[Drafter](http://en.wikipedia.org/wiki/Drafter)

# Prerequisites

1. The lesson plan file should be saved in the .docx format. If the lesson plan is in the old .doc format, convert it to the .docx format by going to the **File** menu and clicking on the **Convert** button, and then saving the file.
2. The name of the lesson plan should contain the word “plan” and the three-digit lesson number. (E.g., “plan\_001.docx”, “001 plan.docx”, “lesson plan\_001\_final.docx” are good filenames; “Lesson 001.docx”, “Lesson 1 plan.docx”, “001.docx” are not.)
3. The drafter will process files in Russian except the word “Этап” must be translated to be “Stage.”

# Recent changes

* User interface overhaul: now you can run the program from any folder.
* The scripts are now generated without built-in keyboard shortcuts. Read <http://www.rmcity.net/wiki/KE_How_To:MS_Word#Setting_shortcut_keys_for_styles.2C_macros.2C_and_many_other_things> to learn how to setup your personal shortcuts if you use them.

# Using the program

### First run (the draft OS file doesn’t exist in the lesson plan’s folder; generating the draft OS file)

1. Mark up the lesson plan with the tags (see bellow).
2. Run the program and select the folder in which the lesson plan file is contained.
3. Click “Yes” for the correct lesson plan file (if there are multiple files in the folder that fit the filnename format, the program will ask about each one once, until you answer”Yes”).
4. Click “Yes” to generate a draft OS file.
5. If you’d like to manually edit the draft OS file before proceding to generating scripts (usually you do on the first run), when the program asks “Continue?”, click “No” – the program will exit. Otherwise, click “Yes” and go to step 2 of Additional runs.

### Editing the draft OS file manually

When editiing the draft OS file manually, you might want to specify the following (the draft scripts get updated during the next additional run):

* Tutor’s actual name instead of $Tutor, to generate random correct branches specific to that tutor.
  + Tutor’s correct responses are automatically generated based on EMW’s specification ([resources/response.csv](resources/responses.csv)).
  + If tutor’s name isn’t Martin, Angela, or Stephanie, the same response, “That’s right” will be substituted in all correct branches.
* Virtual students’ names and strength (these are transferred to the scripts along with the tutor’s name).
* You can change the conditions of the branches (e.g., change “Weak” to “If quiz Problem 1 was solved correctly”). The new conditions will be copied over to the draft scripts.
* You can add tables in the draft OS file (copy one from <resources/pieces.docx>, don’t include the ##$branch? line). That will result in generating additional empty scripts during the next run.

### Additional runs (the draft OS file exists in the lesson plan’s folder; generating/updating the draft scripts)

1. Repeat steps 2 and 3 of the First run.
2. Click “Yes” for the correct lesson plan, just like in the first run. The program will recognize the previously generated draft OS file and will ask you if you want to generate the draft scripts. Say “Yes.” If draft scripts already exist, the program will prompt if you’d like to overwrite them.

# Marking up the lesson plan

You can put one or several of the tags described below in each cell. **The tags are case sensitive.**

* It’s a good idea to place tags where they are easy to spot – at the end of the cell, although it generally does not matter where you put them.

You can also reversibly edit the lesson plan to get the following changes in the scripts:

* Striking-through a whole cell has the same effect as $cut on the Teacher and Student columns. By default, if the Teacher cell is cut, then the Student cell on the same row is also cut.
* If only part of cell is struck through, the ~~struck through text~~ will not be copied into the script and OS drafts. This could be used when you want to *partially* cut the content in a cell.
* Sidebar comments in the lesson plan are now copied into the draft scripts, above the content the comments are about, in the “KE” style.
* The text included in square brackets, like [something something], will now appear as sidebar comments (teacher’s comments). The color of text doesn’t actually matter, just makes it more visible. This applies to the Blackboard, Teacher, and Student columns in the lesson plan.
* The highlighting doesn’t affect anything.

# Tags

The following tags are allowed in the respective columns of a lesson plan. **The tags are case sensitive.**

| Board | Teacher | Student | Notebook |
| --- | --- | --- | --- |
| ($cut)  $new  $new(par)  $OS  $screen  $insa...$end  $insb...$end  $branch(par)...$endbranch | $cut  $OS | $cut  $tutor  $weak  $average  $strong | ($cut)  $wtd : coming one day! |

## “Board” column

### ($cut)

Do not cut the cell.

### $new

Starts a new lesson item.

* Each stage in the lesson plan starts with a new lesson item, so the $new tag isn’t needed for the first lesson item in a stage. The first lesson item in a stage get a the next available 3-digit number (after already used numbers) ending with 10.

### $new(par)

Starts a number of parallel lesson items (lesson level branches), the result depends on the value of the parameter “par.”

* The parameter “par” must be one of the following sequences of symbols:
  + empty:
    - $new()=$new: a single lesson item for all students
  + ‘-’:
    - $new(-): lesson item for all students and labeled skip if behind.
  + ‘w’, ‘a’, ‘s’, ‘wa’, ‘ws’, ‘as’, ‘was’:
    - $new(w): a single weak branch
    - $new(a): a single average branch
    - $new(s): a single strong branch
    - $new(wa): 2 branches: weak; average or strong
    - $new(ws): 2 branches: weak or average; strong
    - $new(as): 2 branches: average; strong
    - $new(was): 3 branches: weak; average; strong
  + any of the letters could be capitalized to indicate “skip if behind.” For example:
    - $new(wA): two branches: weak; average or strong, skip if behind
    - $new(WAS): three branches: weak, skip if behind; average, skip if behind; strong, skip if behind.
* In the draft OS file, you can change the conditions of the branches (e.g., change “Weak” to “If quiz Problem 1 was solved correctly”). The new conditions will be copied over to the draft scripts *during the next additional run*.

### $OS

Copies the content of the cell (in either blackboard or teacher column) into the draft OS file.

* This is done on the first run (when generating the draft OS file); the tag is ignored on the additional runs (when a draft OS file already exists).
* The cell content is copied in the description of the lesson item corresponding to the previous $new tag/$new(par) tag/beginning of a stage.
* Multiple $OS tags per lesson item are allowed.

### $screen

Inserts the content of the cell in the draft script, in “On Screen” style.

* Tables and word math formulas are transfered as is, and pictures are replaced with the text [DRAWING] in the draft script.

### $insa...$end

* + $insa should be placed in a row right AFTER a submit. (So, right after a row where the student cell does not have $cut/$tutor/$weak/$average/$strong.)
  + The content between $insa and its paired $end will be in the no response branch of that submit.
  + $end should be placed in the last row of content to be in the no response branch of that submit.
  + Between $insa and $end you can still use $cut/$tutor/$weak/$average/$strong/$OS.
* $insa...$end might behave strangely when combined with other tags, such as $new or $cut. You are advised to compare with the original lesson plan where you use them.

### $insb...$end

* + $end paired with an $insb tag should be placed in a row BEFORE a submit.
  + The content between $insb and its paired $end will be in the no response branch of that submit. (So the order or content is changed!)
  + $insb should be on the first row of content to be in the no response branch of that submit.
  + Between $insb and $end you can still use $cut/$tutor/$weak/$average/$strong/$OS.
* $insa--$end and $insb--$end might behave strangely if combined with other tags such as $new or $cut. You are advised to compare with the original lesson plan where you use them.

### $branch(par)...$endbranch

* + $branch generates an animator’s note saying that in-script branching begins.
  + $endbranch generate an animator’s note saying that in-script branching ends.
  + The content between $branch and $endbranch will go under the weakest branch.
  + The parameter par works in the exact way as in $new(par). For example, $branch(wA) generates two in-script branches: weak; average or strong, skip if behind. The content between $branch and $endbranch goes under the weak branch, and the average or strong, skip if behind branch will be blank with just an animator’s note.
  + $branch(par)--$endbranch and $insa/b--$end may be used together. Also, within the branch you can still use $cut/$tutor/$weak/$average/$strong/$OS.

## “Teacher” column

### $cut

### $OS

## “Student” column

### $cut

Omit the corresponding submit tree in the draft script.

### $tutor, $strong, $average, $weak

Instead of creating a submit tree, put the contents of the cell in the main flow of the script as being said by a character (tutor, the weak, average or strong student, resp.).

## “Notebook” column

### ($cut)

Similar to the ($cut) tag used in the Board column.

# Limitations

* Pictures (that require linking a file) and MathType formulas are NOT supported. For technical reasons these might never be supported.