Finding Broken Promises
In Asynchronous JavaScript Programs

Saba Alimadadi, Di Zhong, Magnus Madsen, and Frank Tip
Asynchronous JavaScript

```javascript
fs.readdir(source, function (err, files) {
  files.forEach(function (fileName, fileIndex) {
    gm(source + fileName).size(function (err, values) {
      widths.forEach(function (width, widthIndex) {
        this.resize(w, h).write(nameName, function (err) {
          {}
        })
      })
    })
  })
})

// example from callbackhell.com (simplified)

little pyramid of doom!
Promise
JavaScript Promises

```javascript
let p = new Promise(function (resolve, reject) {
});

let p2 = p.then(handleSuccess);
  .then(soMuchSuccess, handleRejection);
  .catch(moreRejection);
```
tcpPortUsed.check(port, 'localhost')
  .then(function (inUse) {
    if (inUse) {
      printErrorAndExit();
    } else {
      printSuccessMessage();
      startServer(port);
    }
  })
Currently any exceptions thrown during server startup cause the process to silently exit. This makes tracking down bugs in config files super frustrating.
In Practice
PromiseKeeper
1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs

[Madsen et al., 2017]
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs

let p1 = Promise.resolve(1);
1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs

```
let p1 = Promise.resolve(1);
let p2 = p1.then(foo);
function foo(x) {
}
```
1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs

```javascript
let p1 = Promise.resolve(1);
let p2 = p1.then(foo);
function foo(x) {
    return x + 1;
}
```
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs

let p1 = Promise.resolve(1);
let p2 = p1.then(foo);
function foo(x) {
    let p3 = Promise.resolve(x);
    return p3;
}
1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs

```
let p1 = Promise.all(p4, p5);
let p2 = p1.then(foo);
function foo(x) {
    let p3 = Promise.resolve(x);
    return p3;
}
```
1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns

unhandled promise rejections
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns

unhandled promise rejections
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns

unsettled promises
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns

unsettled promises
PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns
5. Visualize the graphs and generate reports
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
Broken Promises
115,000 Lines of JS

- Functions: 2,752
- Values: 1,524
- Promises: 4,320
  - 209
  - 4,149
209 rejected promises
209 rejected promises

8698 total
rejected promises

<table>
<thead>
<tr>
<th>subject applications</th>
<th>number of rejected promises</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actions on Google</td>
<td>13</td>
</tr>
<tr>
<td>Feathers Express</td>
<td>69</td>
</tr>
<tr>
<td>Loader Runner</td>
<td>114</td>
</tr>
<tr>
<td>Node Fetch</td>
<td>75</td>
</tr>
<tr>
<td>Node HN API</td>
<td>10</td>
</tr>
<tr>
<td>Promise MySQL</td>
<td>19</td>
</tr>
<tr>
<td>Rollout</td>
<td>2</td>
</tr>
<tr>
<td>SerialPort</td>
<td></td>
</tr>
<tr>
<td>Promise Branch</td>
<td></td>
</tr>
<tr>
<td>Razorpay</td>
<td></td>
</tr>
<tr>
<td>Telegram</td>
<td></td>
</tr>
<tr>
<td>Telegram Mobile</td>
<td></td>
</tr>
</tbody>
</table>
The bar chart shows the number of anti-patterns per category:

- **Unsettled promise**: 53 occurrences
- **Unnecessary promise**: 686 occurrences
- **Multiple settle**: 51 occurrences

These categories together account for 4,194 unique locations with anti-patterns.
Number of anti-patterns

- Unreachable reaction: 2,224
- Missing reject reaction: 1,864
- Broken chain: 632

Infer anti-patterns
Apply heuristics
Detect unique locations
total 1,864
heuristic 920
unique 52

missing reject reactions
missing reject reactions

- total: 1,864
- heuristic: 920
- unique: 52
- rejected: 7

node-fetch
> 4,632,000 weekly downloads
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
example from node-fetch module
Common Developer Practices
JavaScript Promises

```javascript
let p = new Promise((resolve, reject) {
});

let p2 = p.then(handleSuccess)
  .then(handleSuccess, handleRejection)
  .catch(handleRejection);
```

PromiseKeeper

1. Instrument automatically
2. Collect execution traces
3. Infer promise graphs
4. Analyze anti-patterns

https://github.com/nuprl/PromiseKeeper
@saba_a