Rule Engine

Brief Overview

The Rule Engine is used in WalkMe to write rules specifying when to take certain actions. It can be used when building WalkMe items to create logic statements that initiate actions or enable features only when the rule condition/s are evaluated as true or false. Effective rules allow you to make WalkMe responsive and contextual for your users.

The Rule Engine is an integral part of WalkMe. Using the Rule Engine, you can create support, training, or promotional content that appears in the right place, at the right time, to the right audience.

The Rule Engine is used to create rules that define many features such as Flow Steps, Segments, and Goals. Rules can be created to check all kinds of conditions such as elements on screen, the URL, the date and time, or even variables in your software.

For example, you may want to only show WalkMe to the Sales department at your company. Using the Segmentation Center, you can target that user group. After creating a segment, you will define it with a rule that checks if a user is part of the Sales team by looking at their variables.

 Digital Adoption Institute

- View the *Rule Engine and Performance Optimization* course in the *Intro to Build Section* of the *Building a Digital Adoption Project* curriculum
- Don't have a DAI account yet? Sign up here

Use Cases

Some other common uses of the Rule Engine include:

- Create an Error Handling Group in a Smart Walk-Thru to add steps that only appears if the user gets an error message
- Create a Start Point that will direct a Smart Walk-Thru to start steps from a certain place depending on user context
- Create a ShoutOut with an Auto Play rule so that it will play automatically during a certain

time frame
- Create a rule to determine if the input is valid for a Validation [SmartTip](#)

How It Works

Think of rules in WalkMe like a board game. When playing a board game we might learn that IF you land on a certain spot on the board THEN you get to jump ahead a few spaces. This basic form of logic can be applied to understand how the Rule Engine works. For example, IF a user starts on a specific URL, THEN a Smart Walk-Thru should start from the third step in the guidance. This action is taken only IF the criteria we specify is true though, meaning the user is on the specified URL.

The Rule Engine allows you to write conditions. WalkMe uses these conditions in many features by determining that if the condition is true, then something else will happen, for example, a Smart Walk-Thru will play a specialized message, a ShoutOut will play automatically, or a goal will be marked as complete.
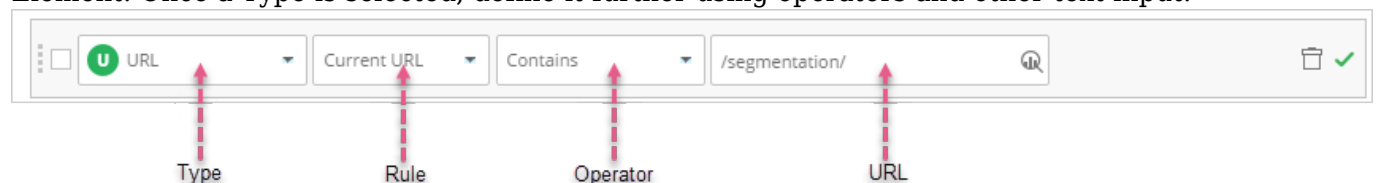
WalkMe checks rules created in the Rule Engine starting with the top statement and moves down the list to each subsequent one. Together, these statements determine if the rule itself is true. If a rule is true, the relevant feature is activated. If a rule is false then the feature is not activated.

The Rule Engine checks the browser window you have open to evaluate each statement, so you must be on the correct page or be logged in as a certain type of user to check if a rule is actually evaluating as true or false. A statement created to check if a user clicks or hovers over an element cannot be evaluated while you have the rule engine open because it's not possible to click or hover over those elements while the rule engine is open.

Creating A Rule

Statement Structure

Each statement is created by first selecting a Type of criteria to check, such as URL or On Screen Element. Once a Type is selected, define it further using operators and other text input.

Rule Types

Rule Types are all the different criteria you can check with a statement. Selecting a Rule Type will then allow you to specify it using additional menus, operators, and input.

Note:

Some Rule Types are only available for specific features.

RULE TYPES	DESCRIPTIONS
On-Screen Element	Checks an element you select on-screen for visibility, content, length, or type
URL	Check either the current URL or referrer URL (the URL from which the user navigated prior to this step)
User Data	Checks cookies in the user's browser, JavaScript variables, or WalkMe Data
Browser Type	Checks the current browser type or window size **Example:** The screen is narrowed, and the launcher covers some element on the screen. In case the user wants that element to take preference and hide the launcher, the following display condition can be created: • Browser > Window Size > Width > More Than > XXX
Date	Checks dates such as day of the week, month, or a specific date. If item is set to auto play until a certain date, it will expire at the end of that day.
Time	Checks specific time of day on the end user's operating system (format HH:MM)
jQuery Element	Checks an element as identified by a jQuery selector. jQuery elements are used when the element is difficult to identify or to optimize performance.
User Activity	Checks whether Goals for Onboarding Tasks have been achieved.

WalkMe Language	Checks the selected language of WalkMe, such as English, French, Spanish, etc. (Note: only when Multi-Language is enabled)
Current Flow	Checks if an iframe, frameset or popup window appears
Salesforce	Checks either the Tab or URL within Salesforce (this is enabled when sfVars is added as a feature)
User Attributes	Checks WalkMe's End User Object for data imported using Incoming Integrations or predefined calculated End User Metrics such as First Seen, Last Seen, Number of Sessions, etc. Learn more.
Platform	Checks the platform WalkMe is running on – Windows / Mac / Web (Only available for the "Workstation" system type)
Segment	Checks if a given segment is met or not, allowing the creation of complex conditions reusing existing segments. When entering a condition referencing a Segment that was already deleted an indication will be provided to the user, and the condition will not reference any Segment.

On-Screen Element Options

When On-Screen Element is selected as a Rule Type, three menu options appear allowing you to refine, change, or view the element you have selected.

- Re-select Element – Change the element for your rule
- Gear – Modify the precision settings for the element – read more about Precision
- View Element – See a screenshot of the selected element
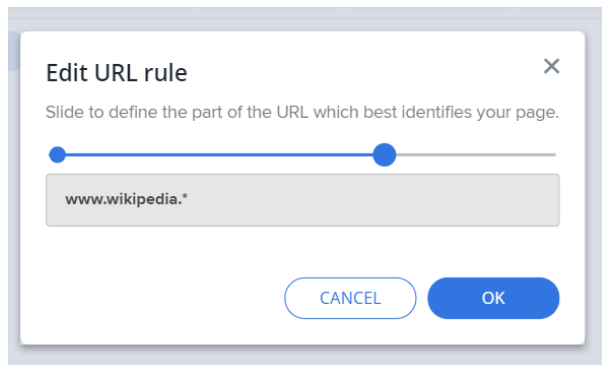
Selecting Operators

Operators set specific criteria for your selected Rule Type. Following is a complete list of all Operators.

Is: The value will be checked as identical, the exact full URL needs to be put in

Contain: The value exists somewhere in the browser URL, in any part of it

Like: The value can be a part of the URL, but its location has to be specified using an asterisk: e.g., */account; more flexible than Matches

Matches: Combination between Is and Like; full URL can be used as a value, or a part of the URL, cut with our tool

Note:

Only specific operators will appear, depending on the element type you selected e.g. The screen element button will not contain the operator *'Length is Equal to'* but it **will** contain the operator 'Clicked'

Category	Operator	Description

Visibility	Is/Is Not Visible:	Checks if the element is visible on the screen
	Exists/Does Not Exist	Checks if the element is or isn't in the HTML code (regardless of its visibility)
	Clicked/Clicked(Sticky)	• Checks if the element was clicked • Clicked(Sticky) means that the element is constantly being searched for* *Clicked(Sticky) must be enabled by request
	Hovered/Hovered(Sticky)	Checks if the element was hovered over *Hovered(Sticky) must be enabled by request
	True/False	Tests if jQuery element is found in the page code
	Element Count	Checks the number of times a jQuery element is identified in the page code

Content	Text is/Is Not	Checks the text for a selected element
	Text Is Like/Is Not like	Checks if a selected element contains specific text with the option to use wildcards. Like/Not Like allows the use of *, also known as a wildcard. Wildcard characters are used to substitute for any other character or characters in a string. Not using a Wildcard before or after your specified input, designates that there is not text either before or after it.
	Contains/Does Not Contain	Checks if the selected element contains specific text or numbers. Ignores any text before or after the input specified. Types include UR, User Data, On-Screen Element
	Is/Is Not	Checks for an exact match for numbers and text elements.
	Text is Empty/Not Empty	Checks if a field is empty or has input in it.
	Was Reached/Was Not Reached	Checks whether Onboarding Tasks and Goals have been reached
	Value is Greater Than/Less Than	Checks if the number is greater or lesser than the input value. Applicable for Types that may include numbers Length
	Length is Equal To/Less Than/Greater Than	Checks the number of characters in a field and compares it to your number value
	Word count is Equal To/Less Than/Greater Than	Checks the number of words used in a field and compares it to your number value

Type	Valid/Invalid Number	Checks if a number has been inputted
	Valid/Invalid Date	Checks if a date has been inputted
	Valid/Invalid Time	Checks if time has been inputted
	Valid/Invalid US Phone Number	Checks if a US phone number is inputted
	Valid/Invalid Email Address	Checks if an email address is inputted
	Is Selected/Is Not Selected	Checks if the element is or is not selected. Is Selected/Is Not Selected is used for radio buttons or checkboxes

Caution

Operators from the "content" group (e.g. "text is", "text is like") should be used with caution (or not at all) when the underlying app is (or will be) in multiple languages.

If a rule is based on text of an element – switching the app to another language may break the build.

Dynamic Conditions

When you are using an on-screen element as a rule type you can create a rule with Dynamic Conditions that will compare elements on the screen.

This can be accomplished by selecting the element you wish to be compared against and creating an attribute for it.

Below is an example where a Popup was created during a Smart Walk-Thru to show that the Total Opportunities is below the Commit.

1. We need to select the element we wish to compare against on-screen.
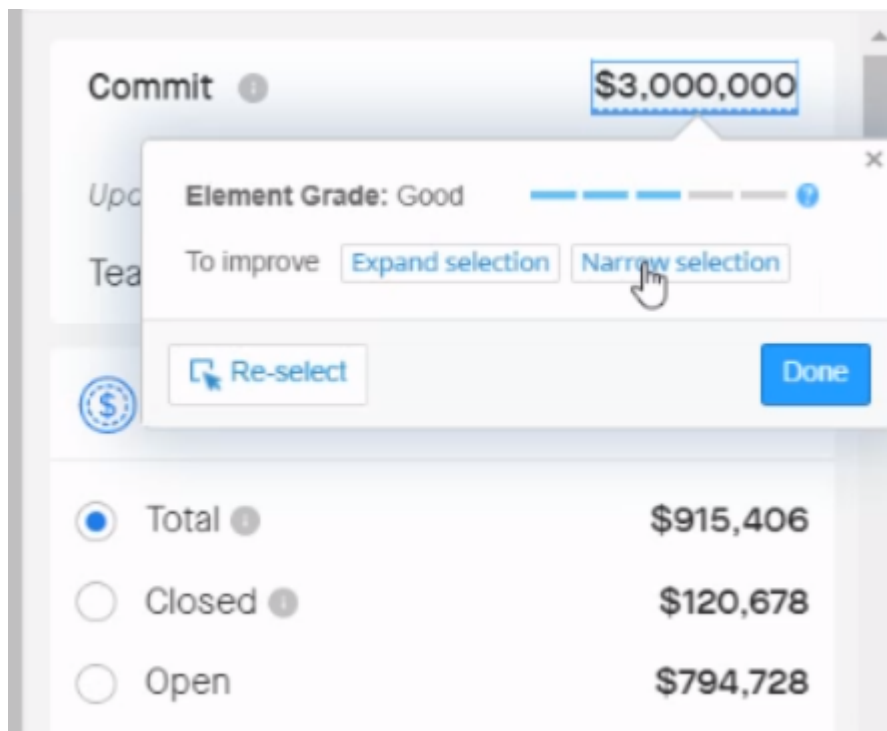
2. Then we create the attribute, in this case, jQuery for the Total Opportunity element.
3. Now we will enter the attribute into the Rule Engine, comparing against the element we have already selected.
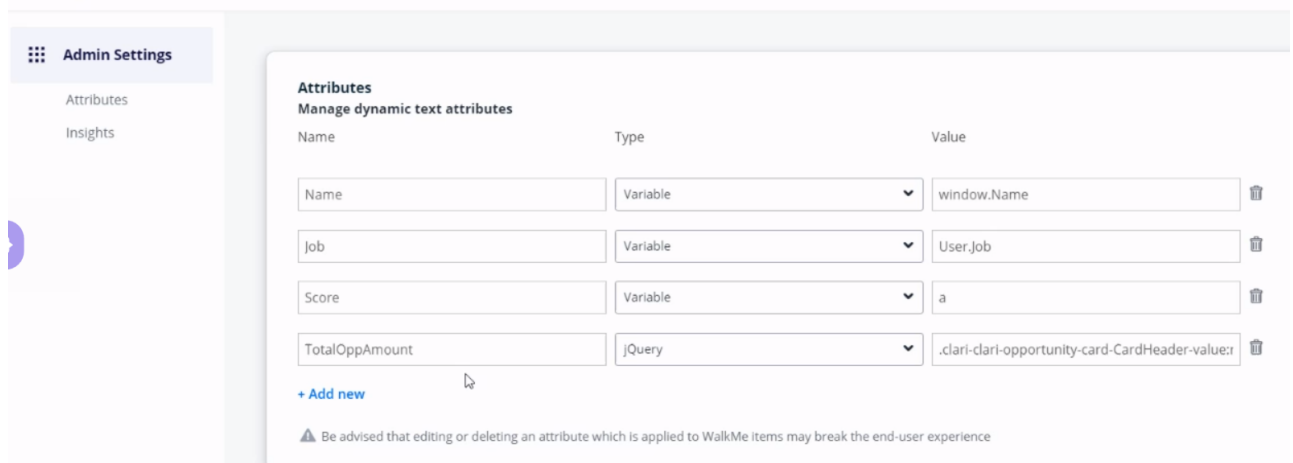
4. Below you can see the Popup we have created.

Creating Rule Input

Some Rule Types require that you add text to complete a statement. For example, using the URL Type requires you to add part of the URL. Some input must be case-sensitive, such as jQuery Element, while another input like URL is not. WalkMe can autofill input for certain Rule Types using the magnifying glass, AKA Current Value Filler. For example, when using the URL rule type use the
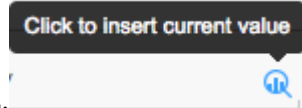
Current Value Filler to autofill the field with your current URL.

Sometimes the text input may be quite long and you can't see all of the entry in the Rule. Simply click into the text field to expand and see full text.

Using Multiple Statements

When creating a Rule you may add multiple statements by clicking the Add button. Using multiple statements allows you to be more specific with your rule.

Use the AND toggle if both statements must be true or use the OR toggle if only one of the
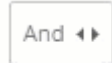
statements must be true.

When Using AND between statements, both statements are checked and must be true. When using OR between statements, either of the rules must be true.

For example, when checking for a Goal we often use multiple statements for a Rule. The first statement checks if the user is on the correct URL AND the second statement checks if the user clicks the submit button. Both statements must be true for the Rule to be true and the Goal to be tallied so we connect this with AND. This allows us to be more specific by verifying that the user didn't click the submit button on another page.

You can also drag and drop Rules into a different order (you will have to first unlink Rules if they are grouped). Simply hover over the left-hand side of the rule, click and move into the desired order. Make sure to re-check your Rules for accuracy. You can add up to 60 Rules in each Rule Engine.

Grouping Statements

Grouping Statements allows you to create complicated rules that check multiple statements at the same time. Similar to placing a parenthesis around a portion of a mathematical statement, creating a group of statements means that the Rule Engine will check **all of these** statements to evaluate if the group as a whole is either True or False.

In the simple example above, the first Statement (URL), is evaluated independently and here is marked as true.

The next two statements are evaluated together: These statements are grouped and connected by an AND relationship. Since one of the statements is false they both evaluate as false.

Since OR is used between the first statement and grouped statement the Rule itself is marked as true because only one of the statements needs to be true for the whole rule to be true.
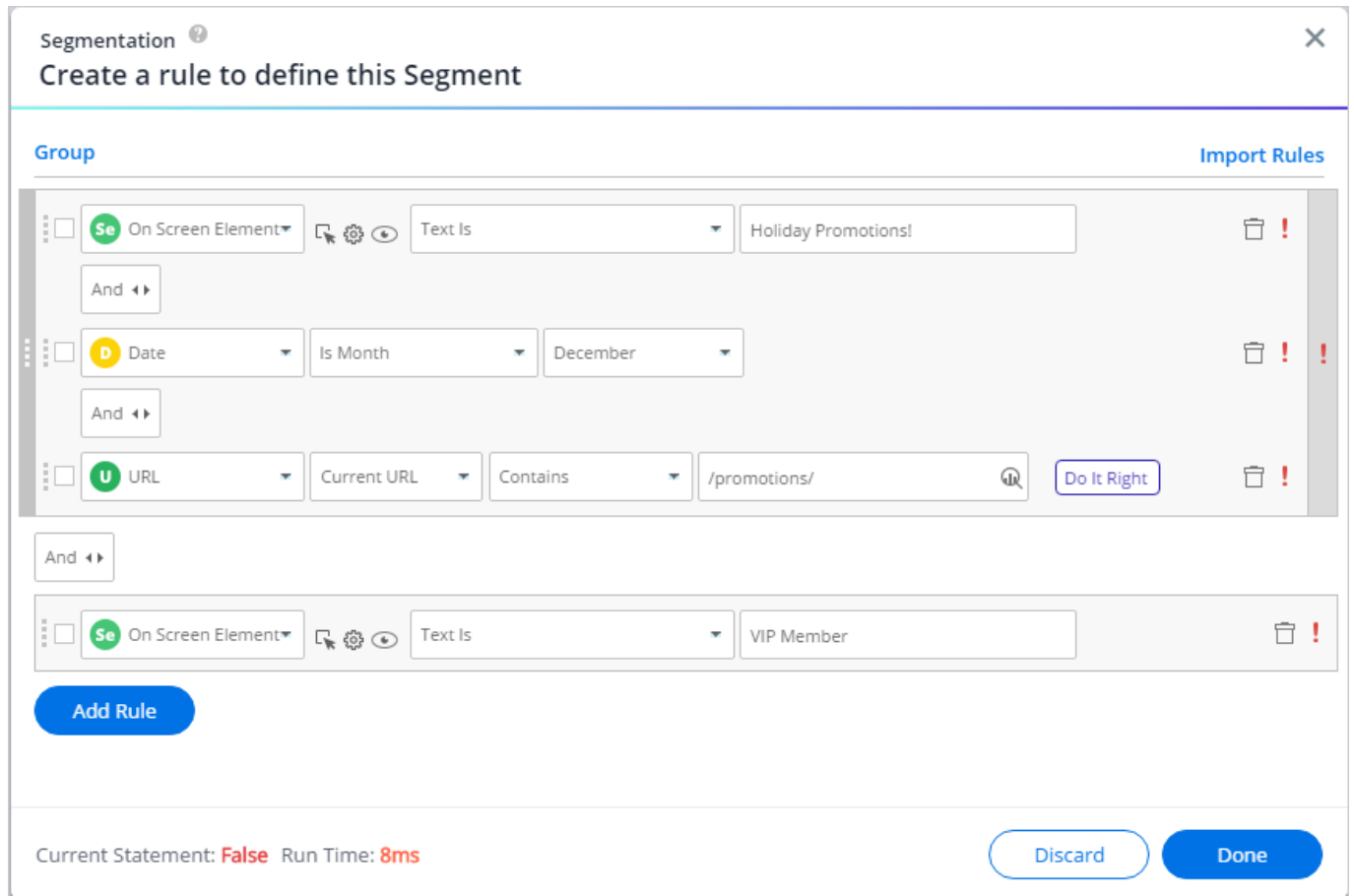
To Group statements, follow these guidelines:

1. Create all of the rules that are intended to be inside the groups, regardless of their hierarchy;

2. Group all of these rules together, regardless of their hierarchy, into a large general group;

3. Group the specific rules inside the container group, as per your needs.

4. Note: It is not possible to include all rules in a single group.

Note:

If you have already created a group to which you would like to add rules, you'll need to first ungroup the already created group, add the rule normally, and regroup the rules with the newly created ones.

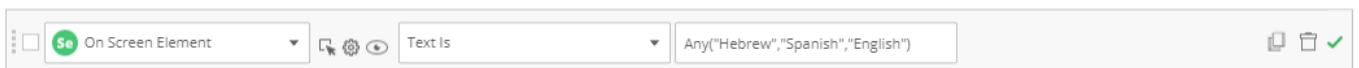Here is an example of a fairly complex grouped segment containing multiple AND statements and an OR statement:

Using Multiple Values

- When creating multiple statements in one rule, you might consider using the multiple value syntax

- To add multiple values, type in – Any("value1″,"value2″,"value3")
  - This syntax must be used as mentioned above, otherwise, it will be evaluated as a single input
  - *Space* can be used between values, but it's not a must
  - Each value should be in "quotation marks"
  - The relationship between the values is "or", so the rule above reads: "On screen element text is either Hebrew or Spanish or English"
  - Currently supported rule types and their operators:

Rule Type	Subtype	
URL	Current URL/Referrer URL	Matches
		Does not match
		Contains
		Does not contain
		Like
		Not Like
		Is
		Is Not
On Screen Element		Text Is
		Text Is not
		Text Is Like
		Text is Not Like
jQuery		Text Is
		Text Is Not
		Text Is Like
		Text is Not Like
User Data	Variable/Cookie/WM Data	Like
		Not Like
		Is
		Is Not

	Worklet/Inbox Item	
Workday	Page/SubPage	Like
		Not Like
		Contains
		Does not contain
		Is
		Is Not
Active Directory		User Member of Group
		User Not a Member of Group
		User is Under OU
		User is Not Under OU

Using this syntax will:

- Improve the performance of large condition trees. By switching from multiple rule lines to one, WalkMe will only need to check for the statement's property once, and then check all of the suggested values of it
- Improve building experience – there is no need to copy the rule/rewrite it. Values can just be added further

Saving a Condition as a Segment

An existing condition can be saved as a Segment, in order to allow easy reuse of the condition in other places.

In order to create a Segment from an existing condition do the following:

1. Open the condition in the Rule Engine
2. Click the **Save as Segment** condition at the bottom of the screen

Save as Segment

Education Week

CANCEL   SAVE

3. Give the new Segment a name
4. The existing condition will be replaced by a single "Segment" condition referencing the newly created Segment with a "Matches" operator

Using Regular Expression as a Condition

Regular Expressions can be used with Rule Types jQuery and On-Screen Element. Regular Expressions (RegEx) allow you to create and use advanced patterns and have full validation flexibility for an input field's value.

Check out the [RegEx tester](#) to see what you can do.

Importing Rules

Importing Rules allows you to easily search for different Apps and choose a Rule you've previously created for Segmentation or Goals.

To import rules, click *Import Rules*, select an App, Then select the item from which you would like to copy the rule.

You can also import start point rules for all logic steps.

Duplicating a Rule

You can duplicate a rule by clicking on the "Clone" icon. This allows you to quickly add multiple rules with similar conditions.

Optimizing Rule Performance

Certain Rule Types are faster for WalkMe to check than others and we recommend that you use these Types whenever possible. The performance difference between Types is generally very small (microseconds), but these can add up when combined with your own site's performance.

We also recommend that you use types that are easy to check for your first statement in a rule. For example, using URL is faster for WalkMe to check than On Screen Element because WalkMe must quickly scan the page to try and identify your selected element, so we first use a statement that checks the URL and then add a second statement that checks for an on screen element.

The following rule types are listed in descending order, from the fastest that WalkMe can check to the slowest:

1. **Light**: URL/Variable/Cookie/User data
2. **Moderate**: On screen element – Visible/Not Visible
3. **Heavy**: jQuery, on screen element – Clicked/Hovered

Performance rules to live by when building

1. Always try to prefer light rules over heavy rules (assuming both are applicable)
2. If you have a heavy rule, always accompany it with a light rule – that way if the light rule is false WalkMe won't even try to evaluate the heavy rule
3. Try not to include too many heavy rules

How Rules Are Checked

WalkMe checks each of statement in order from top to bottom to determine if the specified action or feature should be enabled.

To the right of each statement, WalkMe shows if the statement is True (☑), False (!) or if it is unable to check it currently (?). At the bottom, you'll also see if the entire Rule is True, False or if WalkMe Cannot Assert if it is either.

Limitations

- There's 65,534 character limit for all rules (per Rule Engine screen). A warning message will appear if you exceed the limit and changes will not be saved.
- You can add up to 60 Rules in each Rule Engine

Tip Tuesday Video

To see more Tip Tuesday videos on WalkMe World [click here](#).

Related Resources

- [Segmentation Center](#)
- [Goals and Milestones](#)
- [Onboarding](#)
- [Auto Play](#)
- Looking for information on Current Flow Rule Type? Check out this [Switch to Frames article](#)