

# USER MANUAL

## WHATIDO

### Version 3.0

| Date     | Version | Commentaire                 |
|----------|---------|-----------------------------|
| 10/04/17 | V3.0    | Initial Version of document |
|          |         |                             |
|          |         |                             |

## Table des matières

|         |   |    |
|---------|---|----|
| 1       | Présentation of the application.....                    | 3  |
| 1.1     | General information.....                                | 3  |
| 1.2     | An overview of the product on this document .....       | 3  |
| 1.3     | Use cases of the product .....                          | 4  |
| 2       | Installation.....                                       | 5  |
| 2.1     | Prerequisites .....                                     | 5  |
| 2.2     | Installation of Whatido.....                            | 6  |
| 3       | User Manual.....  | 8  |
| 3.1     | Configuring the Keyboard for Demonstrations .....       | 8  |
| 3.2     | Initial Setup Screen .....                              | 9  |
| 3.3     | Mouse Fixed (Large, Medium, Small) only. ....           | 10 |
| 3.4     | Mobile Mouse (Medium, Small, Tiny) only .....           | 11 |
| 3.5     | Enabling Keyboard Visibility .....                      | 12 |
| 4       | Appendix.....   | 15 |
| 4.1     | Preamble .....  | 15 |
| 4.2     | Keyboard Mapping.....                                   | 16 |
| 4.2.1   | Giving an identifier to the keyboard .....              | 16 |
| 4.2.2   | The files to be created for the keyboard mapping: ..... | 17 |
| 4.2.2.1 | config/keyboards/OCCITAN_Code.properties.....           | 17 |
| 4.2.2.2 | config/keyboards/OCCITAN_CtrlMod.properties.....        | 18 |
| 4.2.2.3 | config/keyboards/OC_oc_Trad.properties.....             | 18 |
| 4.3     | Viewing the Keyboard .....                              | 20 |
| 4.3.1   | Png image of the OCCITAN keyboard .....                 | 20 |
| 4.3.2   | Localisation des touches dans l'image.....              | 20 |
| 4.4     | Putting Sound .....                                     | 21 |
| 4.4.1   | Voices Files .....                                      | 22 |

# 1 Présentation of the application

## 1.1 General information

It is a utility that allows you to see on the screen the actions you do on the mouse (click / right / left / middle and the actions forward / back on the wheel of the mouse as well as actions on the keyboard (except numeric keypad).

The visualization is done through 2 transparent images of the mouse and the keyboard which remains in window always visible (except for some contextual menus where Windows is priority, but it is not very inconvenient).

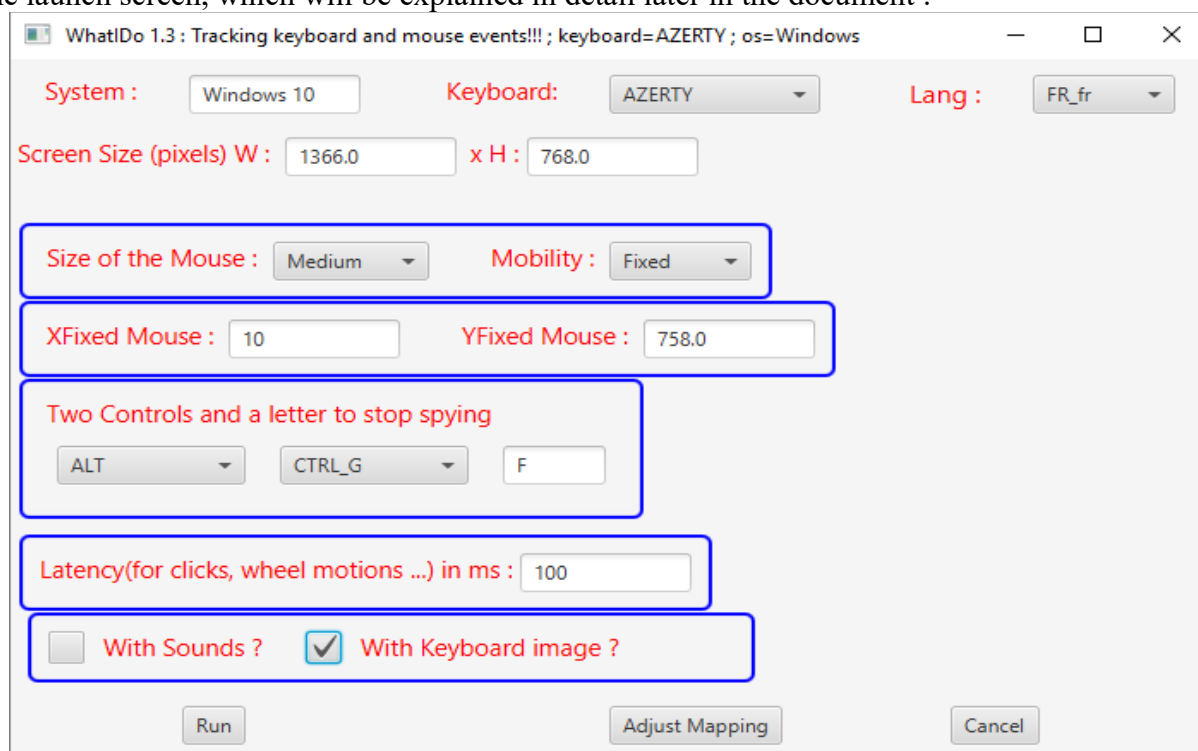
The software also works on Linux and certainly also on OS X (not tested by me on OS X, keyboard mapping to do), see in the appendix of this document how to realize a complete mapping if this software does not work correctly with your Micro / OS / keyboard. There may be deviations from the Windows / AZERTY mapping .

This software is based on the Library **JNativeHook** available at Github site :  
<https://github.com/kwhat/jnativehook>

The binary code is available on Maven Repo, I use the Version 2.1.0 :  
<https://mvnrepository.com/artifact/com.1stleg/jnativehook/2.1.0>

## 1.2 An overview of the product on this document

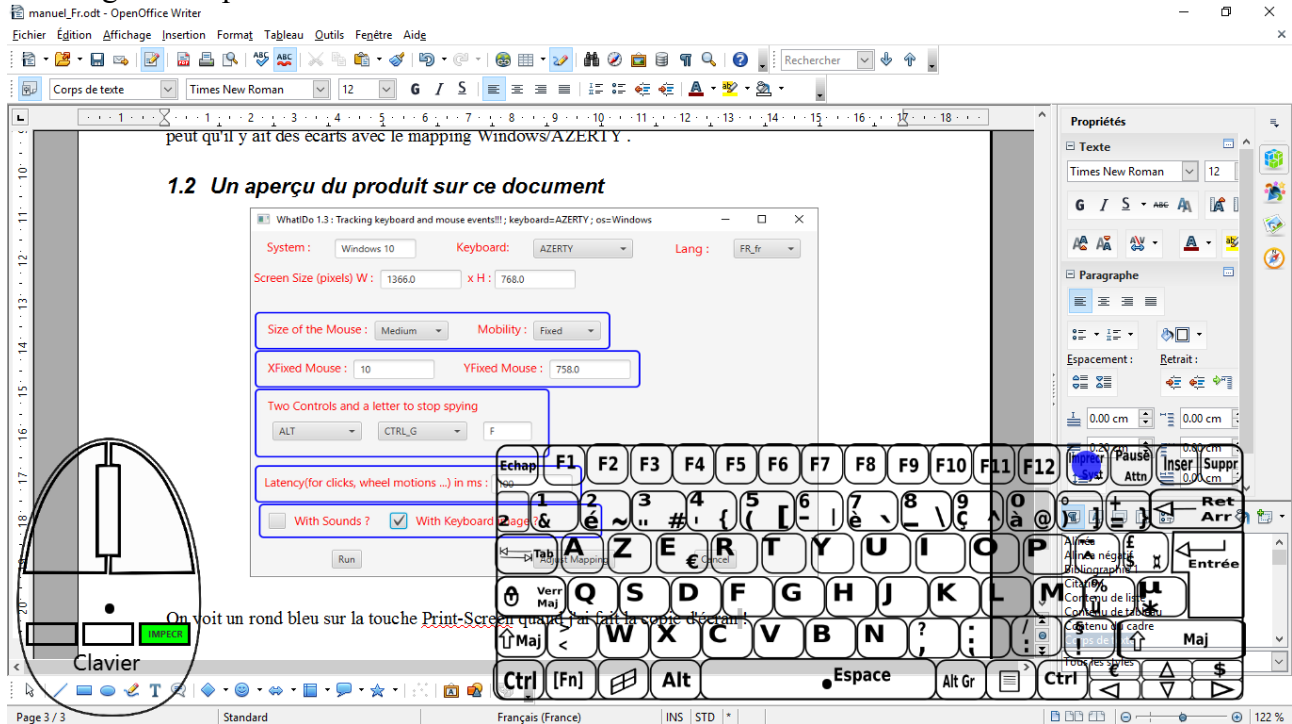
The launch screen, which will be explained in detail later in the document .



The screenshot shows the 'What!Do 1.3 : Tracking keyboard and mouse events!!! ; keyboard=AZERTY ; os=Windows' window. It contains several configuration sections:

- System :** Windows 10
- Keyboard:** AZERTY
- Lang :** FR\_fr
- Screen Size (pixels) W :** 1366.0 **x H :** 768.0
- Size of the Mouse :** Medium **Mobility :** Fixed
- XFixed Mouse :** 10 **YFixed Mouse :** 758.0
- Two Controls and a letter to stop spying**
  - ALT
  - CTRL\_G
  - F
- Latency(for clicks, wheel motions ...) in ms :** 100
- With Sounds ?** ☐ **With Keyboard image ?** ☒
- Buttons:** Run, Adjust Mapping, Cancel

## An image of the product in action:



We see a blue round on the Print-Screen key when I made the screenshot!

You can choose whether to display the keyboard or not, you can activate the sound that describes the actions performed on the mouse and keyboard.

For the mouse, there are 4 sizes to choose from (Large, Medium, Small, Tiny) and being mobile for the 3 smallest sizes and being fixed for the 3 largest sizes.

For mice of the fixed type, the black button in the middle makes it possible to move it in another part of the screen when it impedes.

## 1.3 Use cases of the product

**Whatido** works well with video projectors.

Possible uses are:

- Introduction to computers for the presentation of mouse actions and the keyboard
- Live software presentation by viewing all mouse and keyboard actions
  - ex : Navigation in file explorer, copying / pasting mechanisms, use of office tools ...
- Creation of video tutorials by recording the screen with whatido activated.

Tip: you can run the product twice:

- Once with a fixed Large or Medium mouse
- The other time with the Tiny / Mobile mouse that will follow the Windows cursor during its movements.

The two mice will show the actions performed.

Limitation with Powerpoint in Slideshow mode, the product (mouse image and / or keyboard image) does not appear in the foreground and can not be used in this case, it is necessary to remain in editing mode.

## 2 Installation

### 2.1 Prerequisites

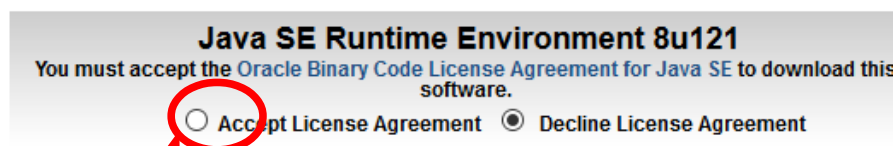
Whatido requires the presence of a recent Java Virtual Machine version greater than 1.8.0\_121. The 32-bit JRE version can be installed from the Oracle site:

<http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html>

Do you want to run Java™ programs, or do you want to develop Java programs? If you want to run Java programs, but not develop them, download the Java Runtime Environment, or JRE™.

If you want to develop applications for Java, download the Java Development Kit, or JDK™. The JDK includes the JRE, so you do not have to download both separately.

JRE 8u121 Checksum



Check the acceptance of the License.

Do you want to run Java™ programs, or do you want to develop Java programs? If you want to run Java programs, but not develop them, download the Java Runtime Environment, or JRE™.

If you want to develop applications for Java, download the Java Development Kit, or JDK™. The JDK includes the JRE, so you do not have to download both separately.

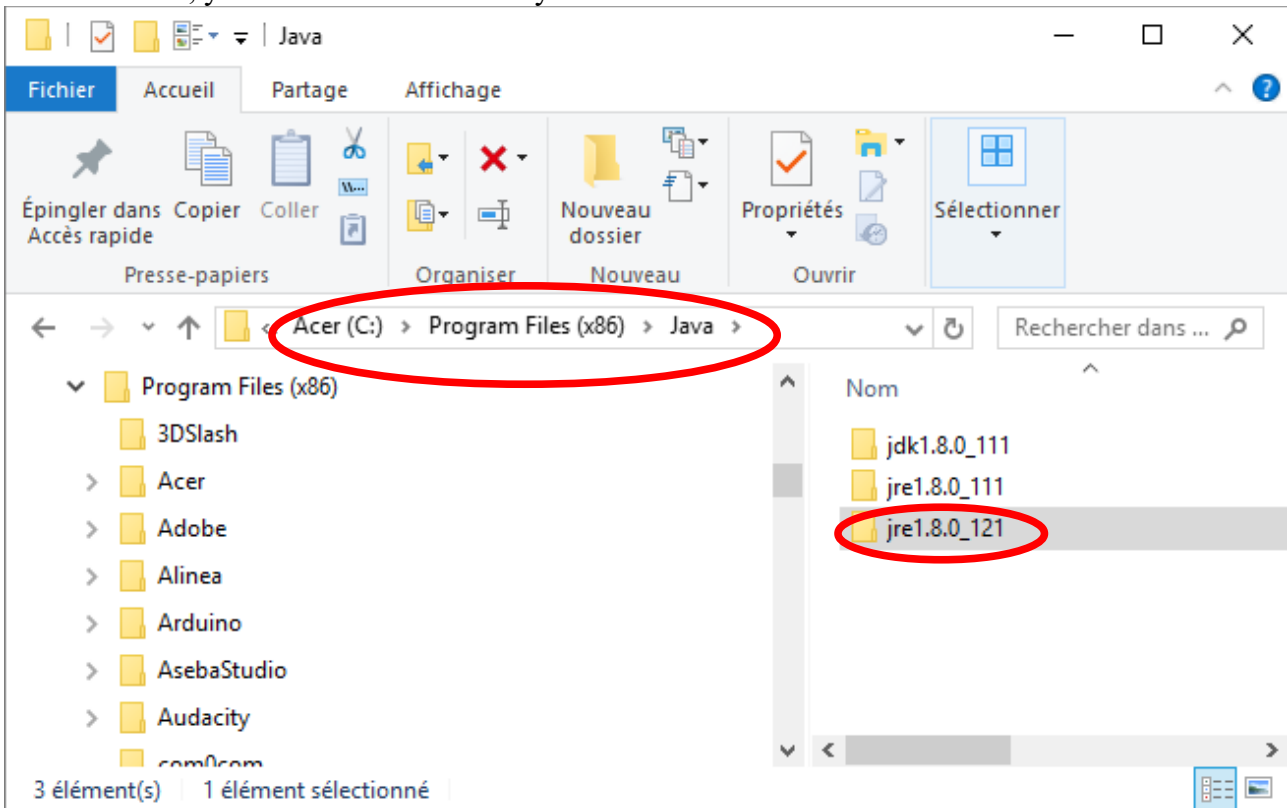
JRE 8u121 Checksum

| Java SE Runtime Environment 8u121   |           |  |
|---|-----------|--|
| You must accept the Oracle Binary Code License Agreement for Java SE to download this software.               |           |  |
| Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you now download this software. |           |  |
| Product / File Description  | File Size | Download   |
| Linux x86   | 56.92 MB  | <a href="#">jre-8u121-linux-i586.rpm</a>         |
| Linux x86   | 72.76 MB  | <a href="#">jre-8u121-linux-i586.tar.gz</a>      |
| Linux x64   | 54.39 MB  | <a href="#">jre-8u121-linux-x64.rpm</a>          |
| Linux x64   | 70.26 MB  | <a href="#">jre-8u121-linux-x64.tar.gz</a>       |
| Mac OS X  | 62.28 MB  | <a href="#">jre-8u121-macosx-x64.dmg</a>         |
| Mac OS X  | 53.91 MB  | <a href="#">jre-8u121-macosx-x64.tar.gz</a>      |
| Solaris SPARC 64-bit  | 52.05 MB  | <a href="#">jre-8u121-solaris-sparcv9.tar.gz</a> |
| Solaris x64   | 49.9 MB   | <a href="#">jre-8u121-solaris-x64.tar.gz</a>     |
| Windows x86 Online  | 0.7 MB    | <a href="#">jre-8u121-windows-i586.exe</a>       |
| Windows x86 Offline   | 53.81 MB  | <a href="#">jre-8u121-windows-i586.exe</a>       |
| Windows x86   | 59.17 MB  | <a href="#">jre-8u121-windows-i586.tar.gz</a>    |
| Windows x64 Offline   | 61.18 MB  | <a href="#">jre-8u121-windows-x64.exe</a>        |
| Windows x64   | 62.66 MB  | <a href="#">jre-8u121-windows-x64.tar.gz</a>     |

Choose the 32-bit version download (i586)

After downloading, it will be necessary to launch the corresponding .exe file in administrator mode (right click on the file, and choose launch in administrator mode).

Once installed, you should be able to see your JRE as shown below:



## 2.2 Installation of Whatido

The product is in the form of a zip whatido archive <Version> Exe.zip ex whatido3Exe.zip.

We assume for the rest of the installation that there is a C:\opt directory on your computer, otherwise you will create it or you will adapt the procedure described below.

- Position the whatido3Exe.zip archive under C:\opt and unzip
- Open the file C:\opt\whatido\script\whatido.cmd and adapt paths to red
 

```
Set PROJECT_HOME=C:\opt\whatido
Set CLASSPATH=%PROJECT_HOME%\lib\jnativehook-2.1.0.jar;.;%PROJECT_HOME%\lib\whatido-3.0.0.jar;
Set JAVA_HOME=C:\Program Files (x86)\Java\jre1.8.0_121\bin
start "" "%JAVA_HOME%\javaw" -Droot=%PROJECT_HOME% -Dhome=%PROJECT_HOME% -cp %CLASSPATH% com.jlp.whatido.Main
Exit
```
- Open the file C:\opt\whatido\script\testingMouseKeyboard.cmd and adapt paths to red

```
Set PROJECT_HOME=C:\opt\whatido
Set CLASSPATH=%PROJECT_HOME%\lib\jnativehook-2.1.0.jar;.;%PROJECT_HOME%\lib\whatido-3.0.0.jar;
Set JAVA_HOME=C:\Program Files (x86)\Java\jre1.8.0_121\bin
"%JAVA_HOME%\java" -Droot=%PROJECT_HOME% -Dhome=%PROJECT_HOME% -cp %CLASSPATH% com.jlp.whatido.MyMouseKeyboardListener
```

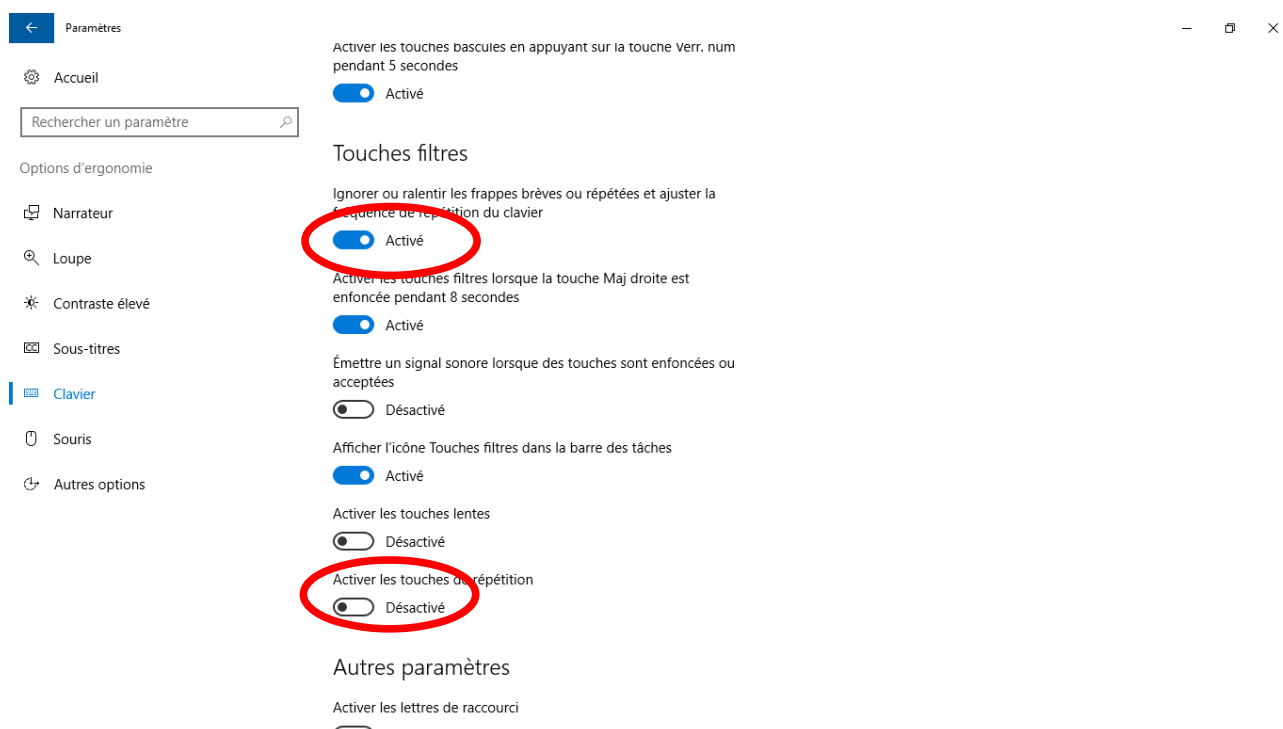
- Make a shortcut on the desktop for the file **C:\opt\whatido\script\whatido.cmd**
- Click on the shortcut to launch **whatido**.

## 3 User Manual

### 3.1 Configuring the Keyboard for Demonstrations

In order to avoid the unwanted repetition of keystrokes, when you hold down the key, the following settings must be made for Windows 10 (there are identical mechanisms for other versions of Windows as well as the different types of Windowing Linux: KDE, XFCE ... ; see tips.pdf):  
Settings → Ergonomics → Keypad → Activate the Filter Keys.

The configuration must conform to the following screen:



You must activate the filter keys and deactivate the repetition of the keys.



### 3.2 Initial Setup Screen

Below this screen with the explanations for each object numbered in the image

1: Automatic detection of the operating system, nothing to enter

2: Selecting the keyboard from a list. Linked to a different mapping based on Keyboard / OS. Choice to make. Keyboards can be added if necessary, see appendix.

3: Choice of language: especially used when the sound is activated. Available in French and English. See Annex for how to extend to other languages.

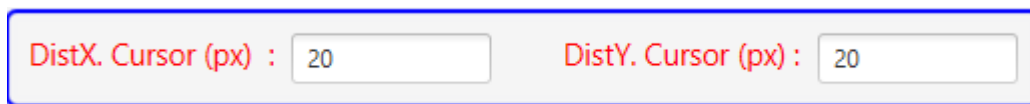
4: Automatic detection of the screen size (including the size available with a video projector). Do not modify

5: Choice to make on mouse size (Large, Medium, Small, Tiny)

6: Choice of the Fixed mouse behavior (still but moveable), or Mobile (follows the system cursor).

Large can only be Fixed, Tiny can only be Mobile. Medium and Small have both possibilities.

7: In the Fixed case, sets the location of the mouse to whatido launch.  
In the **Mobile** case we have:



A screenshot of a configuration window showing two input fields. The first field is labeled 'DistX. Cursor (px) :' and contains the value '20'. The second field is labeled 'DistY. Cursor (px) :' and also contains the value '20'. Both labels and the input boxes are highlighted with a blue border.

which makes it possible to fix the distance in pixel between the system cursor and the image of the mouse .

8: Combination of key to stop whatido (2 controls and one character ex: ALT CTRL\_G F). You can also stop the application by closing the window of the Java icon in the taskbar.

9: To improve the visibility of the actions, it is possible to adjust the latency (in ms) of the mouse and keyboard images after releasing the mouse button or the keyboard key.

10: Possibility to put the sound that describes the actions done on the mouse and the keyboard (quickly tiresome!)

11: Activation of the keyboard display at the bottom right of the screen. A black button in the middle of the spacebar allows to position it elsewhere on the screen.

12: Exit without launching the application

13: keyboard and mouse mapping button Explanations given in appendix, advanced use.

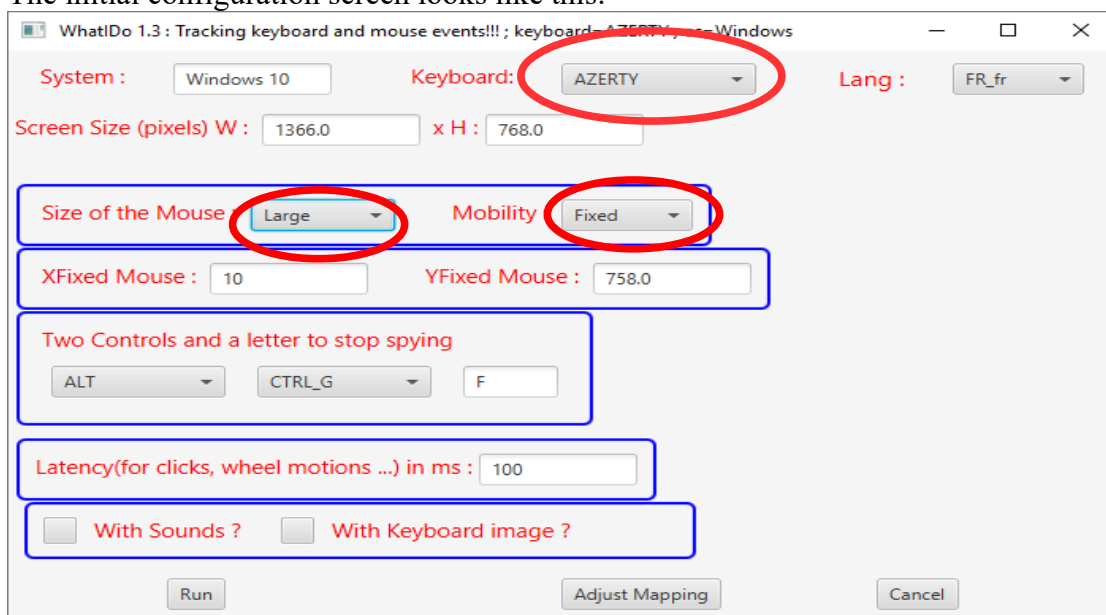
14: Launch button for the application.

### 3.3 Mouse Fixed (Large, Medium, Small) only.

The procedure is described for the Large size mouse, but it is identical for the other 2 sizes Medium and Small.

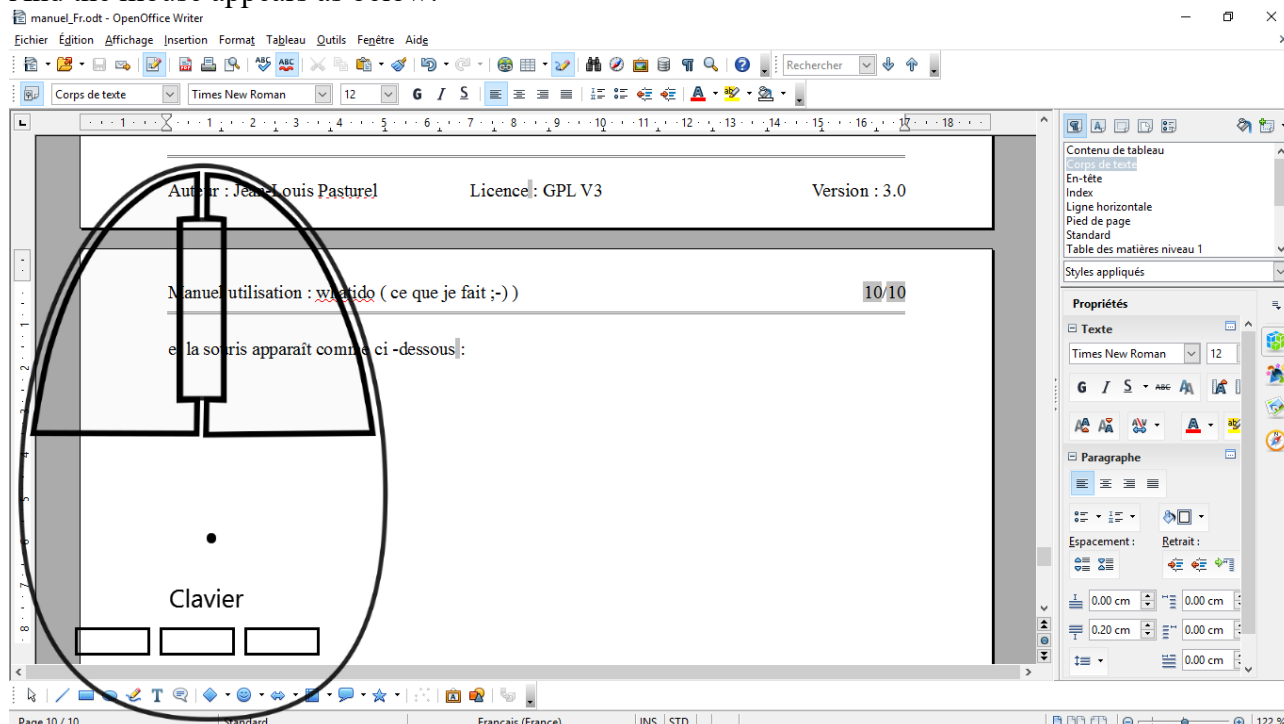
Turning the sound on or off does not affect the operation described below.

The initial configuration screen looks like this:



A screenshot of the 'Whatido 1.3 : Tracking keyboard and mouse events!!!' configuration window. The window has a title bar with standard Windows controls. The main area contains several settings: 'System' is set to 'Windows 10'; 'Keyboard' is set to 'AZERTY' (circled in red); 'Lang' is set to 'FR\_fr'; 'Screen Size (pixels) W' is 1366.0 and 'x H' is 768.0; 'Size of the Mouse' is set to 'Large' (circled in red); 'Mobility' is set to 'Fixed' (circled in red); 'XFixed Mouse' is 10 and 'YFixed Mouse' is 758.0; 'Two Controls and a letter to stop spying' shows 'ALT', 'CTRL\_G', and 'F' buttons; 'Latency(for clicks, wheel motions ...) in ms' is 100; and there are checkboxes for 'With Sounds ?' and 'With Keyboard image ?'. At the bottom are 'Run', 'Adjust Mapping', and 'Cancel' buttons.

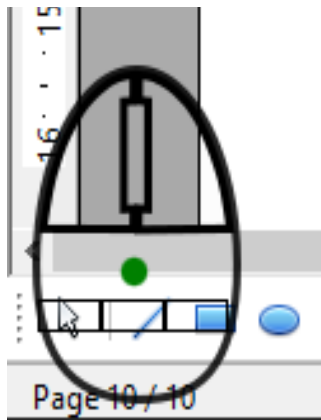
And the mouse appears as below:



Each action on the mouse indicates the button pressed (blue for left button, red for right and center button, high roller move in red and blue down)

The 3 text boxes of the mouse indicate the controls entered (1st and 2nd), the third indicates the character of the keyboard.

If you click on the black mouse button, it becomes Green. If you click elsewhere on the screen, the mouse will relocate to the clicked region.



### 3.4 Mobile Mouse (Medium, Small, Tiny) only

The procedure is described for the Small size mouse, but it is identical for the other 2 sizes Medium and Tiny.

Turning the sound on or off does not affect the operation described below.

The initial configuration screen looks like this:

What!Do 1.3 : Tracking keyboard and mouse events!!! ; keyboard=AZERTY ; os=Windows

System : Windows 10 Keyboard: AZERTY Lang : FR\_fr

Screen Size (pixels) W : 1366.0 x H : 768.0

Size of the Mouse: Small Mobility: Mobile

DistX. Cursor (px) : 20 DistY. Cursor (px) : 20

Two Controls and a letter to stop spying

ALT CTRL\_G F

Latency(for clicks, wheel motions ...) in ms : 100

☐ With Sounds ? ☐ With Keyboard image ?

Run Adjust Mapping Cancel

After the launch, the mouse image will follow the system cursor when it moves. The distance is set by the parameters identified 1 and 2 in the image above.

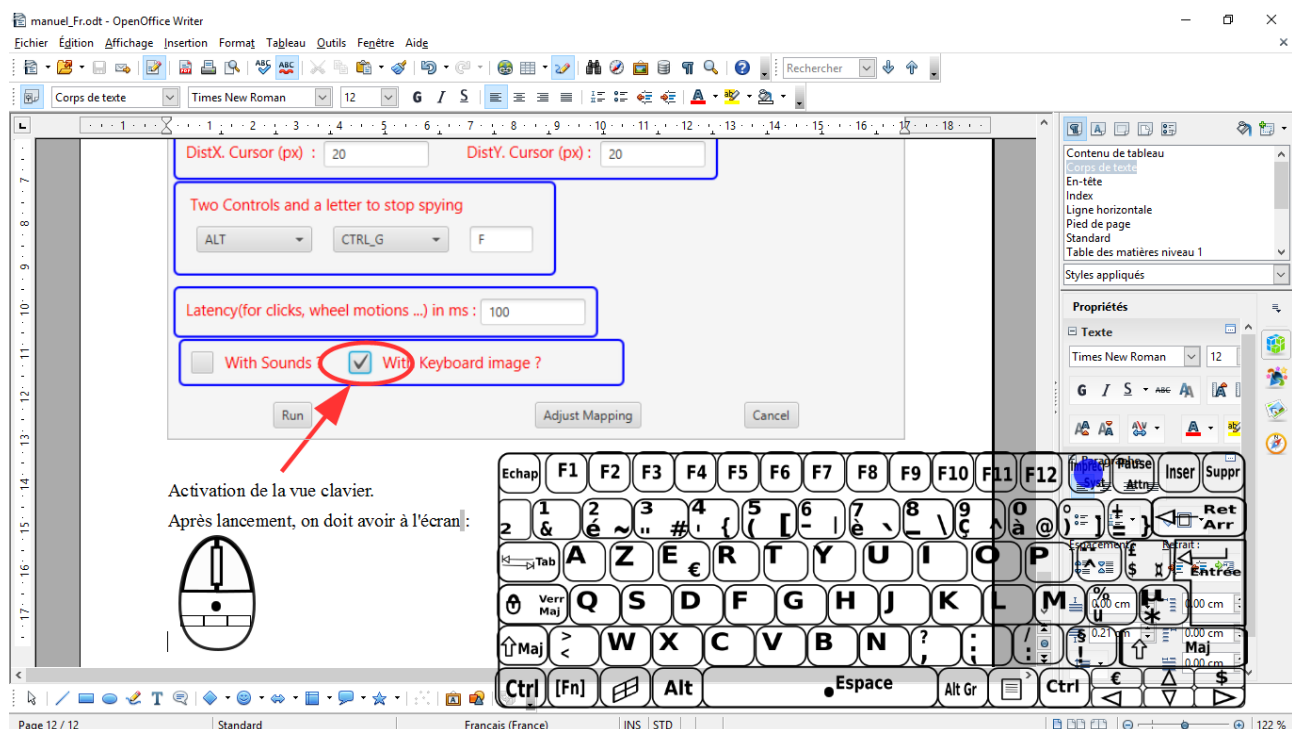
### 3.5 Enabling Keyboard Visibility

We're going to stay with the Small size mouse in mobile mode to activate the keyboard. The behavior of the mouse will not differ from the 2 cases seen just above.

The initial configuration screen looks like this:



Enabling the keyboard view after selecting the correct keyboard in the selection list  
After launch, you must have on screen:



In addition to the mouse actions, it appears on the keyboard image of the red circles (Control type CTRL, ALT, ALTGR) and a blue round according to the keys pressed.

The black dot in the middle of the space bar relocates the keyboard image by clicking it once to make it turn green and then clicking elsewhere on the screen to move the image of the keyboard.

## 4 Appendix

### 4.1 Preamble

If you have not found your keyboard, you can add one but this section will require some computer knowledge, especially for the creation of keyboard images, the use of drawing tools like Inksape:

<https://inkscape.org/fr/telecharger/windows/>

There are versions for Linux and OS X.

To scale the keyboard (800px \* 275px), you can use PhotoFiltre 7:

<http://www.photofiltre-studio.com/pf7.htm>

The configuration of the mouse and keyboards is outsourced to text files located in the config directory. Each line (which is not a comment) of these files is a pair type:

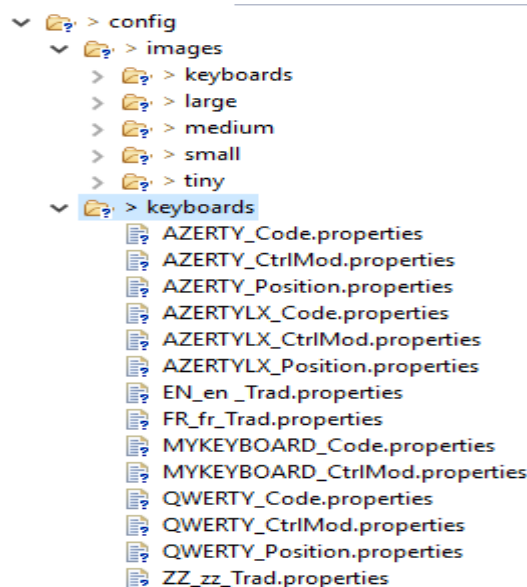
**key = value**

As far as the mouse is concerned, we only manage 3 buttons, the configuration file is for Windows: **config/windowsWheel.properties** :

```
#Mapping Mouse L: Left/Gauche
# R : Right/Droit
# M: Middle/Milieu
#Tue Feb 28 08:57:31 CET 2017
3=M
2=R
1=L
```

One can have, for some mice, a reversal of rank between the central button and the right button. To correct in this file.

**config/keyboards** :







## 4.2.2 The files to be created for the keyboard mapping:

Starting with the name OCCITAN, as far as the keyboard is concerned, it will be necessary to create:

- **config/keyboards/OCCITAN\_Code.properties**
- **config/keyboards/OCCITAN\_CtrlMod.properties**
- **config/keyboards/OC\_oc\_Trad.properties**

### 4.2.2.1 config/keyboards/OCCITAN\_Code.properties

The structure of each line of this file is given below:

**<keycode>\_<modifiers> = <keystroke-identifier>**

Lines starting with # are comments, and are not taken into account by whatido, when reading the file.

How do I get the **keycode** and the **modifiers**?

- Run a command console windows,
- To move under **whatido / script**
- Launch **.\TestingMouseKeyboard.cmd**
- Then close the helloWord window.

```

Microsoft Windows [version 10.0.14393]
(c) 2016 Microsoft Corporation. Tous droits réservés.

C:\Users\Jean-Louis>cd c:\opt\whatido\script

c:\opt\whatido\script>.\testingMouseKeyboard.cmd

c:\opt\whatido\script>Set PROJECT_HOME=C:\opt\whatido

c:\opt\whatido\script>Set CLASSPATH=C:\opt\whatido\lib\jnativehook-2.1.0.jar;. ;C:\opt\whatido\lib\whatido-3.0.0.jar;

c:\opt\whatido\script>Set JAVA_HOME=C:\Program Files (x86)\Java\jre1.8.0_121\bin

c:\opt\whatido\script>"C:\Program Files (x86)\Java\jre1.8.0_121\bin\java" -Droot=C:\opt\whatido -Dhome=C:\opt\whatido -cp C:\opt\whatido\lib\jnativehook-2.1.0.jar;. ;C:\opt\whatido\lib\whatido-3.0.0.jar; com.jlp.whatido.MyMouseKeyboardListener
Button =1
Button =2
Button =1
Key Pressed NativeKeyEvent.getKeyText : Alt
Key Pressed NativeKeyEvent.getKeyCode : 56 ;modifier=8
Key Pressed NativeKeyEvent.getKeyText : Impression d'écran
Key Pressed NativeKeyEvent.getKeyCode : 3639 ;modifier=8
  
```

The key code can be read and the modifier Disregard the getKeyText (pb linked to a library external to the product)

For example, for the a key (lowercase a):

```

Key Pressed NativeKeyEvent.getKeyText : A
Key Pressed NativeKeyEvent.getKeyCode : 30 ;modifier=0
  
```

We will put in the file :

**30\_0 = a**

For the capital A:

```
Key Pressed NativeKeyEvent.getKeyText : A  
Key Pressed getKeyCode : 30 ;modifier=1
```

and :

**30\_1 = A**

For another button like **Print Screen**:

```
Key Pressed NativeKeyEvent.getKeyText : Impression d'écran  
Key Pressed getKeyCode : 3639 ;modifier=0
```

We will put in the file :

**3639\_0 = PRECR**

The PRECR value is arbitrary and can then be translated into the OC\_oc\_Trad.properties file (see below). This should be done for all keys that are not a letter or a number (use the AZERTY\_Code.properties file to name these special characters)

It is necessary to pass all the keys of the keyboard except the numeric keypad, and for each key to play with the controls (CTRL, SHIFT, ALTGR) to have one line, in the configuration file, for each character available on the keyboard. For an AZERTY keyboard, we have about 175 lines (it's a little tedious).

Make the distinction for duplicate keys (Left and Right Shift, Left and Right Control)

#### **4.2.2.2 config/keyboards/OCCITAN\_CtrlMod.properties**

This file lists the keyboard controls and modifications. It is sufficient to report here the part of code found in the previous configuration for these keys only with the inversion of the key and of the value.

We should have a file identical to:

```
SHFT_L = 42_1  
CTRL_L = 29_2  
  
ALT = 56_8  
ALTGR = 56_130  
CTRL_R = 29_32  
SHFT_R = 3638_16
```

#### **4.2.2.3 config/keyboards/OC\_oc\_Trad.properties**

This file allows a translation of the special characters for a display in one of the 3 fields of the

mouse. If you do not match this file, you will see the value of the OCCITAN\_Code.properties file. In java, it is necessary to avoid working directly with characters specific to the language (eg French, accented characters, c cedille ...). This file allows translation into a universal character encoding called unicode.

Example in our case for é (**e acute accent**):

We noted in our file **OCCITAN\_Code.properties** the line:

```
# é eacute
3_0 = EACUTE
```

In our file **OC\_oc\_Trad.properties** on aura :

```
# é eacute
EACUTE = \u00E9
```

The unicode codes are made up for java of \u<codeHexaOfCharacter>

**The Hexadecimal code of unicode characters is given here:**

<https://unicode-table.com/fr/#control-character>

For example: [ => \u005B

|      | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | A   | B   | C   | D   | E   | F   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 0000 | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEL | BS  | HT  | LF  | VT  | FF  | CR  | SO  | SI  |
| 0010 | DLE | DC1 | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM  | SUB | ESC | FS  | GS  | RS  | US  |
| 0020 |     | !   | "   | #   | \$  | %   | &   | '   | (   | )   | *   | +   | ,   | -   | .   | /   |
| 0030 | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | :   | ;   | <   | =   | >   | ?   |
| 0040 | @   | A   | B   | C   | D   | E   | F   | G   | H   | I   | J   | K   | L   | M   | N   | O   |
| 0050 | P   | Q   | R   | S   | T   | U   | V   | W   | X   | Y   | Z   | [   | \   | ]   | ^   | _   |
| 0060 | `   | a   | b   | c   | d   | e   | f   | g   | h   | i   | j   | k   | l   | m   | n   | o   |
| 0070 | p   | q   | r   | s   | t   | u   | v   | w   | x   | y   | z   | {   |     | }   | ~   |    |
| 0080 | XXX | XXX | SPH | NBH | IND | NEL | SBA | ESA | HTB | HTJ | VTS | PLD | PLU | RI  | SS2 | SS3 |
| 0090 | DCB | PU1 | PU2 | STB | COH | MW  | SPA | EPA | SOB | XXX | SCI | OSI | ST  | OBC | PM  | APC |
| 00A0 |    |    |    |    |    |    |    |    |     | ¡   | ¢   | £   | ¤   | ¥   | ¦   | §   |
| 00B0 | ¨   | ©   | ª   | «   | ¬   | ­   | ®   | ¯   | °   | ±   | ²   | ³   | ´   | µ   | ¶   | ·   |

### 4.3 Viewing the Keyboard

This visualization is carried out through 2 files:

- A png image of the keypad for AZERTY keyboard:
  - **config\images\keyboards\AZERTY.png** size ( 800\*275 px)
- A configuration file for the location of the keys on the AZERTY keyboard image:
  - **config\keyboards\AZERTY\_Position.properties**

In our case, the name **AZERTY** must be replaced by **OCCITAN**.

#### 4.3.1 Png image of the OCCITAN keyboard

The goal is to get a **config\images\keyboards\OCCITAN.png** image size of 800 \* 275 px (this size is not required but it is correct for screens or video projectors)

The explanation of the use of Inkscape and PhotoFiltre is beyond the scope of this document.

You can start from the file :

**config\images\keyboards\AZERTY.svg** copied to **config\images\keyboards\OCCITAN.svg**.

Inkscape is then opened, and the characters of the keys are moved / added / deleted in order to put them back on the right key and in place (up / down / right / left) in the key to resemble the actual keyboard.

Then we export the part of the keyboard to the png format under the same directory giving the size in the name: ex **config\images\keyboards\OCCITAN\_1100\*400.png**

We close Inkscape by saving in svg format, we run PhotoFiltre and load the image **config\images\keyboards\OCCITAN\_1100\*400.png**

We duplicate the image, then we pass the size of the image to 800 \* 275 px (We can activate the distortion to have this size).

Afterwards save it as **config\images\keyboards\OCCITAN.png**.

#### 4.3.2 Localisation des touches dans l'image

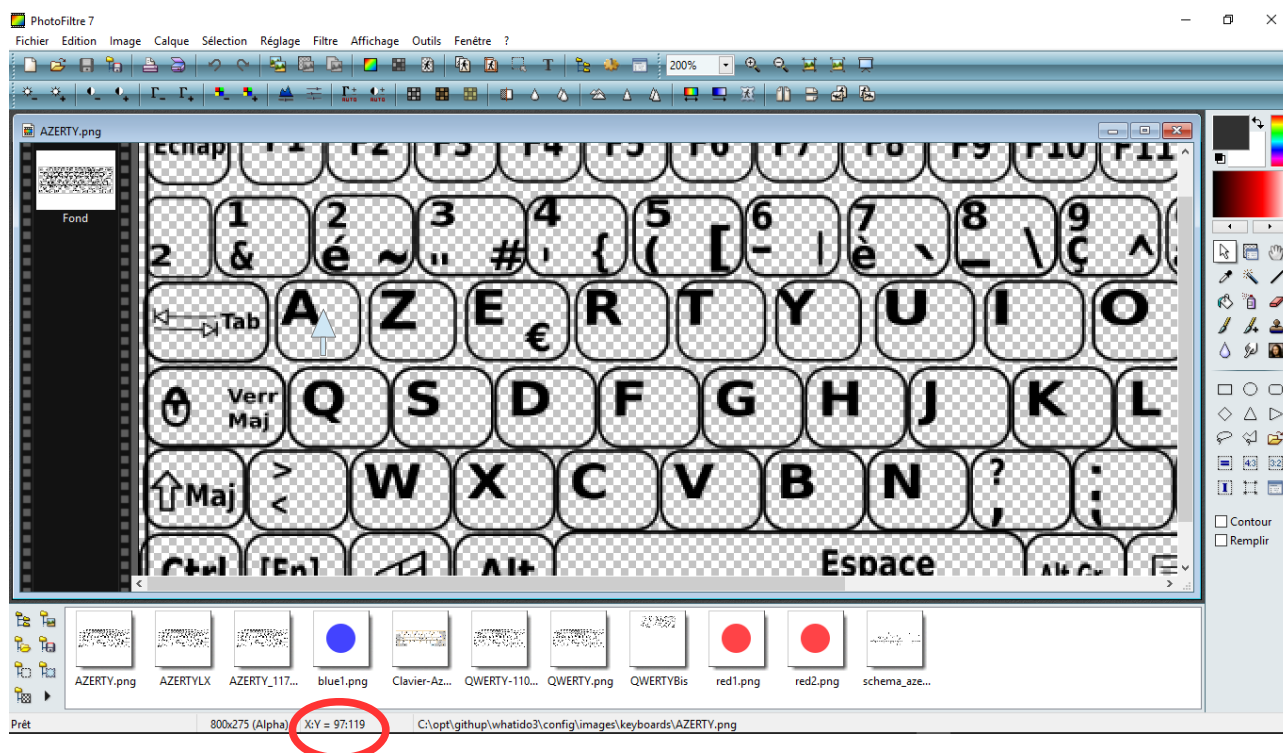
The **config\keyboards\OCCITAN\_Position.properties** file will be used by the product to locate the coordinates of the keystrokes in the keyboard image.

The structure of each line of this file is as follows:

keycode\_Modifier = X\_Y on image keyboard (800px \* 275px) File **OCCITAN.png**

To complete this file, you must:

- Launch the utility **.testingMouseKeyboard.cmd** already seen above
- Load **OCCITAN.png** image into **PhotoFiltre**.
- Then position each key and note the coordinates provided by PhotoFiltre. See image below.



For example, by positioning the cursor on the A key (positioning itself in the center of the key), the following coordinates can be read:

**X:Y = 97:119**

We will then have in the file **config\keyboards\OCCITAN\_Position.properties** the following line :

```
# A
30_0 = 97_119
```

It is thus necessary to parameterize all the keys of the keyboard (without modifiers / modifiers) without taking into account the possibilities linked to the modifiers (one will have for the characters the modifier modifier by default 0) However for the control keys, it will be necessary to add the possibilities of double keys (ALT + CTRL\_G ...) including the modifier, see the file: **AZERTY\_Position.properties**.

Example for a Right Control modifier:

```
# Ctrl D Ctrl R
29_32 = 607_250
29_40 = 607_250
29_33 = 607_250
```

## 4.4 Putting Sound

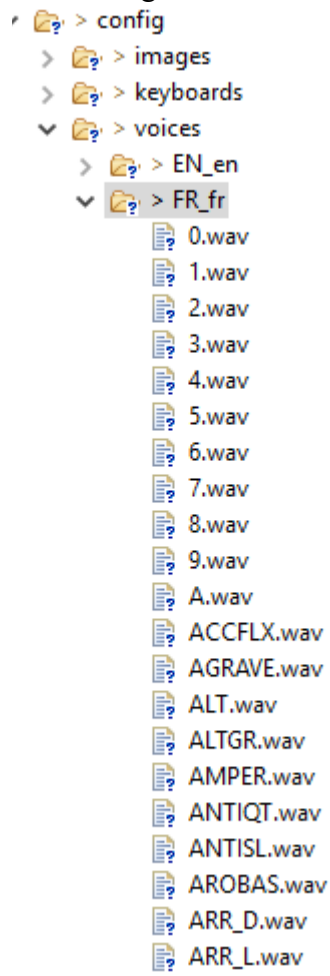
For each action on the mouse or keyboard, you can add a sound description. The sound files are in .wav format.

To record sounds, you can do this in two ways:

- Record using your computer's microphone and recording software ([Audacity](#))
- Using Text To Speech software, that's what I did with [balabolka](#)

### 4.4.1 Voices Files

The voice configuration tree is as follows:



The .wav files must be located under the directory:

**config/voices/OC\_oc** for our example.

The prefix of the file name corresponds to the key of the corresponding line of the file:

**config/keyboards/OCCITAN\_Code.properties**