S2 Z2

Behaviors

Behaviors

- Global
- Strict
- ReadOnly
- Enumerable
- Delete
- Configurable
- Exists

Global scope

```
function show(param){
   var innerParam = param;
   console.log(param);
}

show('test');
```

```
test
```

```
var val1 = 'show’ ;

function show(param){
   var innerParam = param;
   console.log(param);
   console.log(val1);
}

show('test');
```

```
test
show
```

```
function show(param){
   var innerParam = param;
   console.log(param);
}

console.log(innerParam);

show('test');
```

```
ReferenceError: innerParam is not defined
```

```
function show(param){
   var innerParam = param;
   console.log(param);
}

show('test’);

console.log(innerParam);
```

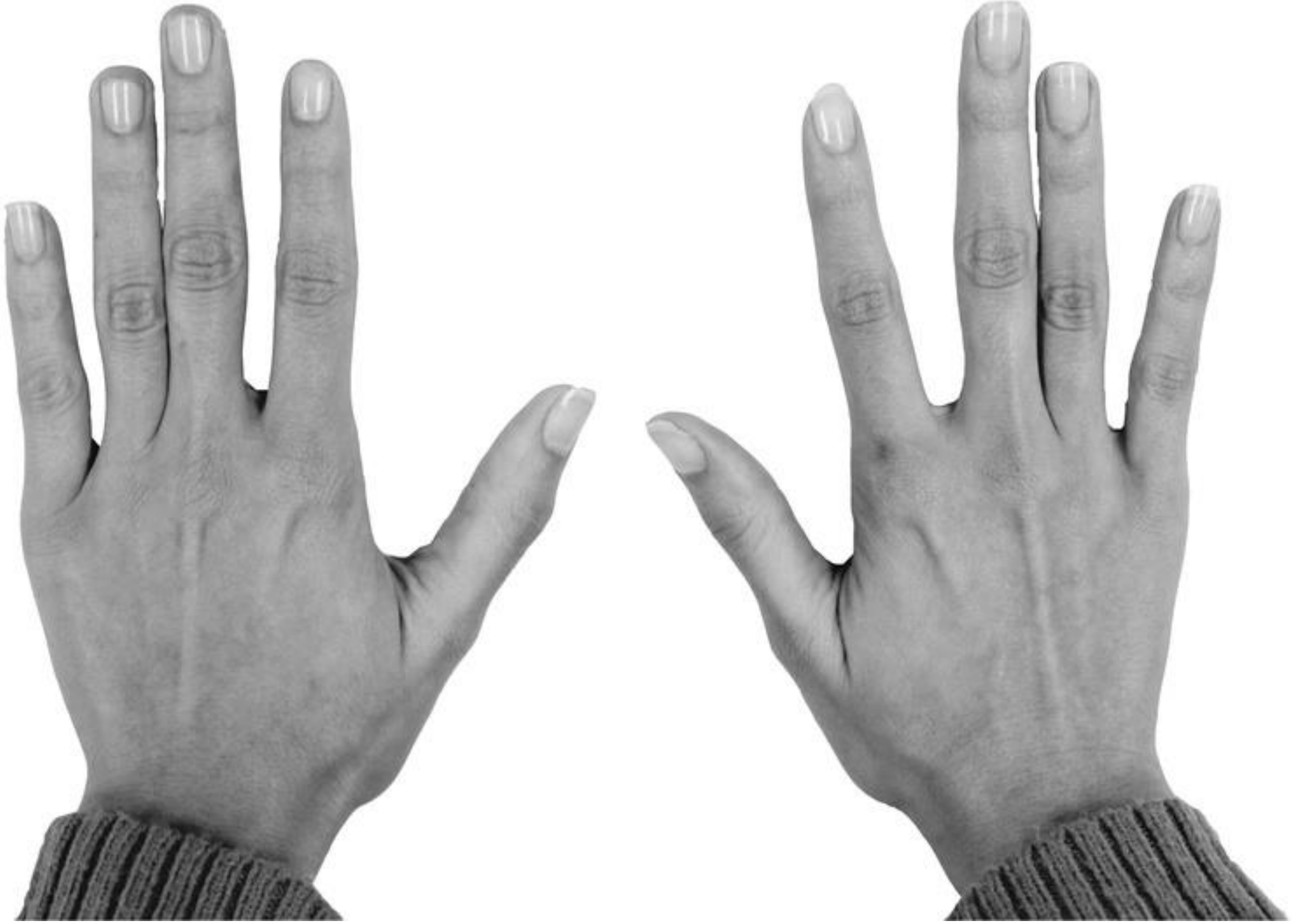
```
test
ReferenceError: innerParam is not defined
```

```
function show(param){
   innerParam = param;
   console.log(param);
}

console.log(innerParam);

show('test');
```

```
ReferenceError: innerParam is not defined
```

```
function show(param){
   innerParam = param;
   console.log(param);
}

show('test');
console.log(innerParam);
```

```
test
test
```



Strict mode

JavaScript please …. just stop helping !!!

```
'use strict’;

function show(param){
   innerParam = param;
   console.log(param);
}

show('test’);

console.log(innerParam);
```

```
innerParam = param;
^
ReferenceError: innerParam is not defined
```

```
function show(param){
   'use strict’;
   innerParam = param;
   console.log(param);
}

show('test’);

console.log(innerParam);
```

```
innerParam = param;
^
ReferenceError: innerParam is not defined
```

```
function show(param) {
    "use strict";
    var innerParam = param;
    console.log(param);
}

show("test");
notCreatedVariable = 5;
console.log(notCreatedVariable);
```

```
test
5
```

```
'use strict’;

notCreatedVariable = 5;
```

```
notCreatedVariable = 5;
^
ReferenceError: notCreatedVariable is not defined
```

```
'use strict’;

var obj = {};

obj.a = 'sdfs’;

console.log(obj);
```

```
Object {a: "sdfs"}
```

```
function foo() {
    console.log(this);
}
foo();
```

```
global {DTRACE_NET_SERVER_CONNECTION: , DTR…
```

```
function foo() {
    "use strict";
    console.log(this);
}
foo();
```

```
undefined
```

```
function foo() {
    "use strict";
    console.log(this);
}
foo.call(this);
```

```
Object {}
```

Read only

```
var obj = {};

Object.defineProperty(obj, 'ro',{
   enumerable: true,
   configurable: true,
   writable: false,
   value: 'Original Value'
});

console.log(obj.ro);
```

```
Original Value
```

```
var obj = {};

Object.defineProperty(obj, 'ro',{
   enumerable: true,
   configurable: true,
   writable: false,
   value: 'Original Value'
});

obj.ro = 'Altered Value’;

console.log(obj.ro);
```

```
Original Value
```

```
'use strict’;

var obj = {};

Object.defineProperty(obj, 'ro',{
enumerable: false,
configurable: false,
writable: false,
value: 'Original Value'
});

obj.ro = 'Altered Value’;

console.log(obj.ro);
```

```
obj.ro = "Altered Value";
^
TypeError: Cannot assign to read only property 'ro' of object '#<Object>'
```

Enumerable

```
var obj = {
    c : 'C Value’
};

obj.a = 'A Value’;

Object.defineProperty(obj, 'b',{
    enumerable: true,
    configurable: true,
    writable: true,
    value: 'B Value'
});

console.log(obj);
for (var key in obj) {
    console.log(key);
}
```

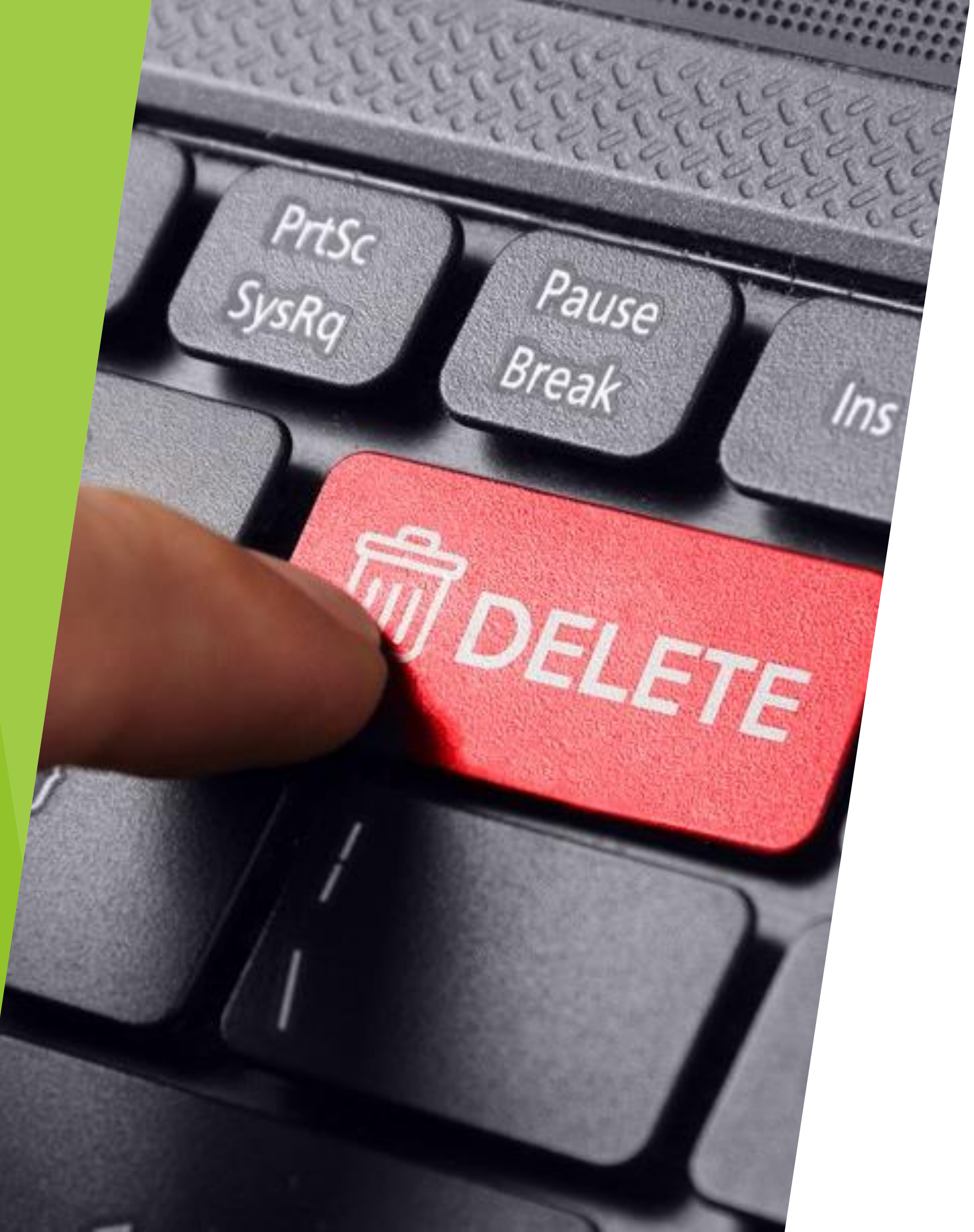
```
Object {c: "C Value", a: "A Value", b: "B Value"}
c
a
b
```

```
var obj = {
    c : 'C Value’
};

obj.a = 'A Value’;

Object.defineProperty(obj, 'b',{
    enumerable: false,
    configurable: true,
    writable: true,
    value: 'B Value'
});

console.log(obj);
for (var key in obj) {
    console.log(key);
}

console.log(obj.b);
```

```
Object {c: "C Value", a: "A Value", b: "B Value"}
c
a
B Value
```

```
var obj = {
    c : 'C Value’
};

obj.a = 'A Value’;

    Object.defineProperty(obj, 'a',{
    enumerable: false
});

Object.defineProperty(obj, 'b',{
    enumerable: false,
    configurable: true,
    writable: true,
    value: 'B Value'
});

console.log(obj);
for (var key in obj) {
    console.log(key);
}

console.log(obj.a);
console.log(obj.b);
```

```
Object {c: "C Value", a: "A Value", b: "B Value"}
c
A Value
B Value
```



Delete

```
var obj = { a: "A", b: "B" };
console.log(obj);
delete obj.b;
console.log(obj);
```

```
Object {a: "A", b: "B"}
Object {a: "A"}
```

```javascript
var x = 6;

delete x;

console.log(x);
```

```
6
```

```
var obj = {
   a:'A’,
   b:'B’
};

delete obj;

console.log(obj);
```

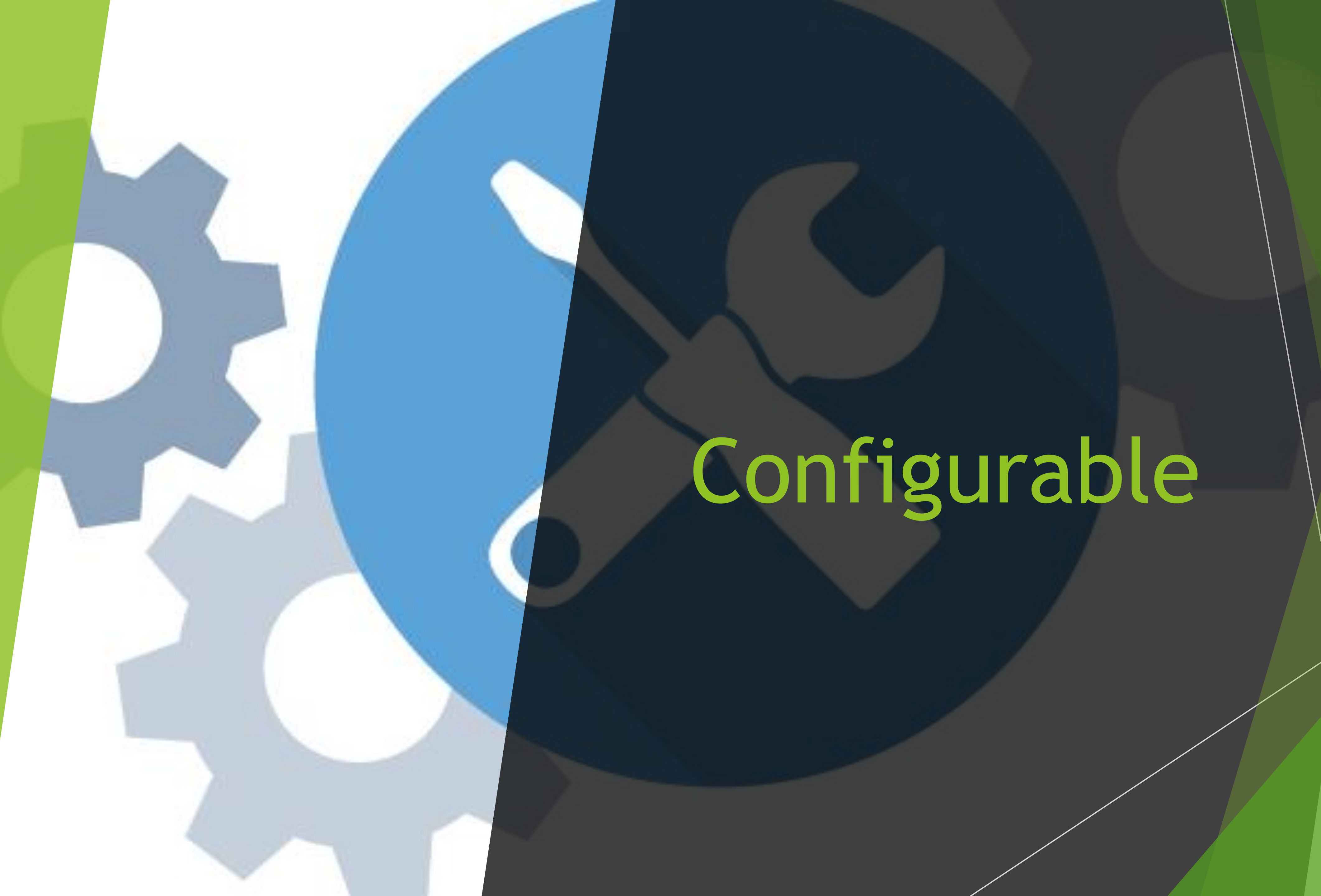
```
Object {a: "A", b: "B"}
```

```
'use strict’;

var x = 6;

delete x;

console.log(x);
```

```
delete x;
^
SyntaxError: Delete of an unqualified identifier in strict mode.
```

```
'use strict’;

var obj = {
   a:'A’,
   b:'B’
};

delete obj;

console.log(obj);
```

```
delete x;
^
SyntaxError: Delete of an unqualified identifier in strict mode.
```



Configurable

```
var myObject = {};
Object.defineProperty(myObject, "a", {
    value: 4,
    writable: true,
    configurable: false,
    enumerable: true
});
console.log(myObject.a); // 4
myObject.a = 5;
console.log(myObject.a); // 5
Object.defineProperty(myObject, "a", {
    value: 6,
    writable: true,
    configurable: true,
    enumerable: true
}); // TypeError - one way
```

TypeError: Cannot redefine property: a

```
var myObject = {
   a: 2
};
console.log(myObject.a); // 2
delete myObject.a;
console.log(myObject.a); // undefined
Object.defineProperty(myObject, "b", {
   value: 3,
   writable: true,
   configurable: false,
   enumerable: true
});
console.log(myObject.b); // 3
delete myObject.b;
console.log(myObject.b); // 3
```

```
// constant
var myObject = {};
Object.defineProperty(myObject,
"FAVORITE_NUMBER", {
    value: 42,
    writable: false,
    configurable: false
});
```

```
var myObject = {
   a: 2
};
Object.preventExtensions(myObject);
myObject.b = 3;
console.log(myObject.b); // undefined
```

```
"use strict";
var myObject = {
   a: 2
};
Object.preventExtensions(myObject);
myObject.b = 3;
console.log(myObject.b); // undefined
```

```
myObject.b = 3;
^
TypeError: Cannot add property b, object is not extensible
```

```
var myObject = {
    a: 2
};
Object.seal(myObject);
delete myObject.a;
myObject.b = 3;
console.log(myObject);
```

```
Object {a: 2}
```

```
var myObject = {
a: 2,
c: undefined
};

console.log(myObject.a); // 2
console.log(myObject.b); // undefined
console.log(myObject.c); // undefined
console.log(myObject.constructor); // function

// Uses prototype
console.log("a" in myObject); // true
console.log("b" in myObject); // false
console.log("c" in myObject); // true
console.log("constructor" in myObject); // true

console.log(myObject.hasOwnProperty("a")); // true
console.log(myObject.hasOwnProperty("b")); // false
console.log(myObject.hasOwnProperty("c")); // true
console.log(myObject.hasOwnProperty("constructor")); // false
```