**[Always “use strict”](http://javascript.info/strict-mode" \l "always-use-strict)**

For now, it’s enough to know about it in general:

1. The "use strict" directive switches the engine to the “modern” mode, changing the behavior of some built-in features. We’ll see the details later in the tutorial.
2. Strict mode is enabled by placing "use strict" at the top of a script or function. Several language features, like “classes” and “modules”, enable strict mode automatically.
3. Strict mode is supported by all modern browsers.
4. We recommended always starting scripts with "use strict". All examples in this tutorial assume strict mode unless (very rarely) specified otherwise

**Variables**

We can declare variables to store data by using the var, let, or const keywords.

* let – is a modern variable declaration.
* var – is an old-school variable declaration. Normally we don’t use it at all, but we’ll cover subtle differences from let in the chapter [The old "var"](http://javascript.info/var), just in case you need them.
* const – is like let, but the value of the variable can’t be changed.

Variables should be named in a way that allows us to easily understand what’s inside them.

**[Uppercase const?](http://javascript.info/variables" \l "uppercase-const)**

We generally use upper case for constants that are “hard-coded”. Or, in other words, when the value is known prior to execution and directly written into the code.

In this code, birthday is exactly like that. So we could use the upper case for it.

In contrast, age is evaluated in run-time. Today we have one age, a year after we’ll have another one. It is constant in a sense that it does not change through the code execution. But it is a bit “less of a constant” than birthday: it is calculated, so we should keep the lower case for it.

**Reuse or create?**

And the last note. There are some lazy programmers who, instead of declaring new variables, tend to reuse existing ones.

As a result, their variables are like boxes into which people throw different things without changing their stickers. What’s inside the box now? Who knows? We need to come closer and check.

Such programmers save a little bit on variable declaration but lose ten times more on debugging.

An extra variable is good, not evil.

Modern JavaScript minifiers and browsers optimize code well enough, so it won’t create performance issues. Using different variables for different values can even help the engine optimize your code.