Hiding Issues in the Issues Tab (GSoC)

Attention: Externally visible, non-confidential
Author: irahatmuneeb@gmail.com
Status: Accepted
Created: 2021-05-20 / **Last Updated**: 2021-08-23

One-page overview

Summary

This feature allows users to selectively hide and unhide issues in the Issues tab, so that they can customize their view to only see the kind of issues they are interested in.

Platforms

Desktop (Windows, Mac, Linux), ChromeOS

Team

irahatmuneeb@gmail.com, wolfi@chromium.org, sigurds@chromium.org

Tracking issue

Chromium: 1175722

Value proposition

As more types of issues are being added and as more console messages are being ported over to the Issues Tab this feature will improve the usability of the Issues Tab and turn it into a more scalable solution for when the amount of issues to be displayed is high. This will help developers single out Issues, promoting a cleaner UI and allowing users to solve one Issue at a time. This feature will also allow hiding third-party Issues, which aren't a direct consequence of the code written by the user.

Code affected

Issues Panel, Chrome DevTools Front-end.

Signed off by

Name	Write (not) LGTM in this row
wolfi@chromium.org	LGTM
sigurds@chromium.org	LGTM
szuend@chromium.org	LGTM% some concerns around architecture
jacktfranklin@chromium.org	LGTM
petermueller@chromium.org	LGTM with the new solution proposed. Let's check on the design of the "hide" section once it's finished

Core user stories

1. As a user, I want to hide some Issues shown in the Issues tab, because either they aren't relevant or third-party.
2. As a user, I should also be able to change the parameter used for hiding Issues.
3. As a user, I should also be able to unhide hidden issues.

Motivation

The motivation behind hiding issues instead of filtering issues is to assign a sense of lower priority to hidden issues. While the hidden issues won't be explicitly shown to the user, they still remain in the UI under a "Hidden Issues Row" to be acted upon albeit not urgently.

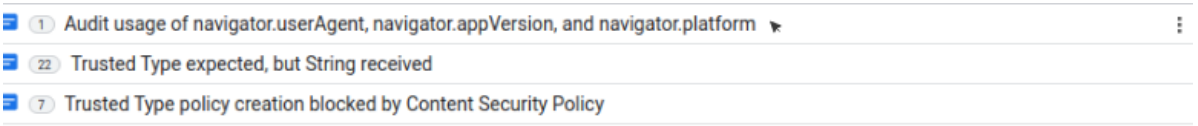
Experimental Designs

As there was no prior specification provided and the feature had to be built completely from the ground up. We discovered complexities in the existing architecture which made it difficult to manage user expectations. Models in which we were able to successfully manage user expectations accompanied a complex UI thus resulting in a poor user experience.
Over the course of GSoC we developed and experimented with multiple prototypes, each solved problems from the previous iterations but led to new ones.
We also created, discovered and fixed bugs as well as made performance optimisations to the existing architecture.

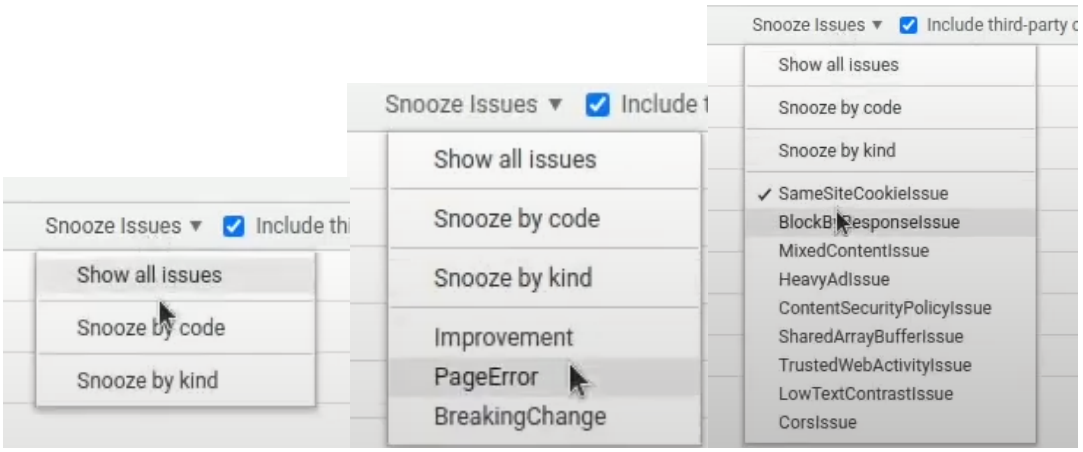
Hiding Issues

- Adding a 3 dots (kebab) icon to each issue in the issues pane representation of an issue, which becomes visible on hovering over the issue. Clicking the icon reveals a drop down

menu with the following options: Hide issue**,** Hide all **[IssueType]** issues**,** hide all **[Page errors / Improvements / Warnings]** .This variation allows finer control over the issues users might want to hide without having to understand chromium specific issue parameters.

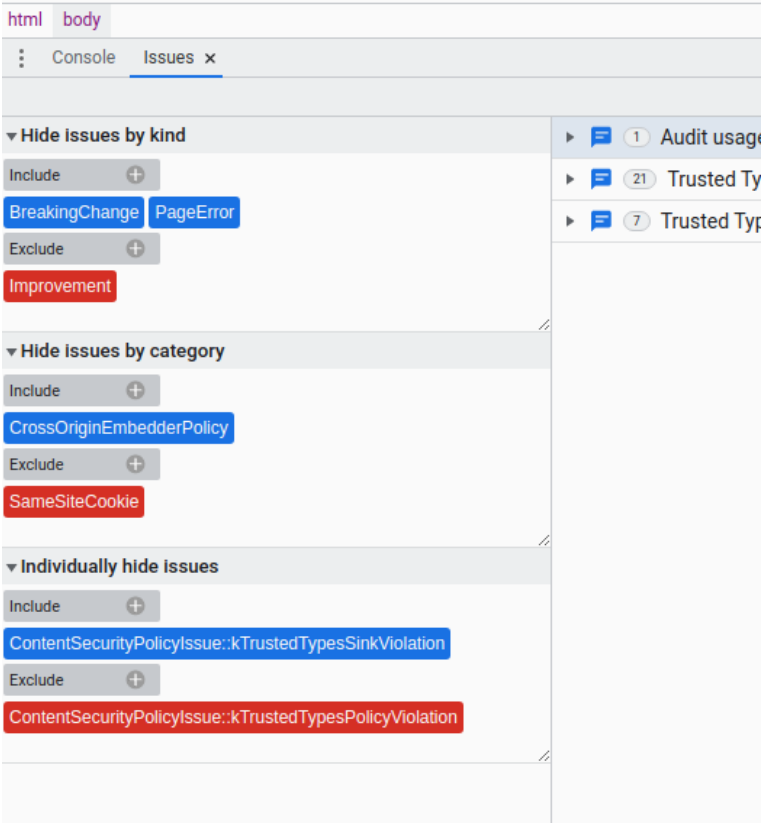


- Another variation is that instead of the context menu being made visible by the 3 dots icons, it becomes visible by right clicking on the issue itself. However, this variation can lead to the feature not being discovered by the user and ultimately being unused.
- Adding a “hide“ button to the IssuesPane toolbar, which triggers a Context menu. The Context Menu then Allows users to choose the type of parameter they wish to hide issues by: issueKind, IssueCode. Each Issue parameter will have a submenu associated with it which will allow users to select the **kind** or **type** (issueCode) of the issue they want to hide. The top-level sub-menus will hide all issues that are identified by an issue code without considering their underlying subtype. Only the first segment of the issue code will be used. For e.g; for **SameSiteCookieIssue::ExcludeSameSiteNoneInsecure::SetCookie**, any issues having **SameSiteCookieIssue** as the first segment of the issue code will be hidden. However, this variation doesn’t give the user control over sub-types of issues and extending the context menu to include various subtypes of issues clutters the UI. It also forces the user to understand chromium specific Issue codes and Issue kinds, which adds complexity and harms user experience.



- In another variation we decided to show users the state of all applied filters on issues by using a side nav. However, this variation had a very complex UI, requiring near identical custom implementation of already existing legacy components only differing in few ways, and absence of user-facing strings for issues parameters like IssueCode and IssueCategory. In this Model:
    - Each filter has an excluded and an included section.
    - All parameters in the included section are Hidden and all parameters in the excluded section are never hidden.

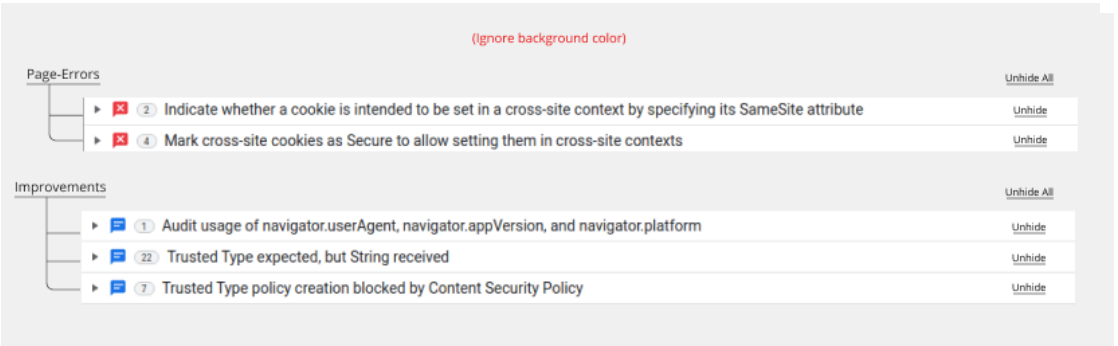
- To unhide issues we just remove it from the included section.



Video: https://drive.google.com/file/d/1jYPy2pYg_zAGXc3WF6FrSTiL2KgnzoOh/view?usp=sharing

Un-Hiding Issues

- Hidden issues are then moved to a small greyed out "**([#number]) issues hidden**" row at the bottom of the issues panel list view. This row could be expanded to see hidden issues as well as make them visible again. The row will group issues by **IssueCategory**, this way users can either unhide specific subtypes of issues or unhide all issues belonging to a particular type. We can further extend functionality by allowing users to group hidden issues, either by **IssueCode** or by **IssueKind**. This variation directly takes into account the subtypes of issues for e.g, **ContentSecurityPolicyIssue::kTrustedTypesPolicyViolation**



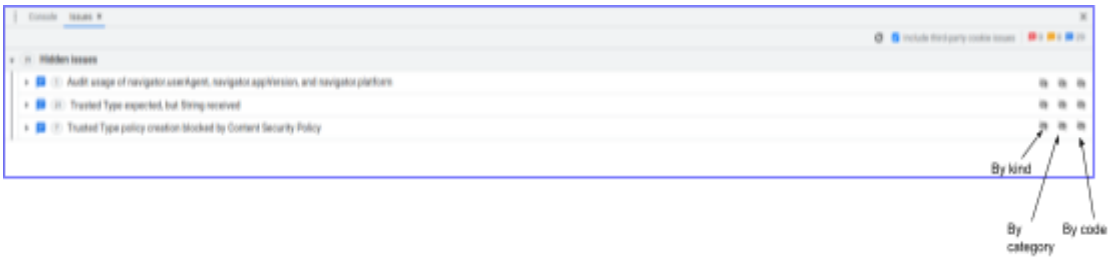
Since issues can be hidden by multiple settings, i.e an issue can be hidden individually, by its category and by its code at the same time, in such situations it's hard to visualize a UI representation which matches user expectations.
Complexities which arose on this feature were:

Whether unhiding an issue which is hidden on multiple levels i.e. by code, by category and by kind should be done in a single action or require as many actions as it has been hidden on e.g **MixedContentIssues** can be hidden individually by their **IssueCode** and furthermore by their **IssueKind** as well. In this case **MixedContentIssue** has been hidden on two levels.

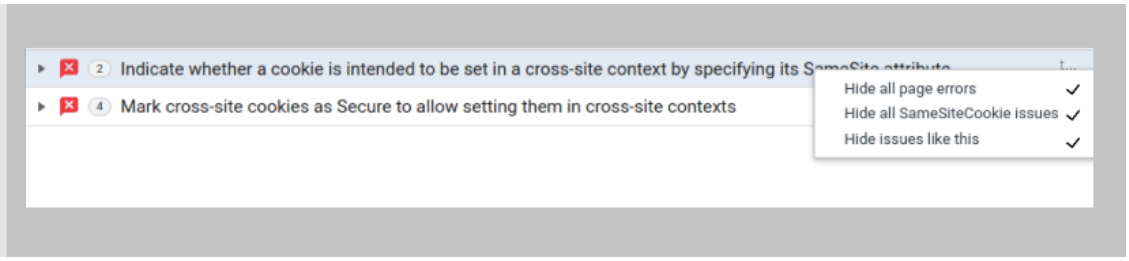
A possible solution for this problem was to show the user on what levels a particular issue is hidden.

This could be achieved in two ways:

- By having 3 icons on the issue header each representing the level an issue is hidden on, for e.g an issue hidden on two levels could have 2 bright icons for the levels it is hidden on and 1 grey out icon for the level it isn't hidden on.



- Or we could have a context menu similar to the context menu we have for hiding issues with the only difference being that the entries on the context menu are checked for the levels an issue is hidden on.



Please note that the menu entries will be changed to “Unhide by” or “Hidden by” and clicking on the entry would remove that particular level of hiding.

Final Feature

In the final feature we have added a **HiddenIssuesMenu** context menu. This menu can be accessed by clicking on the “three dots” menu icon, which becomes visible upon hovering an issue.

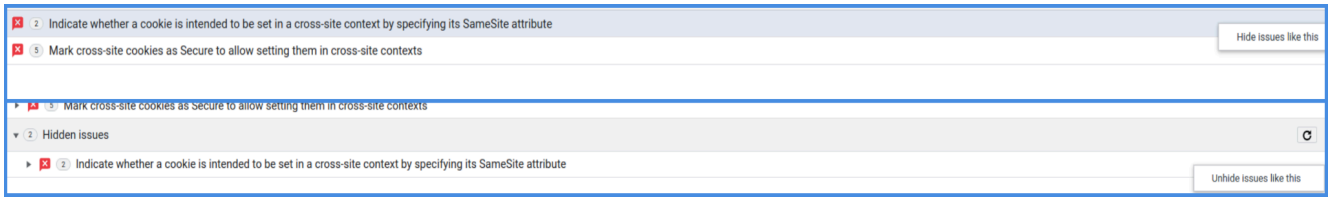
All hidden issues are added to a hidden issues row, which can be expanded to view and “unhide” all hidden issues. Issues can be unhidden by expanding the Hidden issues row, clicking on the “Unhide all issues like this” entry which will be available through the context menu

With this model we avoid hiding any unseen new issues, if an issue hasn’t been encountered before, it will be shown to the user. The user can then decide what to do with the issues. This goes hand in hand with out motivation i.e. we don't want users to forget or ignore issues.

When grouped by kind, all available issues corresponding to an IssueKind can be hidden by using the **HiddenIssuesMenu** attached to the group header.

In the final feature user can:

- Hide issues individually.
- Group and view issues by their respective kinds.
- Hide all currently available issues belonging to a particular kind.
- Unhide issues individually.
- Unhide all issues



Note for GSoC: Grouping with Kind has 2 CLs that are blocked on code-review. Both CLs have an **LGTM** from my primary mentor and need code-review from my other mentor. However they will be landed in the next few days. I have confirmed this with my mentor.

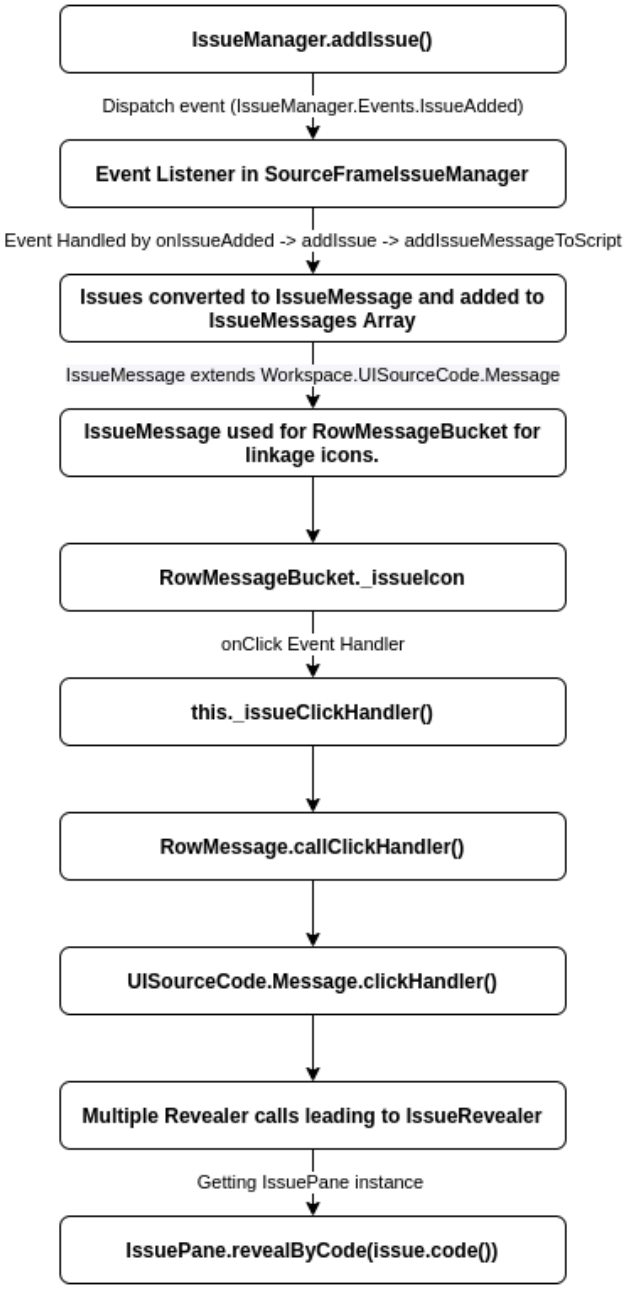
- Group Issues in the issues panel by IssueKind
- Added HideIssuesMenu to group by kind row headers

Here is a video which showcases the feature.

All my CLs including those which will be landed in the next few days can be seen here.

Architecture

A new property called **hidden** is added to the **Issue.ts** class, this property can be changed in the **IssueManager.ts** class by a filter. Currently, all issues are aggregated by **IssueAggregator.ts** and stored in the **aggregatedIssuesByCode** Map. A new Map has been added in **IssueAggregator.ts** which will aggregate and store issues marked **hidden** by **IssueManager.ts.** Issues stored in this map are rendered in the hidden issues row by **IssuePane.ts**.



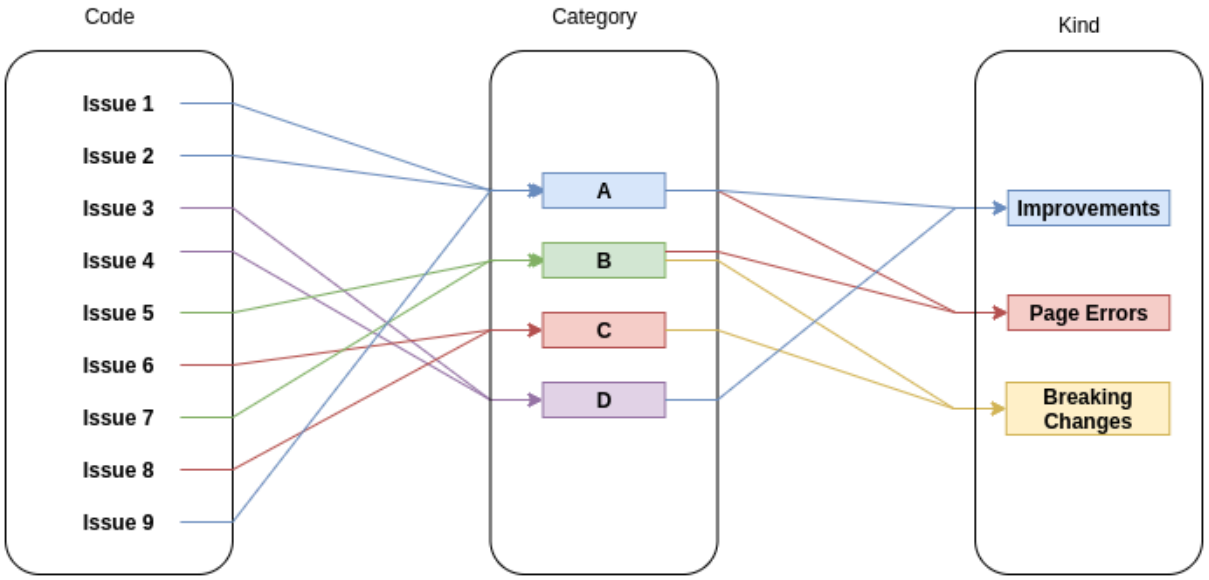
This is a rough representation of the call stack when an issue is added and navigated to through a cross-linked panel for e.g. from the sources panel to the issues panel. Issues that will be marked hidden will only appear in the hidden issues row in the issues panel. However, these hidden issues will still be persistent in cross-linked panels. To make sure cross-linking is not broken, modifications have been made to the **IssuePane.revealByCode()** method such that if an issue is hidden but is navigated to through a cross-linked panel, the hidden issues row should expand displaying the selected issue.

Some improvements made to the existing architecture include, eliminating unnecessary re-parent of issue UI objects and Caching issue UI objects.

Additional details for future work

- Some issues can have the same IssueCode but different IssueKind, because of this during issue aggregation similar issues having different kinds are aggregated into a singular issue corresponding to an encountered issue of the highest priority.
  Page Errors > Breaking Changes > Improvements

  The IssueView UI objects only have access to the AggregatedIssue, so the HiddenIssuesMenu only has access to the highest priority kind. The relationship between **IssueCodes, IssueCategories and IssueKinds** can roughly be modelled as:



- Since the multiple parameters by which issues can be hidden are so closely related, sometimes issues can get hidden by multiple parameters and remain hidden until all of them aren't removed. Managing user expectations related to this behavior can become very complex.
- In the multiple prototypes we developed, we found that Unhiding an issue isn't always the same as the opposite of hiding an issue. It is a separate action leading to branching user expectations.

Rollout plan

Waterfall

Core principle considerations

Speed

We don't expect any significant impact on speed because we would only be filtering issues based on different parameters. Issues are currently filtered in a similar way using the showThirdPartyOnly setting. A similar approach towards filtering console messages is implemented in the console pane.

Security

We don't expect any impact on security because the same amount of information is already surfaced.

Simplicity

Issues Pane in DevTools will become more customizable by implementing this change because issues could be hidden and shown at the user's discretion.

Accessibility

Accessibility will increase because users will be able to hide unwanted issues and concentrate on fixing other issues.

Testing plan

Unit testing and E2E tests.

Followup work

We could add functionality such that website specific "hide" settings are remembered and persisted across browser sessions. This feature in particular can prove to be beneficial to developers working on multiple projects. This would allow them to save "hide" settings and not repeat all steps, especially when the amount of added issues is high. Keeping in mind the API being developed for libraries to add custom issues, this feature would be very useful when working with multiple libraries and projects.