props.enfants

L'objet de chaque composant `props` possède une propriété nommée `children`.

`props.children` renverra tout ce qui se trouve entre les balises JSX d'ouverture et de fermeture d'un composant.

Jusqu'à présent, tous les composants que vous avez vus étaient *des balises à fermeture automatique* , telles que `<MyFunctionComponent />`. Ce n'est pas obligatoire ! Vous pourriez écrire `<MyFunctionComponent></MyFunctionComponent>`, et cela fonctionnerait toujours.

`props.children` renverrait tout entre `<MyFunctionComponent>` et `</MyFunctionComponent>`.

En utilisant `props.children`, nous pouvons séparer le composant externe, `MyFunctionComponent` dans ce cas, du contenu, ce qui le rend flexible et réutilisable.

Regardez **BigButton.js** .

```
import React from 'react';
import LilButton  from './LilButton';

function BigButton(props) {
  console.log(props.children);
  return <button>I am a Big Button.</button>;
}

export default BigButton;

// Example 1 (comme si on étai chez le parent)
<BigButton>
  I am a child of BigButton
</BigButton>


// Example 2
<BigButton>
  <LilButton />
</BigButton>
```

```
// Example 3
<BigButton />
```

- Dans *l'exemple 1,* `<BigButton>` , `props.children` serait égal au texte « Je suis un enfant de BigButton ».
- Dans *l'exemple 2,* `<BigButton>` , `props.children` serait égal à un composant `<LilButton />`.
- Dans *l'exemple 3,* `<BigButton>` , `props.children` serait égal à `undefined`.

Si un composant a plus d'un enfant entre ses balises JSX, `props.children` renverra ces enfants dans un tableau. Cependant, si un composant n'a qu'un seul enfant, `props.children`renverra alors l'unique enfant, *non* encapsulé dans un tableau.

Donner des valeurs par défaut aux accessoires

Jetez un œil au composant `Button`. Notez qu'à la ligne 6, `Button` s'attend à recevoir un accessoire nommé `text`. Le `text` reçu sera affiché à l'intérieur d'un élément `<button>`.

```
import React from 'react';

function Button({text) {

    return (
      <button>{text}</button> //ligne 6
    );
}

export default Button;
```

Et si personne ne passe `text` à `Button`?

Si personne n'en transmet `text`, le `Button` à l'écran de sera vide. Ce serait mieux si `Button` pouvait afficher un message par défaut à la place.

Vous pouvez y parvenir en spécifiant une valeur par défaut pour l'accessoire. Il y a trois façons de procéder !

La première méthode consiste à ajouter une propriété statique `defaultProps` au composant :

```
function Example(props) {
  return <h1>{props.text}</h1>
}

Example.defaultProps = {
  text: 'This is default text',
};
```

Vous pouvez également spécifier la valeur par défaut directement dans la définition de la fonction :

```
function Example({text='This is default text'}) {
   return <h1>{text}</h1>
}
```

Enfin, vous pouvez également définir la valeur par défaut dans le corps de la fonction :

```
function Example(props) {
  const {text = 'This is default text'} = props;
  return <h1>{text}</h1>
}
```

Si aucun texte ne est transmis à `<Example />` , il affichera un message « Ceci est le texte par défaut ».

Si `<Example />` reçoit un texte, il affichera le texte transmis.