**App Config**

1)Go to /meta/app\_config.py and open it with any text editor.

2)Change the value of parameters to whatever you want keeping the original formatting (quotes or capitalization of the first letter).

3)Save the file.

Note: Currently, most changes will not (backend) be processed by the app or will be processed incorrectly (frontend). In the future version, config change will be done via GUI.

**Rules Set (Match Config)**

1)Go to /py/match/match\_config.py and open it with any text editor.

2)Change the value of parameters to whatever you want keeping the original formatting (quotes, capitalization of the first letter, correct brackets, and commas). (There are comments)

3)Save the file.

Note: Backend seems to process rules correctly but frontend might be weird for non-standard values. In the future version, rules change will be done via GUI, as well as there will be an ability to save, load, delete etc rules sets.

**Localization**

1)Go to /meta/localiztion.py and open it with any text editor.

2)Different windows / popups are represented by different group which have formatting of

group\_name = [

[sub\_group\_zero],

[sub\_group\_one],

…

]

3)In order to insert your language into the app, you need to create subgroups for it in every group (Index 0 is reserved for Ukrainian, index 2 is reserved for English), however you can use and change already existing indexes.

4)Fill the subgroups with the translation. Remember to put text inside of quotation marks, as well as in-line formatting (in different languages string variable value might need to be put before / between / after the constant text, so some phrases have two variables to represent that.

5)Remember to put commas after every non-last element in a subgroup, as well as after every non-last subgroup in a group.

6)Save the file.

7)Edit the app config to set a correct language code (subgroup index) for the app.

Note: In the future, language change in the app's config window in the GUI. Adding a new language, however, is anticipated to still be done manually.

**Protocol Image Fill Out**

Note: Currently this is very limited. In future versions, this will be very different. See comments in /py/match/protocol/borders.py for more information.

1)Go to py/match/protocol/borders.py and open it with any text editor.

2)To change location or size of some field, find relative area name in the file, and change the digits to whatever you need.

Digits go as follows: [top\_left\_x, top\_up\_y, top\_right\_x, top\_bottom\_y]. (0, 0) coordinate is top-left. Please, note that y actually increases downwards.

Remember to save a few pixels for an area border.

3)To exclude some areas from being filled out, just put a '#' sign before their name.

4)Currently, the app auto-processes groups of multiple elements with the same fill out logic (like boxes for each digit in match date area or lists of players). The user only needs to specify the area border for the whole group, and add the '\_\*digit\*' (underscore + a digit, 0 for horizontal groups, 1 for vertical) to the area name. However, shall the user decide to process each element manually, they will need to change \*\_digit\* to \*CharIndex\*, where Index is index of an element in a group, starting from zero, as well as specify area for each element. Remember to save a few pixels for an area border.

5)If you want to change the blank protocol image used in the app, get a picture of a new one in high resolution (preferably, a few thousand by a few thousand – online pdf to png convertors would do the trick) and png format, and save it with a name of 'image.png' inside of /py/match/protocol folder.

6)Save the file(s).