# How to contribute to openQA

Also see:
os-autoinst/openQA/tree/master/docs

*2025-02-21*
*by Robert Richardson*

# openQA Overview

- Automated framework for combined testing of operating systems and applications.
- Contributions can improve:
    i. **Feature Set**
    ii. **UI/UX**
    iii. **Stability**
    iv. **Performance**
    v. **Test coverage**
    vi. ...

# We will look at

1. How to find a **issue**
2. How to create a **fork**
3. How to find a **solution** (based on a short example)
4. How to create a **PR**

# 1. How to find a issue?

**Progress and Github**

- We organize our work on progress.opensuse.org
    i. Workable Issues
    ii. Beginner Issues
- You might be used to github issues, we also have those

# Repositories

- os-autoinst:
  - executes tests and starts/controls the SUT e.g. using QEMU
- openQA:
  - web UI, accompanying daemons, scheduler, worker, documentation, support scripts
- os-autoinst-distri-opensuse:
  - the actual tests conducted on http://openqa.opensuse.org
- os-autoinst-needles-opensuse
- os-autoinst-distri-example...

# 2. How to create a fork?

### Setting up the Development Environment

1. **Fork repo(s)** on GitHub and **Clone your fork**:

   ```
   git clone https://github.com/YOUR-USERNAME/openQA.git
   cd openQA
   ```

2. **Set up upstream** to stay in sync:

   ```
   git remote add upstream https://github.com/os-autoinst/openQA.git
   ```

3. **Install dependencies** and set up the development environment as per the project's README.

# Example issue: poo#134840

**What needed improvement?**

- In the **Live View**, there was **no loading indicator**.
- Users might think the liveview feature is broken.

*Goal*:

- **Show a loading spinner** while the content loads.

# 3. Finding a solution (TDD Approach)

1. **Write a failing test** covering the new feature/bug fix.
2. **Implement** a minimal solution to pass the test.
3. **Refactor** if needed.
4. **Check** all tests pass again.

# 3.1 Create one or more (sub)test

***t/ui/18-tests-details.t***

```
subtest 'liveview loading animation is displayed' => sub {
  $driver->find_element_by_link_text('Live View')->click();
  my $loading = $driver->find_element('#liveview-loading');
  ok($loading->is_displayed(), 'spinner is visible');
};

subtest 'loading animation hides after live view starts' => sub {
  my $loading = $driver->find_element('#liveview-loading');
  $driver->execute_script("loadCanvas(...)");
  wait_until { !$loading->is_displayed() }, 'spinner is hidden';
};
```

- Initially fails because there's no spinner or logic to hide it.

# 3.2 Adding the spinner (HTML)

***templates/webapi/test/live.html.ep***

```
...
  <div id="canholder">
+   <div id="liveview-loading">
+     <i class="fa fa-circle-o-notch fa-spin fa-4x"></i>
+   </div>
    <canvas id="livestream"></canvas>
  </div>
...
```

- A simple fork-awesome **spinner** element above the canvas.

# 3.3 Positioning the spinner (CSS)

***assets/stylesheets/result_preview.scss:***

```
#canholder {
  position: relative;
  text-align: center;
  width: 100%;
}

#liveview-loading {
  position: absolute;
  top: 50%; left: 50%;
  transform: translate(-50%, -50%);
  z-index: 10;
}
```

- Centers the spinner on top of the canvas.

# 3.4 Hiding the spinner (JS)

***assets/javascripts/running.js:***

```
...
function loadCanvas(canvas, dataURL) {
  let scrn = new Image();
  scrn.onload = () => {
+    document.getElementById('liveview-loading').style.display = 'none';
+    canvas.getContext('2d').drawImage(scrn, 0, 0);
  };
  scrn.src = dataURL;
}
...
```

- Once the image is loaded, hide the loading indicator.

# 3.5 Verifying functionality

- **Run** the tests (e.g. `prove -v t/ui/18-tests-details.t`).

also see:

- `make test` to run the entire test suite.
- `make coverage` to generate a code-coverage report
- `prove -v` more detailed outputs.
- manual tests

# 4. Create a PR

1. **New branch**:

```
git checkout -b feature-liveview-spinner
```

2. **Tidy** and **Commit**:

```
tools/tidyall -a
git add .
git commit # add issue link & read: https://cbea.ms/git-commit/
```

3. **Push**:

```
git push origin feature-liveview-spinner
```

4. **Open** Pull Request on GitHub.

# Conclusion

- We **identified** the issue.
- Applied **TDD** to add and test a simple spinner.
- Committed and opened a **Pull Request**.

# How to get involved

## Start Contributing Today

- **Check out the full Contributing Guide:**
  openQA Developer Guide
- **Check out beginner-friendly issues**:
  openQA Beginner Issues
- Read Livs Blogpost Getting Started with openQA Development
- **Join the discussion**:

  #openqa:opensuse.org on Matrix

🚀 *Your small contribution can make a big difference!*