**Children’s Position Handling**

Lets say we have this tree:

Row > Text, Image

In this case if we to change the position of Row then this would be a sequence of code

Row.SetRelPos > Row.UpdateAbsTf > Row.**HandleChildPosition**

**HandleChildPosition** is a virtual method, handling a positioning of children relative position

**Relative Positioning**

Relative positioning work in this way:

There are 2 SDL\_Rect relative and an absolute. Absolute used only for rendering and for getting information. While Relative is mostly accessible. Relative is the transform relative from parents’ position

So a child with 12:13 and parent with 15:10. Now becomes a child with 27:23

**Padding**

In current layout, rows and columns have their own padding.

Work on the assumptions that every complex component (Such as: Column, Row, Grid) have an internal margin. Otherwise, they would

**Double Padding**

There could be a situation where a row inside a column would create a double padding spacing

The solution to that is to have padding, that goes in between objects

And margin that define an outer offset from its content

**Parents Destructor**

When a parent destructs, it deletes all the children with him and let his parent know that he is abdicating. It can still can cause pointer problems so it might be good to look at unique pointer