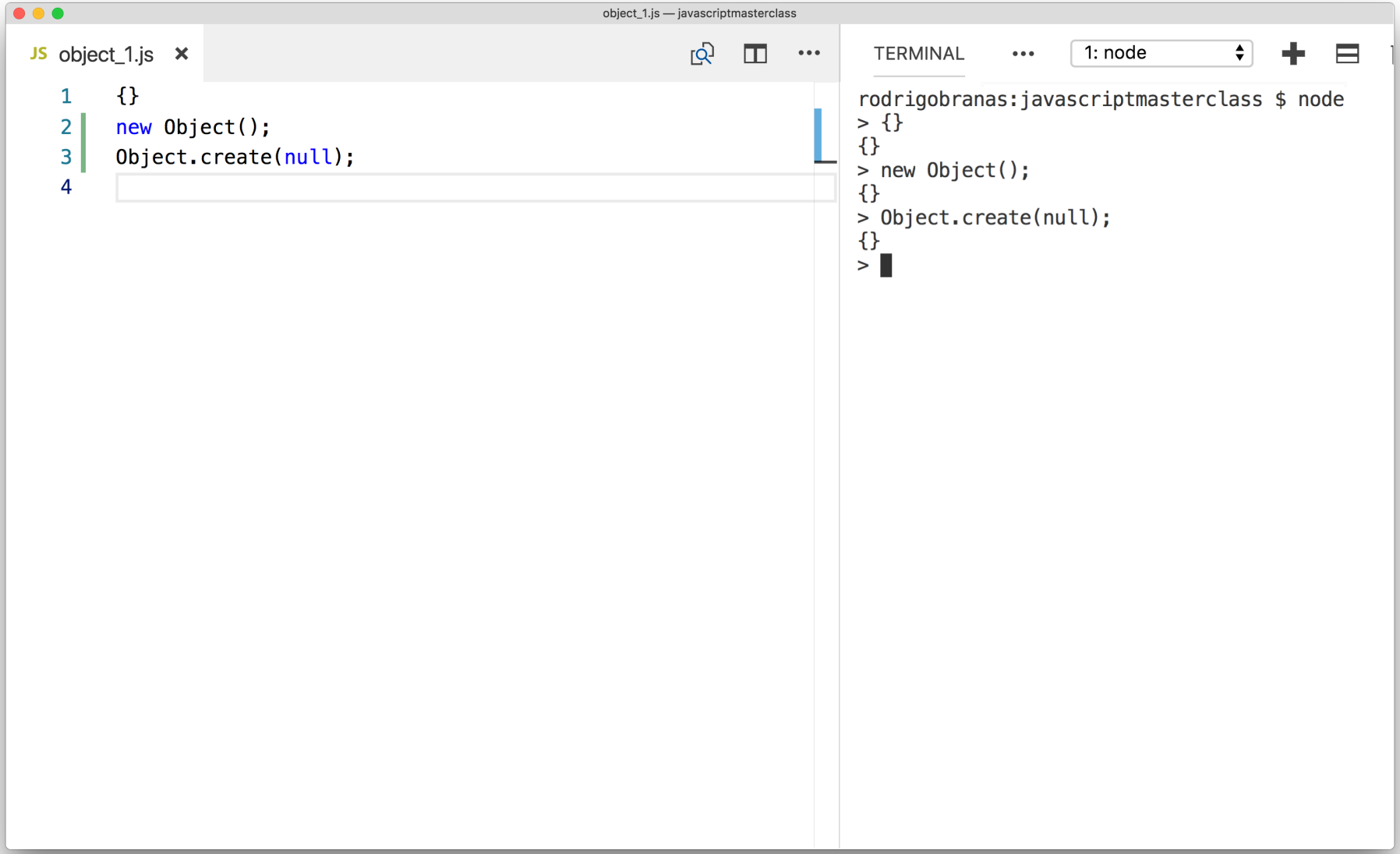
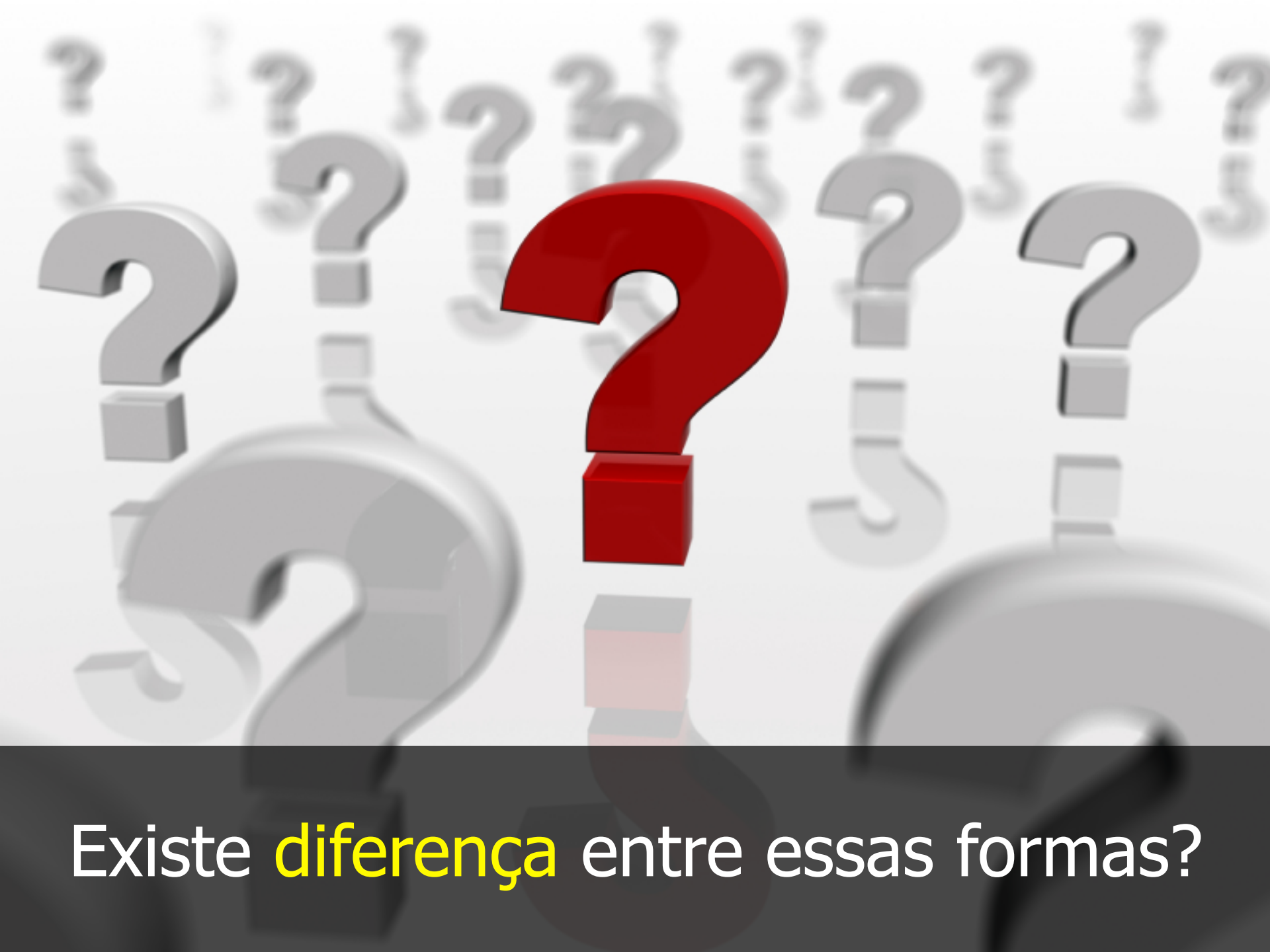

Object

Um objeto é uma coleção dinâmica de propriedades definidas por chaves, que podem ser do tipo String ou Symbol, e valores que podem ser de qualquer tipo de dado

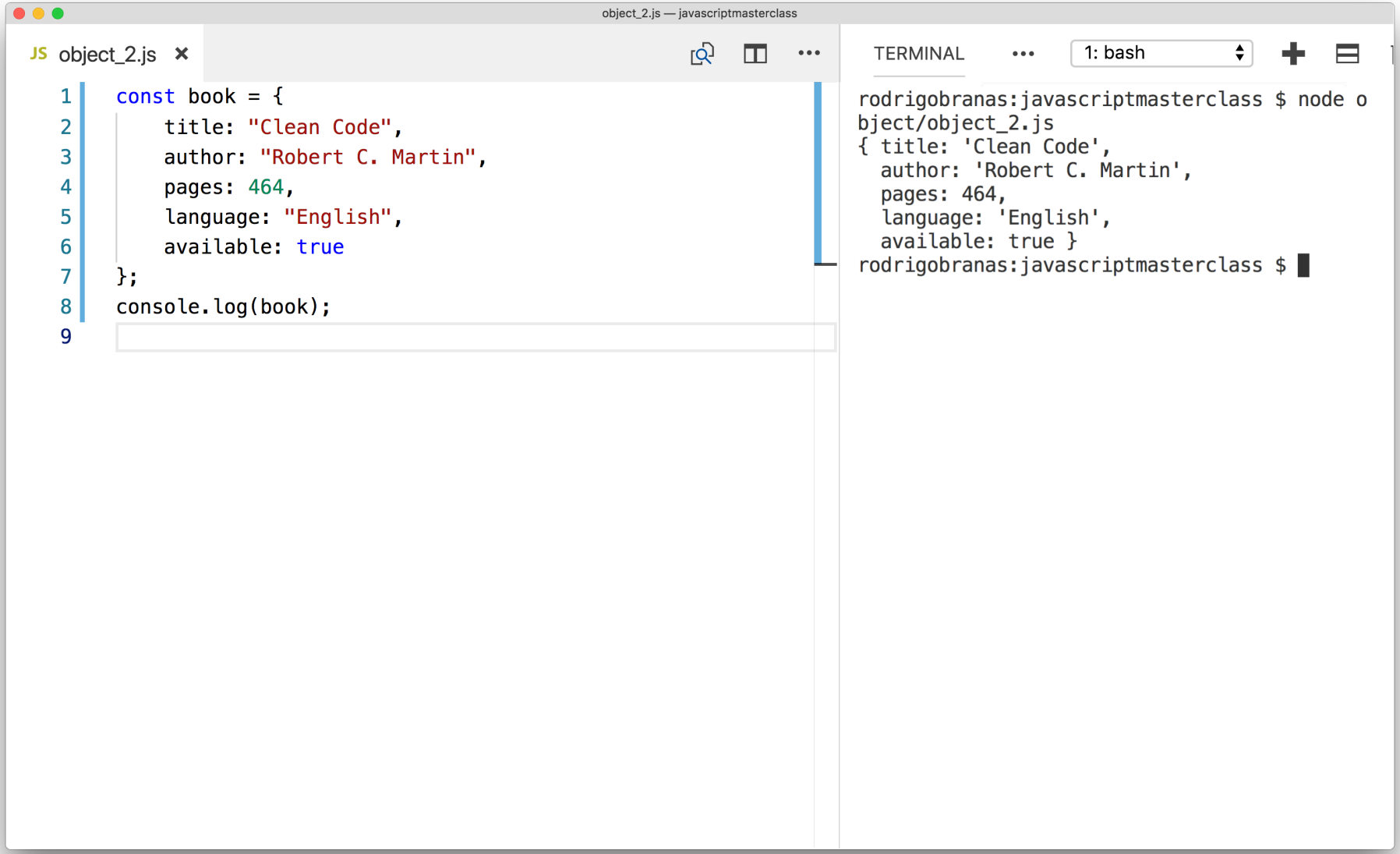
Chave	Valor
title	Clean Code
author	Robert C. Martin
pages	464
language	English
available	true

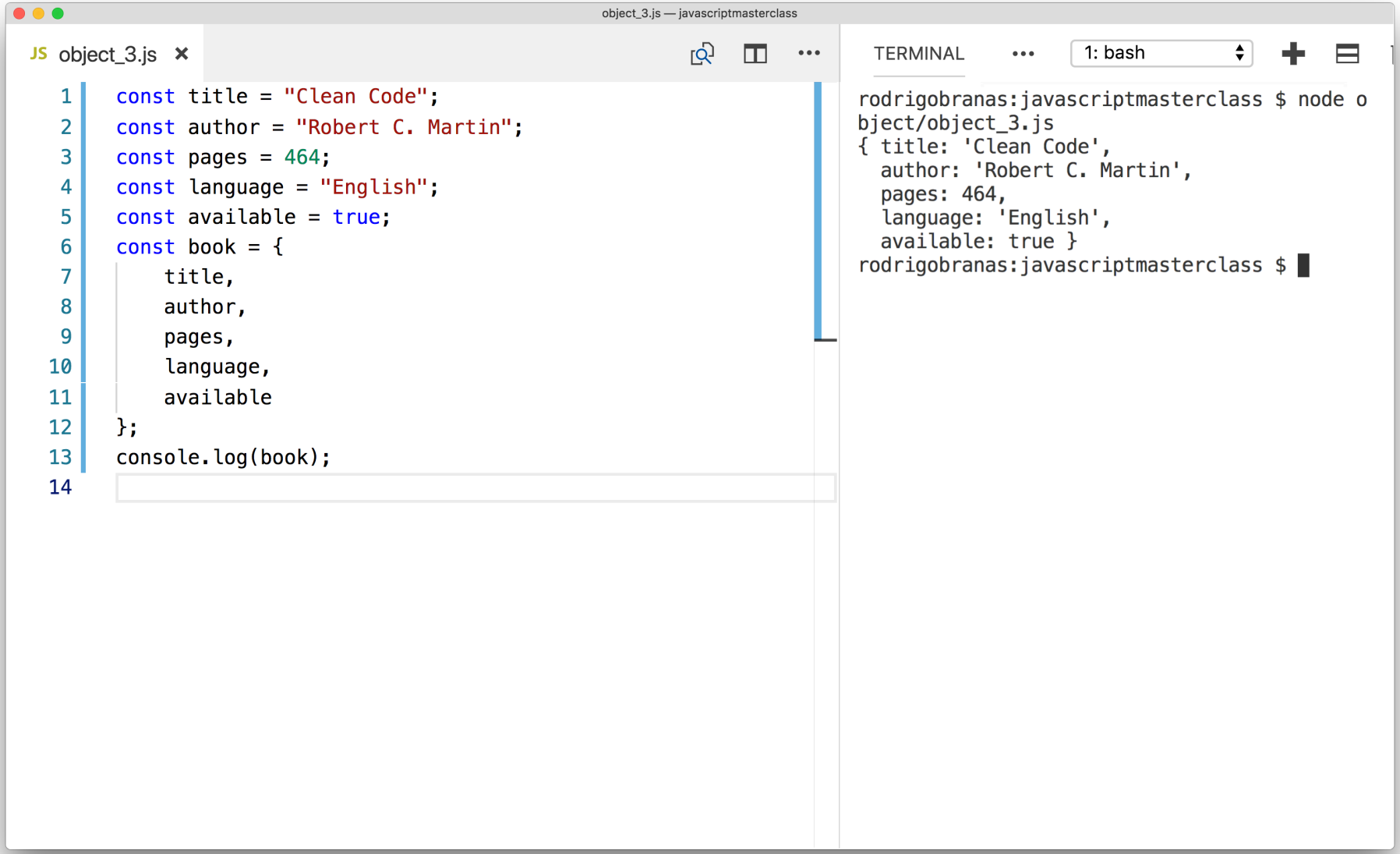
É possível criar objetos de várias formas: pela notação literal, por meio de uma função construtora ou do método create da Object API


object_1.js — javascriptmasterclass
JS object_1.js
{}
new Object();
Object.create(null);
TERMINAL
1: node
rodrigobranas:javascriptmasterclass $ node
> {}
{}
> new Object();
{}
> Object.create(null);
{}
>

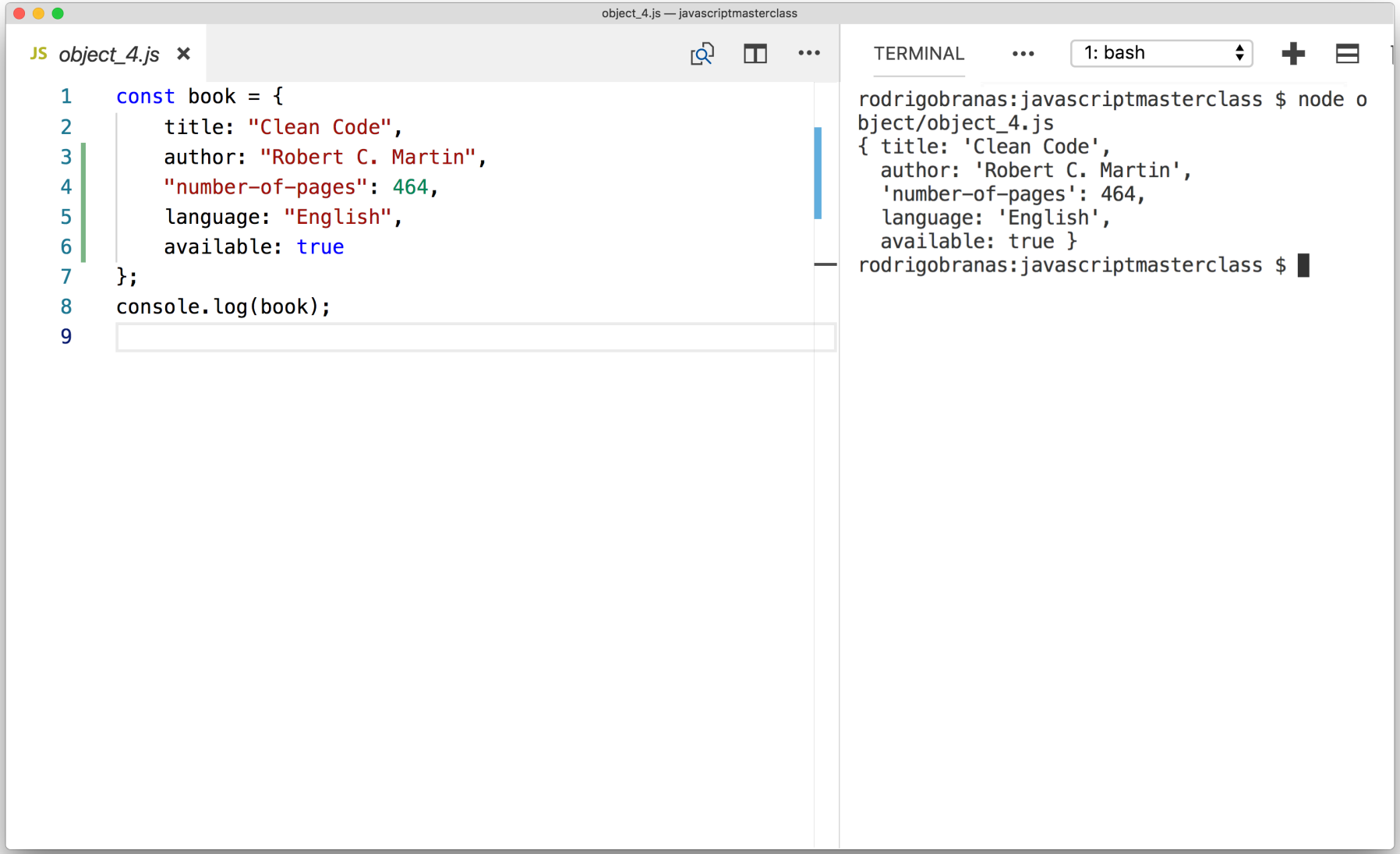

Existe diferença entre essas formas?

Uma das diversas maneiras de atribuir propriedades a um objeto é durante a sua inicialização, pela notação literal

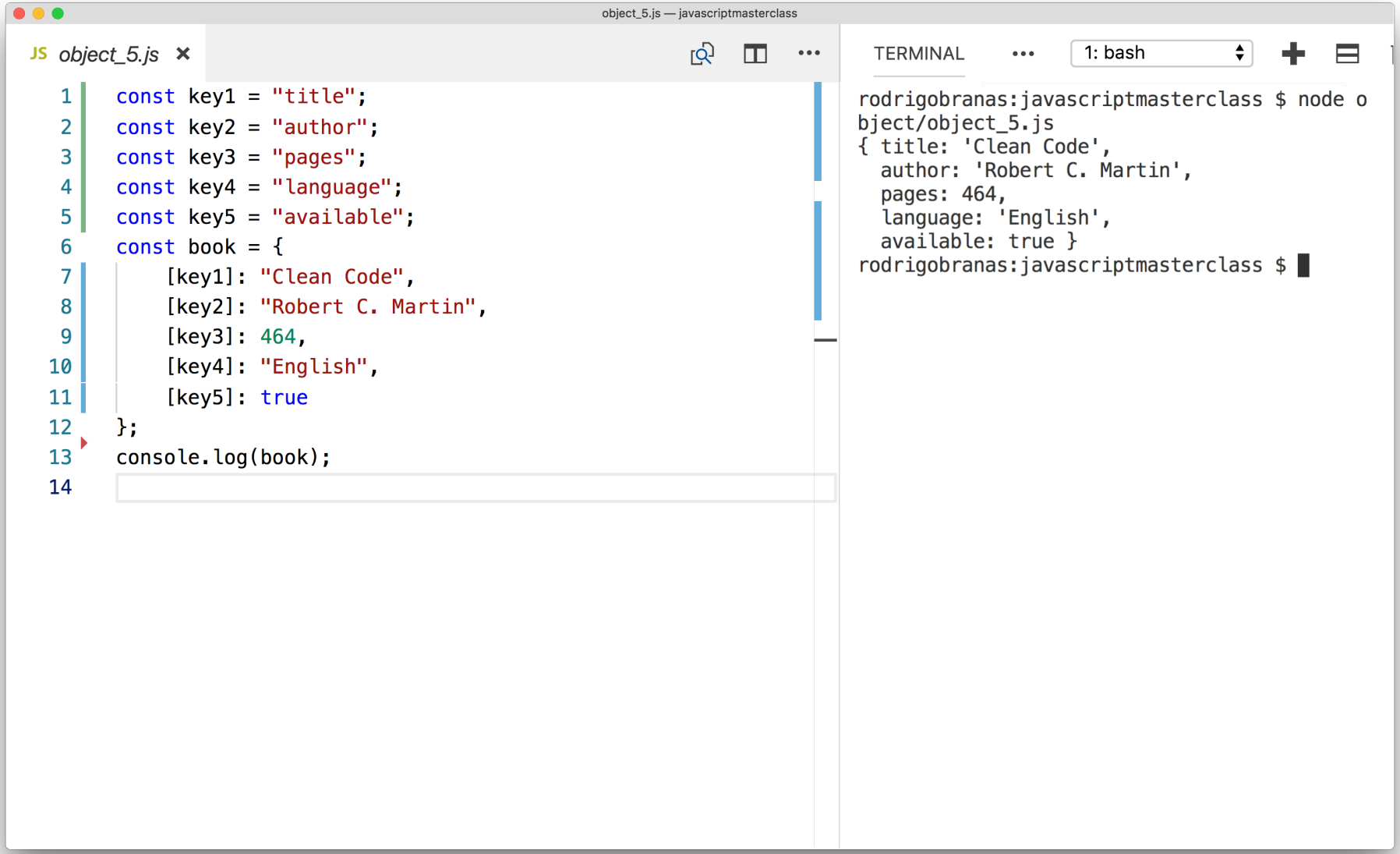

object_2.js — javascriptmasterclass
JS object_2.js
1 const book = {
2 title: "Clean Code",
3 author: "Robert C. Martin",
4 pages: 464,
5 language: "English",
6 available: true
7 };
8 console.log(book);
9
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_2.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
pages: 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $


object_3.js — javascriptmasterclass
object_3.js
1 const title = "Clean Code";
2 const author = "Robert C. Martin";
3 const pages = 464;
4 const language = "English";
5 const available = true;
6 const book = {
7 title,
8 author,
9 pages,
10 language,
11 available
12 };
13 console.log(book);
14
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_3.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
pages: 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $

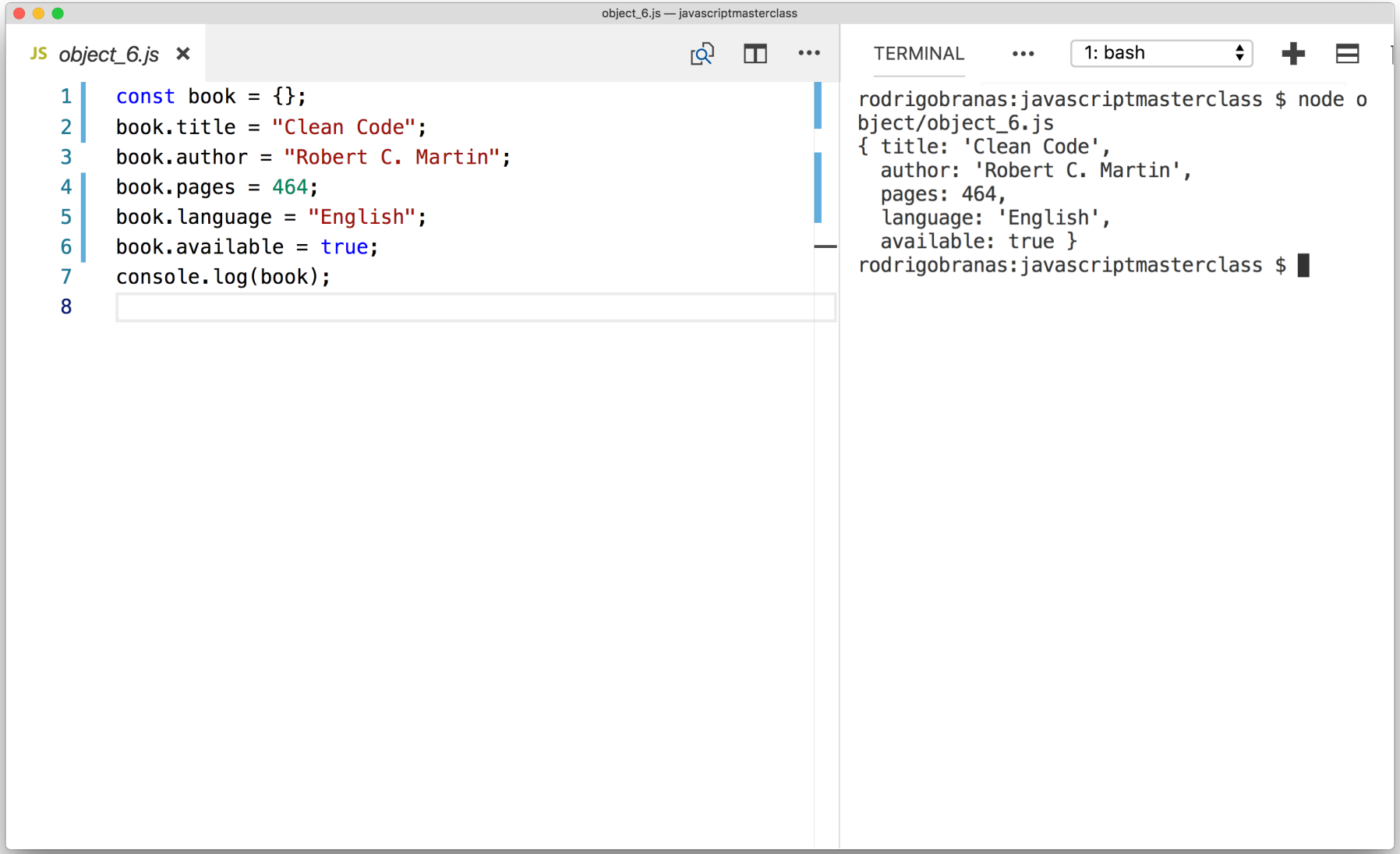
Dependendo da chave é necessário declará-la diretamente como String


object_4.js — javascriptmasterclass
JS object_4.js
1 const book = {
2 title: "Clean Code",
3 author: "Robert C. Martin",
4 "number-of-pages": 464,
5 language: "English",
6 available: true
7 };
8 console.log(book);
9
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_4.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
'number-of-pages': 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $

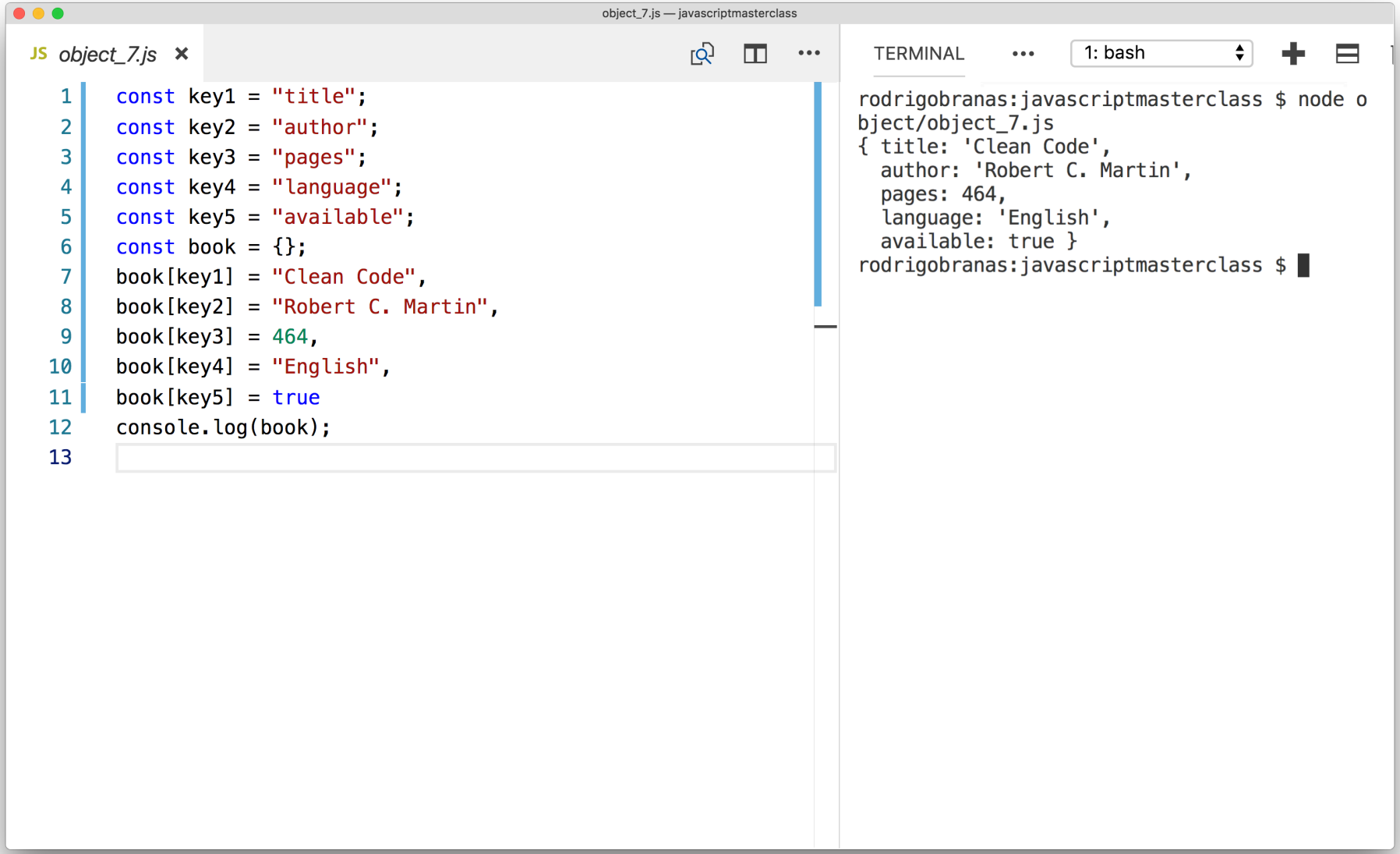
Também é possível computar as chaves em tempo de execução


object_5.js — javascriptmasterclass
JS object_5.js
1 const key1 = "title";
2 const key2 = "author";
3 const key3 = "pages";
4 const key4 = "language";
5 const key5 = "available";
6 const book = {
7 [key1]: "Clean Code",
8 [key2]: "Robert C. Martin",
9 [key3]: 464,
10 [key4]: "English",
11 [key5]: true
12 };
13 console.log(book);
14
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node object/object_5.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
pages: 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $

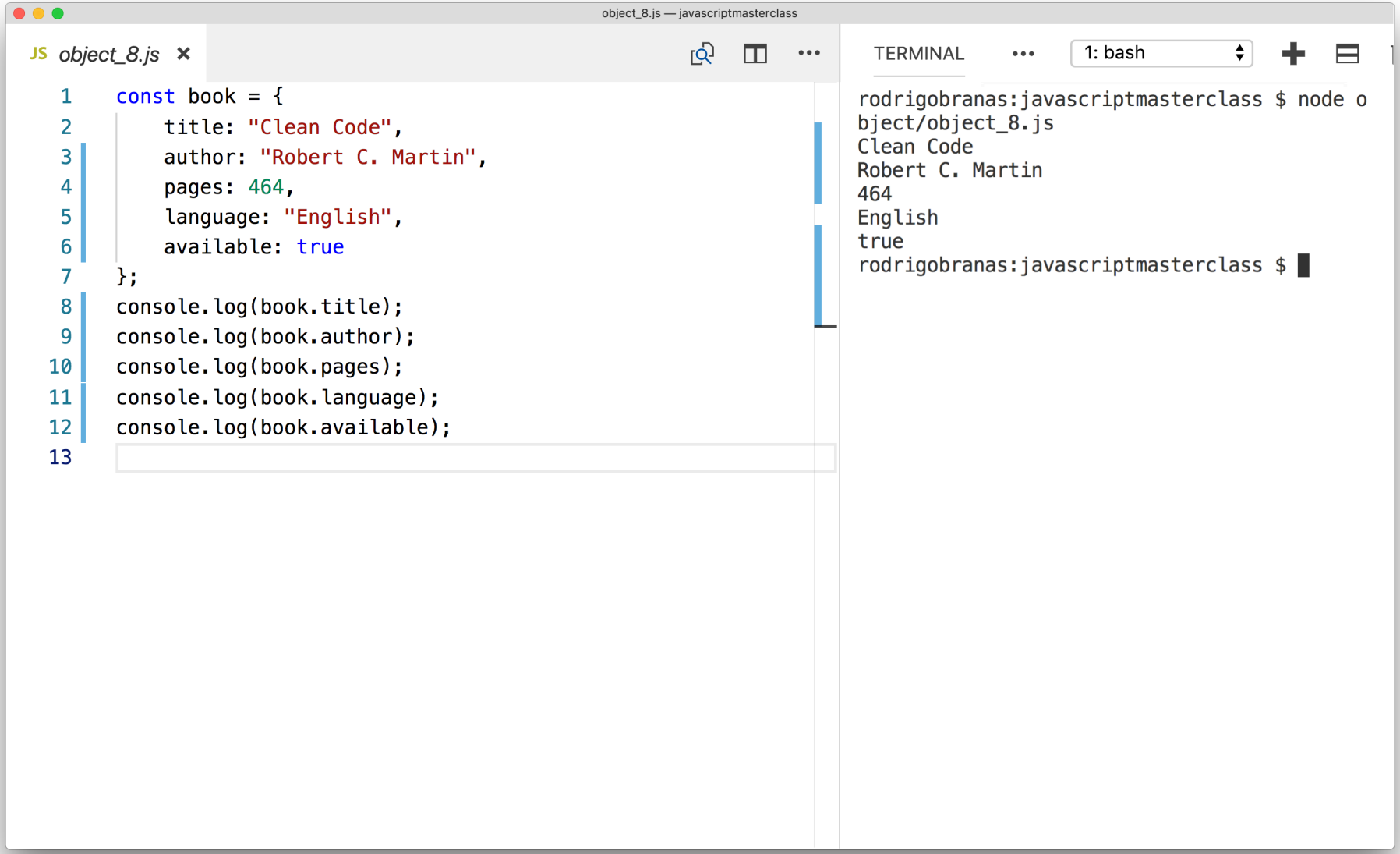
Além da notação literal, é possível atribuir propriedades aos objetos por meio da sua referência


object_6.js — javascriptmasterclass
object_6.js
1 const book = {};
2 book.title = "Clean Code";
3 book.author = "Robert C. Martin";
4 book.pages = 464;
5 book.language = "English";
6 book.available = true;
7 console.log(book);
8
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_6.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
pages: 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $

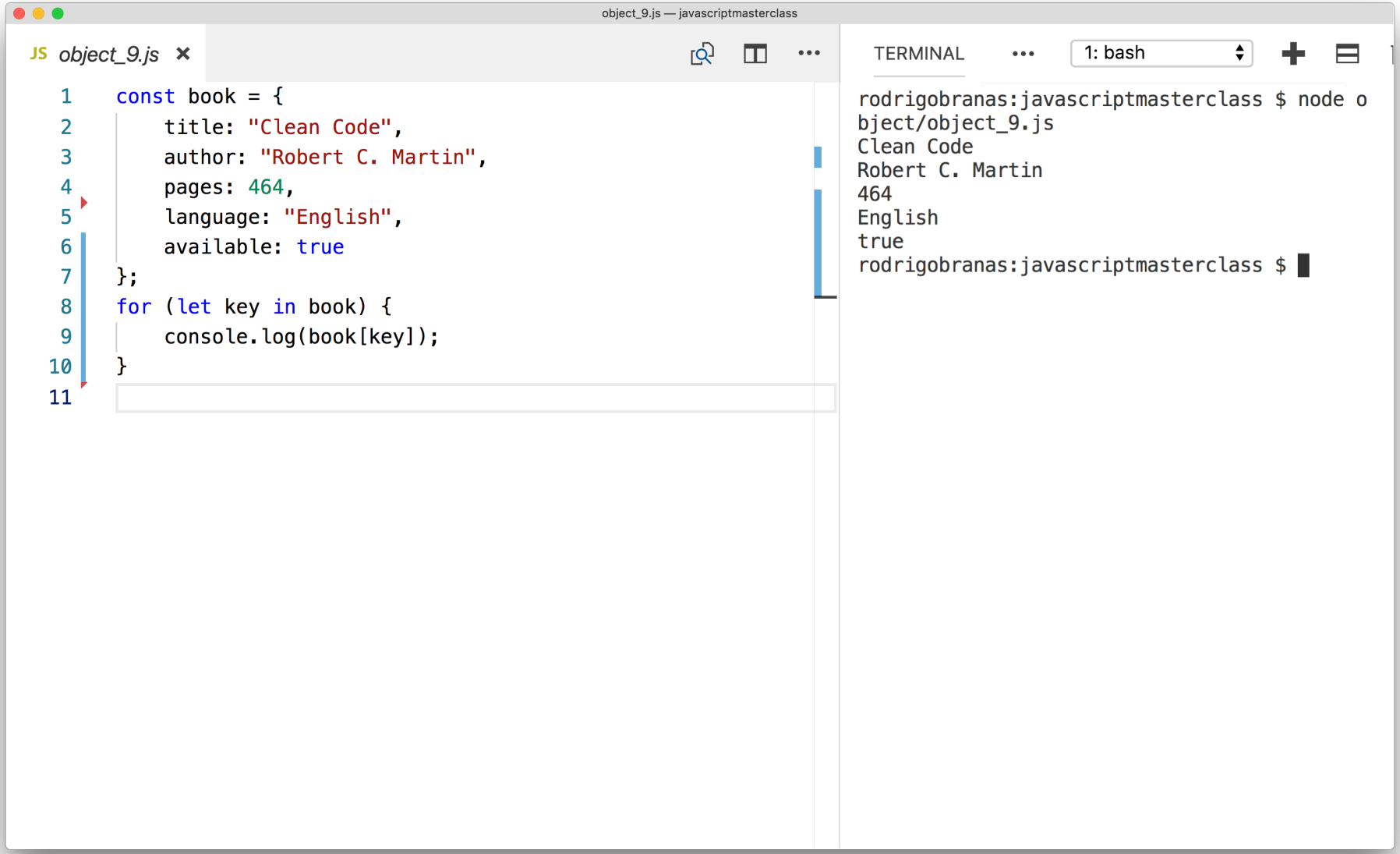
Assim como na notação literal, é possível computar as chaves de um objeto em tempo de execução por meio da sua referência

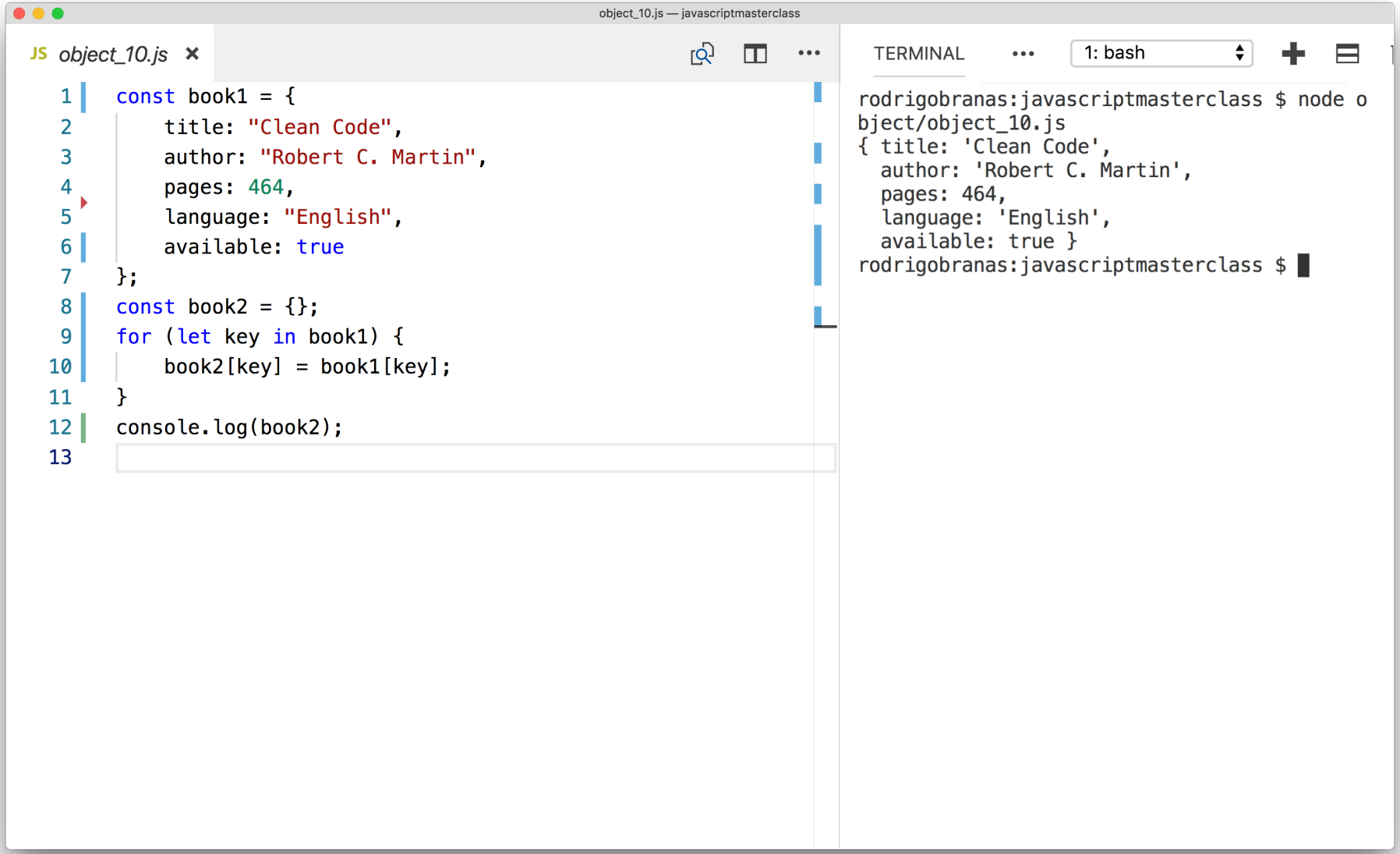

object_7.js — javascriptmasterclass
JS object_7.js
1 const key1 = "title";
2 const key2 = "author";
3 const key3 = "pages";
4 const key4 = "language";
5 const key5 = "available";
6 const book = {};
7 book[key1] = "Clean Code",
8 book[key2] = "Robert C. Martin",
9 book[key3] = 464,
10 book[key4] = "English",
11 book[key5] = true
12 console.log(book);
13
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_7.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
pages: 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $

Cada uma das propriedades de um objeto podem ser consultadas por meio da sua referência, de forma direta


object_8.js — javascriptmasterclass
object_8.js
TERMINAL
1: bash
const book = {
title: "Clean Code",
author: "Robert C. Martin",
pages: 464,
language: "English",
available: true
};
console.log(book.title);
console.log(book.author);
console.log(book.pages);
console.log(book.language);
console.log(book.available);
rodrigobranas:javascriptmasterclass $ node object/object_8.js
Clean Code
Robert C. Martin
464
English
true
rodrigobranas:javascriptmasterclass $

É possível consultar cada uma das propriedades de um objeto por meio da computação das chaves


object_9.js — javascriptmasterclass
JS object_9.js
1 const book = {
2 title: "Clean Code",
3 author: "Robert C. Martin",
4 pages: 464,
5 language: "English",
6 available: true
7 };
8 for (let key in book) {
9 console.log(book[key]);
10 }
11
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_9.js
Clean Code
Robert C. Martin
464
English
true
rodrigobranas:javascriptmasterclass $


object_10.js — javascriptmasterclass
object_10.js
const book1 = {
title: "Clean Code",
author: "Robert C. Martin",
pages: 464,
language: "English",
available: true
};
const book2 = {};
for (let key in book1) {
book2[key] = book1[key];
}
console.log(book2);
TERMINAL
1: bash
rodrigobranas:javascriptmasterclass $ node o
bject/object_10.js
{ title: 'Clean Code',
author: 'Robert C. Martin',
pages: 464,
language: 'English',
available: true }
rodrigobranas:javascriptmasterclass $