

Exercice 1 - Carte de visite

Objectif

Créer un premier composant React utilisant le JSX et les expressions JavaScript.

Concepts abordés

- Syntaxe JSX
 - Expressions `{}` dans le JSX
 - Attributs en camelCase (`className`)
 - Un seul élément parent
-

Contexte

Vous devez créer une carte de visite numérique pour vous présenter.

Énoncé

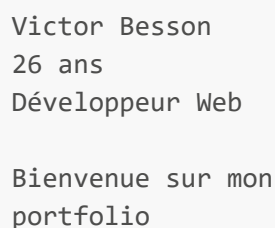
Créer un composant `CarteVisite` qui affiche :

1. Votre prénom et nom
2. Votre âge (calculé à partir de l'année de naissance)
3. Votre métier
4. Un message qui dit "Bienvenue sur mon portfolio"

Contraintes

- Utiliser des variables JavaScript pour les données (pas en dur dans le JSX)
- Calculer l'âge dynamiquement avec l'année courante
- Appliquer une classe CSS `carte-visite` au conteneur principal

Résultat attendu



```
Victor Besson
26 ans
Développeur Web

Bienvenue sur mon
portfolio
```

Indices

► Indice 1 : Structure du composant

```
function CarteVisite() {  
  // Déclarer les variables ici  
  
  return (  
    // JSX ici  
  );  
}  
  
export default CarteVisite;
```

► Indice 2 : Calculer l'âge

```
const anneeNaissance = 1998;  
const age = new Date().getFullYear() - anneeNaissance;
```

► Indice 3 : Insérer une variable dans le JSX

```
<p>{maVariable}</p>  
<p>J'ai {age} ans</p>
```

Points d'attention

Erreur fréquente 1 : Oublier les accolades

```
// ERREUR : affiche littéralement "age"  
<p>age ans</p>  
  
// CORRECT : affiche la valeur de la variable  
<p>{age} ans</p>
```

Erreur fréquente 2 : Utiliser `class` au lieu de `className`

```
// ERREUR : class est un mot réservé en JavaScript  
<div class="carte">  
  
// CORRECT : utiliser className  
<div className="carte">
```

Erreur fréquente 3 : Plusieurs éléments racines

```
// ERREUR : deux éléments au même niveau
return (
  <h1>Titre</h1>
  <p>Texte</p>
)

// CORRECT : un seul parent
return (
  <div>
    <h1>Titre</h1>
    <p>Texte</p>
  </div>
)
```

Correction

```
// src/exercices/ex01-carte-visite/CarteVisite.jsx

function CarteVisite() {
  // =====
  // 1. DÉCLARATION DES VARIABLES
  // On déclare nos données en JavaScript pur, en haut du composant
  // =====
  const prenom = "Victor";
  const nom = "Besson";
  const anneeNaissance = 1998;
  const metier = "Développeur Web";

  // =====
  // 2. CALCUL DYNAMIQUE
  // L'âge est recalculé à chaque rendu du composant
  // Si on ouvre cette page dans 5 ans, l'âge sera correct !
  // =====
  const anneeActuelle = new Date().getFullYear();
  const age = anneeActuelle - anneeNaissance;

  // =====
  // 3. RETOUR DU JSX
  // - Un seul élément parent (la div avec className)
  // - Les {} permettent d'insérer du JavaScript dans le JSX
  // - className au lieu de class (mot réservé en JS)
  // =====
  return (
    <div className="carte-visite">
      {/* Concaténation de variables */}
      <h1>{prenom} {nom}</h1>

      {/* Variable simple */}
      <p>{age} ans</p>
    </div>
  );
}
```

```

    { /* Autre variable */ }
    <p>{metier}</p>

    { /* Texte statique */ }
    <p>Bienvenue sur mon portfolio</p>
  </div>
);
}

// =====
// 4. EXPORT
// export default permet d'importer sans accolades :
// import CarteVisite from './CarteVisite'
// =====
export default CarteVisite;

```

CSS associé (optionnel)

```

/* src/exercices/ex01-carte-visite/CarteVisite.css */

.carte-visite {
  border: 1px solid #ddd;
  border-radius: 8px;
  padding: 1.5rem;
  max-width: 300px;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

.carte-visite h1 {
  margin: 0 0 0.5rem 0;
  font-size: 1.5rem;
  color: #333;
}

.carte-visite p {
  margin: 0.25rem 0;
  color: #666;
}

```

Pour aller plus loin

1. Ajouter un style inline pour mettre le nom en gras
2. Afficher "majeur" ou "mineur" selon l'âge avec un ternaire
3. Ajouter une image de profil

```

// Style inline
<h1 style={{ fontWeight: 'bold', color: 'navy' }}>{prenom} {nom}</h1>

```

```
// Condition
<p>Statut : {age >= 18 ? "Majeur" : "Mineur"}</p>
```