```
// Defina um novo valor de n, mas apenas se for maior que
atual
Defina o próximo (n) {
if (n> this._n) this._n = n;
caso contrário, jogue novo erro ("o número de série só pode ser definido
para um valor maior ");
}
};
serialnum.Next = 10;// Defina o número de série inicial
serialnum.next // => 10
serialnum.Next // => 11: Valor diferente a cada vez
nós chegamos a seguir
Finalmente, aqui está mais um exemplo que usa um método getter para
Implementar uma propriedade com comportamento "mágico":
// Este objeto tem propriedades de acessórios que retornam aleatório
números.
// A expressão "Random.octet", por exemplo, produz um
número aleatório
// entre 0 e 255 cada vez que é avaliado.
const aleatoriamente = {
obtenha Octet () {return Math.floor (Math.random ()*256);},
obtenha uint16 () {return math.floor (math.random ()*65536);},
obtenha int16 () {return
Math.Floor (Math.Random ()*65536) -32768;}
};
```

6.11 Resumo

Este capítulo documentou objetos JavaScript em grande detalhe, cobrindo tópicos que incluem:

Terminologia básica do objeto, incluindo o significado de termos como propriedade enumerável e própria.

Sintaxe literal de objetos, incluindo muitos novos recursos no ES6