

مختصر كتابي: What're Variables?

ما هي المتغيرات؟

What're Variables



المتغير يخزن قيمة لنستخدمها لاحقاً

```
var firstName = "Hello World 1";
```

var	let	const	
✓	✓	✗	تغيير القيمة
الدالة	الكتلة	الكتلة	النطاق

ضع ; في نهاية كل جملة



var



قديمة

let



const



const أولاً ثم let عند الحاجة

مختصر كتابي: What're Variables?

ما هي المتغيرات في JavaScript؟

المتغير (Variable) هو حجر بناء أساسي في JavaScript، ووظيفته ببساطة أن **نخزن فيه قيمة (Value)** لنستخدمها بعد ذلك مرة واثنين وثلاثًا. وهو قريب جدًا من متغيرات CSS التي درسناها في الكورس السابق. نعرّف المتغير بإحدى ثلاث كلمات: `var` أو `let` أو `const`.

كيف نعرّف متغيرًا؟

```
// variable declaration statement
var firstName = "Hello World 1";

// function call statement
console.log(firstName);
```

1. نبدأ بكلمة `var` (اختصار variable).
2. نكتب **اسم المتغير**، وهو الذي نستخدمه لاحقًا لنسترجع القيمة، كأنك تسقي ملقًا لتصل إليه بسرعة.
3. نضع علامة `=` ثم القيمة. النص (String) يوضع بين علامتي تنصيص.
4. ننهي الجملة بفاصلة منقوطة `;`.

الفاصلة المنقوطة (Semicolon)

- تُكتب في آخر كل جملة (Statement)، سواء كانت تعريف متغير أو استدعاء دالة مثل `console.log()`.
- في CSS لو نسيته يفسد الكود. أما في JavaScript فالكود يعمل حتى بدونها، بسبب خاصية **الإدراج التلقائي للفاصلة المنقوطة (Automatic Semicolon Insertion - ASI)**.
- ومع ذلك يُفضّل كتابتها (Good practice): تجعل الكود أسهل في القراءة، وتوضح أين تنتهي كل جملة، وتساعد في حل الأخطاء (Debugging).
- بعض المبرمجين لا يكتبونها أبدًا، والطريقتان صحيحتان، لكننا سنكتبها في هذا الكورس.

الفرق الأول: هل يمكن تغيير القيمة؟

ظلت `var` وحدها منذ ظهور JavaScript حتى عام 2015، حين جاء تحديث كبير أضاف `let` و `const`. الثلاثة تخزن القيمة وتطبعها بنفس الشكل، لكن جرب تغيير القيم:

```
var firstName = "Hello World 1";
let lastName = "Hello World 2";
const position = "Hello World 3";

firstName = "First";
lastName = "Last";
position = "Engineer"; // TypeError: Assignment to constant variable.
```

- `var` و `let` يمكن تغيير قيمتهما بلا مشكلة.
- `const` اختصار **Constant** أي «ثابت»، فقيمتها تبقى كما هي طوال البرنامج، ومحاولة تغييرها تُظهر الخطأ السابق باللون الأحمر.
- رسالة الخطأ تخبرك غالبًا بالمشكلة نفسها وبمكانها بالتحديد.

الفرق الثاني: النطاق (Scope)

لنعرف متغيرين داخل جملة `if` (وهي نوع من الكتل البرمجية Code Block) داخل دالة (Function)، ثم نطبعمها خارج الكتلة:

```
function fn() {
  if (true) {
    var firstName = "Hello World 1";
    let lastName = "Hello World 2";
  }
  console.log(firstName); // Hello World 1
  console.log(lastName); // ReferenceError: lastName is not defined
}

fn();
```

- نطاقها **نطاق الدالة (Function scope)**: تُرى في أي مكان داخل الدالة، حتى خارج كتلة `if`.
- `let` و `const` نطاقهما **نطاق الكتلة (Block scope)**: تُريان فقط داخل الأقواس `{ }` التي عُرِّفتا فيها.
- لاحظ أن الدالة لا تعمل إلا عندما نستدعيها (Call) بكتابة `fn()`.

أيهم أستخدم؟

- ابدأ دائمًا بـ `const`، خاصة لو لم تكن متأكدًا هل ستتغير القيمة أم لا.
- لو احتجت لاحقًا لتغيير القيمة، حوّلها إلى `let`. ولو كنت تعرف من البداية أنها ستتغير، استخدم `let` مباشرة.

- أغلب المبرمجين اليوم يعتبرون `var` قديمة (Outdated)، واستبدلوها بـ `let`.

الخلاصة

- المتغير اسم نخزن فيه قيمة لنستخدمها أكثر من مرة.
- `var` و `let` تتغير قيمتهما، و `const` ثابتة.
- `var` نطاقها الدالة، و `let` و `const` نطاقهما الكتلة.
- القاعدة: `const` أولاً، ثم `let` عند الحاجة، وتجنب `var`.

مصطلحات الدرس

المصطلح	بالإنجليزية	المعنى
المتغير	Variable	اسم نخزن فيه قيمة لنستخدمها لاحقاً
الثابت	Constant (const)	متغير لا يمكن تغيير قيمته
الإدراج التلقائي للفاصلة المنقوطة	ASI	إضافة ; تلقائياً أثناء تنفيذ الكود
النطاق	Scope	المكان الذي يمكن رؤية المتغير فيه
نطاق الدالة	Function scope	يُرى المتغير في الدالة كلها (var)
نطاق الكتلة	Block scope	يُرى المتغير داخل الأقواس فقط (const و let)