Product Owner Guide

A rough guide for those who are product owners on a dwyl project.

Contents

What is a Product Owner?

A Product Owner (PO) is someone who represents the views and needs of the target audience, stakeholders and the business they represent in agile scrum methodology. They must understand and communicate the vision of the product to the rest of the team.

They are the liaison between customers and the development team. This means that they are responsible for working with stakeholders and user groups to determine their needs. Then through collaboration with the Scrum Master and developer team they prioritise and discuss these ideas moulding them into features to be built.

PO Responsibilities:

1. Owner and creator of the product backlog 👑
2. Adding new features to the backlog. 🆕
3. Describing the user-persona(s) in as much detail as possible to make the "pain" they feel *real* (*especially useful when the team is remote and don't often meet with the end-users*) 📝
4. Clearly define, understand and *empathise* with the end-user 👩 💡
5. Approving or critiquing development work for release. ✅

6. Involving customers or users and getting their feedback to continuously shape the product development 👍 👎
7. Maintaining and *updating* the backlog: adjusting the features and *priority* as needed after each sprint or iteration. 🔺 🔻
8. Attach a *business* or *customer* value to each story/task in the backlog to ensure that we are building features that are genuinely wanted/needed by users 💸
9. Have a clear picture of "competitors" and the "market" for similar products/service to avoid "re-inventing" something that already exists! (*i.e. don't try to replace app/site ABC, you are wasting your time! instead find something they aren't doing or are doing poorly and focus on that!*) 🔍
10. Be familiar with *existing* UX/UI through personal use of other/existing apps/services to know what can be done/re-used ♻️
11. Trust the expertise of your technical team on matters of technical importance (although feel free to ask questions to learn more about *why* certain things matter when you may not be familiar with them) 👊
12. Note, in order to be able to perform these responsibilities it's important that POs have been given the authority and means necessary to complete them 💪

Glossary of Key Terms

- **Scrum Master**: tracks team progress, liaises with product owner on requirements, lends expertise on managing end to end projects, ensures features are tested when delivered and manages deployments in conjunction with technical architect and the team.
- **Product backlog:** a prioritised features / to do list for the product found on github. See our ['What is the backlog?' section]() for more detail.
- **Sprint Backlog**: the backlog that is aiming for completion within a given sprint period.
- **Sprint**: a 1-3 week working lifecycle of planning, building, presentation and reflection. We tend to work in 2 week (10 working day) sprints.
- **UI / User Interface**: how the site looks from a design perspective.
- **User testing:** interviews, focus groups and other forms of research conducted with potential/actual users of your product to give feedback and provide insight into what the user's needs are. See https://github.com/dwyl/learn-user-experience-testing to learn more about user testing.
- **Milestone**: a visual way to group issues in github, at dwyl we create one milestone per sprint so we have somewhere to see all of the issues which we estimate to complete by the end of the sprint. See https://github.com/dwyl/github-reference for more.

- To see definitions on the **Sprint Ceremonies** such as **stand ups, sprint demos, retrospectives and sprint planning** go to: https://github.com/dwyl/process-handbook/#ceremonies

What is the backlog?

The backlog is your project's to do list, a list of all the features and things that need doing. This list contains not only the things that you're planning imminently but also ideas you want to discuss in the future or feature requests that are yet to be validated. As not all of the issues in the backlog are ready to work on we use [labels]() to indicate things like priority or whether something is blocking an issue from being worked on. With the labels as a visual cue you can scan the backlog and get a flavour for what's important on the project at present.

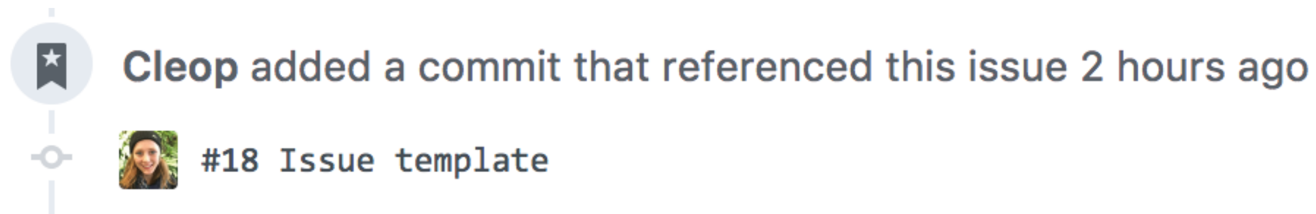
Where can I find the backlog?

In order to keep track of progress on a project we use github, see https://github.com/dwyl/github-reference to learn more about using github. To see the backlog click on the 'issues' tab at the top of your project:

This will display a list of all of the individual things to do in your project. Something which needs doing on github is called an 'issue'.

Why github when we could use trello / basecamp / <-insert-other-site-here->?

We use github because unlike other platforms it acts as a **single source of truth** 👼. What does this mean? It means that github is the one place where the whole team can see the status of the project at a given time. Why can't another site be the single source of truth? Github is the site our developers use to upload their code to, for this reason you can see live updates of their code when it is added to the site, like this:



If we were to use another platform like trello then we'd have to manually update the progress on issues which would not only take time ⏳ but also create a lag between things being done and everyone being able to see 👀 that they've been done. Then we would have 2 records of the status of the project, the truth as told by trello and then the truth as told by the site storing our code. We think this is an unnecessary complication so we keep everything on github! ❤️ :octocat: Still want to know more about why we use github? Go to https://github.com/dwyl/github-reference#why

Product Owners should create issues rather than dwylers

Why?

- As owner of the backlog the PO should know what's in it and how it's been outlined.
- It ensures the team is on the same page.
  - The PO should describe what their imagining of the issue is in their own words. Had another team member written it assumptions or misinterpretations could be made. When it's in the PO's words the team can talk through these in the sprint planning to discuss any areas that don't feel clear.
- Helps you to get familiar with Github and in the practice of using it.
  - Practice will make perfect, you'll see what kinds of questions the team may ask you which will help you realise how you can improve your issue descriptions.
- All issues must be validated by the PO, by you creating them first hand this cuts out back and forth of correction had someone else written it.

Template for making an issue

The 3 components you should include when creating an issue are:

1. A User story
2. Acceptance Criteria
3. A screenshot of a mock-up or a bug on the site (where possible)

Followed by... any other useful/ relevant information for completing the issue.

User Stories

A user story is a phrase which describes the what (you want to do), the who (the issue is for) and the why (why should this be done). To write one complete the following three starters:

As... a team member - **WHO**

I want to... see a user story on every issue I read - **WHAT**

so that... I know the context in which the issue is set to help me understand it better - **WHY**

Here's another example:

As someone in need of a new bin

I want to be able to filter on the online shopping site by 'lid type'

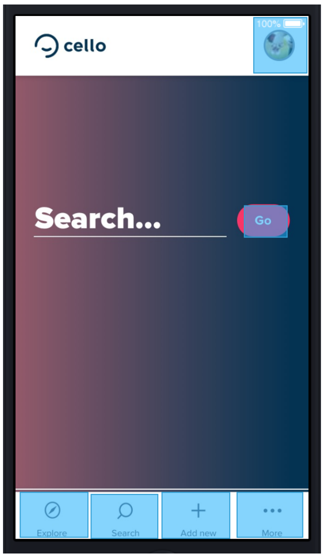
So that I can find a bin that closes securely to stop mice getting into my bin 🐭❌🗑️🧀

Acceptance Criteria

Acceptance criteria are checkboxes that a developer knows they need to complete before that issue can be considered 'done'. They are the points against which the issue will be tested.

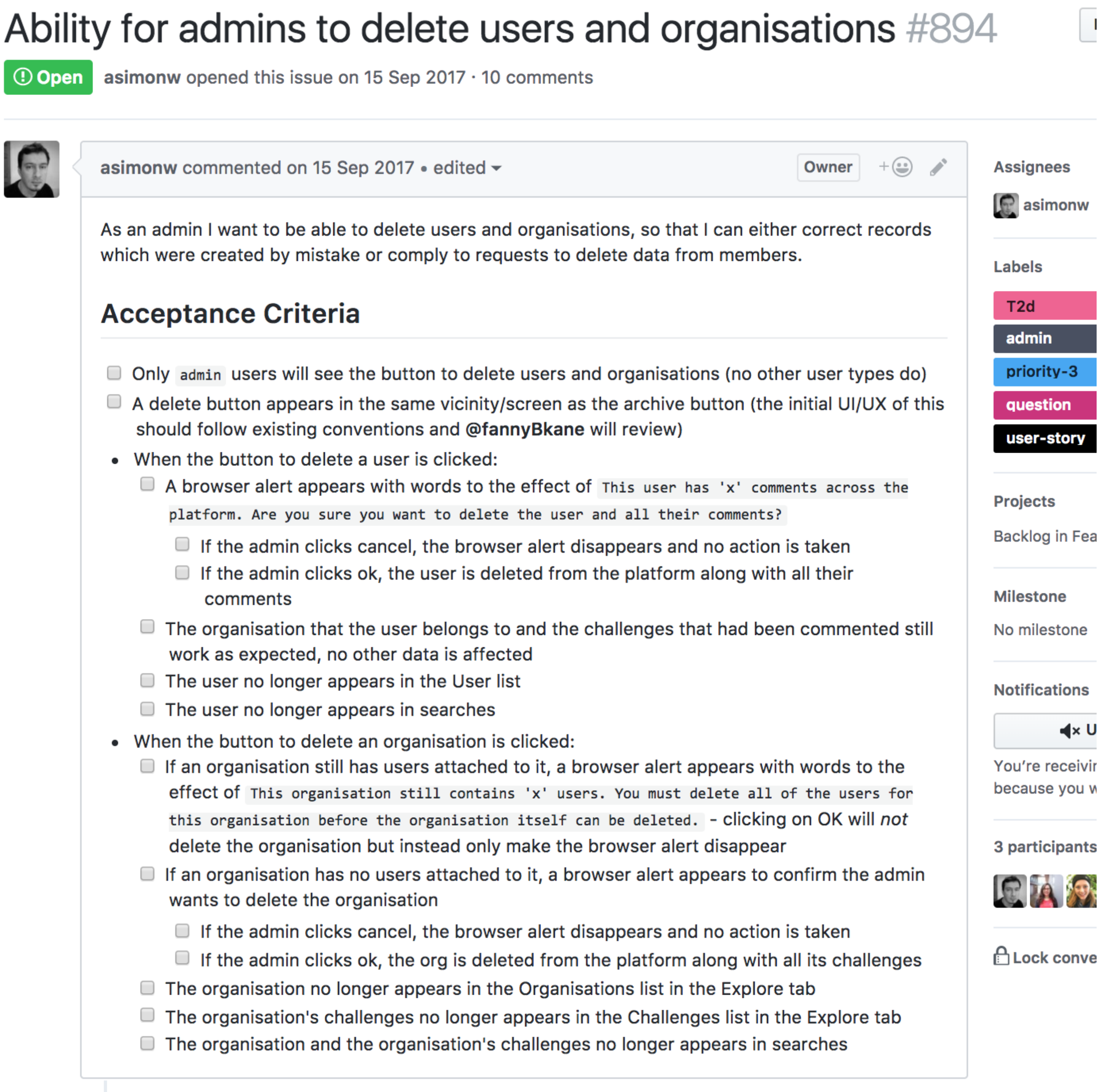
A screenshot or mock up of a new design

A picture which illustrates what the issue should be creating or a screenshot of an existing page showing where a bug is occurring. Remember to include mock ups of both mobile and desktop views for new designs. Give as much contextual detail as possible for bugs e.g. occurs on iPhone 8 and chrome on desktop - see screenshots.



Real world exemplar issue

The following issue has a clearly defined user story and detailed acceptance criteria:



Ability for admins to delete users and organisations #894

Open asimonw opened this issue on 15 Sep 2017 · 10 comments

asimonw commented on 15 Sep 2017 • edited Owner

As an admin I want to be able to delete users and organisations, so that I can either correct records which were created by mistake or comply to requests to delete data from members.

Acceptance Criteria

- [ ] Only `admin` users will see the button to delete users and organisations (no other user types do)
- [ ] A delete button appears in the same vicinity/screen as the archive button (the initial UI/UX of this should follow existing conventions and **@fannyBkane** will review)
- When the button to delete a user is clicked:
  - [ ] A browser alert appears with words to the effect of `This user has 'x' comments across the platform. Are you sure you want to delete the user and all their comments?`
    - [ ] If the admin clicks cancel, the browser alert disappears and no action is taken
    - [ ] If the admin clicks ok, the user is deleted from the platform along with all their comments
  - [ ] The organisation that the user belongs to and the challenges that had been commented still work as expected, no other data is affected
  - [ ] The user no longer appears in the User list
  - [ ] The user no longer appears in searches
- When the button to delete an organisation is clicked:
  - [ ] If an organisation still has users attached to it, a browser alert appears with words to the effect of `This organisation still contains 'x' users. You must delete all of the users for this organisation before the organisation itself can be deleted.` - clicking on OK will *not* delete the organisation but instead only make the browser alert disappear
  - [ ] If an organisation has no users attached to it, a browser alert appears to confirm the admin wants to delete the organisation
    - [ ] If the admin clicks cancel, the browser alert disappears and no action is taken
    - [ ] If the admin clicks ok, the org is deleted from the platform along with all its challenges
  - [ ] The organisation no longer appears in the Organisations list in the Explore tab
  - [ ] The organisation's challenges no longer appears in the Challenges list in the Explore tab
  - [ ] The organisation and the organisation's challenges no longer appears in searches

Assignees
asimonw

Labels
T2d
admin
priority-3
question
user-story

Projects
Backlog in Fea

Milestone
No milestone

Notifications
You're receivir
because you w

3 participants

Lock conve

Find the original issue at: https://github.com/emfoundation/ce100-app/issues/894

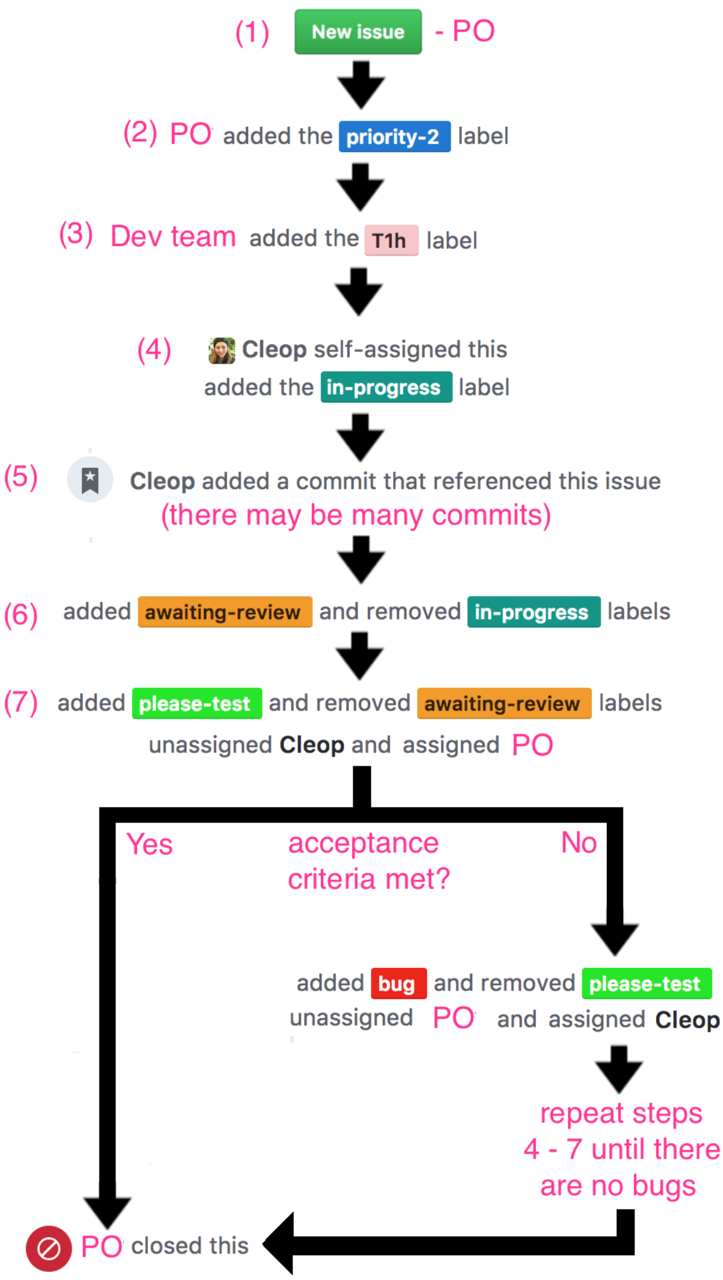
This issue doesn't have mock-ups/screenshots but note that the second checkbox says the delete button should follow existing conventions set out in the style guide.

What are priority labels?

We have 5 priority labels. Issues are worked on in priority order from 1 down to 5. `priority-1` is reserved for emergency use e.g. the live site crashing. In most situations when there are no open `priority-1` issues, `priority-2` issues are therefore the first ones to be worked on. Whilst at the other end of the scale a `priority-5` could be used for an idea for a new feature that's not been fully outlined yet.

The lifecycle of an issue

At dwyl we use github labels to help indicate the type, duration and status of an issue. The following image outlines the lifecycle of an issue using labels. In this example there are 3 actors, the Product Owner (PO), the dev team (Scrum Master and developers) and in this case the developer who has completed the issue: 'Cleop'.



To learn more about our contributing process see: https://github.com/dwyl/contributing.

When to create a new issue when dealing with bugs 🐛 and enhancements 🎀

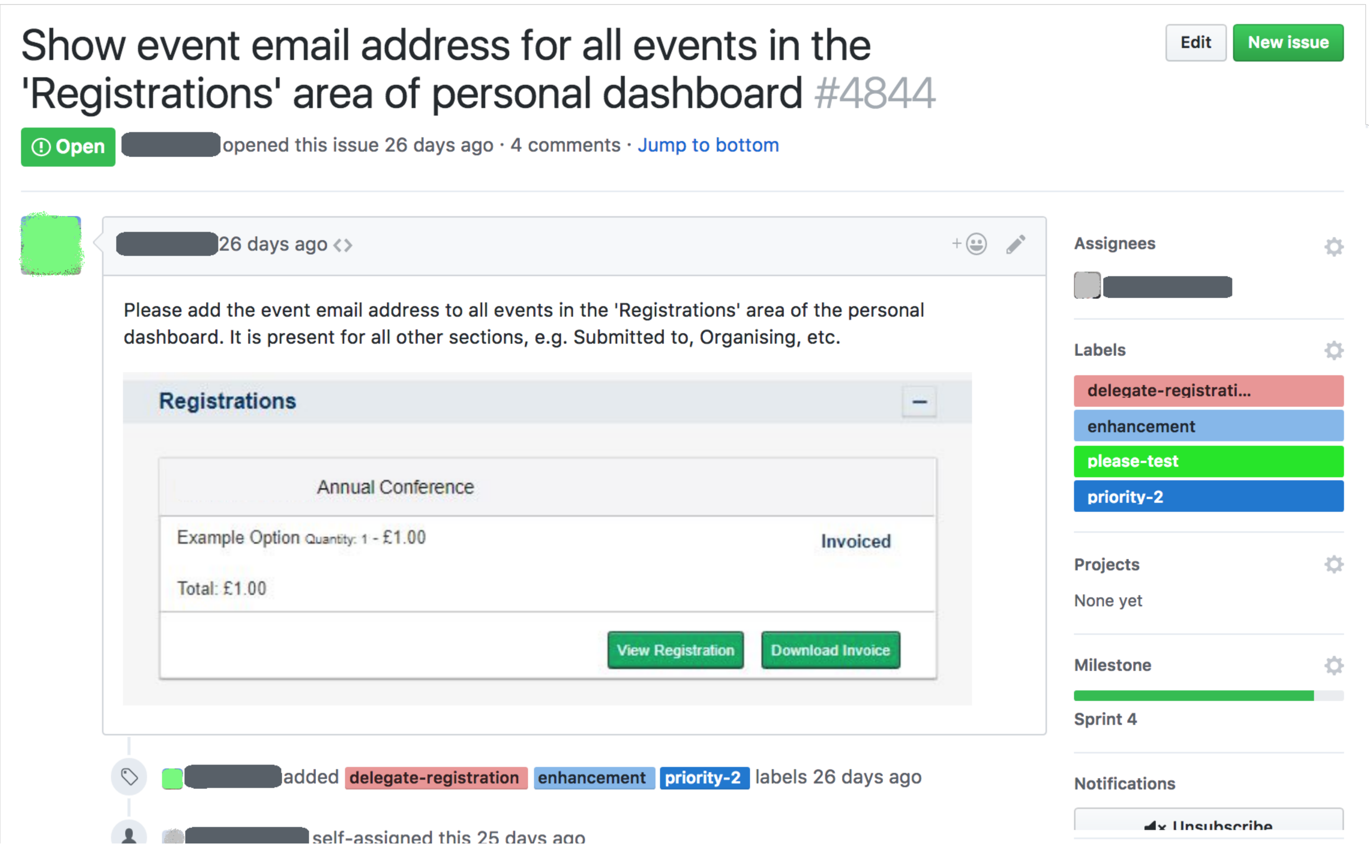
An existing issue is put into `please-test` as the team have finished working on it and all of the acceptance criteria have been fulfilled. You, the product owner test the issue and on reflection you decide that you think the designs would look better with a couple of tweaks - maybe removing the bold on the title, making the logo a bit bigger and changing the copy by a few words. What do you do in this situation?

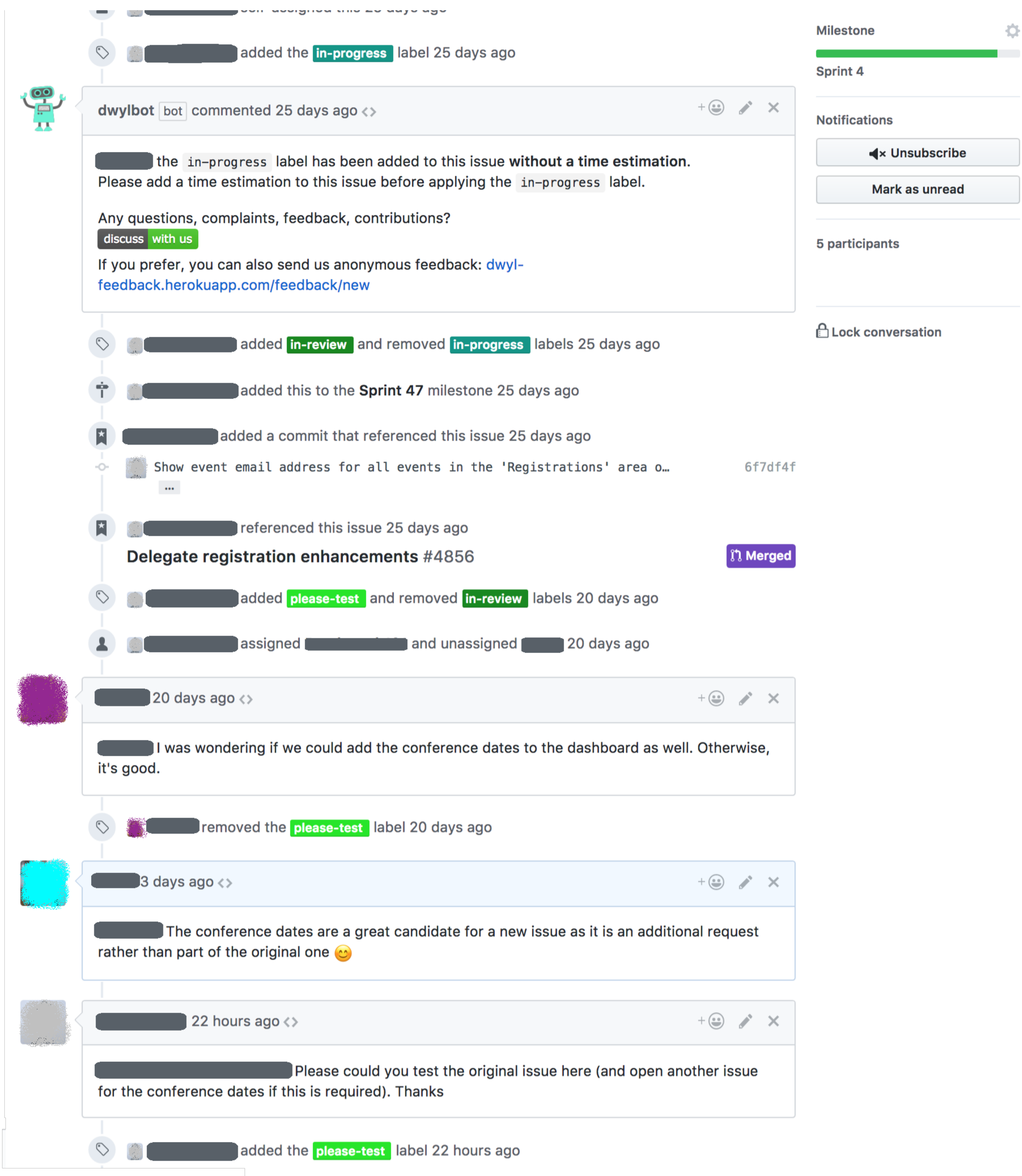
These changes might relate to the changes made in the issue but if they are fundamentally not what you asked for in the first place or not what was shown on the mock-up then they belong in a new issue **NOT** as a 'bug' or as a request at the bottom of the existing issue. Small as they seem, these issues are enhancements and new scope and they should be dealt with separately.

Why?

- These kinds of small changes add up, especially if you change your mind a couple of times or if there are a few across multiple issues.
- The title of the issue may no longer reflect the content inside making it more complicated for people to follow the chains of conversations.
- The extra changes will impact the developer's initial time estimate which is damaging for their accuracy when trying to estimate how long issues will take.
- The extra scope may push something else out of the sprint. Had you prioritised the changes as a new issue you may not have put this tweak above the other issues in the sprint.

Here's an example where a new issue should have been made:





So what changes count as bugs 🐛 then?

Bugs are places where the acceptance criteria has not been met ❎

e.g. you were meant to change the font size across all titles on this page but you missed out the last title - please can you change it?

I submitted the form but I didn't receive an email notification as specified in the acceptance criteria. Please can you fix whatever is stopping the email from being sent?

What is a staging area/site and what is it used for?

A staging site is a replica of your live (aka production) website. It is used as a first stage for testing when you deploy new code / features. It acts as a testing ground so that errors can be caught and fixed before finally deploying and updating the live site. Your staging site is meant for internal not public use.

Internal team tests on staging before the live site is updated. They spot a bug.

The bug is fixed by the dev team.

The internal team tests staging again, they're happy for the changes to be deployed to live.

The live site is updated and users are unaffected by any previous bugs.

Responsiveness standards for testing

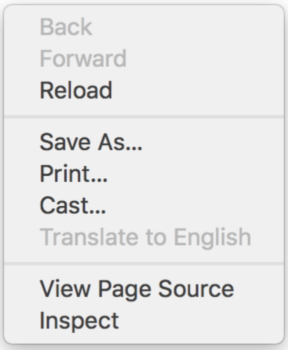
When you begin a project it is likely you will discuss target audience and device compatibility. These conversations will cover the important behaviours of your customers including browser type, device type and any other specific compatibility requirements e.g. accessibility.

Generally speaking most projects will focus on being mobile, tablet and desktop friendly and for these measures we use the inbuilt device sizes provided by browsers to test the application. These devices sizes cover a range of dimensions and a variety of the most popular iOS and android devices. Testing using these devices is not only capturing a large part of the user market but it is also a shared metric and easy way to spot and communicate bugs across a team.

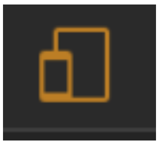
E.g. 'The subheading on the landing page is currently obscured on iPhone 6' gives the development team a quickly understandable and recognisable way of finding and fixing a given problem.

How to test using standard device sizes

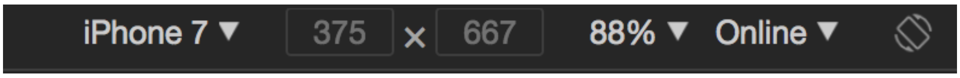
To open 'Inspect' right click on your computer with your browser open or if you are a mac user press control and then click. This menu should appear, then click on 'Inspect':

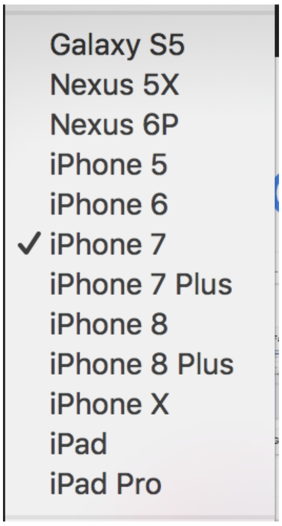


Now your browser has 'Inspect' or 'Dev tools' open you should make sure you are using the device mode. Look for this symbol in the menu:



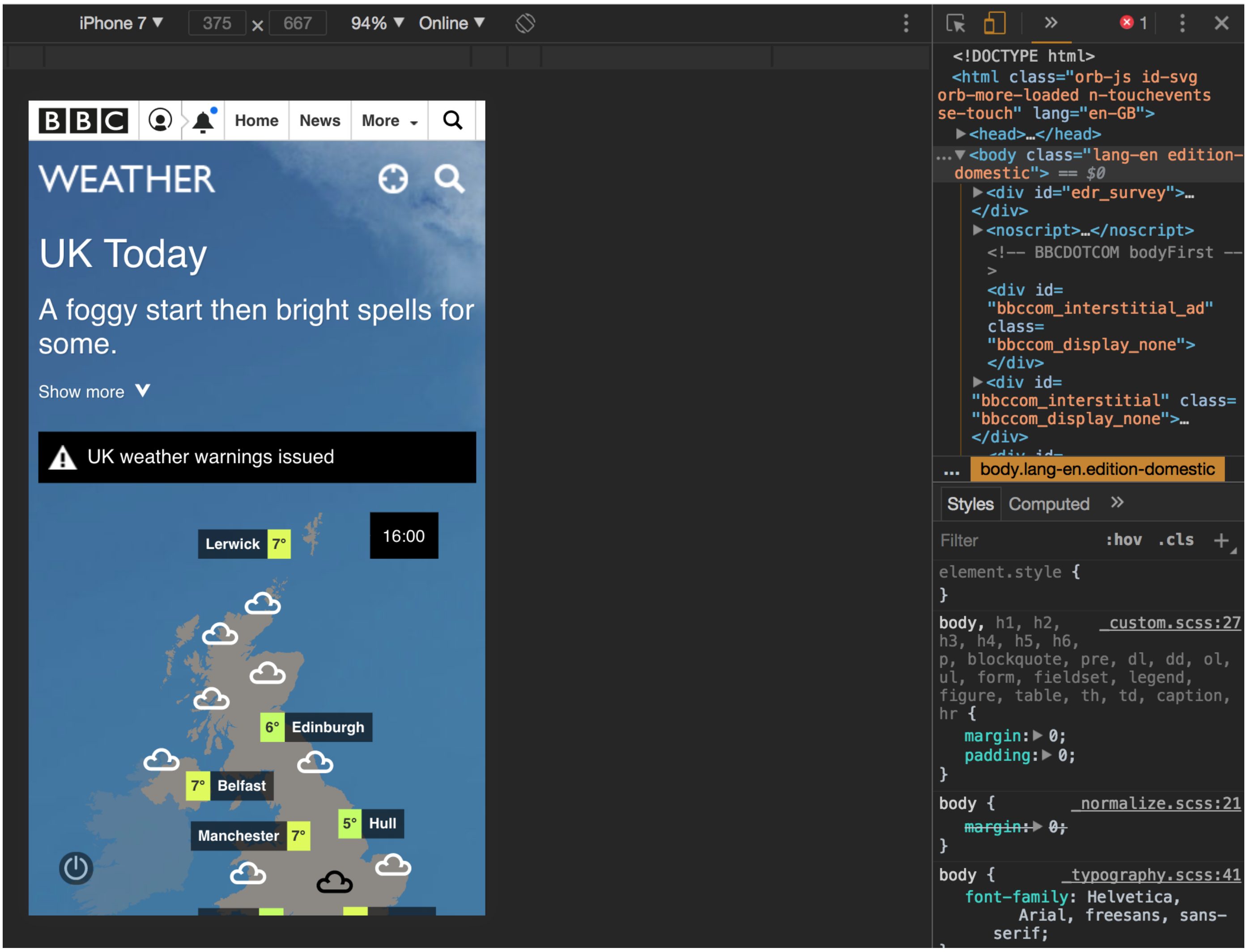
If the symbol is orange then you're already in device mode. If it is not, then click on it so it turns orange. Next, look for this bar and click on the left-hand dropdown menu to select a device:





This list is an automatically updated list contained within your browser. It contains the most popular standard devices sizes for you to test your site against.

Your page will now display according to the dimensions of the device you have selected:
E.g. iPhone 7:



iPhone 7
375
667
94%
Online
BBC
Home
News
More
WEATHER
UK Today
A foggy start then bright spells for some.
Show more
UK weather warnings issued
16:00
Lerwick 7°
6° Edinburgh
7° Belfast
5° Hull
Manchester 7°
<!DOCTYPE html>
<html class="orb-js id-svg orb-more-loaded n-touchevents se-touch" lang="en-GB">
<head>…</head>
<body class="lang-en edition-domestic"> == $0
<div id="edr_survey">… </div>
<noscript>…</noscript>
<!-- BBCDOTCOM bodyFirst -->
<div id="bbccom_interstitial_ad" class="bbccom_display_none"> </div>
<div id="bbccom_interstitial" class="bbccom_display_none">… </div>
body.lang-en.edition-domestic
Styles
Computed
Filter
:hov
.cls
element.style {
}
body, h1, h2, h3, h4, h5, h6, p, blockquote, pre, dl, dd, ol, ul, form, fieldset, legend, figure, table, th, td, caption, hr {
custom.scss:27
margin: 0;
padding: 0;
}
body {
normalize.scss:21
margin: 0;
}
body {
typography.scss:41
font-family: Helvetica, Arial, freesans, sans-serif;

iPad:

So if you come across a bug 🐛 remember to include lots of detail to help your team replicate it. Think:

- What pathway did I take to get to the error? ie. who was I logged in as? 👥 what was I trying to do when it happened?
- What device or browser was I using? 📱 💻 e.g. Internet Explorer v10 or iPhone 8 view.

Then take a screenshot of the bug and put it all into an issue to be fixed 🔧 ✨.

Here is an example of an issue that contains a description of the bug, the critical steps required to reproduce it and which devices it was experienced on:

Transitions between questions Chrome/Android #582

Open stianr opened this issue on 15 Dec 2017 · 4 comments

stianr commented on 15 Dec 2017

At last I seem to have identified a replicable version of the transitions problem that we saw earlier in the term. It seems to occur specifically on chrome for Android. I've replicated it on my own Galaxy S7 after a student showed it on her J5. If you try completing a live quiz, the transitions between questions are incredibly slow, and the screen is blank for much of the time. It can take 30 seconds for a question to appear, and seems to get worse as the quiz progresses (though that may not be a consistent effect).

iteles commented 19 hours ago — Collaborator

We have tried to replicate this unsuccessfully on a Nexus 5X but have dug up another older Android phone to try this.

It could also be a function of the number of questions in the quiz (our test quiz only had 3), the number of students taking it at once or the network itself.

stianr commented 18 hours ago • edited

Just tried again: The problem didn't occur with a short, simple quiz (2 questions, minimal text), but reappeared when running a longer quiz, so hopefully you'll be able to see it on your devices.

To try to replicate the issue, import this quiz into Quodl as lecturer: BJ4s1HZGf and then run as usual, using Chrome on Android as a student to respond. I found that by Q3 transitions were getting unworkably long.

(Fwiw, I used a Galaxy S7 with Chrome 63.0.3239.111 / Android 7.0, so I don't think it's down to processor speed or similar.)

stianr commented 18 hours ago

Okay, to isolate the cause further, I tried two separate quizzes. One had 8 two-option short questions (you can import with this code: Bkig5-TVz); the other had three extremely wordy questions (rkT_9W64G). The first ran fine, the second started stuttering at the first question. It even struggled when I reduced the number of options to 2 (but each option contained a fair amount of text). So it looks like it's the amount of text in a question and its options that is causing the problem.

Again, no problem running this on the same phone with FireFox.

Taken from: https://github.com/cul-2016/quiz/issues/582

Timing demos and deployments

The PO and the Scrum Master liaise with one another to find a time and day suitable for sprint demos and deployments. We do however recommend the following on all occasions:

- Don't deploy on a Friday, if anything breaks it may not be discovered until after the weekend or may need working on over the weekend.
- As generally we deploy after a sprint demo, for this reason we do not start 10 day sprints on Mondays as this will inevitably end with a demo and deployment on a Friday.
- To avoid bugs disrupting sprint demos the team will demo the work they completed up until a few hours before the demo. If you continue deploying to staging up until the demo you drastically increase the likelihood of a last minute rush or a semi-complete feature causing a bug and disrupting the demo.

For these reasons it is important that the PO and Scrum Master set up demos in advance so all of these factors can be taken into consideration.

What is Technical Debt?

Technical debt is the cost to
"re-work" a feature (or entire app) because it was rushed the first time it was built.

Think of Technical Debt like a "loan shark" for your project, if you take "shortcuts" to implement a feature quickly (e.g: by skipping tests) you will pay for it later and usually with "interest".

It may feel abstract and hard to understand but it is crucially not ignored.

In other words, whilst in agile methodology we often talk about iterative growth and only building what is necessary, this should not mean that corners are cut in the short term that will inevitably end up with greater costs of time and resources later down the line.

If technical debt does accrue it is worth reducing it systematically. If left ignored, as other development continues, the debt will grow as the new work added will also need reworking when the debt is finally addressed. Just like how financial debt grows when untouched because of interest.

Note: this is related but distinct from retrospective changes that can be made to a project to make it more performant/relevant/consistent but that could not have been known/foreseen at the time the code was written.

Optimal Team Size - Brooks' Law

Sometimes when a deadline is nearing or timing is looking tight then it can feel tempting to add more people to the team with the aim of getting things done faster. However it is worth considering Brooks' Law which suggests that effectiveness/efficiency can decline with too many people.
How can this be the case?

Bringing on new team members has costs as well as benefits:

- Time to learn and get up-to-speed with context/codebase

- Existing (productive) team members time occupied by reviewing new (far less productive) team members work at the expense of getting on with their own. i.e. output is cut in half not "doubled"!
- Overlap and "stepping on toes" where features are not isolated enough leads to merge conflicts (where one person's work can "undo" someone else's) which can take up way more time.

The best time for the whole team to start is at the "ideation" or "Design Sprint" stage so that no time is wasted on this on-boarding later on.

How do I know what's happening on a project?

This illustration is a guide for POs who want to know the status of their project at any given time. See the glossary if you need reminding what some of the key terms mean.

What's the state of the project?

OR

Attend a stand-up to find out

Check Github

① Go to the **Milestone** and check what percentage is complete.

%

② Look at the issues' **Labels** and assignees.

Assignee Me = ACTION TO BE MADE!

please-test = means check the issue has met the acceptance criteria then close it if you're happy

question = means there's a question for you. Add your answer as a comment.

Other Labels:

in-progress = means someone is currently working on it.

awaiting-review = means this issue is almost complete pending checks from the team.