**Dealing with different keyboard layouts**

 Bookmark this page

**Internationalize your keyboard controls**

**Introduction**

Please do not assume that each key is at the same location on the keyboard in every country!

We've shown how to detect keyup, keydown and keypress events using the DOM API, and how to use the keyCode property of the DOM event.

Be careful when you use the key events in your application, as keyboard layouts vary from one country to another. Most first person shooter games (FPS) use three keys located on the top left of your keyboard to move your character. French AZERTY keyboards will use ZQSD for this (Z = up/move forward, Q and D are for left/right and S is for down/move backward), while US keyboards will use WASD, for example. So keep in mind that keys are not located at the same place on keyboards from different countries.

Extract from the "[Internationalise your keyboard controls](https://hacks.mozilla.org/2017/03/internationalize-your-keyboard-controls/" \t "_blank)" article on MDN, by Julien Wajsberg:

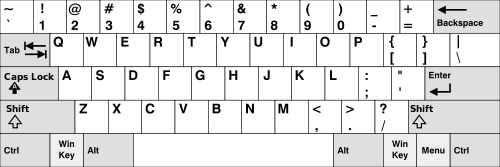
*"Recently I came across two lovely new graphical demos, and in both cases, the controls would not work on my French*[*AZERTY keyboard*](https://en.wikipedia.org/wiki/AZERTY)*.*

*There was the wonderful WebGL 2 technological demo*[*After The Flood*](https://www.youtube.com/watch?v=TT7ugKuUMv0/)*, and the very cute*[*Alpaca Peck*](http://codepen.io/shshaw/full/apwMwM/)*.*[*Shaw*](http://codepen.io/shshaw/)*was nice enough to fix the latter when I told him about the issue. It turns out the Web browser actually exposes a useful API for this."*

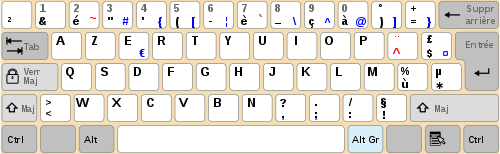
**One keyboard, many layouts**

For details, see [Wikipedia’s keyboard layout page](https://en.wikipedia.org/wiki/Keyboard_layout)!

[**QWERTY**](https://en.wikipedia.org/wiki/QWERTY)**layout, used in US, GB, etc.:**

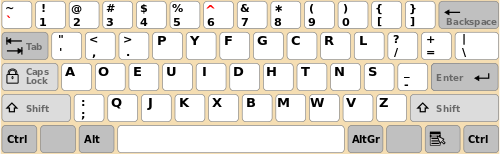


**AZERTY layout, used in some French-speaking countries:**

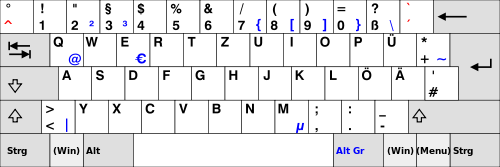


**In addition, QWERTZ keyboards are in use in Germany and other European countries, and DVORAK is another alternative to QWERTY:**

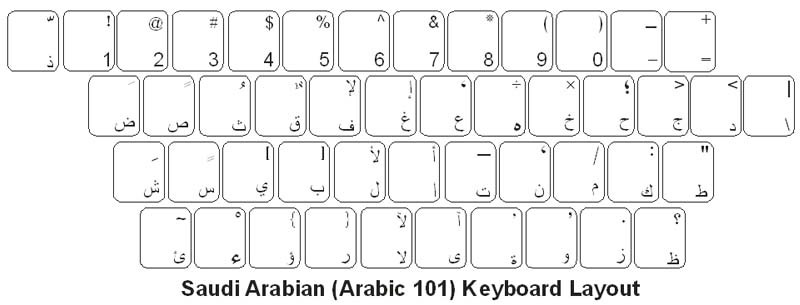
DVORAK:



QWERTZ:



**Saudi Arabic keyboard layout:**



**Bangla National (Jatiyo) keyboard:**



## Key and code properties

 Bookmark this page

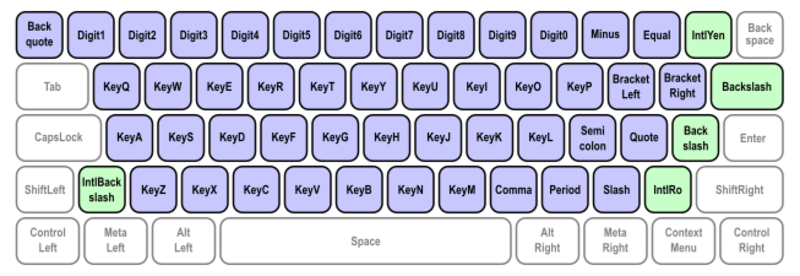
### New recommended properties you can use with modern browsers: key and code

You may have noticed that in some examples from the previous course page about key events, we used event.key in order to display the character that has been typed. The key property has been introduced with a new W3C API called UI Events (or DOM level 3 events), that has been discussed since 2000.  All major browsers have implemented this very practical key property. It comes with another property named code, which is what keyCode should have been. The value of the code property corresponds to a code that is more readable than the value of the oldkeyCode property.

* **key:**when the pressed key is a printable character, you get the character in string form. When the pressed key is not a printable character (for example: Backspace, Control, but also Enter or Tab which actually are printable characters), you get a multi-character descriptive string, like 'Backspace', 'Control', 'Enter', 'Tab'.
* **code**: Gives you the physical key that was pressed, in string form. This means it’s totally independent of the keyboard layout that is being used. So let’s say the user presses the Q key on a QWERTY keyboard. **Then event.code gives you 'KeyQ' while event.key gives you 'q'.  
    
  But when an AZERTY keyboard user presses the A key, he also gets 'KeyQ' as event.code, yet event.key contains 'a'. This happens because the A key on a AZERTY keyboard is at the same location as the Q key on a QWERTY keyboard.**As for numbers, the top digit bar yields values like 'Digit1', while the numeric pad yields values like 'Numpad1'.  
    
  Unfortunately this feature is not yet implemented by Microsoft IE/Edge but support is coming soon to Edge.

#### List of codes, the reference keyboard

There’s no existing keyboard with all the possible keys. That’s why the W3C published [a specification just for this](https://www.w3.org/TR/uievents-code/). You can read about the [existing mechanical layouts](https://www.w3.org/TR/uievents-code/#keyboard-layout) around the world, as well as their [reference keyboard](https://www.w3.org/TR/uievents-code/#code-value-tables). For instance here is their reference keyboard for the alphanumerical part:



You can also read [this document published by the W3C with explanations about all the possible values for the code property](https://www.w3.org/TR/uievents-key/).

Also read through [the examples given in the specification](https://w3c.github.io/uievents/#code-examples). They show very clearly what happens when the user presses various types of keys, both for code and key.

#### Example that displays the key and code values with your current keyboard

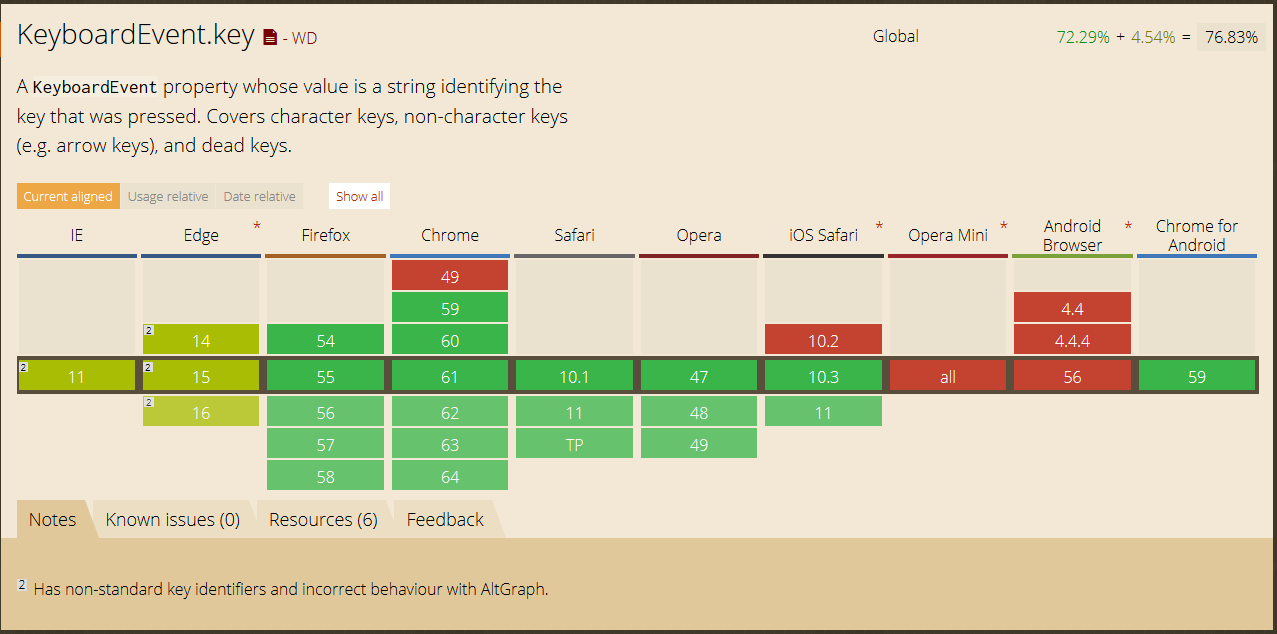
#### 

I encourage you to take a look and get at least an overview of this specification.

Please note that the W3C has also published a sibling specification describing the values for the key property.

#### Before looking at examples, let's see the current Web browser support for these properties:

As of September 2017:





See [an up-to-date version of the above table](http://caniuse.com/#feat=keyboardevent-code).