openQA Project - action #72130

check and/or reduce runtime of t/api/04-jobs.t

2020-09-30 09:40 - okurz

Status:	Workable	**Start date:**	2020-09-30
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Feature requests	**Estimated time:**	0.00 hour
Target version:	future		
Difficulty:			

Description

Observation

https://app.circleci.com/pipelines/github/os-autoinst/openQA/4393/workflows/73728ebb-2783-487d-bf09-79f3ee2270ef/jobs/42262 shows

```
[06:58:59] t/api/04-jobs.t ..................... 27/? Bailout called.  Further testing stopped:  te
st exceeds runtime limit of '120' seconds
FAILED--Further testing stopped: test exceeds runtime limit of '120' seconds
```

Acceptance criteria

- **AC1:** t/api/04-jobs.t internal timeout is significantly lower than currently, e.g. 20s

Suggestions

- Profile runtime of test and identify points for improvement
- Optimize test, potentially split
- Reduce timeout in test module

Related issues:

Copied from openQA Project - action #72127: check and/or reduce runtime of t/...	**Resolved**	**2020-09-30**

History

#1 - 2020-09-30 09:40 - okurz

- Copied from action #72127: check and/or reduce runtime of t/44-scripts.t added

#2 - 2020-10-02 09:20 - okurz

- Status changed from In Progress to Feedback

https://github.com/os-autoinst/openQA/pull/3437

#3 - 2020-10-06 11:04 - okurz

- Description updated

- Category changed from Concrete Bugs to Feature requests

- Status changed from Feedback to Workable

- Assignee deleted (okurz)

- Priority changed from High to Low

PR is merged. With this the timeout limit should be more stable. Updated ticket to describe next tasks in detail, i.e. optimize test module

#4 - 2020-11-02 15:53 - mkittler

- Status changed from Workable to In Progress

- Assignee set to mkittler

#5 - 2020-11-02 16:14 - mkittler

- Status changed from In Progress to Workable

- Assignee deleted (mkittler)

After your PR the time limit is at 30 seconds which seems good considering the test needs a little bit more than 20 seconds.

I wouldn't split the test because that just adds more database initialization overhead. There are no points where this test "hangs" waiting for something. As such I couldn't find much room for optimization and I wouldn't rework the entire test just so squeeze a few seconds out of it.

#6 - 2020-11-04 17:34 - okurz

- Target version changed from Ready to future

Thank you for looking into this. I think you have not conducted a profiling run though and I consider 20s of runtime for a local non-UI test rather long still. Also, the test module is 1.3k lines long so also not that easily maintainable. Splitting is not the only option but still we could move out the "extra test" parsing test parts into a separate test module. I consider it more common for developers to run individual test modules and leave the complete test set to CI runs by default.

Nevertheless it is ok if we come back to this some time later, not now :)