Introduction to the the box

A introduction tutorial to the the box syntax:

The box uses C was it’s programming language, to access the low level capabilitys of the language. The box is a software engine, which aims to develop for all parts of the project.

- How the syntax looks like?

works like Html CSS, but is made with File objects.

All variables are global, that's how it's done. The color_ is a reference to the object color.h, which is done manually is not called in code.

- How it's done?

Files inherit from another, a color.h should be used in files that need color. A file that does not need it should be in higher hierquy. This is the logic for the software. It helps to know when something is going to share or not.

Like a applications actions, several tools will save actions, so actions.h is in higher hierquy. Some one may ask why not use only pre processor. Because if we go in to MVC model view contoler we have several Hierarchy. “this was the point i started to adopt this style of code” we are going to need Hierarchys in several dimensions. Its a logic of programation.

-Hierarchy examle 2

path will not use color is above, interface will use is bellow. If you know this logic when you are programming you already know when to move things.

path.h
color.h

interface.h

-Hierarchy examle 2

Let’s say. You decided to move application actions from your tool, like a code editor : save, open file, etc... If you are adopting this style is just a matter to see where it fits in the Hierarchy, if you have a text editor and a code editor. Two tools that will use actions.h. We add action, to includes and is working. You can tweek large amounts of code width this logic. It all ways works.

```
path.h
color.h

interface.h

application actions.h

----

tool code editor

tool text editor
```

-Hierarchy explain

In *a way is a large MVC so I call it structural* ***programming.*** Every thing is in a structure. Questions:

- forum user ask : people will need to study?

Ya, you just need to study a bit the files, but once you understand it is very easy to program. You can write large extension of code, width out the need to test it. It always work. The stuff that work in a small MVC, will also work in a large MVC. Because both are Hierarchy's.

File objects

A example of a file object, they are pre processor allways in .h extension. So all files will see them. And .c files can work on them.

```
include “color.h”

color_text = "";
color_background = "";
```

Values are persistent until new is added.

```
(include “color.h”)
color_text = "black";
color_background = "white";
button1();
button2();
button3();
color_text = "green";
```

```
button4();
```

All buttons will have text black and background white. Button 4 will have white background and green text. Only one property is updated.

It's the same was CSS. Values are persistent until a new value is add that override part of the values.

the color.h will simple have global variables.

```
color_text;

color_background;
```

To change font. font.h will have global variables on that.

```
font_weight;

font_italic;

font_style;
```

A example with the font

```
color.h

font.h

color_text = "black";

color_background = ""white;

button1();

button2();

button3();

color_text = "green";

font_weight = "bold";

font_style = "arial";


button4();
```

A costume button 4 with a font arial, bold weight, and color text green and a inherit background color white;

The same can be applied to functions. At the moment i don't have a real example will make one for the tutorial. “Costume_function”.

```
color.h

font.h

interface.h

costume_function.h

color_text = "black";
```

```
color_background = ""white;
button_costumized(); addeds : border, solid, 1 px;
"
property_border_style: solid;
property_borde_width: 1px;
"
button1();
button2();
button3();
color_text = "green";
font_weight = "bold";
font_style = "arial";
button4();
```

All 4 buttons will have a customized border. First 3 will have: a text color black and background white. 4º button will have a customized style was before. text color green, font weight bold and a different font arial.

No variables are passed to functions at any time, only when needed or variables repeated because, they are inherited, only one time will a variable be declared.

A FILE object is declared only one time in file.h

```
file.h
FILE * FILE_pointer;
```

A example with the file pointer.

```
color.h
font.h
interface.h
costume_function.h
file.h

open_file(); // Get styles

"
receives FILE_Pointer; From file.h
```

```
"

color_text = "black";
color_background = ""white;

button_costumized(); addeds : border, solid, 1 px;
"
property_border_style: solid;
property_borde_width: 1px;
"

button1();
button2();
button3();

color_text = "green";
font_weight = "bold";
font_style = "arial";

button4();

open_file(); // Get styles for a menu

"
receives FILE_Pointer; From file.h
"
```

The style is the same was previous example, now the open*file() is using the same file* pointer which is a object variable, if needed to store values, which are a less thing since most time we will be doing something and outputting the values. Are stored in arrays, structures, or files, with the values and passed to the functions, with the same thing global variables.

The variables with the stored information have plural names, to be different from objects, which are single abstraction. For example “char menus”; will store information that is needed to call back. Maybe a menu position, to know which are where. But since many information is in files, object abstraction is probably enough.

Store information and load dynamic information

```
(tool.h)

interface_index_keys = {"interface_id", "menu_position", “text_color”,
“background_color”, “”};

“information_menu.txt”

20, 1, black, white;
```

The information will be in txt, files, it will make it really dynamic, can chance any thing in the software. The format choose a simple keys to load and to map, if you see it have the same name of variables. and files have limited information to speed the search process.

The keys map the file, and will also work with simple idex, which are separated files with only the id's to speed search of information in files. Like a database work but only with txt files.

With this format we can load a hole project from a txt file, and probably will not consume many resources.