Keyboard maker project

Low fidelity prototype

Nicolas, Antonin, Eva et Leïla

Low fidelity prototype

1. Interface

As described in the previous reports, our software will have 2 goals: defining the geometry of a custom keyboard and setting the keycodes for each key (the layout). With our requirement gathering, we discovered that these 2 goals are very linked one to another when prototyping: the geometry influences the layout and the reverse is also true.

For this reason, we chose to include the "geometric" features and the "layout" features in one single mode. It has benefits (the user can switch from one mode to another more easily) and drawbacks (if the tools were separated in 2 different modes, it could be faster to access them).

1.1. App logic

The whole app revolves around the main view which is always visible. Other aspects are dynamically added on the side menu or as contextual menus.

The main view in comprised of several parts:

- The entire state of the keyboard, most of which is visible to the user at any time
- The tool bar at the top, with the tool currently selected highlighted in the interface.
- A layer system, organizing the keys placed by the user. More on that below.
- A side menu giving the user access to several parameters depending on their current tool and selection.

The tool enabled by default is the selection tool. When the user clicks on a key, it will open all it's metadata in the right panel.

They are a few pop-up menus for specific features, like setting the keycode for a key.

1.2. Geometry features

The user starts creating with the "create new key" tool. Each time he clicks on the canvas, a new key is created. The user can change the dimensions of the newly created keys, and every subsequent key will be created with these dimensions.

There is also a "geometry" tool which allows the user to translate, rotate and scale the keys.

The user can also change the geometry of the keys by right-clicking on them. In the right-click menu, there is also an option to duplicate the key down or right, a given number of times.

Some of these tools may also be substituted by key combos for expert users.

1.3. Layer logic

The layer menu at the bottom left of the screen indicates which layer is selected. It is very important because the characters and behaviours of each key will depend on he active layer (for example, the base layer for the letters and the second layer for the symbols).

Each layer is activated by a combination of keys. The user can selected them from the interface. The special keys that activate a layer have a special color.

1.4. Layout features

Regarding the layout, each key has 2 properties: a **behaviour** and a list of **codes**.

The behaviour describes what type of key it is: a normal key, a modifier (like ctrl), a combo … The code is simply th character (like A, 1 or $) or the function (F12, HOME, volume down …) triggered by the key.

The 2 properties are always visible on the interface. The user can change them in the right panel.

One of the most time-consuming task is setting the letters on the keyboard. To make it easier, the user can double-click on any key, and it will write on the interface the message <press a key to set>. Then, any key the user presses will be registered in the app.

1.5. Templates

The user can also start from an already existing keyboard (a geometry + a layout).

Then, the user can modify everything and save it as a new keyboard.

2. Preview

When the user is happy with his keyboard, he can generate a visualization. It consists in a svg image for each layer. The exported file will most likely be JSON that the user can then import back onto the tool later.