Logic Apps in Azure

Azure Logic Apps is a cloud service that helps you schedule, automate, and orchestrate tasks, business processes, and workflows when you need to integrate apps, data, systems, and services across enterprises or organizations. Logic Apps simplifies how you design and build scalable solutions for app integration, data integration, system integration, enterprise application integration (EAI), and business-to-business (B2B) communication, whether in the cloud, on premises, or both.

For example, here are just a few workloads you can automate with logic apps:

- Process and route orders across on-premises systems and cloud services.
- Send email notifications with Office 365 when events happen in various systems, apps, and services.
- Move uploaded files from an SFTP or FTP server to Azure Storage.
- Monitor tweets for a specific subject, analyze the sentiment, and create alerts or tasks for items that need review.

How does Logic Apps work?

→ Every logic app workflow starts with a trigger, which fires when a specific event happens, or when new available data meets specific criteria. Many triggers provided by the connectors in Logic Apps include basic scheduling capabilities so that you can set up how regularly your workloads run. For more complex scheduling or advanced recurrences, you can use a Recurrence trigger as the first step in any workflow.

→ Each time that the trigger fires, the Logic Apps engine creates a logic app instance that runs the actions in the workflow. These actions can also include data conversions and workflow controls, such as conditional statements, switch statements, loops, and branching. For example, this logic app starts with a Dynamics 365 trigger with the built-in criteria "When a record is updated". If the trigger detects an event that matches this criteria, the trigger fires and runs the workflow's actions.

♦ Why use Logic Apps?

→ With businesses moving toward digitization, logic apps help you connect legacy, modern, and cutting-edge systems more easily and quickly by providing prebuilt APIs as Microsoft-managed connectors. That way, you can focus on your apps' business logic and functionality. You don't have to worry about building, hosting, scaling, managing, maintaining, and monitoring your apps. Logic Apps handles these concerns for you. Plus, you pay only for what you use based on a consumption pricing model.

→ In many cases, you won't have to write code. But if you must write some code, you can create code snippets with Azure Functions and run that code on-demand from logic apps. Also, if your logic apps need to interact with events from Azure services, custom apps, or other solutions, you can use Azure Event Grid with your logic apps for event monitoring, routing, and publishing.

→ Logic Apps, Functions, and Event Grid are fully managed by Microsoft Azure, which frees you from worries about building, hosting, scaling, managing, monitoring, and maintaining your solutions. With the capability to create "serverless" apps and solutions, you can just focus on the business logic. These services automatically scale to meet your needs, make integrations faster, and help you build robust cloud apps with minimal code.

Create automated, schedule-based, recurring workflows by using Azure Logic Apps

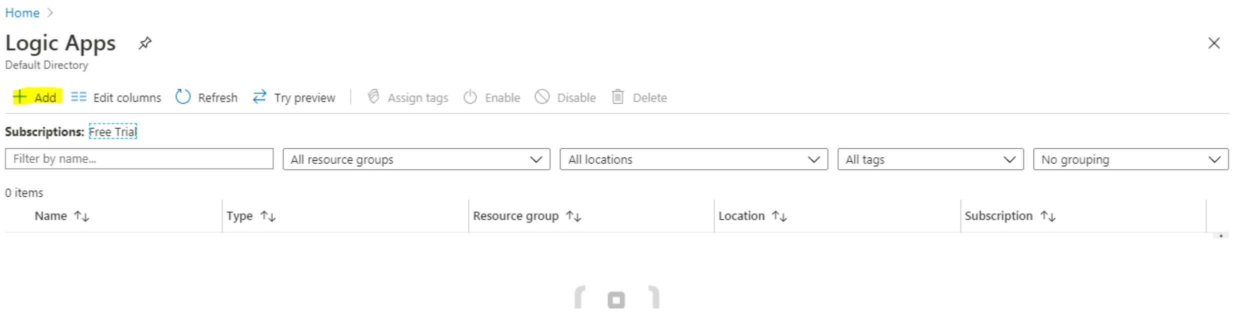
→ This tutorial shows how to build a logic app and automate a recurring workflow that runs on a schedule. Specifically, this example logic app runs every weekday morning and checks the travel time, including traffic, between two places. If the time exceeds a specific limit, the logic app sends email with the travel time and the extra time necessary for your destination.

Sign in to the Azure portal

- Sign in to the Azure Portal with your Azure account credentials.

Create your logic app

- Search the Logic App in Search box in Home Page.
- It will open the Logic App Page as shown below.



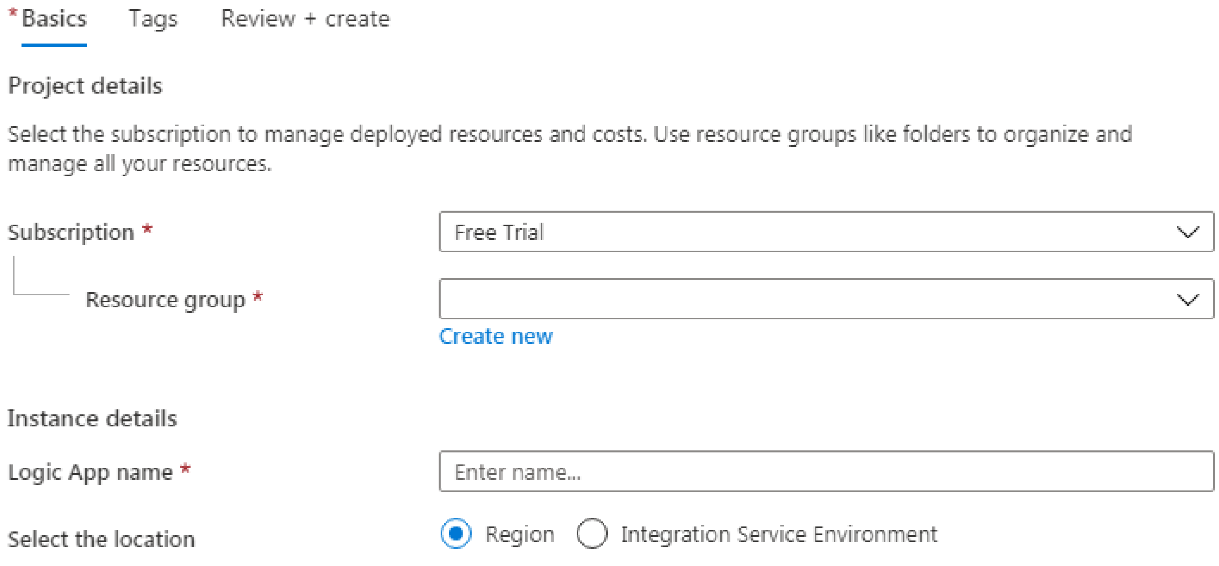
Subscriptions: Free Trial

Filter by name... | All resource groups | All locations | All tags | No grouping

0 items

Name ↑↓	Type ↑↓	Resource group ↑↓	Location ↑↓	Subscription ↑↓

- Now Click on the Add button in the **Logic App** page to create the Logic App.



Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Free Trial
Resource group *	 Create new

Instance details

Logic App name *	Enter name...
Select the location	● Region ○ Integration Service Environment

- Create the New Resource group by click on the Create New link.

Name :TestLB

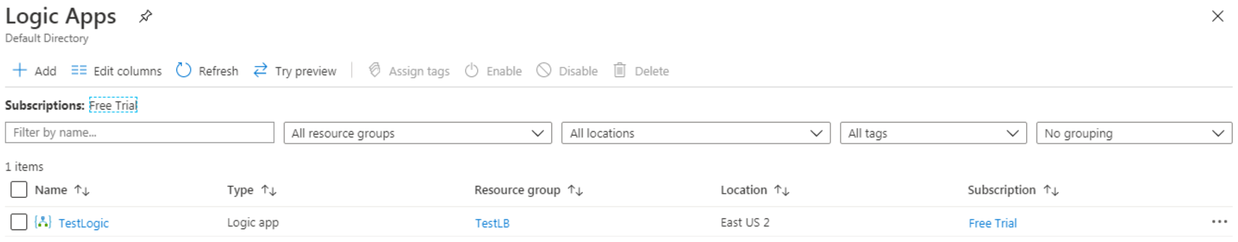
Logic App Name: TestLogic(It should be unique name)

Select the Location: Region

Location: East US2

Log Analytics: Keep as it is

- Then click on **Review + Create** and the **Create**.



Logic Apps

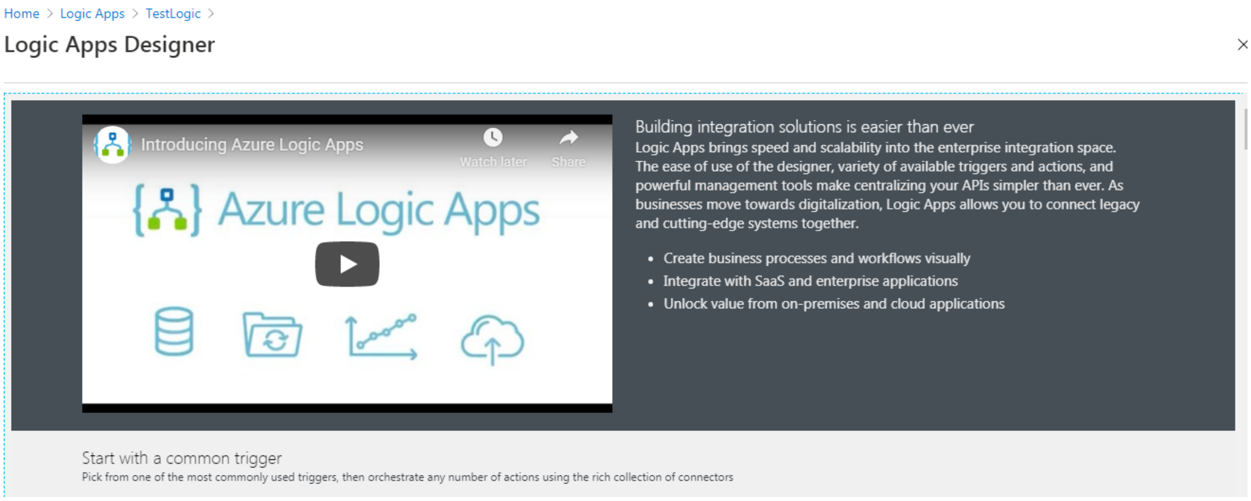
Default Directory

Subscriptions: Free Trial

1 items

Name	Type	Resource group	Location	Subscription
TestLogic	Logic app	TestLB	East US 2	Free Trial

- Your Logic App is created with the name **TestLogic**. Now click on the **TestLogic** it will navigate to **Logic Apps Designer** page.



Home > Logic Apps > TestLogic >

Logic Apps Designer

Building integration solutions is easier than ever

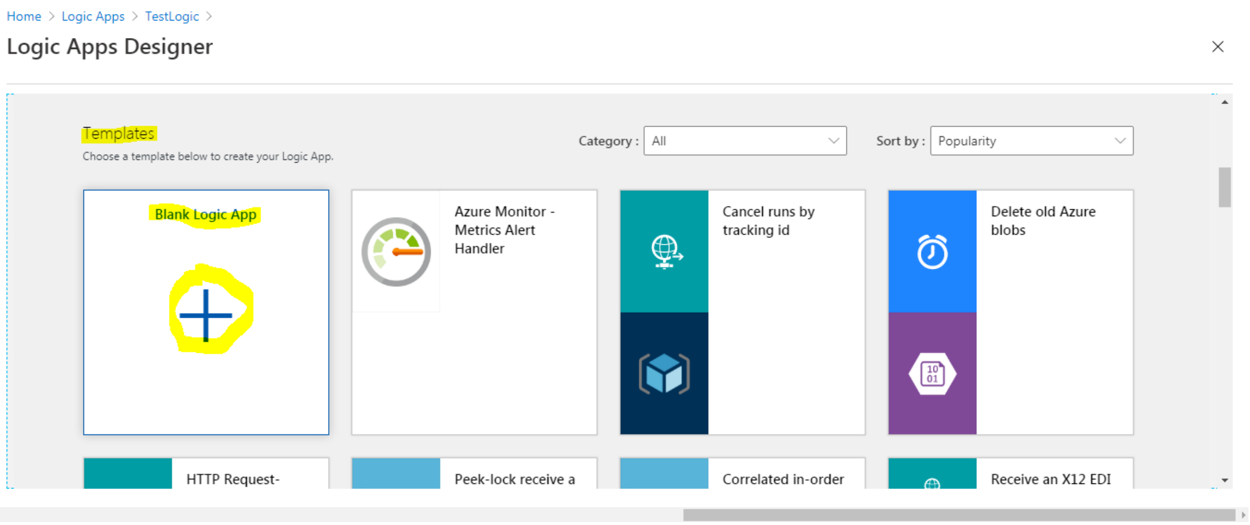
Logic Apps brings speed and scalability into the enterprise integration space. The ease of use of the designer, variety of available triggers and actions, and powerful management tools make centralizing your APIs simpler than ever. As businesses move towards digitalization, Logic Apps allows you to connect legacy and cutting-edge systems together.

- Create business processes and workflows visually
- Integrate with SaaS and enterprise applications
- Unlock value from on-premises and cloud applications

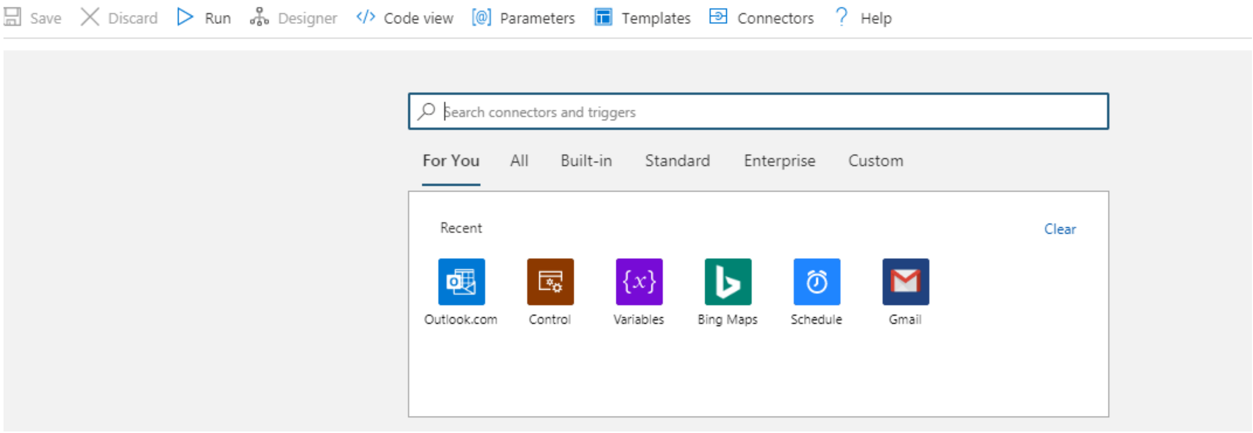
Start with a common trigger

Pick from one of the most commonly used triggers, then orchestrate any number of actions using the rich collection of connectors

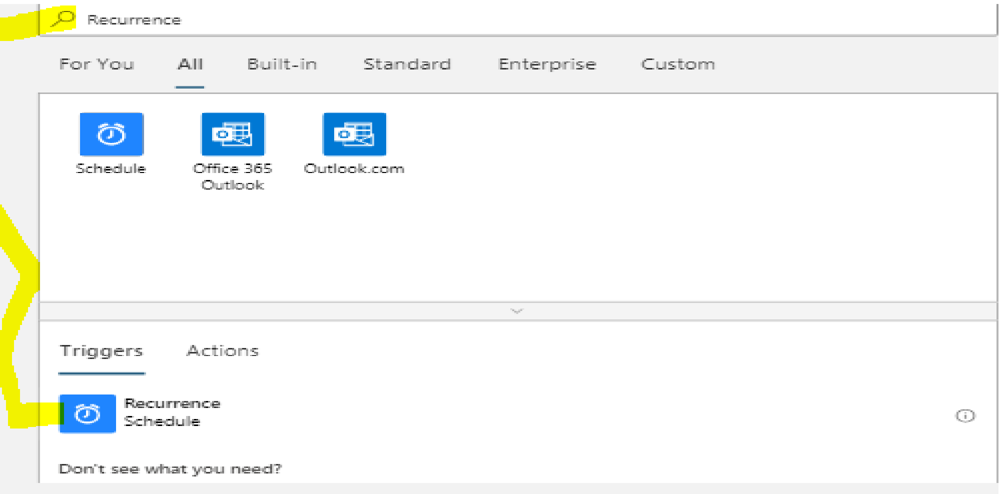
- Now scroll down your page to **Templates** and click on **Blank Logic App** as shown below image.



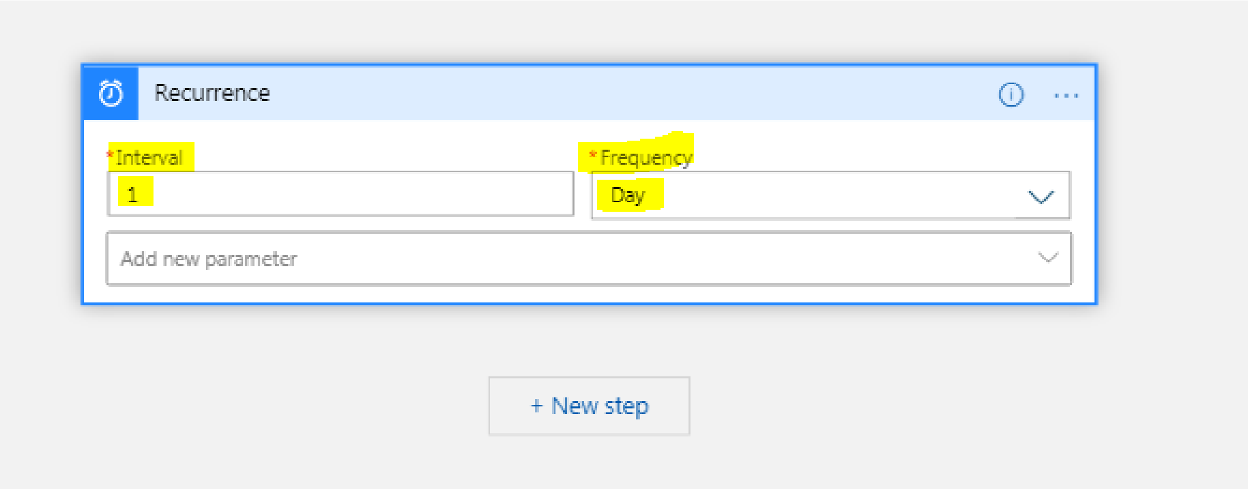
- This is will navigate to another page.



- Now search the **Recurrence** in the seaarch connectors and triggers box.



- Now select the **Recurrence** button under the **Triggers**. It will Open the new page.
- Inside the trigger change this properties.

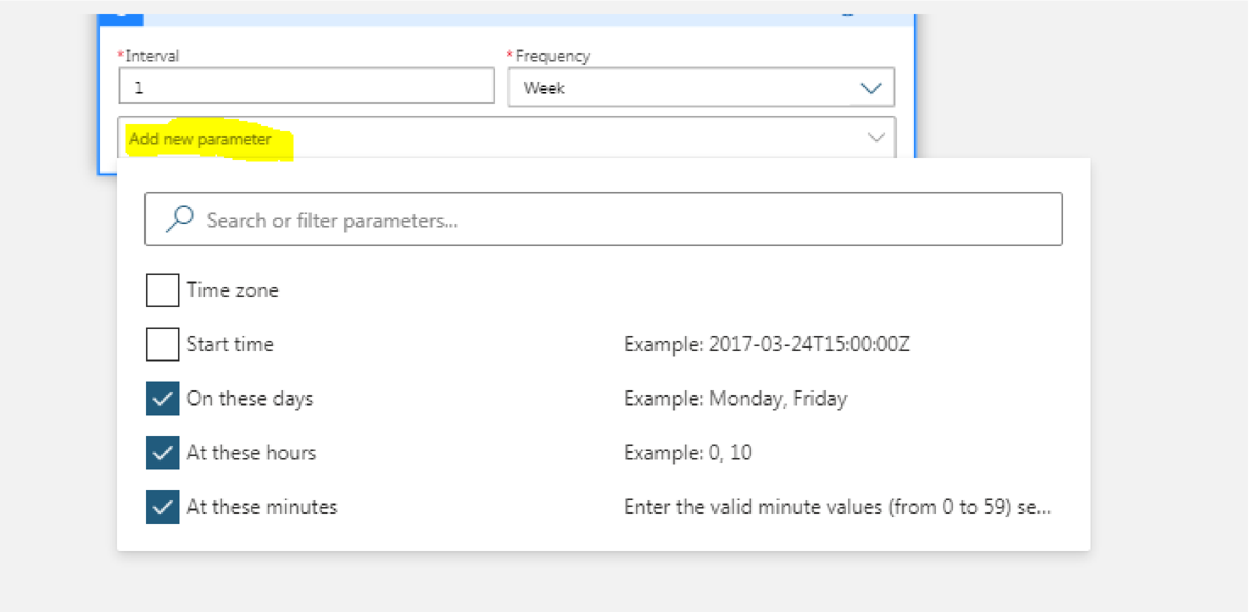


*Interval: 1

*Frequency: Day

Property	Required	Value	Description
Interval	Yes	1	The number of intervals to wait between checks
Frequency	Yes	Day	The unit of time to use for the recurrence

- Under **Interval** and **Frequency**, open the **Add new parameter** list, and select these properties to add to the trigger.
  - **On these days**
  - **At these hours**
  - **At these minutes**



*Interval: 1

*Frequency: Week

Add new parameter

Search or filter parameters...

☐ Time zone

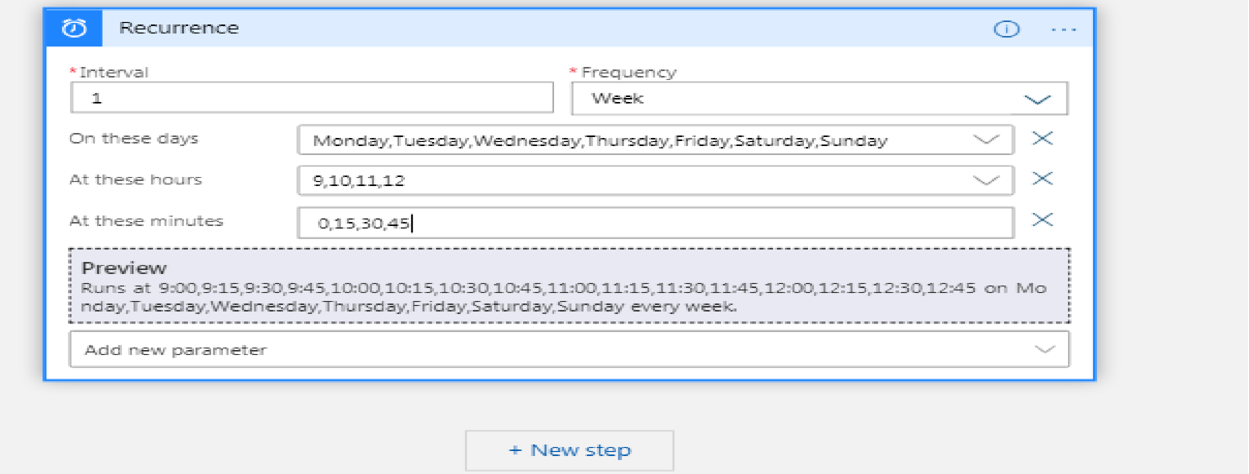
☐ Start time — Example: 2017-03-24T15:00:00Z

☑ On these days — Example: Monday, Friday

☑ At these hours — Example: 0, 10

☑ At these minutes — Enter the valid minute values (from 0 to 59) se...

- Now set the values for additional properties as shown and described here.



Recurrence

*Interval: 1

*Frequency: Week

On these days: Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday

At these hours: 9,10,11,12

At these minutes: 0,15,30,45

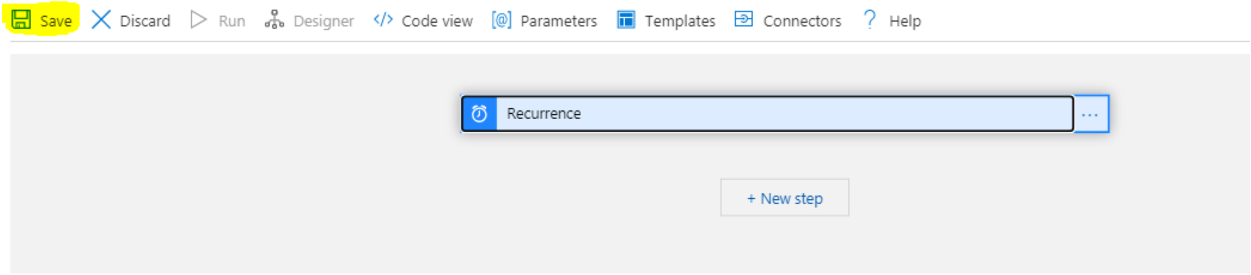
Preview
Runs at 9:00,9:15,9:30,9:45,10:00,10:15,10:30,10:45,11:00,11:15,11:30,11:45,12:00,12:15,12:30,12:45 on Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday every week.

Add new parameter

+ New step

Property	Value	Description
On these days	Monday,Tuesday,Wednesday, Thursday,Friday	Available only when **Frequency** is set to "Week"
At these hours	9,10,11,12	Available only when **Frequency** is set to "Week" or "Day". Select the hours of the day to run this recurrence. This example runs at the 9,10,11 and 12-hour marks.
At these minutes	0,15,30,45	Available only when **Frequency** is set to "Week" or "Day". Select the minutes of the day to run this recurrence. This example runs every 15 minutes starting at the zero-hour mark.

- This trigger fires every weekday, every 15 minutes, starting at 9:00 AM and ending at 12:45 PM.
- Save your logic app. On the designer toolbar, select **Save**.

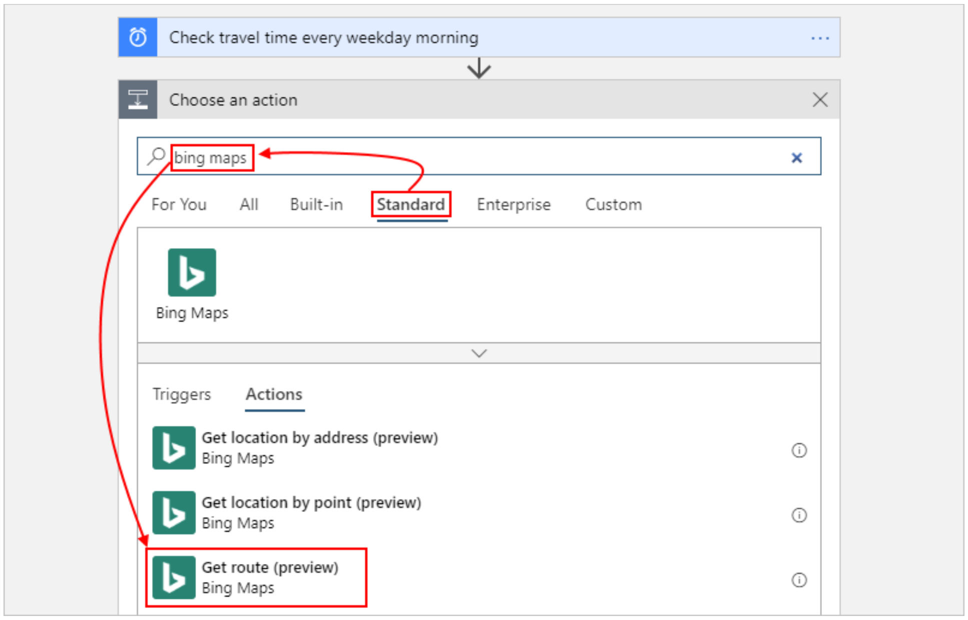


- Your logic app is now live but doesn't do anything other recur. So, add an action that responds when the trigger fires.

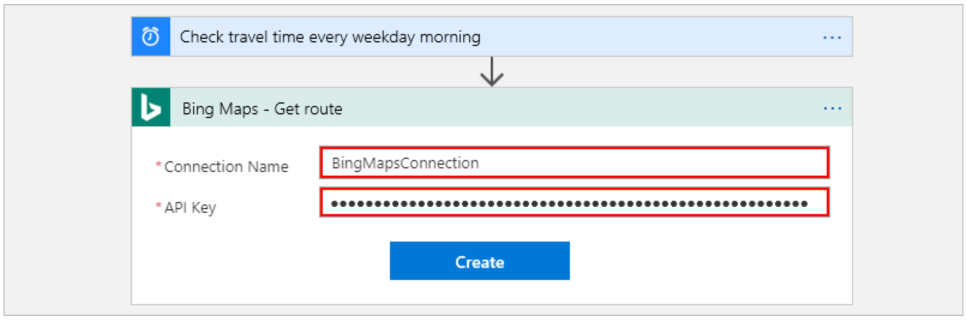
Get the travel time for a route

Now that you have a trigger, add an action that gets the travel time between two places. Logic Apps provides a connector for the Bing Maps API so that you can easily get this information.

- In the Logic App Designer, under your trigger, select **New step**.
- Under **Choose an action**, select **Standard**. In the search box, enter "bing maps" as your filter, and select the **Get route** action.



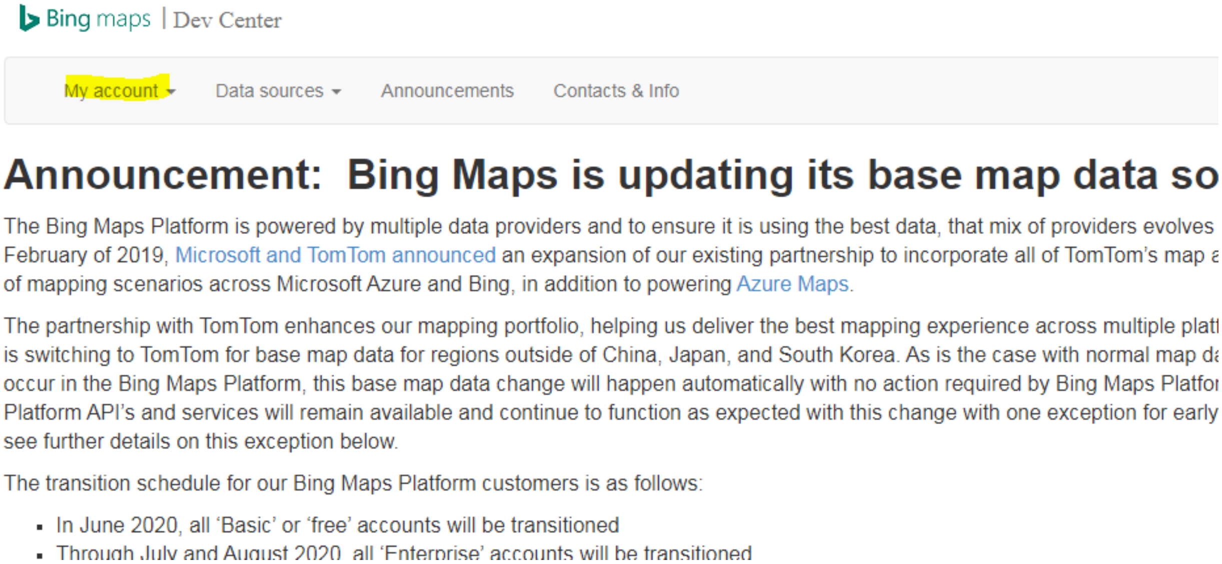
- If you don't have a Bing Maps connection, you're asked to create a connection. Provide these connection details, and select **Create**.



*Connection Name: BingMapsConnection

*API Key: ••

- To get the API key login into the https://www.bingmapsportal.com/
- Login with your Microsoft credentials.
- Now click on the MyAccount and then My Keys.



Announcement: Bing Maps is updating its base map data so

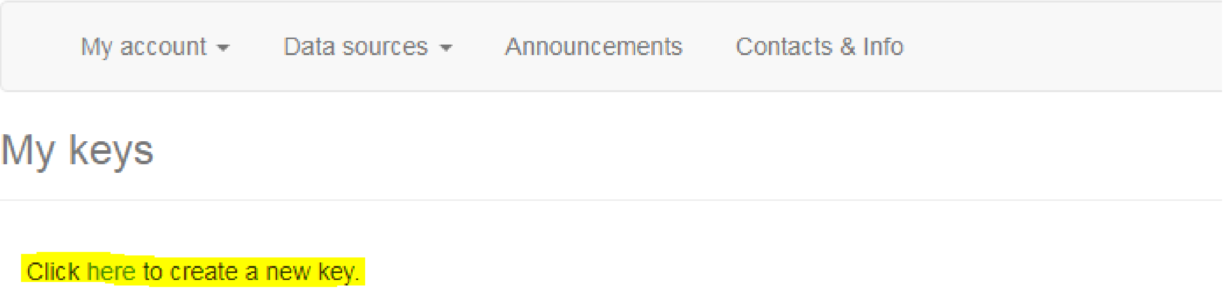
The Bing Maps Platform is powered by multiple data providers and to ensure it is using the best data, that mix of providers evolves February of 2019, Microsoft and TomTom announced an expansion of our existing partnership to incorporate all of TomTom's map a of mapping scenarios across Microsoft Azure and Bing, in addition to powering Azure Maps.

The partnership with TomTom enhances our mapping portfolio, helping us deliver the best mapping experience across multiple platf is switching to TomTom for base map data for regions outside of China, Japan, and South Korea. As is the case with normal map da occur in the Bing Maps Platform, this base map data change will happen automatically with no action required by Bing Maps Platfor Platform API's and services will remain available and continue to function as expected with this change with one exception for early see further details on this exception below.

The transition schedule for our Bing Maps Platform customers is as follows:

- In June 2020, all 'Basic' or 'free' accounts will be transitioned
- Through July and August 2020, all 'Enterprise' accounts will be transitioned

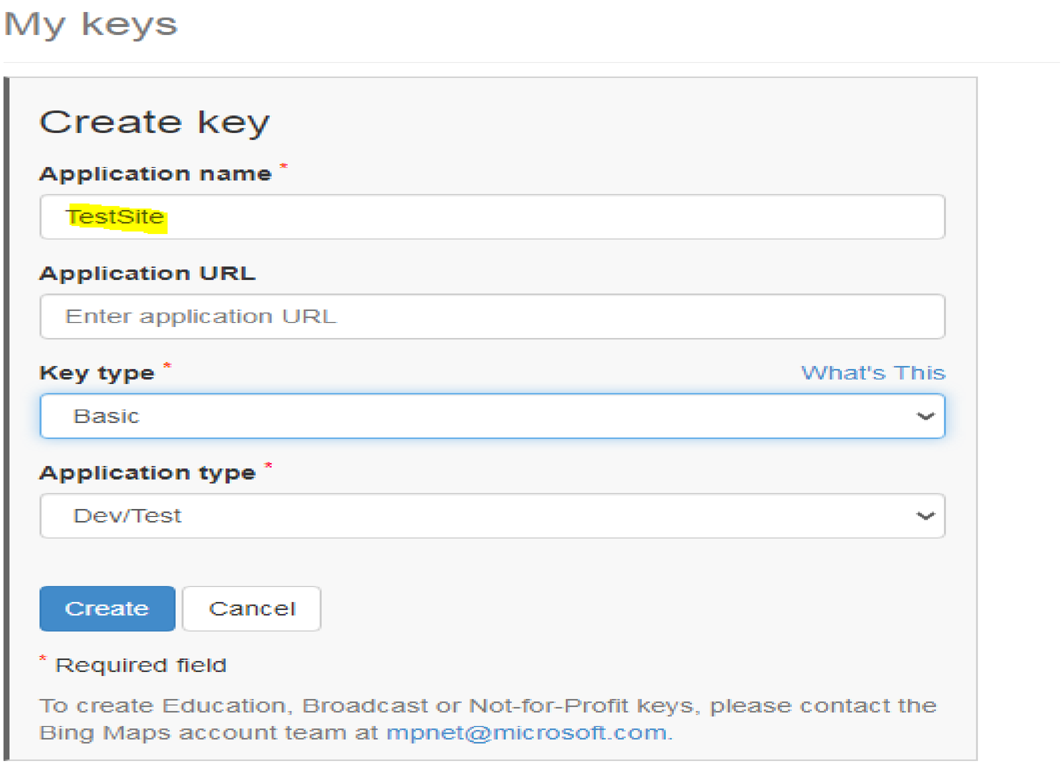
- Now click on the create key link as shown



My keys

Click here to create a new key.

- Fill the details as shown below after that click on create button.



My keys

Create key

Application name * TestSite

Application URL Enter application URL

Key type * Basic

Application type * Dev/Test

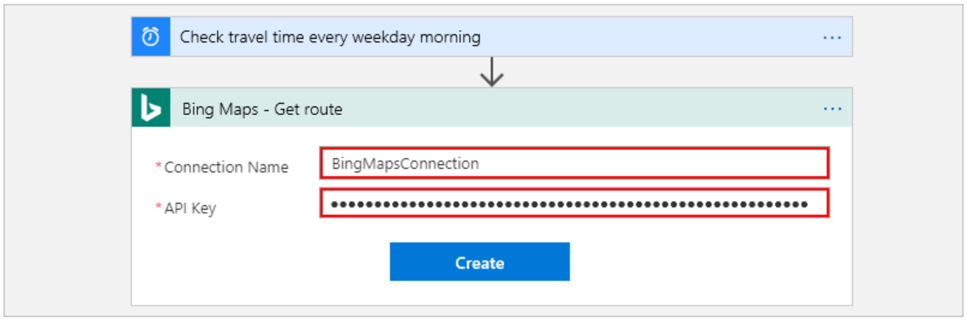
* Required field

To create Education, Broadcast or Not-for-Profit keys, please contact the Bing Maps account team at mpnet@microsoft.com.

- Now click on the Show key button and copy the encrypted key in to notepad.
- Now go to BingMaps action and set the

Connection Name as BingMapsconnection and place the encrypted key in the API key and click on create.

Note: Be careful while coping the API key. If you copy wrongly connection will not be established properly.



Check travel time every weekday morning

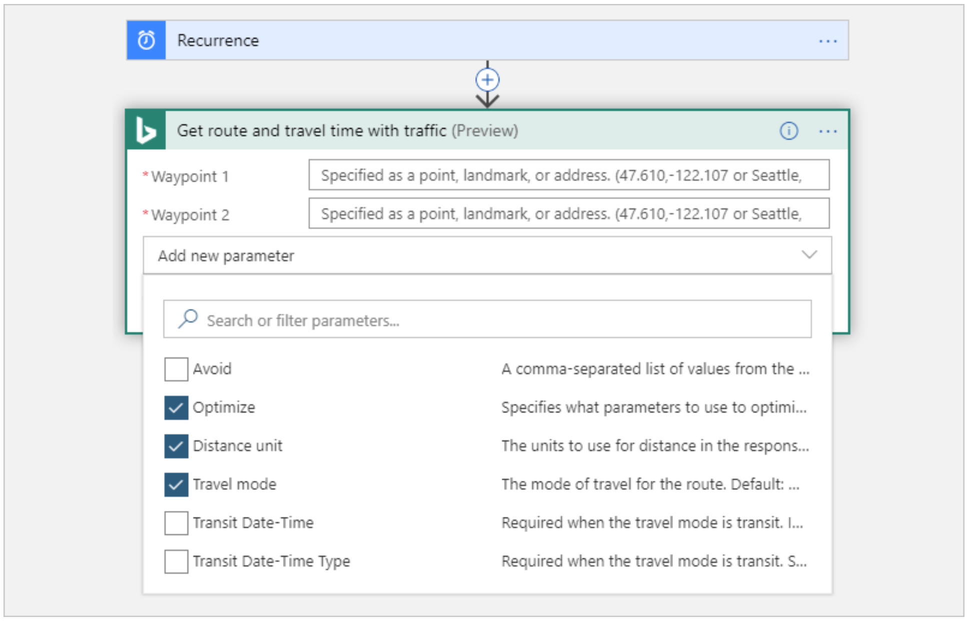
Bing Maps - Get route

*Connection Name BingMapsConnection

*API Key ••

Property	**Required**	**Value**	**Description**
Connection Name	Yes	BingMapsConnection	Provide a name for your connection. This example uses "BingMapsConnection".
API Key	Yes	*<your-Bing-Maps-key>*	Enter the Bing Maps key that you previously received.

- Rename the action with this description: `Get route and travel time with traffic`
- Inside the action, open the **Add new parameter list**, and select these properties to add to the action.
  - **Optimize**
  - **Distance unit**
  - **Travel mode**



Recurrence

Get route and travel time with traffic (Preview)

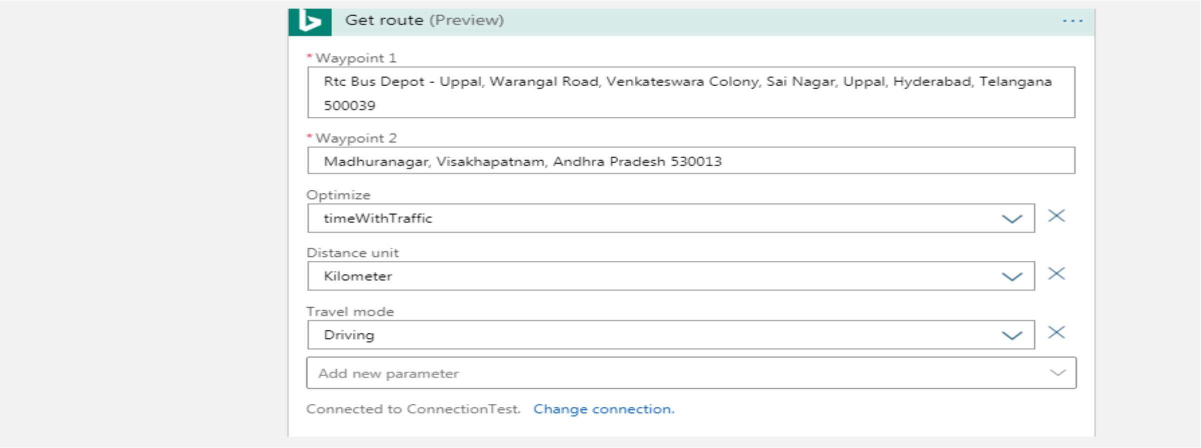
*Waypoint 1	Specified as a point, landmark, or address. (47.610,-122.107 or Seattle,
*Waypoint 2	Specified as a point, landmark, or address. (47.610,-122.107 or Seattle,

Add new parameter

Search or filter parameters...

□ Avoid	A comma-separated list of values from the ...
☑ Optimize	Specifies what parameters to use to optimi...
☑ Distance unit	The units to use for distance in the respons...
☑ Travel mode	The mode of travel for the route. Default: ...
□ Transit Date-Time	Required when the travel mode is transit. I...
□ Transit Date-Time Type	Required when the travel mode is transit. S...

- Now set the values for the action's properties as shown and described here.



Get route (Preview)

* Waypoint 1

Rtc Bus Depot - Uppal, Warangal Road, Venkateswara Colony, Sai Nagar, Uppal, Hyderabad, Telangana 500039

* Waypoint 2

Madhuranagar, Visakhapatnam, Andhra Pradesh 530013

Optimize

timeWithTraffic

Distance unit

Kilometer

Travel mode

Driving

Add new parameter

Connected to ConnectionTest. Change connection.

Property	Required	Value	Description
Waypoint 1	Yes	*<start-location>*	Your route's origin
Waypoint 2	Yes	*<end-location>*	Your route's destination
Optimize	No	timeWithTraffic	A parameter to optimize your route, such as distance, travel time with current traffic, and so on. Select the "timeWithTraffic" parameter.
Distance unit	No	*<your-preference>*	The unit of distance for your route. This example uses "Mile" as the unit.
Travel mode	No	Driving	The travel mode for your route. Select "Driving" mode.

- Save your logic app.

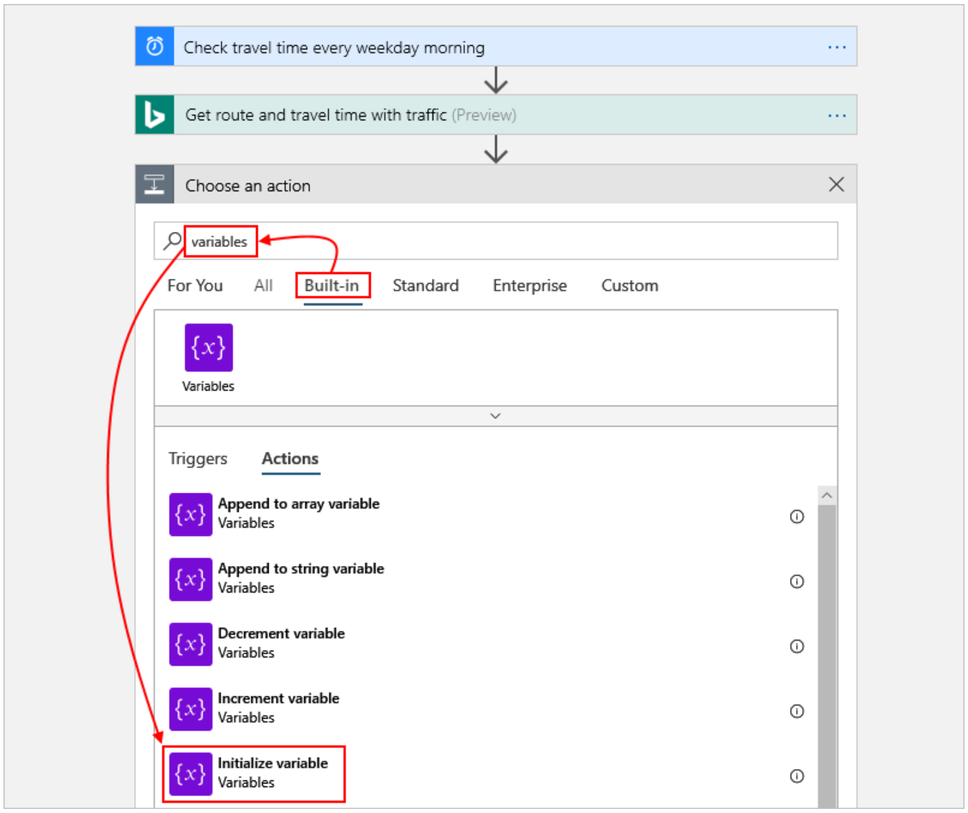
Next, create a variable so that you can convert and store the current travel time as minutes, rather than seconds. That way, you can avoid repeating the conversion and use the value more easily in later steps.

Create a variable to store travel time

Sometimes, you might want to run operations on data in your workflow, and then use the results in later actions. To save these results so that you can easily reuse or reference them, you can create variables to store those results after processing them. You can create variables only at the top level in your logic app.

By default, the previous **Get route** action returns the current travel time with traffic in seconds from the **Travel Duration Traffic** property. By converting and storing this value as minutes instead, you make the value easier to reuse later without converting again.

- Under the **Get route** action, select **New step**.
- Under **Choose an action**, select **Built-in**. In the search box, enter "variables", and select the **Initialize variable** action.

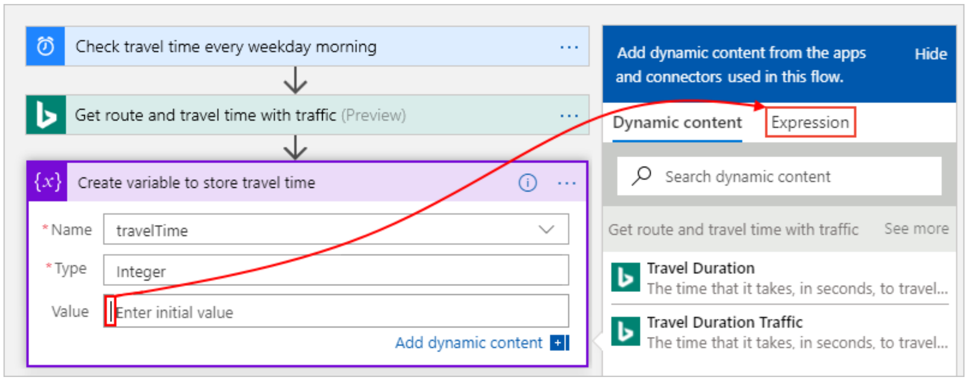


- Rename this action with this description: `Create variable to store travel time`
- Provide the details for your variable as described here:

Property	**Required**	**Value**	**Description**
Name	Yes	travelTime	The name for your variable. This example uses "travelTime".
Type	Yes	Integer	The data type for your variable
Value	No	An expression that converts the current	The initial value for your variable

		travel time from seconds to minutes (see steps under this table).	

a. To create the expression for the **Value** property, click inside the box so that the dynamic content list appears. If necessary, widen your browser until the list appears. In the dynamic content list, select **Expression**.


Check travel time every weekday morning

Get route and travel time with traffic (Preview)

Create variable to store travel time

*Name travelTime

*Type Integer

Value Enter initial value

Add dynamic content from the apps and connectors used in this flow.

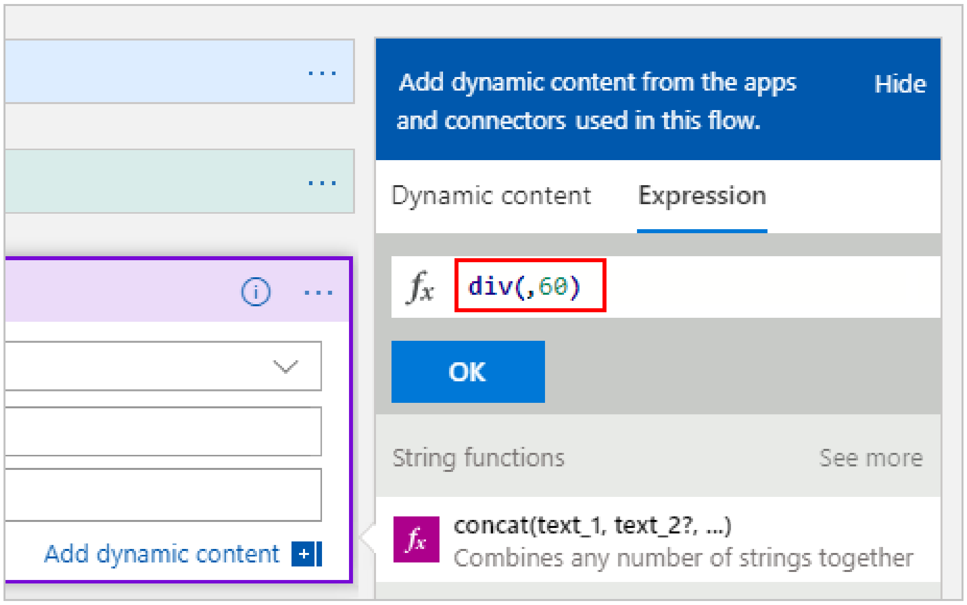
Get route and travel time with traffic

Travel Duration
The time that it takes, in seconds, to travel...

Travel Duration Traffic
The time that it takes, in seconds, to travel...

b. When you click inside some edit boxes, the dynamic content list appears. This list shows any properties from previous actions that you can use as inputs in your workflow. The dynamic content list has an expression editor where you can select functions to run operations. This expression editor appears only in the dynamic content list.

c. In the expression editor, enter this expression: `div(,60)`

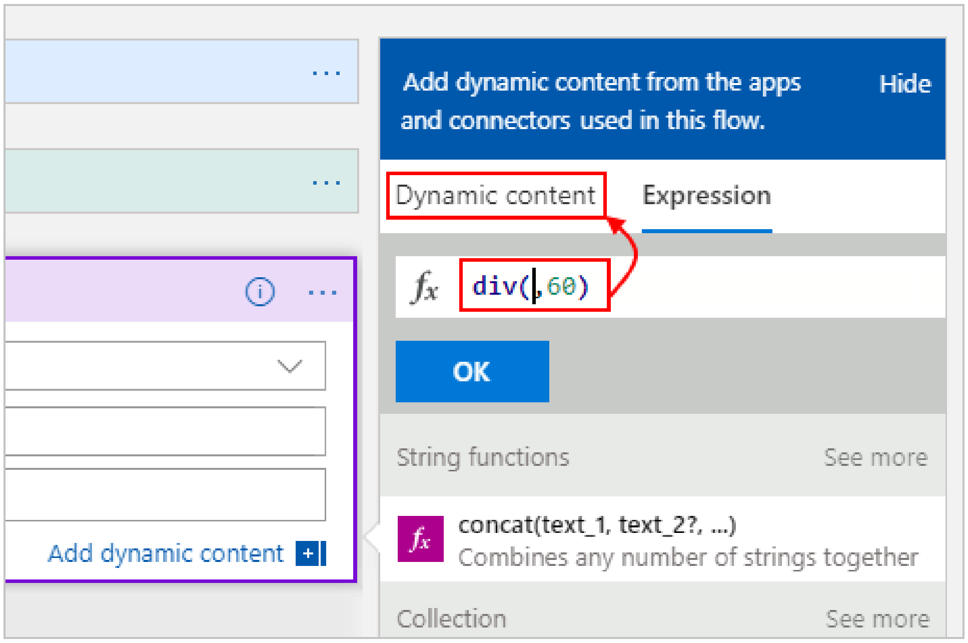

Add dynamic content from the apps and connectors used in this flow.

fx div(,60)

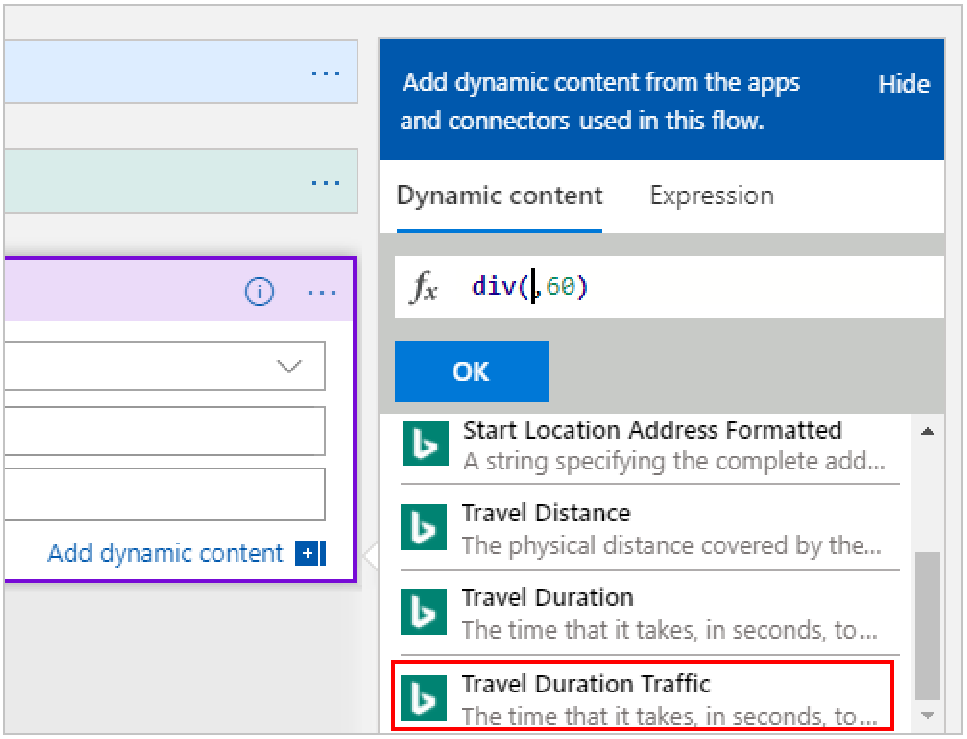
String functions

concat(text_1, text_2?, ...)
Combines any number of strings together

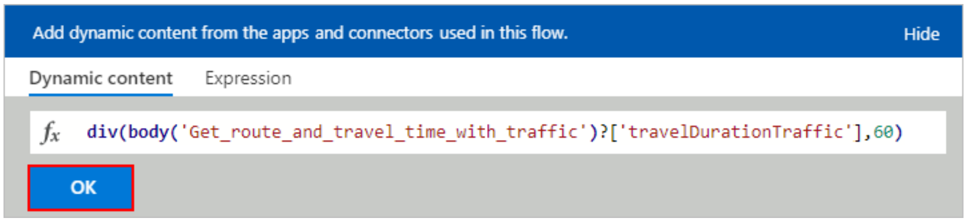
d. Put your cursor inside the expression between the left parenthesis (**(**) and the comma (**,**). select **Dynamic content**.


Add dynamic content from the apps and connectors used in this flow.
Hide
Dynamic content
Expression
div(,60)
OK
String functions
See more
concat(text_1, text_2?, ...)
Combines any number of strings together
Collection
See more
Add dynamic content

e. In the dynamic content list, select **Travel Duration Traffic**.

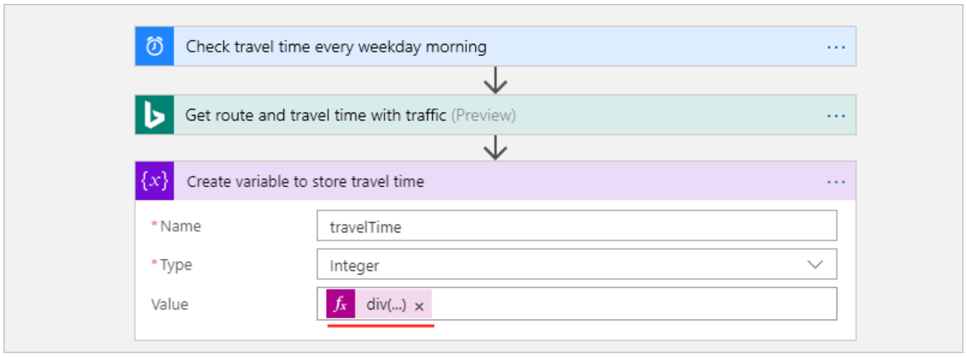

Add dynamic content from the apps and connectors used in this flow.
Hide
Dynamic content
Expression
div(,60)
OK
Start Location Address Formatted
A string specifying the complete add...
Travel Distance
The physical distance covered by the...
Travel Duration
The time that it takes, in seconds, to...
Travel Duration Traffic
The time that it takes, in seconds, to...
Add dynamic content

f. After the property value resolves inside the expression, select **OK**.


Add dynamic content from the apps and connectors used in this flow.
Hide
Dynamic content
Expression
OK

```
div(body('Get_route_and_travel_time_with_traffic')?['travelDurationTraffic'],60)
```

g. The **Value** property now appears as shown here:


Check travel time every weekday morning
Get route and travel time with traffic (Preview)
Create variable to store travel time

*Name: travelTime

*Type: Integer

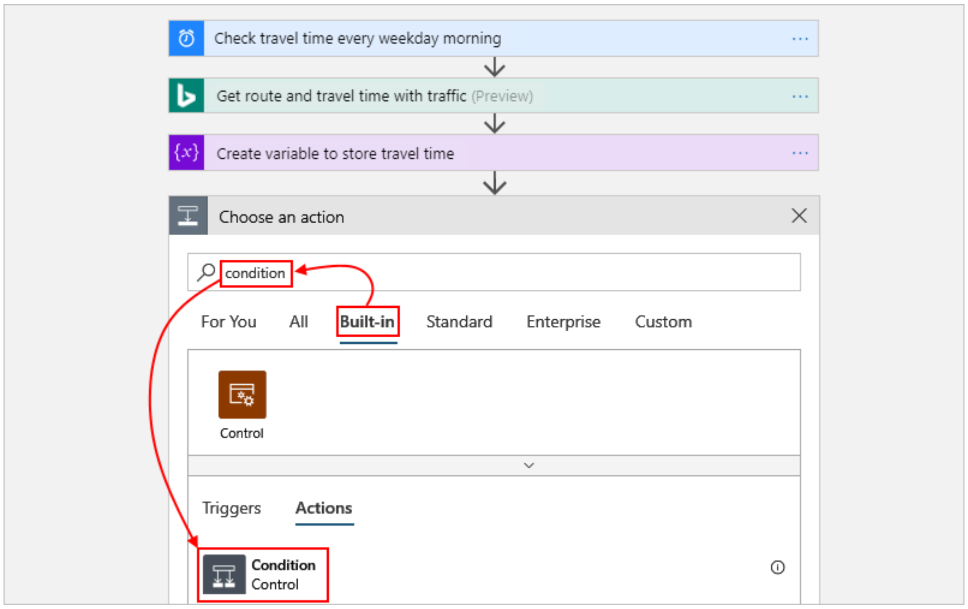
Value: div(...)

- Save your logic app.

Next, add a condition that checks whether the current travel time is greater than a specific limit.

Compare the travel time with limit

- Under the previous action, select **New step**.
- Under **Choose an action**, select **Built-in**. In the search box, enter "condition" as your filter. From the actions list, select the **Condition** action.

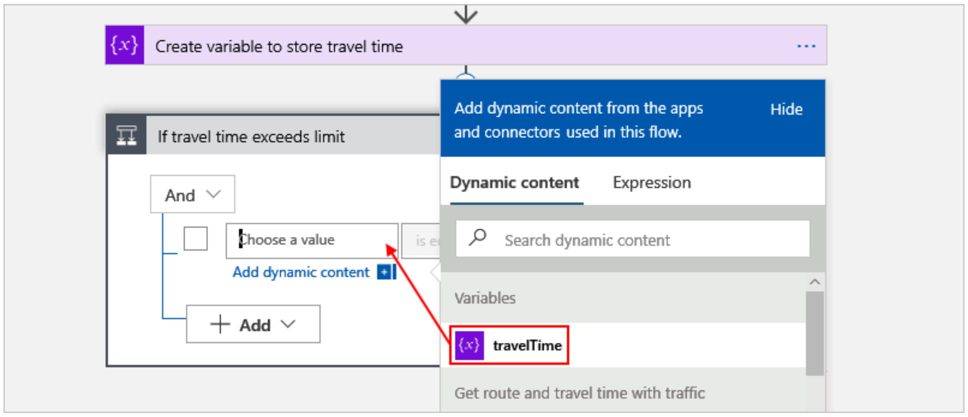

Check travel time every weekday morning
Get route and travel time with traffic (Preview)
Create variable to store travel time
Choose an action
condition
For You
All
Built-in
Standard
Enterprise
Custom
Control
Triggers
Actions
Condition
Control

- Rename the condition with this description: `If travel time exceeds limit`

- Build a condition that checks whether the **travelTime** property value exceeds your specified limit as described and shown here:

  In the condition, click inside the **Choose a value** box on the condition's left side.

  From the dynamic content list that appears, under **Variables**, select the **travelTime** property.



Create variable to store travel time

If travel time exceeds limit

And

Choose a value

Add dynamic content

Add

Add dynamic content from the apps and connectors used in this flow.

Hide

Dynamic content

Expression

Search dynamic content

Variables

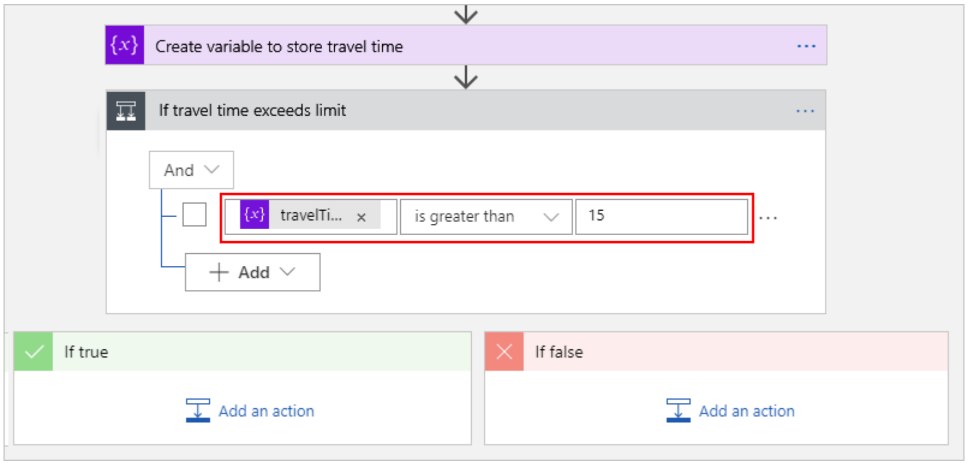
travelTime

Get route and travel time with traffic

  In the middle comparison box, select the **is greater than** operator.

  In the **Choose a value** box on the condition's right side, enter this limit: 15

  When you're done, the condition looks like this example:



Create variable to store travel time

If travel time exceeds limit

And

travelTi... × | is greater than | 15

Add

If true

Add an action

If false

Add an action

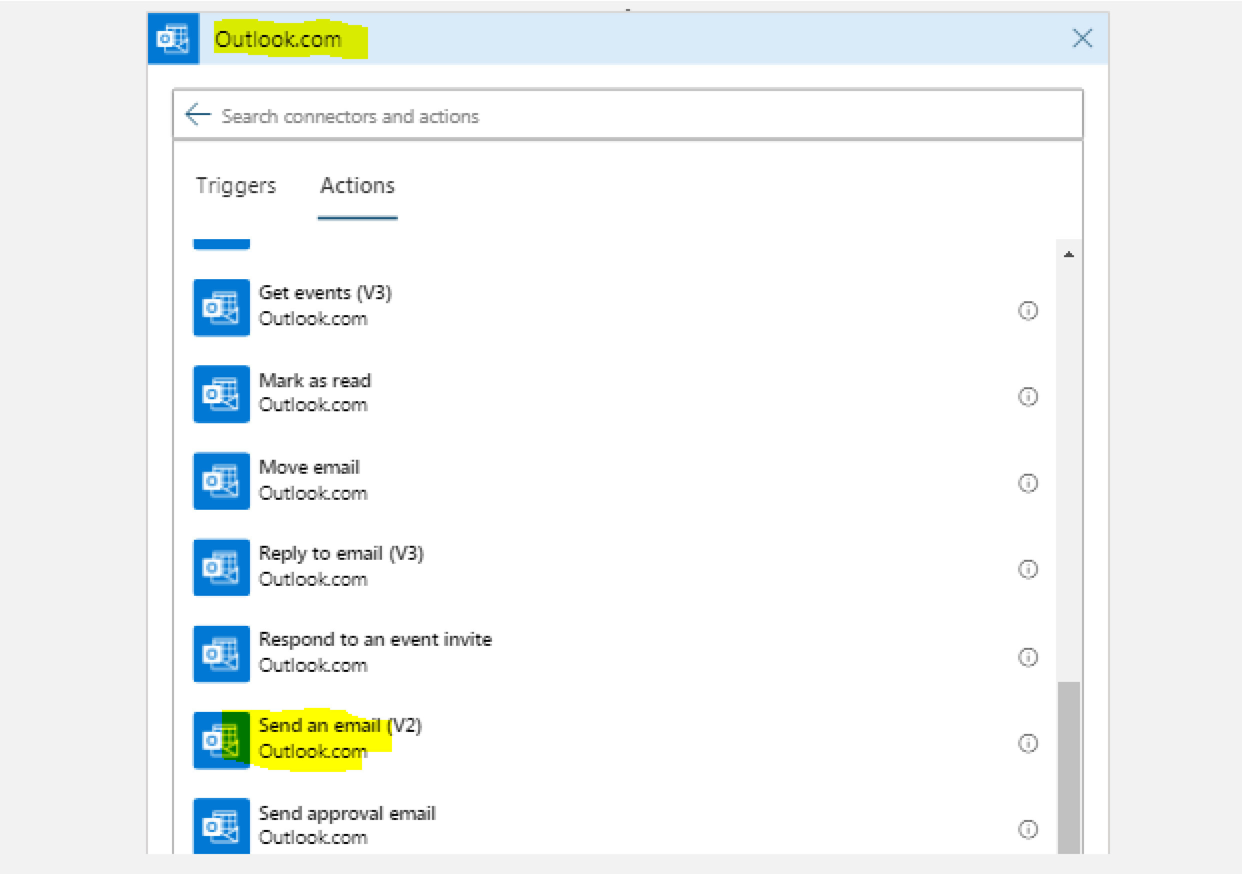
- Save your logic app.

Next, add the action to run when the travel time exceeds your limit.

Send email when limit exceeded

Now, add an action that emails you when the travel time exceeds your limit. This email includes the current travel time and the extra time necessary to travel the specified route.

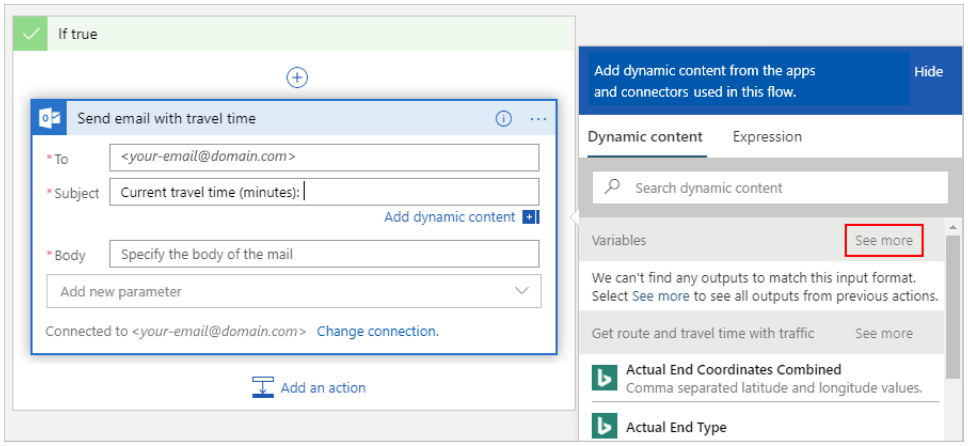
- In the condition's **If true** branch, select **Add an action**.
- Under **Choose an action**, select **Standard**. In the search box, enter "send email". The list returns many results, so first select the email connector that you want, for example:



  - For personal Microsoft accounts, select **Outlook.com**.
  - When the connector's actions appear, select "send email action" that you want to use, for example:
- If you don't already have a connection, you're asked to sign in to your email account.
- Logic Apps creates a connection to your email account.
- Rename the action with this description: `Send email with travel time`
- In the **To** box, enter the recipient's email address. For testing purposes, use your email address.
- In the **Subject** box, specify the email's subject, and include the **travelTime** variable.

  Enter the text `Current travel time (minutes):` with a trailing space.

  In the dynamic content list, under **Variables**, select **See more**.



Add dynamic content from the apps and connectors used in this flow.

*To <your-email@domain.com>

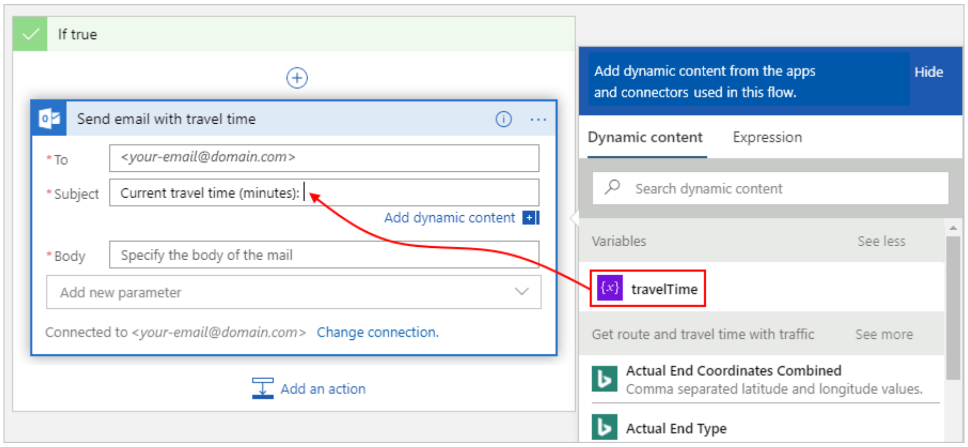
*Subject Current travel time (minutes):

*Body Specify the body of the mail

Connected to <your-email@domain.com> Change connection.

We can't find any outputs to match this input format. Select See more to see all outputs from previous actions.

After **travelTime** appears under **Variables**, select **travelTime**.



Add dynamic content from the apps and connectors used in this flow.

*To <your-email@domain.com>

*Subject Current travel time (minutes):

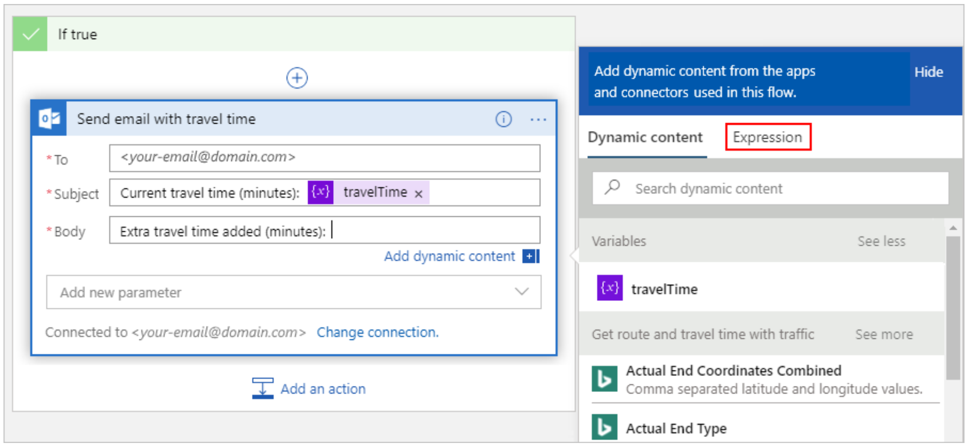
*Body Specify the body of the mail

Connected to <your-email@domain.com> Change connection.

- In the **Body** box, specify the content for the email body.

Enter the text `Add extra travel time (minutes):` with a trailing space.

In the dynamic content list, select **Expression**.



Add dynamic content from the apps and connectors used in this flow.

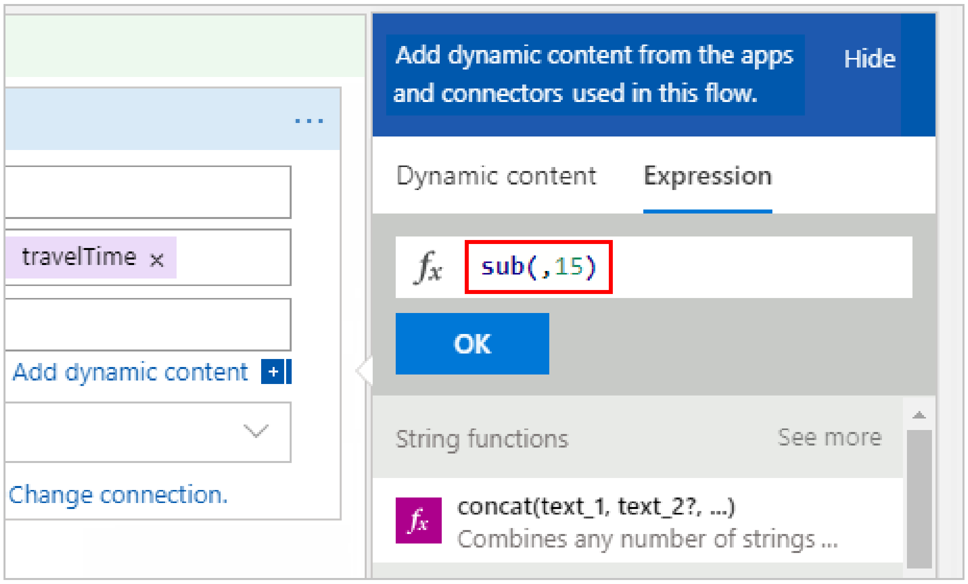
*To <your-email@domain.com>

*Subject Current travel time (minutes): travelTime ×

*Body Extra travel time added (minutes):

Connected to <your-email@domain.com> Change connection.

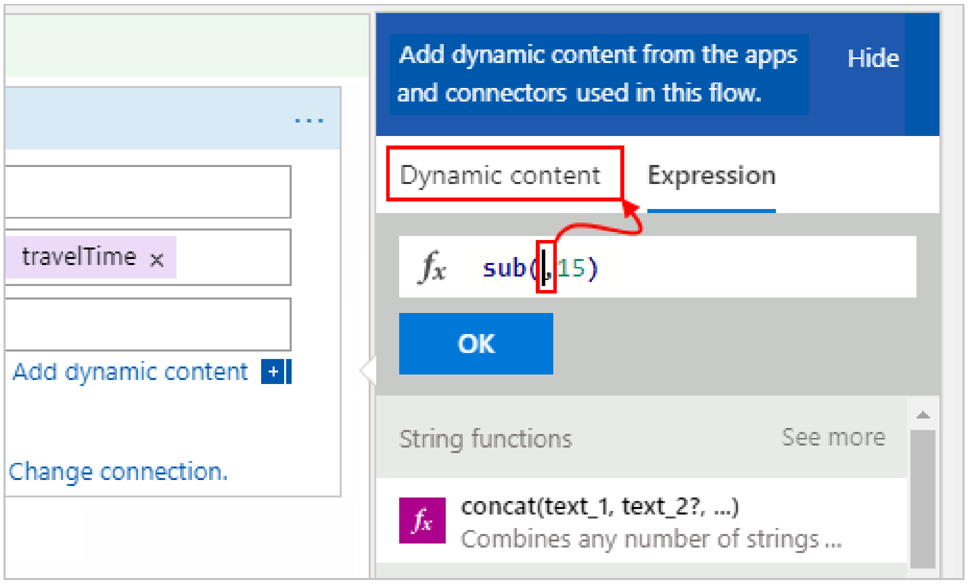
In the expression editor, enter this expression so that you can calculate the number of minutes that exceed your limit: `sub(,15)`



Add dynamic content from the apps and connectors used in this flow.

```
sub(,15)
```

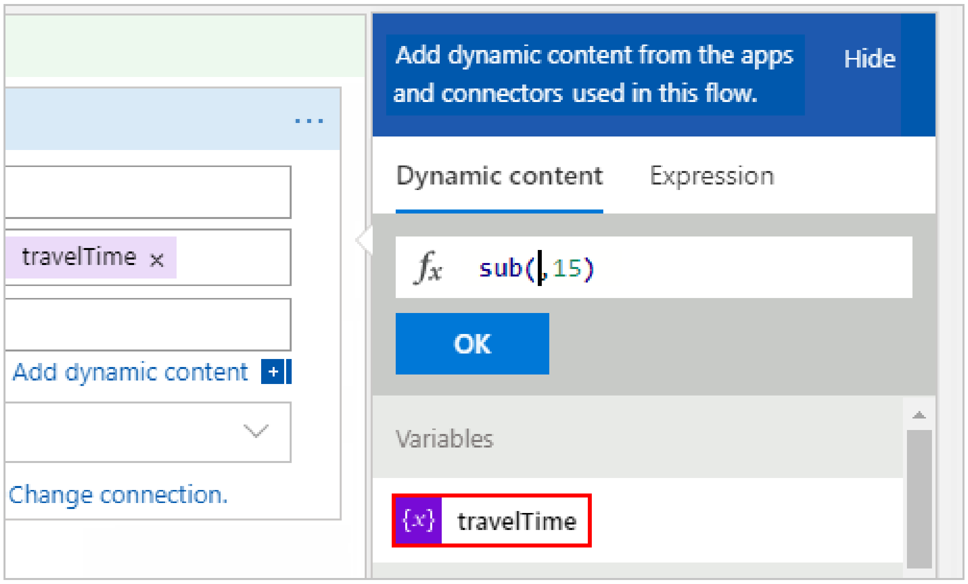
Put your cursor inside the expression between the left parenthesis (**(**) and the comma (**,**). Select **Dynamic content**.



Add dynamic content from the apps and connectors used in this flow.

```
sub(,15)
```

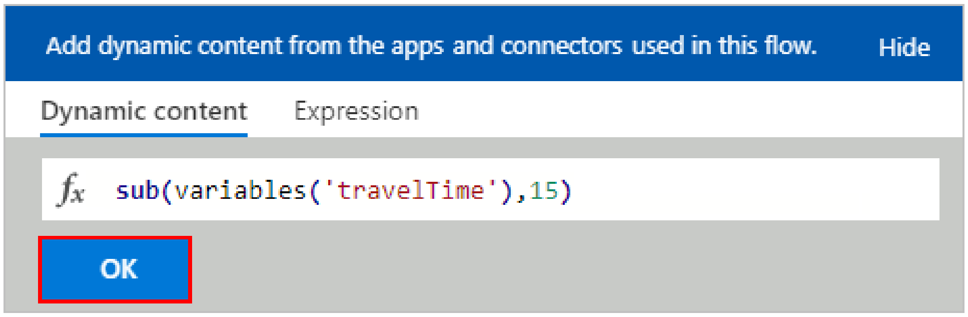
Under **Variables**, select **travelTime**.



Add dynamic content from the apps and connectors used in this flow.

```
sub(,15)
```

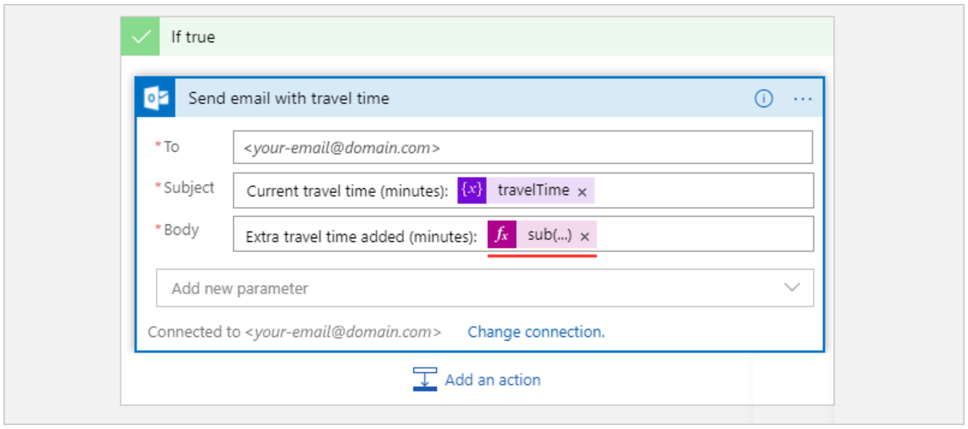
After the property resolves inside the expression, select **OK**.



Add dynamic content from the apps and connectors used in this flow.

```
sub(variables('travelTime'),15)
```

The **Body** property now appears as shown here:



If true

Send email with travel time

*To: <your-email@domain.com>

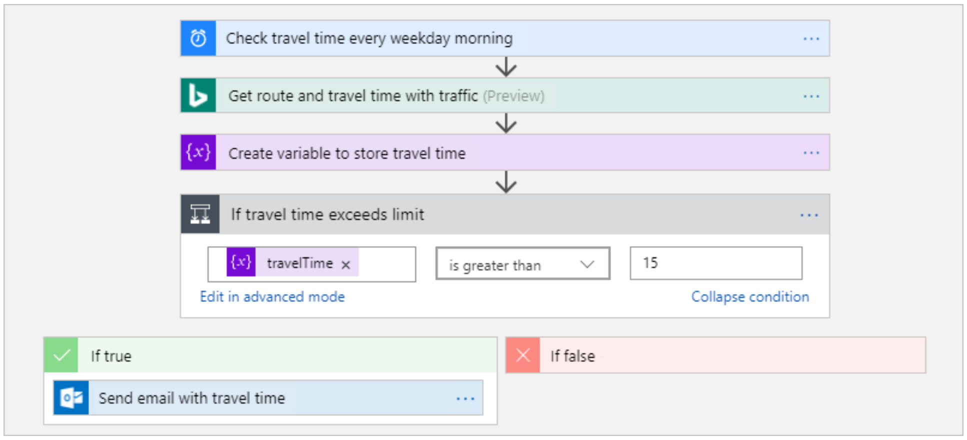
*Subject: Current travel time (minutes): travelTime ×

*Body: Extra travel time added (minutes): sub(...) ×

Connected to <your-email@domain.com> Change connection.

- Save your logic app.

Next, test your logic app, which now looks similar to this example:

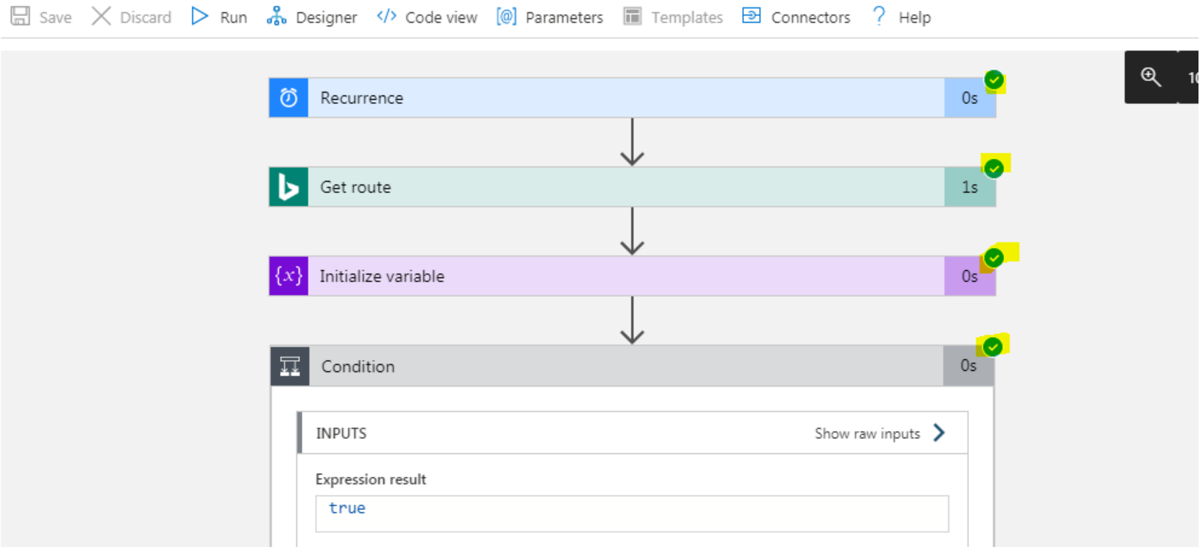

Check travel time every weekday morning
Get route and travel time with traffic (Preview)
Create variable to store travel time
If travel time exceeds limit
Edit in advanced mode
Collapse condition
If true
Send email with travel time
If false

travelTime × | is greater than | 15

Run your logic app

To manually start your logic app, on the designer toolbar bar, select **Run**.

- If the trigger successful the following screen will display.

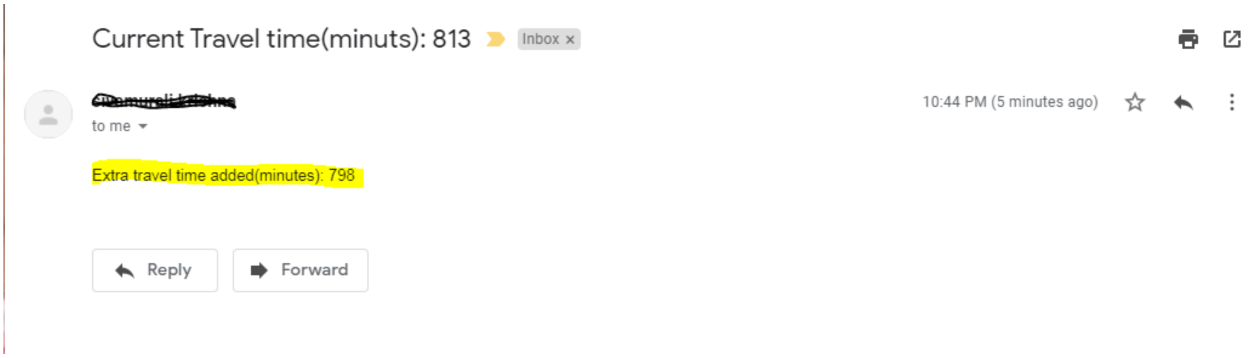


Expression result

true

Note: If any thing went wrong it will show with red color. Then go to the designer and check the error.

- If the current travel time stays under your limit, your logic app does nothing else and waits or the next interval before checking again.
- If the current travel time exceeds your limit, you get an email with the current travel time and the number of minutes above your limit. Here is an example email that your logic app sends:



Current Travel time(minuts): 813

[redacted]

to me

10:44 PM (5 minutes ago)

Extra travel time added(minutes): 798

If you don't get any emails, check your email's junk folder. Your email junk filter might redirect these kinds of mails. Otherwise, if you're unsure that your logic app ran correctly, see Troubleshoot your logic app.