
```

```
/* Function cloneEmptyDU */
```

```
/* Function for Code Data Execution */
```

Creating enumerable class instances

```
function classInstance(className,subClassName) {
    var template = doc.getElementById(className+'-template'),
        node = null;
    if (template) {
        node = template.cloneNode(true);
        if (template.hasAttribute('counter')) {
            var countingElement = doc.getElementById('global-counter'),
            counter =
                countingElement.getAttribute('global-counter')-0+3;
            countingElement.setAttribute('global-counter',''+counter);
            node.setAttribute('counter',''+counter);
            node.id = className+'-'+counter;
        }
        if (subClassName) node.className = subClassName;
    }
    return node;
}
```

B.2.1 Functions for code and data handling

Function for code and data execution

```
function codeDataExec(evt) {
    var cfsRef = evt.target.parentElement.parentElement.parentElement,
        runId = cfsRef.getAttribute('counter');
    switch(evt.target.className) {
        case 'cfs-ref-code': case 'cfs-ref-data':
            codeDataAccess(evt);
            break;
        case 'cfs-ref-start':
            codeDataRunning(evt);
            break;
        case 'cfs-ref-stop':
            if (cfsRunners.stopRequest[runId]) {
                cfsRunners.stopRequest[runId] = true;
            }
            break;
        case 'cfs-ref-stdout':
            if (cfsRunners.state[runId]=='finished') {
                evt.target.textContent =
                        'Output␣for␣console.log␣or␣stdout';
                evt.target.setAttribute('state','idle');
            }
            break;
        case 'cfs-ref-stderr':
            if (cfsRunners.state[runId]=='finished') {
                evt.target.textContent =
                  'Output␣for␣throw␣Exception␣or␣stderr';
                evt.target.setAttribute('state','idle');
            }
            break;
    }
}
```

Code and data access

```
function codeDataAccess(evt) {
    var cfsRefX = evt.target,
        sid = cfsRefX.getAttribute('ref-sid'), docx;

    if (cfsRefX.className=='cfs-ref-data') {
        docx = doc;
    } else {
        docx = dc2Globals.docs['Description'];
    }
    doc.cfsLineState.ref = cfsRefX;

    if (docx.cfsLineState.lastId!=sid ||
        docx.cfsLineState.modified[sid]) {
        fillCFSLineSelector(docx,doc.cfsLineState.sel,sid);
        docx.cfsLineState.lastId = sid;
        delete docx.cfsLineState.modified[sid];
    } else {
        changeCFSDefault(doc.cfsLineState.sel,0);
    }

    doc.cfsLineState.div.style.left =
                        evt.clientX+doc.body.scrollLeft+10;
    doc.cfsLineState.div.style.top =
                        evt.clientY+doc.body.scrollTop+10;

    doc.cfsLineState.div.style.display = 'block';
    mState.on = true; mState.el = doc.cfsLineState.div;

}
```

Assignment CFS to menu item

```
function fromStreamCodeToMenuFunction(itemName) {
    var codeLines = getCfsContent(doc,doc.cfsState.id,'js',false),
        rex = new RegExp('//[␣]*'+itemName);

    if (!rex.test(codeLines[0])) {
        cl('not␣menu␣item?:␣'+codeLines[0]);
        return null;
    }
    var code = codeLines.join('\n'),
        codePrefix =
            "try␣{\n"+
            "(function()␣{\n",
        codeSuffix =
            "\n␣␣␣␣return␣main;})();\n"+
            "}␣catch(err)␣{"+
            "alert(err.message+'␣--->␣line:␣'+(err.lineNumber-2));}",
        codeId  =
            "//#␣sourceURL=my-"+doc.cfsState.id+".js",
        evalCode  =  codePrefix+code+codeSuffix+codeId;

    //cl('CODE for EVALUATION:\n'+evalCode);

    return eval(evalCode);
}
```

Code and data running

```
function codeDataRunning(evt) {

    var trStdOut = evt.target.parentElement,
        cfsStdOut = trStdOut.lastElementChild,
        trData = trStdOut.previousElementSibling,
        trCode = trData.previousElementSibling;
        if (!trCode.lastElementChild.hasAttribute('ref-sid'))  return;
    var cfsStdErr = trStdOut.nextElementSibling.lastElementChild,
        docx = dc2Globals.docs['Description'],
        codeSid, codeHdr, tr, ext, codeLines, code,
        dataSid, dataHdr, dataLines = null;

    codeSid =  trCode.lastElementChild.getAttribute('ref-sid');
    codeHdr = docx.getElementById('cfs-hdr-'+codeSid);
    tr = codeHdr.firstElementChild.firstElementChild,
    ext = tr.firstElementChild.textContent;
    codeLines = getCfsContent(docx,codeSid,ext,false);
    code = codeLines.join('\n');

    if (trData.lastElementChild.hasAttribute('ref-sid')) {
        dataSid =  trData.lastElementChild.getAttribute('ref-sid');
        dataHdr = docx.getElementById('cfs-hdr-'+dataSid);
        tr = dataHdr.firstElementChild.firstElementChild,
        ext = tr.firstElementChild.textContent;
        dataLines = getCfsContent(doc,dataSid,ext,false);
    }

    var runId =
    trStdOut.parentElement.parentElement.getAttribute('counter');
    cfsRunners.stopRequest[runId] = false;

    var stdin, stderr, console;

    stdin = {
        id: runId,
        lineIndx: 0,
        read: function() {
            if (!dataLines) return null;
            stdin.lineIndx = dataLines.length;
            return dataLines.join('\n');
        },
        readLine: function() {
            if (!dataLines) return null;
            if (stdin.lineIndx==dataLines.length) return null;
            return dataLines[stdin.lineIndx++];
        },
    }

    stderr = {
        id: runId,
        write: function(obj) {
            var span = doc.createElement('SPAN');
            span.innerHTML = ''+obj+'<br>';
            cfsStdErr.appendChild(span);
        },
        clear: function() {
            cfsStdErr.innerHTML = '';
        },
    }

    console = {
        id: runId,
        log: function(obj) {
            var span = doc.createElement('SPAN');
            span.innerHTML = ''+obj+'<br>';
            cfsStdOut.appendChild(span);
        },
        clear: function() {
            cfsStdOut.innerHTML = '';
        },
        isStopRequest: function() {
            return cfsRunners.stopRequest[runId];
        },
    }


    var codePrefix =
            "try␣{\n"+
            "(function()␣{\n",
        codeSuffix =
            "\n␣␣␣␣return␣main;})()();\n"+
            "}␣catch(err){stderr.write("+
                "err.message+' ---> line: '+(err.lineNumber-2));}",
            codeId =
                "//# sourceURL=my-"+codeSid+".js";

    if (cfsStdOut.getAttribute('state')=='idle') {
        console.clear(); stderr.clear();
    }
    console.log('-------- Restart for application ('+codeSid+')');
    stderr.write('---Restart---');

    cfsStdOut.setAttribute('state','running');
    cfsStdErr.setAttribute('state','running');
    cfsRunners.state[runId] = 'running';
    var evalCode = codePrefix+code+codeSuffix+codeId;
    //cl('CODE for EVALUATION:\n'+evalCode);
    eval(evalCode);
    cfsStdOut.setAttribute('state','finished');
    cfsStdErr.setAttribute('state','finished');
    cfsRunners.state[runId] = 'finished';
    cfsRunners.stopRequest[runId] = false;
}
```

B.2.2 Verification of DU tree structure

Verify DU tree

```
function verifyDUTree(div,frameName) {
    var msg;
    var dus = div.getElementsByClassName('du');
    if (dus.length==0) {
        msg = 'Missing DU in the frame: '+frameName;
        alert(msg); throw msg;
    }
    var root = doc.getElementById(doc.duRI);
    if (!root) {
          msg = 'Missing␣root␣in␣the␣frame:␣'+frameName;
          alert(msg); throw msg;
     }
     var visited = {}, i = dus.length;
     while (i--) {
          var id = dus[i].id;
          if (id in visited) {
               msg = 'There␣two␣document␣units␣with␣id:␣'+id;
               alert(msg); throw msg;
          } else {
               visited[id] = 1;
          }
     }
     function traverse(node) {
          visited[node.id] += 1;
          if (visited[node.id]==3) {
               msg = 'DU␣structure␣is␣not␣a␣tree';
               alert(msg); throw msg;
          }
          var childId = node.getAttribute('down'), child;
          if (duExists(childId)) {
               do {
                    child = doc.getElementById(childId);
                    traverse(child);
                    childId = child.getAttribute('right');
               } while(duExists(childId));
          }
     }
     var node = root, nodeId = root.id;
     while (true) {
          traverse(node);
          nodeId = node.getAttribute('right');
          if (duExists(nodeId)) node = doc.getElementById(nodeId);
```

```
        else break;
    }
    i = dus.length;
    while (i--) {
        var id = dus[i].id;
        if (visited[id]==1) {
            msg = 'Document␣unit'+id+'␣is␣not␣reachable';
            alert(msg); throw msg;
        }
    }
}
```

Get DU relatives id

```
function duGetRelativesId(du,what) {
    var rlvs = [], el = du, id;
    switch(what) {
        case 'children':
            id = el.getAttribute('down');
            while (duExists(id)) {
                rlvs.push(id);
                el = doc.getElementById(id);
                id = el.getAttribute('right');
            }
            return rlvs;
        case 'ancestors':
            id = el.getAttribute('up');
            while (duExists(id)) {
                rlvs.push(id);
                el = doc.getElementById(id);
                id = el.getAttribute('up');
            }
            return rlvs;
        case 'siblings':
            var idLeft = el.getAttribute('left'), id = el.id;
            while (duExists(idLeft)) {
                id = idLeft;
                el = doc.getElementById(id);
                idLeft = el.getAttribute('left');
            }
            rlvs.push(id); id = el.getAttribute('right');
            while (duExists(id)) {
                rlvs.push(id);
                el = doc.getElementById(id);
                id = el.getAttribute('right');
            }
            return rlvs;
    }
}
```

B.3 Functions for DU creation, insertion, and deletion

We assume that before insertion or deletion the element y is already detached from its last location in DU tree. Why? Since it is more frequent creation of a new unit than moving the existing from one place to another and naturally as a new object it is isolated from the other units.

Insert DU sibling

```
function duInsertSibling(y,x,where) {
    var owhere = opposite[where];
    var zId = x.getAttribute(where);
    if (duExists(zId)) {
        var z = doc.getElementById(zId);
        z.setAttribute(owhere,y.id);
    }
    x.setAttribute(where,y.id);
    y.setAttribute(where,zId); y.setAttribute(owhere,x.id);
    y.setAttribute('up',x.getAttribute('up'));
    if (where=='left' && !duExists(y.getAttribute('left'))) {
        var duParentId = x.getAttribute('up');
        if (duExists(duParentId)) {
            var duParent = doc.getElementById(duParentId);
            duParent.setAttribute('down',y.id);
        }
    }
    /*
    //console.log('inserted sibling:'); console.log(y);
    //console.log('on side: '+where+ ' of unit:'); console.log(x);
    //console.log('the opposite is: '+opposite[where]);
    */
}
```

Insert first born DU

```
function duInsertFirstBorn(y,x) {
    var down = x.getAttribute('down');
    if (duExists(down)) return;
    y.setAttribute('up',x.id);
    x.setAttribute('down',y.id);
}
```

Insert DU there

```
function duInsertThere(y,x,where) {
    if (where=='down') {
        duInsertFirstBorn(y,x);
    } else {
        duInsertSibling(y,x,where)
    }
}
```

Get id for adjacent DU

```
function duGetAdjacentId(du) {
    //console.log('in duGetAI');

    if (du.id==doc.duRI) return ['none','du-0'];
    var whereId = du.getAttribute('right');
    if (duExists(whereId)) return ['right',whereId];
    whereId = du.getAttribute('left');
    if (duExists(whereId)) return ['left',whereId];
    whereId = du.getAttribute('up');
    if (duExists(whereId)) return ['up',whereId];
}
```

Detach DU from DU tree

```
function duDetach(y) {

    var left = y.getAttribute('left'),
        right = y.getAttribute('right'),
        up = y.getAttribute('up');
    if (duExists(left)) {
        var leftDU = doc.getElementById(left);
        leftDU.setAttribute('right',right);
        y.setAttribute('left','du-0');
    }
    if (duExists(right)) {
        var rightDU = doc.getElementById(right);
        rightDU.setAttribute('left',left);
        y.setAttribute('right','du-0');
    }
    if (duExists(up)) {
        var upDU = doc.getElementById(up);
        if (upDU.getAttribute('down')==y.id) {
            upDU.setAttribute('down',right);
        }
```

```
            y.setAttribute('up','du-0');
        }
}
```

Remove detached DU

```
function duRemoveDetached(du,how) {
        if (how=='trash') {
            du.setAttribute('removed-to','trash');
            //du.parentNode.removeChild(du);
            // REDO fails if we remove it from DOM

        } else {
            du.setAttribute('removed-to','forest');
            var root = doc.getElementById(doc.duRI);
            duInsertSibling(du,root,'right');
        }
}
```

Get DU or class element on path

```
function getDUonPath(path) {
        var i = path.length;
        while(i--) {
            if (path[i].className=='du') {
                return path[i];
            }
        }
        return null;
}

function getElementOnPath(path,cn) {
        var i = path.length;
        while(i--) {
            if (path[i].className==cn) {
                return path[i];
            }
        }
    return null;
}
```

There is a question raised for cloning nodes in HTML DOM: what happens if `id` attribute is defined which should be unique? To avoid conflicts change the value of `id` in the new node before its appending somewhere into the document.

Clone empty DU

```
function duCloneEmpty() {
    var du = classInstance('du');
    framecontent.appendChild(du);
    cl('du␣counter:␣'+du.getAttribute('counter'));
    cl('du␣template␣counter:␣'+
        doc.getElementById('du-template').getAttribute('counter'));
    return du;
}
```

B.4 Handlers for items of DU menu

New DU

```
function duNew(mevt,check) {
    var ctxDU = null;
    if (check) {
         ctxDU = getDUonPath(mevt.elementPath);
         if (ctxDU && ctxDU.id!=doc.duRI) return true;
         return false;
    }
    ctxDU = getDUonPath(mevt.elementPath);
    var du = duCloneEmpty();

    duInsertSibling(du,ctxDU,'right');
    showDU(du);

    function undo() {
        duDetach(du); duRemoveDetached(du,'trash');
        showDU(ctxDU); setFocus(mevt);
    }

    function redo() {
        du.removeAttribute('removed-to');
        duInsertSibling(du,ctxDU,'right');
        showDU(du);
    }

    return [undo,redo];
}
```

Remove DU

```
function duRemove(mevt,check) {
    var ctxDU = null;
    if (check) {
        ctxDU = getDUonPath(mevt.elementPath);
        if (!ctxDU) return false;
        if (!ctxDU.hasAttribute('removed-to') && ctxDU.id!=doc.duRI)
            return true;
        return false;
    }
    ctxDU = getDUonPath(mevt.elementPath);

    //console.log('in duRemove');

    var whereId = duGetAdjacentId(ctxDU);
    var where = whereId[0], adId = whereId[1];
    var du = doc.getElementById(adId);

    duDetach(ctxDU);
    duRemoveDetached(ctxDU,'forest');
    showDU(du);

    function undo() {
        duDetach(ctxDU);
        ctxDU.removeAttribute('removed-to');
        duInsertThere(ctxDU,du,opposite[where]);
        showDU(ctxDU); setFocus(mevt);
    }

    function redo() {
        duDetach(ctxDU);
        duRemoveDetached(ctxDU,'forest');
        showDU(du);
    }

    return [undo,redo];
}
```

Delete DU

```
function duDelete(mevt,check) {
    var ctxDU = null;
    if (check) {
        ctxDU = getDUonPath(mevt.elementPath);
        if (!ctxDU || !ctxDU.hasAttribute('removed-to')) return false;
        if (ctxDU.getAttribute('removed-to')=='forest') return true;
        return false;
    }
    ctxDU = getDUonPath(mevt.elementPath);
    //console.log('in duDelete');

    var whereId = duGetAdjacentId(ctxDU);
    var where = whereId[0], adId = whereId[1];
    var du = doc.getElementById(adId);

    duDetach(ctxDU);
    duRemoveDetached(ctxDU,'trash');
    showDU(du);

    function undo() {
        duDetach(ctxDU);
        ctxDU.setAttribute('removed-to','forest');
        duInsertThere(ctxDU,du,opposite[where]);
        showDU(ctxDU); setFocus(mevt);
    }

    function redo() {
        duDetach(ctxDU);
        duRemoveDetached(ctxDU,'trash');
        showDU(du);
    }

    return [undo,redo];
}
```

Set visible DU as target

```
function duMakeTarget(mevt,check) {
    var ctxDU = null;
    if (check) {
        ctxDU = getDUonPath(mevt.elementPath);
        if (ctxDU && ctxDU.id!=doc.duRI) return true;
        return false;
    }
    ctxDU = getDUonPath(mevt.elementPath);


    var previousTargetDU = targetDU;
    targetDU = ctxDU;

    function undo() {
        targetDU = previousTargetDU;
    }

    function redo() {
        targetDU = ctxDU;
    }

    return [undo,redo];

}
```

Attach before DU

```
function duAttachBefore(mevt,check) {
    if (check) {
        if (!targetDU) return false;
        var ctxDU = getDUonPath(mevt.elementPath);
        if (!ctxDU) return false;
        var leftId = ctxDU.getAttribute('left');
        if (leftId==targetDU.id || ctxDU.id==targetDU.id ||
            ctxDU.id==doc.duRI) return false;
        return true;
    }

    return duAttachSibling(mevt,'left');
}
```

Attach after DU

```
function duAttachAfter(mevt,check) {
    if (check) {
        if (!targetDU) return false;
        var ctxDU = getDUonPath(mevt.elementPath);
        if (!ctxDU) return false;
        var rightId = ctxDU.getAttribute('right');
        if (rightId==targetDU.id || ctxDU.id==targetDU.id ||
            ctxDU.id==doc.duRI) return false;
        return true;
    }

    return duAttachSibling(mevt,'right');
}
```

Attach sibling DU

```
function duAttachSibling(mevt,side) {
    var ctxDU = getDUonPath(mevt.elementPath);

    //console.log('to attach '+targetDU+' on side '+side);

    var whereId = duGetAdjacentId(targetDU),
        where = whereId[0], adId = whereId[1],
        previousTargetDU;

    function undo() {
        targetDU = previousTargetDU;
        duDetach(targetDU);
        var adElement = doc.getElementById(adId);
        duInsertThere(targetDU,adElement,opposite[where]);
        showDU(ctxDU); setFocus(mevt);
    }

    function redo() {
          duDetach(targetDU);
          duInsertSibling(targetDU,ctxDU,side);
          showDU(targetDU);
          previousTargetDU = targetDU; targetDU = null;
     }

     redo();

     return [undo,redo];
}
```

Attach new DU as first child

```
function duAttachNewAsFirstChild(mevt,check) {
     var ctxDU;
     if (check) {
          ctxDU = getDUonPath(mevt.elementPath);
          if (!ctxDU) return false;
          if (duExists(ctxDU.getAttribute('down'))) return false;
          return true;
     }
     ctxDU = getDUonPath(mevt.elementPath);
     var du = duCloneEmpty();
     duInsertFirstBorn(du,ctxDU);
     showDU(du);

     function undo() {
          duDetach(du);
          duRemoveDetached(du,'trash');
          showDU(ctxDU); setFocus(mevt);
     }

     function redo() {
          du.removeAttribute('removed-to');
```

```
            duInsertFirstBorn(du,ctxDU);
            showDU(du);
        }

        return [undo,redo];
}
```

Attach target as first child

```
function duAttachTargetAsFirstChild(mevt,check) {
        var ctxDU;
        if (check) {
            if (!targetDU) return false;
            ctxDU = getDUonPath(mevt.elementPath);
            if (!ctxDU) return false;
            if (duExists(ctxDU.getAttribute('down'))) return false;
            return true;
        }
        ctxDU = getDUonPath(mevt.elementPath);

        //console.log('in duAttach TAFC');
        var whereId = duGetAdjacentId(targetDU),
            where = whereId[0], adId = whereId[1],
            previousTargetDU;

        function undo() {
            targetDU = previousTargetDU;
            duDetach(targetDU);
            var adElement = doc.getElementById(adId);
            duInsertThere(du,adElement,opposite[where]);
            showDU(ctxDU); setFocus(mevt);
        }

        function redo() {
            duDetach(targetDU);
            duInsertFirstBorn(targetDU,ctxDU);
            showDU(targetDU);
            previousTargetDU = targetDU;
            targetDU = null;
        }

        redo();

        return [undo,redo];

   }
```

B.5 UNDO/REDO small engine

UNDO engine operation

```
function undo() {
     if (undoTop<0) return;
     var undo = undoStack[undoTop];
     undoTop -= 1;
     undo();
     if (undo.figResizable) adjustSizeOfFigures(undo.figTargetElement);
     redoVisible(true);
     if (undoTop<0) undoVisible(false);
}
```

REDO engine operation

```
function redo() {
     if (redoTop<0 || redoTop==undoTop) return;
     var redo = redoStack[undoTop+1];
     undoTop += 1;
     redo();
     if (redo.figResizable) adjustSizeOfFigures(redo.figTargetElement);
     undoVisible(true);
```

```
    if (redoTop==undoTop) redoVisible(false);
}
```

Push undo/redo engine operation

```
function pushUndoRedo(undoRedo) {
    if (!undoRedo) return;

    var i = redoTop-undoTop;
    while (i--) {
        undoStack.pop();
        redoStack.pop();
    }
    undoStack.push(undoRedo[0]);
    redoStack.push(undoRedo[1]);

    undoTop += 1; redoTop = undoTop;
    undoVisible(true); redoVisible(false);
}
```

B.6 Handlers for items of BOX menu

Get index of ECELL

```
function getIndexOfEE(path,nest) {
    var n = path.length, k = 0, i;
    for (i=0;i<n;i++) {
        var cn = path[i].className;
        if (cn=='ecell') {
            if (k==nest) return i;
            k += 1;
        }
    }
    return -1;
}
```

Get depth of ECELL

```
function getDepthOfEE(path,ind) {
    var n = path.length, d = -1, i;
    for (i=ind+1;i<n;i++) {
        var cn = path[i].className;
        if (cn=='vbox' || cn=='hbox') {
            d += 1;
        }
    }
    return d;
}
```

```
/* Utils for Cell Management */
```

Split cell up

```
function splitCellUp(mevt,check) {
    if (check) {
        var i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        return true;
    }
    return splitCellVertical(mevt,'up');
}
```

Split cell down

```
function splitCellDown(mevt,check) {
    if (check) {
        var i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        return true;
    }
    return splitCellVertical(mevt,'down');
}
```

Split cell vertically

```
function splitCellVertical(mevt,where) {
    var i = getIndexOfEE(mevt.elementPath,mevt.nest),
        depth = getDepthOfEE(mevt.elementPath,i),
        ee = mevt.elementPath[i], eeTr = ee.parentElement,
        eeTbody = eeTr.parentElement, ctxBox = eeTbody.parentElement,
        ecell, tr, tbody, vbox, vboxEcell, newTr;
    if (ctxBox.className=='vbox') {
        tr = cloneCellTemplate('ecell-tr-template');
        ecell = tr.firstElementChild;
        ecell.setAttribute('depth',depth);

        eeTbody.insertBefore(tr,eeTr);
        if (where=='down') {
            eeTbody.removeChild(eeTr);
            eeTbody.insertBefore(eeTr,tr);
        }

    } else {
        vboxEcell = cloneCellTemplate('vbox-ecell-template');
        vbox = vboxEcell.firstElementChild;
        tbody = vbox.firstElementChild; tr = tbody.firstElementChild;
        ecell = tr.firstElementChild;
        ecell.setAttribute('depth',depth+1);

        newTr = doc.createElement('TR');

        eeTr.insertBefore(vboxEcell,ee); eeTr.removeChild(ee);
        newTr.appendChild(ee); ee.setAttribute('depth',depth+1);
        if (where=='up') {
            tbody.appendChild(newTr);
        } else {
            tbody.insertBefore(newTr,tr);
        }
    }


    function undo() {
        if (ctxBox.className=='vbox') {
            eeTbody.removeChild(tr);
            tr.setAttribute('removed-to','trash');
        } else {
            newTr.removeChild(ee);
            eeTr.insertBefore(ee,vboxEcell);
            ee.setAttribute('depth',depth);
            eeTr.removeChild(vboxEcell);
            vboxEcell.setAttribute('removed-to','trash');
        }
    }

    function redo() {
        if (ctxBox.className=='vbox') {
            tr.removeAttribute('removed-to');
            eeTbody.insertBefore(tr,eeTr);
            if (where=='down') {
                eeTbody.removeChild(eeTr);
                eeTbody.insertBefore(eeTr,tr);
            }
        } else {
            vboxEcell.removeAttribute('removed-to');
            eeTr.insertBefore(vboxEcell,ee);
            eeTr.removeChild(ee);
            newTr.appendChild(ee); ee.setAttribute('depth',depth+1);
        }
    }

    return [undo,redo];
}
```

Clone cell template

```
function cloneCellTemplate(templateName) {
    var elTemplate = doc.getElementById(templateName),
        el = elTemplate.cloneNode(true);
    return el;
}
```

Split cell horizontally

```
function splitCellLeft(mevt,check) {
    if (check) {
        var i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        return true;
    }
    var result = splitCellHorizontal(mevt,'left');
    splitCellLeft.figResizable = true;
    splitCellLeft.figTargetElement =
            splitCellHorizontal.figTargetElement;
    return result;
}

function splitCellRight(mevt,check) {
    if (check) {
        var i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        return true;
    }
    var result = splitCellHorizontal(mevt,'right');
    splitCellRight.figResizable = true;
    splitCellRight.figTargetElement =
            splitCellHorizontal.figTargetElement;
    return result;
}

function splitCellHorizontal(mevt,where) {
    var i = getIndexOfEE(mevt.elementPath,mevt.nest),
        ee = mevt.elementPath[i],
        depth = getDepthOfEE(mevt.elementPath,i);

    var eeParent, eeBox, targetParent,
        ecell = null, hboxEcell = null;


    redo();

    function redo() {

        eeParent = ee.parentElement;
        eeBox = eeParent.parentElement.parentElement;

        redo.figResizable = true;
        redo.figTargetElement = eeBox;
        splitCellHorizontal.figTargetElement = eeBox;

        if (eeBox.className=='hbox') {
            if (ecell) {
                ecell.removeAttribute('removed-to');
            } else {
                ecell = cloneCellTemplate('ecell-template');
            }
            ecell.setAttribute('depth',depth);
            var eeBigBrother = ee.nextElementSibling;
            if (where=='left') {
                eeParent.insertBefore(ecell,ee);
            } else {
                if (eeBigBrother) {
                    eeParent.insertBefore(ecell,eeBigBrother);
                } else {
                    eeParent.appendChild(ecell);
                }
            }
        } else {
            if (hboxEcell)  {
                hboxEcell.removeAttribute('removed-to');
            } else {
                hboxEcell = cloneCellTemplate('hbox-ecell-template');
            }
            hboxEcell.setAttribute('depth',depth);
            eeParent.removeChild(ee); eeParent.appendChild(hboxEcell);
            var targetGrandpa =
            hboxEcell.firstElementChild.firstElementChild;
            targetParent = targetGrandpa.firstElementChild;
            var targetSibling = targetParent.firstElementChild;
            targetSibling.setAttribute('depth',depth-0+1);
            ee.setAttribute('depth',depth-0+1);
            if (where=='right') {
                targetParent.insertBefore(ee,targetSibling);
            } else {
                targetParent.appendChild(ee);
            }
        }

    }

    function undo() {
        undo.figResizable = true;
```

```
          undo.figTargetElement = eeBox;

          if (eeBox.className=='hbox') {
               eeParent.removeChild(ecell);
               ecell.setAttribute('removed-to','trash');
          } else {
               targetParent.removeChild(ee);
               eeParent.removeChild(hboxEcell);
               hboxEcell.setAttribute('removed-to','trash');
               eeParent.appendChild(ee);
               ee.setAttribute('depth',depth);
          }
     }

     return [undo,redo];

}
```

B.7 Handlers for list boxes

Remove empty list item

```

function removeEmptyListItem(mevt,check) {
     if (check) {
          if (inEdition['lbel']) return false;
          var lbel = mevt.element, tr = lbel.parentElement,
               ecell = tr.lastElementChild;
          if (ecell.childElementCount>0) return false;
          return true;
     }
     var lbel = mevt.element, tr = lbel.parentElement,
          tbody = tr.parentElement, lbox = tbody.parentElement,
          ecell = lbox.parentElement, lboxRemoved = false,
          nextTr = tr.nextElementSibling;

     if (tbody.childElementCount==1) {
          ecell.removeChild(lbox);
          ecell.textContent = 'ECEll';
          lbox.setAttribute('remove-to','trash');
          lboxRemoved = true;
     } else {
          tbody.removeChild(tr);
          tr.setAttribute('remove-to','trash');
          if (nextTr) formatLabelsFrom(nextTr,lbel.textContent);
     }

     function undo() {
          if (lboxRemoved) {
               lbox.removeAttribute('remove-to');
               ecell.textContent = ''; ecell.normalize();
               ecell.appendChild(lbox);
          } else {
               tr.removeAttribute('remove-to');
               if (nextTr) {
                   tbody.insertBefore(tr,nextTr);
                   formatLabelsFrom(tr,lbel.textContent);
               } else {
                   tbody.appendChild(tr);
               }
          }
     }

     function redo() {
          if (lboxRemoved) {
               ecell.removeChild(lbox);
               ecell.textContent = 'ECEll';
               lbox.setAttribute('remove-to','trash');
          } else {
               tbody.removeChild(tr);
               tr.setAttribute('remove-to','trash');
               if (nextTr) formatLabelsFrom(nextTr,lbel.textContent);
          }
     }

     return [undo,redo];
}
```

Add item before

```
function addItemBefore(mevt,check) {
    return addItem(mevt,check,'before');
}
```

Add item after

```
function addItemAfter(mevt,check) {
    return addItem(mevt,check,'after');
}
```

Add item

```
function addItem(mevt,check,where) {
    if (check) {
        if (inEdition['lbel']) return false;
        return true;
    }

    var lbel = mevt.element, itemLabel = lbel.textContent,
        tr = lbel.parentElement,
        tbody = tr.parentElement,
        trLboxTemplate = doc.getElementById('tr-lbox-template'),
        newTr = trLboxTemplate.cloneNode(true);
    newTr.removeAttribute('id');

    redo();

    function redo() {
        if (newTr.hasAttribute('removed-to')) {
            newTr.removeAttribute('removed-to');
        }
        if (where=='before') {
            tbody.insertBefore(newTr,tr);
            formatLabelsFrom(newTr,itemLabel);
        } else {
            var nextTr = tr.nextElementSibling;
            if (nextTr) {
                tbody.insertBefore(newTr,nextTr);
            } else {
                tbody.appendChild(newTr);
            }
            formatLabelsFrom(tr,itemLabel);
        }
    }

    function undo() {
        tbody.removeChild(newTr);
        newTr.setAttribute('removed-to','trash');
        formatLabelsFrom(tr,itemLabel);
    }

    return [undo,redo];
}
```

Parse label

```
function parseLabel(alabel) {
    var res = alabel.match(/([^\d]*)(\d+)([^\d]*)/);
    if (res) {
        return {type:'numeric', prefix:res[1],
                suffix:res[3], value:res[2]};
    }
    res = alabel.match(/([^A-Za-z]*)([A-Za-z])([^A-Za-z ]*)/);
    if (res) {
        return {type:'alpha', prefix:res[1],
                suffix:res[3], value:res[2]};
    }
    return {type:'fixed', prefix:'',
            suffix:'', value:alabel};
}
```

Format labels from

```
function formatLabelsFrom(begTr,alabel) {
    var res = parseLabel(alabel);
    var cur = begTr, val;

    if (res.type=='numeric') {
        val = res.value-0;
    } else if (res.type=='alpha') {
        val = res.value.charCodeAt(0);
    } else {
        val = res.value;
    }
    do {
        if (res.type=='numeric') {
            cur.firstElementChild.textContent =
                res.prefix+(val+'')+res.suffix;
        } else if (res.type=='alpha') {
            cur.firstElementChild.textContent =
                res.prefix+String.fromCharCode(val)+res.suffix;
        } else {
            cur.firstElementChild.textContent = val;
        }

        if (res.type!='fixed') val += 1;

        cur = cur.nextElementSibling;

    } while (cur);
}
```

B.8 API for clipboard

Rules at clipboard handling:

- Top ecell remains,.
- Content is moved, i.e. vbox table, hbox table, p , img, etc.

Put into clipboard

```
function putIntoClipboard(content) {
    var clipTemplate = doc.getElementById('ci-template'),
        clipItem = clipTemplate.cloneNode(true),
        activeClip = doc.getElementById('active-clip'), ruler;
    if (activeClip) {
        activeClip.removeAttribute('id');
        ruler = activeClip.firstElementChild;
        ruler.className = 'clip-ruler';
    }
    clipItem.id = 'active-clip';
    ruler = clipItem.firstElementChild;
    ruler.className = 'active-clip-ruler';
    clipItem.style.display='block';
    clipItem.appendChild(content);
    var f =clipboard.firstElementChild.nextElementSibling;
    if (f) {
        clipboard.insertBefore(clipItem,f);
    } else {
        clipboard.appendChild(clipItem);
    }
    return {active: clipItem, lastActive: activeClip};
}
```

Undo put into clipboard

```
function undoPutIntoClipboard(result) {
    var activeClip = result.active, lastActive = result.lastActive,
        content = activeClip.lastElementChild, ruler;
    activeClip.removeAttribute('id');
    activeClip.removeChild(content);
    clipboard.removeChild(activeClip);
    if (lastActive) {
        lastActive.id = 'active-clip';
        ruler = lastActive.firstElementChild;
        ruler.className = 'active-clip-ruler';
    }
    return [content,activeClip];
}
```

Redo put into clipboard

```
function redoPutIntoClipboard(content,clip) {
    clip.appendChild(content);
    var activeClip = doc.getElementById('active-clip'), ruler;
    if (activeClip) {
        activeClip.removeAttribute('id');
        ruler = activeClip.firstElementChild;
        ruler.className = 'clip-ruler';
    }
    clip.id = 'active-clip';
    ruler = clip.firstElementChild;
    ruler.className = 'active-clip-ruler';
    var f =clipboard.firstElementChild.nextElementSibling;
    if (f) {
        clipboard.insertBefore(clip,f);
    } else {
        clipboard.appendChild(clip);
    }
    return {active: clip, lastActive: activeClip};
}
```

Item on path which belongs to clipboard

```
function itemWithinClipboard(elementPath) {
    var i = elementPath.length;
    while (i--) {
        var el = elementPath[i];
        if (el.className=='ci') return el;
    }
    return null;
}
```

Move/copy content to clipboard

```
function moveContentToClipboard(mevt,check) {
    return contentToClipboard(mevt,check,'move');
}
function copyContentToClipboard(mevt,check) {
    return contentToClipboard(mevt,check,'copy');
}
```

Content to clipboard

```
function contentToClipboard(mevt,check,mode) {
    if (check) {
        if (itemWithinClipboard(mevt.elementPath)) return false;
        var i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        var ecell = mevt.elementPath[i];
        if (ecell.childElementCount==0) return false;
        return true;
    }


    var i = getIndexOfEE(mevt.elementPath,mevt.nest),
        ecell = mevt.elementPath[i],
        content = ecell.firstElementChild, contentCopy, result,
        undoResult, clip;

    if (mode=='move') {
        ecell.removeChild(content);
        ecell.textContent = 'ECELL';
        contentCopy = content;
    } else {
        contentCopy = content.cloneNode(true);
    }

    result = putIntoClipboard(contentCopy);

    function undo() {
        var cc = undoPutIntoClipboard(result);
        contentCopy = cc[0]; clip = cc[1];
        clip.setAttribute('removed-to','trash');
        if (mode=='move') {
            ecell.textContent = ''; ecell.normalize();
            ecell.appendChild(contentCopy);
        } else {
            contentCopy.setAttribute('removed-to','trash');
        }
    }

    function redo() {
        if (mode=='move') {
            content = ecell.firstElementChild;
            ecell.removeChild(content);
            ecell.textContent = 'ECELL';
            contentCopy = content;
        } else {
            contentCopy.removeAttribute('removed-to');
        }
        clip.removeAttribute('removed-to');
        result = redoPutIntoClipboard(contentCopy,clip);
    }

    return [undo,redo];
}
```

Merge elements

```
function mergeElements(sourceParent,targetParent,bigSibling) {
    var n = sourceParent.childElementCount, i;
    for (i=0;i<n;i++) {
        var el = sourceParent.firstElementChild;
        sourceParent.removeChild(el);
        if (bigSibling) {
            targetParent.insertBefore(el,bigSibling);
        } else {
            targetParent.appendChild(el);
        }
    }
    return n;
}
```

Undo merge elements

```
function undoMergeElements(sourceParent,targetParent,bigSibling,cec) {
    var i, el, prevEl;
    cl('source␣parent:'); cl(sourceParent);
    cl('target␣parent:'); cl(targetParent);
    cl('big␣sibling:'); cl(bigSibling);

    for (i=0;i<cec;i++) {
        if (bigSibling) {
            el = bigSibling.previousElementSibling;
        } else {
            el = targetParent.lastElementChild;
        }
        cl('el:'); cl(el);
        cl('cec:'+cec)
        targetParent.removeChild(el);
        if (i) {
            sourceParent.insertBefore(el,prevEl);
        } else {
            sourceParent.appendChild(el);
        }
        prevEl = el;
    }
}
```

Get from clipboard

```
function getFromClipboard() {
    var activeClip = doc.getElementById('active-clip');
    if (!activeClip) return null;
    clipboard.removeChild(activeClip);
    var content = activeClip.lastElementChild, ruler;
    activeClip.removeAttribute('id');
    ruler = activeClip.firstElementChild;
    ruler.className = 'clip-ruler';
    activeClip.removeChild(content);
    activeClip.setAttribute('removed-to','trash');
    return {content: content, clip: activeClip};
}
```

Undo get from clipboard

```
function undoGetFromClipboard(result) {
    var clip = result.clip,
        content = result.content,
        ruler = clip.firstElementChild;
    clip.removeAttribute('removed-to');
    clip.appendChild(content);
    ruler.className = 'active-clip-ruler';
    clip.id = 'active-clip';
    var f =clipboard.firstElementChild.nextElementSibling;
    if (f) {
        clipboard.insertBefore(clip,f);
    } else {
        clipboard.appendChild(clip);
    }
}
```

Remove empty cell

```
function removeEmptyCell(mevt,check) {

    if (check) {
        var grandpa, i, ecell;
        i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        ecell = mevt.elementPath[i];
        var f = ecell.firstElementChild;
        if (f!=null) cl('f␣class:'+f.className);
        if (f!=null && f.className=='p-editable' &&
            f.textContent.length==0) {
            ecell.normalize(); ecell.textContent = 'ECELL';
            return true;
        }
        if (ecell.childElementCount>0) return false;
        if (ecell.hasAttribute('in-list')) return false;
        var depth = ecell.getAttribute('depth');
        cl('depth:'+depth);
        if (ecell.getAttribute('depth')=='0') {
            grandpa = ecell.parentElement.parentElement;
            if (grandpa.childElementCount==2) return false;
        }
        return true;
    }

    var i = getIndexOfEE(mevt.elementPath,mevt.nest),
        ecell = mevt.elementPath[i];

    var box, rowParent, bigBrother, grandpa, bigUncle,
        sourceParent, targetParent, targetBigBrother,
        targetRowParent, ehbox, evbox, content, cec, rpcec,
        gcec, flag, upperBox;

    removeEmptyCell.figResizable = true;
    redo();

    function redo() {
        rowParent = ecell.parentElement;
        grandpa = rowParent.parentElement;
        box = grandpa.parentElement;

        redo.figResizable = true;


        if (box.className=='hbox') {
            bigBrother = ecell.nextElementSibling;
            rowParent.removeChild(ecell);
            ecell.setAttribute('removed-to','trash');
            rpcec = rowParent.childElementCount;
            flag = 10;
            if (rpcec==1) {
                ehbox = rowParent.firstElementChild;
                evbox = box.parentElement;

                upperBox =
                evbox.parentElement.parentElement.parentElement;
                redo.figTargetElement = upperBox;
                removeEmptyCell.figTargetElement = upperBox;

                content = ehbox.firstElementChild;
                if (!content || (content.className=='p-editable' &&
                    content.textContent.length==0)) {
                    flag = 11;
                    evbox.removeChild(box);
                    evbox.textContent = 'ECELL';
                    box.setAttribute('removed-to','trash');
                } else if (content.className!='vbox') {
                    flag = 12;
                    ehbox.removeChild(content);
                    ehbox.setAttribute('removed-to','trash');
                    evbox.removeChild(box);
                    box.setAttribute('removed-to','trash');
                    evbox.appendChild(content);
                } else {
                    flag = 13;
```

```
                    ehbox.removeChild(content);
                    ehbox.setAttribute('removed-to','trash');
                    sourceParent = content.firstElementChild;
                    targetParent = evbox.parentElement.parentElement;
                    targetBigBrother =
                    evbox.parentElement.nextElementSibling;
                    targetRowParent = evbox.parentElement;
                    targetParent.removeChild(targetRowParent);
                    targetRowParent.setAttribute('removed-to','trash');
                    box.setAttribute('removed-to','trash');
                    cec = mergeElements(sourceParent,targetParent,
                                        targetBigBrother);
                }
            }
        } else {
            bigUncle = rowParent.nextElementSibling;
            grandpa.removeChild(rowParent);
            rowParent.setAttribute('removed-to','trash');
            gcec = grandpa.childElementCount;
            flag = 20;
            if (gcec==1) {
                evbox = grandpa.firstElementChild.firstElementChild;
                ehbox = box.parentElement;

                upperBox = ehbox.parentElement.parentElement;
                redo.figTargetElement = upperBox;
                removeEmptyCell.figTargetElement = upperBox;

                content = evbox.firstElementChild;
                if (!content || (content.className=='p-editable' &&
                    content.textContent.length==0)) {
                    flag = 21;
                    evbox.removeChild(box);
                    evbox.textContent = 'ECELL';
                    box.setAttribute('removed-to','trash');
                } else if (content.className!='hbox') {
                    flag = 22;
                    evbox.removeChild(content);
                    evbox.setAttribute('removed-to','trash');
                    ehbox.removeChild(box);
                    box.setAttribute('removed-to','trash');
                    ehbox.appendChild(content);
                } else {
                    flag = 23;
                    evbox.removeChild(content);
                    evbox.setAttribute('removed-to','trash');
                    sourceParent =
                    content.firstElementChild.firstElementChild;
                    targetBigBrother = ehbox.nextElementSibling;
                    targetParent = ehbox.parentElement;
                    targetParent.removeChild(ehbox);
                    ehbox.setAttribute('removed-to','trash');
                    cec = mergeElements(sourceParent,targetParent,
                                        targetBigBrother);
                }
            }
        }

    }

    function undo() {
        if (box.className=='hbox') {
            undo.figResizable = true;
            undo.figTargetElement = upperBox;

            if (bigBrother) {
```

```
            rowParent.insertBefore(ecell,bigBrother);
        } else {
            rowParent.appendChild(ecell);
        }

        ecell.removeAttribute('removed-to');
        if (rpcec==1) {
            if (flag==11) {
                box.removeAttribute('removed-to');
                evbox.textContent = ''; evbox.normalize();
                evbox.appendChild(box);
            } else if (flag==12) {
                evbox.removeChild(content);
                box.removeAttribute('removed-to');
                evbox.appendChild(box);
                ehbox.removeAttribute('removed-to');
                ehbox.appendChild(content);
            } else {
                cl('HERE');
                undoMergeElements(sourceParent,targetParent,
                                   targetBigBrother,cec);
                box.removeAttribute('removed-to');
                targetRowParent.removeAttribute('removed-to');
                if (targetBigBrother) {
                    targetParent.insertBefore(targetRowParent,
                                               targetBigBrother);
                } else {
                    targetParent.appendChild(targetRowParent);
                }
                ehbox.removeAttribute('removed-to');
                ehbox.appendChild(content);
            }
        }
```

```
    } else {
        rowParent.removeAttribute('removed-to');
        if (bigUncle) {
            grandpa.insertBefore(rowParent,bigUncle);
        } else {
            grandpa.appendChild(rowParent);
        }
        if (gcec==1) {
            if (flag==21) {
                box.removeAttribute('removed-to');
                evbox.textContent = ''; evbox.normalize();
                evbox.appendChild(box);
            } else if (flag==22) {
                ehbox.removeChild(content);
                box.removeAttribute('removed-to');
                ehbox.appendChild(box);
                ehbox.removeAttribute('removed-to');
                evbox.appendChild(content);
            } else {
                undoMergeElements(sourceParent,targetParent,
                                  targetBigBrother,cec);
                ehbox.removeAttribute('removed-to');
                if (targetBigBrother) {
                    targetParent.insertBefore(ehbox,
                                              targetBigBrother);
                } else {
                    targetParent.appendChild(ehbox);
                }
                evbox.removeAttribute('removed-to');
                evbox.appendChild(content);
            }
        }
    }
    }

    return [undo,redo];

}
```

Paste empty cell from clipboard

```
function pasteEmptyCellFromClipboard(mevt,check) {
    var i, ecell;
    if (check) {
        var activeClip;
        if (itemWithinClipboard(mevt.elementPath)) return false;
        i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        ecell = mevt.elementPath[i];
        if (ecell.childElementCount!=0) return false;
        activeClip = doc.getElementById('active-clip');
        if (!activeClip) return false;
        return true;
    }
    i = getIndexOfEE(mevt.elementPath,mevt.nest);
    ecell = mevt.elementPath[i];
    var parentBox, boxClass,
        grandpa, bigUncle, clipGrandpa,
        targetParent, bigBrother, clipParent,
        content, cec, result;

    pasteEmptyCellFromClipboard.figResizable = true;

    redo();

    function redo() {
        parentBox = ecell.parentElement.parentElement.parentElement;

        redo.figResizable = true;
        redo.figTargetElement = parentBox;
        pasteEmptyCellFromClipboard.figTargetElement = parentBox;

        boxClass = parentBox.className;
        result = getFromClipboard();
        content = result.content;
        switch (content.className) {
            case 'vbox':
                if (boxClass=='vbox') {
                    bigUncle = ecell.parentElement.nextElementSibling;
                    grandpa = ecell.parentElement.parentElement;
                    grandpa.removeChild(ecell.parentElement);
                    clipGrandpa = content.firstElementChild;
                    cec = mergeElements(clipGrandpa,grandpa,bigUncle);
                } else {
                    ecell.textContent = ''; ecell.normalize();
                    ecell.appendChild(content);
                }
                break;
            case 'hbox':
                if (boxClass=='hbox') {
                    bigBrother = ecell.nextElementSibling;
                    targetParent = ecell.parentElement;
                    targetParent.removeChild(ecell);
                    clipParent =
                    content.firstElementChild.firstElementChild;
                    cec = mergeElements(clipParent,targetParent,
                                        bigBrother);
                } else {
                    ecell.textContent = ''; ecell.normalize();
                    ecell.appendChild(content);
                }
                break;
            default:
                    ecell.textContent = ''; ecell.normalize();
                    ecell.appendChild(content);
                break;
        }


    }

    function undo() {
        undo.figResizable = true;
        undo.figTargetElement = parentBox;

        switch (content.className) {
            case 'vbox':
                if (boxClass=='vbox') {
                    undoMergeElements(clipGrandpa,grandpa,bigUncle,cec);
                    if (bigUncle) {
                        grandpa.insertBefore(ecell.parentElement,
                                             bigUncle);
                    } else {
                        grandpa.appendChild(ecell.parentElement);
                    }
                } else {
                    ecell.removeChild(content);
                    ecell.textContent = 'ECELL';
                }
                break;
            case 'hbox':
                if (boxClass=='hbox') {
                    undoMergeElements(clipParent,parent,bigBrother,cec);
                    if (bigBrother) {
                        parent.insertBefore(row,bigBrother);
                    } else {
                        parent.appendChild(row);
                    }
                } else {
                    ecell.removeChild(content);
                    ecell.textContent = 'ECELL';
                }
                break;
            default:
                    ecell.removeChild(content);
                    ecell.textContent = 'ECELL';
                break;
        }
        undoGetFromClipboard(result);
    }


    return [undo,redo];
}
```

Switch active clip item

```
function switchActiveClipItem(mevt,check) {
    if (check) {
        var ci = itemWithinClipboard(mevt.elementPath);
        if (ci) return true;
        return false;
    }

    var ci = itemWithinClipboard(mevt.elementPath),
        ac, flag;

```

```
redo();

function redo() {
    if (ci.id=='active-clip') {
        flag = 1;
        ci.removeAttribute('id');
        ci.firstElementChild.className = 'clip-ruler';
    } else {
        ac = doc.getElementById('active-clip');
        if (ac) {
            flag = 2;
            ac.removeAttribute('id');
            ac.firstElementChild.className = 'clip-ruler';
        }
        ci.id = 'active-clip';
        ci.firstElementChild.className = 'active-clip-ruler';
    }
}

function undo() {
    if (flag==1) {
        ci.id = 'active-clip';
        ci.firstElementChild.className = 'active-clip-ruler';
    } else {
        if (flag==2) {
            ac.id = 'active-item';
            ac.firstElementChild.className = 'active-clip-ruler';
        }
        ci.removeAttribute('id');
        ci.firstElementChild.className = 'clip-ruler';
    }
}

```

```
    return [undo,redo];
}
```

Remove clip to trash

```
function removeClipToTrash(mevt,check) {
    if (check) {
        var ci = itemWithinClipboard(mevt.elementPath);
        if (ci) return true;
        return false;
    }

    var ci = itemWithinClipboard(mevt.elementPath), ns;

    redo();

    function redo() {
        ns = ci.nextElementSibling;
        clipboard.removeChild(ci);
        ci.setAttribute('remove-to','trash');
    }

    function undo() {
        ci.removeAttribute('remove-to');
        if (ns) {
            clipboard.insertBefore(ci,ns);
        } else {
            clipboard.appendChild(ci);
        }
    }

    return [undo,redo];
}
```

B.9 API for `Streams of Fragments`

1. Start new stream of code fragments:

```
function onCFSNameBlur(evt) {
    var name = evt.target.textContent,
        cfsHdr =
        evt.target.parentElement.parentElement.parentElement,
        sid = cfsHdr.getAttribute('counter'),
        cfsFrags = doc.getElementsByClassName('cfs-'+sid),
        i = cfsFrags.length, cfsFrag;

    name = name.replace(/\s/g,'␣');
    name = name.replace(/ [ ]+/g,'␣');
    name = name.trim();
    evt.target.textContent = name;

    while (i--) {
        cfsFrag = cfsFrags[i];
        cfsFrag.title = name+'␣('+cfsFrag.className.slice(4)+')';
    }
    if (evt.target.hasAttribute('data')) {
        evt.target.style.color = 'Salmon';
    }
    doc.cfsState.toFill = true;


    var cfsRefs = inCfsReference(evt.target,true),
        i = cfsRefs.length;
    if (i>0) {
        var tr = cfsHdr.firstElementChild.firstElementChild,
            ext = tr.firstElementChild.textContent,
            extName = ext+':␣'+name+'␣('+sid+')';
        while (i--) {
                cfsRefs[i].textContent = extName;
            }
        }
}

function onCFSNameFocus(evt) {
    evt.target.style.color = 'DarkBlue';
}

function simpleAssign(objTo,objFrom) {
    if (!objFrom) return objTo;
    for (x in objFrom) objTo[x] = objFrom[x];
    return objTo;
}

function startNewCFS(mevt,check) {
    if (check) return true;
    var ecell = mevt.element,
        cfsHdr = classInstance('cfs-hdr'),
        cfsName =
        cfsHdr.lastElementChild.firstElementChild.lastElementChild,
        prevState, sid, nsid, nestLine, nestedIn;

    redo();

    function undo() {
        ecell.removeChild(cfsHdr);
        cfsHdr.setAttribute('removed-to','trash');
        cfsName.removeEventListener('blur',onCFSNameBlur,true);
        cfsName.removeEventListener('focus',onCFSNameFocus,true);
        doc.cfsState = simpleAssign({},prevState);
        ecell.textContent = 'ECELL';
    }

    function redo() {
        if (cfsHdr.hasAttribute('remove-to')) {
            cfsHdr.removeAttribute('remove-to');
        }
        cfsName.addEventListener('blur',onCFSNameBlur,true);
        cfsName.addEventListener('focus',onCFSNameFocus,true);
        prevState =  simpleAssign({},doc.cfsState);

        doc.cfsState.toFill = true;
        doc.cfsState.id = cfsHdr.getAttribute('counter');
        ecell.textContent = ''; ecell.normalize();
        ecell.appendChild(cfsHdr);
        cfsName.focus();
    }

    return [undo,redo];
}
```

2. Continue stream of fragments:

```
function newCodeFragment(mevt,check) {
    if (check) {
        if (!doc.cfsState.id) return false;
        return true;
    }
    var ecell = mevt.element;

    var cfsFrag = classInstance('cfs-frag','cfs-'+doc.cfsState.id),
        codeLines = classInstance('code-lines');
    cfsFrag.firstElementChild.appendChild(codeLines);

    var cfsHdr = doc.getElementById('cfs-hdr-'+doc.cfsState.id),
        cfsName =
        cfsHdr.lastElementChild.firstElementChild.lastElementChild;

    redo();

    function redo() {
        if (cfsFrag.hasAttribute('removed-to')) {
            cfsFrag.removeAttribute('removed-to');
        } else {
            cfsFrag.title =
            cfsName.textContent+'␣('+cfsFrag.className.slice(4)+')';
        }
        cfsFrag.addEventListener('blur',onCFSFragBlur,true);
        cfsFrag.addEventListener('focus',onCFSFragFocus,true);

        ecell.textContent = ''; ecell.normalize();
        ecell.appendChild(cfsFrag);

        doc.cfsLineState.modified[doc.cfsState.id] = true;
        codeLines.lastElementChild.focus();
    }

    function undo() {
        cfsFrag.setAttribute('removed-to','trash');

        cfsFrag.removeEventListener('blur',onCFSFragBlur,true);
        cfsFrag.removeEventListener('focus',onCFSFragFocus,true);

        ecell.removeChild(cfsFrag);
        ecell.textContent = 'ECELL';
    }

    return [undo,redo];
}
```

3. Assign stream of code fragments for its nesting line or resuming its definition:

```
function getStreamIdFromFragment(el) {
    while(el.className!='du') {
        if (el.id.startsWith('cfs-frag-'))
            return el.className.slice(4); //'cfs-\d+'
        el = el.parentElement;
    }
    return null;
}
function numberOfDataStreams(cfsNames) {
    var i = cfsNames.length, n = 0;

    while(i--) {
        if (cfsNames[i].hasAttribute('data')) n += 1;
    }
    return n;
}

function assignCodeStream(mevt,check) {
    if (check) {
        var cfsNames =
            framecontent.getElementsByClassName('cfs-name');

        var nestLine =
            getElementOnPath(mevt.elementPath,'nest-line');
        if (nestLine)  {
            if (cfsNames.length<2) return false;
            return true;
        }

        var cfsRefData =
            getElementOnPath(mevt.elementPath,'cfs-ref-data'),
           nod = numberOfDataStreams(cfsNames),
                noc = cfsNames.length-nod;

        if (cfsRefData) {
            cl('NOD: '+nod);
            if (nod==0) return false;
            return true;
        }

        var cfsRefCode =
            getElementOnPath(mevt.elementPath,'cfs-ref-code');

        if (cfsRefCode) {
            var framecontentx =
                dc2Globals.framecontents['Description'],
                cfsNames =
                framecontentx.getElementsByClassName('cfs-name'),
                nod = numberOfDataStreams(cfsNames),
                noc = cfsNames.length-nod;
            //cl('NOC:'+noc);
            if (noc==0) return false;
            return true;
        }

        if (noc==0) return false;

        return true;
    }

    var nestLine = getElementOnPath(mevt.elementPath,'nest-line');
    if (nestLine) {
        cl('nest');
        return linkNestedStream(mevt,nestLine);
    }
```

```

    var cfsRefCode =
        getElementOnPath(mevt.elementPath,'cfs-ref-code');
    if (cfsRefCode) {
        cl('code');
        return linkReferencedStream(mevt,cfsRefCode);
    }

    var cfsRefData =
        getElementOnPath(mevt.elementPath,'cfs-ref-data');
    if (cfsRefData) {
        cl('data');
        return linkReferencedStream(mevt,cfsRefData);
    }
    cl('switch');
    return switchCodeStream(mevt);

}
function linkNestedStream(mevt,nestLine) {
    var nsid = nestLine.getAttribute('nest-sid');
    doc.cfsState.nl = nestLine;
    cl('state␣in␣nest:␣'+doc.cfsState)

    fillCFSSelector(doc.cfsState.sel);
    doc.cfsState.toFill = true;


    doc.cfsState.div.style.left = mevt.x+mevt.ox+10;
    doc.cfsState.div.style.top = mevt.y+mevt.oy+10;

    doc.cfsState.div.style.display = 'block';
    mState.on = true; mState.el = doc.cfsState.div;

    var newNsid = null;

    function undo() {
        newNsid = nestLine.getAttribute('nest-sid');
        if (nsid) {
            nestLine.setAttribute('nest-sid',nsid);
        } else {
            nestLine.removeAttribute('nest-sid');
            nestLine.firstElementChild.textContent = '??';
            nestLine.lastElementChild.style.color = 'orange';
        }
    }

    function redo() {
        if (newNsid) {
            nestLine.setAttribute('nest-sid',newNsid);
            nestLine.firstElementChild.innerHTML =
                                            specialSymbols.link;
            nestLine.lastElementChild.style.color = 'green';
            nestLine.title = 'NEST('+newNsid+')';
        } else {
            nestLine.removeAttribute('nest-sid');
            nestLine.firstElementChild.textContent = '??';
            nestLine.lastElementChild.style.color = 'orange';
        }
    }

    return [undo,redo];

}

function linkReferencedStream(mevt,cfsRefX) {
    var rsid = cfsRefX.getAttribute('ref-sid'),
        newRsid = null;

    doc.cfsState.ref = cfsRefX;
    cl('state␣in␣ref:␣'+doc.cfsState)

    fillCFSSelector(doc.cfsState.sel);
    doc.cfsState.toFill = true;

    doc.cfsState.div.style.left = mevt.x+mevt.ox+10;
    doc.cfsState.div.style.top = mevt.y+mevt.oy+10;

    doc.cfsState.div.style.display = 'block';
    mState.on = true; mState.el = doc.cfsState.div;

    function undo() {
        newRsid = cfsRefX.getAttribute('ref-sid');
        if (rsid) {
            cfsRefX.setAttribute('ref-sid',rsid);
        } else {
            cfsRefX.removeAttribute('ref-sid');
            if (cfsRefX.className=='cfs-ref-code') {
                cfsRefX.textContent =
                        'Assign␣code␣stream␣from␣menu';
            } else {
                cfsRefX.textContent =
                        'Assign␣data␣stream␣from␣menu';
            }
            cfsRefX.style.color = '#d4d4d4';
        }
    }

    function redo() {
        if (newRsid) {
            cfsRefX.setAttribute('ref-sid',newRsid);
```

```
                cfsRefX.style.color = 'green';
                cfsRefX.title = 'REFERENCE('+newRsid+')';
            } else {
                cfsRefX.removeAttribute('ref-sid');
                if (cfsRefX.className=='cfs-ref-code') {
                    cfsRefX.textContent = 'Assign␣code␣stream␣from␣menu';
                } else {
                    cfsRefX.textContent = 'Assign␣data␣stream␣from␣menu';
                }
                cfsRefX.style.color = '#d4d4d4';
            }
        }

        return [undo,redo];
}
function resumeCodeStream(mevt,check) {
        if (check) return true;
        switchCodeStream(mevt);
}

function switchCodeStream(mevt) {

        var prevStateId =  doc.cfsState.id, newStateId;

        if (doc.cfsState.toFill) {
            cl('state␣in␣resume:␣'+doc.cfsState)
            fillCFSSelector(doc.cfsState.sel);
            doc.cfsState.toFill = false;
        } else {
            changeCFSDefault(doc.cfsState.sel,0);
        }

        doc.cfsState.div.style.left = mevt.x+mevt.ox+10;
```

```
    doc.cfsState.div.style.top = mevt.y+mevt.oy+10;
    doc.cfsState.div.style.display = 'block';
    mState.on = true; mState.el = doc.cfsState.div;

    function undo() {
        newStateId = doc.cfsState.id;
        doc.cfsState.id = prevStateId;
    }

    function redo() {
        doc.cfsState.id = newStateId;
    }

    return [undo,redo];
}
```

4. Integrate current stream:

```
var h2tHelper = doc.createElement('P');

function getCfsContent(docx,sid,clang,withLinks) {

    function content(node) {
        node.normalize();
        h2tHelper.innerHTML =
        node.innerHTML.replace(/<br>/g,'X~~X<br>');
        var txt = h2tHelper.textContent;
        txt = txt.replace(/X~~X/g,'\n');
        var clist = txt.split('\n');
        if (clist[clist.length-1].length==0) clist.pop();
        return clist;
    }

    function commentCfsName(name) {
        var langPattern = langComments[clang];
        if (langPattern) {
            return langPattern.replace('XXX',name);
        } else {
            return '?<'+name+'>?';
        }
    }

    var frags = docx.getElementsByClassName('cfs-'+sid),
        n = frags.length, frag, fid, tr, lgroups = [], gcontent,
        nestNameDecor, lgroupIds = [], lineId;

    for (var i=0; i<n; i++) {
        frag = frags[i]; //fid = frag.id.slice(9);
        tr = frag.firstElementChild.firstElementChild;
        while (tr) {
            lineId =  tr.getAttribute('counter');
            if (tr.className=='nest-line') {
                if (tr.hasAttribute('nest-sid')) {
                    var nsid = tr.getAttribute('nest-sid');
                    var cfsHdr =
                    docx.getElementById('cfs-hdr-'+nsid);
                    if (cfsHdr) {
                        gcontent = getCfsContent(docx,nsid,
                                                 clang,withLinks);
                        if (withLinks) {
                            lgroups.extend(gcontent[0]);
                            lgroupIds.extend(gcontent[1]);
                        } else {
                            lgroups.extend(gcontent);
                        }
                    } else {
                        tr.removeAttribute('nest-sid');
                        nestNameDecor = commentCfsName(
                                            content(tr.lastElementChild));
                        lgroups.push(nestNameDecor);
                        if (withLinks) lgroupIds.push(lineId+'-0');
                    }
                } else {
                    nestNameDecor = commentCfsName(
                                        content(tr.lastElementChild));
                    lgroups.push(nestNameDecor);
                    if (withLinks) lgroupIds.push(lineId+'-0');
                }
            } else {
                var lines = content(tr.lastElementChild);
                lgroups.extend(lines);
                if (withLinks) {
                    for (var j=0;j<lines.length;j++) {
                        lgroupIds.push(lineId+'-'+j);
                    }
                }
            }
            tr = tr.nextElementSibling;
        }
    }

    if (withLinks) {
        return [lgroups,lgroupIds];
    } else {
        return lgroups;
    }
}
function getCFSId(mevt) {
    var el = mevt.element;
    while (el && el.className!='du') {
        if (el.id.startsWith('cfs-frag-')) {
            updateFragLinesId(el,1);
            return el.className.slice(4);
        }
        el = el.parentElement;
    }
    return null;
}

function inCfsReference(el,all) {
    function getSidForNameElement(el) {
        var hdr = el.parentElement.parentElement.parentElement;
        return hdr.id.slice(8);
    }
    var sid = getSidForNameElement(el),
        framecontentx = framecontent,
        isData = el.hasAttribute('data'), cfsRefs;
    if (isData) {
        cfsRefs =
        framecontentx.getElementsByClassName('cfs-ref-data');
    } else {
        var frameNames = ['Description','Configuration','Comments'],
            i = 3;
        cfsRefs = [];
        while(i--) {
            framecontentx = dc2Globals.framecontents[frameNames[i]];
            cfsRefs.extend(framecontentx.getElementsByClassName(
                                'cfs-ref-code'));
        }
    }
    var i = cfsRefs.length;

    if (all) {
        var li = [];
```

```
          while(i--) {
              if (cfsRefs[i].getAttribute('ref-sid')==sid)
                  li.push(cfsRefs[i]);
          }
          return li;
      } else {
          while(i--) {
              if (cfsRefs[i].getAttribute('ref-sid')==sid)
                  return true;
          }
          return false;
      }
}
function codeSwitchWithData(mevt,check) {
      if (check) {
          var el = mevt.element;
          if (el.className!='cfs-name') return false;
          if (inCfsReference(el,false)) return false;
          return true;
      }
      var el = mevt.element;
      if (el.hasAttribute('data')) {
          el.removeAttribute('data');
          el.style.color = 'DarkBlue';
      } else {
          el.setAttribute('data','true');
          el.style.color = 'Salmon';
      }
      return null;
}
function integrateCurrentStream(mevt,check) {
      if (check) {
          var el = mevt.element;
          if (el.className=='cfs-name') return true;
          if (el.className=='td-editable' && el.hasAttribute('code'))
              return true;
          return false;
      }
      var el = mevt.element, sid, cfsHdr, codeTr, cfsContent,
          cfsRange, html, rangeHtml, n, clang, nameTr;

      if (el.className=='cfs-name') {
          cfsHdr = el.parentElement.parentElement.parentElement;
          sid = cfsHdr.getAttribute('counter');
          cl('sid␣from␣cfs-name:␣'+sid);
      } else if (el.className=='td-editable' &&
                 el.hasAttribute('code')) {
          sid = getCFSId(mevt);
          cfsHdr = doc.getElementById('cfs-hdr-'+sid);
          cl('sid␣from␣cfs-lines:␣'+sid);
      }

      codeTr = cfsHdr.firstElementChild.lastElementChild;
      nameTr = cfsHdr.firstElementChild.firstElementChild;
      clang = nameTr.firstElementChild.textContent;
      cl('lang:␣'+clang);

      cfsContent = getCfsContent(doc,sid,clang,false);

      html = cfsContent.join('<br>');
      codeTr.lastElementChild.innerHTML = html;

      // Array.from(lgroups.keys());
      n = html.split('<br>').length; cfsRange = Array(n);
      for (var i=0; i<n; i++) cfsRange[i] = i+1;
      rangeHtml = cfsRange.join('<br>');
      codeTr.firstElementChild.innerHTML = rangeHtml;

      codeTr.style.display = '';


      function redo() {
          codeTr.firstElementChild.innerHTML = rangeHtml;
          codeTr.lastElementChild.innerHTML = html;
      }

      function undo() {
          codeTr.firstElementChild.innerHTML = '';
          codeTr.lastElementChild.innerHTML =
                          'Alt+IC␣-␣for␣stream␣integration';
      }

      return [undo,redo];
  }
```

5. Code fragments stream – selection:

```
function cfsCmp(eni1,eni2) {
    if (eni1.ext==eni2.ext) {
        if (eni1.name<eni2.name) {
            return -1;
        } else if (eni1.name>eni2.name) {
          return +1;
        } else {
            return (eni1.id-eni2.id);
        }
    } else if (eni1.ext<eni2.ext) {
        return -1;
    } else {
        return +1;
    }

}
function changeCFSDefault(selector,selId) {
    var selIndx = selector.selectedIndex;
    if (selIndx<0) return;
    cl('selected␣index:'+selIndx);
    if (selIndx==selId) return;

    var options = selector.options;
    options[selId].selected = true;
    options[selIndx].selected = false;
}

function isNested(hdr,nls) {
    var sid = hdr.id.slice(8), i = nls.length;
    while (i--) {
        if (nls[i].getAttribute('nest-sid')==sid) return true;
    }
    return false;
}

function fillCFSSelector(selector) {
    var mainOnly = false, isData = false, isCode = false,
        framecontentx = framecontent,
        nlHdr = null;
    if (doc.cfsState.ref) {
        mainOnly = true;
        if (doc.cfsState.ref.className=='cfs-ref-data') {
            isData = true;
        } else {
            isCode = true;
            framecontentx = dc2Globals.framecontents['Description'];
        }
    } else if (doc.cfsState.nl) {
        var frag = doc.cfsState.nl.parentElement.parentElement,
            sid = frag.className.slice(4),cfsName;
        nlHdr = doc.getElementById('cfs-hdr-'+sid);
        cfsName =
        nlHdr.firstElementChild.firstElementChild.lastElementChild;
        if (cfsName.hasAttribute('data')) {
            isData = true;
        } else {
            isCode = true;
        }
    }
    cl('main␣only?: '+mainOnly);
    cl('data?: '+isData);

    var cfsHdrs = framecontentx.getElementsByClassName('cfs-hdr'),
        nls =  framecontentx.getElementsByClassName('nest-line'),
        i = cfsHdrs.length, cfsENI = [];
    while (i--) {
        var hdr = cfsHdrs[i],
            tr = hdr.firstElementChild.firstElementChild,
            ext = tr.firstElementChild.textContent;
        if (mainOnly && isNested(hdr,nls)) continue;
        var cfsName =
        hdr.firstElementChild.firstElementChild.lastElementChild,
            hasData = cfsName.hasAttribute('data');
        if (isData && !hasData) continue;
        if (isCode && hasData) continue;

        if (nlHdr && nlHdr.id==hdr.id) continue;

        cfsENI.push({ext: ext, name: tr.lastElementChild.textContent,
                    id: hdr.id.slice(8)});
    }

    var n = cfsENI.length;
    if (n==0) return;

    if (n>1) cfsENI.sort(cfsCmp);


    var option = selector.firstElementChild.nextElementSibling,
        id, name, ext;
    cl('first␣option:␣'+selector.firstElementChild.textContent);
    for (var i=0;i<n;++i) {
        id = cfsENI[i].id;
        name = cfsENI[i].name;
        ext = cfsENI[i].ext;
        if (!option) {
            option = doc.createElement('OPTION');
            selector.appendChild(option);
        }
        //option.textContent = ''; option.normalize();
        option.textContent = ext+':␣'+name+'␣('+id+')';
        option.value = id;
        option.style.display = '';
        if (option.selected) option.selected = false;

        option = option.nextElementSibling;
    }
    while (option) {
        option.style.display = 'none';
        if (option.selected) option.selected = false;
```

```
            option = option.nextElementSibling;
      }
}

var kspaces = '␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣';

function fillCFSLineSelector(docx,selector,sid) {

      var cfsHdr = docx.getElementById('cfs-hdr-'+sid),
            nameTr = cfsHdr.firstElementChild.firstElementChild,
            clang = nameTr.firstElementChild.textContent,
            linesLidlers = getCfsContent(docx,sid,clang,true),
            lines = linesLidlers[0], lidlers = linesLidlers[1],
            n = lines.length, k = Math.floor(1+Math.log2(n)), i,
            option = selector.firstElementChild.nextElementSibling,
            id, name, ext;

      for (i=0;i<n;++i) {
            if (!option) {
                  option = doc.createElement('OPTION');
                  selector.appendChild(option);
            }
            var spaces =
                  kspaces.slice(0,k-Math.floor(1+Math.log2(i+1)));
            option.textContent = spaces+(i+1)+':␣'+lines[i];
            option.value = lidlers[i];
            option.style.display = '';
            if (option.selected) option.selected = false;

            option = option.nextElementSibling;
      }
      while (option) {
            option.style.display = 'none';
            if (option.selected) option.selected = false;
```

```
        option = option.nextElementSibling;
    }
}
```

6. Define code stream reference:

```
function setCodeReference(mevt,check) {
    if (check) return true;

    var ecell = mevt.element,
      cfsRef = classInstance('cfs-ref');

    redo();

    function redo() {
        if (cfsRef.hasAttribute('removed-to')) {
            cfsRef.removeAttribute('removed-to');
        }
        ecell.textContent = ''; ecell.normalize();
        ecell.appendChild(cfsRef);
    }

    function undo() {
        cfsRef.setAttribute('removed-to','trash');
        ecell.removeChild(cfsRef);
        ecell.textContent = 'ECELL';
    }

    return [undo,redo];
}
```

7. Nest stream of fragments:

```
function onNestLineBlur(evt) {
    var name = evt.target.textContent,
        tr = evt.target.parentElement;
    name = name.replace(/\s/g,'␣');
    name = name.replace(/ [ ]+/g,'␣');
    name = name.trim();
    evt.target.textContent = name;

    if (tr.hasAttribute('nest-sid')) {
        evt.target.style.color = 'green';
    } else {
        evt.target.style.color = 'magenta';
    }
}

function onNestLineFocus(evt) {
    evt.target.style.color = 'DarkBlue';
}

function nestCodeStream(evt,check) {
    if (check) {
        if (evt.target.className=='td-editable') {
            if (evt.target.hasAttribute('code')) return true;
        }
        return false;
    }
    var tr = evt.target.parentElement,
        tbody = tr.parentElement,
        trNew = classInstance('nest-line');

    redo();

    function redo() {
        if (trNew.hasAttribute('removed-to')) {
            trNew.removeAttribute('removed-to');
        }
        trNew.addEventListener('blur',onNestLineBlur,true);
        trNew.addEventListener('focus',onNestLineFocus,true);
        tbody.insertBefore(trNew,tr);
        trNew.title = 'NO␣LINK';

        trNew.lastElementChild.focus();
    }

    function undo() {
        tbody.removeChild(trNew);
        trNew.setAttribute('removed-to','trash');
        trNew.removeEventListener('blur',onNestLineBlur,true);
        trNew.removeEventListener('focus',onNestLineFocus,true);
        if (tr.className=='code-lines')
            tr.lastElementChild.focus();
    }

    return [undo,redo];
}
```

8. Add code lines above:

```
function addCodeLinesAbove(evt,check) {
    cl('AA');
    return addCodeLines(evt,check,'above');
}
```

9. Add code lines below:

```
function addCodeLinesBelow(evt,check) {
    return addCodeLines(evt,check,'below');
}

function textCentered(evt,check) {
    return textAligned(evt,check,'center');
}

function textLeftAligned(evt,check) {
    return textAligned(evt,check,'left');
}


function textRightAligned(evt,check) {
    return textAligned(evt,check,'right');
}

function textAligned(evt,check,where) {
    if (check) {
        if (evt.target.className=='p-editable') {
            return true;
        }
        return false;
    }
    evt.target.style['text-align'] = where;
    return null;
}

function textBigger(evt,check) {
    return textSizeChange(evt,check,'+');
}

function textSmaller(evt,check) {
    return textSizeChange(evt,check,'-');
}

function textSizeChange(evt,check,trend) {
    if (check) {
        if (evt.target.className=='p-editable') {
            return true;
        }
        return false;
    }
    var p = evt.target, fontSize = p.style['font-size'], val = 105;
    if (trend=='-') val = 95;

    if (fontSize) {
        var fs = Number(fontSize.slice(0,-1));
        p.style['font-size'] = Math.round(fs*val/100)+'%';
    } else {
        p.style['font-size'] = val+'%';
    }
    return null;
}

function onCFSFragBlur(evt) {
    var frag = evt.currentTarget,
        sid = frag.className.slice(4);
    updateFragLinesId(frag,1);
    displayFragLinesId(frag,'visible');
    doc.cfsLineState.modified[sid] = true;
}

function onCFSFragFocus(evt) {
    var frag = evt.currentTarget;
    displayFragLinesId(frag,'hidden');
}


function numberOfLines(tr) {
    var td = tr.lastElementChild,
        lines = td.innerHTML.split('<br>'), len = lines.length;
        if (lines[len-1].length==0) len -= 1;
        return len;
}

function updateFragLinesId(frag,startId) {
    var tr = frag.firstElementChild.firstElementChild,
        id = ''+startId, td;
    while (tr) {
        if (tr.className=='code-lines') {
            td = tr.firstElementChild;
            td.textContent = id;
            id = id-0+numberOfLines(tr);
        }
        tr = tr.nextElementSibling;
    }
    return id;
}

function displayFragLinesId(frag,mode) {
    var tr = frag.firstElementChild.firstElementChild, td;
    while (tr) {
        td = tr.firstElementChild;
        if (tr.className=='code-lines') td.style.visibility = mode;
        tr = tr.nextElementSibling;
    }
}

function addCodeLines(evt,check,where) {
    if (check) {
        if (evt.target.className=='td-editable') {
            if (evt.target.hasAttribute('code')) return true;
        }
        return false;
    }
    var tr = evt.target.parentElement,
        tbody = tr.parentElement,
        trNew = classInstance('code-lines'),
        nextTr = tr.nextElementSibling;

    redo();

    function redo() {
        if (trNew.hasAttribute('removed-to')) {
            trNew.removeAttribute('removed-to');
        }
        if (where=='above') {
            tbody.insertBefore(trNew,tr);
        } else {
            if (nextTr) {
                tbody.insertBefore(trNew,nextTr);
            } else {
                tbody.appendChild(trNew);
            }
        }
        trNew.lastElementChild.focus();
    }

    function undo() {
        tbody.removeChild(trNew);
        trNew.setAttribute('removed-to','trash');
        tr.lastElementChild.focus();
    }

    return [undo,redo];
}
```

B.10 API for ECELL

Init paragraph

```
function initParagraph(mevt,check) {
    if (check) return true;
    var ecell = mevt.element,
          pel = doc.createElement('P');

    pel.textContent = 'Edit␣your␣text,␣please!';
    pel.setAttribute('contenteditable','true');
    pel.className = 'p-editable';
    pel.setAttribute('lang','en');

    ecell.textContent = ''; ecell.normalize();
    ecell.appendChild(pel);

    function undo() {
        ecell.removeChild(pel);
        pel.setAttribute('removed-to','trash');
        ecell.textContent = 'ECELL';
    }

    function redo() {
        pel.removeAttribute('removed-to');
        ecell.textContent = ''; ecell.normalize();
        ecell.appendChild(pel);
    }

    return [undo,redo];
}
```

Set list box

```
function setListBox(mevt,check) {
    if (check) return true;
    var ecell = mevt.element;

    var lboxTemplate = doc.getElementById('lbox-template'),
          lbox = lboxTemplate.cloneNode(true);
    lbox.removeAttribute('id');

    ecell.textContent = ''; ecell.normalize();
    ecell.appendChild(lbox);

    function undo() {
        ecell.removeChild(lbox);
        lbox.setAttribute('removed-to','trash');
        ecell.textContent = 'ECELL';
    }

    function redo() {
        lbox.removeAttribute('removed-to');
        ecell.textContent = ''; ecell.normalize();
        ecell.appendChild(lbox);
    }

    return [undo,redo];

}
```

Set math cell

```
function setMathCell(mevt,check) {
}
```

Set live media cell

```
function setLiveMediaCell(mevt,check) {
    if (check) {
        if (doc.id!='Description') return false;
        if (!navigator.mediaDevices) {
            alert('mediaDevices()␣not␣supported.');
            return false;
        }
        return true;
    }

    var ecell = mevt.element,
        constraints = {
            audio: true,
            video: { width: { min: 600, max: 700 },
                     height: { min: 400, max: 500 },
                     require: ["width", "height"]
                   },
        },
        v = doc.createElement('VIDEO');

    v.setAttribute('autoplay','');
    v.setAttribute('controls','');

    navigator.mediaDevices.getUserMedia(constraints)
        .then(stream => v.mozSrcObject = stream)
        .then(() => new Promise(
                    resolve => v.onloadedmetadata = resolve))
        .then(() => success())
        .catch(failed);

    var failed =
    e => alert(e.name +":␣"+ e.message +"␣:␣"+ e.lineNumber);
    var success = function() {
            v.style.display = 'block';
            v.style.width = '100%';
            v.style.margin = 'auto';
            v.className = 'live-video';
            ecell.textContent = ''; ecell.normalize();
            ecell.appendChild(v);
            liveMediaList.push([v,ecell]);
            cl('Live video: '+v.videoWidth+' x '+v.videoHeight);
        }
    return null;
}
```

Set image cell

```

function setImageCell(mevt,check) {
    if (check) {
        var i,ecell;
        i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        ecell = mevt.elementPath[i];
        if (ecell.childElementCount>0) return false;
        if (!mediaboards.image.targetFigure) return false;
        return true;
    }
    return setMediaCell(mevt,mediaboards.image);
}
```

Set sound cell

```
function setSoundCell(mevt,check) {
    if (check) {
        var i,ecell;
        i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        ecell = mevt.elementPath[i];
        if (ecell.childElementCount>0) return false;
        if (!mediaboards.sound.targetFigure) return false;
        return true;
    }
    return setMediaCell(mevt,mediaboards.sound);
}
```

Set movie cell

```
function setMovieCell(mevt,check) {
    if (check) {
        var i,ecell;
        i = getIndexOfEE(mevt.elementPath,mevt.nest);
        if (i<0) return false;
        ecell = mevt.elementPath[i];
        if (ecell.childElementCount>0) return false;
        if (!mediaboards.movie.targetFigure) return false;
        return true;
    }
    return setMediaCell(mevt,mediaboards.movie);
}
```

Set media cell

```
function setMediaCell(mevt,mediaboard) {
    var i = getIndexOfEE(mevt.elementPath,mevt.nest),
        ecell = mevt.elementPath[i],
        fig = mediaboard.targetFigure;

    redo();

    function redo() {
        mediaboard.transfer(fig,'s2d',ecell);
        mediaboard.targetFigure = null;
        fig.removeAttribute('active');
    }

    function undo() {
        mediaboard.transfer(fig,'d2s');
        mediaboard.targetFigure = fig;
        fig.setAttribute('active','true');
    }

    return [undo,redo];
}
```

Move to media board

```
var mediaTag2Type = {img:'image',audio:'sound',video:'movie'};
function moveToMediaboard(mevt,check) {
    if (check) {
        var target = mevt.element;
        if (target.tagName=='IMG' || target.tagName=='AUDIO' ||
            target.tagName=='VIDEO') {
            var fig =  target.parentElement;
            if (fig.tagName.toUpperCase()!='FIGURE') return false;
            var ecell = fig.parentElement;
            if (ecell.className!='ecell') return false;
            return true;
        }
        return false;
    }

    var media = mevt.element,
        tag = media.tagName.toLowerCase(),
        mediaboard = mediaboards[mediaTag2Type[tag]],
        fig = media.parentElement,
        ecell = fig.parentElement;

    redo();

    function redo() {
        mediaboard.transfer(fig,'d2s');
    }

    function undo() {
        mediaboard.transfer(fig,'s2d',ecell);
    }

    return [undo,redo];
}
```

Adjust size of figures

```
function adjustSizeOfFigures(targetElement) {
    var figs = targetElement.getElementsByTagName('FIGURE'),
        i = figs.length;
    //cl('figures in the box? '+i);
    while (i--) {
        var fig = figs[i], ecell = fig.parentElement;
        fig.style['max-width'] = ''+ecell.clientWidth+'px';
    }
}
```