TodoItem Component

In this lesson, we create a component to show one ToDo item.

WE'LL COVER THE FOLLOWING

TodoItem

Based on several hooks we created, we can now create the TodoItem component.

```
// ./components/TodoItem.js

import React from 'react';

import { useQuery } from '../hooks/useQuery';
import { useDeleteTodo } from '../hooks/useDeleteTodo';
import { useToggleTodo } from '../hooks/useToggleTodo';
import { useFlasher } from '../utils';

const renderHighlight = (title, query) => {
  if (!query) return title;
  const index = title.indexOf(query);
  if (index === -1) return title;
  return (
    <React.Fragment>
      {title.slice(0, index)}
      <b>{query}</b>
      {title.slice(index + query.length)}
    </React.Fragment>
  );
};

const TodoItem = ({ id, title, completed }) => {
```

```
  const { getQuery } = useQuery();
  const deleteTodo = useDeleteTodo();
  const toggleTodo = useToggleTodo();
  return (
    <li ref={useFlasher()}>
      <input
        type="checkbox"
        checked={!!completed}
        onChange={() => toggleTodo(id)}
      />
      <span
        style={{
          textDecoration: completed ? 'line-through' : 'none',
        }}
      >
        {completed ? title : renderHighlight(title, getQuery())}
      </span>
      <button onClick={() => deleteTodo(id)}>Delete</button>
    </li>
  );
};

export default React.memo(TodoItem);
```

We wrap this component with React.memo when exporting. This allows eliminating re-renders if this item is not changed.

Next

In the next lesson, we create a component `NewTodo` to create a new ToDo item.