Conditions imbriquées

Dans certaines situations, pour gagner un peu en performance (et donc sur la durée d'exécution de votre script), vous ne voudrez tester des conditions que si une condition préalable a été validée. Reprenons l'exemple de vérification de la longueur de mot de passe. Si l'utilisateur n'a pas tapé son password, il n'y a alors pas "besoin" de vérifier la longueur de celui-ci :

```
const password = ""; // l'utilisateur n'a

if (password) {
  if (password.length > 5 && password.leng
  // si la longueur du password est supéri
  console.log("Votre mot de passe est corr
  } else {
  console.log("Votre mot de passe n'est pa
  }
}
```

Si vous exécutez le code ci-dessus, vous constaterez que aucun `console.log` ne s'affiche ! Et pour cause : étant donné que la première condition `if (password)` n'est pas remplie, alors tous le code contenu entre les accolades associées à cette instruction `if` est ignoré. Et ce, même s'il contient d'autre conditions !

> 🧯 Vous aurez sans doute aussi à mettre en place des conditions pour **sécuriser l'exécution de votre code**. Par exemple vérifier qu'une variable n'est pas `undefined` ou `null` avant d'essayer d'exploiter une de ses propriétés ! Observez les exemples ci-dessous 👀