

[Welcome home !](#)[Règlement intérieur](#)[Poster une issue](#)[JavaScript](#)[Environnement de dev](#)[👋 Semaine 1](#)[Terminal](#)[Variables](#)[Conditions](#)[🎥 Vidéos](#)[Introduction et utilisation](#)

Les opérateurs logiques

[Boucles](#)[👋 Semaine 2](#)[Manipuler des chaînes de caractères](#)[Fonctions](#)[Tableaux](#)[👋 Semaine 3](#)[Déroulement d'un programme](#)[Les booléens](#)[Les valeurs truthy et falsy](#)[Notion de scope](#)[Conditions d'existence](#)[Conditions imbriquées](#)[👋 Semaine 4](#)[Objets](#)[Requêtes à API](#)[Backend & Strapi](#)[HTML / CSS](#)

Les opérateurs logiques

Parfois, vous aurez besoin de mettre en place une logique plus complexe, nécessitant des regroupements de conditions, ou un enchaînement de celles-ci.

Vous pouvez optimiser vos conditions en utilisant des **opérateurs logiques**. C'est un mot bien pompeux pour désigner la possibilité d'entremêler plusieurs conditions dans un seul `if` ou `else if`. Prenons l'exemple de vérification de mot de passe suivant dans lequel nous voulons des mot de passe qui font entre 6 et 9 caractères :

```
const password = "azerty";

// Les conditions suivantes vérifient que
if (password.length < 5) {
  console.log("Votre mot de passe est trop court");
} else if (password.length > 10) {
  console.log("Votre mot de passe est trop long");
} else {
  console.log("Votre mot de passe est correct");
}
```

Vous pouvez alors utiliser l'**opérateur logique** `||`, qui veut littéralement dire "**OU BIEN**". Le code suivant produit **exactement** le même résultat que le précédent :

```
const password = "azerty";

if (password.length < 5 || password.length > 10) {
  console.log("Votre mot de passe n'est pas valide");
} else {
  console.log("Votre mot de passe est correct");
}
```

De la même façon, vous pouvez utiliser l'opérateur logique `&&`, que l'on peut traduire par "**AINSI**".