madcap software

USER GUIDE

MADCAP FLARE ONLINE

What's New Guide

MadCap Software
1660 17th Street, Suite 201
Denver, Colorado 80202
858-320-0387
www.madcapsoftware.com

THIS PDF WAS CREATED USING MADCAP FLARE.

CONTENTS

APPENDIX

CHAPTER 1

Introduction

Following are new features available in Flare Online.

For more information about each feature discussed in this manual, open the Help system and refer to the "What's New" topic. Links are provided in some feature descriptions, taking you to topics that contain additional information and steps.

"Flare Online" on page 6

- MadCap Central renamed Flare Online
- Text and logo changes throughout interface
- Think of legacy Flare as Flare Desktop

"Collaborative Authoring" on page 11

- The collaborative authoring feature lets multiple people work on an online document simultaneously (i.e., collaboratively and concurrently edit the same file at the same time)
- Multiple users can see changes from other authors in real-time, without the worry of losing work or running into conflicts
- A Workspace Overview promotes work management by seeing at-a-glance status such as who's working or what's been completed
- The workspace offers project coordination that brings each team member's authoring work together in one centralized place

CHAPTER 2

Flare Online

MadCap Central has been renamed Flare Online. It's the same cloud-based application that you've come to know (with some exciting new collaborative authoring benefits also now available); it just has a new name and a somewhat new look. As far as legacy Flare is concerned, you can now think of it as Flare Desktop.

This chapter discusses the following:

Why?

The reason for this change is that, over time, Central has grown so much in terms of authoring features that it really has morphed into a cloud-based version of Flare. A Flare author could start a project, work only in the cloud (without ever needing or touching Flare Desktop), and end up with state-of-the art online and print-based outputs.

What is Different?

In the interfaces for both Flare Online and Flare Desktop, you will notice that the text "Central" has been replaced throughout with "Flare Online." There might be a few "corner" locations where Central is still used, or behind-the-scenes occurrences. Mostly, however, the new term replaces the old term.

Along with the text changes, there is a new color scheme and logo for Flare Online.

Flare Online vs. Flare Desktop

If Flare Online is becoming more and more a standalone version of Flare Online in the cloud, then the obvious question is "Do I need Flare Desktop at all?" The answer to that is going to be different, depending on your needs. Some people might only need Flare Online. Others might only need Flare Desktop. And still others might need both, involving the movement of content between the two applications.

- Flare Online is a great choice if you want a very simplified process with the most essential documentation elements, as well as the special offerings that are available only in the cloud (e.g., hosting output, collaborative authoring, analytics).
- Flare Desktop is ideal if you must work locally and need advanced authoring features that have been developed over decades (e.g., context-sensitive Help, meta tags, heavy customization of print-based outputs).

To make an informed decision, you should consider the benefits of both applications.

Flare Online Benefits	Flare Desktop Benefits
■ Collaborative authoring ■ Integrate with ChatGPT (AI Assist) ■ Configure single sign-on (SSO) ■ Host output with custom domains and vanities ■ Publish private output ■ View analytics on output ■ Assign user permissions ■ Use integrated checklists ■ Collaborate with reviewers ■ Create and manage tasks ■ Integrate with Slack ■ Easy cloud-based translation process ■ Customize dashboards and widgets ■ Communicate via message center	■ Highly specialized authoring features ■ Editor structure bars ■ Dynamic topic preview ■ Full-featured text editor ■ Heavy-duty print-based customization ■ Micro content (enhanced search, chatbots, etc) ■ Context-sensitive help ■ Link viewer ■ Advanced cascading stylesheet editing ■ Integrated responsive web design ■ Learning and development integration ■ Various search engine choices ■ Creation of publishing destinations

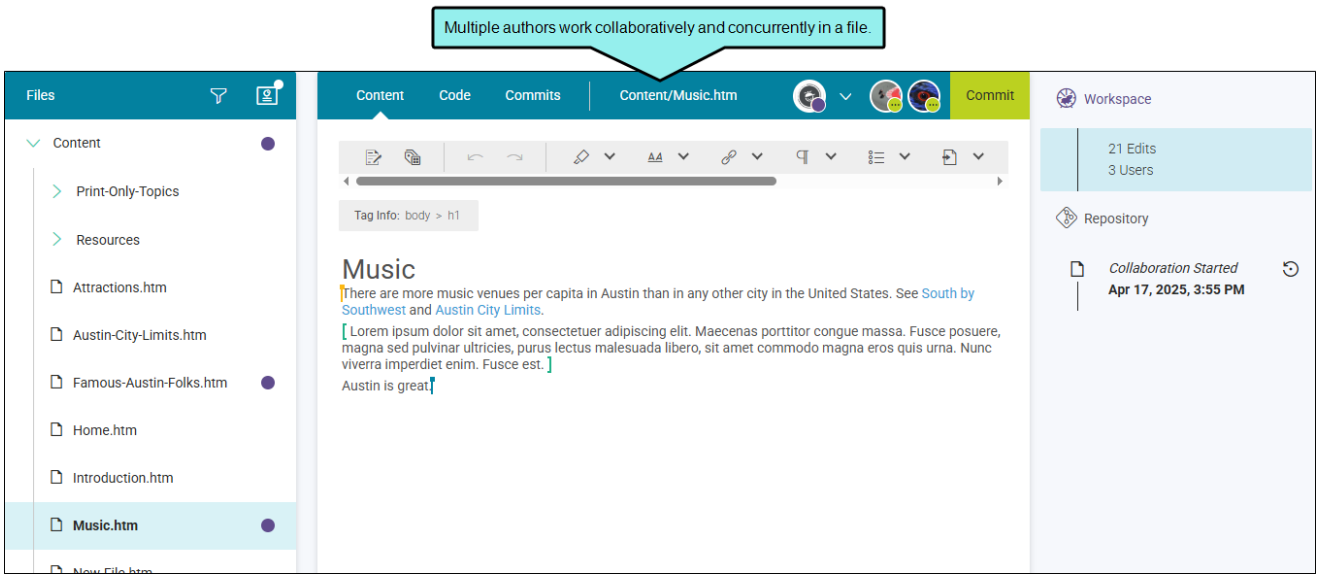
CHAPTER 3

Collaborative Authoring

Until now, each person authoring files in Flare Online worked independently. That individual was unaware if others were authoring the same files and what they were adding, removing, or changing in those files. In addition, if two authors were simultaneously editing the same file, the first person to commit the file had precedence. When the second person committed the file, that author was alerted to the issue and prompted to refresh to get the latest files. Unfortunately, that could result in the second author losing changes and having to redo edits.

Starting with this release, authors work collaboratively in files, with full transparency of edits from everyone in that project. Some benefits that collaborative authoring offers are:

- Working on an online document simultaneously (i.e., the same file at the same time).
- Awareness of changes from other authors in real-time without the worry of losing work or running into conflicts.
- At-a-glance status such as who's working or what's been completed.
- Authoring and coordinating work together in one centralized place.



This chapter discusses the following:

Permission Required?

Editing content and project files is an activity available to users with the Author status. By default, users with Author status have the following permissions set:

- ☑ Create/Edit Files

  If this is deselected, then viewing files in a read-only mode is allowed. On the left side of the page, the Files vertical three-dot menu is not available.

- ☑ Edit Code

  If this is deselected, the XHTML in the Code view is read-only.

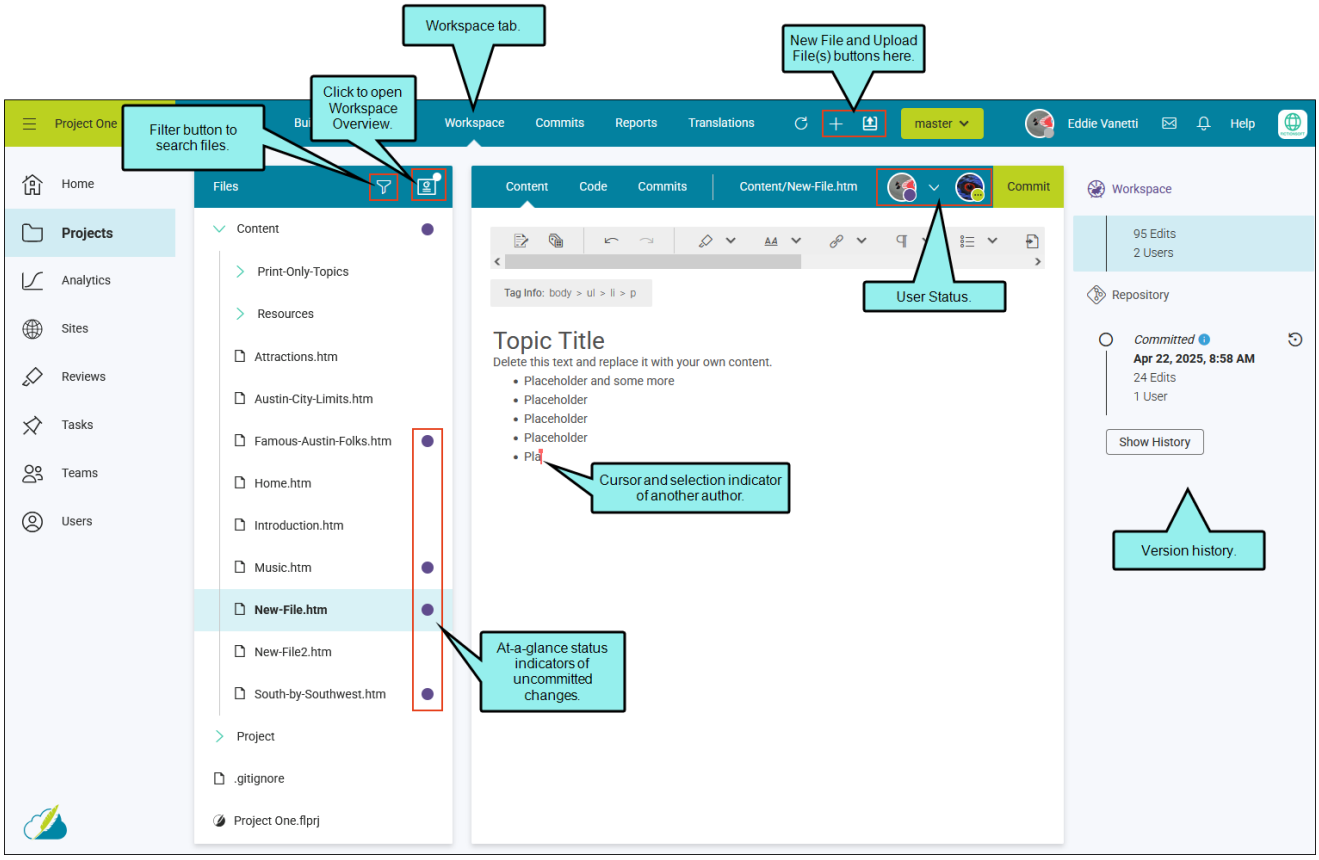
  Editing code is regarded as a capability for an advanced user. If not done properly, the code can become malformed quickly. Administrators can prevent users from editing the code by deselecting the Edit Code permission.

For more information about permissions, see the Help system.

What are the Changes to the Editing Workspace?

To accommodate for collaborative authoring, you will notice some changes to the user interface. These improvements will slightly impact your authoring and editing workflow.

New for this release is a Workspace tab in the main toolbar. This replaces the Files tab. Both single and multiple authors will use the workspace for writing and editing files.



- Cursor and selection indicators There are visual cues that display for each user, indicating that multiple people are using the file and other authors are highlighting text selections.
- Filter The Files tree includes a filter button. Use to search project files based on file type or file status.
- New File and Upload File(s) Buttons are located in the main toolbar, one for creating a new file and one for uploading file(s). Previously, this functionality was launched from the Files tree three-dot menu.
- Workspace Overview Click the button to initiate an overview, which gives a report on the workspace. Use to manage your workspace and to identify items, such as files that are not committed, user status, and what needs to be done.
- Status indicators The Files tree shows at-a-glance status. For example, a purple circle next to a file indicates that it has uncommitted changes.
- User status A user's avatar or initials display in the editor's local toolbar when editing a file. Avatars of any other authors working in the file will display next to yours. Your avatar is unique because you can click the drop-down next to it and select your status from the menu.
- Version history When editing a file, a timeline displays with a history of file commits and versions including edits.

States of the Workspace

When you and other authors are editing a file, it is the same file, in a working state, for everyone. That is why you can see changes happening.

- In Progress A file is in a non-committed state and is opened or being edited by authors. The version history indicates the file is in progress and displays the number of edits and users in the workspace.
- Ready to Commit (Optional) A file is done being edited. The file is saved locally, but it is still a working copy and needs to be committed to the repository. Use this state if you are done editing but you notice others are still working on the file. That way you are not committing other authors' changes before they are finished.
- Commit A user or manager can commit a final version of file changes live to the repository. The action of committing (e.g., pressing the Commit button in the dialog) saves the edits and commits changes from all users—not just your changes—to the repository. When a file is committed the user statuses are cleared.

User Status

For every author in a file, you will see each user's avatar with status in the editor's toolbar.

- Avatar (or initials) with a purple circle  This indicates a user is In Progress with a file.
- Avatar (or initials) with ...  The three-dot icon indicates that a user has a file open and is actively editing it. If you click off the current file to open a different file, the icon changes to show that you are In Progress (i.e., a purple circle).
- Avatar (or initials) with a check mark  An author is done making edits and has indicated this by selecting "Ready to Commit." The file is not committed yet. If a commit occurs, the check mark disappears and the version history shows the file as committed.

> NOTE If there are too many users to show each avatar in the toolbar, a three-dot icon indicates more users  .

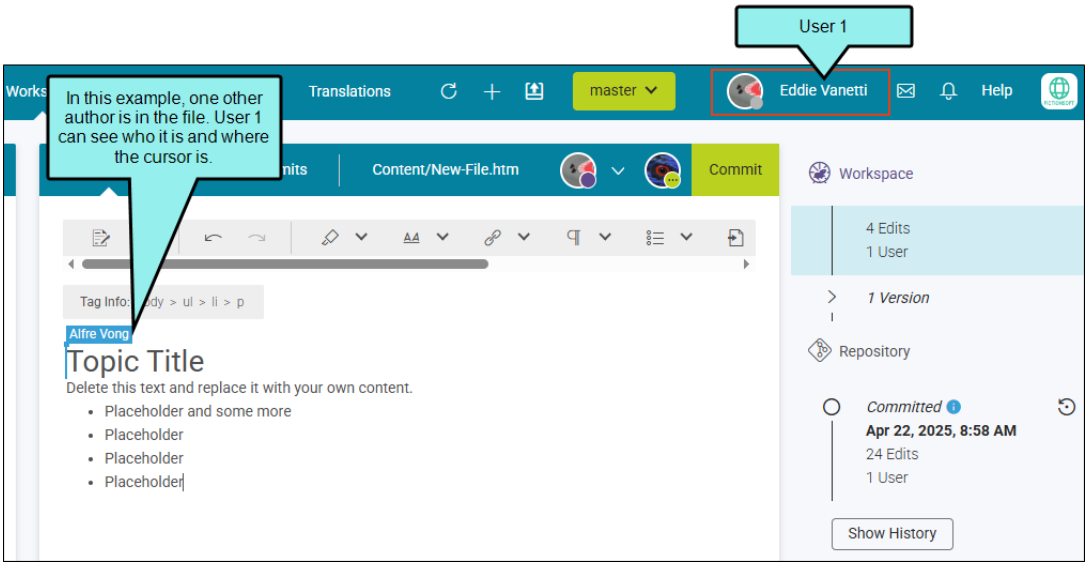
Visible Changes From Others

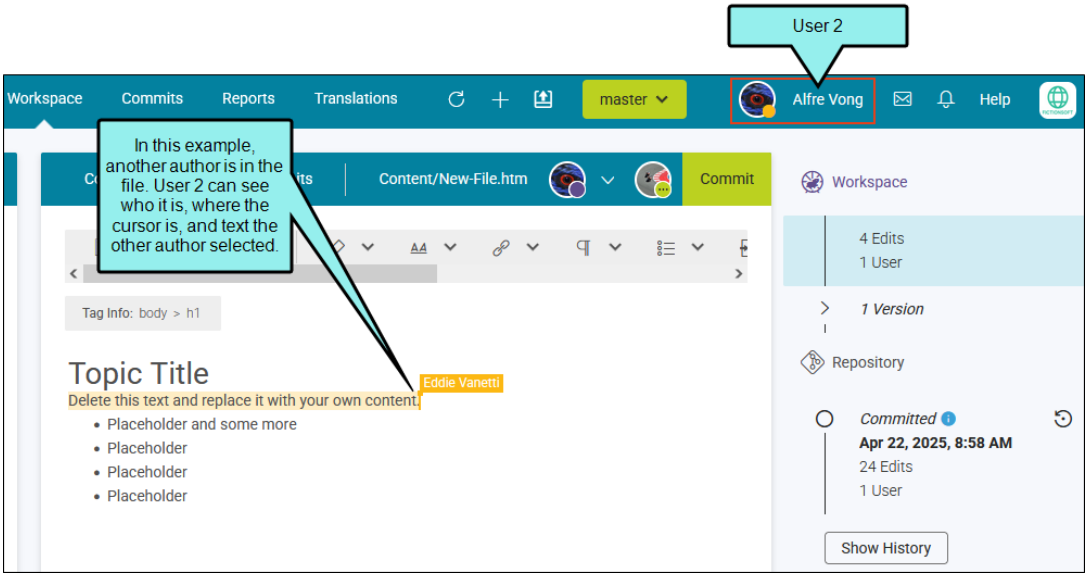
When a file is selected from the Files tree, it opens in the Content Editor—in a working state.

A single user can begin making edits to the file. Once editing occurs, the version history on the right is activated and is updated as edits happen. When the editing is done, the file can be committed into the repository.

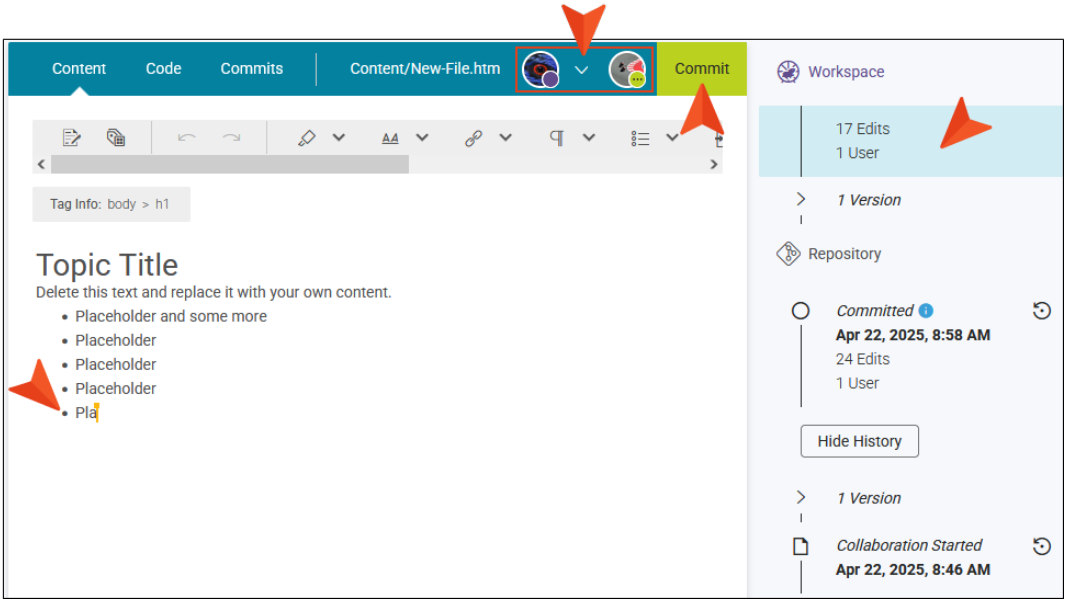
If multiple users are using the same file, you can see the changes the other authors are making to it in real-time. There are visual cues that display for each user, indicating that multiple people are using the file and other authors are highlighting text selections.

☆ EXAMPLE Two authors are using the same file in the editor. Notice the cursor and selection functionality that is visible to each author.





As soon as someone starts editing the file, edits are displayed in real-time for each user, the Commit button is enabled, and the version history is activated for the file.



(After a short time, the name associated with a cursor and selection highlights of content disappear to remove unnecessary marks from the workspace.)

Do I Have to Worry About Conflicts?

One of the biggest benefits of collaborative authoring is that you can avoid conflicts that you might otherwise encounter, at least when authors are editing files in the same branch in Flare Online. That's because each author's changes are visible and automatically incorporated when the commit is made. You can collaboratively and concurrently author a file without the worry of losing work or running into conflicts.

Keep in mind, however, that conflicts could still eventually arise later under certain circumstances. For example, you might have made changes to a file in Flare Desktop, and when you do a pull, you might see conflicts with changes made in the same file in Flare Online. Another example is when you use Flare Desktop to merge files from one branch into another; if changes are different in the same file, you might need to resolve conflicts.

Auto-Merging Files With External Commits

Other authors can work in the same file as you using an external source, such as Flare Desktop. Since Flare Online uses collaborative real-time technology it does not lose track of external changes or create conflicts, but rather invokes an auto-merging process where you are always guaranteed a valid document.

For example, you are working on a file, and someone else edits the same file in an external source and "pushes" the change to Flare Online. When you commit your changes to the file, Flare Online detects external commits and starts the auto-merge process. You can choose to review the external commits, or proceed to commit merged changes.

EXAMPLE The following is common auto-merging workflow. Imagine a project is bound between Flare Online and Flare Desktop, and two authors are editing the same file, on the same branch, in the same project.

User 1 Open a file in Flare Online's Content Editor.

User 2 Open the same file, but in an external source (e.g., Flare Desktop).

User 2Edit the file by changing the content.

User 1 Edit the same file by changing the content.

User 2 Commit and push changes in Flare Desktop to the remote repository (i.e., Flare Online). See the Flare Desktop Help.

User 1 Commit the same file in Flare Online's editor. Clicking the Commit button starts the auto-merge process. Upon the initial commit, the outside edits (or external commits) display in the workspace.

User 1 Cancel to review the external commits in the workspace, or proceed to commit merged changes.

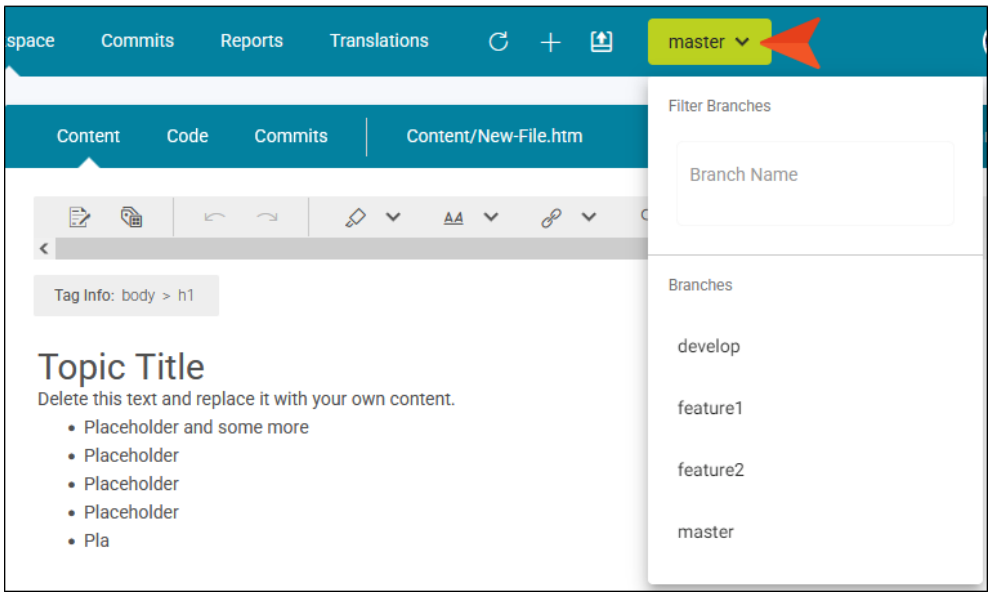
NOTE The version history and your working copy of the file in the Content Editor do not display external commits merged until you attempt to commit the file (even though behind-the-scenes external commits exist in the repository).

NOTE Flare Online is a Git-based system, where files are version controlled and tracked using the system's source control. The system implements "rules" for resolving possible conflicts. The platform allows for secure co-authoring from anywhere with any device, and accessibility to your online files from the Git repository with no memory or performance impacts.

Multiple Authors — How to Edit an Existing File

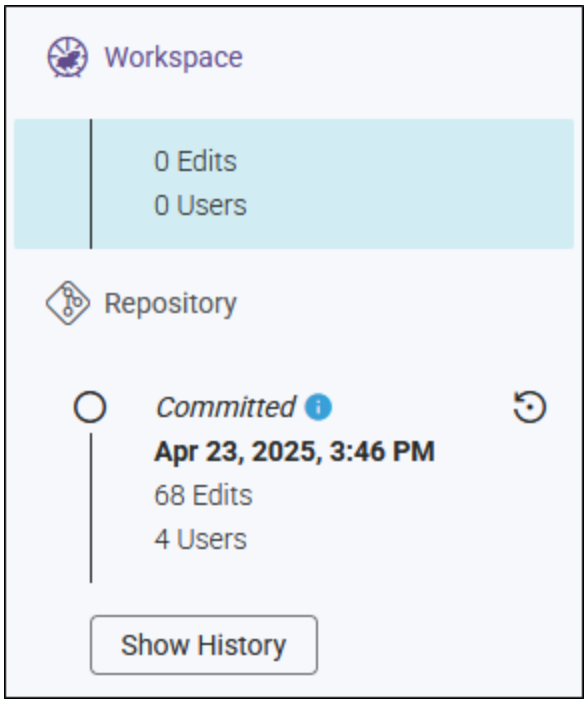
For multiple authors editing the same file at the same time, the following workflow serves as a guide.

1. On the left side of the Flare Online interface, click **Projects**.
2. Select a project to open it.
3. Click the **Workspace** tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.

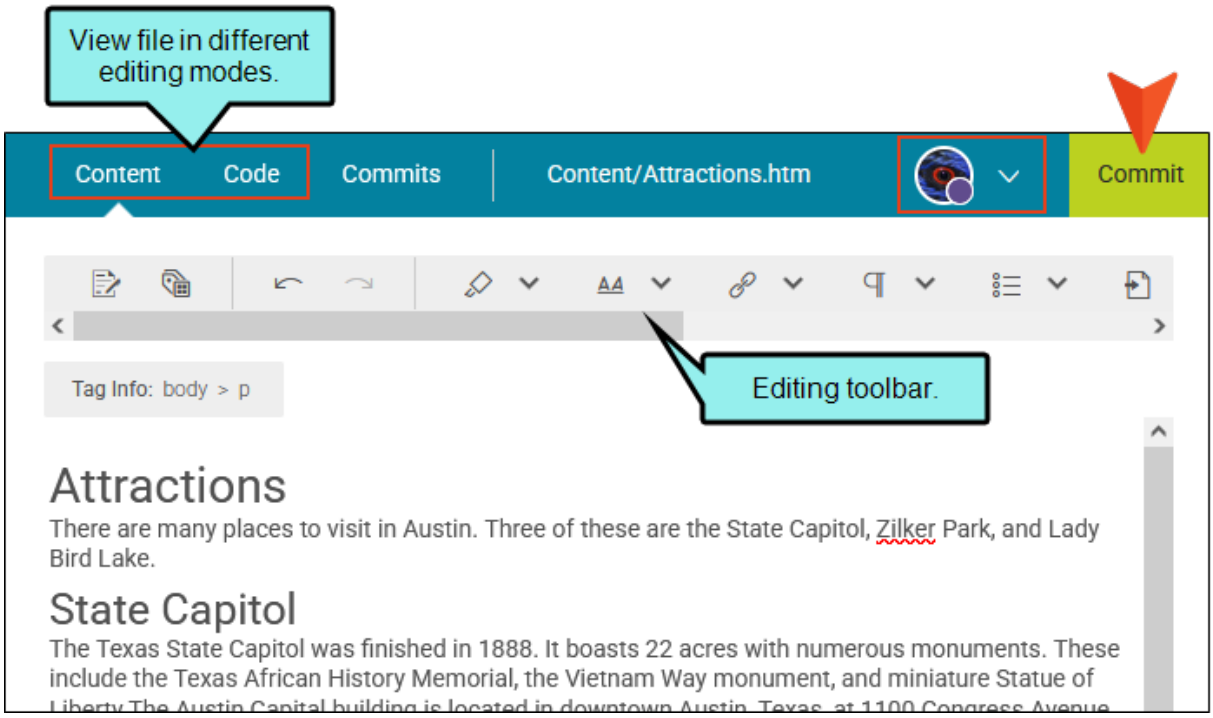


5. (Optional) Click  . The Workspace Overview opens to display various items (e.g., status, authors, type) about edited files in a project. (If files are already in a working state, the Open Workspace Overview button shows with a circle in the upper-right corner  .)
6. From the left side of the page, expand the existing folders to navigate to a file. You can also click  to search for a specific file.

7. Select a file. It displays in the editor to the right. The right gutter switches from showing project activities to displaying a version history for the file.



8. Click in the workspace and make an edit (this activates the file in an uncommitted state for editing, auto-sets your status to In Progress, enables the Commit button, and begins to populate version history). Use the toolbar at the top of the editor to manage the content. Also, the tabs at the top of the editor (i.e., Content, Code, and Commits), allow you to switch modes so you can edit the content or markup, and view commit details.



NOTE The tracking changes option is off by default for collaborative authoring. From the Content Editor's toolbar, click Toggle Tracking Changes to enable or disable the feature.

9. When you are done editing a file, it needs to be committed into the Git repository. You can look at the local toolbar or the Workspace Overview to see the status of others.

   Do one of the following:

   - If you determine that you are the last one to finish editing the file, then it is safe to commit the file. From the upper-right corner of the workspace, click **Commit**. (The Commit dialog will be free of a warning about other users in the file.)

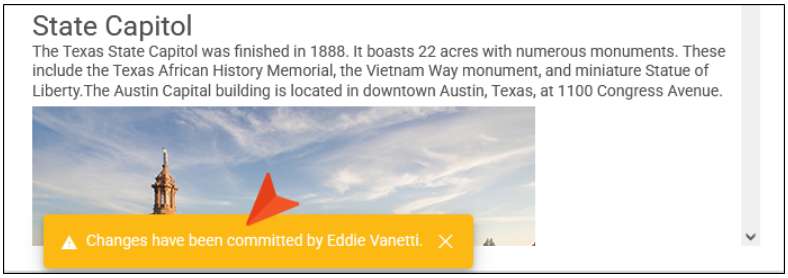
     Alternatively, you can use the Workspace Overview to perform the commit operation. When a file or many files are in a Ready to Commit state, you can commit them one at a time or in bulk.

   - If other authors are still editing, you can let others know that you are finished with the file. From the local toolbar, select the drop-down by your avatar and click **Ready to Commit**.

     When all authors have indicated they are done and the file is ready, then any author or manager can commit the file. If this is the case, click **Commit**.

     > NOTE If multiple files are In Progress, but only one is marked as Ready to Commit, the Commit button is active. This is because you do not have to wait until all the files are complete before committing them. For example, an author goes on vacation but the In Progress state for the file needs to be committed before the author returns.

NOTE If other authors are still working on a file and their statuses indicate In Progress, you can still perform a commit (although you should probably communicate with them first). If you do commit a file while other users are in this state, they will see a notification in their editor.



10. In the Create New Commit dialog, verify the new file path, and enter a **Commit Message**.
11. Click **Commit**.

NOTE With multiple authors, it is the user's responsibility to be aware of other people editing the same file.

NOTE If an author is editing a file, but never commits the file to the repository, another user (author or manager) can commit the file at any time.

Committing Edits

Why do I have to commit edits? If you open and view a file, it is in a committed state in the project's repository. If you click to edit a file, it then enters a "workspace mode," essentially making a working copy of the file from the repository. In this mode, the file is in a non-committed state with pending changes. When editing is completed, the file needs to be committed back to the project's repository.

In general, be aware of the following when committing edits:

- Single file If you are editing a file and click Commit, you are committing what you are looking at in the editor. It can contain edits just from you (one user), or edits from several authors in the same file. (You can also commit a single file via the Workspace Overview.)
- Multiple files Instead of committing a single file, you can bulk commit multiple files at once. Note that many files can be in an uncommitted state which might contain edits from you and other authors.
- Navigating to other files When you work in a file, you can move to another file and work in that, then move to a third file and make changes, and so on. In other words, you do not need to commit an open file before navigating elsewhere. The changes in each file are automatically saved, but they won't be added to the repository until you (or another author) finally perform a commit.
- Committing edits can affect other users Because collaborative authoring is always happening, authors performing a commit are doing so not only for their changes, but also for any other author's changes for the same file. For this reason, it's important to stay in communication with other authors to make sure you are not committing changes that they do not yet wish to be committed. However, the editing process includes safeguards and visible cues in the interface to commit files without issues.

How to Commit Edits

Do one of the following:

- In the Content Editor, from the upper-right corner of the workspace, click **Commit**.
- In the Workspace Overview, select a file (i.e., the check box next to the file in the grid) and from the upper-right corner click **Commit**. Alternatively, click the vertical three-dot icon next to the file and select **Commit**. You can bulk commit files with edits.

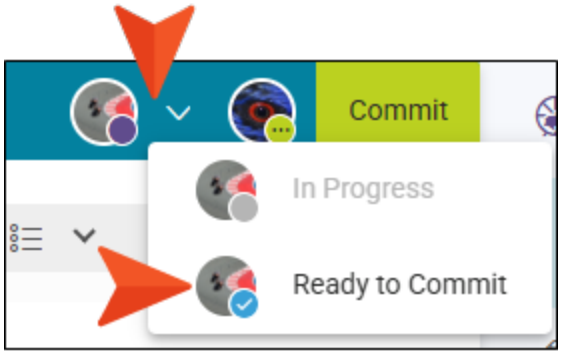
NOTE It is a best practice to simultaneously commit all related files based on the changes you are making. For example if you edited four files to complete a changed feature, then commit all four files together when your changes are done. This ensures a completed collective version of your work in the revision history that you can view and revert back to.

NOTE Until commits are made, changes from those files will not be included in any builds that are generated.

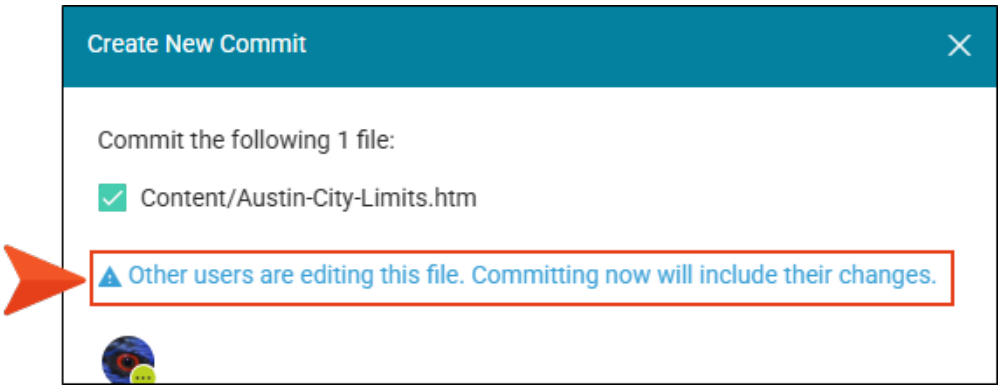
Selecting Ready to Commit

What if you are finished making changes in a file and do not plan to make any more, but other authors are still working in that file? If you perform a commit, you are going to commit their changes as well, and they might not want that.

Fortunately, there is an alternative, you can click your small avatar at the top of the file being edited, and from the drop-down you can select Ready to Commit. This is simply a way of signaling to others that they are free to perform a commit for your changes as well as theirs.

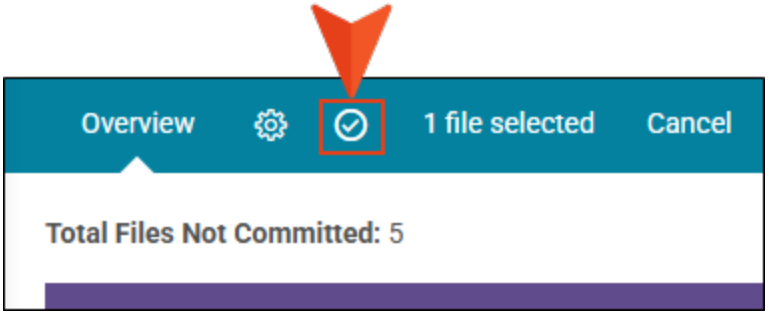


If you do happen to click the Commit button when there are still uncommitted changes from others in the project, you will see a message warning you of this. (If other users have all selected Ready to Commit then you will not see the warning.)



Of course, you can ignore the warning and continue with the commit, but this will commit changes for others at the same time.

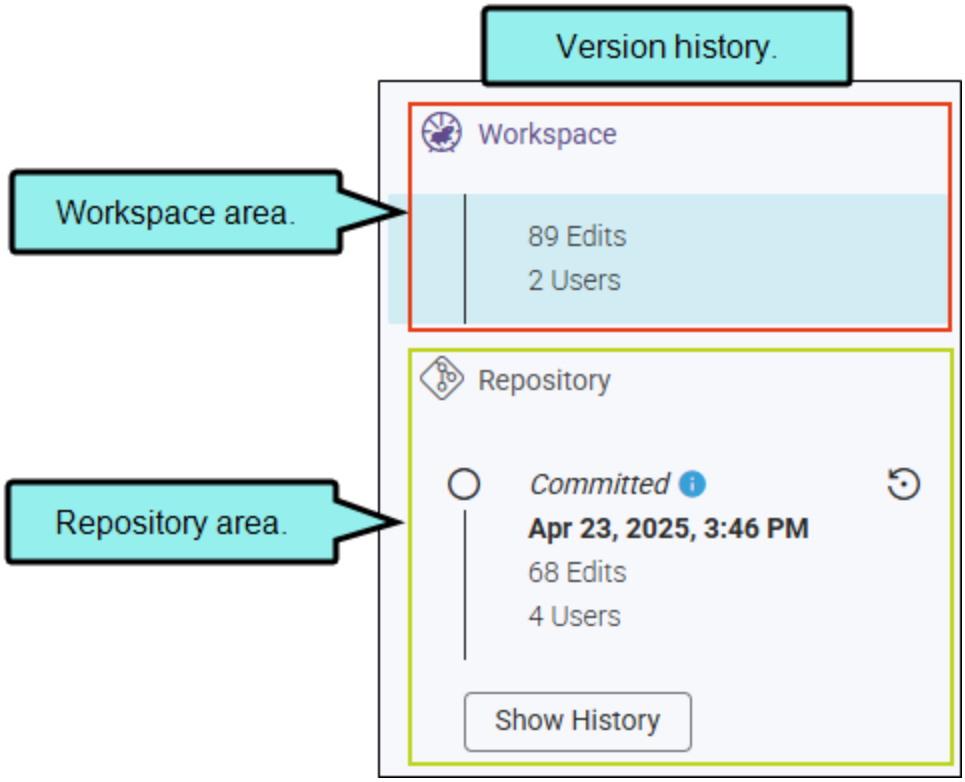
NOTE If you set the status to Ready to Commit, but you realize you have more changes to make, you can update the status again. From the drop-down next to your avatar, select In Progress. Or, from the Workspace Overview, select the file, and click the Change My File Status button to update it.



Using Version History

The version history area showcases more benefits of collaborative authoring. It maintains a revision history of changes in the project's repository where you can access commit details, view previous versions, and restore a version if need be. It also displays a count of edits and users working on a particular file in the workspace.

The major sections of version history are the workspace and the repository areas. The workspace tells you about an uncommitted working copy of the file. The repository tells you about the committed versions of the file in the repository. The version history for a certain file can get quite long depending on the amount of work applied to it.

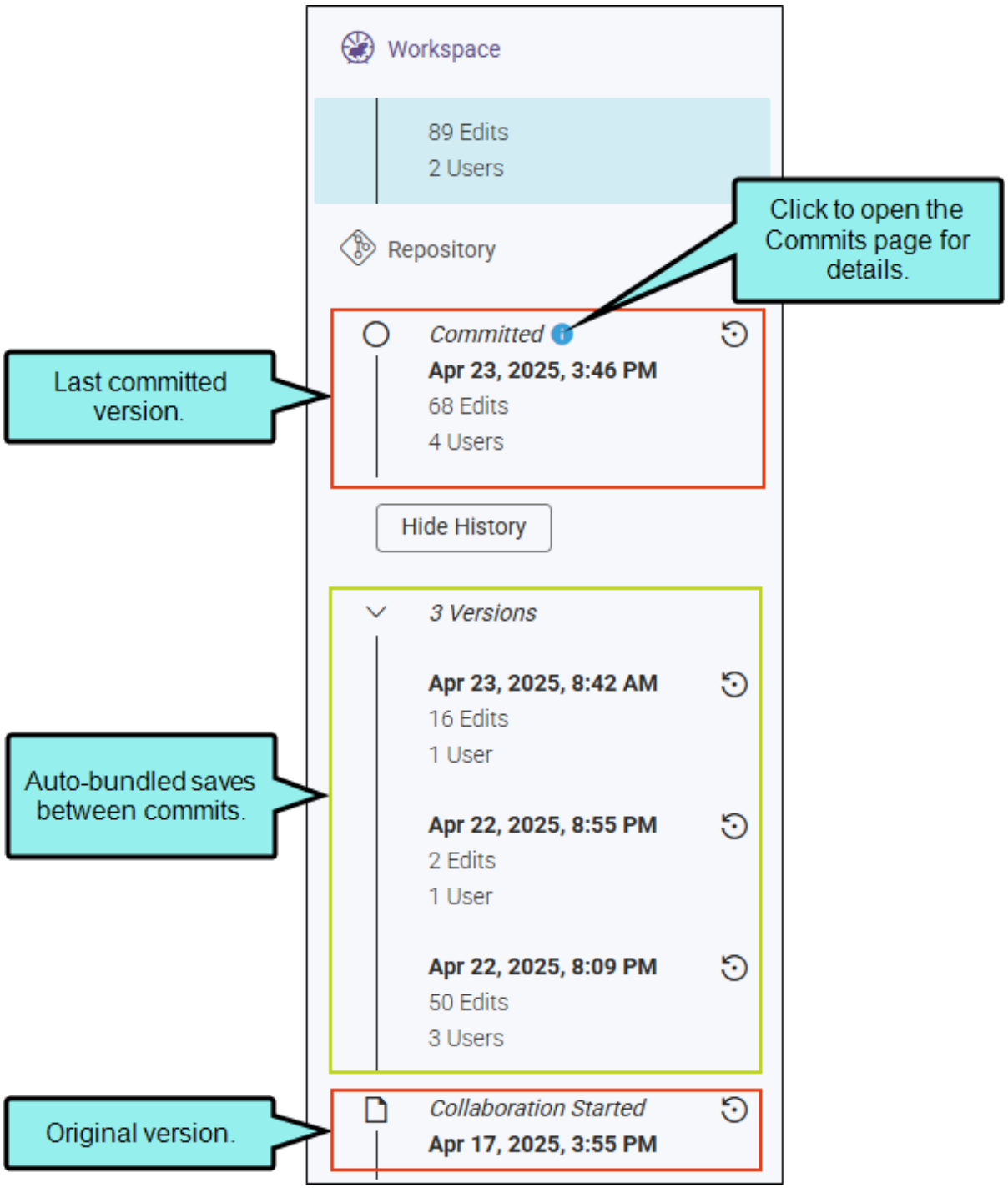


Following are a few things to know about the version history:

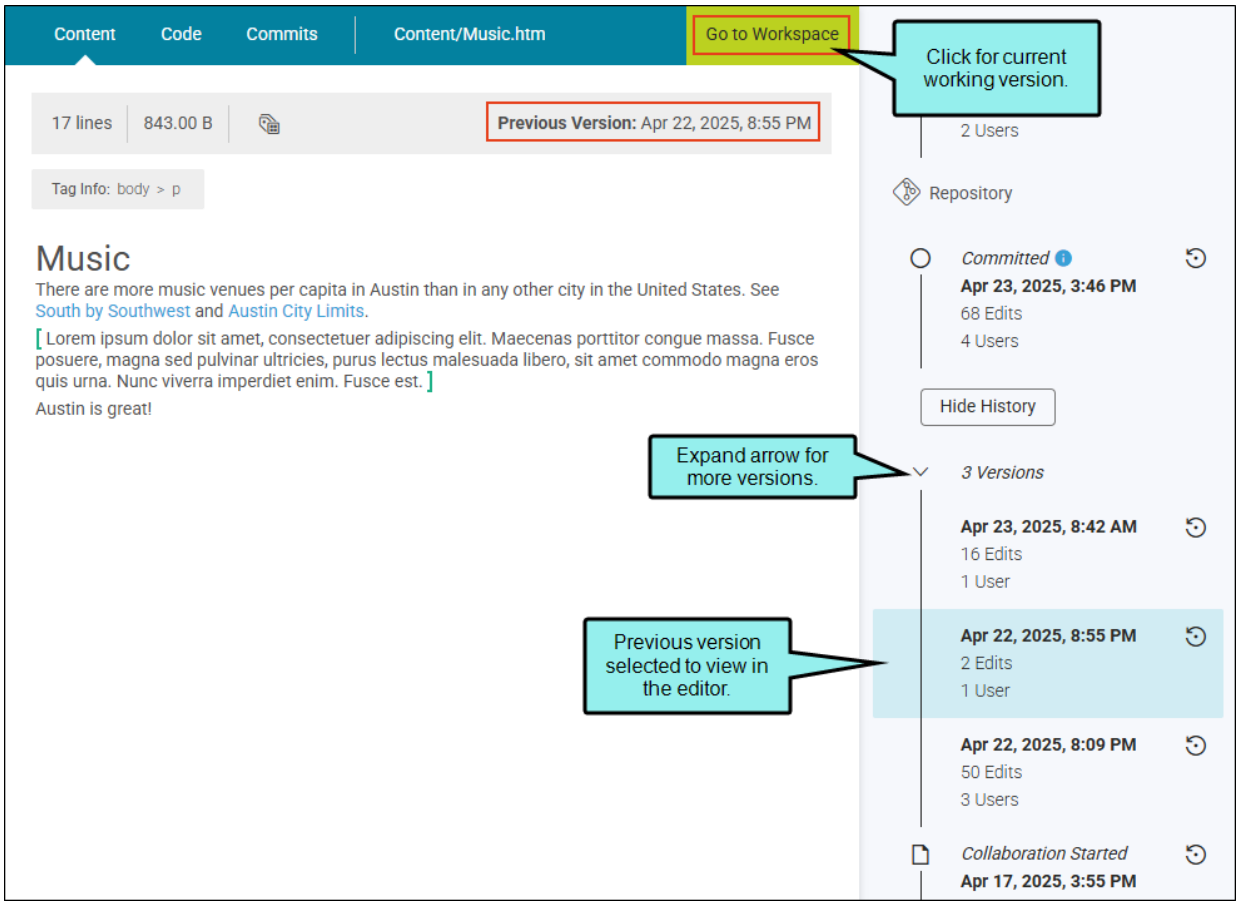
- The last committed version always displays in the repository area. When the history is expanded it shows from the end, not the beginning. In other words, the latest version displays first at the top of the list, and older versions move down in the list.
- The edits number can get high (e.g., 254 edits). Edits are counted per action (i.e., roughly one character change equals one edit). The system controls the count of edits by auto-bundling changes (i.e., creating a working version of the file in the repository) between commits, every 30 minutes.
- The version history is updated based on the branch; and there is only one workspace per branch.

How to View a Previous Version

1. On the left side of the Flare Online interface, click Projects.
2. Select a project to open it.
3. Click the Workspace tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.
5. From the left side of the page, select a file. The last committed version displays in the editor, and the version history area shows on the right side of the screen.
6. In the version history area, click Show History. Upon selection the button changes to Hide History—so you can collapse the history. If revisions exist for the file, the timeline expands to show the versions (i.e., original when it was created, auto-bundled saves between commits, and committed files.)


Workspace
89 Edits
2 Users
Repository
Committed
Apr 23, 2025, 3:46 PM
68 Edits
4 Users
Hide History
3 Versions
Apr 23, 2025, 8:42 AM
16 Edits
1 User
Apr 22, 2025, 8:55 PM
2 Edits
1 User
Apr 22, 2025, 8:09 PM
50 Edits
3 Users
Collaboration Started
Apr 17, 2025, 3:55 PM
Click to open the Commits page for details.
Last committed version.
Auto-bundled saves between commits.
Original version.

7. Select a previous version. (To select an auto-bundled version, you might need to click the down arrow to expand the timeline further.)
8. The editor displays the version selected. You might notice older edits.
9. Click Go to Workspace to return to the current working version.



NOTE To help identify certain versions, there are a number of clues in the timeline to guide you. Ask yourself if the version you are looking for was committed to the repository, or was it auto-bundled? If committed, do you know who performed the commit? Do you know the date and time of the needed version? Do you know of edits to specifically look for? If the file is a committed version, you can also click the information icon to view detailed changes for the commit.

How to Revert to a Previous Version

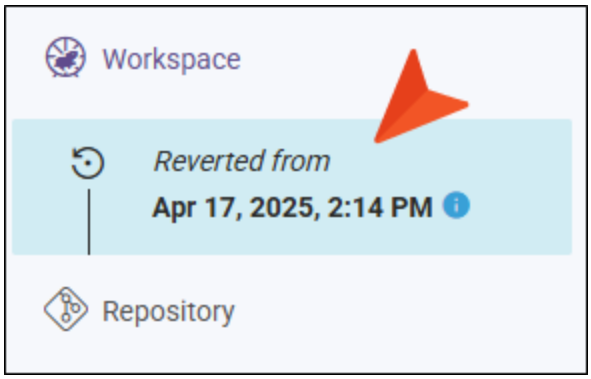
You can revert to a previous version of the file (i.e., to the original version, a committed version, or an auto-bundled version).

1. On the left side of the Flare Online interface, click **Projects**.
2. Select a project to open it.
3. Click the **Workspace** tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.
5. From the left side of the page, select a file. The last committed version displays in the editor, and the version history shows on the right side of the screen.
6. In the version history area, find the instance of the file that you want to restore. (You might need to click Show History or expand auto-bundled versions in the timeline.)

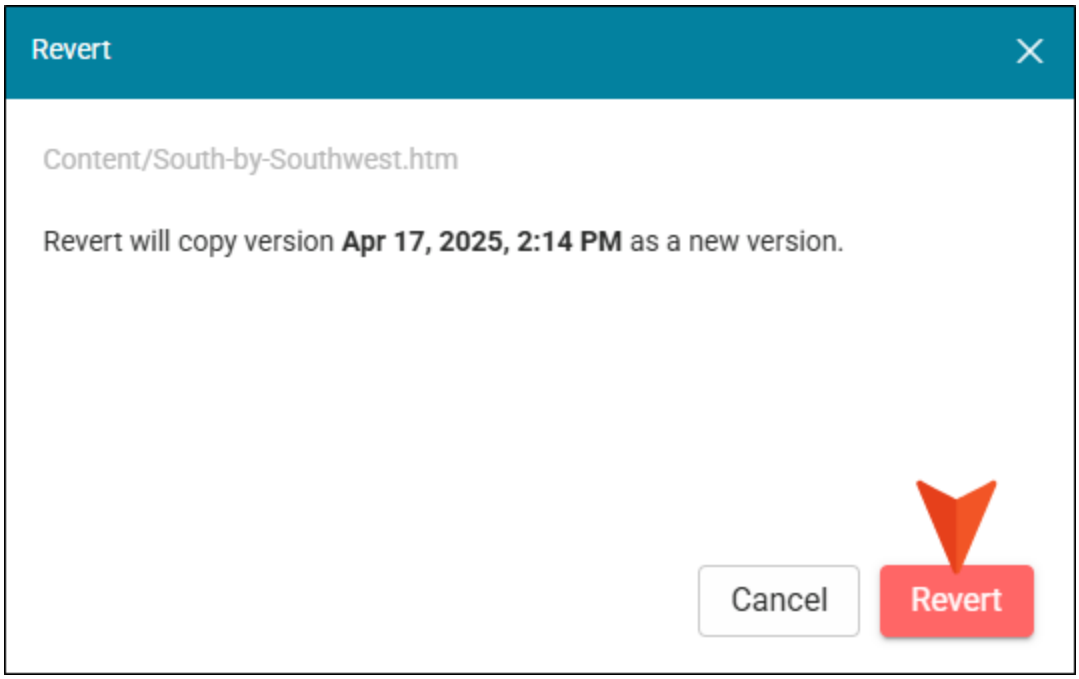
   > TIP It might be a good idea to select the version and view it first, before reverting it, just to make sure that version is the one you want.

7. Click (**Revert to this version**).

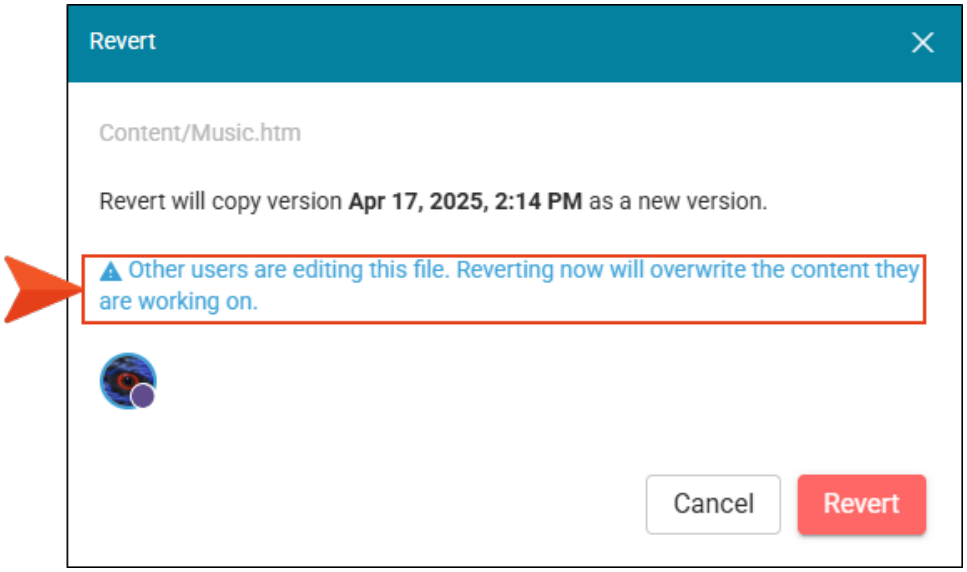
   This action copies the version as a new version, and records it as such in the workspace section.



8. The Revert dialog opens. If no other users are editing the file and it is safe to revert it, click Revert.



NOTE If other users are detected as working in the file that you want to revert, the Revert dialog will warn you.



In this case, you might want to click Cancel. Then, communicate with other team members about the conflict, and decide what action to take that works for your team.

9. Even though the revert updates the workspace version to what it was, it still needs to be committed to the repository. Click Commit in the upper-right corner of the editor.
10. In the Create New Commit dialog, enter a Commit Message.
11. Click Commit. The reverted file displays as the latest committed version in the repository.

NOTE If other authors are editing a file that is reverted, they will see a warning banner display at the bottom of the editor alerting them to the fact. Unless your intent is to overwrite other people's content, it is best to determine the status of the workspace and of other users to avoid problems.

EXAMPLE Multiple users are collaboratively editing a file. One user accidentally deletes all the contents of the topic (and the other users notice deleted text in real-time as they work). The team communicates about the issue and all agree to revert to a previous version of the file. The advantage to this feature is that you can select a previously committed file or select one of the auto-bundled versions to restore. One of the authors is able to identify the version where all the content and most recent edits exist, and reverts to it. The others see the reverted file in their own editor and continue editing until they are ready to commit. According to the timeline, the new version of the file displays at the top of the history as the latest version.

Workspace Overview

The Workspace Overview is essentially a report of the project's current collaborative authoring state. Managers and authors alike may find it useful for monitoring the workspace and for identifying items, such as files not committed, user status (e.g., who is working on a file or who has completed a file), and what needs to be done.

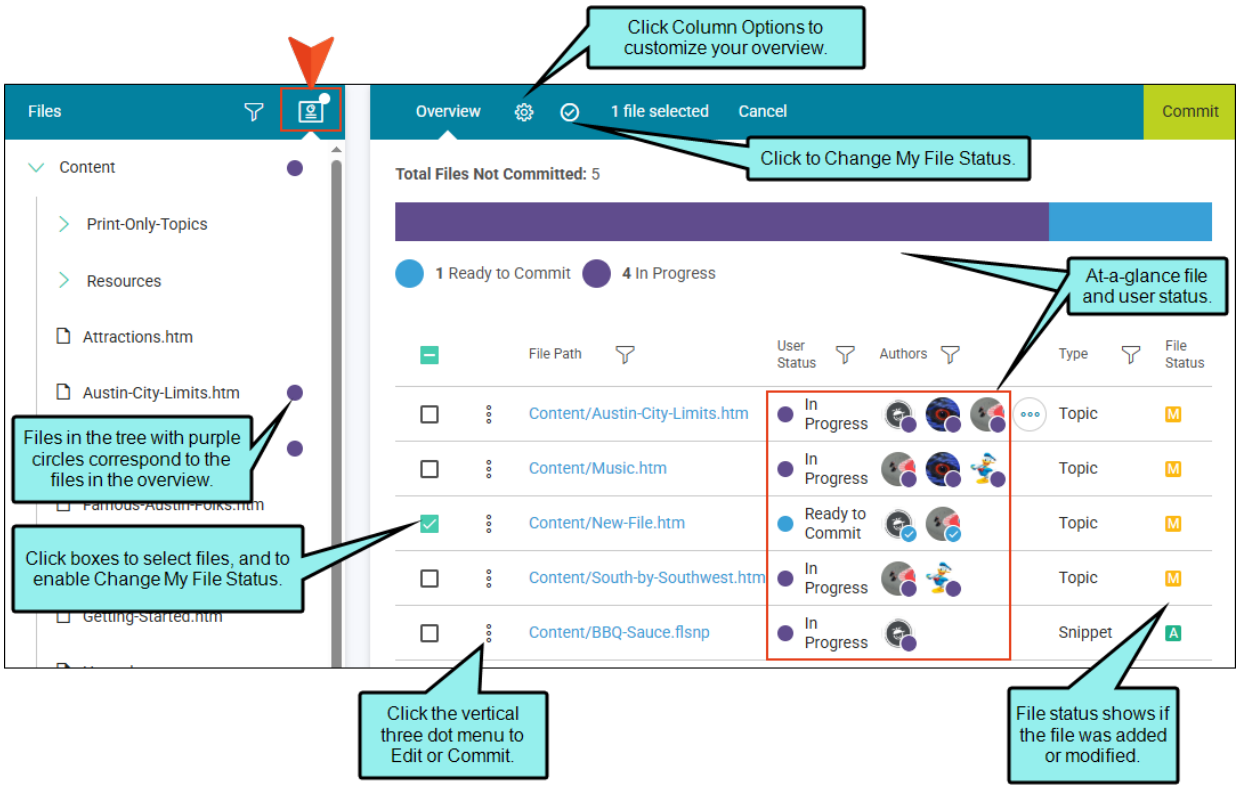
You can customize the columns that display in the overview and perform certain tasks like changing the file status, launching the Content Editor from a file listed, or committing a file. You can also commit many files at once in bulk, to the Git repository.

How to Open the Workspace Overview

1. On the left side of the Flare Online interface, click **Projects**.
2. Select a project to open it.
3. Click the **Workspace** tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.
5. Click  . The Workspace Overview opens to display various items (e.g., status, authors, type) about edited files in a project. (If files are already in a working state, the Open Workspace Overview button shows with a circle in the upper-right corner.)

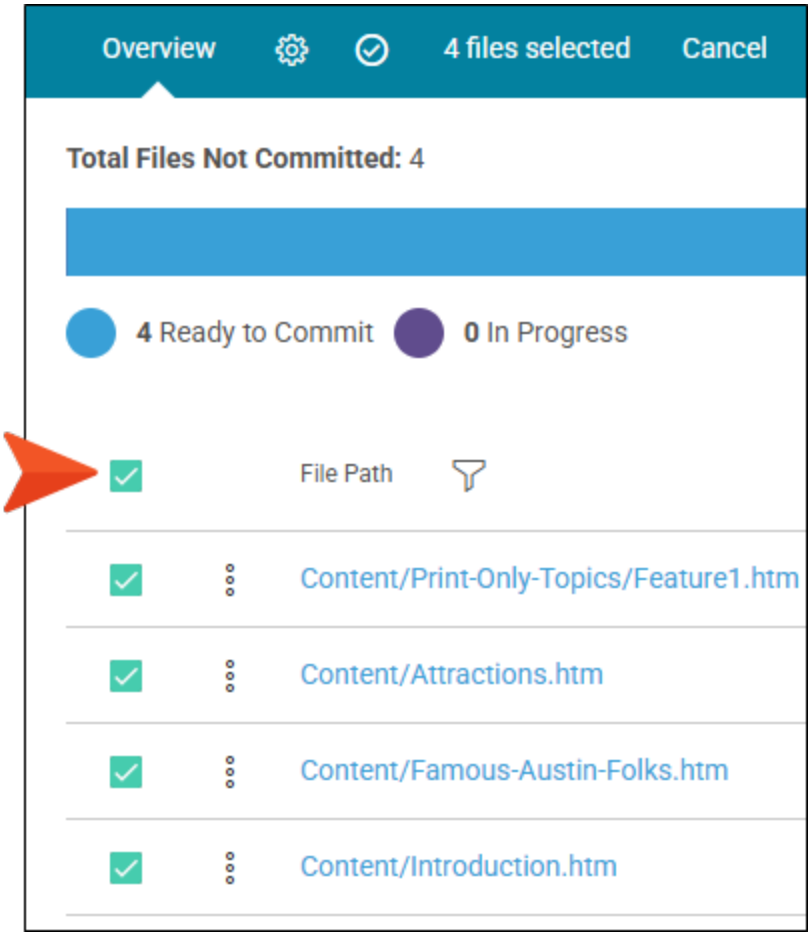
   If there are files uncommitted, you will see a snapshot of the total files not committed (i.e., files that are either In Progress, or files Ready to Commit).

   If all files are committed (i.e., no files are being edited), the overview grid is empty.


Click Column Options to customize your overview.
Files
Overview
1 file selected
Cancel
Commit
Click to Change My File Status.
Content
Print-Only-Topics
Resources
Attractions.htm
Austin-City-Limits.htm
Total Files Not Committed: 5
1 Ready to Commit
4 In Progress
At-a-glance file and user status.
File Path
User Status
Authors
Type
File Status
Content/Austin-City-Limits.htm
In Progress
Topic
M
Content/Music.htm
In Progress
Topic
M
Content/New-File.htm
Ready to Commit
Topic
M
Content/South-by-Southwest.htm
In Progress
Topic
M
Content/BBQ-Sauce.flsnp
In Progress
Snippet
A
Files in the tree with purple circles correspond to the files in the overview.
Click boxes to select files, and to enable Change My File Status.
Click the vertical three dot menu to Edit or Commit.
File status shows if the file was added or modified.

How to Commit Multiple Files at Once

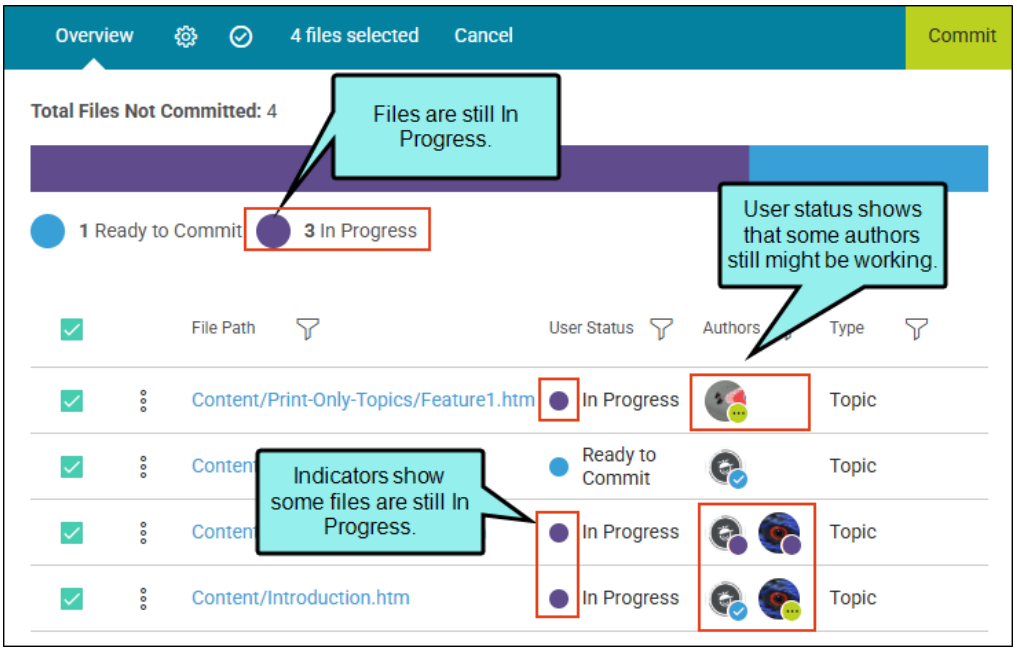
1. In the Workspace Overview, click the check box column header. This selects all the files in the list (for a bulk commit). You can also deselect a few of the files if you want to commit most of files, but not all of them.



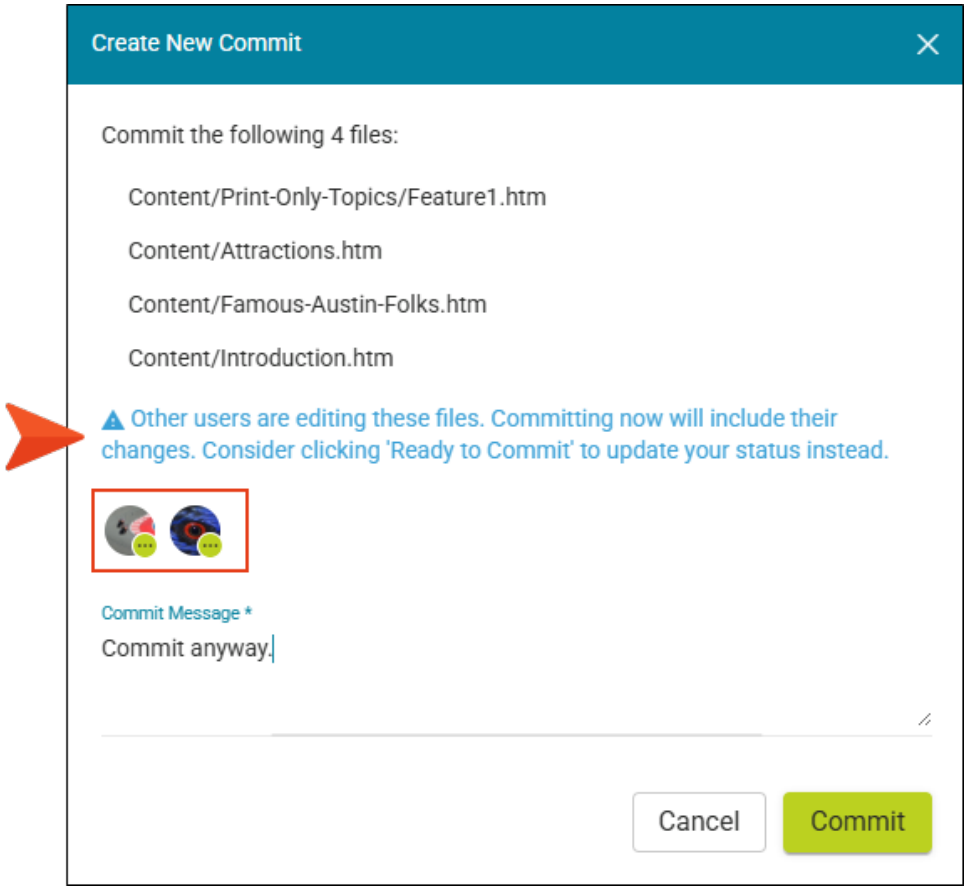
2. Click **Commit** in the upper-right corner.
3. In the Create New Commit dialog, enter a **Commit Message** and click **Commit**. Alternatively, you can click **Cancel**.

EXAMPLE Before you do a bulk commit, use the Workspace Overview to determine if committing the files selected is a good idea. In most cases, authors and files status should indicate "Ready to Commit" for all files. Otherwise, you are going to commit changes that someone else might not want committed yet.

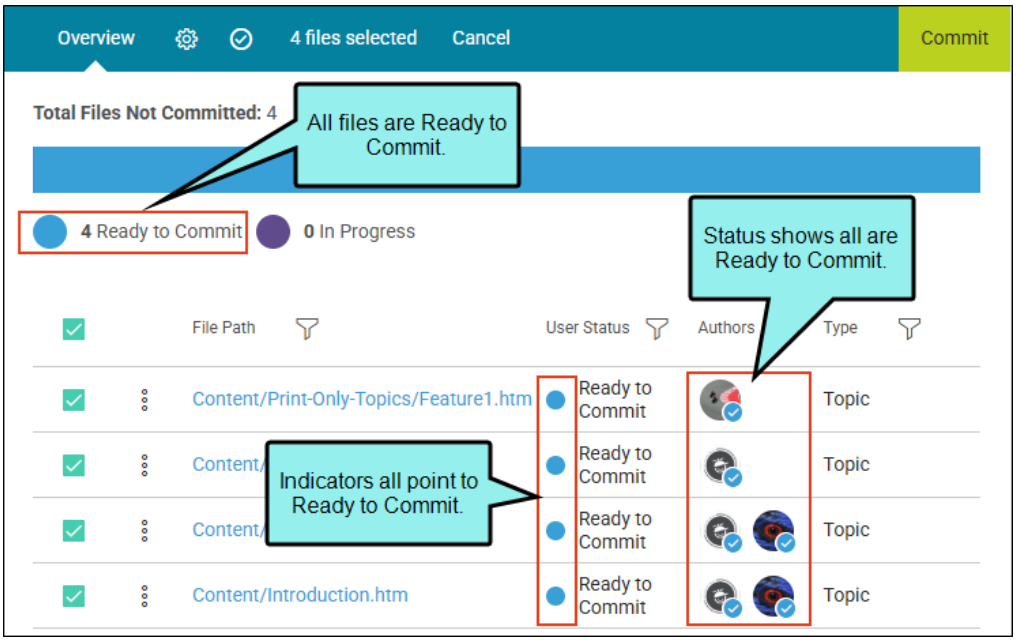
The following Workspace Overview shows an example of files and authors still at work.



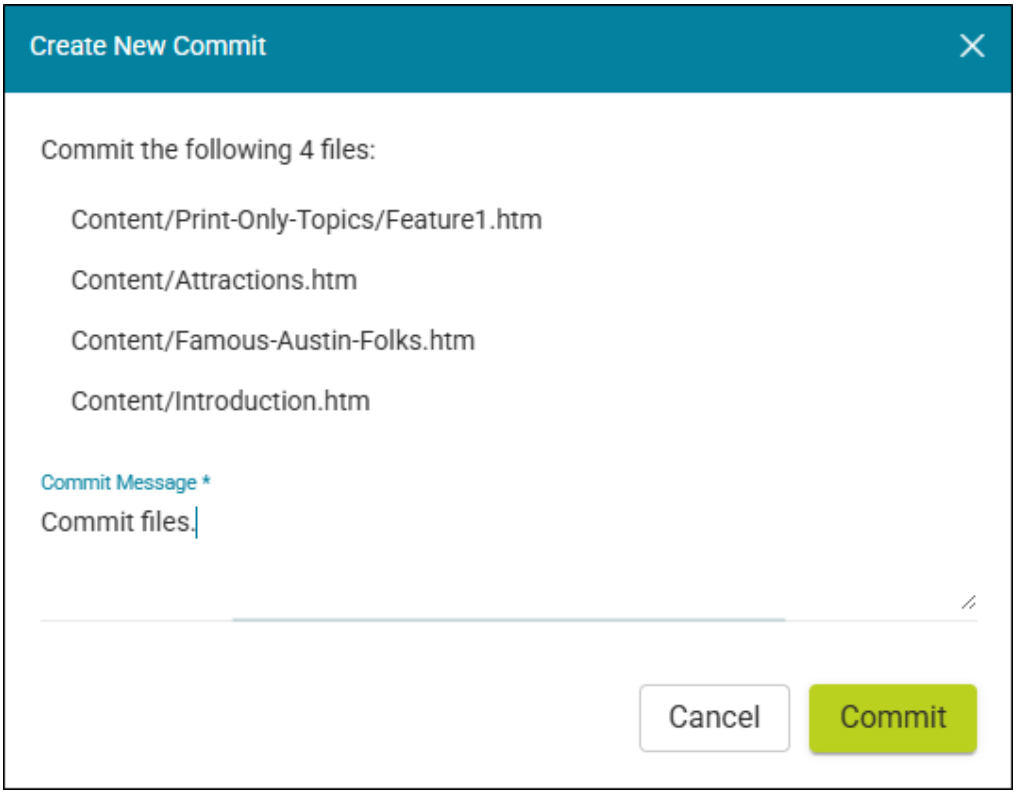
☆ It is probably not a good idea to commit these items yet. If you do, a warning displays in the Commit dialog.



This is a better example. The files and authors are all set for a commit.



☆ No warning displays in the Commit dialog. And all your co-workers are probably happy that their changes made it into the files without a problem and headache.



What’s Noteworthy?

NOTE What is the repository versus the workspace? Flare Online stores files virtually in a Git-based system. If you open and view a file, it is in a committed state in the project’s repository. If you click to edit a file, it then enters a “workspace mode,” essentially making a working copy of the file from the repository. In this mode, the file is in a non-committed state with pending changes. When editing is completed, the file needs to be committed back to the project’s repository.

NOTE What if I don't want others involved in certain files? If you plan to make changes in files, but would prefer that other authors do not make changes in those same files, there is currently no way to lock files in collaborative authoring. The best solution is to create another branch in Flare Desktopand work in that branch by yourself. You might need to ask other authors not to work in that same branch. Then later, when you are finished with all changes, you can merge that branch into another using Flare Desktop.

NOTE Certain types of files (e.g., readme, text, css) are single-commit items and exist in the repository only. Conversely, HTM files such as topics, snippets, and templates are supported by collaborative authoring. You can open multiple files in the workspace, retain all changes, and then commit them in bulk.

PDFs

The following PDFs are available for download from the Help system.

AI Assist Guide

Analytics Guide

Authoring Guide

Branding Guide

Building Output Guide

Checklists Guide

Conditions Guide

Getting Started Guide

Images and Multimedia Guide

License Management and Purchasing Guide

Links Guide

Projects Guide

Reports Guide

Reviews Guide

Security Whitepaper

Sites Guide

Snippets Guide

Source Control Guide

Targets Guide

Tasks Guide

Topics Guide

Translation Guide

Users and Teams Guide

Variables Guide

What's New Guide

Widgets Guide