I ❤️ testing

Code coverage

Watch mode

Test framework

Test Runner

Reporters

I ❤️ testing

Assertion library

Mocking library

Test Utilities

Code coverage

Watch mode

Test framework

Test Runner

Reporters

I ❤️ the idea of testing

Assertion library

Mocking library

Test Utilities

more effort => less tests

less effort => more tests

2016

Watch mode

Code coverage

Test framework

Test Runner

Reporters

Jest

Assertion library

Mocking library

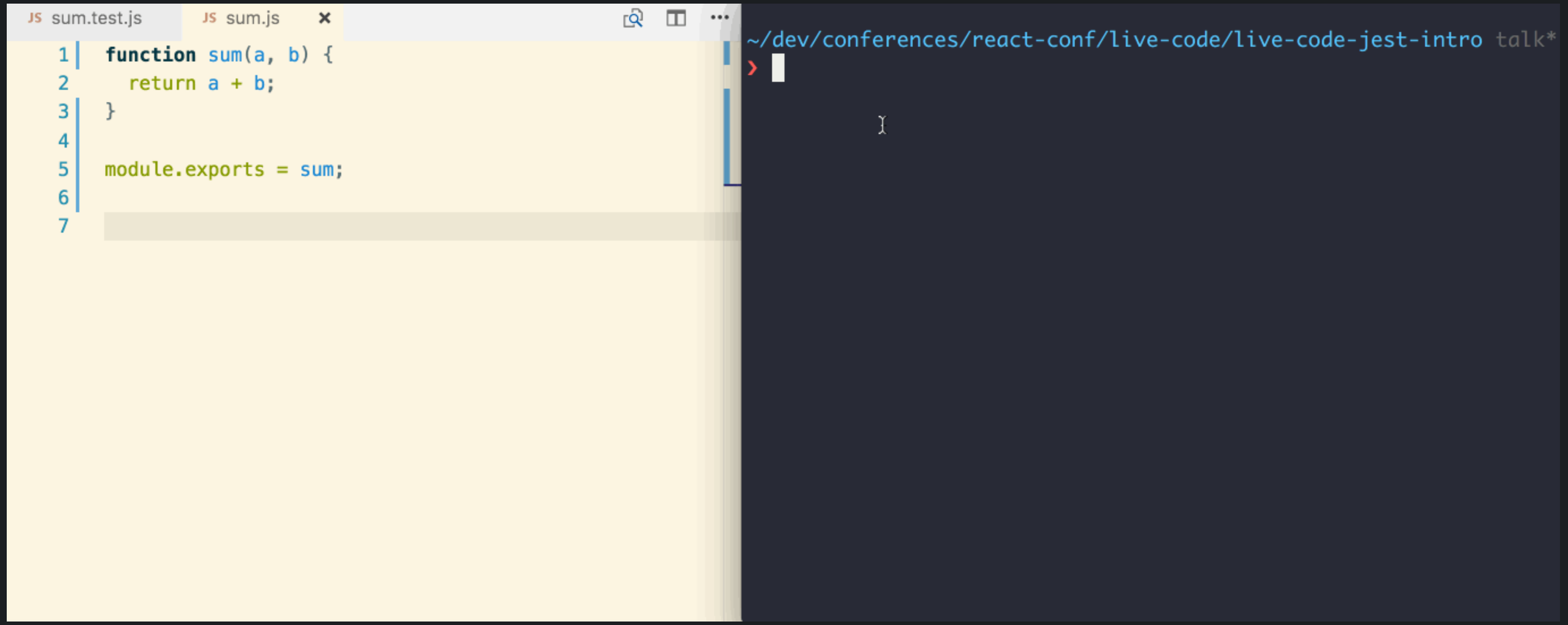
Test Utilities

Jest

Zero config

- 6x faster
- instant run
- snapshot tests

Let’s setup Jest


sum.test.js
sum.js
1 function sum(a, b) {
2 return a + b;
3 }
4
5 module.exports = sum;
6
7
~/dev/conferences/react-conf/live-code/live-code-jest-intro talk*
›

Emoji Cinema

Start typing to see emojis!

Emoji Cinema

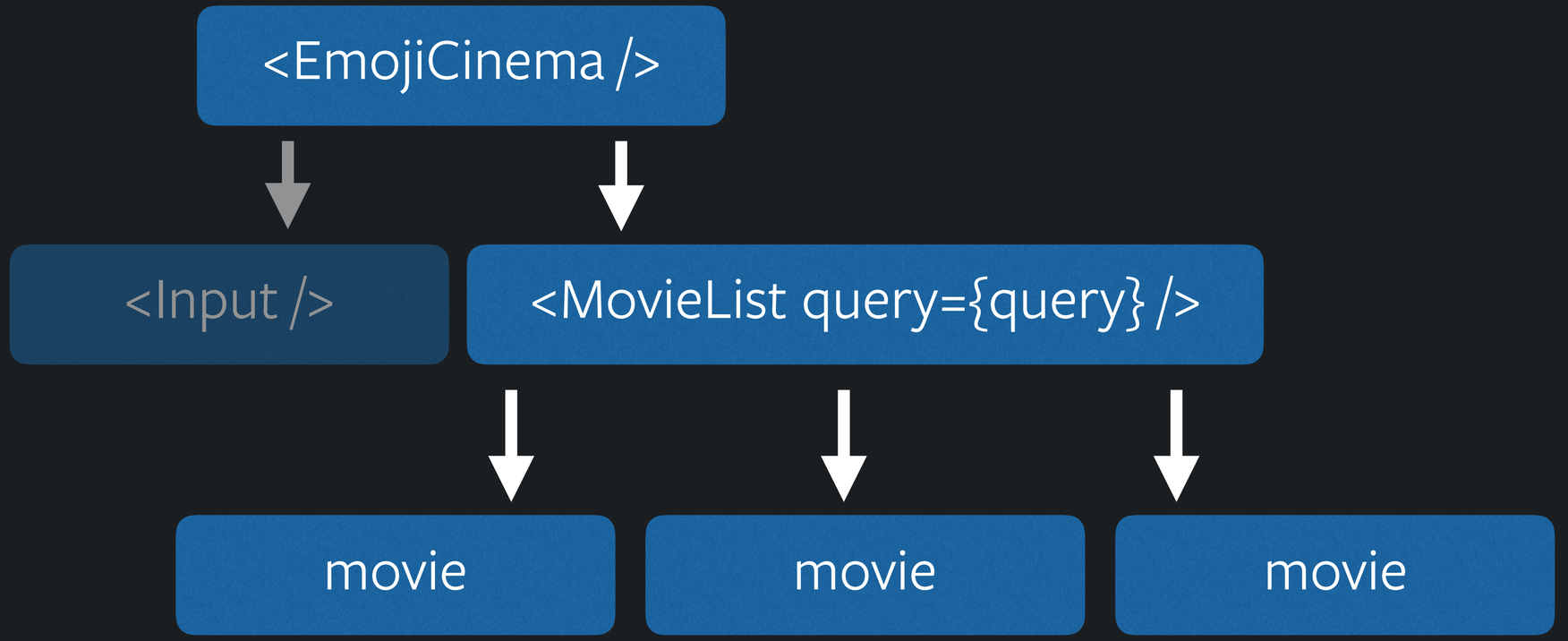
Frozen

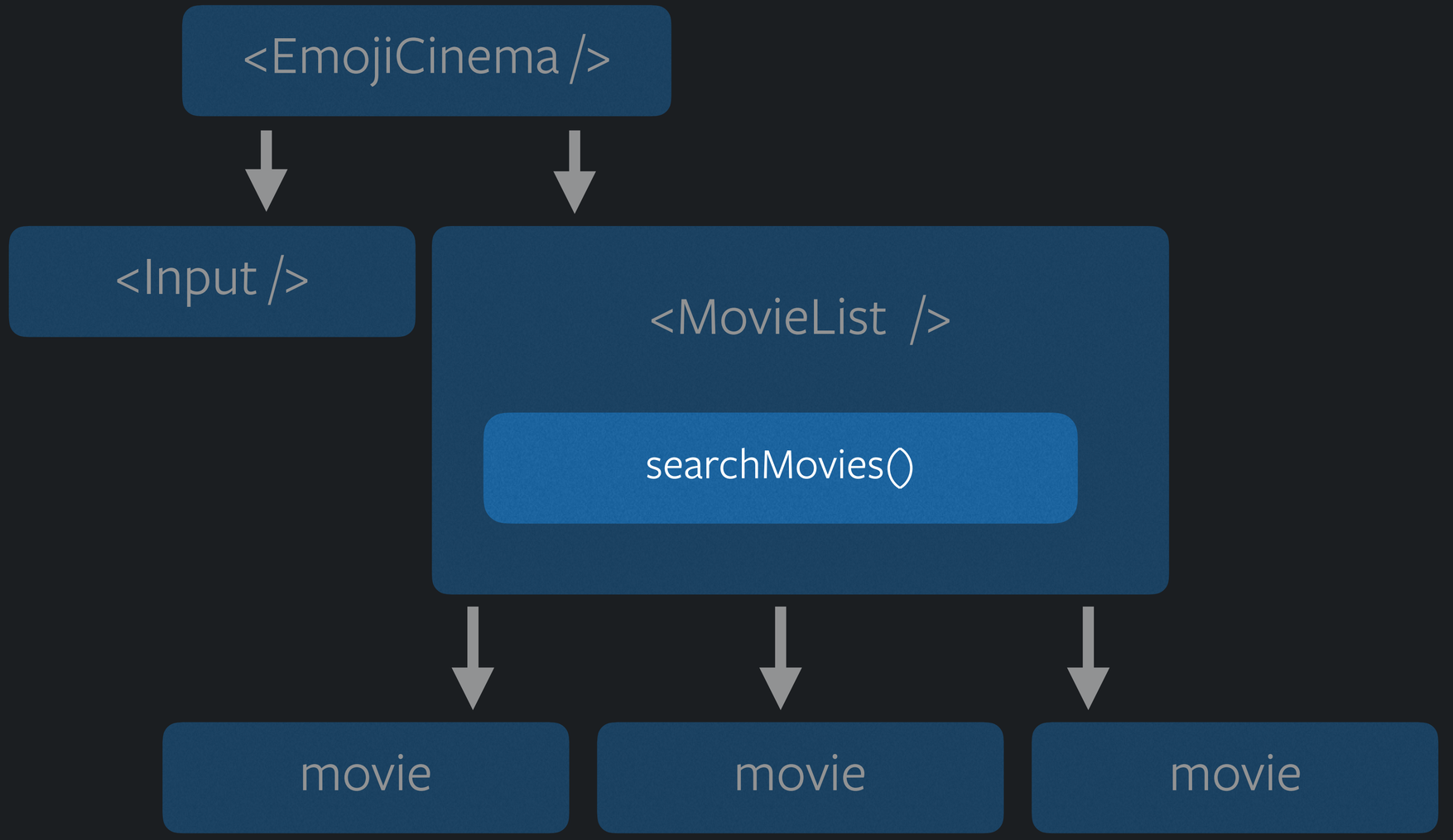
❄️👗 Frozen

Emoji Cinema

Finding Nemo

Finding Nemo


<EmojiCinema />
<Input />
<MovieList query={query} />
movie
movie
movie


<EmojiCinema />
<Input />
<MovieList />
searchMovies()
movie
movie
movie

searchMovies

searchMovies

```
function searchMovies(query) {
  return emojiMap.filter((movie) => {
    return movie.title.includes(query)
  });
}
```

```
const emojiMap = [
  { title: 'Frozen', emoji: '❄️ 👗'},
  { title: 'Finding Nemo', emoji: '🔍🐠'},
  { title: 'Kung Fu Panda', emoji: '👊💥🐼'},
  { title: 'Harry Potter', emoji: '👨👓⚡'},
  { title: 'Hunger Games', emoji: '🍗🍖🎮'},
  { title: 'The Lion King', emoji: '🦁👑'},
  { title: 'ET', emoji: '👽🚴‍♂️🌛'},
];

function searchMovies(query) {
  return emojiMap.filter((movie) => {
    return movie.title.includes(query)
  });
}
```

How would we test this?

```
expect(searchMovies('F')).toEqual([someMovies]);
```

```
const searchMovies = require('./searchMovies');

it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([]);
})
```

```
const searchMovies = require('./searchMovies');

it('Returns the movies that match the query', () => {
  console.log(searchMovies('F'));
  // expect(searchMovies('F')).toEqual([]);
})
```

```
const searchMovies = require('./searchMovies');

it('Returns the movies that match the query', () => {
  console.log(searchMovies('F'));
  // expect(searchMovies('F')).toEqual([]);
})
```

Jest

```
console.log src/searchMovies.test.js:13
  [ { title: 'Frozen', emoji: '❄👗' },
    { title: 'Finding Nemo', emoji: '🔍🐠' },
    { title: 'Kung Fu Panda', emoji: '👊💥🐼' } ]
```

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([
    { title: 'Frozen', emoji: '❄️ 👗' },
    { title: 'Finding Nemo', emoji: '🔍🐠' },
    { title: 'Kung Fu Panda', emoji: '👊💥🐼' },
  ]);
})
```

Jest

```
PASS src/movieUtils.test.js
  ✓ Returns the movies that match the query (1ms)
```


FROZEN ✌

```
Difference:

- Expected
+ Received

@@ -2,10 +2,14 @@
    Object {
      "emoji": "❄👗",
      "title": "Frozen",
    },
    Object {
+     "emoji": "❄👗✌",
+     "title": "Frozen 2",
+   },
+   Object {
      "emoji": "🔍🐠",
      "title": "Finding Nemo",
    },
```

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([
    { title: 'Frozen', emoji: '❄️ 👗' },
    { title: 'Finding Nemo', emoji: '🔍🐠' },
    { title: 'Kung Fu Panda', emoji: '👊💥🐼' },
  ]);
});
```

Jest

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([
    { title: 'Frozen', emoji: '❄️ 👗' },
    { title: 'Frozen 2', emoji: '❄️ 👗 ✌️' },
    { title: 'Finding Nemo', emoji: '🔍🐠' },
    { title: 'Kung Fu Panda', emoji: '👊💥🐼' },
  ]);
});
```

Jest

```
PASS src/movieUtils.test.js
  ✓ Returns the movies that match the query (1ms)
```

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([
    { title: 'Frozen', emoji: '❄️ 👗' },
    { title: 'Frozen 2', emoji: '❄️ 👗 ✌️' },
    { title: 'Finding Nemo', emoji: '🔍🐠' },
    { title: 'Kung Fu Panda', emoji: '👊💥🐼' },
  ]);
});
```

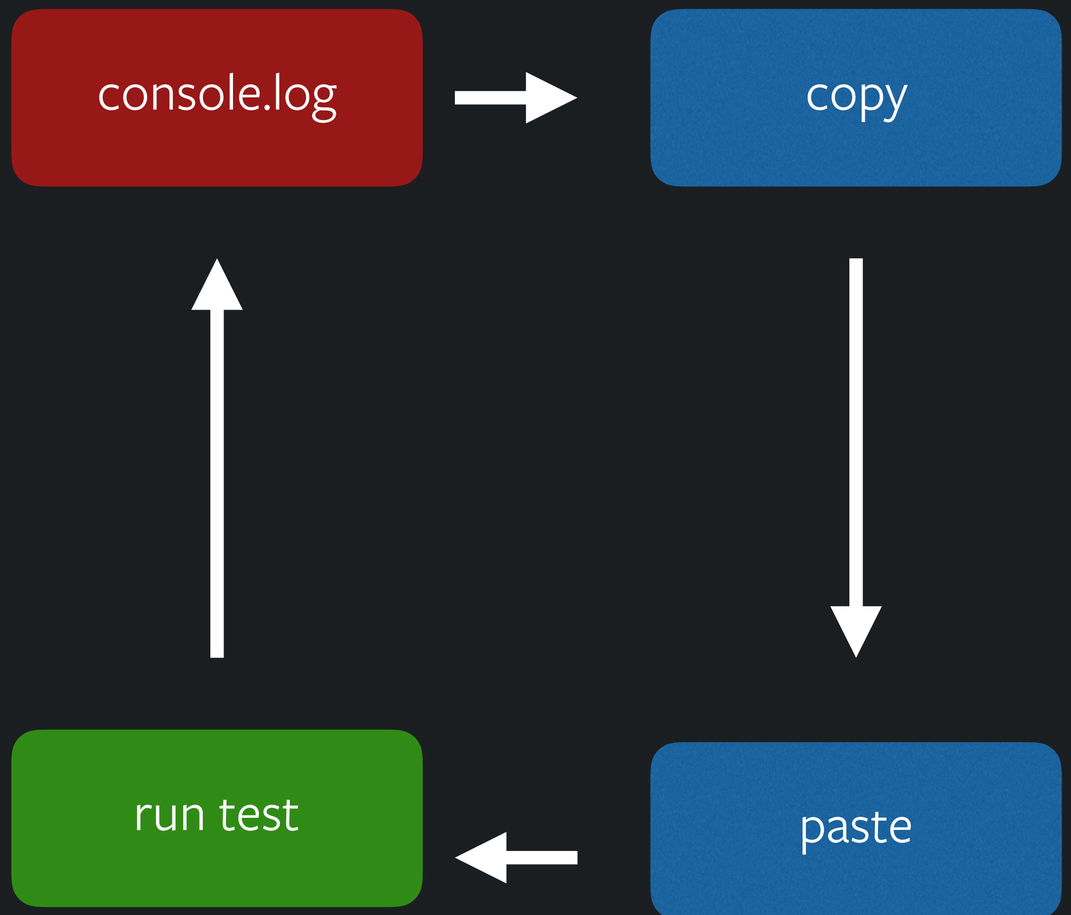
Jest

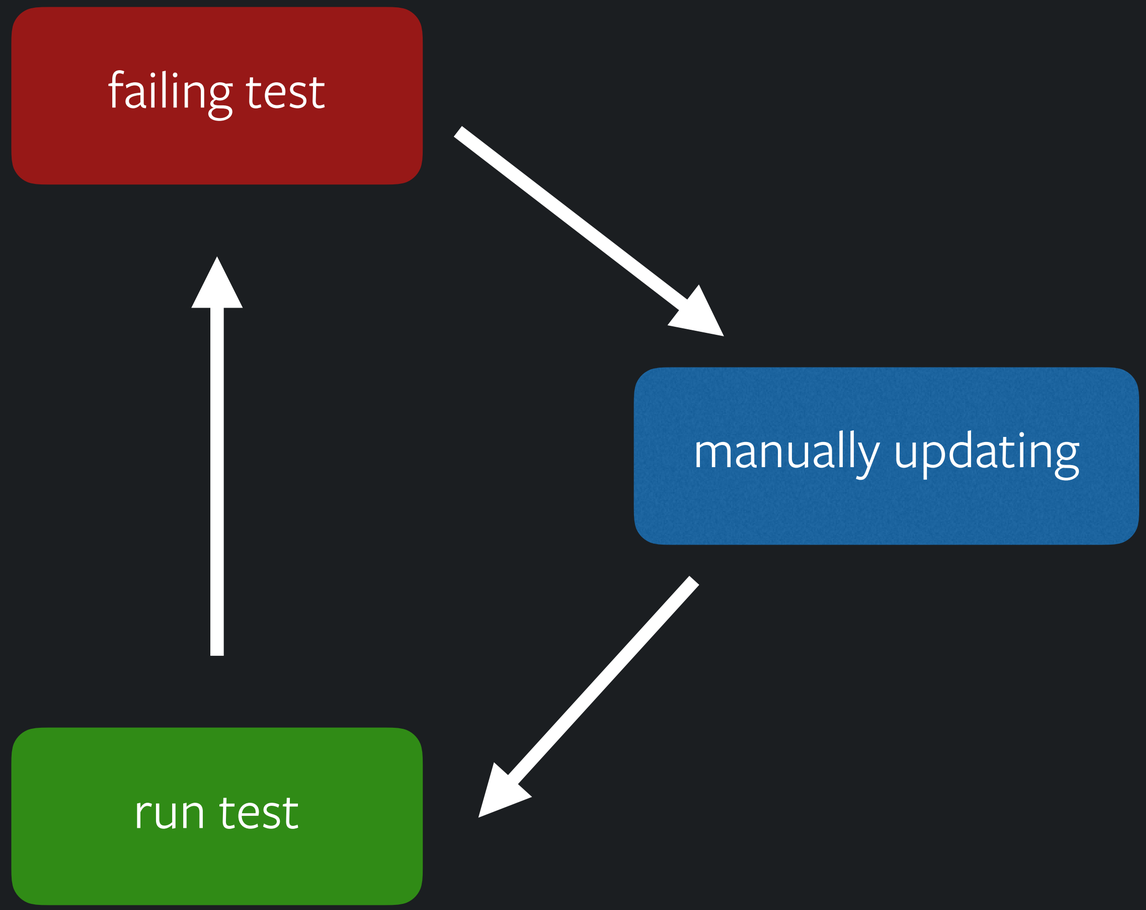
```
- Expected
+ Received

@@ -1,18 +1,22 @@
 Array [
   Object {
     "emoji": "❄️👗",
     "title": "Frozen",
+    "year": 2013,
   },
```

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([
    { year: 2013, title: 'Frozen', emoji: '❄️ 👗' },
    { year: 'TBA', title: 'Frozen 2', emoji: '❄️ 👗 ✌️' },
    { year: 2003, title: 'Finding Nemo', emoji: '🔍🐠' },
    { year: 2008, title: 'Kung Fu Panda', emoji: '👊💥🐼' },
  ]);
});
```

Jest

```
PASS src/movieUtils.test.js
  ✓ Returns the movies that match the query (1ms)
```


console.log
copy
paste
run test


failing test
manually updating
run test

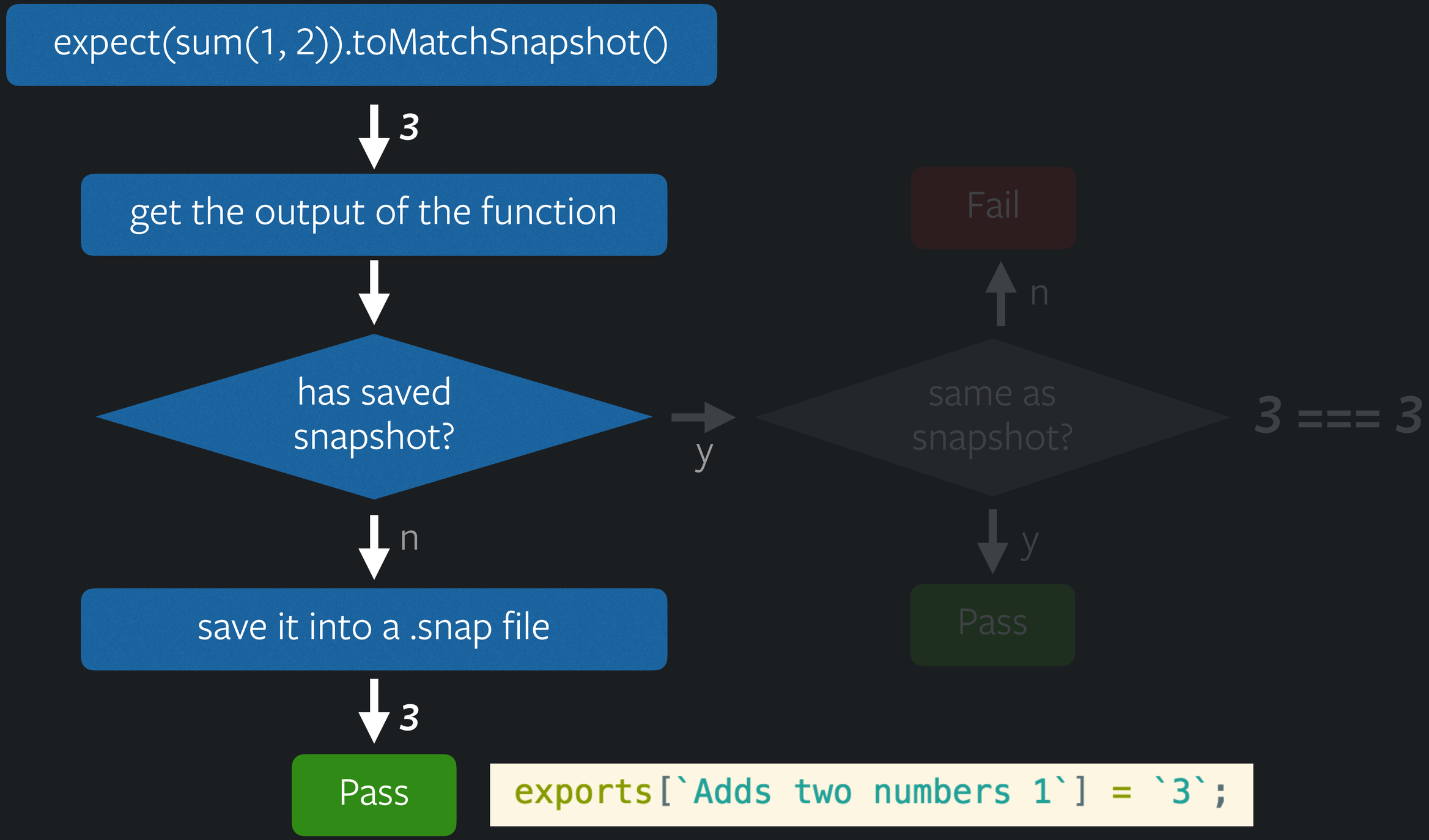
Problems with this approach

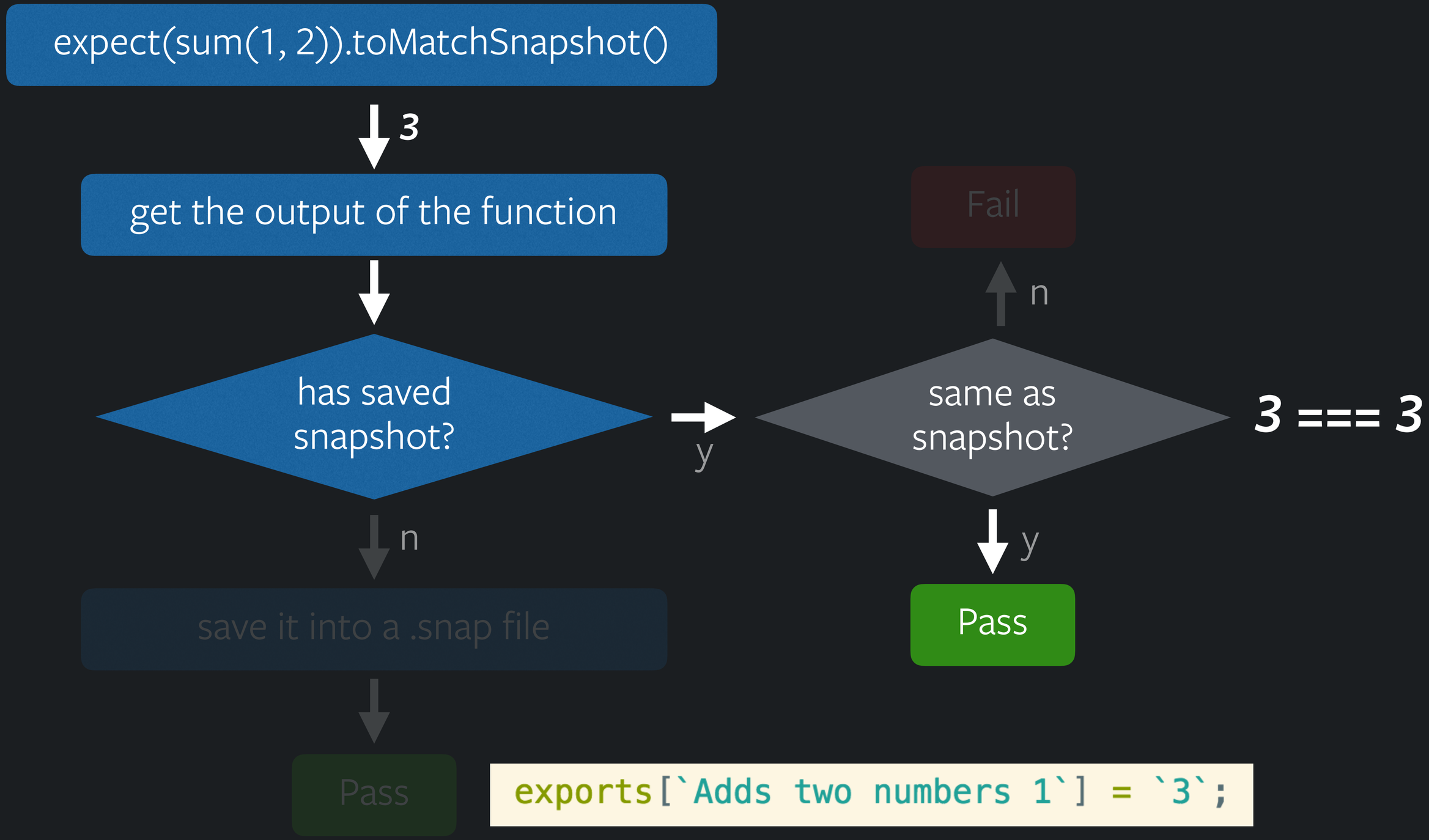
- Manual process
- Does not adapt to changes
- Hard to maintain

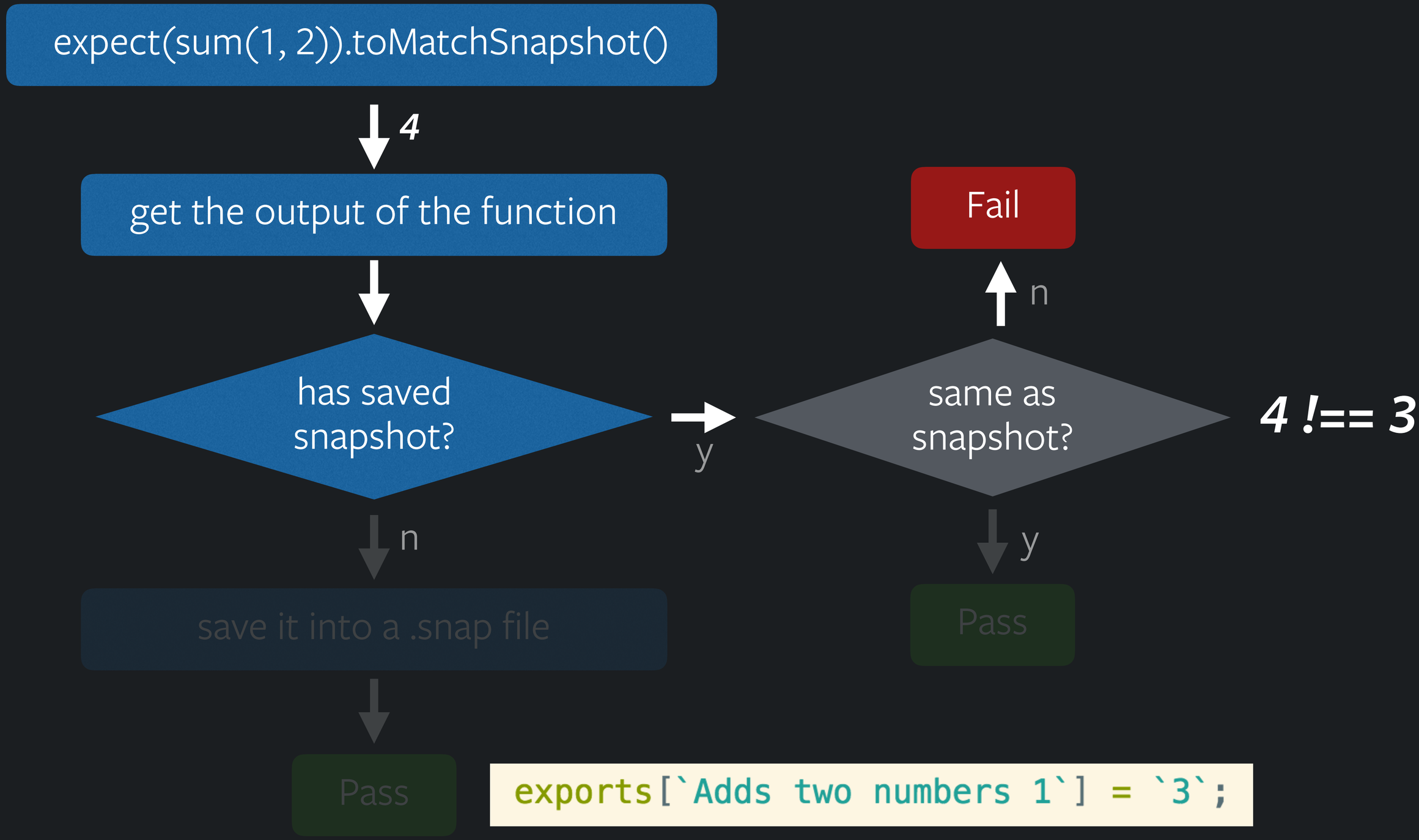
Snapshot Testing

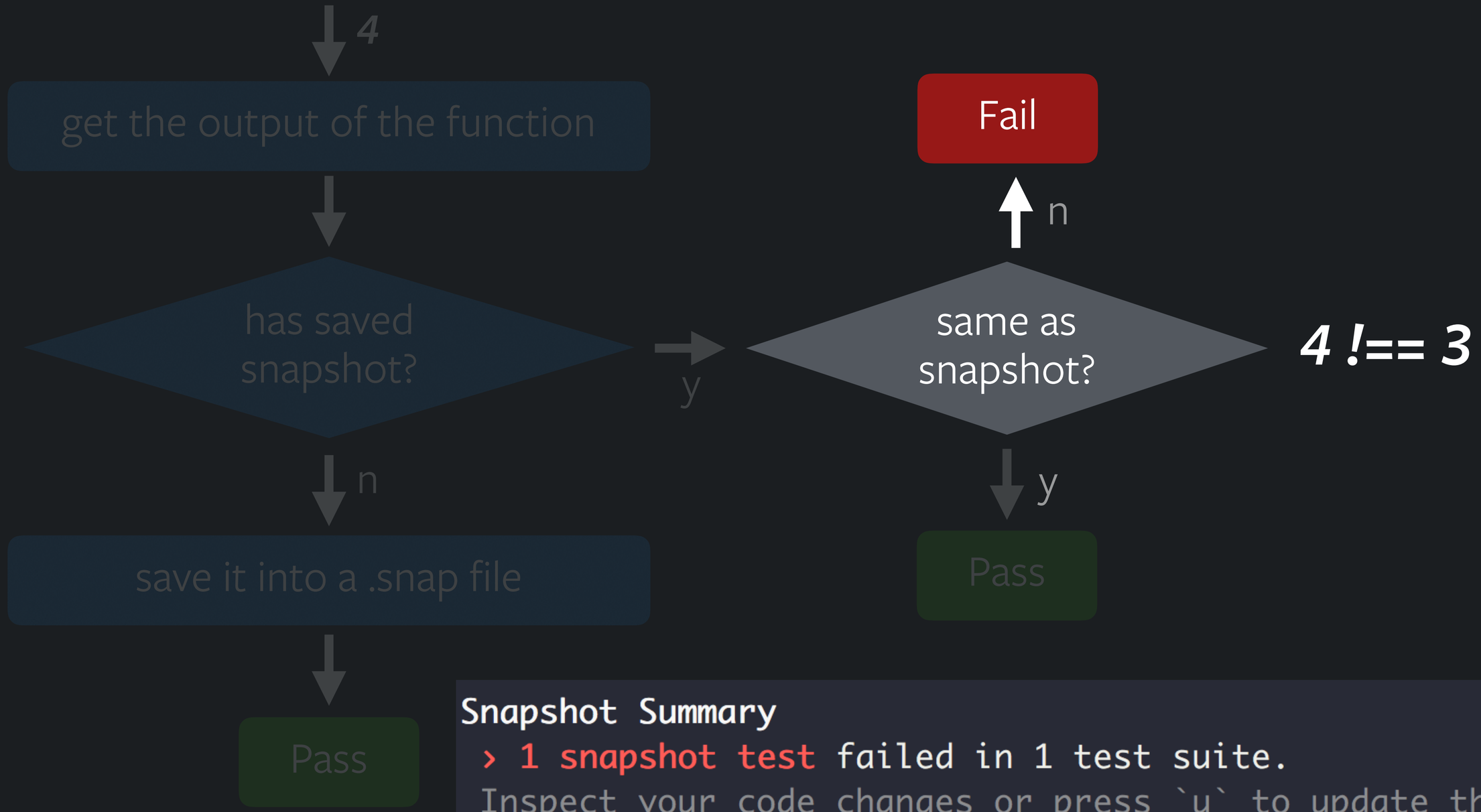
```
expect(sum(1, 2)).toBe(3)
```

```
expect(sum(1, 2)).toMatchSnapshot()
```


expect(sum(1, 2)).toMatchSnapshot()
3
get the output of the function
has saved snapshot?
y
same as snapshot?
3 === 3
n
Fail
n
save it into a .snap file
y
Pass
3
Pass
exports[`Adds two numbers 1`] = `3`;


expect(sum(1, 2)).toMatchSnapshot()
3
get the output of the function
has saved snapshot?
y
same as snapshot?
3 === 3
n
Fail
n
save it into a .snap file
y
Pass
Pass
exports[`Adds two numbers 1`] = `3`;


expect(sum(1, 2)).toMatchSnapshot()
4
get the output of the function
has saved snapshot?
y
same as snapshot?
4 !== 3
n
Fail
n
save it into a .snap file
y
Pass
Pass
exports[`Adds two numbers 1`] = `3`;




```
Snapshot Summary
 › 1 snapshot test failed in 1 test suite.
 Inspect your code changes or press `u` to update them.
```

Snapshot benefits

- Automated process, to manage snapshots
- Adapt to changes
- Easy to maintain


EXPLORER
OPEN EDITORS
LIVE-CODE-JEST-INTRO
.vscode
coverage
node_modules
src
sum.js
sum.test.js
package.json
yarn.lock
sum.test.js
sum.js
1 const sum = require('./sum');
2
3 it('Adds up two numbers', () => {
4 expect(sum(1, 2)).toBe(3);
5 });
PASS src/sum.test.js
✓ Adds up two numbers (1ms)
Test Suites: 1 passed, 1 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 0.029s, estimated 1s
Ran all test suites related to changed files.
Watch Usage
› Press a to run all tests.
› Press p to filter by a filename regex pattern.
› Press t to filter by a test name regex pattern.
› Press q to quit watch mode.
› Press Enter to trigger a test run.

Emoji Cinema

F

❄️👗 Frozen

🔍🐠 Finding Nemo

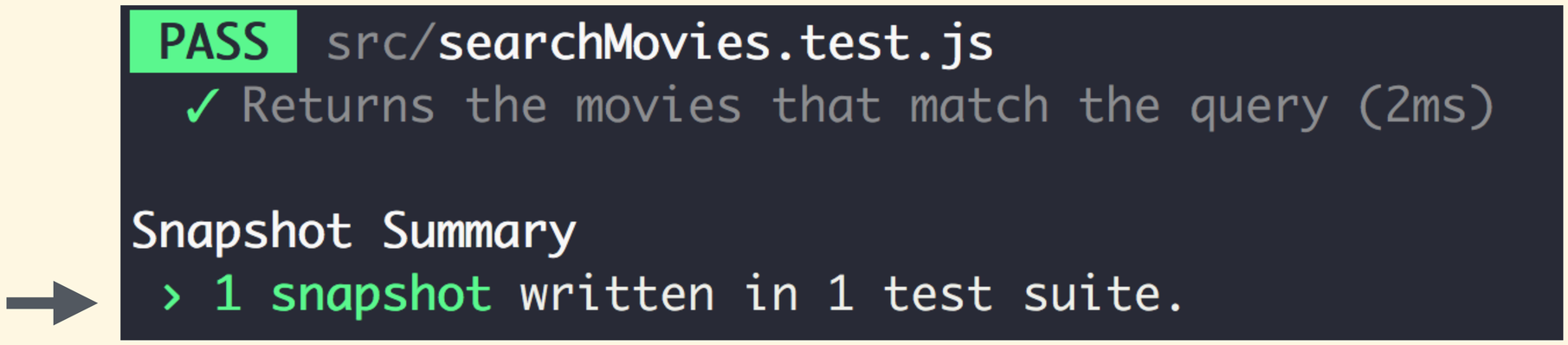
👊💥🐼 Kung Fu Panda

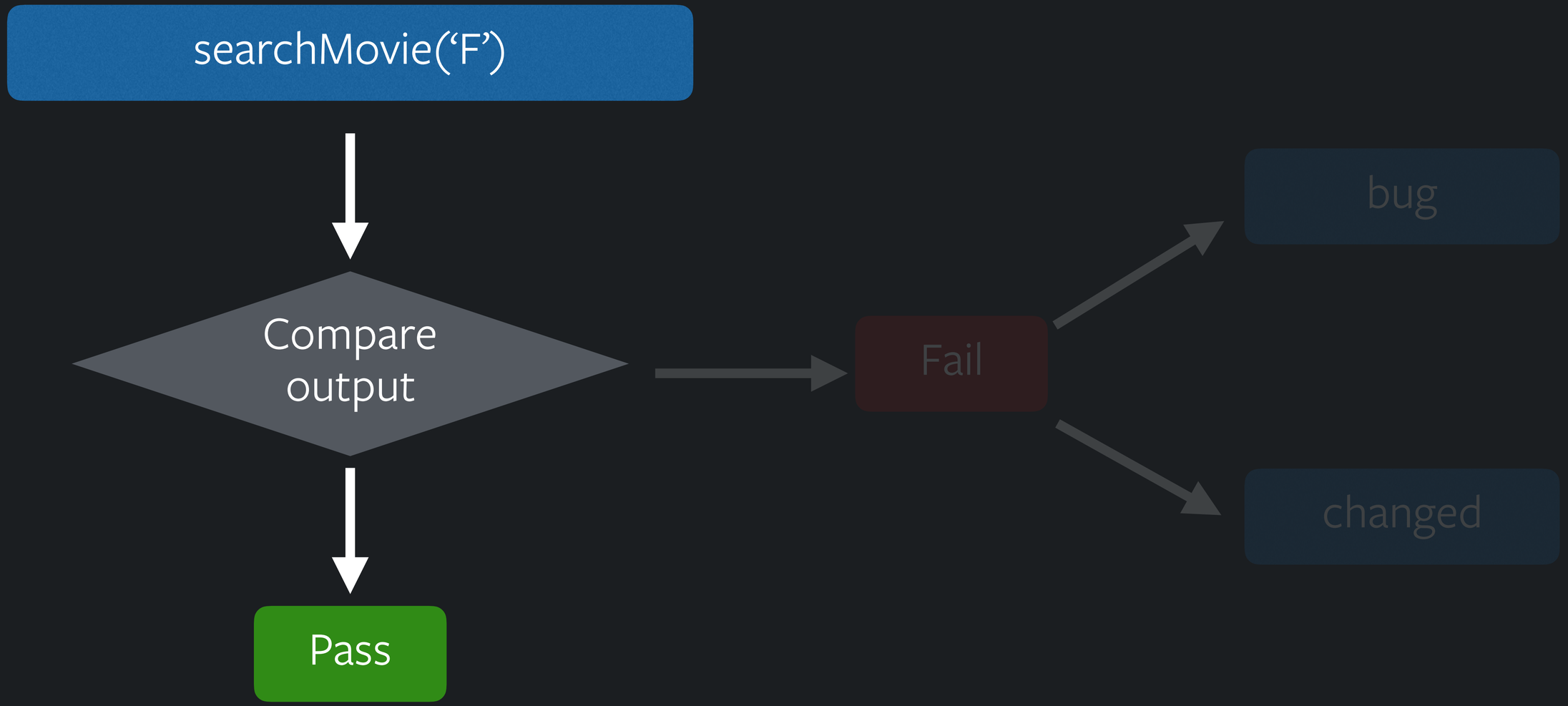
```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toEqual([
    { title: 'Frozen', emoji: '❄️ 👗' },
    { title: 'Finding Nemo', emoji: '🔍🐠' },
    { title: 'Kung Fu Panda', emoji: '👊💥🐼' },
  ]);
});
```

Jest

```
PASS src/movieUtils.test.js
  ✓ Returns the movies that match the query (1ms)
```

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toMatchSnapshot();
});
```

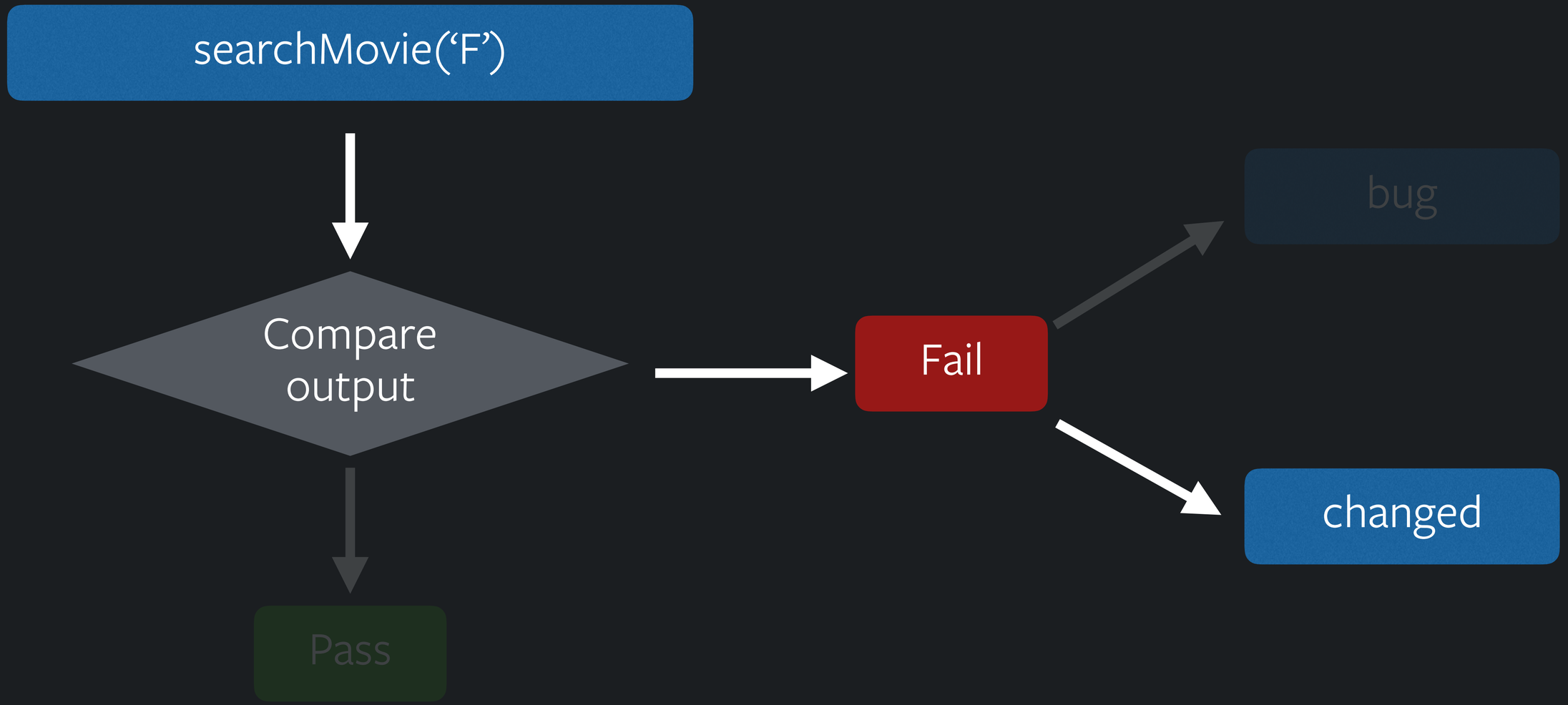




searchMovie('F')
Compare output
Pass
Fail
bug
changed

```
exports[`Returns the movies that match the query 1`] = `
Array [
  Object {
    "emoji": "❄️ 👗",
    "title": "Frozen",
  },
  Object {
    "emoji": "🔍🐠",
    "title": "Finding Nemo",
  },
  Object {
    "emoji": "👊💥🐼",
    "title": "Kung Fu Panda",
  },
]
`;
```

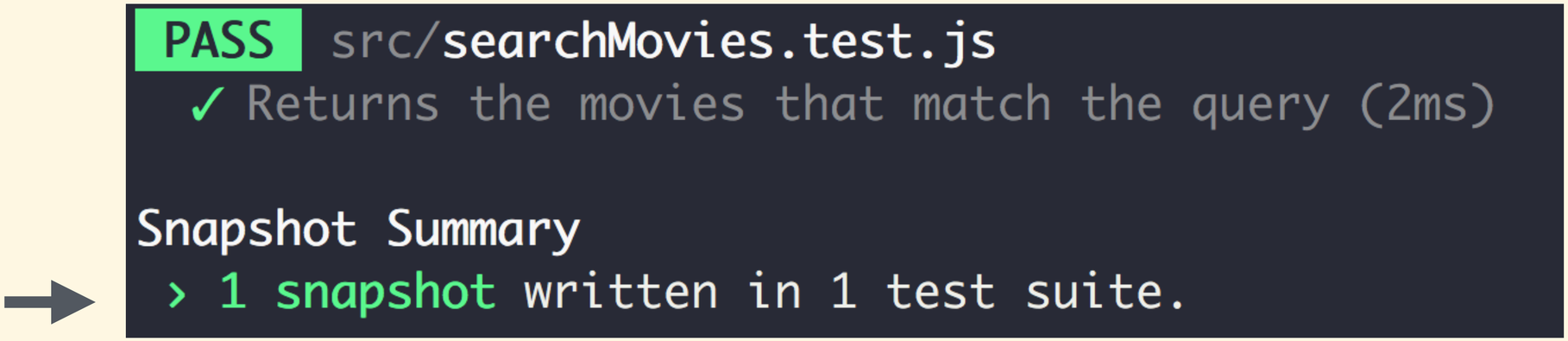

FROZEN ✌

```
expect(value).toMatchSnapshot()

Received value does not match stored snapshot 1.

- Snapshot
+ Received

@@ -2,10 +2,14 @@
    Object {
      "emoji": "❄👗",
      "title": "Frozen",
    },
    Object {
+     "emoji": "❄👗✌",
+     "title": "Frozen 2",
+   },
+   Object {
      "emoji": "🔍🐠",
      "title": "Finding Nemo",
    },
    Object {
      "emoji": "👊💥🐼",
```


searchMovie('F')
Compare output
Pass
Fail
bug
changed

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toMatchSnapshot();
});
```

Jest



```
exports[`test Returns the movies that match the query 1`] = `
Array [
  Object {
    "emoji": "❄️ 👗",
    "title": "Frozen",
  },
  Object {
    "emoji": "❄️ 👗✌️ ",
    "title": "Frozen 2",
  },
  Object {
    "emoji": "🔍🐠",
    "title": "Finding Nemo",
  },
  Object {
    "emoji": "👊💥🐼",
    "title": "Kung Fu Panda",
  },
]
`;
```

```
Array [
  Object {
    "emoji": "❄️ 👗",
    "title": "Frozen",
  },
  Object {
    "emoji": "❄️ 👗✌️ ",
    "title": "Frozen 2",
  },
  Object {
    "emoji": "🔍🐠",
    "title": "Finding Nemo",
  },
  Object {
    "emoji": "👊💥🐼",
    "title": "Kung Fu Panda",
  },
]
```

```
Array [
  "❄️ 👗 Frozen",
  "❄️ 👗✌️ Frozen 2",
  "🔍🐠 Finding Nemo",
  "👊💥🐼 Kung Fu Panda",
]
```

Snapshot Serializers

What should the snapshot look like

```
expect.addSnapshotSerializer({
  test: (val) => val.title && val.emoji,
  print: (val) => `${val.emoji} ${val.title}`
});
```

```
expect.addSnapshotSerializer({
  test: (val) => val.title && val.emoji,
  print: (val) => `${val.emoji} ${val.title}`
});
```

```
expect.addSnapshotSerializer({
  test: (val) => val.title && val.emoji,
  print: (val) => `${val.emoji} ${val.title}`
});
```

```
expect.addSnapshotSerializer({
  test: (val) => val.title && val.emoji,
  print: (val) => `${val.emoji} ${val.title}`
});
```

```
expect.addSnapshotSerializer({
  test: (val) => val.title && val.emoji,
  print: (val) => `${val.emoji} ${val.title}`
});

it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toMatchSnapshot();
});
```



```
FAIL src/searchMovies.test.js
  • Returns the movies that match the query
```

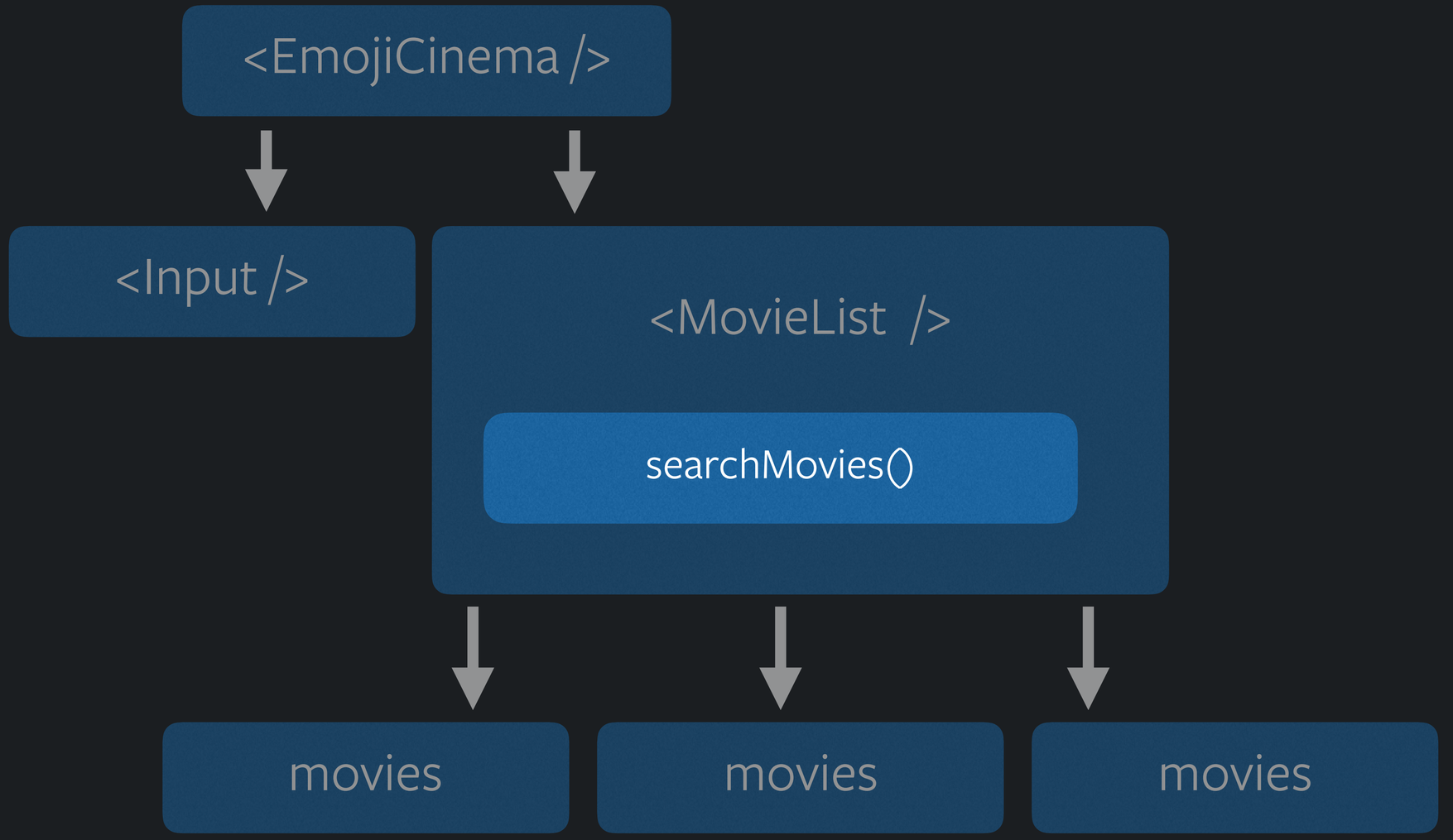
```
expect(value).toMatchSnapshot()

Received value does not match stored snapshot 1.

- Snapshot
+ Received

 Array [
-  Object {
-    "emoji": "❄👗",
-    "title": "Frozen",
-  },
-  Object {
-    "emoji": "❄👗✌",
-    "title": "Frozen 2",
-  },
-  Object {
-    "emoji": "🔍🐠",
-    "title": "Finding Nemo",
-  },
-  Object {
-    "emoji": "👊💥🐼",
-    "title": "Kung Fu Panda",
-  },
+  ❄👗 Frozen,
+  ❄👗✌ Frozen 2,
+  🔍🐠 Finding Nemo,
+  👊💥🐼 Kung Fu Panda,
 ]
```

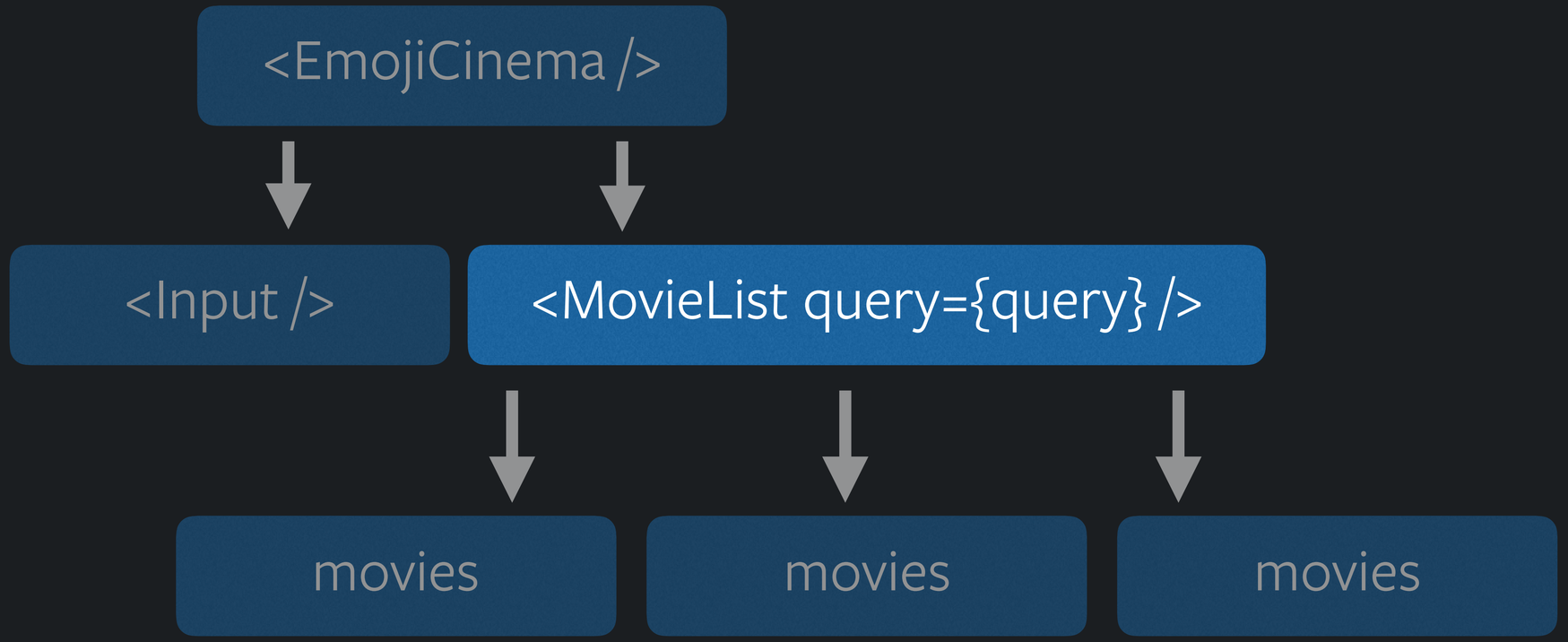
```
exports[`test Returns the movies that match the query 1`] = `
Array [
  "❄️ 👗 Frozen",
  "❄️ 👗✌️ Frozen 2",
  "🔍🐠 Finding Nemo",
  "👊💥🐼 Kung Fu Panda",
]
`;
```

```
it('Returns the movies that match the query', () => {
  expect(searchMovies('F')).toMatchSnapshot();
});
```

Jest

```
PASS src/movieUtils.test.js
  ✓ Returns the movies that match the query (1ms)
```


<EmojiCinema />
<Input />
<MovieList />
searchMovies()
movies
movies
movies


<EmojiCinema />
<Input />
<MovieList query={query} />
movies
movies
movies

How would we test this?

- Simply don’t test it
- Selenium: flaky, slow, and high maintenance
- Snapshots

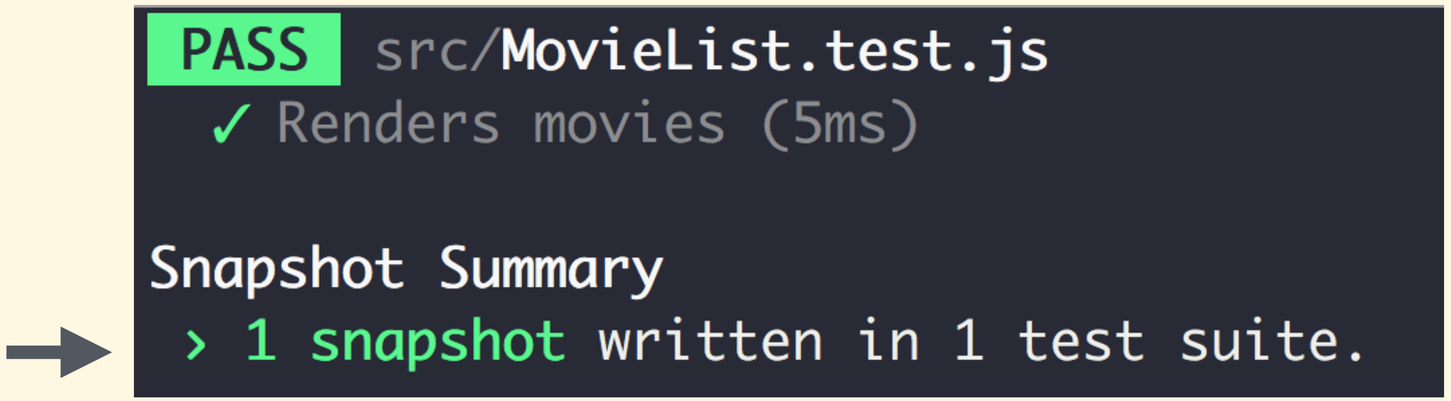
How would we test this?

```
import React from 'react';
import renderer from 'react-test-renderer'
import MovieList from './MovieList';

it('Renders movies', () => {
  expect(component).toMatchSnapshot();
});
```

```
import React from 'react';
import renderer from 'react-test-renderer'
import MovieList from './MovieList';

it('Renders movies', () => {
  const component = renderer.create(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

```
it('Renders movies', () => {
  const component = renderer.create(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

Jest



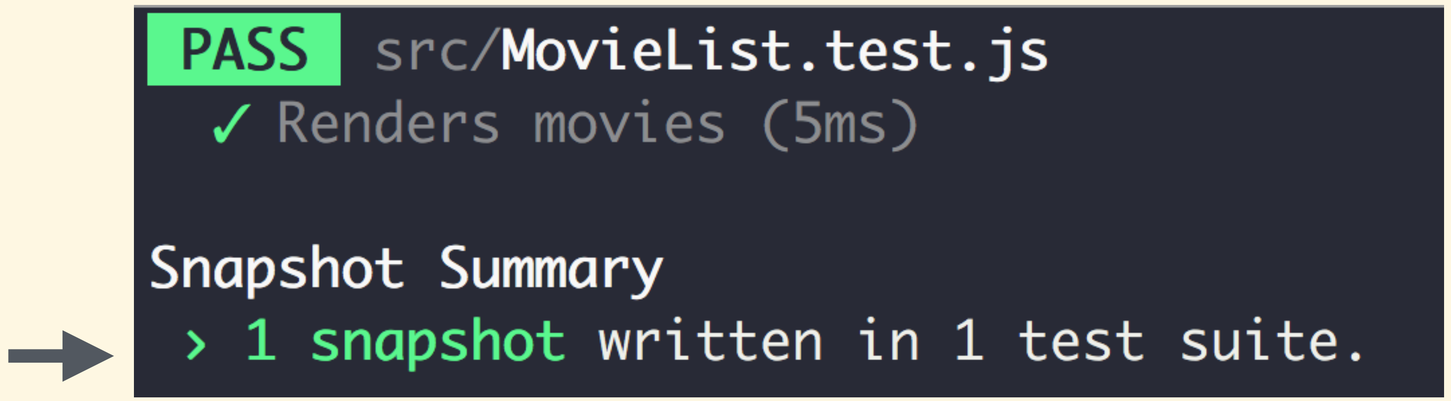
```
exports[`test Renders movies 1`] = `
<ul>
  <li
    className="Movie">
    <div>
      ❄️ 👗 Frozen
    </div>
  </li>
  <li
    className="Movie">
    <div>
      🔍🐠 Finding Nemo
    </div>
  </li>
  <li
    className="Movie">
    <div>
      👊💥🐼 Kung Fu Panda
    </div>
  </li>
</ul>
`;
```


FROZEN ✌

```
expect(value).toMatchSnapshot()

Received value does not match stored snapshot 1.

- Snapshot
+ Received

@@ -6,10 +6,16 @@
      </div>
    </li>
    <li
      className="Movie">
      <div>
+       ❄️👗✌️Frozen 2
+     </div>
+   </li>
+   <li
+     className="Movie">
+     <div>
```

```
it('Renders movies', () => {
  const component = renderer.create(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

Jest



What about Enzyme?

Hint: 👀 Serializers

```
it('Renders movies', () => {
  const component = renderer.create(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

```
import { mount } from 'enzyme';
import enzymeSerializer from 'enzyme-to-json/serializer';

expect.addSnapshostSerializer(enzymeSerializer);

it('Renders movies', () => {
  const component = mount(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

Jest

```
PASS src/MovieList.test.js
  ✓ Renders movies (3ms)
```

```
import { shallow } from 'enzyme';
import enzymeSerializer from ‘enzyme-to-json/serializer’;

expect.addSnapshostSerializer(enzymeSerializer);

it('Renders movies', () => {
  const component = shallow(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

Snapshots over time

- Our application is not static
- Capture the transformation of an object over time

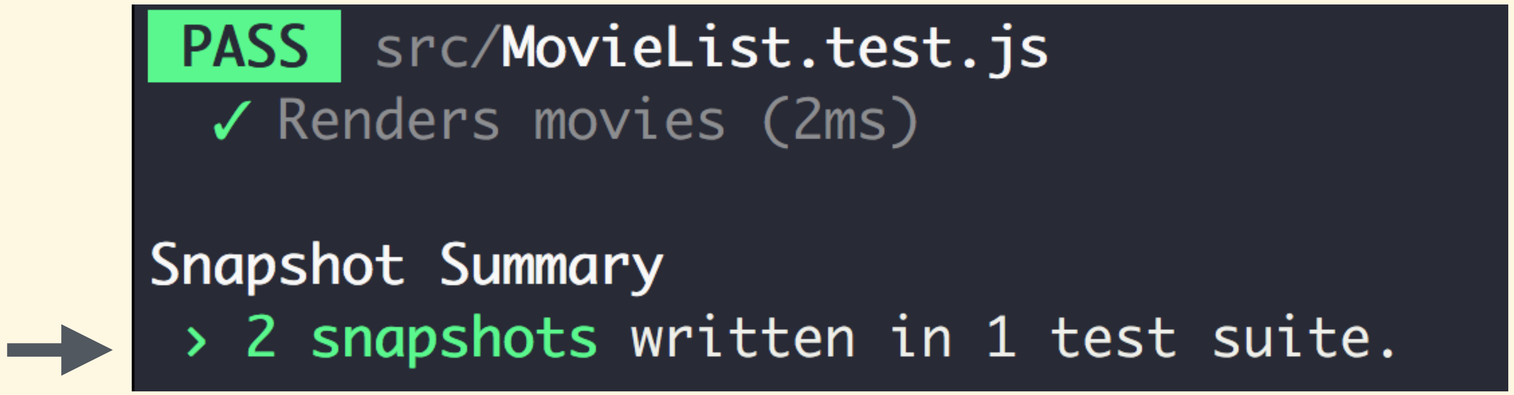
```
it('Renders movies', () => {
  const component = renderer.create(
    <MovieList query="F" />
  );

  expect(component).toMatchSnapshot();
});
```

Jest

```
PASS src/MovieList.test.js
  ✓ Renders movies (3ms)
```

```
it('Renders movies', () => {
  ['F', 'Fro', 'Frozen 2'].forEach((query) => {
    const component = renderer.create(
      <MovieList query={query} />
    );

    expect(component).toMatchSnapshot();
  });
});
```

Jest



<MovieList query="Fro" />

```
exports[`test Renders movies 2`] = `
<ul>
  <li
    className="Movie">
    <div>
      ❄️ 👗 Frozen
    </div>
  </li>
  <li
    className="Movie">
    <div>
      ❄️ 👗✌️ Frozen 2
    </div>
  </li>
</ul>
`;
```

<MovieList query="Frozen 2" />

```
exports[`test Renders movies 3`] = `
<ul>
  <li
    className="Movie">
    <div>
      ❄️ 👗✌️ Frozen 2
    </div>
  </li>
</ul>
`;
```

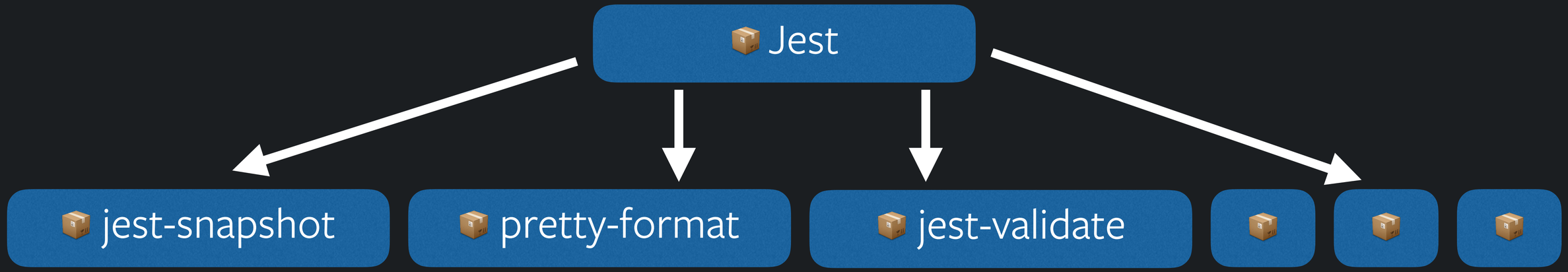
jest-movie

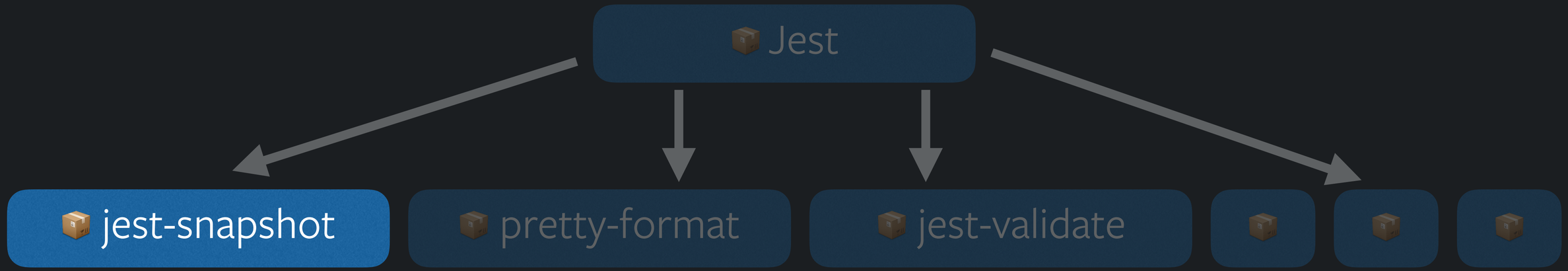
- Movie, where each snapshot is a frame
- Tools for visualizing snapshots over time
- Is built on top of other Jest packages

```
File: MovieList.test.js.snap
Movie: test Renders movies
Frame: 1/3

 › Press left arrow to see previous frame.
 › Press right arrow to see next frame.

<ul>
  <li className="Movie">
    <div>❄️👗 Frozen</div>
  </li>
  <li className="Movie">
    <div>❄️👗✌️ Frozen 2</div>
  </li>
  <li className="Movie">
    <div>🔍🐟 Finding Nemo</div>
  </li>
  <li className="Movie">
    <div>👊💥🐼 Kung Fu Panda</div>
  </li>
</ul>
```

Jest is a testing platform

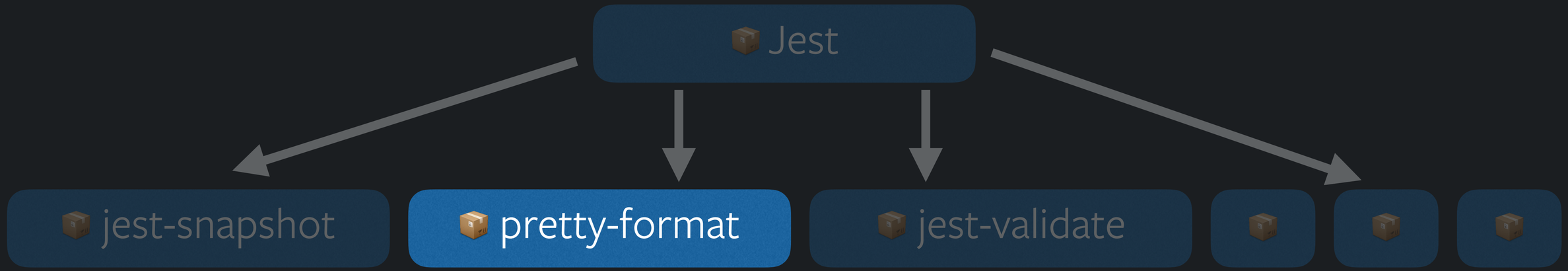
Jest is a testing platform


📦 Jest
📦 jest-snapshot
📦 pretty-format
📦 jest-validate
📦
📦
📦



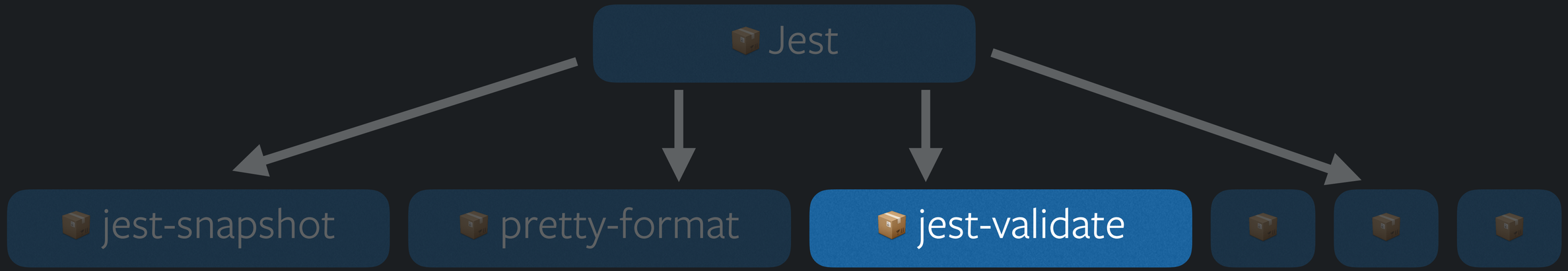
Standalone snapshot package

- Manage and update snapshots
- Manage serializers
- toMatchSnapshot()



Stringify any JavaScript value

- Supports all built-in JavaScript types
- Plugin system
- React and Immutable.js



Generic configuration validation tool

- descriptive validation errors
- deprecation warnings
- configuration examples

What’s next?

- An open area of research
- Invite you to contribute

Thanks! 😀

@rogeliog