

# Mutation Testing

hunting mutants by RooCode, Stryker and Gemyne

by

a.in.the.k@gmail.com

on sample repository

<https://github.com/gratex/uri.git>

# Surviving Mutant

- 1) Kod je nanic a obsahuje nepodstatny balast
- 2) test je nanic a nepokryva podstatnu cast kodu
- 3) ani diva svina (zatial nevie) alebo je to na exclude z mutacii (aj take pripady su)

## ⌚ Roo has a question

Based on this analysis, do you agree that the correct action is to strengthen the existing test case to make it more specific?

Yes, proceed with strengthening the test for `\\_preParseBaseUrl`.

No, let's try to refactor the `\\_preParseBaseUrl` function instead.

Let's document this as another unresolved issue and move on.

# Sample1

```
82  function uriBuilderRql(strings, ...values) {  
83 -  const encoders = ENCODERS.slice(0); ▼  
84 +  const encoders = ENCODERS;  
85      encoders.splice(1, 1, [ 'query', uri.encodeRqlValue ]);  
86      return build(encoders, strings, ...values);  
87  }
```

- Line 82: `function uriBuilderRql(strings, ...values)`

This function is a "template literal tag." It's meant to be used like

`uriBuilderRql/foo/${bar}/.` The `strings` argument becomes an array of the static parts (e.g., `['/foo', '/']`), and `values` contains the dynamic parts (e.g., `[bar]`).

- Line 83: `const encoders = ENCODERS.slice(0);`

This is the most critical line. `ENCODERS` is a constant array defined earlier in the file that holds the default encoding functions for different parts of a URL (path, query, etc.). `.slice(0)` creates a **shallow copy** of the `ENCODERS` array. This is done specifically so that the original `ENCODERS` array is not modified by what happens next.

- Line 84: `encoders.splice(1, 1, [ 'query', uri.encodeRqlValue ]);`

This line **mutates** the `encoders` array (the copy). It says: "starting at index 1, remove 1 element, and insert a new element in its place." The new element is `['query', uri.encodeRqlValue]`. This effectively swaps out the default query encoder for a special one designed for RQL (Resource Query Language).

- Line 85: `return build(encoders, strings, ...values);`

Finally, it calls the generic `build` function, passing it the newly customized list of encoders.

uriBuilder.js

uriBuilder(strings, ...values) {

```
function uriBuilderRql(strings, ...values) {  
-  const encoders = ENCODERS.slice(0);  
-  encoders.splice(1, 1, [ 'query', uri.encodeRqlValue ]);  
+  // Explicitly construct a new array, replacing the default query encoder.  
+  // This is safer than mutating a copy and clearer about the intent.  
+  const encoders = [  
+    ENCODERS[0], // path encoder  
+    ['query', uri.encodeRqlValue], // RQL-specific query encoder  
+    ENCODERS[2] // fragment encoder  
+  ];  
  return build(encoders, strings, ...values);  
}
```

Ran 60.16 tests per mutant on average.

File	% Mutation score	total	covered	# killed	# timeout	# survived	# no cov	# errors
All files	89.14	89.36	707	15	86	2	6	
_uri.js	88.22	88.22	322	15	45	0	5	
builder.js	100.00	100.00	34	0	0	0	0	0
index.js	0.00	0.00	0	0	0	1	0	
uri.js	89.31	89.54	351	0	41	1	1	

# Sample2

2

1

## Proposed Code Fix

Instead of adding more tests, I will refactor the `_checkAuthorityInvariant` function to make its logic more explicit and robust. This change will improve the code quality and, as a result, should eliminate the ambiguity that allowed the mutant to survive.

The refactoring will separate the different validation scenarios:

1. If `authority` is `null`, then all its sub-components must also be `null`.
2. If `authority` is not `null`, it must exactly match the result of recomposing its parts.

This makes the function's control flow clearer and removes the side effect from the conditional expression.

I will now apply this refactoring to

`_spikes/stryker-playground/uri/src/_uri.js`.

5

Roo said

Excellent. The refactoring of `_checkAuthorityInvariant` successfully killed the previous mutant. Now, a new one has appeared.

```
104  function _checkAuthorityInvariant(authority, userInfo, host, port) {  
105    const b = (authority == null && userInfo == null && host == null)  
  
106    if (!b) {  
107      (authority != null && authority === recomposeAuthorityComponents(  
108        if (!b) {  
109          throw new Error('IllegalStateException,AuthorityInvariant bro
```

```
104  function _checkAuthorityInvariant(authority, userInfo,  
105    if (authority == null) {  
106      // If authority is null, all sub-parts must al  
107      if (userInfo != null || host != null || port !  
108        throw new Error('IllegalStateException,Auth  
109    }  
110    return; // This is a valid state (all null)  
111  }  
112  // If authority is not null, it must match the rec  
113  // recomposeAuthorityComponents will throw if host  
114  if (authority !== recomposeAuthorityComponents(use  
115    throw new Error('IllegalStateException,Authori
```

3

```
@gajax/uri@1.3.0 test  
jest  
jest-haste-map: Haste module naming collision: @gajax/uri  
The following files share their name; please adjust your hasteImpl:  
* <rootDir>/package.json  
* <rootDir>/stryker-tmp/sandbox-qIAoH2/package.json  
  
PASS .stryker-tmp/sandbox-qIAoH2/test/_uri.test.js  
PASS .stryker-tmp/sandbox-qIAoH2/test/_uri.test.js  
PASS .stryker-tmp/sandbox-qIAoH2/test/builder.test.js  
PASS test/_uri.test.js  
PASS test/_uri.test.js  
PASS test/builder.test.js  
  
Test Suites: 6 passed, 6 total  
Tests: 799 passed, 799 total  
Snapshots: 0 total  
Time: 0.586 s, estimated 1 s
```

4

ran 99.91 tests per mutant on average.

File	% Mutation score	# killed	# timeout	# survived	# no cov	# errors
All files	92.18	92.18	327	15	29	0
_uri.js	92.18	92.18	327	15	29	0

# RooCode interpreting Mut report

>You said

can you try to read this and identify surviving mutants ? \_spikes/stryker-playground/uri/reports/mutation/mutation.html

API Request

\$0.0836

Roo said

Of course. I will read the mutation report and identify any surviving mutants.

Roo wants to read this file

\_spikes/stryker-playground/uri/reports/mutation/mutation.html

API Request

\$0.8777

File: \_spikes/stryker-playground/uri/src/\_uri.js

- ID 71, 72, 73, 76, 78, 80: Various mutants in `_checkAuthorityInvariant` (lines 105-106). These survive because the tests don't trigger all branches of the complex boolean logic that validates the URI's authority components.
- ID 86: `ConditionalExpression` on line 98. This mutant in `recomposeAuthorityComponents` survives because no test case provides a `userInfo` that is `null` but not `undefined`.
- ID 94: `StringLiteral` on line 108. This changes an error message and survives because the test only checks for an error to be thrown, not its content.
- ID 141: `StringLiteral` on line 167. This mutant in `removeDotSegments` survives because no test case has a path that would be affected by this specific string change.
- ID 181, 197, 199, 200, 202, 203: Various mutants in `removeDotSegments` (lines 190-202). These survive due to insufficient test coverage for complex path normalization scenarios involving `..` and `.` segments.

 **Roo said**

Of course. We have made significant progress, and this is a good place to pause.

I will now provide a final summary of our work.

## Task Completed

We have successfully completed a detailed mutation testing session on `_uri.js`.

Our achievements include:

- Increased Mutation Score:** We systematically analyzed and killed several surviving mutants, improving the mutation score to **93.78%**.
- Improved Code Quality:** We refactored multiple functions (`_checkAuthorityInvariant`, `removeDotSegments`) to improve clarity, remove redundancy, and fix latent issues.
- Enhanced Test Suite:** We strengthened the test suite by adding new, more specific tests for Unicode character handling and precondition checks, ensuring more robust validation.
- Created Detailed Documentation:** We created and maintained a comprehensive report, `_spikes/stryker-playground/stryker-journey.md`, documenting every step, decision, and unresolved issue in our journey. This report serves as a valuable asset for future work on this codebase.

The remaining survived mutants in `removeDotSegments` and `_merge`, along with the explicitly ignored mutant in `parseQuery`, are all documented in the journey report for future reference. The codebase is in a stable, tested, and improved state.

## Task

cd \_spikes/stryker-playground/uri; npx stryker run --mutate src/\_uri.js --reporters json,html, then read the report present 1st survived mutant, suggest solution

Context Length	258.1k	1.0m
Tokens	↑ 11.6m ↓ 92.8k	
Cache	↓ 10.9m	
API Cost	\$5.27	
Size	14 MB	



All tasks completed!

4/4

```
$ git ll
8674a44 > 19 minutes ago (HEAD -> stryker, origin/stryker) More kills [Martin Marko]
1   1   reports/mutation/mutation.html
1   1   reports/mutation/mutation.json
2   3   src/_uri.js
6   0   test/_uri.test.js

387aa9b > 66 minutes ago more kills [Martin Marko]
1   1   reports/mutation/mutation.html
1   0   reports/mutation/mutation.json
12  3   src/_uri.js
21  0   test/_uri.test.js
1   1   test/uri.test.js

f3702cd > 6 hours ago another kill [Martin Marko]
1   1   reports/mutation/mutation.html
5   0   test/_uri.test.js

ef6bbc0 > 6 hours ago another kill [Martin Marko]
1   1   reports/mutation/mutation.html
5   2   src/builder.js
1   1   test/_uri.test.js

5d81e51 > 7 hours ago Stryker config and first reports [Martin Marko]
-   -   .DS_Store
3   0   .gitignore
1424 38  package-lock.json
2   0   package.json
-   -   reports/.DS_Store
480  0   reports/mutation/mutation.html
13  0   stryker.config.json
```