July 2020

Phase2

Declarative & Manageable State Management with XState

Hello.

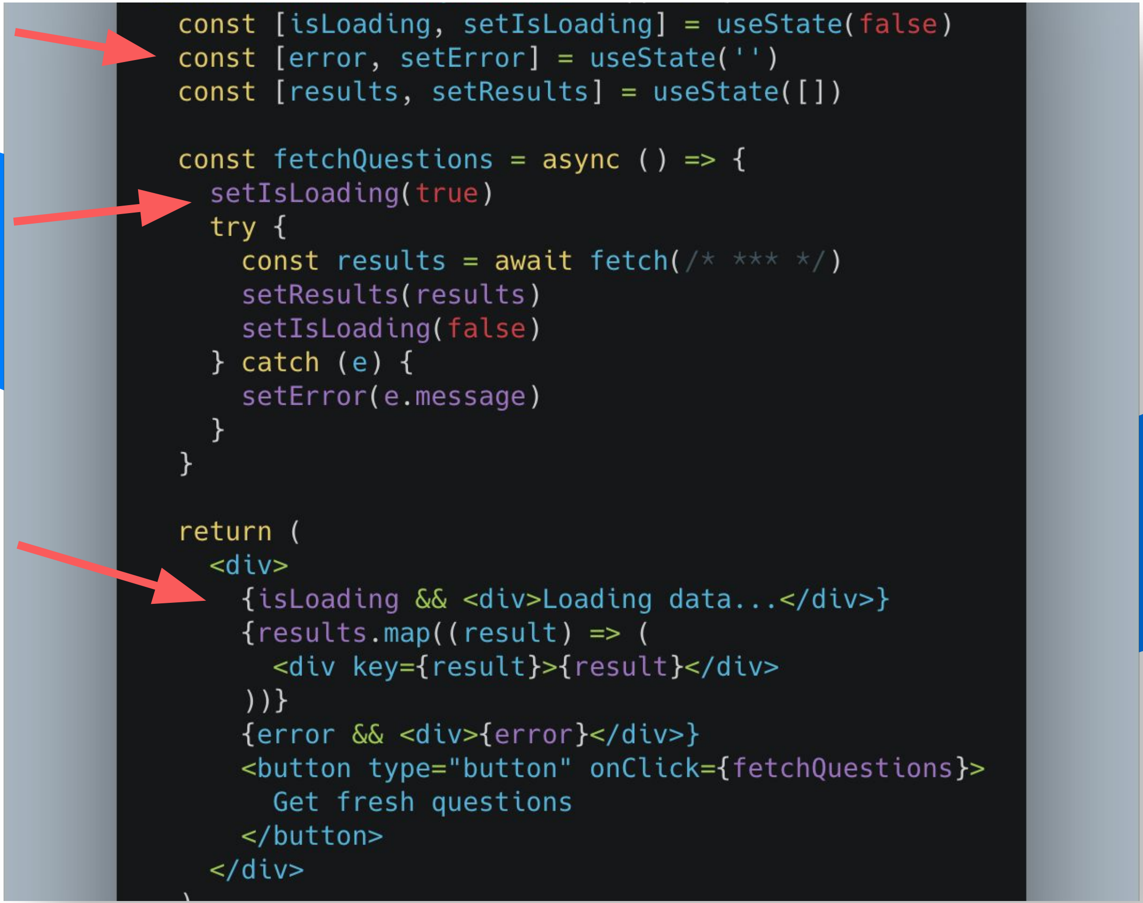
Daniel Lemay

Senior Developer

@dslemay

State Management Pitfalls

Sample fetch without state machines

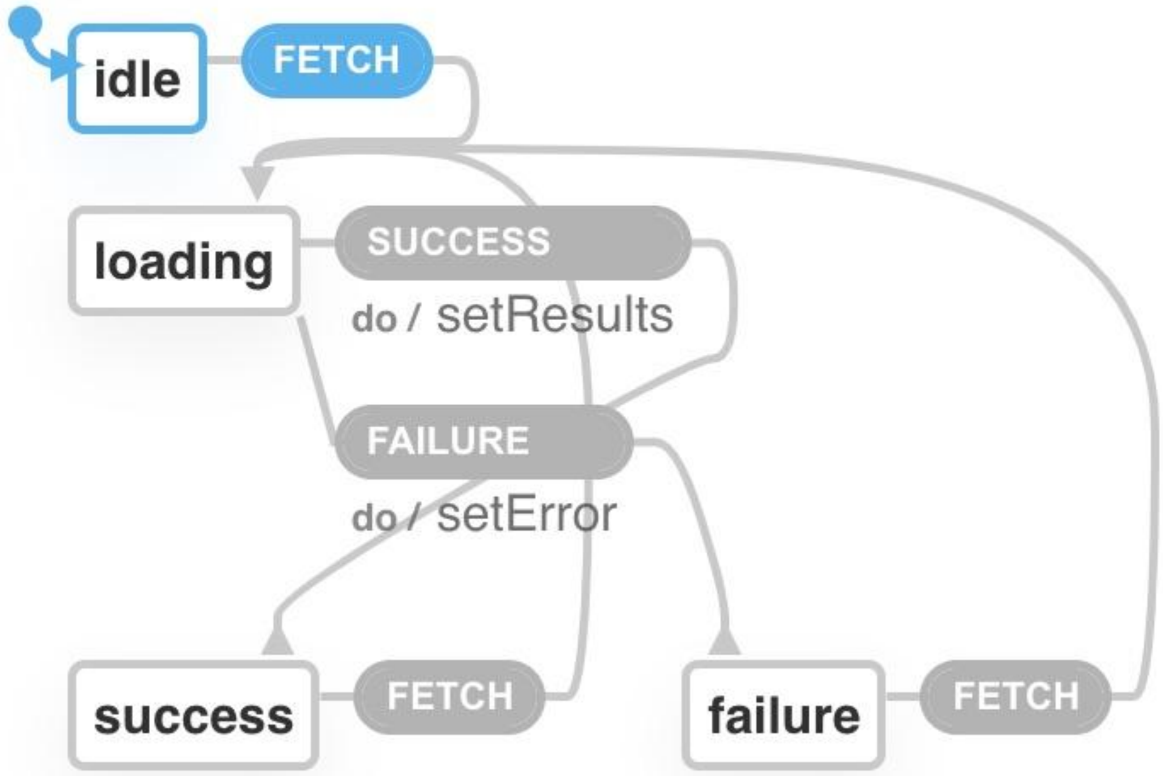


```
const [isLoading, setIsLoading] = useState(false)
const [error, setError] = useState('')
const [results, setResults] = useState([])

const fetchQuestions = async () => {
  setIsLoading(true)
  try {
    const results = await fetch(/* *** */)
    setResults(results)
    setIsLoading(false)
  } catch (e) {
    setError(e.message)
  }
}

return (
  <div>
    {isLoading && <div>Loading data...</div>}
    {results.map((result) => (
      <div key={result}>{result}</div>
    ))}
    {error && <div>{error}</div>}
    <button type="button" onClick={fetchQuestions}>
      Get fresh questions
    </button>
  </div>
```

Combinatorial Explosion

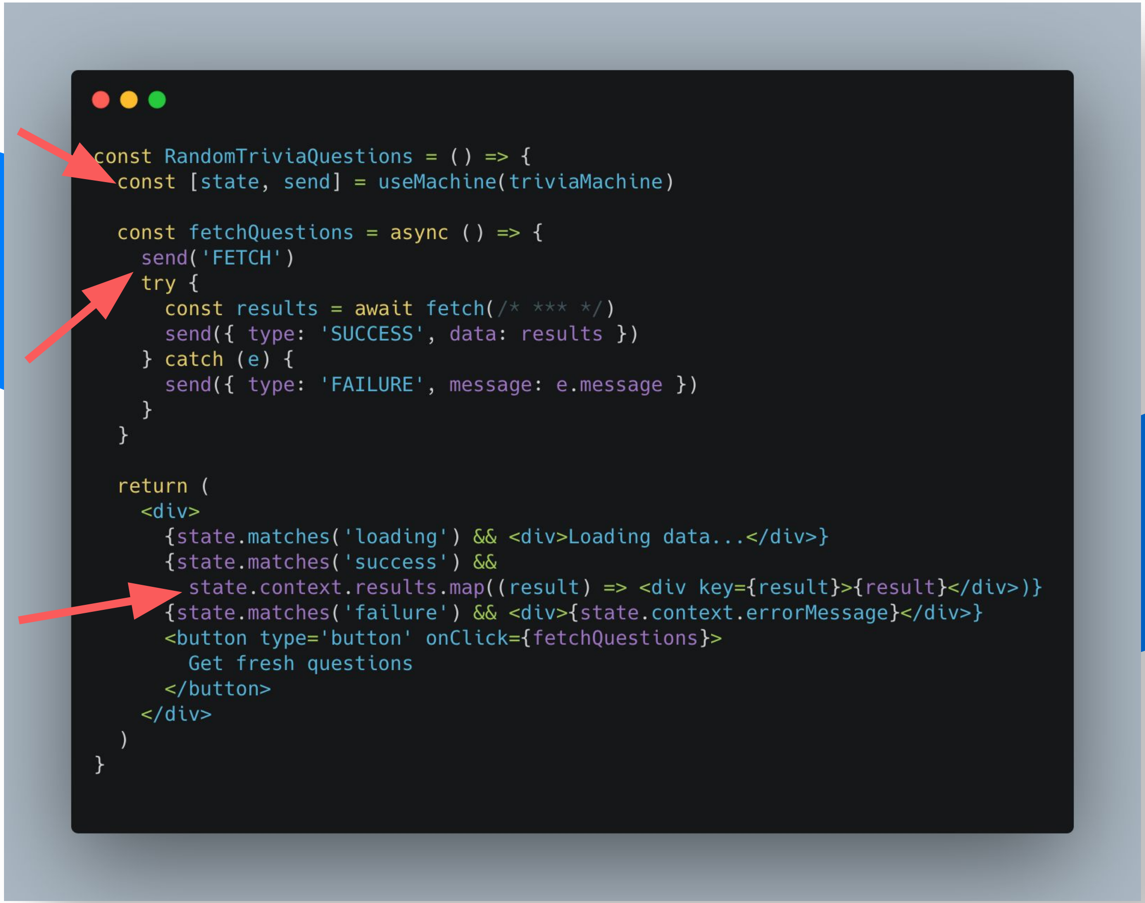
The Solution: State Machines

01. Enumerate potential states
02. Declare events to transition between states
03. Store additional data in context



State Machine Visualization

Using a State Machine



```
const RandomTriviaQuestions = () => {
  const [state, send] = useMachine(triviaMachine)

  const fetchQuestions = async () => {
    send('FETCH')
    try {
      const results = await fetch(/* *** */)
      send({ type: 'SUCCESS', data: results })
    } catch (e) {
      send({ type: 'FAILURE', message: e.message })
    }
  }

  return (
    <div>
      {state.matches('loading') && <div>Loading data...</div>}
      {state.matches('success') &&
        state.context.results.map((result) => <div key={result}>{result}</div>)}
      {state.matches('failure') && <div>{state.context.errorMessage}</div>}
      <button type='button' onClick={fetchQuestions}>
        Get fresh questions
      </button>
    </div>
  )
}
```

State Machine Benefits

- Deterministic and declarative
- Business logic is not directly tied to a UI framework/library
- Scale between small and complex implementations
- Extend or adapt a state machine with future requirements
- Optionally, configure context when starting a machine

Thank you.

Phase2