three examples demonstrating the use of `useEffect` in React:

1. **Fetching Data from an API**:

```jsx
import React, { useState, useEffect } from 'react';

function DataFetching() {
  const [data, setData] = useState(null);

  useEffect(() => {
    const fetchData = async () => {
      try {
        const response = await fetch('https://api.example.com/data');
        const result = await response.json();
        setData(result);
      } catch (error) {
        console.error('Error fetching data:', error);
      }
    };

    fetchData();
  }, []);

  return (
    <div>
      {data ? (
        <p>Data: {JSON.stringify(data)}</p>
      ) : (
        <p>Loading...</p>
      )}
    </div>
  );
}

export default DataFetching;
```

2. **Setting Up a Timer**:

```jsx
import React, { useState, useEffect } from 'react';

function Timer() {
```

```
  const [seconds, setSeconds] = useState(0);

  useEffect(() => {
    const intervalId = setInterval(() => {
      setSeconds(prevSeconds => prevSeconds + 1);
    }, 1000);

    // Cleanup function to clear the interval when component unmounts
    return () => clearInterval(intervalId);
  }, []); // Empty dependency array to run effect only once

  return (
    <div>
      <p>Timer: {seconds} seconds</p>
    </div>
  );
}

export default Timer;
```

3. **Subscribing to Window Resize Events**:

```jsx
import React, { useState, useEffect } from 'react';

function WindowResizeListener() {
  const [windowSize, setWindowSize] = useState({
    width: window.innerWidth,
    height: window.innerHeight
  });

  useEffect(() => {
    const handleResize = () => {
      setWindowSize({
        width: window.innerWidth,
        height: window.innerHeight
      });
    };

    window.addEventListener('resize', handleResize);

    // Cleanup function to remove event listener when component unmounts
    return () => {
```

```
      window.removeEventListener('resize', handleResize);
    };
  }, []); // Empty dependency array to run effect only once

  return (
    <div>
      <p>Window Width: {windowSize.width}px</p>
      <p>Window Height: {windowSize.height}px</p>
    </div>
  );
}

export default WindowResizeListener;
```

These examples demonstrate different scenarios where `useEffect` is useful, such as fetching data, setting up timers, and subscribing to events. Each example uses `useEffect` to perform specific side effects in a React functional component.