What is a JavaScript object?

What isn't it?

A map from strings to any value except `undefined`.

It's not derived from a class.

Give the object literal syntax.

```
{
  key1 : val1,
  ...
}
```

Keys are automatically strings so they don't require quotes unless they are reserved words.

How are the values in objects accessed?

```
obj["quoted-key"]
obj.key
```

What are the consequences of bad object keys?

Accessing a non-existent key results in `undefined`.

Accessing a key of `undefined` results in a `TypeError`.

How do you update/augment objects?

Through assignment on the keys.

Object literals are linked to what?

Object.prototype

How does this book suggest creating new objects?

Using a special function.

```
if (typeof Object.create !== 'function') {
    Object.create = function (o) {
        var F = function () {};
        F.prototype = o;
        return new F();
    }
}
```

How is the prototype chain affected by update/access of members?

- For updates only the object's descendants get the new member, not the ancestors.
- For deletion only the object is affected.

What are two common problems with object access?

How are these problems solved?

- Functions are accessed, even when only data is desired. A conditional with `typeof` reflection can be used to filter undesired members.
- The entire prototype chain is accessed. `hasOwnProperty` can be used to discriminate inherited members.

Describe object deletion.

Use `delete` on a member. It does not affect ancestors.

How can you reduce the risks inherent in global state?

Use a single global object variable for all the program's state.