openQA Project - action #71857

coordination # 80142 (Blocked): [saga][epic] Scale out: Redundant/load-balancing deployments of openQA, easy containers, containers on kubernetes

coordination # 55364 (Resolved): [epic] Let's make codecov reports reliable

flaky/unstable/sporadic test coverage from t/34-developer_mode-unit.t

2020-09-24 15:54 - okurz

Status:	Resolved	**Start date:**	2020-09-24
Priority:	High	**Due date:**	
Assignee:	cdywan	**% Done:**	0%
Category:	Organisational	**Estimated time:**	0.00 hour
Target version:	Ready		
Difficulty:			

Description

Observation

See https://codecov.io/gh/os-autoinst/openQA/pull/3417/changes from https://github.com/os-autoinst/openQA/pull/3418

Changes in t/lib/OpenQA/SeleniumTest.pm:wait_for_element

```
368:         return 0;
```

Changes in t/34-developer_mode-unit.t:wait_for_finished_handled

```
101:             });
102:         Mojo::IOLoop->one_tick;
103:         Mojo::IOLoop->remove($timer);
```

Acceptance criteria

- **AC1:** t/34-developer_mode-unit.t has 100% reliable statement coverage

Suggestions

- Collect coverage of the single test t/34-developer_mode-unit.t locally
- Optional: Reproduce the flakyness
- Ensure within the test that all code is explicitly called
- Extend to cover the line in t/lib/OpenQA/SeleniumTest.pm as well

History

#1 - 2020-10-09 13:24 - cdywan

- Status changed from Workable to In Progress

- Assignee set to cdywan

I'll give it a go

#2 - 2020-10-09 17:02 - okurz

awesome \o/

#3 - 2020-10-19 14:55 - cdywan

I suppose *Progress* was misleading since I didn't get far yet and mostly started looking into producing coverage results on my machine (i.e. the first of the *suggestions*).

#4 - 2020-10-20 12:43 - okurz

I would consider that part of "progress" so all good :)

#5 - 2020-10-21 15:31 - cdywan

https://github.com/os-autoinst/openQA/pull/3480

#6 - 2020-11-03 12:09 - okurz

mergify finally merged https://github.com/os-autoinst/openQA/pull/3480 .
https://codecov.io/gh/os-autoinst/openQA/src/master/t/34-developer_mode-unit.t shows 99.48% statement coverage. Created
https://github.com/os-autoinst/openQA/pull/3502 to reach 100% statement coverage in the test module. To ensure we have AC1 covered I suggest to monitor tests in circleCI in both "master" and from pull requests and crosscheck if the coverage analysis still shows flaky coverage for the code parts mentioned in the initial ticket description.

EDIT: https://github.com/os-autoinst/openQA/pull/3502 is merged meanwhile. My recommendations above still hold.

#7 - 2020-11-03 17:57 - okurz

It seems that something, maybe your changes, have made t/34-developer_mode-unit.t less stable, e.g. see the failure

https://app.circleci.com/pipelines/github/os-autoinst/openQA/4717/workflows/fe05b340-2332-4085-83e0-37a51e501c63/jobs/45023

```
[17:32:04] t/34-developer_mode-unit.t ............................... 2/? Bailout called.  Further testing st
opped:  test exceeds runtime limit of '60' seconds
FAILED--Further testing stopped: test exceeds runtime limit of '60' seconds
```

can you please look into this?

#8 - 2020-11-04 09:39 - okurz

- Priority changed from High to Urgent

I saw this in other pull requests as well, e.g.
https://build.opensuse.org/package/live_build_log/devel:openQA:TestGithub:PR-3505/openQA/openSUSE_Factory/x86_64 failing the same, I force merged the PR https://github.com/os-autoinst/openQA/pull/3505 now for the same reason. So setting to "Urgent".

#9 - 2020-11-05 18:42 - okurz

- Due date set to 2020-11-06

cdywan can you look into the issue that is impacting most if not all tests on circleCI and/or OBS right now? If not feel free to ask someone else to take over

#10 - 2020-11-05 19:55 - okurz

- Assignee changed from cdywan to okurz

please forgive me when I rip this out of your hopefully-not-so-cold hands but this is impacting (likely) every PR and failing package builds :D

Using https://github.com/okurz/scripts/blob/master/count_fail_ratio I ran

```
count_fail_ratio prove -l t/34-developer_mode-unit.t
```

and found

```
## count_fail_ratio: Run: 20. Fails: 8. Fail ratio 40%
```

with the failures being stopped by the timeout and increasing timeout does not help – as https://github.com/os-autoinst/openQA/pull/3510 already showed yesterday.

Reverting your PR shows

```
## count_fail_ratio: Run: 20. Fails: 0. Fail ratio 0%
```

Created
https://github.com/os-autoinst/openQA/pull/3518
but you are very welcome to take the ticket back after we have that quick-fix revert included and you can take a look again.

#11 - 2020-11-06 08:12 - cdywan

okurz wrote:

> please forgive me when I rip this out of your hopefully-not-so-cold hands but this is impacting (likely) every PR and failing package builds :D

That is absolutely fine. In the past you didn't mind the queue being broken for days and I was the one complaining. Happy to see that you want to

unblock fast

#12 - 2020-11-06 08:14 - cdywan

- Assignee changed from okurz to cdywan

- Priority changed from Urgent to High

#13 - 2020-11-06 09:08 - okurz

https://build.opensuse.org/package/show/devel:openQA/openQA is fine again as well as tests in https://app.circleci.com/pipelines/github/os-autoinst/openQA?branch=master

cdywan wrote:

> That is absolutely fine. In the past you didn't mind the queue being broken for days

well, that was a bit different though as I think checks in OBS were not failing so the package was still ok and usable, we just could not have green checks in PRs. So the impact is limited to contributors of openQA, not users.

#14 - 2020-11-12 18:03 - cdywan

- Description updated

- Due date changed from 2020-11-06 to 2020-11-13

Although much of my logs was lost, some notes from my attemps to reproduce coverage changes:

- I thought maybe I could copy cover_db/cover.14 around, but it seems incompatible with -test
- -report compilation which produces messages like Uncovered statement at ... line ... and seems nicer in a terminal than html_basic (HTML with syntax highlighting). Although it won't tell you what the statement is either.
- We could add explicit tests for test utils... but in reality coverage seems to report misses even on the tests (e.g. Uncovered statement at t/01-test-utilities.t line 42) and I'm not sure that this is solving the problem

https://github.com/os-autoinst/openQA/pull/3552

#15 - 2020-11-12 20:17 - okurz

I am not sure if your notes tell that you are still wondering what's going on and what to do or do you have a well layed out plan :D

I ran while [$? = 0]; do rm -rf cover_db/ ; make coverage TESTS=t/34-developer_mode-unit.t | grep 't/34-developer_mode-unit.t.*100\.0'; done locally to reproduce the problem.

With your patch applied I get:

```
95      sub wait_for_finished_handled {
96          # wait until the finished event is handled (but at most 5 seconds)
97          my $timer = Mojo::IOLoop->timer(
98              5.0 => sub {
99  0               Mojo::IOLoop->stop;
100 4           });
101 4       Mojo::IOLoop->one_tick;
102 4       Mojo::IOLoop->remove($timer);
103 4       $finished_handled_mock->unmock_all();
104 4       is($finished_handled, 1, 'finished event handled within 5 seconds');
```

(hard to copy here).

this means that the command Mojo::IOLoop->stop; isn't called, or not reliably always called.

You can write Mojo::IOLoop->stop; # uncoverable statement to just ignore if that single line is covered or not.

#16 - 2020-11-13 16:41 - cdywan

okurz wrote:

> I am not sure if your notes tell that you are still wondering what's going on and what to do or do you have a well layed out plan :D

My notes are general workflow items, that arguably belong in the epic, since I feel like verifying fixes like this should be more straightforward.

> I ran while [$? = 0]; do rm -rf cover_db/ ; make coverage TESTS=t/34-developer_mode-unit.t | grep 't/34-developer_mode-unit.t.*100\.0'; done locally to reproduce the problem.

Yes, this is the kind of command/pattern I was trying to come up with, something easy enough for this and other coverage fixes to verify if they work/ are reliable.

And it shows t/34-developer_mode-unit.t 100.
0 100.0 with the current version of my brach.

I'd alter it to make coverage TESTS=t/34-developer_mode-unit.t | grep 't/34-developer_mode-unit.t.' since otherwise there is just nothing if you ran it on e.g. the original version of my branch. But that might be down to personal preference.

Output of affected lines would be even better.

> this means that the command Mojo::IOLoop->stop; isn't called, or not reliably always called.

Yeah, never seems to be called in my runs. But it looks like sub{} avoids that problem - the current code in master actually has no code in it, I was adding it to try and make codecov happy.

#17 - 2020-11-13 19:03 - okurz

PR merged after I tried it myself locally and it looked good.

Before we close this ticket I would like to reflect on https://progress.opensuse.org/issues/71857#note-7 . Maybe we can understand how we missed the problem initially and then hopefully even find an improvement.

#18 - 2020-11-16 09:21 - cdywan

okurz wrote:

> PR merged after I tried it myself locally and it looked good.
>
> Before we close this ticket I would like to reflect on https://progress.opensuse.org/issues/71857#note-7 . Maybe we can understand how we missed the problem initially and then hopefully even find an improvement.

The timeout was caused by moving from

```
my $timer = Mojo::IOLoop->timer(5.0 => sub {
    });
Mojo::IOLoop->one_tick;
Mojo::IOLoop->remove($timer);
```

to wait_for_or_bail_out { $finished_handled } 'finished event'; which has a different wait, one is event loop-based, the other one is a Perl loop.

Maybe it's worth seeing if we want to use the event loop elsewhere?

#19 - 2020-11-16 09:22 - cdywan

- Status changed from In Progress to Feedback

#20 - 2020-11-16 11:31 - cdywan

- Due date changed from 2020-11-13 to 2020-11-16

#21 - 2020-11-16 13:00 - okurz

ok, I understood how it could break but: Maybe we can understand how we missed the problem initially and then hopefully even find an improvement in our testing or process? So not really about t/34-developer_mode-unit.t

In #71857#note-6 I mentioned: "mergify finally merged https://github.com/os-autoinst/openQA/pull/3480 . […] To ensure we have AC1 covered I suggest to monitor tests in circleCI in both "master" and from pull requests and crosscheck if the coverage analysis still shows flaky coverage for the code parts mentioned in the initial ticket description."
So maybe your test change worked sporadically hence tests were all good in the PR but would not have shown as successful when executing the test module locally multiple times. As you did not mention in neither ticket nor PR that you would have executed the test multiple times locally I assume you have not done so, have you?

- *Improvement idea 1*: Either include in our review guidelines to ask for multiple executions of test modules. I would not know how to realistically implement the "stability test" within each PR as we would need to wait very long for circleCI runs to complete, am I right?

I think like i suggested closely monitoring tests in circleCI would have shown the regression but this also after the merge.

One more point: I pointed out the regression and you did not react, then I set the ticket to "Urgent" but still did not react until I took over. Just very objectively I wonder: Have you received the notification? Have you missed the update? Was something else distracting you? Should others than me have seen the ask for help?

#22 - 2020-11-16 16:44 - cdywan

okurz wrote:

> In #71857#note-6 I mentioned: "mergify finally merged https://github.com/os-autoinst/openQA/pull/3480 . […] To ensure we have AC1 covered I suggest to monitor tests in circleCI in both "master" and from pull requests and crosscheck if the coverage analysis still shows flaky coverage for the code parts mentioned in the initial ticket description."
> So maybe your test change worked sporadically hence tests were all good in the PR but would not have shown as successful when executing the test module locally multiple times. As you did not mention in neither ticket nor PR that you would have executed the test multiple times locally I assume you have not done so, have you?

Yes, I didn't observe it ever failing locally.

- *Improvement idea 1*: Either include in our review guidelines to ask for multiple executions of test modules. I would not know how to realistically implement the "stability test" within each PR as we would need to wait very long for circleCI runs to complete, am I right?

> I think like i suggested closely monitoring tests in circleCI would have shown the regression but this also after the merge.

I also re-triggered tests to be sure it wasn't a 33% chance like I've seen before. But last time I baked that into the makefile I got some "confused" comments :-D

> One more point: I pointed out the regression and you did not react, then I set the ticket to "Urgent" but still did not react until I took over. Just very objectively I wonder: Have you received the notification? Have you missed the update? Was something else distracting you? Should others than me have seen the ask for help?

I think I mentioned it before. We have had other flaky test regressions in master for days if not weeks. It seems like you're deciding which ones are more critical in a way I don't understand.

#23 - 2020-11-16 18:02 - okurz

- Status changed from Feedback to Resolved

cdywan wrote:

> […]
> I think I mentioned it before. We have had other flaky test regressions in master for days if not weeks. It seems like you're deciding which ones are more critical in a way I don't understand.

Yeah, sorry. I am not a deterministic machine ;) But here very clearly I raised the priority and eventually reverted because suddenly many or most tests failed in t/34-developer_mode-unit.t

Ok, but it seems I am annoying you a lot so let's leave it and set to "Resolved".

#24 - 2020-11-24 10:19 - okurz

- Due date deleted (2020-11-16)