

# Team collaboration

## First, be a team!

### Get to know each other

- What are their working styles?
- What is the best way you can work together?

### Communicate

- Talk. A lot. (Teams messages are good, but calls and meetings foster invaluable discussions. Don't skip them.)
- Decide as a team who will be responsible for what.
- Set clear expectations around what progress looks like, deadlines, etc.

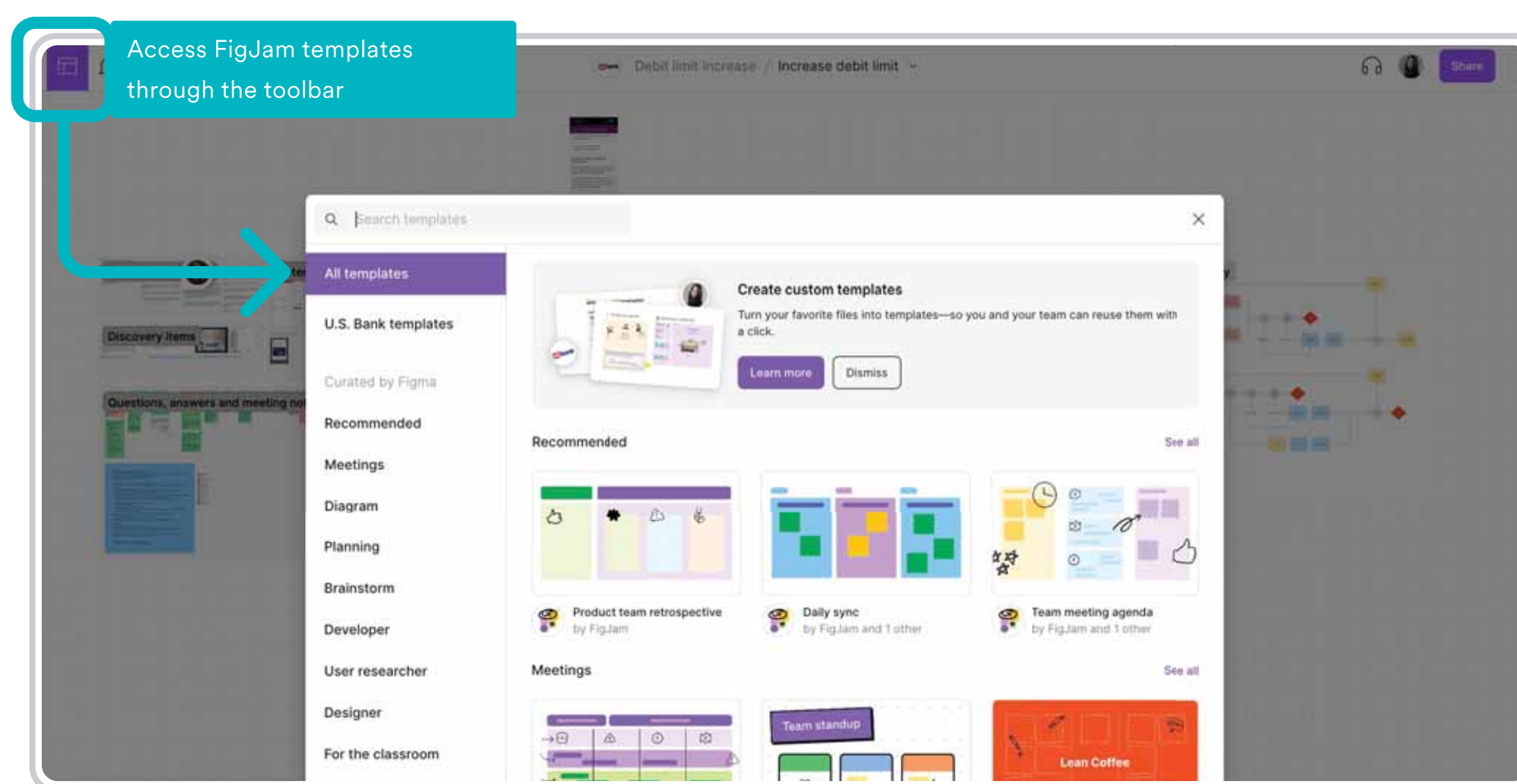
## Optional: Get the creative juices flowing in FigJam.

### FigJam: the whiteboarding tool

Much like Mural or InVision's Freehand, FigJam offers a collaborative digital whiteboard the whole team can work in simultaneously.

Though these files don't have all the same features as a design file, FigJam is a great space to share initial ideas, pose questions, gather competitor screens and start collaborating as a team.

FigJam also offers a variety of templates to help facilitate meetings, brainstorm sessions, research exercises and more. For new teams, new projects (or both!) this can be the perfect place to start.



## Next, create a collaborative working environment in your file.

### Make sure everyone has a seat at the table.

All UX Design roles are important and should be able to edit the file. Each team member should be comfortable doing the following:

- Working in Figma
- Navigating documents
- Working with components and variants
- Editing text

If you have teammates who aren't comfortable completing these tasks, set up some training time. The best way to get comfortable with a new software is to get in it and start playing around! The [Figma Standards & Education team](#) can be a great place to start.

### Build edit-friendly layouts.

- Use autolayout to keep your page spacing consistent.
- Create [content components](#) to allow for global editing.
- Turn common mockups into local components for consistent display and easy edits.

### Track and mark changes.

- Create notes to keep your team informed about updates.
- Call out changes for developers.
- Keep a revision history.

# Working in Figma

**TIP:** You can minimize the number of notification emails by setting your profile to receive only mentions and replies in the notifications section of your profile settings.

## Keep the document tidy

Figma is a collaborative space. Keep your files organized so any colleague can find what they're looking without having to track you down.

- Use Frames (F) or Sections (shift + S) to group related screens.
- Only add finalized screens to the wireframe page, and keep WIP separate.
- Merge/delete/archive branches when you're done with them.



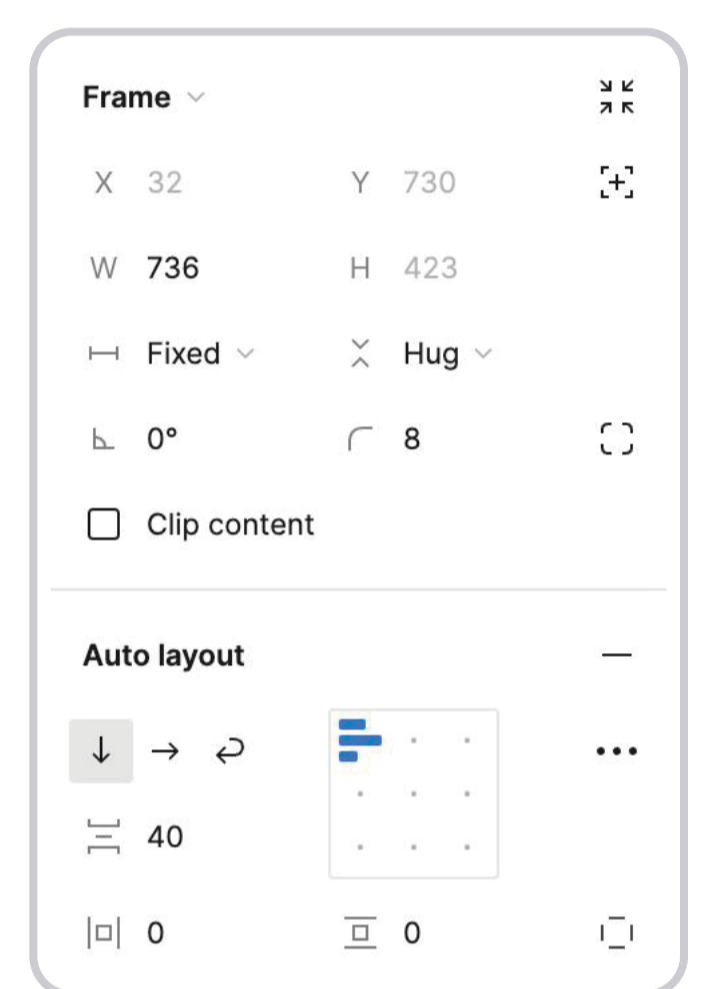
## Use auto layout

When building out your designs, be sure each component and screen has an appropriate auto layout set up so content can flex without overlapping other screen elements.

Auto layout allows you to set the following parameters:

- Margins
- Spacing between elements
- Wrapping rules (fixed width, fill container, hug contents)

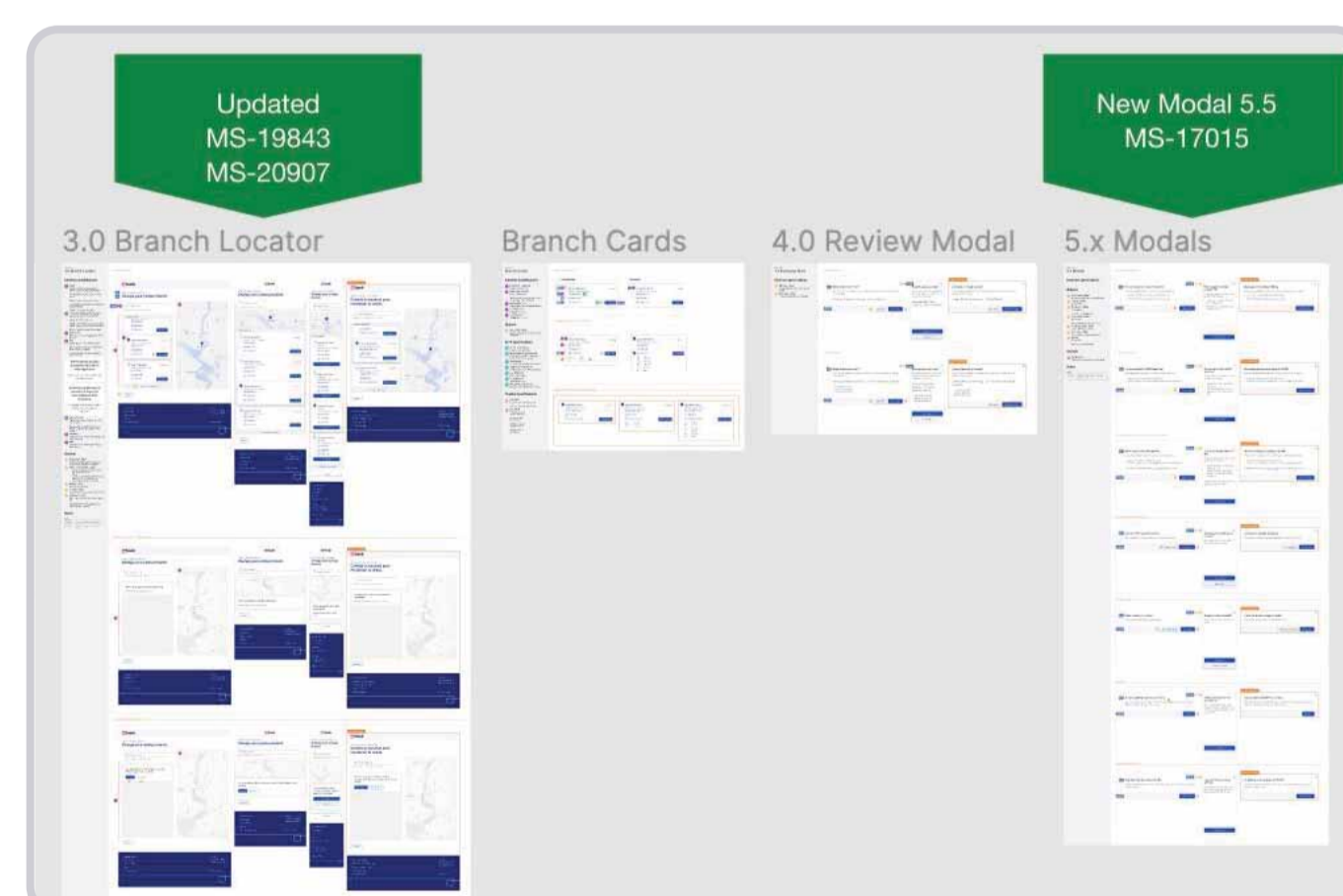
Check out this [guide from Figma](#) for more information or explore the [auto layout playground](#).



## Naming and labels

Name your branches, pages, sections, layers, groups and frames. Names should be simple and recognizable. Use title case and avoid atomic design language.

To ensure updates are easy to find, add large-font labels and visual indicators.



## Less is more

When possible, only create what is necessary for groups, layers, frames, etc. Avoid over-nesting.



- [Home](#)
- [Team collaboration](#)
- [Working in Figma](#)
- [Content in Figma](#)
- [Components](#)
- [Variants](#)
- [Branching](#)
- [Accessibility](#)
- [Version history](#)
- [File delivery](#)
- [Go to standards](#)

# Content in Figma

## Tips, tricks and shortcuts

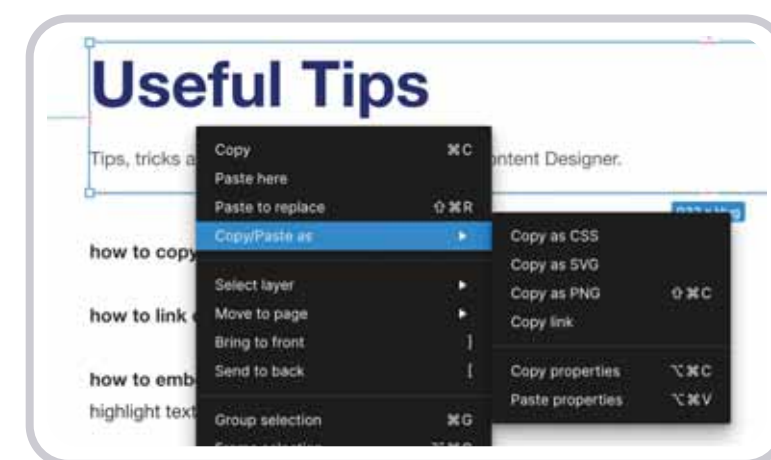
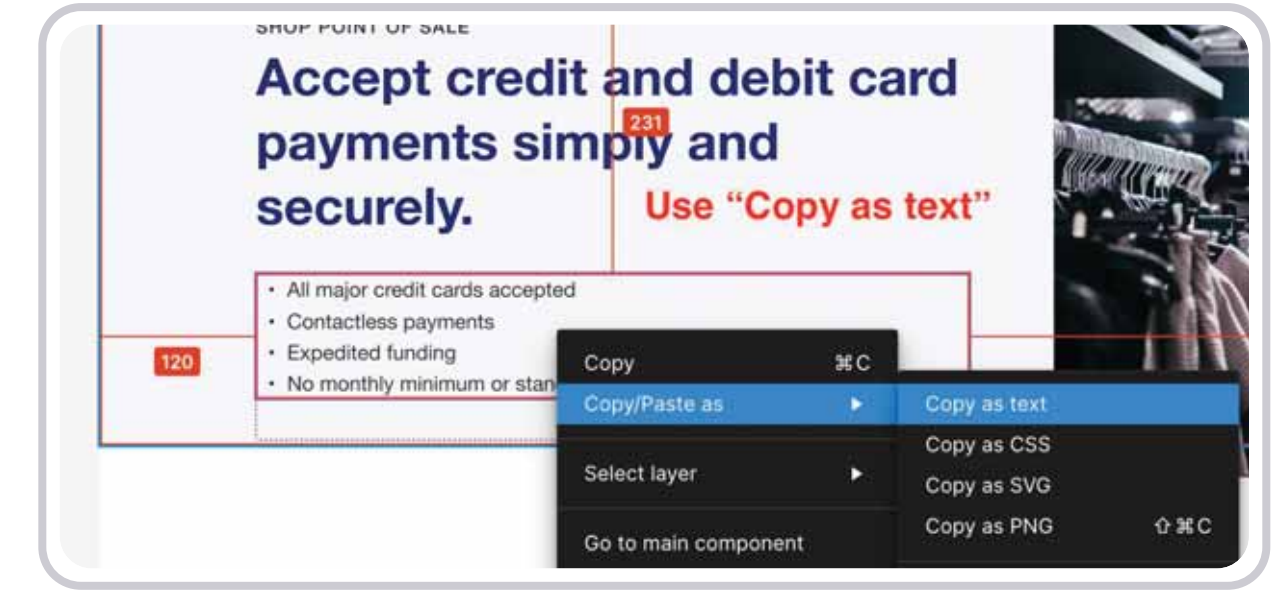
### How to...

#### Copy/paste text if you don't have edit access

- Select textbox > right-click > Copy/Paste as > Copy as text
- Don't use right-click > Copy

#### Link directly to a frame within a page

- Select the frame > right-click/CTRL > Copy/Paste as > Copy link
- Link will take you directly to the frame, not just the page.

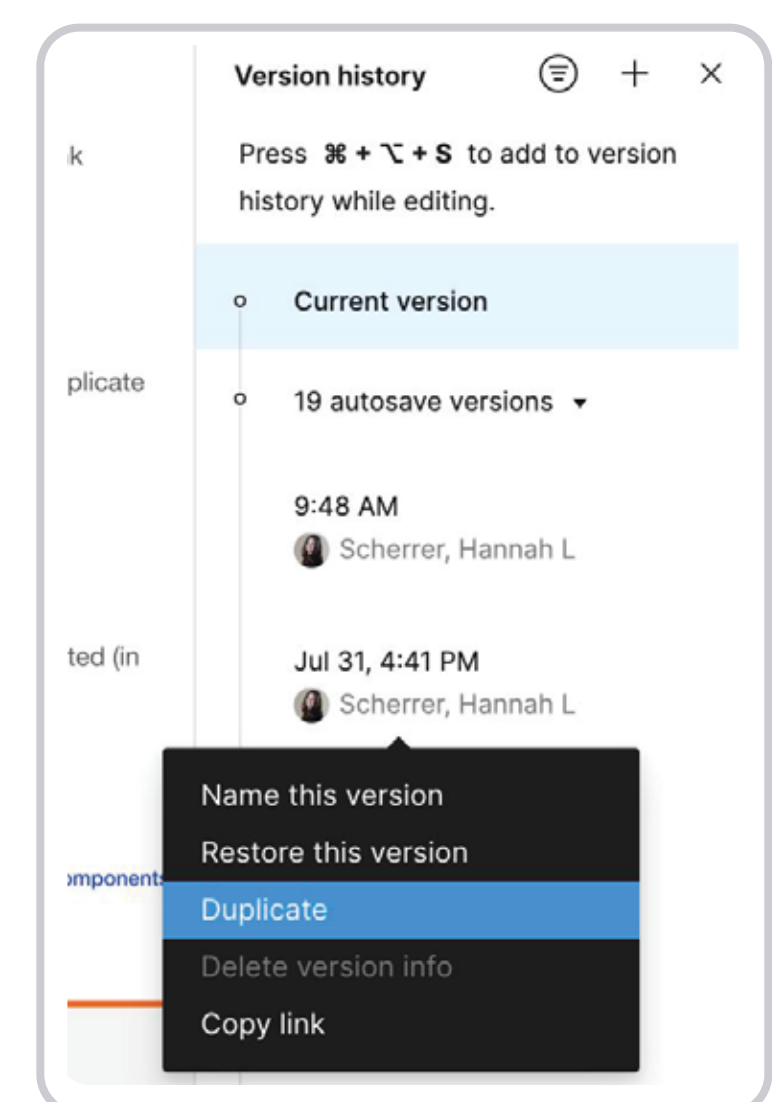


#### Recover content from a previous version

Previous versions are view-only. To copy content, you'll need to duplicate a version. Don't use restore, or you could affect updates to the file.

#### To duplicate:

- Select the version you want to recover.
- Click the horizontal menu next to it, and select Duplicate.
- Figma will show a notification that the version has been duplicated (in Drafts).
- Click Open to open the duplicated file in a new tab.



#### Shortcuts for content (Mac)

##### Figma-specific

- Keyboard shortcut menu: ctrl + shift + ?
- Show/hide comments: shift + c
- Show/hide side nav bars: cmd + .
- Embed a link in text: highlight text > cmd + k
- Select a text layer in one click (deep select): hover over field > t
- Use find and replace:
  - cmd + f
  - Search box appears in top-left, just under Figma navigation.
  - Switch to 'replace' mode or change other properties in 'Settings' – to the right of the search box. Settings icon:

##### All applications

- en-dash: option + hyphen key
- em-dash: shift + option + hyphen key
- ellipsis: option + ;
- trademark ™: option + 2
- copyright ©: option + g
- registered mark ®: option + r
- Single left quote ‘: option + ]
- Single right quote ’: shift + option + ]
- Double left quote “: option + [
- Double right quote ”: shift + option + [
- Nonbreaking space: option + space
- Character viewer: ctrl + cmd + spacebar (or try fn key)

[Learn more about our Content in Figma standards >](#)

- Home
- Team collaboration
- Working in Figma
- Content in Figma
- Components**
- Variants
- Branching
- Accessibility
- Version history
- File delivery
- Go to standards

# Components

## Why use components?


Components save us from unnecessary rework. Instead of having to redesign standard screen elements, you can use the already-baked component from the Shield library (or other libraries as needed).

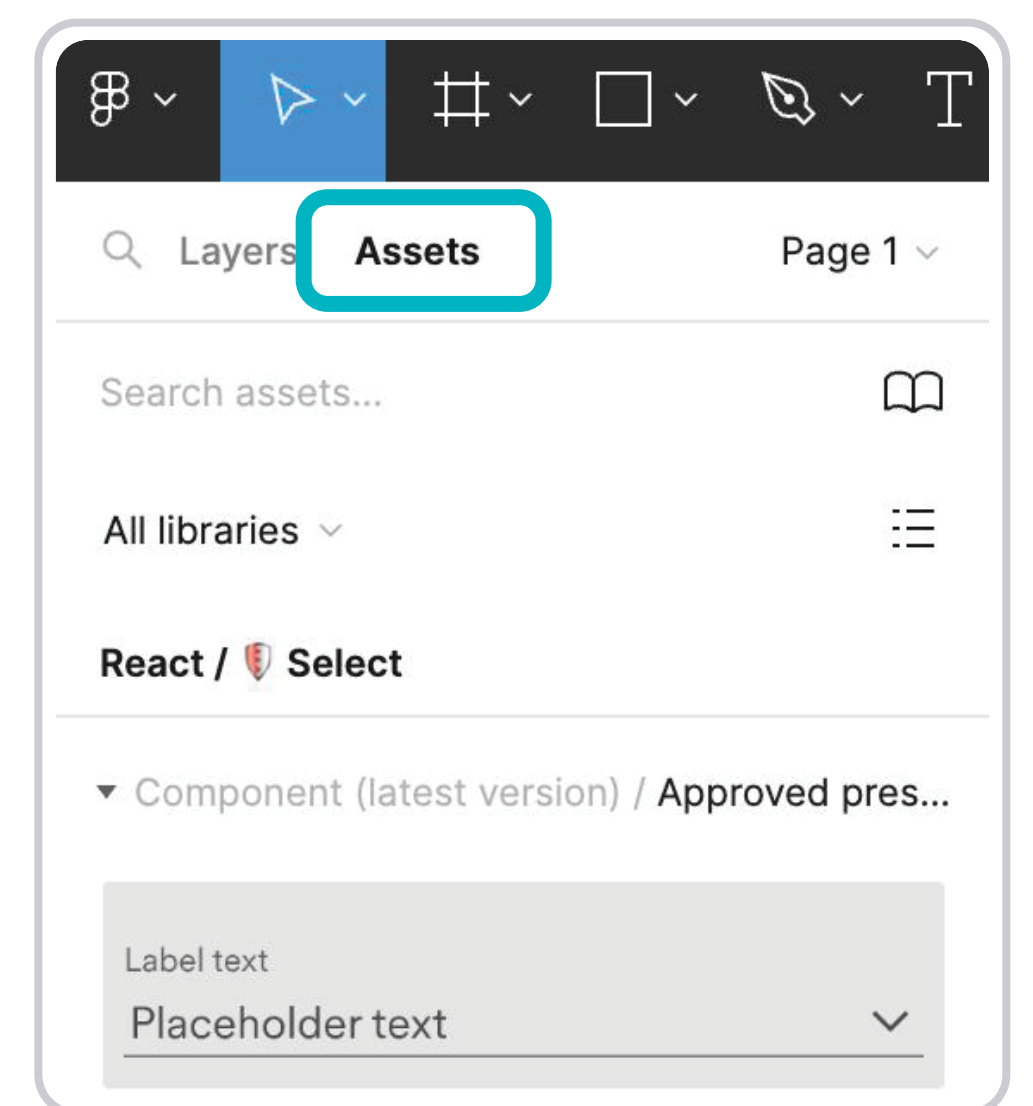
By using instances of the same component across screens, you can select, edit and update all instances in one place.

## When to use components

Short answer: Always. Use [Shield components](#) as the building blocks of your designs. From there, create local (in-file) components to control content and designs in one place, no matter how many instances you need.

## How to access components

- Go to your file's Assets tab.
- Select the library dropdown or use the search field to find the necessary component from your connected libraries.  
Each bold accordion label signifies an individual library file. In our example, we've opened the React /  Shield Select library component.
- Drag and drop what you need into your file.




## How to use components for content management

Content components are local main design components that you use as your content source of truth.

Use them by creating components out of appropriately styled headings, text and other elements (including Shield components). Your new component is your single source of truth.

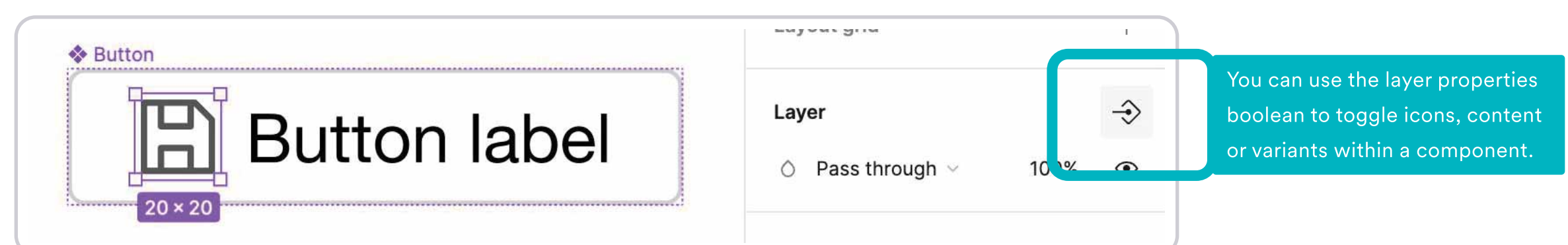
Keep your file clean and manageable by organizing these components into their own clearly labeled section or frame.

 **TIP:** Need to use the same content across multiple files? Consider using the Qi method and creating a [Single Source library file](#).

## Avoid multiple versions of the same component

Make use of variants within Figma by combining similar components into variants. As an example, don't make 10 buttons when 1 button with 10 variants will do.

Use layer booleans to show or hide elements instead of creating excessive variants.



[See what Figma has to say about Components](#) >



- Home
- Team collaboration
- Working in Figma
- Content in Figma
- Components
- Variants**
- Branching
- Accessibility
- Version history
- File delivery
- Go to standards

# Variants

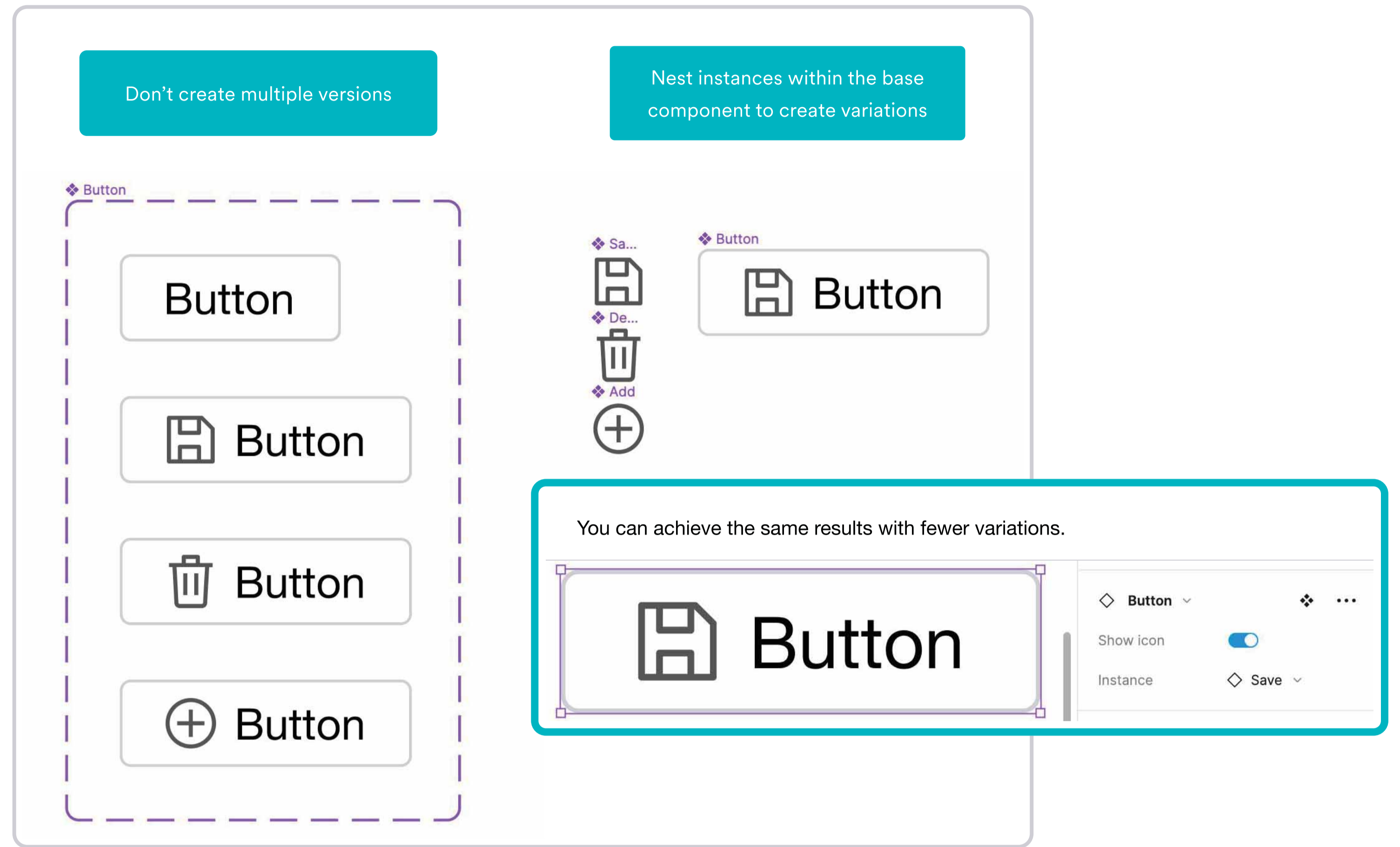
## Why use variants?

- Maintain localized and selectable component variations.
- Simplify instances with configurability. Adjust the properties instead of creating a separate screen variation.
- Replace existing components instantly using a toggle or dropdown.

## When to use variants

When you have multiple versions of similar components or content.

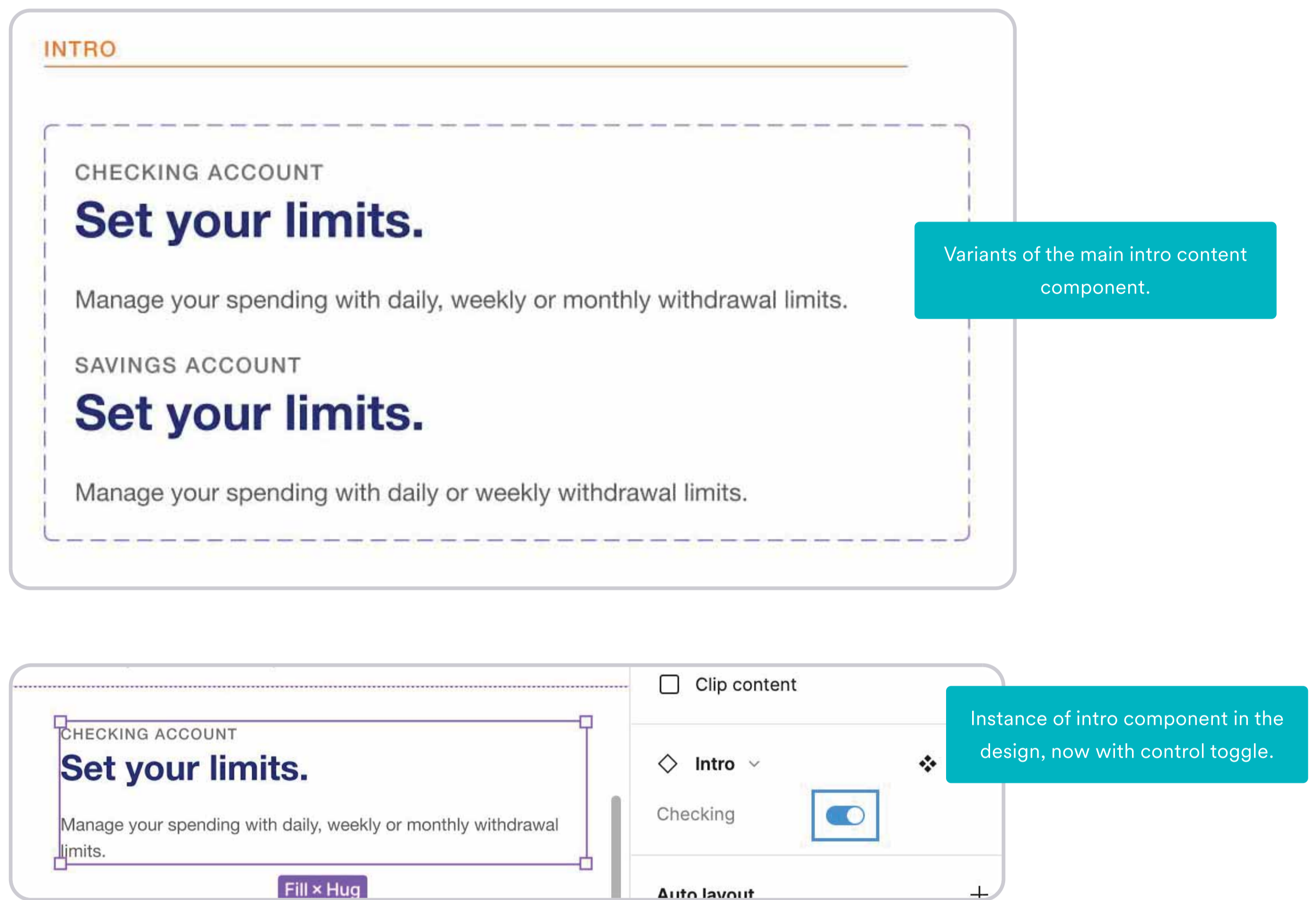
Avoid creating too many variations of a component to the point it becomes unmanageable. Consider nesting instances within a component (using interchangeable parts and layer booleans) or use base components instead.



## Using variants for dynamic content options

Do you have pieces of copy that need to change slightly when presented to different audiences? Instead of creating a content component for each audience, use variants to easily swap the text without having to change the components within the design.

In the example below, the intro content component has two variants, one for checking accounts and one for savings. The instance in the design now has a toggle option to swap between the checking intro and the non-checking intro.

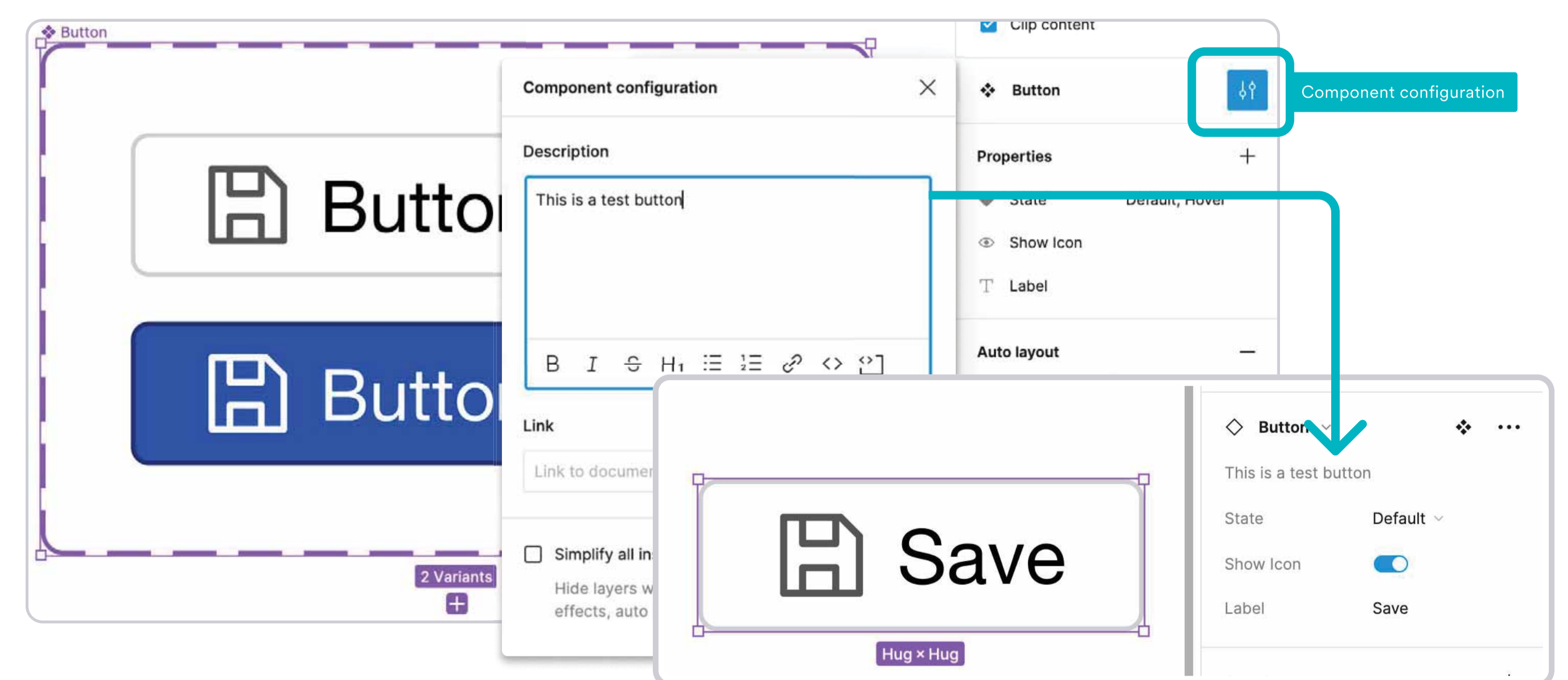


## Follow Shield conventions when naming variants

Toggle variant: use "yes" and "no" as the options to stay consistent with Shield guidelines.

## Adding descriptions

Components, variant groups and individual variants within a group can have a description added to help describe and define usage.



## Be consistent with variant properties

Name layers, groups and auto layouts.

Keep property names, values and order consistent.

[See what Figma has to say about Variants](#) >



- Home
- Team collaboration
- Working in Figma
- Content in Figma
- Components
- Variants
- Branching**
- Accessibility
- Version history
- File delivery
- Go to standards

# Branching

## What is branching?

Branches are exploratory spaces that allow designers to safely try new ideas without affecting the main design file. When you're ready, the changes from your branch can be merged into your main file.

## When to use branches

### Contributing to a shared library

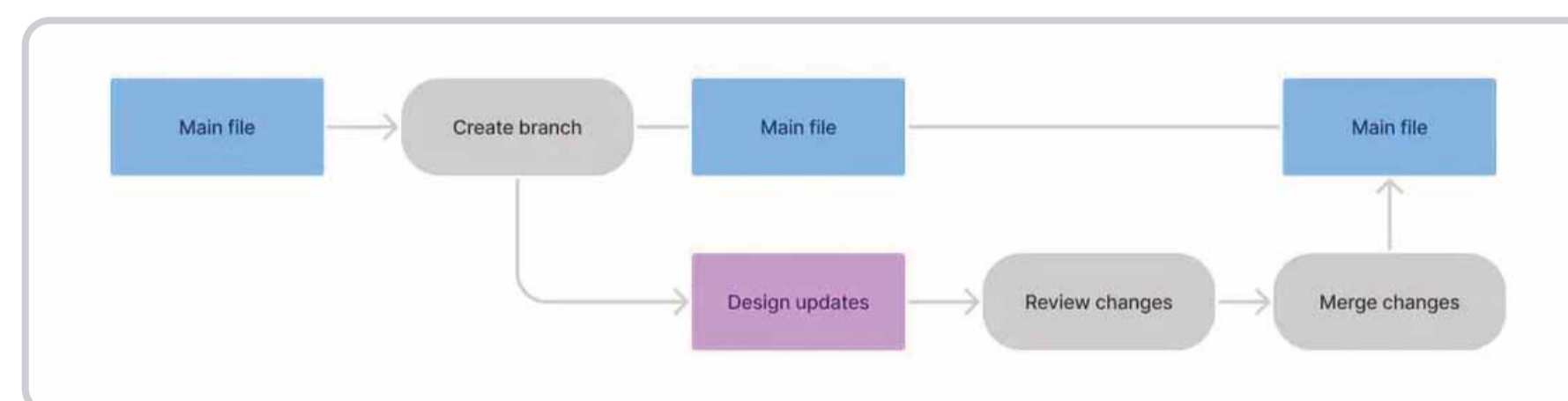
Non-editors can create a branch to demonstrate their proposal for a component modification or addition to existing libraries.

### Modifying components or variants

Test modifications to complex components, create new components and variants or modify a shared style without breaking your main file.

### Freezing features for dev handoff

If your team is delivering in sprints, you can preserve the main file for development while you work a sprint ahead. Create a branch for the next sprint that your team can work in. You can use this branch for getting product owner approval and sharing updates with your team. When it's time to hand off the updates, merge the changes into the main file.



## Best practices when using branches

### Naming branches

Name your branches to clearly communicate what they contain or address. Prefix the branch name with something to quickly identify it. This could be a story number, type of work or similar:

- UX-5141: Redesign login screen
- Layout fix for payment info
- [contrib] - Propose a new design for tabs

### Requesting review

If you are a viewer on a file you won't be able to merge the branch directly and need to request a review from an editor.

When requesting a review give as much context as possible. Fill out the description field with the details of your proposal or issue being solved to help the editors know what they are reviewing in your contribution.

### Merging branches

When you want to apply your changes to the main file, you can merge the branch. If you're an editor on the main file, start the merge to review any changes from main in the branch and resolve any conflicts that appear.

After a branch has been merged into the main file, it's automatically archived along with all comments made within it and can no longer be restored. If you need to undo a merge, you can restore a previous version of the file from the version history.

### Archiving branches

When you have designs in a branch that you've decided not to move forward with right now, you may want to archive it in case you or your team want to come back to it at a later time. If your branches have been archived without being merged into the main one, they can be viewed or restored at any time, making it easy to recover them if you need to revisit an idea.



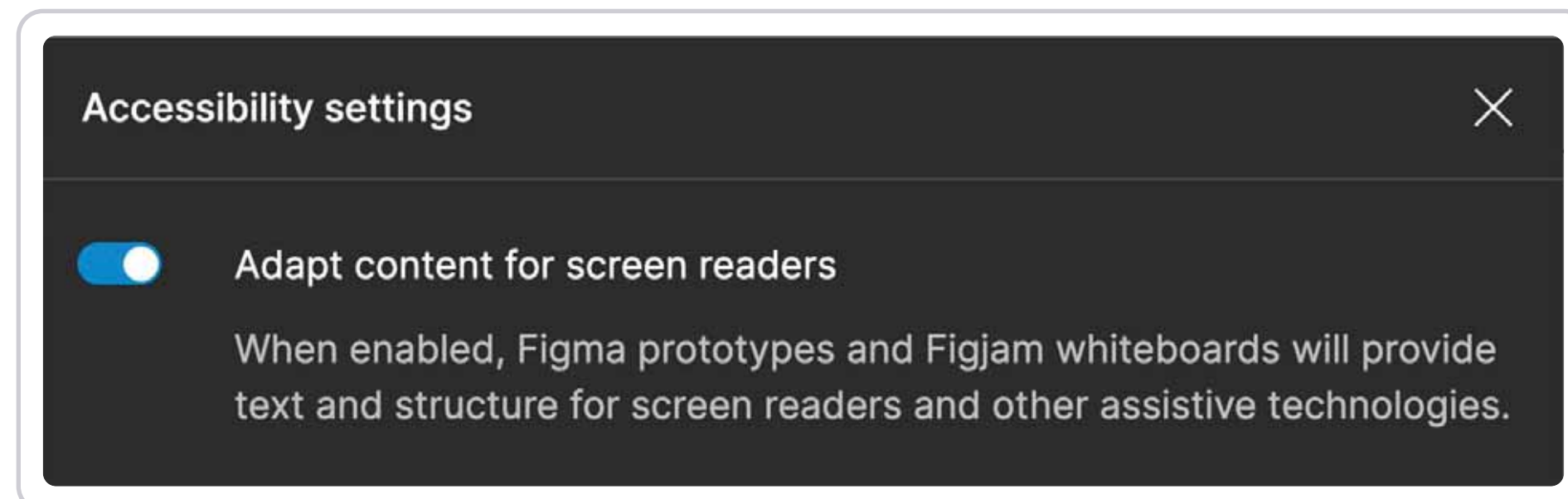
- [Home](#)
- [Team collaboration](#)
- [Working in Figma](#)
- [Content in Figma](#)
- [Components](#)
- [Variants](#)
- [Branching](#)
- [Accessibility](#)
- [Version history](#)
- [File delivery](#)
- [Go to standards](#)

# Accessibility

## Create screen reader friendly prototypes

The “adapt content for screen readers” feature in Figma converts any prototype into a basic HTML page. This HTML page can be navigated and interacted with via a screen reader like VoiceOver, JAWS or NVDA.

This feature is activated at the user level and can be found when viewing a prototype in the Options menu under Accessibility settings.



### How does it work?

When “adapt content for screen readers” is turned on, and the HTML page is generated, the following occurs:

1. Top-layer frames , components , instances , and overlays become an article element.
2. Text is wrapped in a paragraph tag and links inside of text will be wrapped in a link tag.
3. Lists will be turned into ordered/unordered lists.
4. Any shape with an image fill will become an image, and the layer's name becomes the alt text `<img alt= "layer's name">`. This includes any non-top-layer-frame that only houses an image.
5. An element with an "On Click" interaction will be converted into a button or a link depending on the type of interaction.

Screen readers can navigate through the page's text and will hear the names of any layer, frame, component, or overlay. They can also interact with any buttons or links generated by "On Click" interactions.

### How is it different from a native web experience?

Screen readers rely on a web page's programmatic structures, like headings and form fields, for context and navigation. Prototypes in accessible mode don't have some, or all, of these structures.

That's okay since prototypes aren't meant to be full-fledged web experiences. That said, we can take some small steps to pass some of the design's structure to the screen reader and ensure what gets read is in the order we want.

### How are screen reader friendly prototypes used?

#### Team members who use screen readers

Screen readers are used by people who are blind, have low vision, or have reading differences like dyslexia. These prototypes ensure all team members can reference your designs in prototype mode.

#### Accessible research

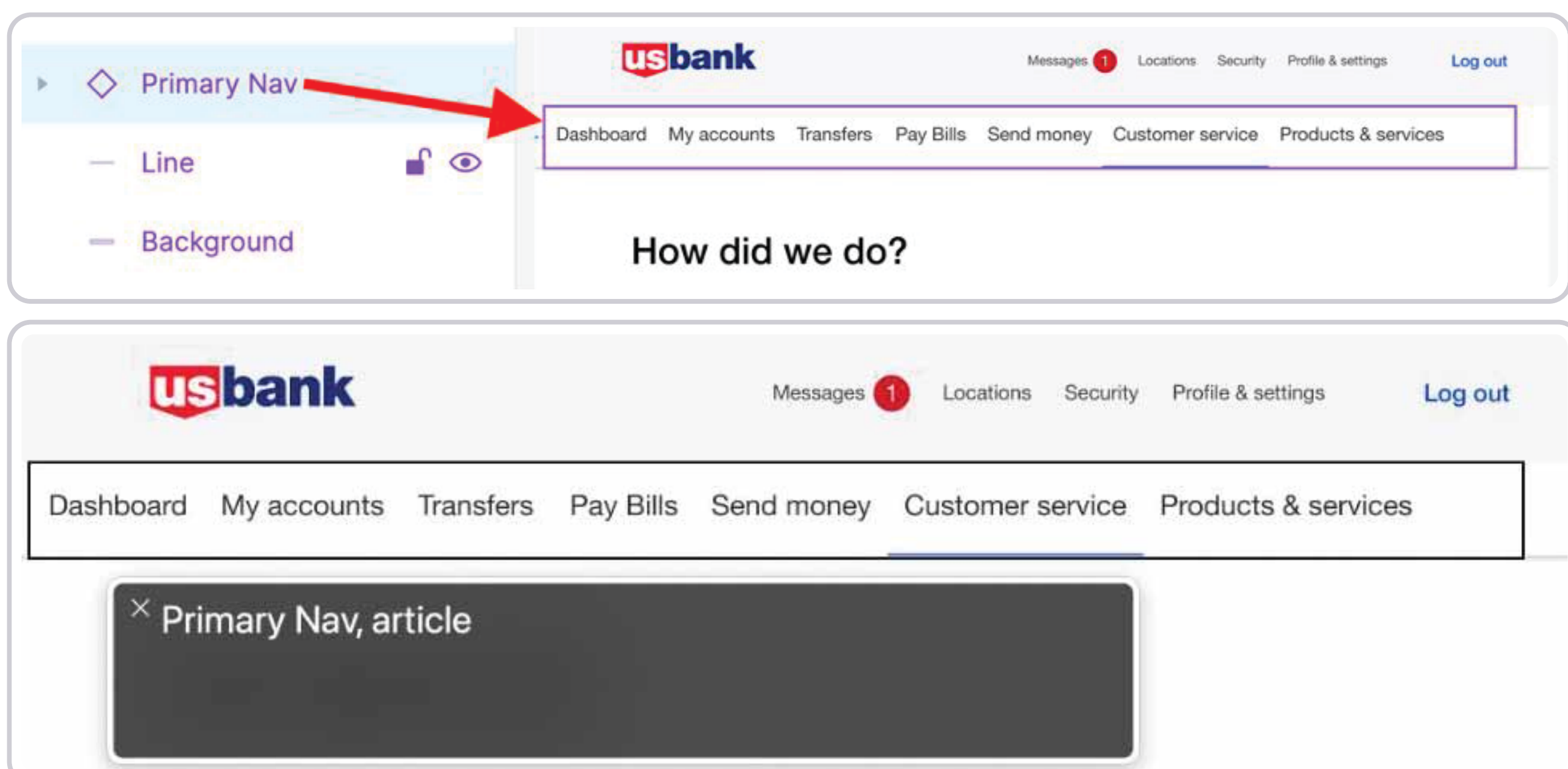
Screen reader friendly prototypes allow research to include a more diverse set of participants. Individuals who use screen readers can explore your design and give feedback which helps to ensure your designs are fully accessible.

## Tips for optimizing screen reader friendly prototypes

You'll notice the tips for ensuring prototypes are screen reader friendly are in many cases the same best practices provided throughout this guide. That's because good design and accessible design are the same.

### Components

Since components become articles when adapted for screen readers in prototype mode, they are a great way to provide some semantic structure. When a screen reader encounters the component, it will announce the corresponding region in the design.



The “Primary Nav” component example above shows the VoiceOver screen reader's caption panel displaying what the screen reader announces, “Primary Nav, article.”

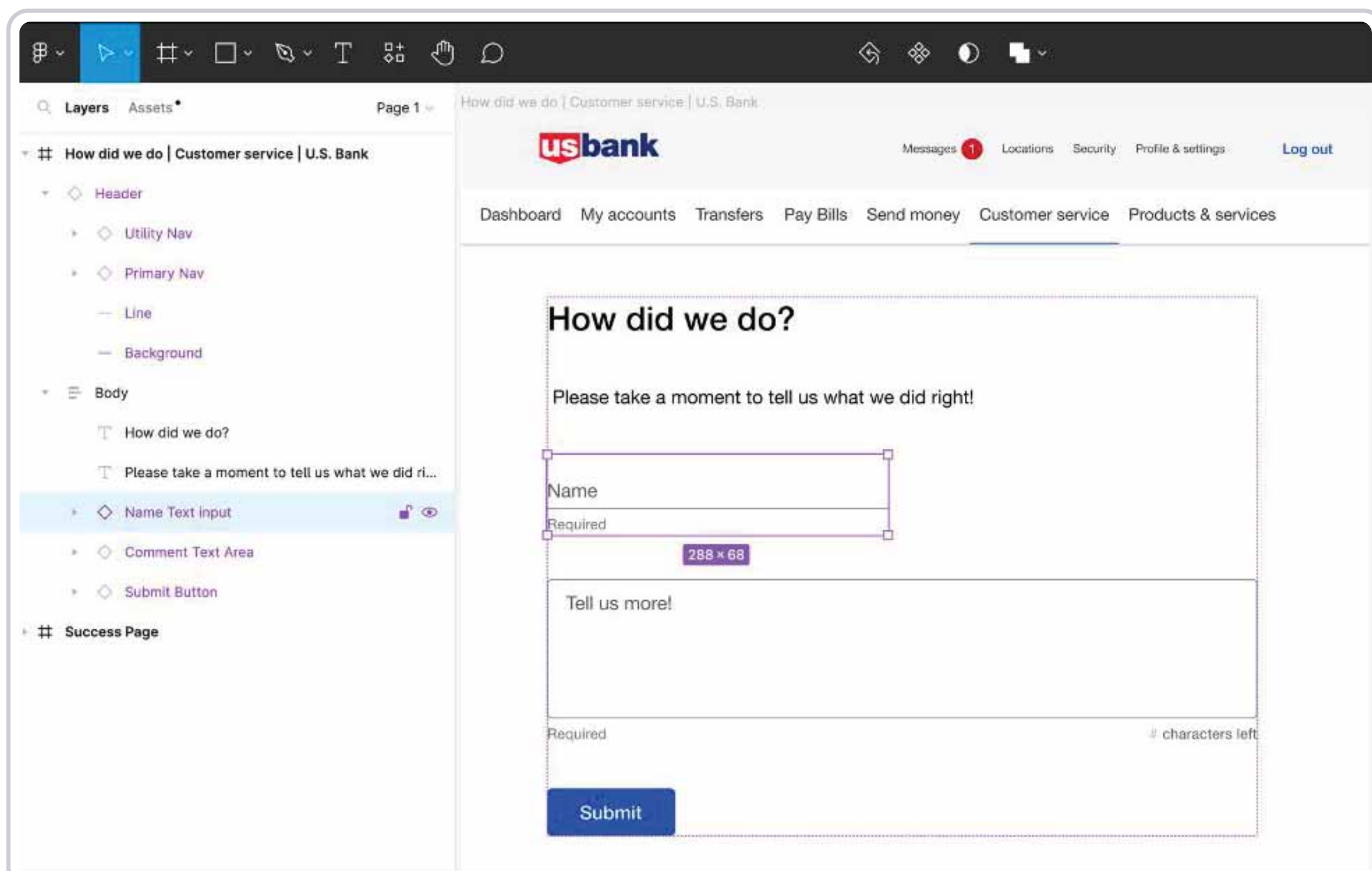
### Give layers understandable names

As shown in the example above, screen readers announce whatever name you've given your top-layer frames , components , instances and overlays from the layers list. These elements can also be presented in a list and used as navigation for screen reader users.

In order for your prototype to be understandable and navigable you must use clear, descriptive names for each of these elements.

### Layer order = reading order

When rendering your prototype, Figma will “build” your design by reading your layers list from top to bottom. It is essential that the order of your layers list matches the visual order of your design (commonly left to right, top to bottom). Otherwise, screen reader users will be presented information out of order.



In the example above, frames and instances have all been given clear, understandable names in the layers list. The screen reader user will hear these readable names as they navigate the prototype.

The top-level frame has been given the intended page title. When a screen reader encounters the beginning of the prototype, they will hear the frame's name announce, similar to native web.

[Learn more about our Accessibility standards >](#)

[Continue to Version history](#)



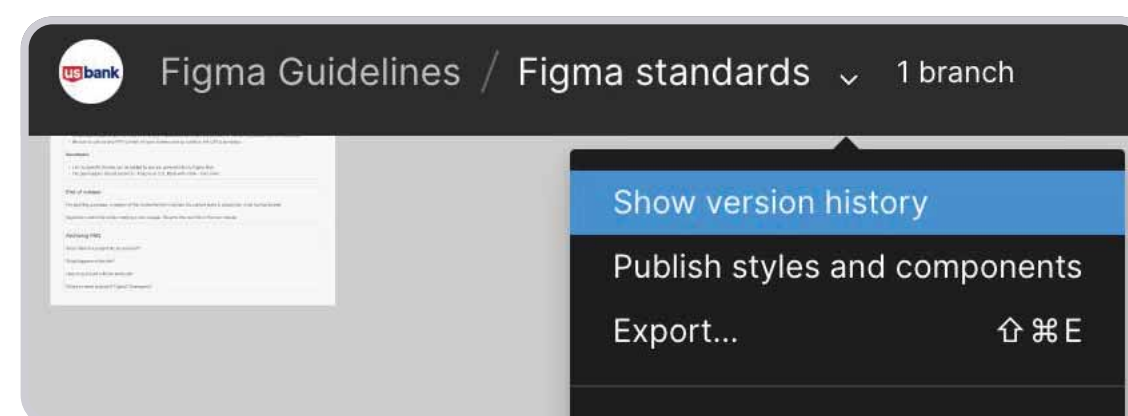
## Figma Best Practices

- Home
- Team collaboration
- Working in Figma
- Content in Figma
- Components
- Variants
- Branching
- Accessibility
- Version history**
- File delivery
- Go to standards

# Version history

## Autosaved versions

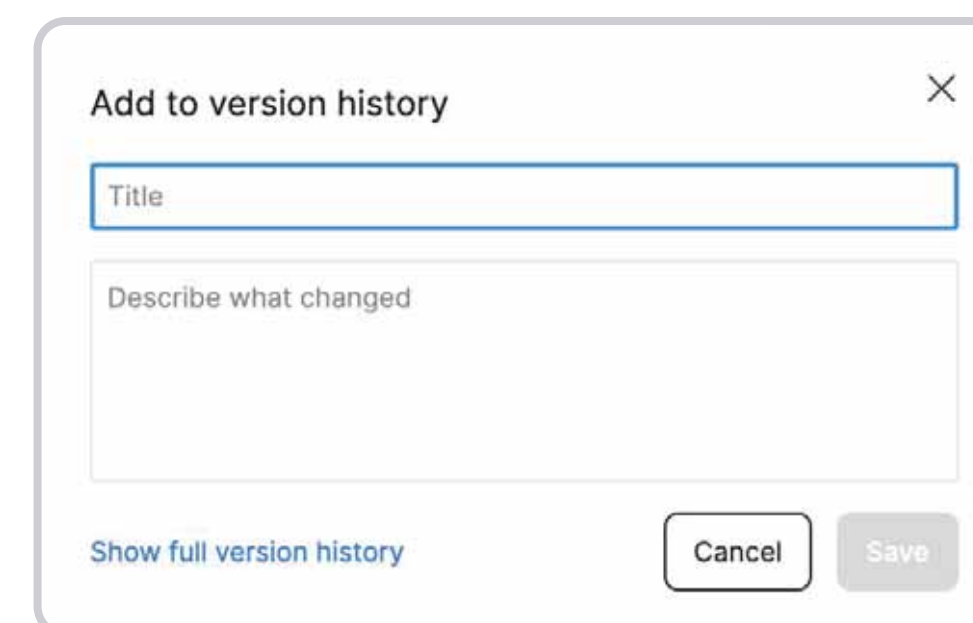
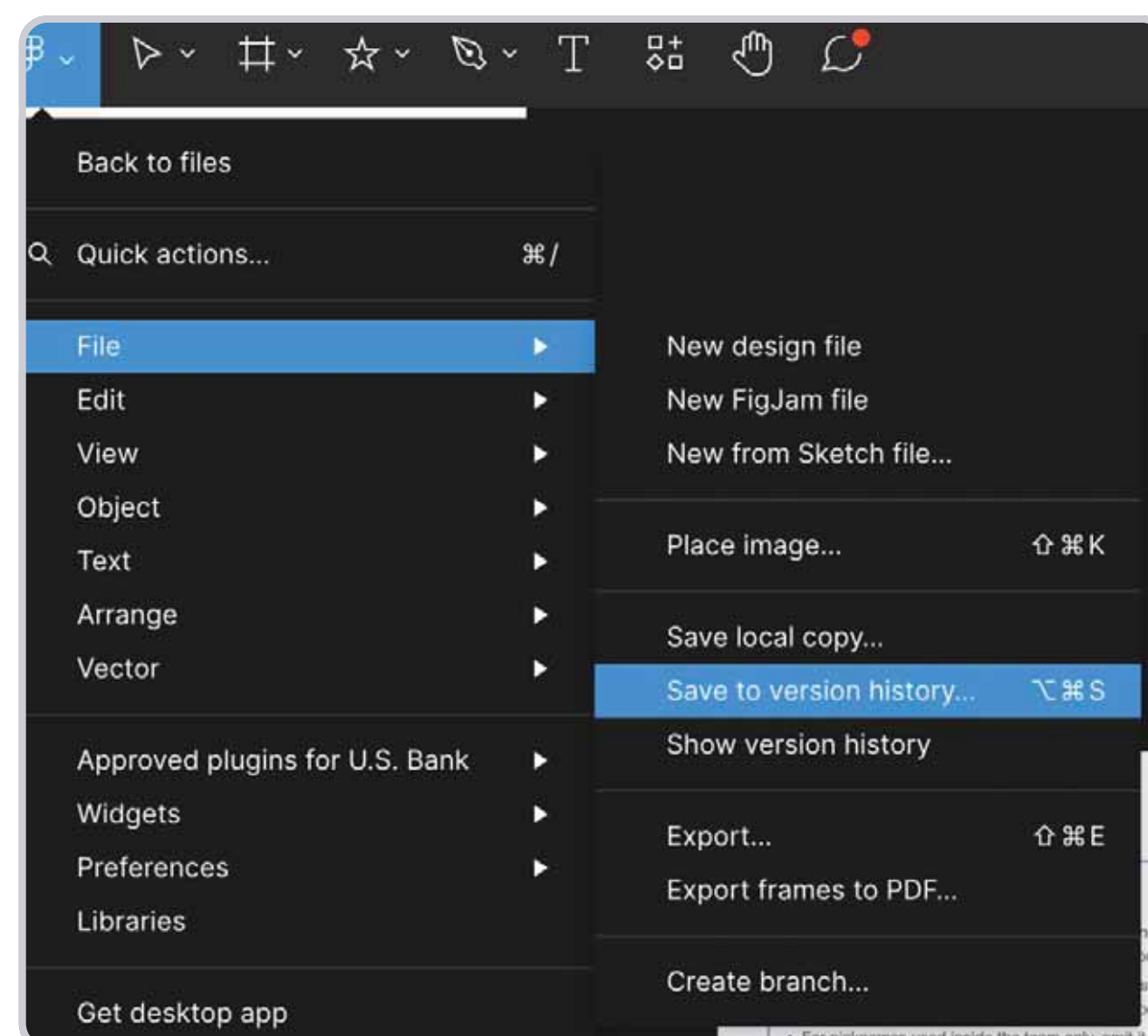
Figma records every change made to a file within its version history. You can access it through the main menu (File > Show version history) or from the dropdown at the top of your file.



## Named versions

To make your version history easier to navigate, you can save a named version. This can be helpful when you change a file name, incorporate changes from a review or make other significant updates.

File > Save to version history > enter a descriptive name (and optional detail in the “Describe” box) > Save



Example names; descriptions:

- Content manager review; Edits and responses to Logan review
- Redesign using cards; This follows feedback from Roman on 5/25/2022
- Tier 1 review comments; July 10, 2022
- Ready for final reviews; Updates by Shiv and Kendall after Tier 1

[Learn about our Version control standards >](#)

[Continue to File delivery](#)



## Figma Best Practices

Home

Team collaboration

Working in Figma

Content in Figma

Components

Variants

Branching

Accessibility

Version history

File delivery

Go to standards

# File delivery

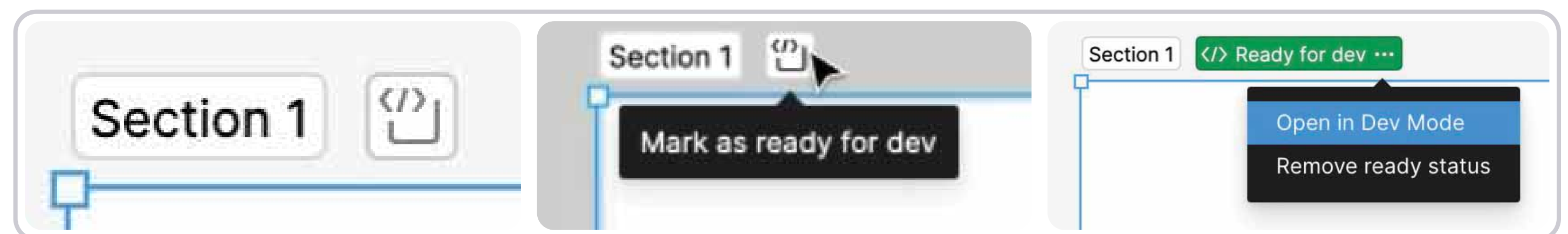
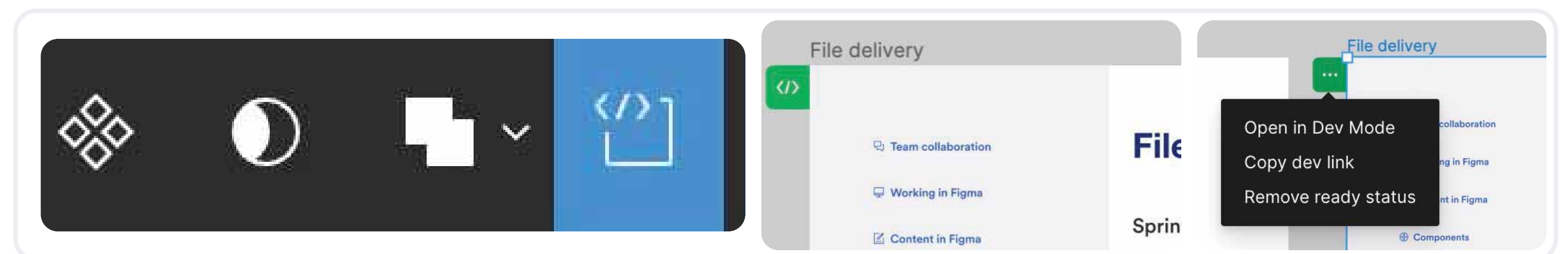
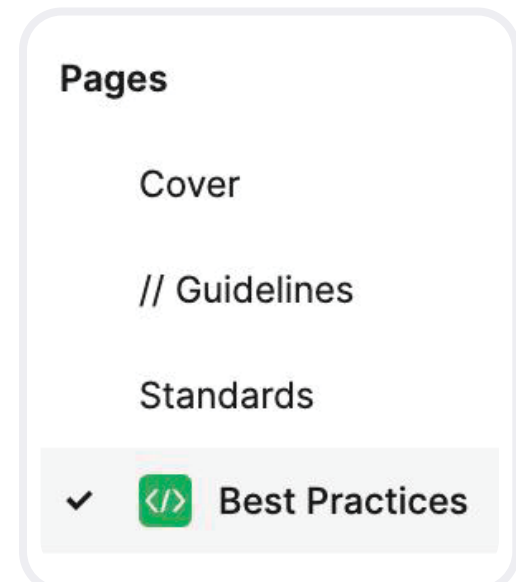
## Sprint handoff - Ready for dev

To make it as easy as possible for developers to find updates they need to make, use the “Ready for dev” feature.

**Step 1: Select the completed component, instance, frame or section.**

**Step 2: Click the “Mark as ready for dev” icon in the toolbar or section heading**

Doing so will highlight the page in the file navigation as well as the element that includes your changes. Your developers will be able to select the dev indicator, open the work in dev mode and compare the changes.



When using this feature, be sure that your screens are ready for AEM authoring, dev and CAT (if applicable).

[Learn more about our File delivery standards >](#)

[Continue to Standards](#)



# Revision history

---

| Date       | Screen ID  | Description  | Author     |
|------------|--|--|------------|
| 04/07/2023 | Accessibility  | Added accessibility standards per the A11Y team.   | W McDonald |
| 04/11/2023 | Libraries  | Hiding libraries until there is more info  | W McDonald |
| 04/12/2023 | Variants   | New visuals  | W McDonald |
| 08/02/2023 | Team collab<br>Working in Figma<br>Content in Figma<br>Components<br>Variants<br>File delivery | Team collab - Added FigJam section, jumplink to content components<br>Working in Figma - Added screenshots in first two sections<br>Content in Figma - Net-new frame<br>Components - Added How to access components and How to use components for content management<br>Variants - Added Using variants for dynamic content options<br>File delivery - Net-new frame | W McDonald |

Select the columns you need from the design panel before detaching.

Press Command + D to add a row