Les conditions

Table des matières

I. Contexte

Durée : 45 mn

Environnement de travail : repl.it

Pré-requis : Les variables, l'objet console

Contexte

Lors de nos développements, nous serons amenés à exécuter des instructions de manière conditionnelle. La condition est une des briques primordiales d'un langage de programmation, nous allons voir comment se matérialisent les choix en JavaScript.

II. La condition if

Objectifs

- Apprendre comment tester des conditions grâce au `if`
- Comprendre le fonctionnement du `if ... else`

Mise en situation

Nous allons voir comment tester des conditions en JavaScript afin de décider s'il faut ou non exécuter différentes instructions.

L'instruction if

L'instruction `if` vérifie si une condition est évaluée à vrai ou à faux, dans le but d'effectuer divers traitements spécifiés ensuite. Chaque instruction `if` est suivie d'accolades qui contiennent les traitements à effectuer dans le cas où la condition testée est évaluée vrai.

La syntaxe est la suivante : `if (condition) { traitement à effectuer si la condition est vraie }`.

```
let firstname = 'Tristan';

if (firstname == 'Tristan') { // On passe dans cette condition
  console.log("ma condition est vraie !");
}

if (firstname == 'Jules') { // On ne passe pas dans cette condition
  console.log("ma condition est vraie !");
}
```

Si des instructions `if` sont imbriquées les unes dans les autres, le programme exécutera le traitement pour chaque condition évaluée vraie.

Remarque

Dans l'exemple précédent, l'opérateur == a été utilisé : attention, il ne faut pas le confondre avec l'opérateur = qui nous sert à affecter une valeur à une variable. L'opérateur == signale que nous souhaitons vérifier l'égalité entre la valeur de gauche et la valeur de droite.

Si la condition donnée n'est pas remplie, il ne se passe rien et le programme exécute directement les instructions après l'accolade fermante du bloc `if`.

```
let firstname = 'Tristan';


if (firstname == 'Tristan') {
  console.log("ma condition est vraie !"); // Exécuté si la condition est remplie
}

console.log("la suite des instructions"); // Exécuté dans tous les cas
```

Il est possible de chaîner plusieurs instructions `if`, ainsi que de les imbriquer les unes dans les autres. Si elles sont chaînées, le programme testera chaque condition l'une après l'autre, et à chaque fois que l'une d'elle sera évaluée à vrai, il exécutera le code placé entre les accolades de l'instruction en question.

```
let age = 24;
let firstname = 'Tristan';
let ville = 'Nantes';


if (age == 24) {
  if (firstname == 'Tristan') {
    if (ville == 'Nantes') {
      // Toutes les conditions sont remplies
    }
  }
}
```

L'instruction if ... else

Il est possible de définir un bloc de code à exécuter dans le cas où la condition ne serait pas remplie, via une instruction `if...else`.

La syntaxe est la suivante : `if (condition) { traitement effectué si la condition est remplie} else { traitement effectué si la condition n'est pas remplie }`

```
let age = 24;

if (age == 24) {
  console.log("L'age est bien 24 !");
} else {
  console.log("L'age est différent de 24");
}
```

Cette instruction permet de couvrir beaucoup plus de cas, puisqu'elle permet de dire quoi faire dans un cas ou dans l'autre. Elle prend en compte les cas où la condition est vraie, ainsi que ceux où elle est fausse, ce qui n'est pas le cas d'un simple `if`.

L'instruction if...else if...else

Il est également possible d'utiliser l'instruction `else if` de manière à couvrir plusieurs cas et à n'exécuter que le bloc de code correspondant à la condition évaluée à vrai. Ce bloc peut s'utiliser plusieurs fois de suite, on parle alors de chaîne. Il est important, si un traitement doit être fait, peu importe le résultat, de terminer la chaîne par un dernier bloc `else` qui s'exécutera dans le cas où aucune des conditions ne serait évaluée à vrai.

```js
let age = 24;

if (age == 20) {
  // âge 20
} else if (age == 22 ) {
  // âge 22
} else if (age == 24 ) {
  // âge 24
} else {
  // L'âge n'est pas 20, 22 ou 24
}
```

Si une des conditions est évaluée à vrai, le programme n'évaluera pas les conditions suivantes. Il est donc important de bien choisir l'ordre dans lequel évaluer les conditions, de manière à obtenir le comportement souhaité.

Syntaxe **À retenir**

- Nous avons vu qu'il était possible de n'exécuter un bloc de code que dans le cas où une condition donnée serait remplie.
- Il est également possible de chaîner et d'imbriquer ces vérifications au besoin, ainsi que de définir un bloc de code à exécuter par défaut, dans le cas où les conditions définies ne seraient pas remplies, grâce au `if...else`.

Complément

Condition if

III. Exercice : Appliquez la notion

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :

[1]

Question 1

[solution n°1 p.15]

Écrivez le code permettant d'afficher un message pour vérifier si un utilisateur a atteint la majorité via `console.log()`, uniquement dans le cas où la variable `age` est supérieure ou égale à 18 ans.

```js
let age = 18;
```

Question 2

[solution n°2 p.15]

Dans le cas où l'utilisateur n'a pas l'âge de la majorité, affichez un message lui signifiant qu'il est mineur.

```js
let age = 18;
```

IV. Les expressions ternaires

Objectif

- Comprendre à quoi servent les expressions ternaires et savoir les utiliser

1 https://repl.it/

Mise en situation

Nous allons voir comment l'opérateur ternaire va nous permettre d'évaluer des conditions de manière plus concise.

? :

L'opérateur ternaire (`? :`) est le seul opérateur à comporter trois opérandes. Une expression ternaire se définit la plupart du temps sur une seule ligne, et vérifie si une condition est vraie ou fausse. Dans un cas comme dans l'autre, un bloc de code sera exécuté par la suite.

La structure est la suivante : `condition à tester ? code à exécuter si vrai : code à exécuter si faux`.

```
let x = 0;
let y = 9;
let isEqual;

// Cette condition...
if (x == y) {
  isEqual = true
} else {
  isEqual = false
}

// ... est équivalente à cette condition ternaire
x == y ? isEqual = true : isEqual = false
```

Les expressions ternaires sont un peu moins claires à la lecture du code, et l'écriture d'une condition ternaire sur plusieurs lignes est encore plus difficile à comprendre. C'est pour cette raison que, même s'il est tout à fait possible d'écrire une ternaire sur plusieurs lignes, il est généralement considéré comme une bonne pratique de les mettre sur une seule ligne et de ne les utiliser que pour exprimer des conditions simples. Il est aussi possible de chaîner plusieurs instructions à exécuter dans la ternaire en les entourant de parenthèses et en les séparant par des virgules.

```
x == y ? (isEqual = true, console.log('égalité entre x et y !')) : isEqual = false
```

Syntaxe **À retenir**

- Nous avons vu que l'opérateur ternaire permettait d'évaluer une condition équivalente à un `if...else` de manière plus concise.

Complément

Expression ternaire

V. Exercice : Appliquez la notion

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :

[1]

1 https://repl.it/

Question

[solution n°3 p.15]

Écrivez le code permettant d'afficher dans un `console.log` l'article "M" ou "Mme" devant le nom de famille, en fonction du genre de la personne. Utilisez la forme d'une condition ternaire.

```
let lastname = 'Brassens';
let genre = 'homme';
let title;
```

Indice :

Il sera nécessaire d'ajouter des parenthèses autour de l'expression ternaire si vous l'utilisez telle quelle dans la concaténation avec le nom de famille.

VI. La condition switch

Objectif

- Comprendre à quoi sert la condition `switch` et comment l'utiliser

Mise en situation

L'instruction `switch` permet d'évaluer une expression donnée et d'exécuter les instructions correspondant au cas qui y est associé.

L'instruction `switch` évalue une expression et effectue le traitement correspondant au résultat obtenu. L'évaluation de l'expression fournie n'est faite qu'une fois. Elle permet d'éviter les `if...else`, très longs dans le cas où beaucoup de possibilités doivent être testées.

S'il y a plusieurs correspondances, la première condition évaluée à vrai est exécutée.

C'est une bonne pratique d'inclure dans l'instruction `switch` un cas par défaut, dans l'hypothèse où aucune des expressions ne seraient évaluées à *true*.

La syntaxe est la suivante :

```
switch (condition) {
  case valeur1:
    // Instructions à exécuter lorsque le résultat de la condition correspond à valeur1
    instructions1;
    break;
  case valeur2:
    // Instructions à exécuter lorsque le résultat de la condition correspond à valeur2
    instructions 2;
    break;
  default:
    // Instructions à exécuter lorsqu'aucune des valeurs ne correspond à la condition à évaluer
    instructions par défaut;
    break;
}

```

Le programme commencera par rechercher une correspondance avec la valeur proposée pour chaque case. S'il n'en trouve aucune, il exécutera le bloc `default`. Les instructions dans le bloc `default` ne sont exécutées que dans le cas où aucune correspondance ne serait faite avec les valeurs des cas définis.

L'instruction `break` à la fin de chaque bloc d'instructions n'est pas obligatoire, mais elle permet de s'assurer qu'une fois qu'une condition a été vérifiée et que le bloc d'instructions correspondant a été exécuté, l'exécution du `switch` lui-même sera court-circuitée. Sans l'instruction `break`, le `switch` continuerait jusqu'à la fin pour trouver d'autres conditions qui pourraient être vérifiées. Avec le `break`, on met fin immédiatement au `switch` après avoir vérifié une condition.

Exemple

```
let fruit = "kiwi";

switch (fruit) {
  case "banane":
    console.log("Vous avez choisi une banane.");
    break;
  case "pomme":
    console.log("Vous avez choisi une pomme.");
    break;
  case "kiwi":
    console.log("Vous avez choisi un kiwi.");
    break;
  case "clémentine":
    console.log("Vous avez choisi une clémentine.");
    break;
  default:
    console.log("Vous n'avez pas choisi un fruit de la liste.");
    break;
}

// Affichera "Vous avez choisi un kiwi."
```

Syntaxe **À retenir**

- Nous avons vu que la condition `switch` permettait de tester de multiples valeurs et de n'exécuter que le bloc de code correspondant à l'expression testée, de manière plus simple et lisible qu'en chaînant de nombreux `if...else`.

Complément

Condition switch

VII. Exercice : Appliquez la notion

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :

[1]

1 https://repl.it/

Question

[solution n°4 p.15]

Écrivez le code permettant d'afficher le nom d'une chaîne de télévision en fonction de son numéro avec une syntaxe `switch`. Un numéro ne correspondant qu'à une seule chaîne, il faudra faire en sorte que le code soit le plus performant possible, en sortant du `switch` immédiatement après avoir trouvé la bonne chaîne. Si le numéro de chaîne n'est pas trouvé, il faudra informer l'utilisateur par un message console.

Indice :

https://www.csa.fr/Informer/PAF-le-paysage-audiovisuel-francais/Les-chaines-de-la-TNT/La-numerotation-des-chaines[1]

- 1 : TF1
- 2 : France 2
- 3 : France 3
- 4 : Canal +
- 5 : France 5
- 6 : M6
- 7 : Arte

VIII. Auto-évaluation

A. Exercice final

Exercice 1

[solution n°5 p.16]

Exercice

L'usage d'une condition ne peut se faire que sur le test d'une chaîne de caractères.

- ○ Vrai
- ○ Faux

Exercice

L'instruction dans les accolades d'une condition est exécutée...

- ○ Tout le temps
- ○ Seulement si la condition est évaluée fausse
- ○ Seulement si la condition est évaluée vraie
- ○ Jamais

Exercice

Comment exprimer de multiples conditions ?

- ○ else if
- ○ if else
- ○ or
- ○ option

Exercice

1 Numérotation de chaînes de télévision - CSA

Quel opérateur arithmétique teste l'égalité entre deux valeurs ?

- ○ =
- ○ equals
- ○ ==
- ○ isSame

Exercice

Exprimer une condition sous la forme ternaire...

- ☐ Est beaucoup plus performant
- ☐ Est souvent moins lisible quand la condition est complexe
- ☐ Est la seule manière d'exécuter du code seulement si la condition n'est pas remplie
- ☐ Est une forme raccourcie d'un if...else

Exercice

Il est possible d'exécuter plusieurs instructions dans une condition ternaire.

- ○ Vrai
- ○ Faux

Exercice

Il est possible de créer une "chaîne" de conditions.

- ○ Vrai
- ○ Faux

Exercice

La syntaxe `switch` est...

- ☐ L'équivalent du if...else if...else
- ☐ Parfois plus performante que le if...else if...else
- ☐ Plus adaptée pour de nombreux tests

Exercice

Le mot-clé dans un `switch` qui sert à sortir prématurément est...

- ○ stop
- ○ halt
- ○ break
- ○ exit

Exercice

Le dernier "case" d'un `switch` est généralement...

- ○ Il n'y a pas particulièrement de dernier switch, on indique seulement les conditions qu'on souhaite tester
- ○ fallback
- ○ default

B. Exercice : Défi

Développeur sur un site de recettes de cuisine, on nous a donné la responsabilité de développer un message qui sera affiché à l'utilisateur quand il se connectera à son profil. Ce message est conditionné par les informations que l'utilisateur a saisies lors de son inscription.

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :

1

Question

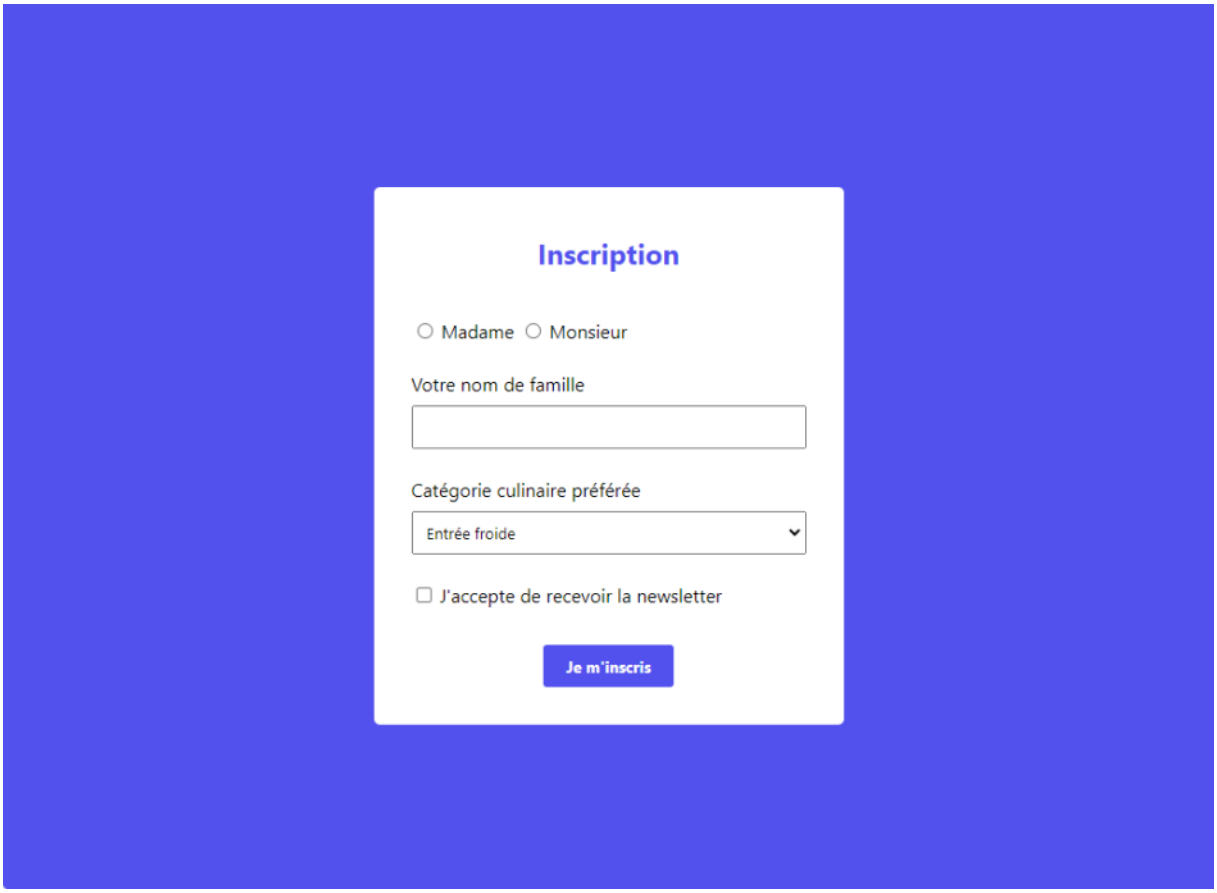
[solution n°6 p.17]

Lors de leur inscription, les utilisateurs ont saisi le formulaire suivant :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="style.css">
  <script src="script.js" defer></script>
  <title>Inscription</title>
</head>
<body>
  <form id="register-form">
    <h1>Inscription</h1>
    <div class="input-group">
      <input type="radio" id="female" name="civility">
      <label for="female">Madame</label>

      <input type="radio" id="male" name="civility">
      <label for="male">Monsieur</label>
    </div>

    <div class="input-group">
      <label for="name">Votre nom de famille</label>
      <input type="text" id="name" name="name">
    </div>

    <div class="input-group">
      <label for="category">Catégorie culinaire préférée</label>
      <select name="category" id="category">
        <option value="cold-starter">Entrée froide</option>
        <option value="soup">Soupe</option>
        <option value="main-course">Plat</option>
        <option value="fruit-dessert">Dessert (fruits)</option>
        <option value="chocolate-dessert">Dessert (chocolat)</option>
      </select>
```

1 https://repl.it/

```
    </div>

    <div class="input-group">
      <input type="checkbox" id="newsletter" name="newsletter">
      <label for="newsletter">J'accepte de recevoir la newsletter</label>
    </div>

    <div class="input-action">
      <button type="submit">Je m'inscris</button>
    </div>
  </form>
</body>
</html>
```



Le message affiché devra être sous la forme suivante :

"Bonjour `[civilité de l'utilisateur]` `[nom de l'utilisateur]`. `["Inscrivez-vous vite à notre newsletter" / "Merci de vous être abonné à notre newsletter"]`, elle permettra de vous envoyer de délicieuses recettes de votre catégorie préférée, `[catégorie culinaire préférée de l'utilisateur]`!"

- Pour conditionner l'affichage de la civilité de l'utilisateur, nous utiliserons une condition ternaire sur la donnée `civility` du formulaire.
- Pour conditionner l'affichage du message "Inscrivez-vous vite à notre newsletter" ou "Merci de vous être abonné à notre newsletter", nous utiliserons une condition `if...else` sur la donnée `newsletter` du formulaire.
- Pour conditionner l'affichage de la catégorie préférée, nous utiliserons un `switch` sur la donnée `category` du formulaire, la valeur par défaut sera "Entrée froide".

- Le message sera affiché dans une alerte JavaScript :

```
alert('Le message à afficher');
```

Vous pouvez utiliser ce code JavaScript de base pour récupérer les données du formulaire HTML.

```
const el = document.getElementById('register-form');
el.addEventListener('submit', (event) => {
  event.preventDefault();

  let civility = event.target.female.checked ? 'female' : 'male';
  let name = event.target.name.value;
  let category = event.target.category.value;
  let newsletter = event.target.newsletter.checked;

  // Les variables déclarées ci-dessus contiennent les données du formulaire
});
```

Voici aussi le code CSS pour le rendu de la page (fichier `style.css`) :

```
* {
  box-sizing: border-box;
  font-family: system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI",
    "Roboto", "Oxygen", "Ubuntu", "Cantarell", "Fira Sans",
    "Droid Sans", "Helvetica Neue", sans-serif;
}

body {
  align-items: center;
  background: rgb(82, 81, 238);
  display: flex;
  height: 100vh;
  justify-content: center;
}

form {
  border-radius: 5px;
  background: rgb(255, 255, 255);
  padding: 0 2rem;
  width: 25rem;
}

h1 {
  color: rgb(82, 81, 238);
  font-size: 1.5rem;
  margin: 2.5rem 0;
  text-align: center;
}

.input-group {
  margin-top: 1.5rem;
}

input[type="text"],
select {
  margin-top: 0.5rem;
  padding: 0.5rem;
  width: 100%;
}

.input-action {
```

```
  align-items: center;
  display: flex;
  justify-content: center;
  margin: 2rem 0;
}

button {
  background: rgb(82, 81, 238);
  border: 0;
  border-radius: 3px;
  color: white;
  font-weight: bold;
  padding: 0.6rem 1.2rem;
}
```

Solutions des exercices

p. 5 Solution n°1

```
let age = 18;

if (age >= 18) {
  console.log('Vous êtes majeur.');
}
```

p. 5 Solution n°2

```
let age = 17;

if (age >= 18) {
  console.log('Vous êtes majeur.');
} else {
  console.log('Vous êtes mineur.');
}
```

p. 7 Solution n°3

```
let lastname = 'Brassens';
let genre = 'homme';
let title;

genre == 'homme'? title = 'M' : title = 'Mme';

console.log(title + “ “ + lastname);
```

p. 9 Solution n°4

```
let message;
let tvStation=1;

switch (tvStation) {
  case 1:
    message = 'TF1';
    break;
  case 2:
    message = 'France 2';
    break;
  case 3:
    message = 'France 3';
    break;
  case 4:
    message = 'Canal +';
    break;
  case 5:
    message = 'France 5';
    break;
  case 6:
    message = 'M6';
    break;
```

```
  case 7:
    message = 'Arte';
    break;
  default:
    message = 'Chaîne inconnue';
}

console.log(message);
```

Exercice p. 9 **Solution n°5**

Exercice

L'usage d'une condition ne peut se faire que sur le test d'une chaîne de caractères.

- ○ Vrai
- ◉ Faux

Exercice

L'instruction dans les accolades d'une condition est exécutée...

- ○ Tout le temps
- ○ Seulement si la condition est évaluée fausse
- ◉ Seulement si la condition est évaluée vraie
- ○ Jamais

Exercice

Comment exprimer de multiples conditions ?

- ◉ else if
- ○ if else
- ○ or
- ○ option

Exercice

Quel opérateur arithmétique teste l'égalité entre deux valeurs ?

- ○ =
- ○ equals
- ◉ ==
- ○ isSame

Exercice

Exprimer une condition sous la forme ternaire...

- [ ] Est beaucoup plus performant
- [x] Est souvent moins lisible quand la condition est complexe
- [ ] Est la seule manière d'exécuter du code seulement si la condition n'est pas remplie
- [x] Est une forme raccourcie d'un if...else

Exercice

Il est possible d'exécuter plusieurs instructions dans une condition ternaire.

- (•) Vrai
  En encadrant ces instructions dans des parenthèses, chaque instruction étant séparée par une virgule.
- ( ) Faux

Exercice

Il est possible de créer une "chaîne" de conditions.

- (•) Vrai
- ( ) Faux

Exercice

La syntaxe `switch` est...

- [x] L'équivalent du if...else if...else
- [x] Parfois plus performante que le if...else if...else
- [x] Plus adaptée pour de nombreux tests

Exercice

Le mot-clé dans un `switch` qui sert à sortir prématurément est...

- ( ) stop
- ( ) halt
- (•) break
- ( ) exit

Exercice

Le dernier "case" d'un `switch` est généralement...

- ( ) Il n'y a pas particulièrement de dernier switch, on indique seulement les conditions qu'on souhaite tester
- ( ) fallback
- (•) default

p. 11 **Solution n°6**

```
const el = document.getElementById('register-form');
el.addEventListener('submit', (event) => {
  event.preventDefault();

  let civility = event.target.female.checked ? 'female' : 'male';
  let name = event.target.name.value;
  let category = event.target.category.value;
  let newsletter = event.target.newsletter.checked;

  // Les variables déclarées ci-dessus contiennent les données du formulaire

  let displayedCivility = civility === 'female' ? 'Mme.' : 'M.';
  let newsletterMessage;

  if (newsletter) {
    newsletterMessage = 'Merci de vous être abonné à notre newsletter';
  } else {
    newsletterMessage = 'Inscrivez-vous vite à notre newsletter';
  }

  let displayedCategory;

  switch (category){
    case 'cold-starter':
      displayedCategory = 'Entrée froide';
      break;
    case 'soup':
      displayedCategory = 'Soupe';
      break;
    case 'main-course':
      displayedCategory = 'Plat';
      break;
    case 'fruit-dessert':
      displayedCategory = 'Dessert (fruits)';
      break;
    case 'chocolate-dessert':
      displayedCategory = 'Dessert (Chocolat)';
      break;
    default:
      displayedCategory = 'Entrée froide'
  }

  alert(`Bonjour ${displayedCivility} ${name}. ${newsletterMessage}, elle permettra de vous envoyer de délicieuses recettes de votre catégorie préférée, ${displayedCategory}!`);
});

```