**JSHint**

JSHint — вариант JSLint с большим количеством настроек. JSLint прекрасно справлялся со своей задачей, но у него было две проблемы: не все рекомендации крокфорда нравились людям, а изменить настройки JSLint было невозможно.

Через некоторое время появлися JSHint. По большому счету это тот же JSLint только с возможностью управлять набором правил по которым валидируется код.

**JSLint**

JSHint - бесплатный, с открытым исходным кодом инструмент для проверки кода JavaScript.

Его легко настроить для соответствия определенным требованиям к коду и условиям его выполнения. Нужно просто отметить необходимые опции и среду выполнения кода (браузер, ES5, Node.js, Rhino), что позволяет распознавать предопределенные глобальные переменные.

Примечание: Данный проект является ответвлением популярного JSLint , который разработал Дуглас Крокфорд (Douglas Crockford - автор JSON).

Требования к коду

1. Глобальные и локальные переменные. Глобальные переменные по праву можно назвать величайшим злом не только в javascript, но и во всех языках программирования. Думаю, не надо объяснять, почему (если надо то вот). В javascript глобальные переменные можно объявить в локальном контексте, что только усугубляет ситуацию, потому что иногда можно объявить глобальную переменную по ошибке, просто поторопившись, и, если повезет с названием переменной, то разработчик который будет сопровождать ваш код сойдет с ума, пытаясь понять смысл, который попытался вложить в глобальную переменную автор.
2. Точка с запятой. При написании кода на javascript надо зарубить на носу раз и навсегда — точки с запятой ставятся после каждой инструкции. Исключения: for, function, if, switch, try и while. Обратите внимание, что не надо ставить точку с запятой при объявлении функции.
3. Перенос длинной строки. Строка более 80 символов — плохо. Забудьте про widescreen-мониторы, дело не в скроллинге, строка кода должна охватываться одним взглядом. Но и строки разбивать надо особо — чтобы после в месте разрыва строки явно отслеживалась синтаксическая незавершенность. JSLint допускает разрыв строки только после следующих операторов:

, . ; : { } ( [ = < > ? ! + - \* / % ~ ^ | &

== != <= >= += -= \*= /= %= ^= |= &= << >> || &&

=== !== <<= >>= >>> >>>=

JSLint запрещает перевод длинной инструкции после строки, числа, идентификатора, закрывающей скобки или суффикс-операторов: ) ] ++ —

1. Локальные пространства имен. В javascript локальное пространство имен порождает только одна инструкция — функция. Ни for, ни while, ни if не порождают локальных пространств имен.
2. Обязательные блоки. Строго обязательно использовать блок (оформляется фигурными скобками) при использовании if, for и других инструкций.

if (condition) statement;

// надо заменить на

if (condition) {

statement;

}

И первый и второй код работают одинаково, однако первого на практике следует избегать.

1. Конструкции языка.
   1. for in. Такого рода циклы надо использовать с осторожностью, поскольку проходя по всем членам объекта или массива, этот цикл также пройдет и по унаследованным свойствам/методам прототипа. JSLint требует чтобы тело такого цикла в обязательном порядке было обернуто блоком if.

for (name in object) {

if (object.hasOwnProperty(name)) {

....

}

}

* 1. switch. Обязательно использовать break после каждого case
  2. void. Не надо использовать это. Лучше обычный undefined, во избежание недопонимания.
  3. == и !=. Использовать с особой осторожностью. Необходимо помнить, что эти операторы приводят тип данных, то есть ' \t\r\n' == 0 это true.

**Источники**

<http://habrahabr.ru/post/74419/>

<http://habrahabr.ru/post/175283/>

<http://habrahabr.ru/post/189872/>

<http://ekaragodin.com/2015/08/06/validatsiia-javascript-koda/>