Avoiding Text Truncation

In this module, you'll learn:

- What text truncation is.
- Why it's important to avoid text truncation.
- How to avoid text truncation.
- Best practices to avoid text truncation.

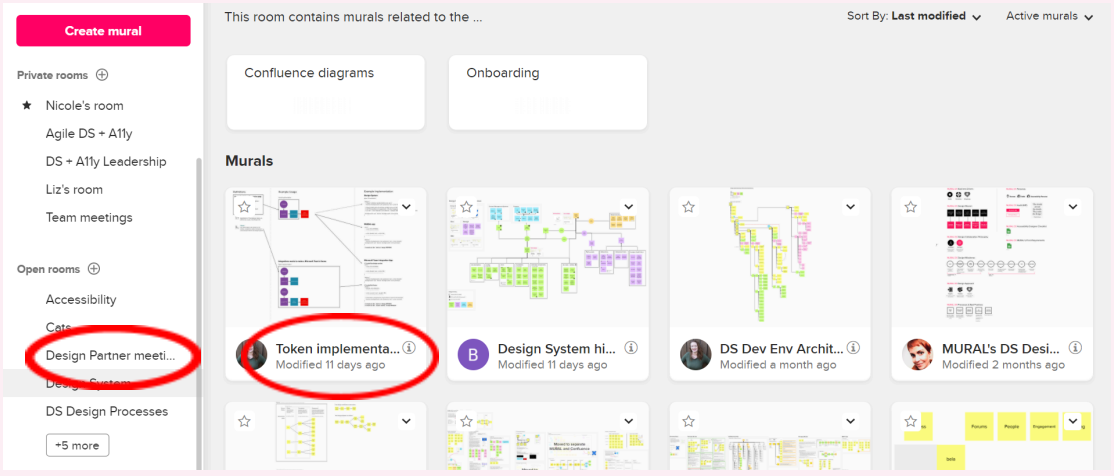
What is text truncation?

Text truncation is when text is shortened or cut off from view, usually by using an ellipsis, but it can occur in many different and unintended ways. For instance, if an end-user increases font size, text can become crowded within inline items and containers, columns, and horizontal spaces.

Example: Truncated text cuts off letters or words. This is an example of truncated te...

As a designer at MURAL, you may encounter this mainly in areas that require user input, such as names on MURAL cards or rooms.

Example: The screenshot below features an example of truncated text on MURAL cards and on the names of rooms.



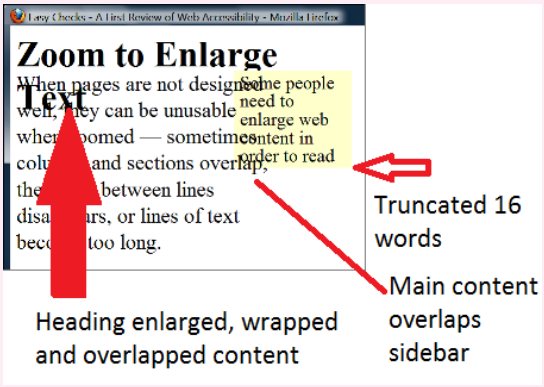
Why is it important to avoid text truncation?

Accessible designs avoid text truncation whenever possible, keeping in mind a user's needs, behavior, and environment. They are built to support devices of different sizes, zoom magnification, and translation to other languages.

When text is truncated, people with disabilities and screen reader, keyboard, or mobile device users may be unable to see, read, or access the content. For example, many screen readers will read a title rather than the text onscreen unless an end-user interacts with the element, in which case the text off-screen may not be recognized.

Many people with disabilities use tools to resize or change the appearance of content to eliminate potential barriers. It's important to consider how these tools will interact with your design.

Example: This W3C example demonstrates how content can become obscured when text overlaps and truncates.



How do I avoid text truncation?

Your job as a designer is to ensure a design follows Web Content Accessibility Guidelines (WCAG) for Resizing Text and other related guidelines listed in the resources section at the end of this module.

A design that follows WCAG prevents text that is zoomed or enlarged from becoming truncated, clipped, or obscured whenever possible. Collaborators should be able to see the full text in some way, whether through keyboard tabbing, expand buttons, or other means.

The best way to avoid text truncation is to create a responsive design with a layout that supports the moderate expansion of text and prevents it from being clipped as the viewport shrinks or expands.

Definition: Responsive designs adapt to a collaborator's behavior and environment, including screen size, platform, and orientation.

Example: Text in a mural should still be visible and avoid truncation when zooming in and out or viewing with a mobile phone.

To avoid text truncation:

- Use effective responsive designs that:
    - Allow end-users to adjust or override text spacing and resizing without truncation, wrapping, clipping, overflowing, or otherwise obscuring the text. They should be able to change the spacing of letters, words, paragraphs, and lines of text without losing content or functionality.
    - Set minimum heights on elements so that they can grow as the text grows. Take into consideration how text and element sizes affect the design to prevent text truncation. For example, fixed headers and footers may consume the whole visible area making the content in the middle inaccessible.
        - When creating hand-off deliverables, use the design system type tokens that have been converted to a relative unit (e.g. rem, em) for developers.
        - For use cases, where you're not using the design system, you can convert your pixel font sizes to a relative unit by dividing the pixel value by the baseline (most commonly 16px, 10 px for “Classic”).
        - For example, if you’re designing a feature that uses 28px fonts, it would be:
            - 28/16 = 1.75rems for the Hana redesign
            - 28/10 - 2.8rems for the Classic look and feel of the product
    - Ensure text containers resize as the viewport size changes so that text does not truncate or overflow.
    - Provide sufficient space between columns of text to prevent text truncation and overflow.

Example: As a MURAL designer, you’re tasked with tweaking a mural menu to display with different font sizes based on user settings that don’t truncate text.

    - Display as much content as possible in an accessible font size while preventing text truncation whenever possible as font size increases (see the Accessible text and typography module for more information on fonts and font sizes).

- Be sure to your design doesn’t break!
  Provide minimum heights, rather than static heights, on the items below. Always include padding or inset spacing on elements with a solid background to ensure they don’t bleed into one another.

  By ensuring the following criteria are met (without losing functionality), you’re also helping developers who don’t have an accessibility background implement these elements according to guidelines:
    - Line height (line spacing): At least 1.5 times the font size.
    - Paragraph spacing: At least 2 times the font size.
    - Letter spacing (tracking): At least 0.12 times the font size.
    - Word spacing: At least 0.16 times the font size.

- Language exception: When the language of the content doesn't use one or more of these text style properties, it needs to only conform to the existing properties.
    - For instance, letter-spacing is not applicable for languages written using logographs instead of letters, such as Chinese. Word spacing and line-height, however, would still be applicable in this case.

Resource: For more information about making creating accessible text, see the module, **"Using Accessible Typography."**

- Test how content is presented on multiple devices because of the differences in font rendering and how text can flow across devices. Remember that displaying all text is always the preferred option to decrease an end-user's level of effort and cognitive load:
- Test text spacing in pages with multiple layouts. When a page contains more than one type of layout (e.g., a column layout and asymmetrical layout), be sure the content still functions and all text can be viewed without truncation, clipping, or overlapping.

Best practices, in a nutshell

Here's a summary of what you can do to follow WCAG standards and avoid text truncation:

- Use effective responsive designs that respond automatically to changes in text size and spacing.
- Ensure end-users can change or customize content to meet WCAG criteria while not losing functionality.
- Test how content is presented on multiple devices because of the differences in font rendering and how text can flow across devices.
- Test text spacing in pages with multiple layouts.

Resources:

- Understanding 1.4.12 Text Spacing.
- C28: Specifying the size of text containers using em units, WCAG 2.1 technique.
- C35: Allowing for text spacing without wrapping, WCAG 2.1 technique.
- C36: Allowing for text spacing override, WCAG 2.1 technique.
- F104: Failure of Success Criterion 1.4.12 due to clipped or overlapped content when text spacing is adjusted, WCAG 2.1 technique.

Test your knowledge

Choose one or more answers to the following questions.

1. As a MURAL designer, you're asked to create a solution to the truncated text in MURAL room names. What's the most accessible solution?
    A. Use an ellipsis.
    B. Allow user text to wrap without truncating, clipping, or overlapping.
    C. Create static line-heights, paragraphs spacing, and word spacing.
    D. Use an accessible tooltip.

2. The guideline for word spacing says the space should allow at least _____ times the font size.
    A. 0.18.
    B. 0.15.
    C. 0.16.

3. The best way to avoid text truncation, clipping, or overlapping is to:
    A. Use a responsive design.
    B. Allow end-users to change text size and spacing.
    C. All of the above.
    D. None of the above.

4. One way to ensure a screen reader user can see the full content in a design is to:
    A. Add a tooltip that appears on mouseover.
    B. Add an ARIA (Accessible Rich Internet Application) label with the full text.
    C. Add text to the tab order and show the tooltip when it receives focus.