**Tap targets are not sized appropriately**

May 2, 2019 • Updated Aug 21, 2019

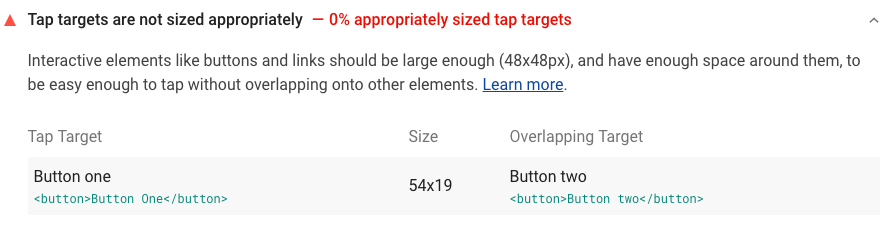
Appears in: [SEO audits](https://web.dev/lighthouse-seo)

Tap targets are the areas of a web page that users on touch devices can interact with. Buttons, links, and form elements all have tap targets.

Many search engines rank pages based on how mobile-friendly they are. Making sure tap targets are big enough and far enough apart from each other makes your page more mobile-friendly and accessible.

**How the Lighthouse tap targets audit fails** [**#**](https://web.dev/tap-targets/?utm_source=lighthouse&utm_medium=devtools#how-the-lighthouse-tap-targets-audit-fails)

[Lighthouse](https://developers.google.com/web/tools/lighthouse/) flags pages with tap targets that are too small or too close together:



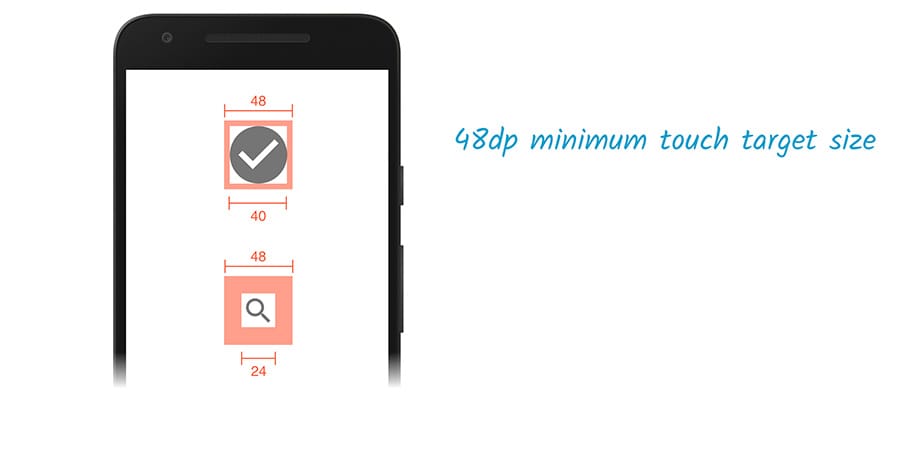
Targets that are smaller than 48 px by 48 px or closer than 8 px apart fail the audit. When the audit fails, Lighthouse lists the results in a table with three columns:

|  |  |
| --- | --- |
| **Tap Target** | The tap target that is inappropriately sized. |
| **Size** | The size of the target's bounding rectangle in pixels. |
| **Overlapping Target** | Which other tap targets, if any, are too close. |

Each SEO audit is weighted equally in the Lighthouse SEO Score, except for the manual [**Structured data is valid**](https://web.dev/structured-data) audit. Learn more in the [Lighthouse Scoring Guide](https://developers.google.com/web/tools/lighthouse/v3/scoring).

**How to fix your tap targets** [**#**](https://web.dev/tap-targets/?utm_source=lighthouse&utm_medium=devtools#how-to-fix-your-tap-targets)

**Step 1:** Increase the size of tap targets that are too small. Tap targets that are 48 px by 48 px never fail the audit. If you have elements that shouldn't *appear* any bigger (for example, icons), try increasing the padding property:

Use padding to make tap targets bigger without changing the appearance of an element.

**Step 2:** Increase the spacing between tap targets that are too close together using properties like margin. There should be at least 8 px between tap targets

## What is a size tap target?

A tap target is any element on a web page that a user interacts with. These include action buttons, links, ads, etc. that a user taps on when accessing a web page using a touchscreen.

The sizing of a tap target is triggered when tap targets like links or buttons are packed closely together or are too small for a user to click on them. Smaller screens pose a greater challenge as the size of tap targets gets scaled down on such devices. To prevent users from getting frustrated, or hitting the wrong button, space out tap targets and make them large enough such that that the users can click on the correct one without their finger pads touching other field forms.

## Why sizing tap targets?

On an average, the size of the finger pad of an adult is approximately 10mm wide. Android recommends a minimum tap target size of about 7 mm for any element. Once the viewport has been configured correctly, you should ensure that the tap targets are at least 48 CSS pixels wide or tall.

**Three things affect how tap targets are sized:**

* The font size of the textual content
* The media queries used on the web page
* The configuration of the viewport

## How to size tap targets

Any tap target the user will access on a touchscreen, including search bars, navigational links, buttons, etc. should meet the minimum size requirement of 7 mm and also have enough space around it when placing other tap targets. There should be no other tap target within at least 5 mm both vertically as well as horizontally of a particular tap target.

A typical example that can be given here is when there is a list (ordered or unordered) in which the tap targets can get too close together on a smaller screen.

* [Text Hover](https://siteauditor.com/technical-seo/size-tap-targets/)

To avoid making it difficult for the user to access them or inadvertently clicking on the wrong tap target, you must define in the CSS style sheet, how the list should be rendered when it is accessed on a small screen. In this case, your CSS will contain a code that essentially conveys that when the screen size is small, a larger line height should be used, whereas when the screen size is large, a standard line height can be used.

When you use buttons on a web page, you will notice that it is already larger as compared to the rest of the textual content on the page.

* [Text Hover](https://siteauditor.com/technical-seo/size-tap-targets/)

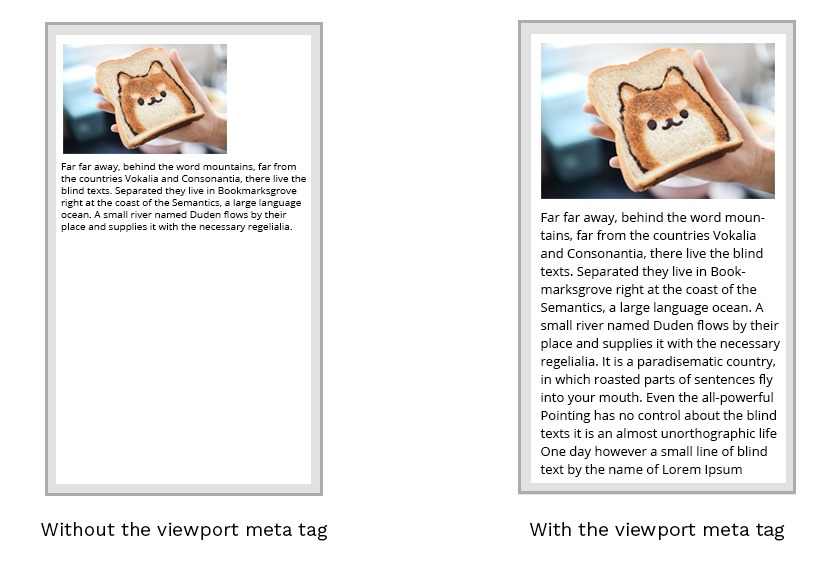
Buttons also need to have enough room around it. The button should be large and visible and stand out from the rest of your text. If on smaller screens, the button scales down to the size of your text, it will defeat the purpose of acting as a call to action button and also make it difficult for users to interact with.

When using adverts, ensure that enough room has been provided with the ad copy.

* [Text Hover](https://siteauditor.com/technical-seo/size-tap-targets/)

People can accidentally click on an ad, even when they don't intend to if enough space has not been provided. This might attract a penalty for you from your Ad provider.

CSS viewport is the zone in a browser window that is visible at once without scrolling. Setting up the CSS [<meta>](https://www.bitdegree.org/learn/html-meta) viewport tag is the most effective way to improve how web pages look on smaller screens.



**The concept of the viewport**

The CSS viewport refers to the part of the website, which is visible in the browser window. Therefore, the **viewport** is usually **not the same size** as the actual page. Mobile phones and other devices with smaller screens display pages in a **virtual window** or **viewport**.

The main issue is that narrower screens make the virtual viewport **shrink** to display all areas of a web page. Therefore, some websites do not look good on mobile devices.

**Manipulating the viewport**

You might assume that setting [media queries](https://www.bitdegree.org/learn/responsive-media) is enough to **fix this issue**. However, they might not help if the virtual viewport does not match the specified **breakpoints**.

Therefore, you should consider setting the **meta viewport tag** for changing the size and scaling viewports.

* width=device-width part of the tag sets the width of the page to **respond** to the width of the screen.
* initial-scale=1.0 part of the tag sets the **initial zoom level** of the page.

This example uses the <meta> viewport tag:

**Example**

<meta name="viewport" content="width=device-width, initial-scale=1.0">

These settings tell the browsers to display a website at the width of the screen of the device used. Therefore, if the screen width is 320px, the browser window will be the same.

**Note:** if your website is not created to be **responsive**, you should not use the viewport <meta> tag since it might create issues.

The CSS <meta> viewport tag accepts the following properties:

|  |  |
| --- | --- |
| **Property** | **Description** |
| height | Sets the height of the virtual device viewport. |
| width | Sets the width of the virtual device viewport. |
| initial-scale | Sets the zoom level for when the page is first opened. |
| minimum-scale | Sets the minimum zoom level that the user can zoom. |
| maximum-scale | Sets the maximum- zoom level that the user can zoom. |
| user-scalable | Sets a flag that lets the device to zoom in and out. |

Mobile browsers manage the **orientation changes** differently. Mobile Safari simply **zooms** pages when the orientation changes to landscape. If you wish to keep settings of scale CSS, include the maximum-scale=1> value to prevent this behavior of mobile browsers.