# Project “Complex structured variables”

Damir Buskulic, Université Savoie Mont Blanc

## La Ruche, introduction

La Ruche is an interactive editor of exercises for the [WIMS educational platform](http://wims.auto.u-psud.fr/wims/) (see also [the WIMSEDU association](https://wimsedu.info/?page_id=68). The WIMS platform offers a rich panoply of resources and exercises with free access or for personalized study. The pedagogical resources include

* exercises with automatic feedback and correction, which allow the learners to work at their own pace;
* interactive exercises with randomly assigned data, which allow the learners to work on the same type of exercise but with different data and as often as they wish;
* many other functionalities not relevant for our purpose…

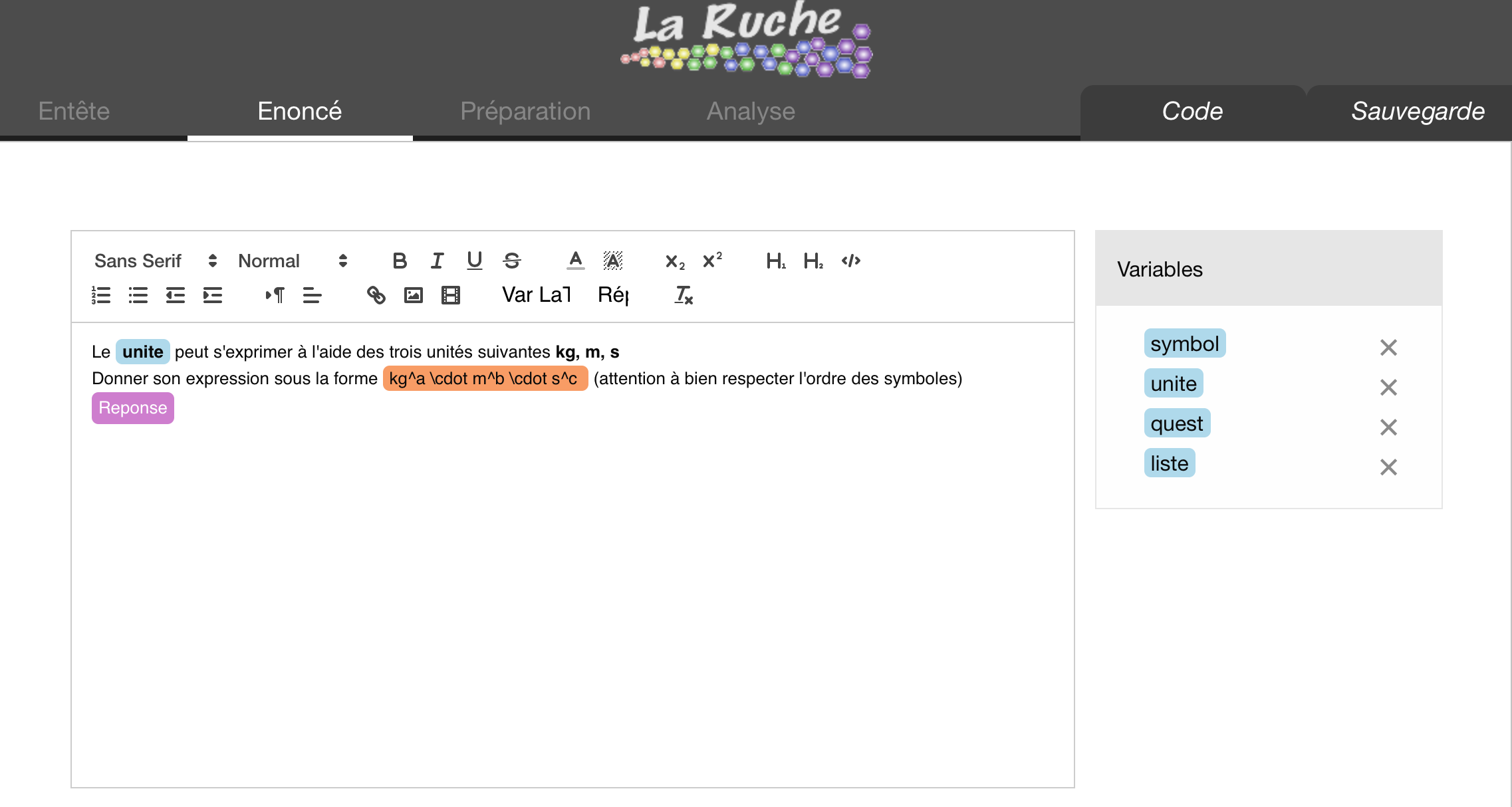
La Ruche, as an interactive editor of exercises, produces OEF (Open Exercise Format) code which is specific to WIMS. This code may be sent or copied to a WIMS server that will generate the exercise, show it to a student, mark it, etc... WIMS aiming at the development of exercises by teachers, it appeared that some people were not confident in writing code in the OEF language (you probably know some of them…). This is why we started the development of an editor that would be as much as possible interactive, wysiwyg and that would interact with a WIMS server.

The editor is a client side program written in JavaScript and using for now some open-source parts or packages :

* the [Foundation](http://foundation.zurb.com) framework
* the [Blockly](https://developers.google.com/blockly/) interactive programing editor
* the [Quill](http://quilljs.com) word editor which itself uses
* the [KaTex](https://khan.github.io/KaTeX/) library (mathematical equations display)

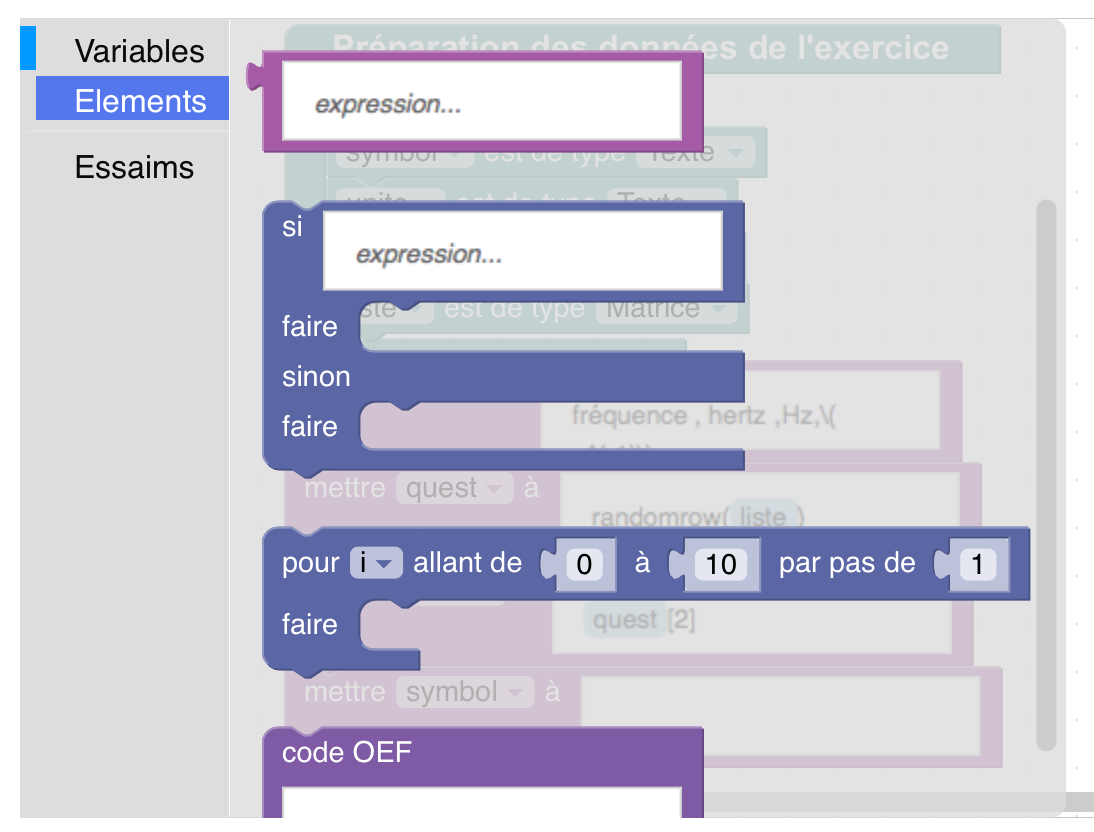
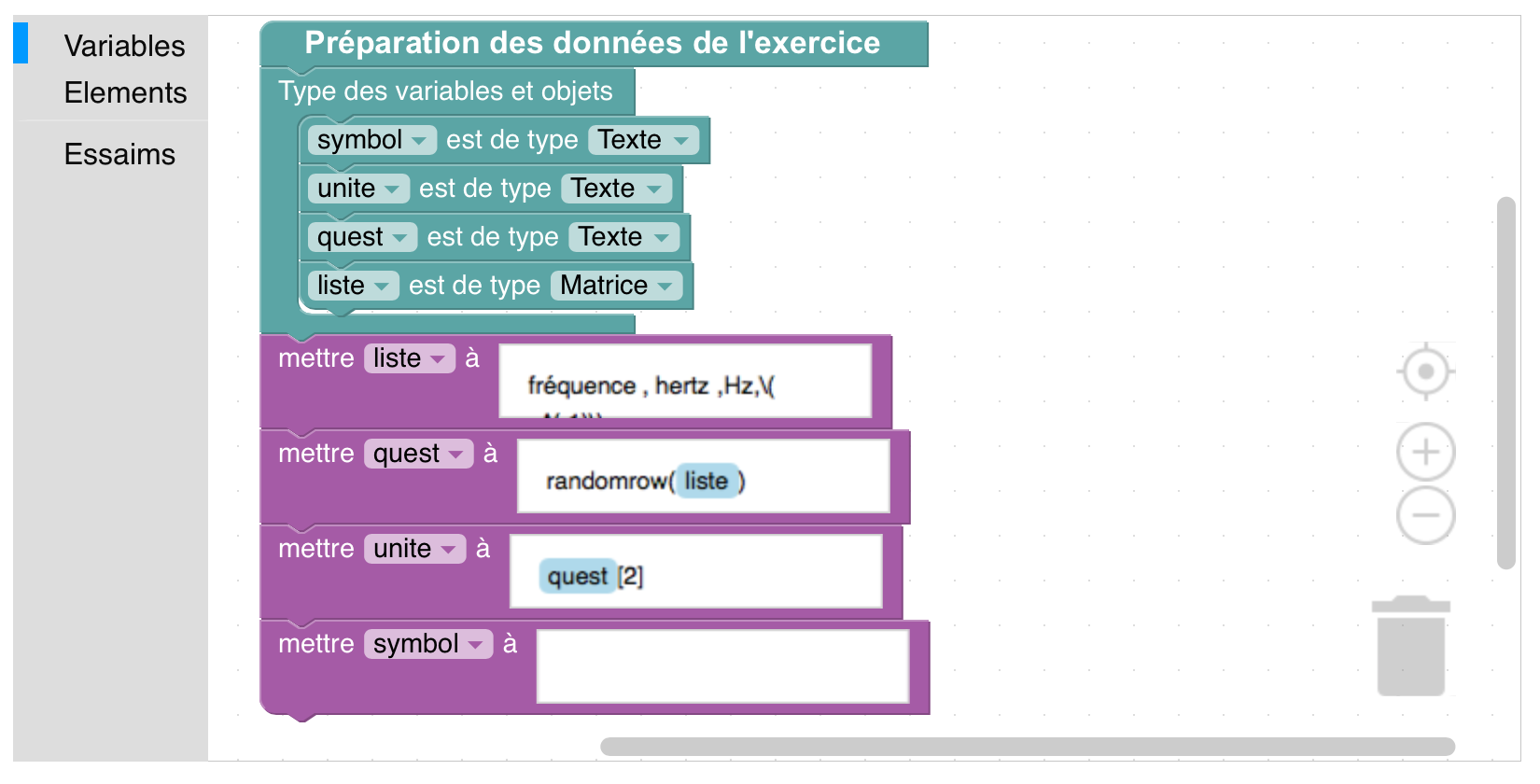
as well as somme utility packages and libraries :

* the JQuery library
* the [highlight](https://highlightjs.org/) library
* the [html2canvas](http://html2canvas.hertzen.com/) library



The editor is divided into four main and two auxiliary parts/tabs :

* the **Header** tab where the user enters meta-data for the exercise (title, author name, etc…)
* the **Statement** tab where the user enters the statement of his exercise in a Rich Text Editor, namely a special [Quill](http://quilljs.com) editor. This statement may contain references to variables, drawings, objects that are prepared in the **Preparation** tab
* the **Preparation** tab, where the user can make all the needed computations to set the variables, for example setting some of them as random. To that effect, a graphical Blockly editor is used.

* the **Analysis** tab, where the answer type for the student is set, as well as all necessary action (in another Blockly editor) after the student has entered his answer and pushed the "Send" button.

the two auxiliary tabs are :

* the **Code** tab where the OEF code is generated and displayed, so as to be copied to the WIMS server.
* the **Saving** tab that is used to generate a JSON format string corresponding to the exercise, save it or restore the exercise from a previously saved string.

## Constraints

For various reasons, we decided of the following constraints on the code development :

* the code is and should remain open-source
* the client-side is written in JavaScript and HTML5 (no Java)
* the code comments are in english
* the naming conventions will be discussed with the developpers
* vanilla JS is good, with JQuery where needed. The Foundation framework is good but should not be used very extensively so as not to prevent future evolutions (this may be negotiated on a case by case basis)
* the development is done with an Agile Software Development approach, to be discussed with the developers. This is needed because of the possible changes or adaptations that are inherent to the project.
* the development is done in collaboration with the WIMS and La Ruche developers

# Project details

## Rationale

In the current version of La Ruche, the variables created by the user may be of a few simple types :

* Real number
* Integer number
* Rational number
* Matrix
* Text
* Function
* Complex number

Actually, all the types are represented in the OEF language by text strings. In the text editor parts, the variables are represented by their name in a blue rounded rectangle : .

In the case of a matrix or an array, to extract a single value, there is no other way than to write explicitly : [] which is not very user friendly.

The project consists in

* Developing a way to treat arrays and matrices in a user-friendly and graphic way, though efficient
* Developing a very general variable object that could have complex type like a drawing (generated or external), a formula, an image, a video link, a piece of a drawing…

Both of these should have a consistent look and behavior throughout the editor, while giving the desired generated OEF code.

## Steps

The project has the following steps :

* + Development of a prototype with arrays and matrices
  + Tests and integration into LaRuche
  + Development of a generic object
  + Implementation in the case of
    - a drawing
    - a formula (the generation of the graphics will be done via Katex)
    - an image
    - the link to a video
    - … as needed and discussed with the developpers
  + Tests and integration into LaRuche

Some objects are already implemented into the Quill editor (equations, images, video,…) and need only a connector to LaRuche.

The project extends to March 2019, there are a few user testing and feedback sessions foreseen (so called “WIMSATHONs”).