JavaScript in JSX with Curly Braces

JSX lets you write HTML-like markup inside a JavaScript file, keeping rendering logic and content in the same place. Sometimes you will want to add a little JavaScript logic or reference a dynamic property inside that markup. In this situation, you can use curly braces in your JSX to open a window to JavaScript.

You will learn

- How to pass strings with quotes
- How to reference a JavaScript variable inside JSX with curly braces
- How to call a JavaScript function inside JSX with curly braces
- How to use a JavaScript object inside JSX with curly braces

Passing strings with quotes

When you want to pass a string attribute to JSX, you put it in single or double quotes:

App.js

⤓ Download ↻ Reset ↗ Fork

```
export default function Avatar() {
  return (
    <img
      className="avatar"
      src="https://i.imgur.com/7vQD0fPs.jpg"
      alt="Gregorio Y. Zara"
    />
  );
}

```

Here, `"https://i.imgur.com/7vQD0fPs.jpg"` and `"Gregorio Y. Zara"` are being passed as strings.

But what if you want to dynamically specify the `src` or `alt` text? You could **use a value from JavaScript by replacing `"` and `"` with `{` and `}`**:

App.js

⤓ Download ↻ Reset ↗ Fork

```
export default function Avatar() {
  const avatar = 'https://i.imgur.com/7vQD0fPs.jpg';
  const description = 'Gregorio Y. Zara';
  return (
    <img
      className="avatar"
      src={avatar}
      alt={description}
    />
```

```
    />
  );
}

```

Notice the difference between `className="avatar"`, which specifies an `"avatar"` CSS class name that makes the image round, and `src={avatar}` that reads the value of the JavaScript variable called `avatar`. That's because curly braces let you work with JavaScript right there in your markup!

Using curly braces: A window into the JavaScript world

JSX is a special way of writing JavaScript. That means it's possible to use JavaScript inside it—with curly braces `{ }`. The example below first declares a name for the scientist, `name`, then embeds it with curly braces inside the `<h1>`:

App.js

Download Reset Fork

```
export default function TodoList() {
  const name = 'Gregorio Y. Zara';
  return (
    <h1>{name}'s To Do List</h1>
  );
}

```

Gregorio Y. Zara's To Do List

Try changing the `name`'s value from `'Gregorio Y. Zara'` to `'Hedy Lamarr'`. See how the list title changes?

Any JavaScript expression will work between curly braces, including function calls like `formatDate()`:

App.js

Download Reset Fork

```
const today = new Date();

function formatDate(date) {
  return new Intl.DateTimeFormat(
    'en-US',
    { weekday: 'long' }
  ).format(date);
}

export default function TodoList() {
  return (
    <h1>To Do List for {formatDate(today)}</h1>
  );
}

```

To Do List for Thursday

Where to use curly braces

You can only use curly braces in two ways inside JSX:

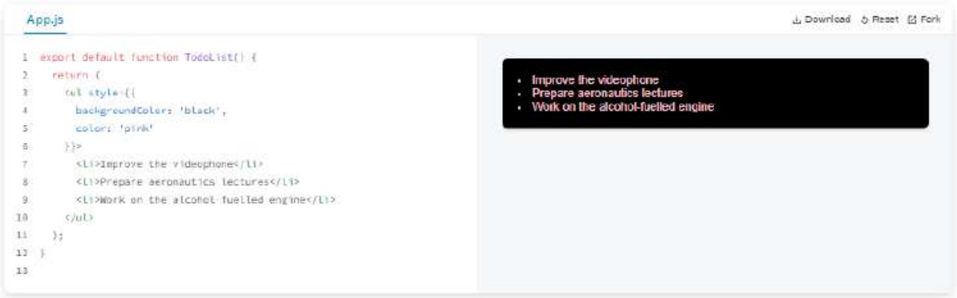
You can only use curly braces in two ways inside JSX:

1. **As text** directly inside a JSX tag: `<h1>{name}'s To Do List</h1>` works, but `<{tag}>Gregorio Y. Zara's To Do List</{tag}>` will not.
2. **As attributes** immediately following the `=` sign: `src={avatar}` will read the `avatar` variable, but `src="{avatar}"` will pass the string `"{avatar}"`.

Using "double curlies": CSS and other objects in JSX

In addition to strings, numbers, and other JavaScript expressions, you can even pass objects in JSX. Objects are also denoted with curly braces, like `{ name: "Hedy Lamarr", inventions: 5 }`. Therefore, to pass a JS object in JSX, you must wrap the object in another pair of curly braces: `person={{ name: "Hedy Lamarr", inventions: 5 }}`.

You may see this with inline CSS styles in JSX. React does not require you to use inline styles (CSS classes work great for most cases). But when you need an inline style, you pass an object to the `style` attribute:



```
export default function TodoList() {
  return (
    <ul style={{
      backgroundColor: 'black',
      color: 'pink'
    }}>
      <li>Improve the videophone</li>
      <li>Prepare aeronautics lectures</li>
      <li>Work on the alcohol-fuelled engine</li>
    </ul>
  );
}
```

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine

Try changing the values of `backgroundColor` and `color`.

You can really see the JavaScript object inside the curly braces when you write it like this:

```
<ul style={
  {
    backgroundColor: 'black',
    color: 'pink'
  }
}>
```

The next time you see `{{` and `}}` in JSX, know that it's nothing more than an object inside the JSX curlies!

Pitfall

Inline `style` properties are written in camelCase. For example, HTML `<ul style="background-color:`

`black">` would be written as `<ul style={{ backgroundColor: 'black' }}>` in your component.

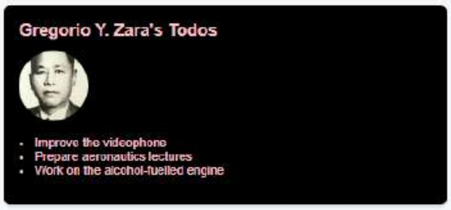
More fun with JavaScript objects and curly braces

You can move several expressions into one object, and reference them in your JSX inside curly braces:

App.js

Download Reset Fork

```
const person = {
  name: 'Gregorio Y. Zara',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src="https://i.imgur.com/7vQD0fPs.jpg"
        alt="Gregorio Y. Zara"
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```



Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine

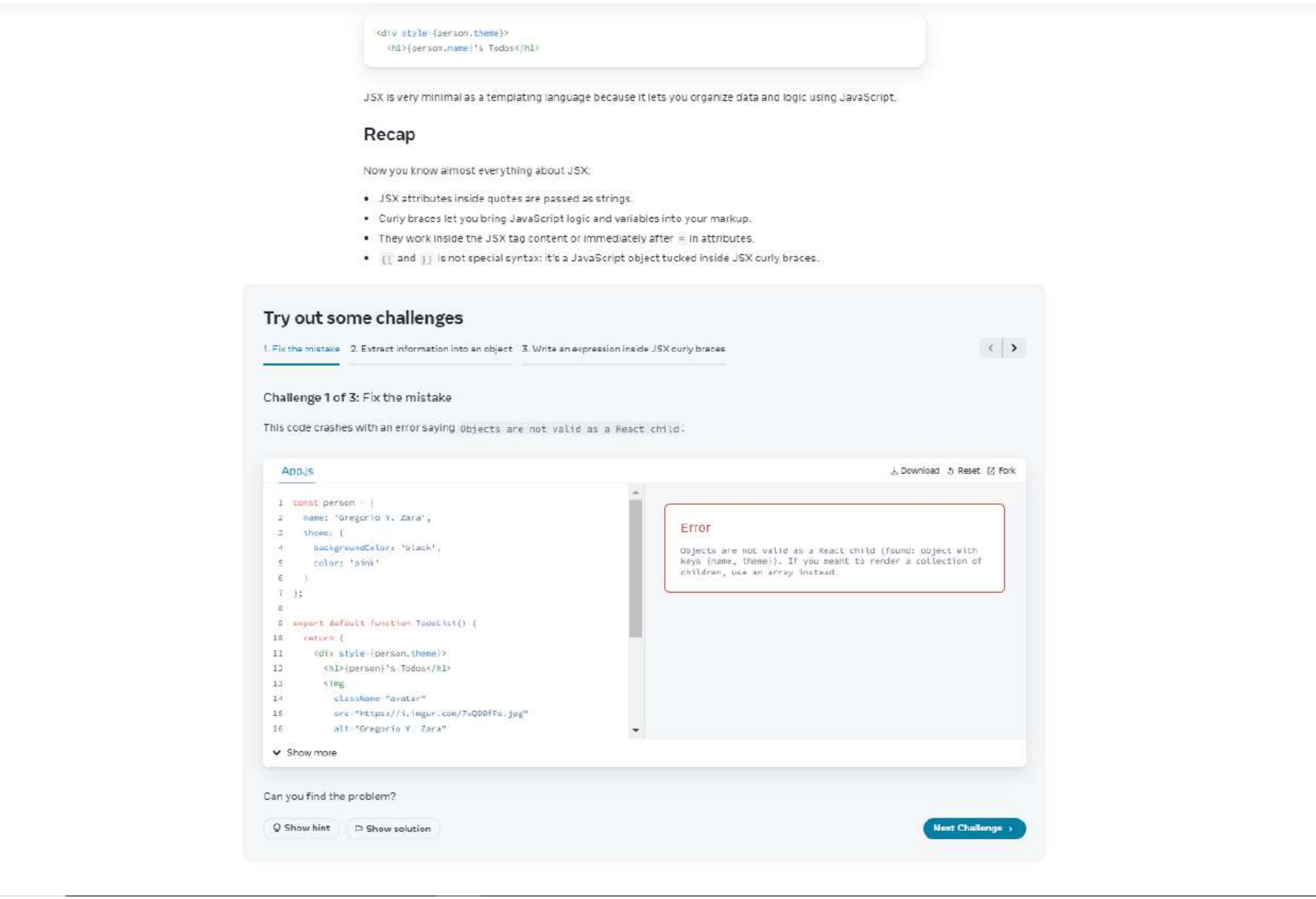
Show less

In this example, the `person` JavaScript object contains a `name` string and a `theme` object:

```
const person = {
  name: 'Gregorio Y. Zara',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};
```

The component can use these values from `person` like so:

```
<div style={person.theme}>
```



```
<div style={person.theme}>
  <h1>{person.name}'s Todos</h1>
```

JSX is very minimal as a templating language because it lets you organize data and logic using JavaScript.

Recap

Now you know almost everything about JSX:

- JSX attributes inside quotes are passed as strings.
- Curly braces let you bring JavaScript logic and variables into your markup.
- They work inside the JSX tag content or immediately after `=` in attributes.
- `{{` and `}}` is not special syntax: it's a JavaScript object tucked inside JSX curly braces.

Try out some challenges

1. Fix the mistake 2. Extract information into an object 3. Write an expression inside JSX curly braces

Challenge 1 of 3: Fix the mistake

This code crashes with an error saying `Objects are not valid as a React child`:

App.js

Download Reset Fork

```
const person = {
  name: 'Gregorio Y. Zara',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person}'s Todos</h1>
      <img
        className="avatar"
        src="https://i.imgur.com/7vQD0fPs.jpg"
        alt="Gregorio Y. Zara"
```

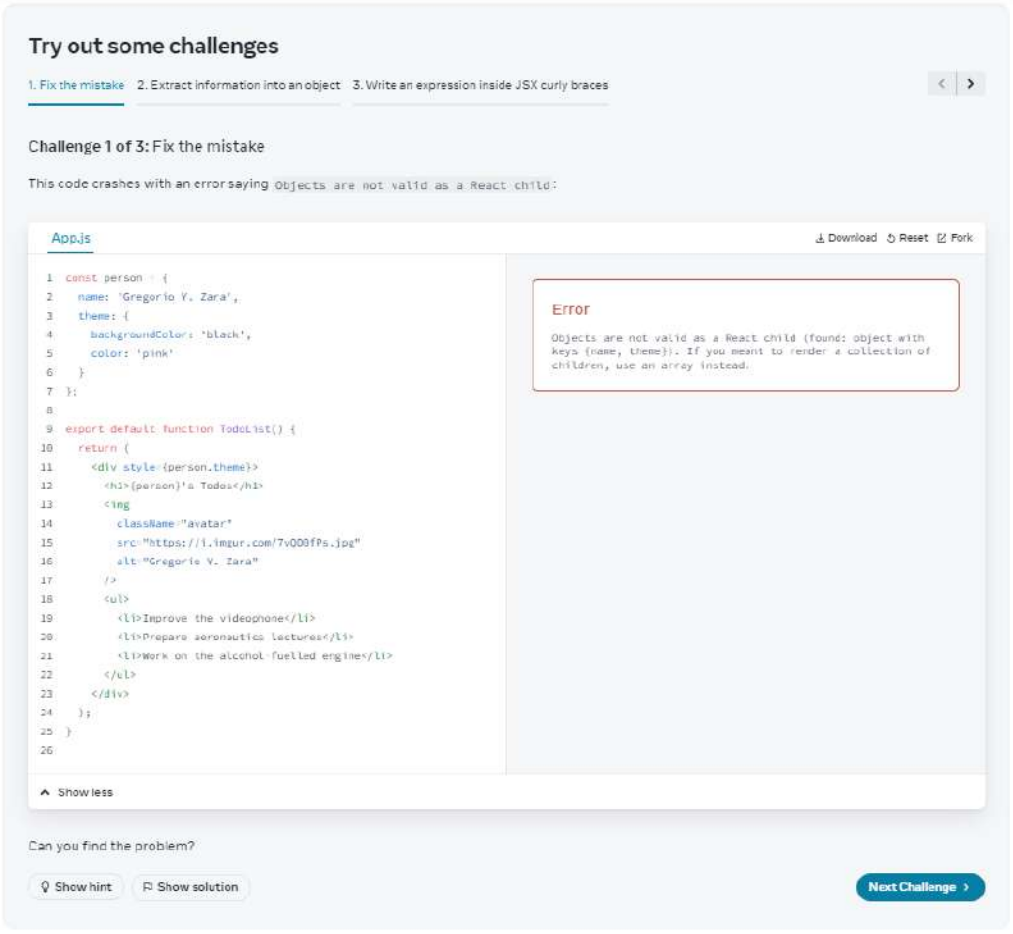
Error

```
Objects are not valid as a React child (found: object with
keys {name, theme}). If you meant to render a collection of
children, use an array instead.
```

Show more

Can you find the problem?

Show hint Show solution Next Challenge



Try out some challenges

1. Fix the mistake 2. Extract information into an object 3. Write an expression inside JSX curly braces

Challenge 1 of 3: Fix the mistake

This code crashes with an error saying `Objects are not valid as a React child`:

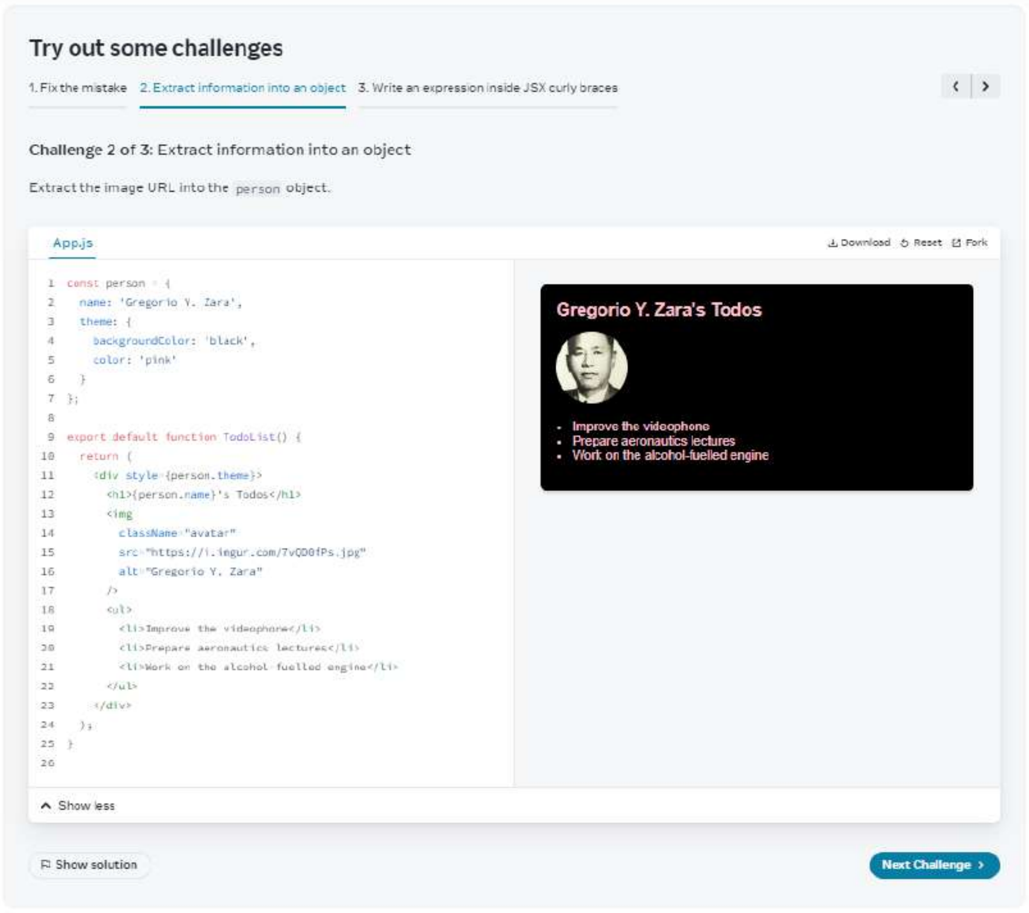
App.js

```
const person = {
  name: 'Gregorio Y. Zara',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person}'s Todos</h1>
      <img
        className="avatar"
        src="https://i.imgur.com/7vQD0fPs.jpg"
        alt="Gregorio Y. Zara"
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```

Error

```
Objects are not valid as a React child (found: object with
keys {name, theme}). If you meant to render a collection of
children, use an array instead.
```

Can you find the problem?



Try out some challenges

1. Fix the mistake 2. Extract information into an object 3. Write an expression inside JSX curly braces

Challenge 2 of 3: Extract information into an object

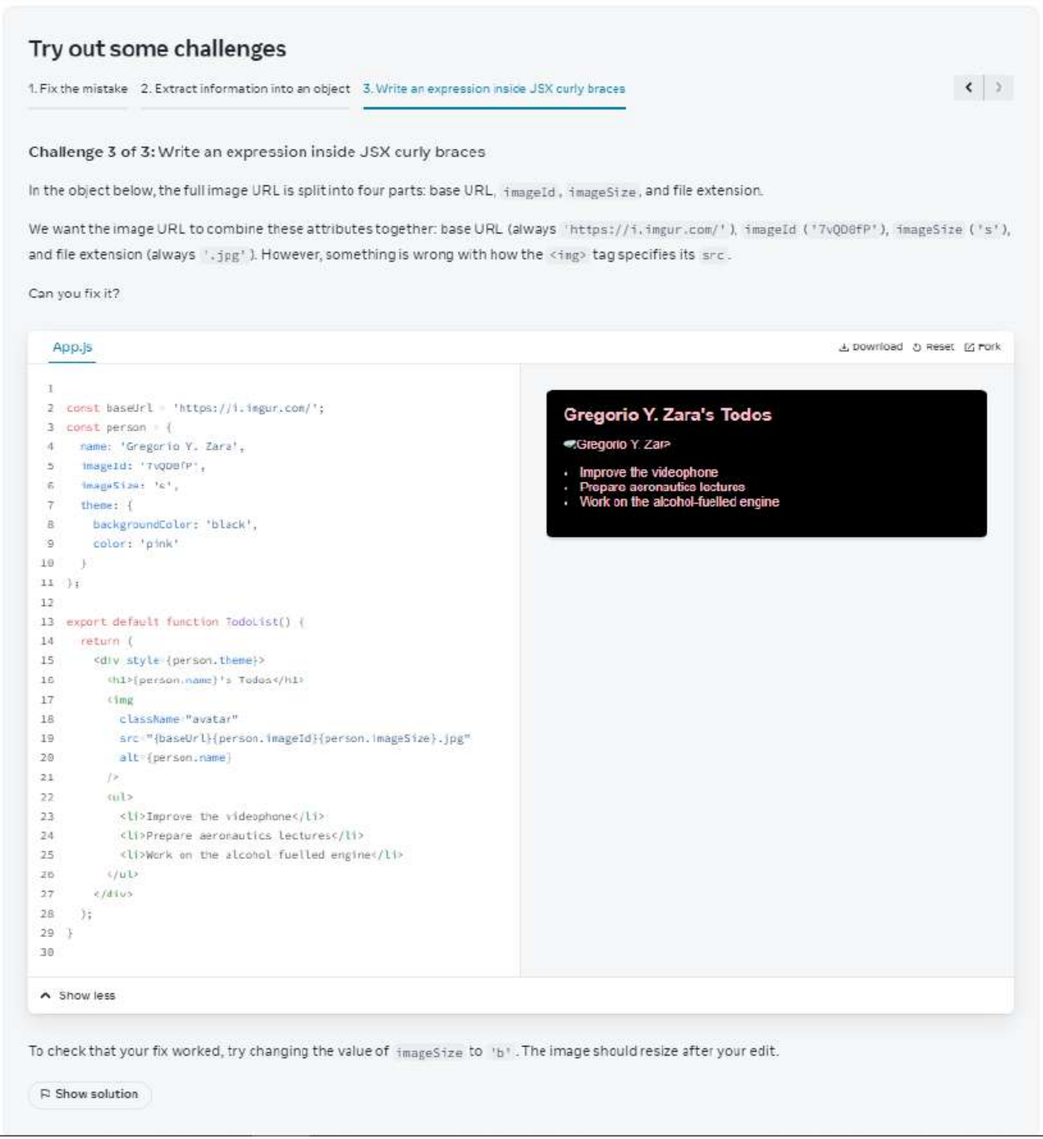
Extract the image URL into the `person` object.

App.js

```
const person = {
  name: 'Gregorio Y. Zara',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src="https://i.imgur.com/7vQD0fPs.jpg"
        alt="Gregorio Y. Zara"
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```

Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine



Try out some challenges

1. Fix the mistake
2. Extract information into an object
3. Write an expression inside JSX curly braces

Challenge 3 of 3: Write an expression inside JSX curly braces

In the object below, the full image URL is split into four parts: base URL, `imageId`, `imageSize`, and file extension.

We want the image URL to combine these attributes together: base URL (always `'https://i.imgur.com/'`), `imageId` (`'7vQD0fP'`), `imageSize` (`'s'`), and file extension (always `'.jpg'`). However, something is wrong with how the `<img>` tag specifies its `src`.

Can you fix it?

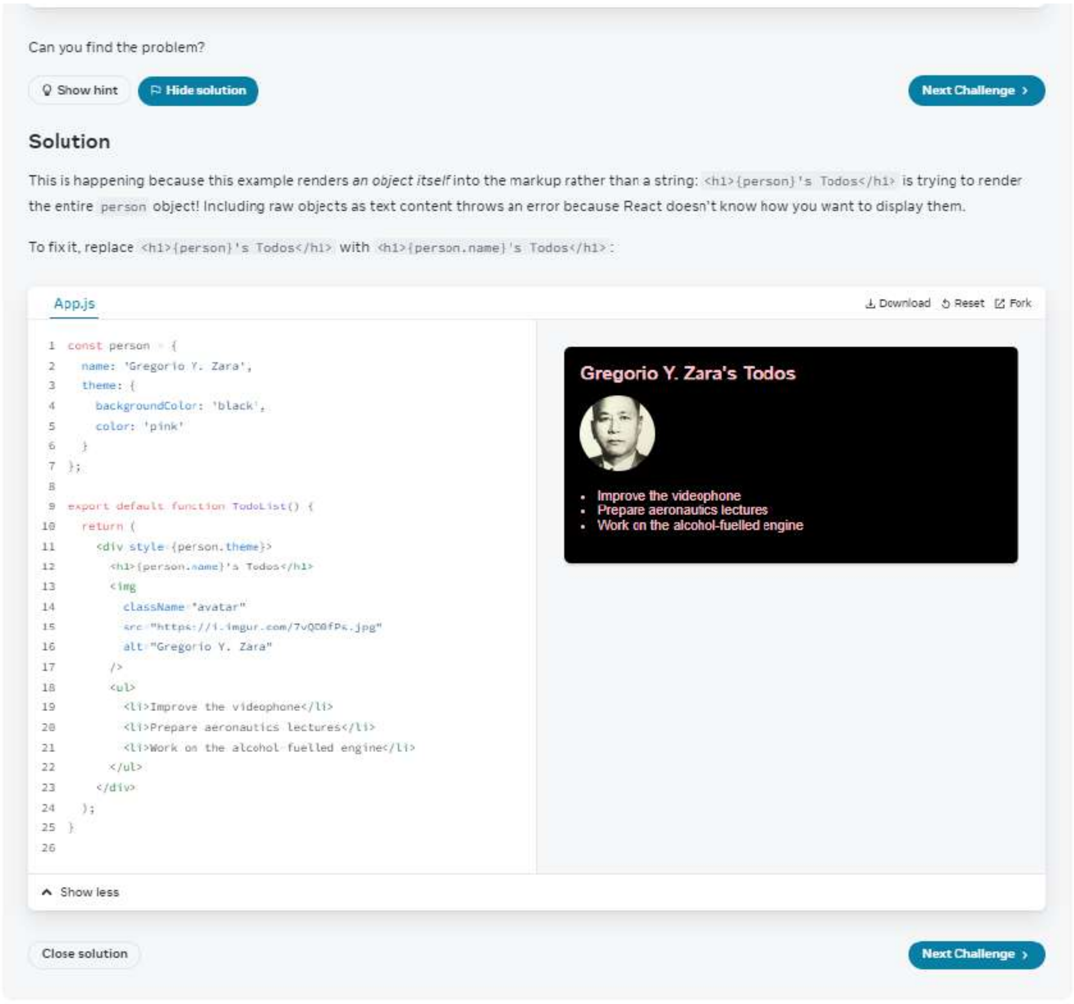
```jsx

const baseUrl = 'https://i.imgur.com/';
const person = {
  name: 'Gregorio Y. Zara',
  imageId: '7vQD0fP',
  imageSize: 's',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src="{baseUrl}{person.imageId}{person.imageSize}.jpg"
        alt={person.name}
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}

```

Gregorio Y. Zara's Todos

Gregorio Y. Zara

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine

To check that your fix worked, try changing the value of `imageSize` to `'b'`. The image should resize after your edit.



Can you find the problem?

Solution

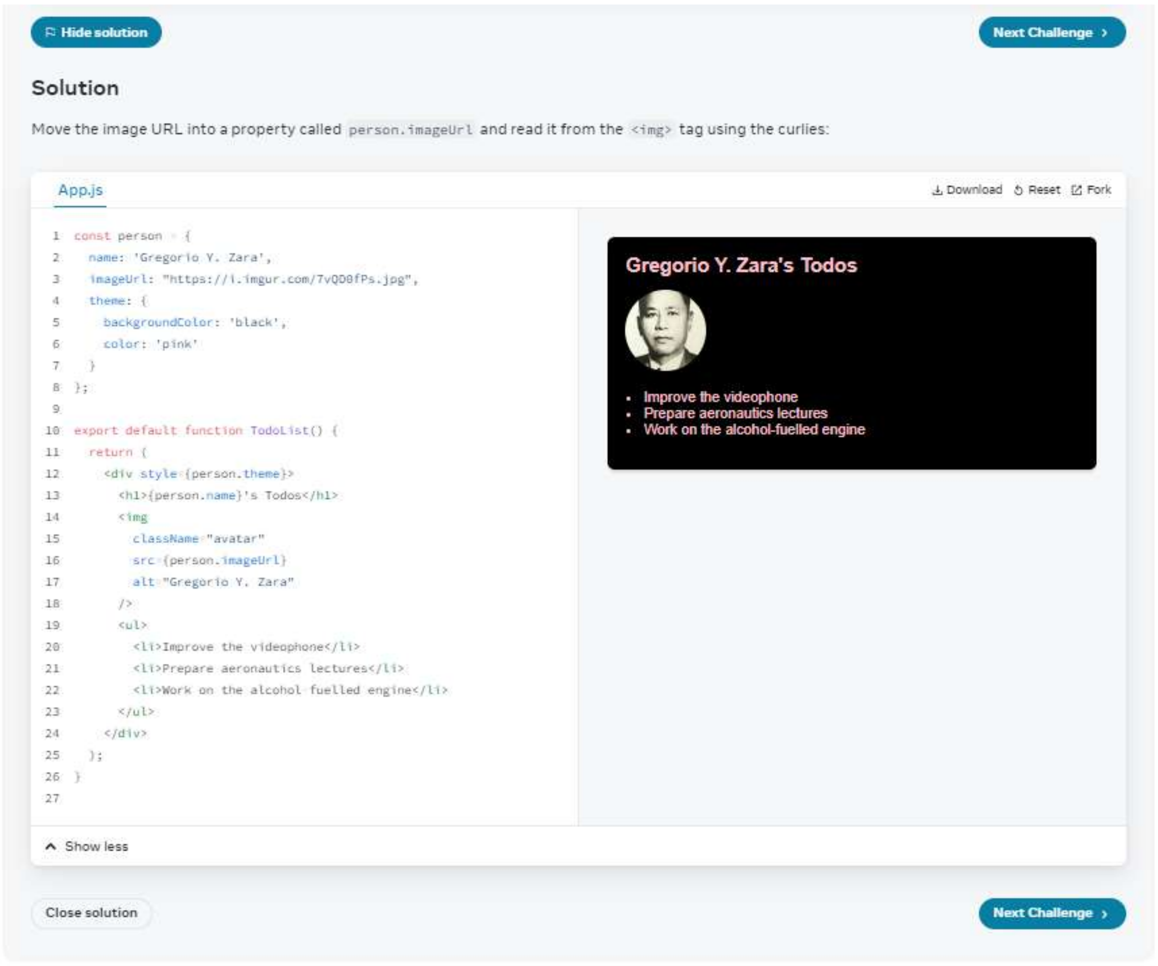
This is happening because this example renders *an object itself* into the markup rather than a string: `<h1>{person}'s Todos</h1>` is trying to render the entire `person` object! Including raw objects as text content throws an error because React doesn't know how you want to display them.

To fix it, replace `<h1>{person}'s Todos</h1>` with `<h1>{person.name}'s Todos</h1>`:

```
const person = {
  name: 'Gregorio Y. Zara',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src="https://i.imgur.com/7vQD0fPs.jpg"
        alt="Gregorio Y. Zara"
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```

Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine



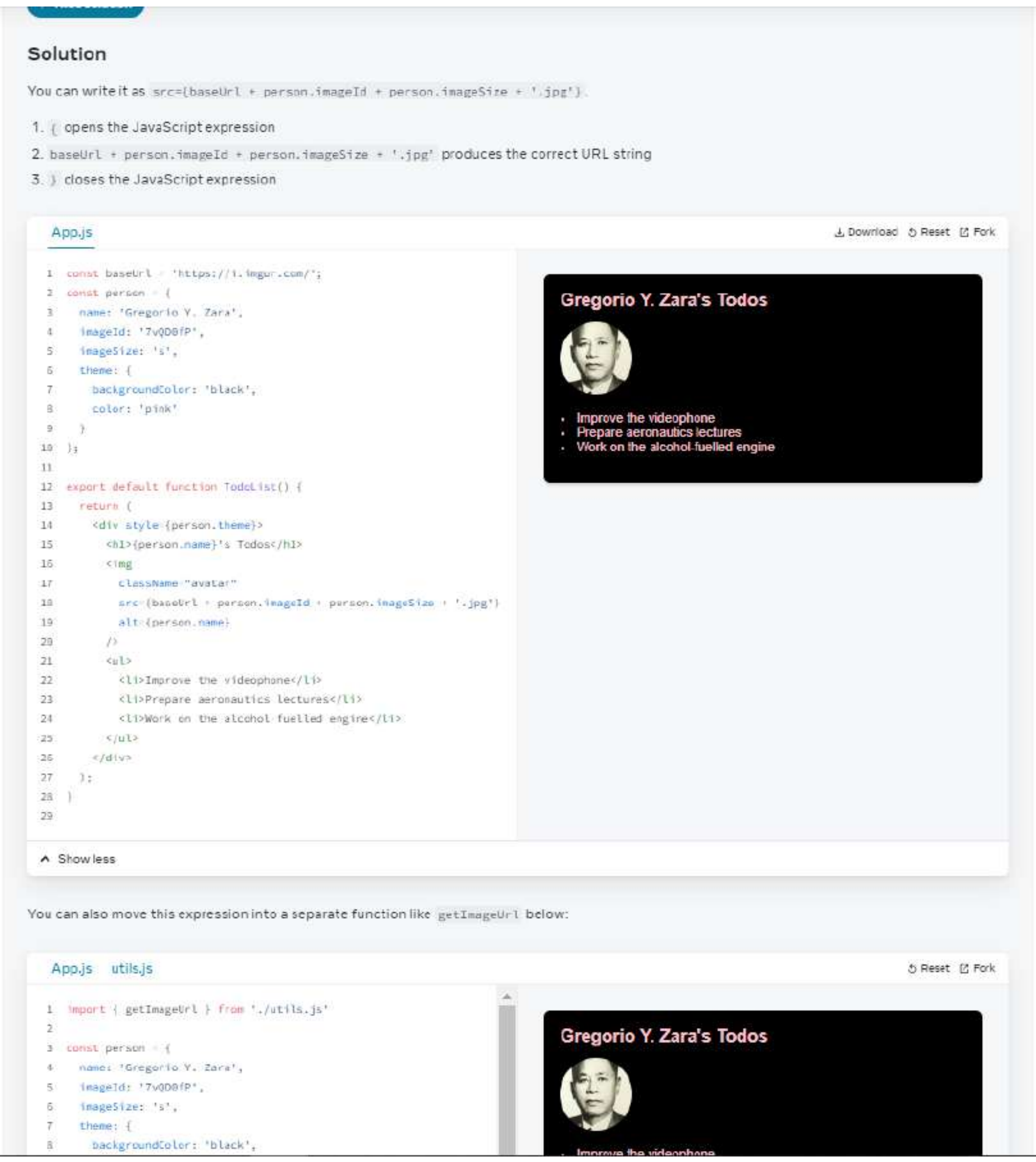
Solution

Move the image URL into a property called `person.imageUrl` and read it from the `<img>` tag using the curlies:

```js
const person = {
  name: 'Gregorio Y. Zara',
  imageUrl: "https://i.imgur.com/7vQD0fPs.jpg",
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src={person.imageUrl}
        alt="Gregorio Y. Zara"
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```

Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine



Solution

You can write it as `src={baseUrl + person.imageId + person.imageSize + '.jpg'}`.

1. `{` opens the JavaScript expression
2. `baseUrl + person.imageId + person.imageSize + '.jpg'` produces the correct URL string
3. `}` closes the JavaScript expression

```
const baseUrl = 'https://i.imgur.com/';
const person = {
  name: 'Gregorio Y. Zara',
  imageId: '7vQD0fP',
  imageSize: 's',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src={baseUrl + person.imageId + person.imageSize + '.jpg'}
        alt={person.name}
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```

Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine

You can also move this expression into a separate function like `getImageUrl` below:

```
import { getImageUrl } from './utils.js'

const person = {
  name: 'Gregorio Y. Zara',
  imageId: '7vQD0fP',
  imageSize: 's',
  theme: {
    backgroundColor: 'black',
```

Gregorio Y. Zara's Todos

- Improve the videophone

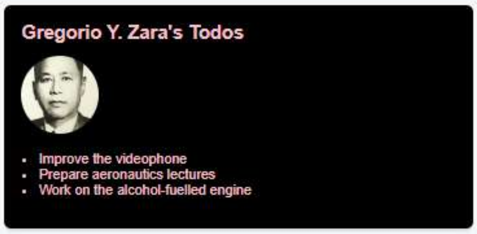
Show less

You can also move this expression into a separate function like `getImageUrl` below:

App.js utils.js

Reset Fork

```
import { getImageUrl } from './utils.js'

const person = {
  name: 'Gregorio Y. Zara',
  imageId: '7vQD0fP',
  imageSize: 's',
  theme: {
    backgroundColor: 'black',
    color: 'pink'
  }
};

export default function TodoList() {
  return (
    <div style={person.theme}>
      <h1>{person.name}'s Todos</h1>
      <img
        className="avatar"
        src={getImageUrl(person)}
        alt={person.name}
      />
      <ul>
        <li>Improve the videophone</li>
        <li>Prepare aeronautics lectures</li>
        <li>Work on the alcohol-fuelled engine</li>
      </ul>
    </div>
  );
}
```



Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine

Show less

Variables and functions can help you keep the markup simple!

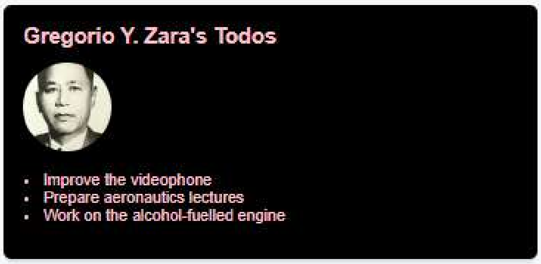
Close solution

You can also move this expression into a separate function like `getImageUrl` below:

App.js utils.js

Reset Fork

```
export function getImageUrl(person) {
  return (
    'https://i.imgur.com/' +
    person.imageId +
    person.imageSize +
    '.jpg'
  );
}

```



Gregorio Y. Zara's Todos

- Improve the videophone
- Prepare aeronautics lectures
- Work on the alcohol-fuelled engine

Variables and functions can help you keep the markup simple!

Close solution

PREVIOUS
Writing Markup with JSX

NEXT
Passing Props to a Component