Content Masking Overview

Content masking hides an element or holds an element on a page until after Monetate's actions complete, which thus allows all site elements to appear at the same time.

> ℹ️ You cannot use content masking with the asynchronous version of the Monetate tag.

Masks are applied before a visitor qualifies for an experience in the initial Monetate payload. They apply the style `visibility: hidden;` to the selected element(s), which keeps the allotted area of the element in the page flow so that the layout doesn't noticeably shift.

When Masks Execute

The ultimate goal of a Monetate action is to blend seamlessly with your native site content. To achieve this goal, when you need to replace native content or hide it entirely, you must use content masking. A mask allows a temporary blank area on the page to show rather than the native site content. After the correct action has fired, Monetate removes the mask and the content served by Monetate appears to the site visitor as though it's loading normally and is native to the site.

Masks must behave differently than other parts of Monetate actions because they must execute immediately upon landing and remain in effect until Monetate serves the content. If a mask doesn't fire immediately, a visitor may see momentarily the native site content that is then replaced with new Monetate-delivered content. This situation is known as *flicker*, and it negatively impacts the customer experience. Flicker is highly dependent on the site visitor's load time and the placement of the Monetate tag on your site. You can also reduce flicker by effectively using masks on the correct element selectors and the correct URL paths.

Without Masking Enabled

This animation demonstrates a site loading with a Monetate action but has been slowed significantly for demonstration purposes.

With Masking Used Correctly

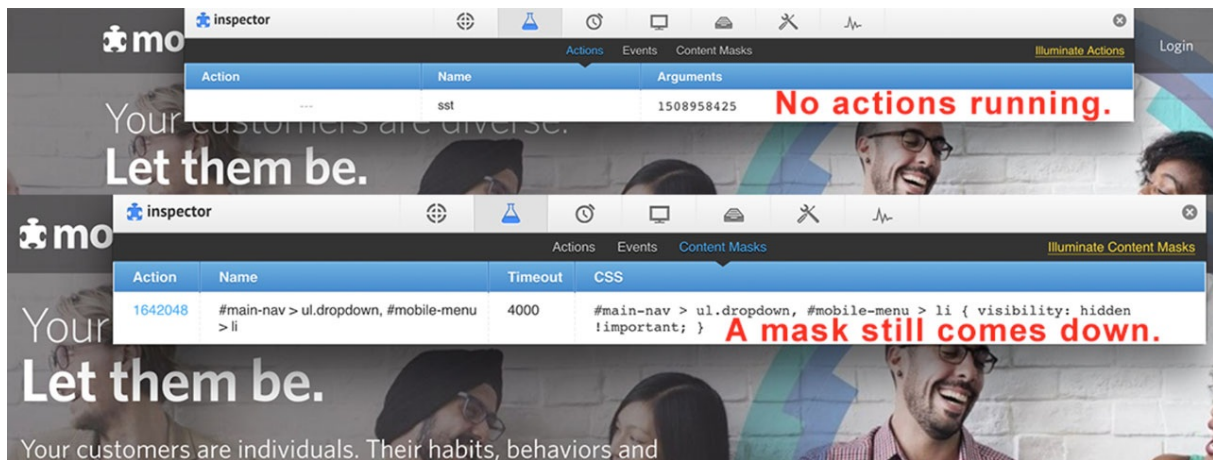This animation demonstrates a site loading with a Monetate action but has been slowed significantly for demonstration purposes.

Native site content rendering with masking enabled on the element we are relplacing

Understanding how masks differ from actions and experiences is important. They're a separate entity and don't adhere to the same set of rules. Masks evaluate only the information provided by the URL and aren't subject to additional conditions that restrict individual actions or whole experiences. They fire independently of other action conditions or who targets. Here's a summary of the rules that masks follow:

- Masks evaluate only the URL.
- Masks aren't restricted by action or experience conditions.
- Masks *always run* if the URL condition matches and the action with the mask is currently active to any audience, even if the site visitor doesn't directly qualify or the action doesn't fire for them.

In this example, a visitor hasn't qualified for any experiences and wasn't served any actions. A mask is still delivered in the Monetate payload.

When to Use Content Masking

There are a few situations when using content masking makes sense:

- If you use the synchronous tag to run Monetate on your site, then you should use content masking on all Hide/Show actions.
- If you use a combination of actions that includes hiding content in one area and then inserting content in another, ensure you apply masks to all the involved areas.
- If the action replaces items on a page, especially above the fold, then you should use content masking.

Consider this example. You use an action that replaces an element in the main navigation bar. In this situation masking the entire navigation bar rather than just the part where the action takes place makes sense so that all navigation items appear to load at the same time.

Best Practices

Keep these best practices in mind when you intend to use content masking on your page:

- Don't mask more than 30% of the visible screen.
- Using content masking may not be necessary for items that appear below the fold.
- Only use content masking when it's necessary to optimize performance on the page.
- You never need to use content masking when using an Insert action.

Applying Content Masks

Enable content masking on the **Mask** tab of Action Builder when building an action by toggling **Content Masking** to **ON**. Next, choose the element that you want to mask.

> **CSS Selector of Element to Mask** is automatically populated with the element or element container that you selected using the DOM selector tool after you chose the type of action you want to build.

The element selector doesn't allow CSS3 selectors, Sizzle selectors (such as `\#menu \> li.nav:eq(2)`), or pseudo-class selectors (such as `\#menu \> li.nav:nth-child(3)`). If you use one of these, a warning appears under **CSS Selector of Element to Mask**.

You must alter the text in the CSS selector before you apply a mask.

> ℹ️ If you're not comfortable editing the text yourself or if you have questions about what selectors to use, submit a support ticket using the Monetate Technical Support portal (support.monetate.com).

Finally, input the path or URL for where to apply the mask. Monetate can match the entire URL, with or without parameters, or the URL paths against a regular expression. Limit content masking to pages on which actions occur.

After you create the mask, you may experience an initial delay before it's visible on your site. This only occurs for newly created masks.

> ℹ️ The experience must be active for 30 minutes before the content mask becomes visible to site visitors.

Once the content mask is ready, it prevents all the specified site elements from displaying before the action that affects them. Once the mask fires, it persists for up to 4 seconds. If the action takes longer than 4 seconds to load, then the mask disappears and the page loads normally. This measure ensures that your visitor's experience isn't negatively impacted by Monetate's content masking.

Masks aren't limited to specific pages through WHO targets and action conditions because they have to run as soon as possible before any kind of data collection or user information is known on the page.

Masking and URLs

Masks are entirely dependent on the URL. To understand how to properly instruct Action Builder to apply a mask, you must understand the URL or path that determines when to apply a mask.

> ℹ️ To check the full URL of the page you are on when in Action Builder, enter the following code in your browser's JavaScript console:
>
> document.URL
>
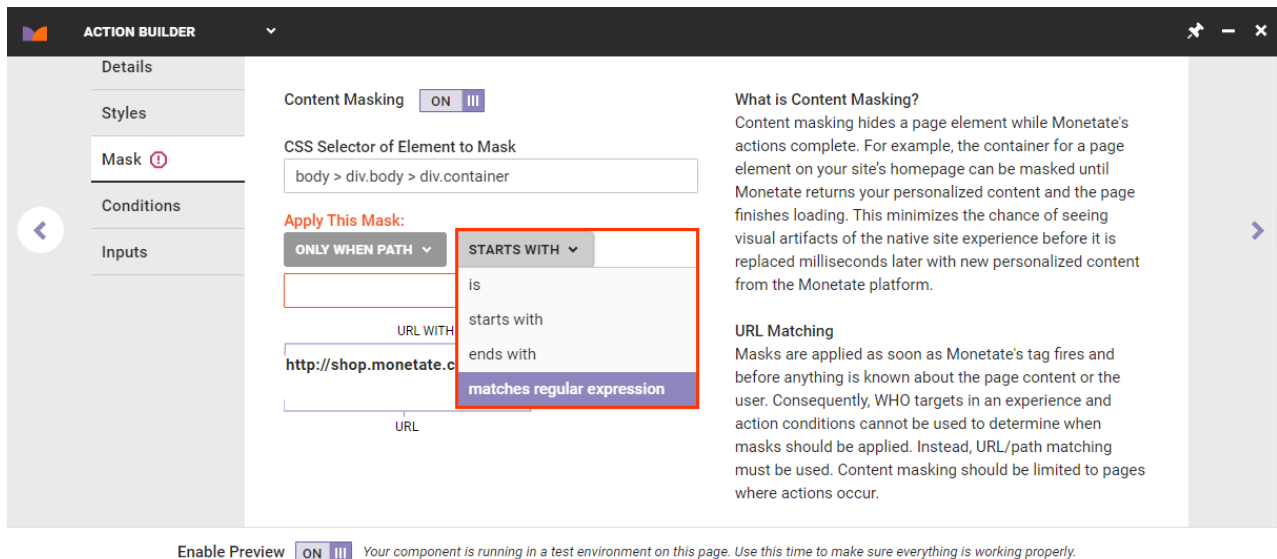> If you want to return the path name, use the following code:
>
> document.location.pathname

This table shows five examples of masking rules applied at different times based on their paths and URLs.

Example	Mask Target
Apply This Mask: ONLY WHEN PATH ⌄ STARTS WITH ⌄ `/`	Targets every page on your site
Apply This Mask: ONLY WHEN PATH ⌄ STARTS WITH ⌄ `/shop/`	Targets every page in the /shop/ directory and subdirectories
Apply This Mask: ONLY WHEN PATH ⌄ MATCHES REGULAR EXPRESSION ⌄ `(.*)html$`	Targets only pages that end with .html
Apply This Mask: ONLY WHEN URL ⌄ IS ⌄ `https://shop.example.com`	Targets the exact URL provided, with any parameters present excluded
Apply This Mask: ONLY WHEN URL (W/ PARAMS) ⌄ IS ⌄ `https://shop.example.com/?_cm=email`	Targets the exact URL provided, with any parameters present included

Regular Expressions and Content Masking

In some cases the mask fires on more or fewer pages than you may expect. To most effectively use masking, you can leverage regular expressions to match patterns in your site URLs with the **matches regular expression** option rather than the **is**, **starts with**, or the **ends with** options. You can ensure masks only fire on the appropriate pages rather than all pages by matching a portion of a URL path.

Regular expressions allow you to create versatile and flexible content masks because in some cases, depending on the structure of the URL, they allow you to create targeting akin to a `pagetype` or `category` with a pattern to match the structure and content provided in the URL.

Each character in a regular expression is either understood to be a metacharacter with special meaning or a regular character with literal meaning. Together, you can use them to identify textual material of a given pattern.

For example, you could use the regular expression `\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,6}\b` to search for any email address.

- `\b[A-Z0-9._%+-]+` matches any letters `A-Z`, numbers `0-9`, or special characters `._%+-`
- `+` symbolizes at least one of the previous content
- You can expect to find `@` in the address and a domain consisting of any letters `A-Z`, numbers `0-9`, or special characters `._%+-` followed by `.`
- The generic top-level domain (gTLD), such as .com, .net, or .org, which is matched by at least two and no more than six `{2,6}` of the preceding letters `A-Z`

With regard to content masking, you can write regular expressions to identify and match parts of a URL to enable complete control over which pages or URL patterns to look for to thus ensure the masks fire exactly where you need them and not in other places.

Regular Expressions Quick References

The basic references for regular expressions below are a good starting point:

Anchors

Character	Use	Example
^	Start of string or the start of line in multi-line pattern	`^abc` matches `abc`, `abcdefg`, `abc123`, etc.

Character	Use	Example
$	End of string or the end of line in multi-line pattern	abc$ matches abc, endsinabc, 123abc, etc.

Quantifiers

Character	Use	Example
*	0 or more of the preceding thing	ab*c matches ac, abc, abbc, abbbc, etc.
{3}	Exactly 3 of the preceding thing	ab{3}c matches abbbc
+	Or more of the preceding thing	ab+c matches abc, abbc, abbbc, etc.
{3,}	3 or more of the preceding thing	ab{3}c matches abbbc, abbbbc, abbbbbc, etc.
?	0 or 1 of the preceding thing	an? matches an a followed by zero or one n character
{3,5}	3, 4, or 5 of the preceding thing	a{3,5} matches aaaaa, aaaa or aaa, but not two characters aa or a single character a

Groups and Ranges

> ℹ Ranges are inclusive.

Character	Use	Example
.	Any character except new line (\n) **Note**: To specifically match a period, you must escape the dot by using a slash: \.	Any letter (for example, a) can be matched by the . as well as all numbers (for example, 0) and special characters (for example, %)
(a|b)	a or b	hey|hello matches hey or hello
(...)	Any content within the parentheses is considered in a capture group	(abc){2} matches abcabc
(?:...)	Passive (noncapturing) group	(?:abc){3} matches abcabcabc
[abc]	Range (a or b or c)	[abc] matches b
[^abc]	Not (a or b or c)	[^abc] matches any lowercase character that isn't a, b, or c, so f matches

Character	Use	Example
[^abc]	Not (a or b or c)	[^abc] matches any lowercase character that isn't a, b, or c, so f matches
[a-q]	Lowercase letter from a to q	[a-q] matches all characters alphabetically from a to q, so h matches but r doesn't match
[A-Q]	Uppercase letter from A to Q	[A-Q] matches uppercase characters the same way, so H matches but h doesn't
[0-7]	Digits from 0 to 7	[0-7] matches 6 but doesn't match 8

Viewing Content Masks

Any action in the account that includes content masking is noted with a mask icon and **MASKING** below its description on the Actions list page.

After an experience that includes an action with content masking has been active for at least 30 minutes, you can then view the mask for the action.

> ℹ The default experience preview functionality cannot display content masking.

You should always view the mask within the experience to ensure it's working.