Variables

> Do not use `const` or `let` when targeting Internet Explorer without a transpiler.

- Always use `const` or `let` to declare variables. Not doing so will result in global variables. We want to avoid polluting the global namespace.

```
// bad
superPower = new SuperPower();

// good
const superPower = new SuperPower();
```

- Use `const` for all of your references; avoid using `var`.

```
// bad
var a = 1;
var b = 2;

// good
const a = 1;
const b = 2;
```

- Use one `const` or `let` declaration per variable or assignment.

```
// bad
const items = getItems(),
    goSportsTeam = true,
    dragonball = 'z';

// bad
// (compare to above, and try to spot the mistake)
const items = getItems(),
    goSportsTeam = true;
    dragonball = 'z';

// good
```

```
const items = getItems();
const goSportsTeam = true;
const dragonball = 'z';
```

- Group all your `const`s and then group all your `let`s.

```
// bad
let i, len, dragonball,
    items = getItems(),
    goSportsTeam = true;

// bad
let i;
const items = getItems();
let dragonball;
const goSportsTeam = true;
let len;

// good
const goSportsTeam = true;
const items = getItems();
let dragonball;
let i;
let length;
```

- Assign variables where you need them, but place them in a reasonable place.

```
// bad - unnecessary function call
function checkName(hasName) {
  const name = getName();

  if (hasName === 'test') {
    return false;
  }

  if (name === 'test') {
    this.setName('');
    return false;
  }
```

```
  return name;
}

// good
function checkName(hasName) {
  if (hasName === 'test') {
    return false;
  }

  const name = getName();

  if (name === 'test') {
    this.setName('');
    return false;
  }

  return name;
}
```

- Don’t chain variable assignments.

```
// bad
(function example() {
  // JavaScript interprets this as
  // let a = ( b = ( c = 1 ) );
  // The let keyword only applies to variable a; variables b and c become
  // global variables.
  let a = b = c = 1;
}());

console.log(a); // throws ReferenceError
console.log(b); // 1
console.log(c); // 1

// good
(function example() {
  let a = 1;
  let b = a;
  let c = a;
}());
```

```
console.log(a); // throws ReferenceError
console.log(b); // throws ReferenceError
console.log(c); // throws ReferenceError

// the same applies for `const`
```

- Avoid linebreaks before or after `=` in an assignment. If your assignment violates `max-len`, surround the value in parens

```
// bad
const foo =
  superLongLongLongLongLongLongLongFunctionName();

// bad
const foo
  = 'superLongLongLongLongLongLongLongString';

// good
const foo = (
  superLongLongLongLongLongLongLongFunctionName()
);

// good
const foo = 'superLongLongLongLongLongLongLongString';
```

Revision #3
Created 29 September 2020 06:00:19
Updated 8 October 2020 03:27:01