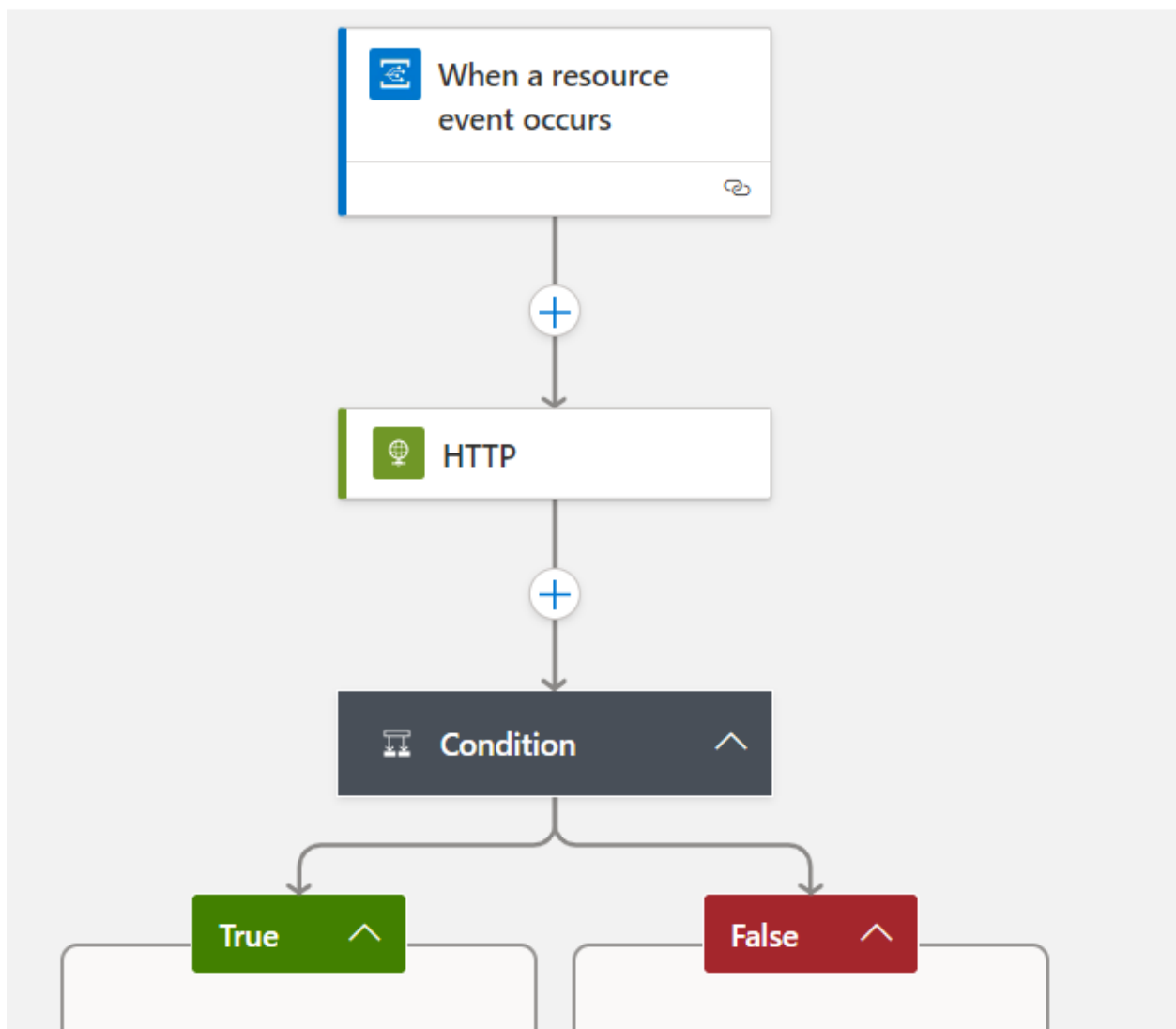


Consume events with Logic Apps

- Article
- 07/06/2022
- 8 minutes to read

This tutorial shows how to use Azure Logic Apps to process Azure Health Data Services Fast Healthcare Interoperability Resources (FHIR®) events. Logic Apps create and run automated workflows to process event data from other applications. You will learn how to register a FHIR event with your Logic App, meet a specified event criteria, and perform a service operation.

Here's an example of a Logic App workflow:



The workflow is on the left and the trigger condition is on the right.

Overview

Follow these steps to create a Logic App workflow to consume FHIR events:

1. Set up prerequisites
2. Create a Logic App
3. Create a Logic App workflow

Prerequisites

Before you begin this tutorial, you need to have deployed a FHIR service and enabled events. For more information about deploying events, see [Deploy Events in the Azure portal](#).

Creating a Logic App

To set up an automated workflow, you must first create a Logic App. For more information about Logic Apps, see [What is Azure Logic Apps?](#)

Specify your Logic App details

Follow these steps:

1. Go to the Azure portal.
2. Search for "Logic App".
3. Click "Add".
4. Specify Basic details.
5. Specify Hosting.
6. Specify Monitoring.
7. Specify Tags.
8. Review and create your Logic App.

You now need to fill out the details of your Logic App. Specify information for these five categories. They are in separate tabs:

Create Logic App ...

Basics Hosting Monitoring Tags Review + create

Create a logic app, which lets you group workflows as a logical unit for easier management, deployment and sharing of resources. Workflows let you connect your business-critical apps and services with Azure Logic Apps, automating your workflows without writing a single line of code.

Project Details

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

Subscription * ⓘ

Resource Group * ⓘ [Create new](#)

Instance Details

Logic App name * .azurewebsites.net

Publish * ☒ Workflow ☐ Docker Container

Region *

i Not finding your App Service Plan? Try a different region or select your App Service Environment.

Review + create

< Previous

Next : Hosting >

- Tab 1 - Basics
- Tab 2 - Hosting
- Tab 3 - Monitoring
- Tab 4 - Tags
- Tab 5 - Review + Create

Basics - Tab 1

Start by specifying the following basics:

Project details

- Subscription
- Resource Group

Select a current subscription and specify an existing or new resource group.

Instance details

- Logic App name
- Publish type
- Region

Create a name for your Logic App. You must choose Workflow or Docker Container as your publishing type. Select a region that is compatible with your plan.

Plan

- Plan type
- App Service Plan
- Sku and size

Choose a plan type (Standard or Consumption). Create a new Windows Plan name and specify the Sku and size.

Zone redundancy

- Zone redundancy deployment

Enabling your plan will make it zone redundant.

Hosting - Tab 2

Continue specifying your Logic App by clicking "Next: Hosting".

Storage

- Storage type
- Storage account

Choose the type of storage you want to use and the storage account. You can use Azure Storage or add SQL functionality. You must also create a new storage account or use an existing one.

Monitoring - Tab 3

Continue specifying your Logic App by clicking "Next: Monitoring".

Monitoring with Application Insights

- Enable Application Insights
- Application Insights
- Region

Enable Azure Monitor application insights to automatically monitor your application. If you enable insights, you must create a new insight and specify the region.

Tags - Tab 4

Continue specifying your Logic App by clicking "Next: Tags".

Use tags to categorize resources

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups.

This example will not use tagging.

Review + create - Tab 5

Finish specifying your Logic App by clicking "Next: Review + create".





Review your Logic App


Your proposed Logic app will display the following details:

- Subscription
- Resource Group
- Logic App Name
- Runtime stack
- Hosting
- Storage
- Plan
- Monitoring


If you're satisfied with the proposed configuration, click "Create". If not, click "Previous" to go back and specify new details.

First you'll see an alert telling you that deployment is initializing. Next you'll see a new page telling you that the deployment is in progress.


 Delete  Cancel  Redeploy  Refresh

 We'd love your feedback! →

Deployment is in progress

 Deployment name:
Subscription:
Resource group:





Start time:
Correlation ID:





Deployment details [\(Download\)](#)


Resource	Type	Status	Operation details
No results.			

If there are no errors, you will finally see a notification telling you that your deployment is complete.

 Delete  Cancel  Redeploy  Refresh


 We'd love your feedback! →

 **Your deployment is complete**



Deployment name:
Subscription:
Resource group:

Start time:
Correlation ID:



▼ Deployment details [\(Download\)](#)


^ Next steps


[Go to resource](#)







Your Logic App dashboard


Azure creates a dashboard when your Logic App is complete. The dashboard will show you the status of your app. You can return to your dashboard by clicking Overview in the Logic App menu. Here's a Logic App dashboard:


Home > [mylogicapp222222222](#) >


 **Workflowspindrift323** ...


 Workflow


 Run Trigger ▼  Refresh  Enable  Disable  Delete  Feedback

 Overview

 Code

 Designer

 Settings

 Access Keys

^ Essentials

Logic app : [mylogicapp222222222](#)

Status : Enabled

State type : Stateful


Workflow URL : [https://mylogicapp222222222.azurewebsites.net:443/runtime/webhooks/workf...](#)

Trigger Type : ApiConnectionWebhook

Application Insights : [mylogicapp222222222](#)

Health Status : Healthy

Get started [Run History](#) Trigger history

 Resubmit

You can do the following activities from your dashboard.

- Browse
- Refresh
- Stop
- Restart
- Swap
- Get Publish Profile
- Reset Publish Profile
- Delete

Creating a Logic App workflow

When your Logic App is running, follow these steps to create a Logic App workflow:

1. Initialize a workflow
2. Configuring a workflow
3. Designing a workflow
4. Adding an action
5. Giving FHIR Reader access
6. Adding a condition
7. Choosing a condition criteria
8. Testing your condition

Initializing your workflow

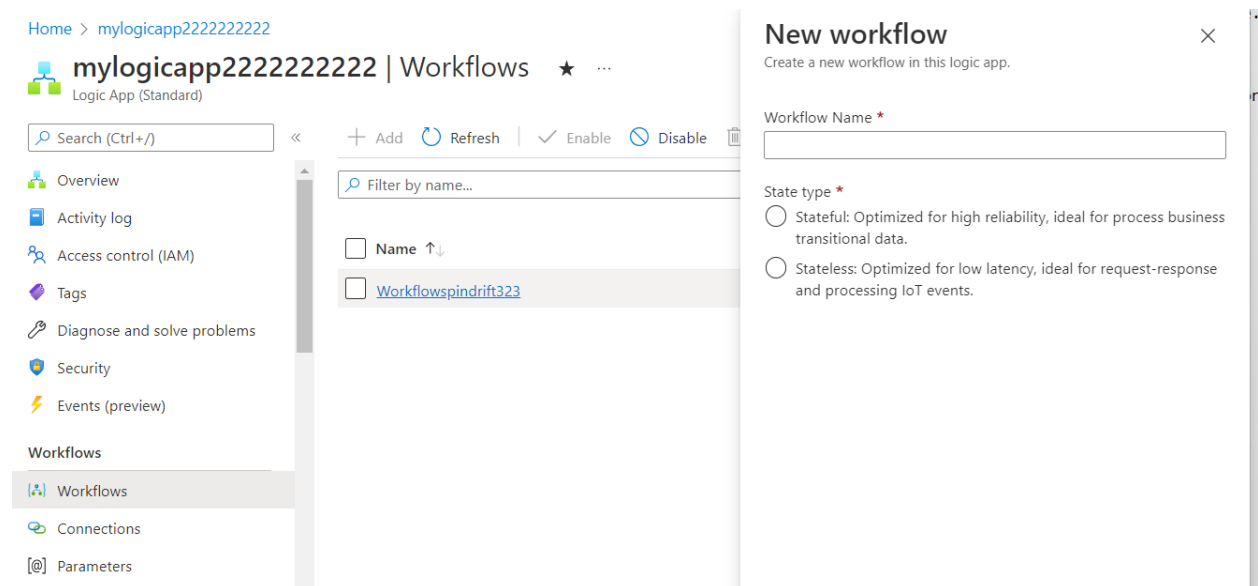
Before you begin, you'll need to have a Logic App configured and running correctly.

Once your Logic App is running, you can create and configure a workflow. To initialize a workflow, follow these steps:

1. Start at the Azure portal.
2. Click "Logic Apps" in Azure services.
3. Select the Logic App you created.
4. Click "Workflows" in the Workflow menu on the left.
5. Click "Add" to add a workflow.

Configuring a new workflow

You will see a new panel on the right for creating a workflow.



You can specify the details of the new workflow in the panel on the right.

Creating a new workflow for the Logic App

To set up a new workflow, fill in these details:

- Workflow Name
- State type

Specify a new name for your workflow. Indicate whether you want the workflow to be stateful or stateless. Stateful is for business processes and stateless is for processing IoT events.

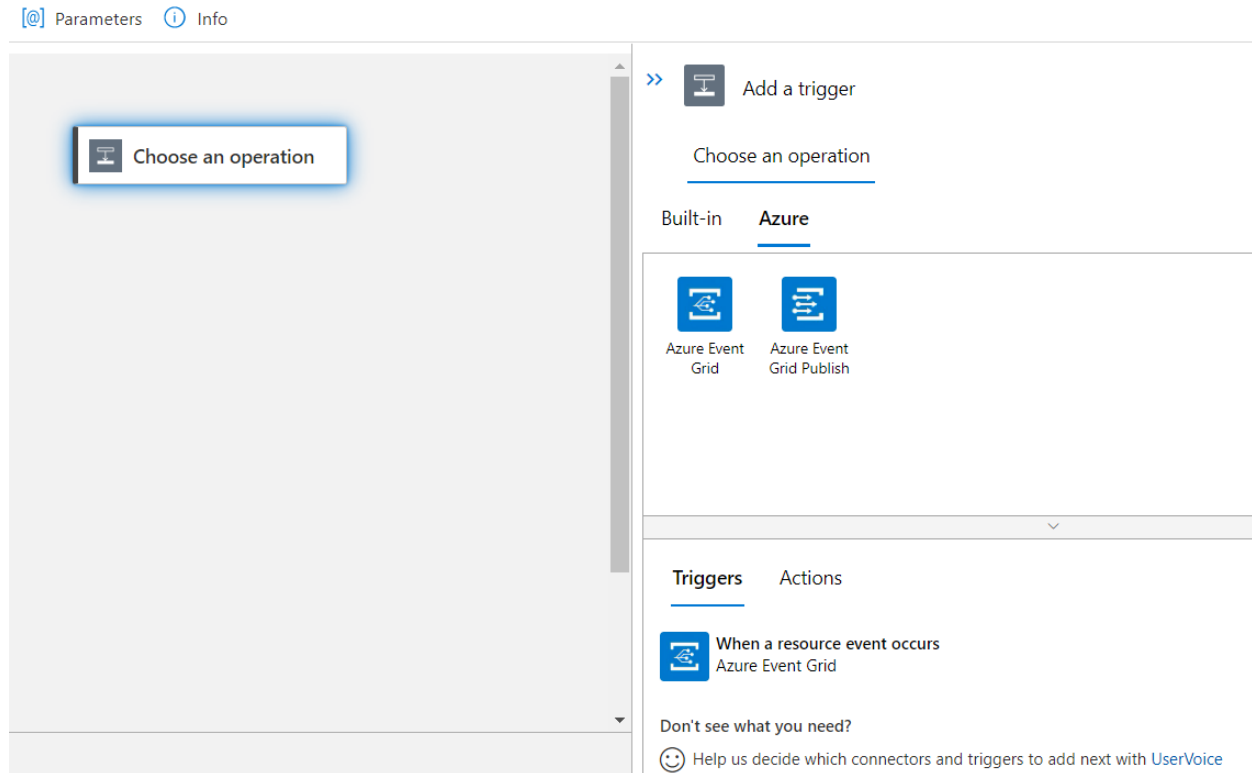
When you have specified the details, click "Create" to begin designing your workflow.

Designing the workflow

In your new workflow, click the name of the enabled workflow.

You can write code to design a workflow for your application, but for this tutorial, choose the Designer option on the Developer menu.

Next, click "Choose an operation" to display the "Add a Trigger" blade on the right. Then search for "Azure Event Grid" and click the "Azure" tab below. The Event Grid is not a Logic App Built-in.



When you see the "Azure Event Grid" icon, click on it to display the Triggers and Actions available from Event Grid. For more information about Event Grid, see [What is Azure Event Grid?](#).

Click "When a resource event occurs" to set up a trigger for the Azure Event Grid.

To tell Event Grid how to respond to the trigger, you must specify parameters and add actions.

Parameter settings

You need to specify the parameters for the trigger:

- Subscription
- Resource Type
- Resource Name
- Event type item(s)

Fill in the details for subscription, resource type, and resource name. Then you must specify the event types you want to respond to. The event types used in this article are:

- Resource created
- Resource deleted
- Resource updated

For more information about event types, see [What FHIR resource events does Events support?](#).

Adding an HTTP action

Once you have specified the trigger events, you must add more details. Click the "+" below the "When a resource event occurs" button.

You need to add a specific action. Click "Choose an operation" to continue. Then, for the operation, search for "HTTP" and click on "Built-in" to select an HTTP operation. The HTTP action will allow you to query the FHIR service.

The options in this example are:

- Method is "Get"
- URL is "concat('https://', triggerBody()?['subject'], '/_history/', triggerBody()?['dataVersion'])'".
- Authentication type is "Managed Identity".
- Audience is "concat('https://', triggerBody()?['data']['resourceFhirAccount'])"

Allow FHIR Reader access to your Logic App

At this point, you need to give the FHIR Reader access to your app, so it can verify that the event details are correct. Follow these steps to give it access:

1. The first step is to go back to your Logic App and click the Identity menu item.
2. In the System assigned tab, make sure the Status is "On".
3. Click on Azure role assignments. Click "Add role assignment".
4. Specify the following:
 - Scope = Subscription
 - Subscription = your subscription
 - Role = FHIR Data Reader.

When you have specified the first four steps, add the role assignment by Managed identity, using Subscription, Managed identity (Logic App Standard), and select your Logic App by clicking the name and then clicking the Select button. Finally, click "Review + assign" to assign the role.

Add a condition

After you have given FHIR Reader access to your app, go back to the Logic App workflow Designer. Then add a condition to determine whether the event is one you want to process. Click the "+" below HTTP to "Choose an operation". On the right, search for the word "condition". Click on "Built-in" to display the Control icon. Next click Actions and choose Condition.

When the condition is ready, you can specify what actions happen if the condition is true or false.

Choosing a condition criteria

In order to specify whether you want to take action for the specific event, begin specifying the criteria by clicking on "Condition" in the workflow on the left. You will then see a set of condition choices on the right.

Under the "And" box, add these two conditions:

- resourceType
- Event Type

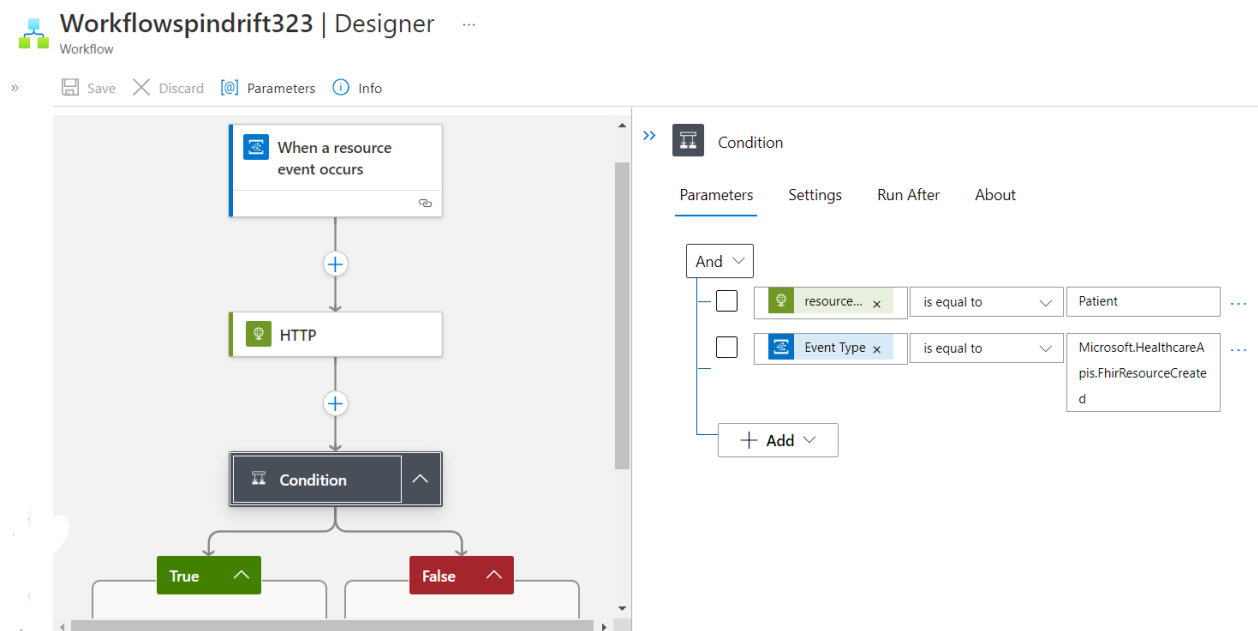
resourceType

The expression for getting the resourceType is `body('HTTP')?['resourceType']`.

Event Type

You can select Event Type from the Dynamic Content.

Here is an example of the Condition criteria:

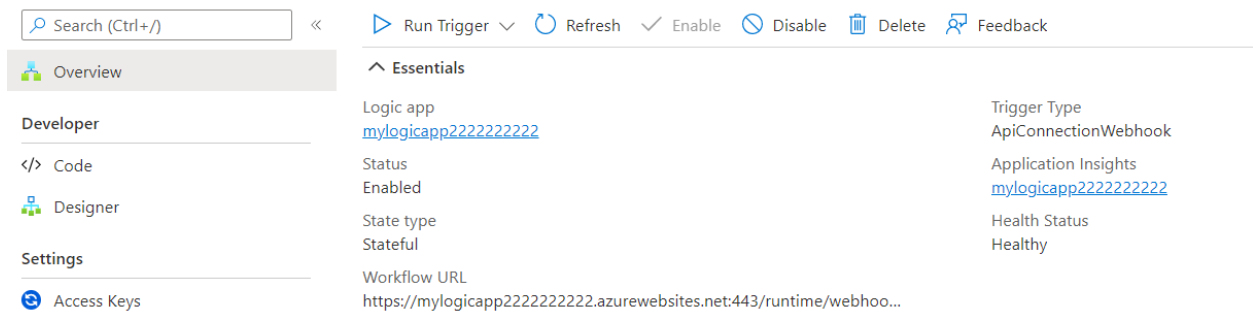


Save your workflow

When you have entered the condition criteria, save your workflow.

Workflow dashboard

To check the status of your workflow, click Overview in the workflow menu. Here is a dashboard for a workflow:



You can do the following operations from your workflow dashboard:

- Run trigger
- Refresh
- Enable
- Disable
- Delete

Condition testing


Save your workflow by clicking the "Save" button.

To test your new workflow, do the following steps:







1. Add a new Patient FHIR Resource to your FHIR Service.
2. Wait a moment or two and then check the Overview webpage of your Logic App workflow.
3. The event should be shaded in green if the action was successful.
4. If it failed, the event will be shaded in red.

Here is an example of a workflow trigger success operation:

Home > Logic apps > mylogicapp222222222 >

 **Workflowspindrift323** ...

Workflow

»  Run Trigger  Refresh  Enable  Disable  Delete  Feedback

Essentials View Cost JSON View

Logic app : [mylogicapp222222222](#)

Status : Enabled

State type : Stateful




Trigger Type : ApiConnectionWebhook


Application Insights : [mylogicapp222222222](#)

Health Status : Healthy


Workflow URL : <https://mylogicapp222222222.azurewebsites.net:443/runtime/webhook...>


Get started Run History **Trigger history**

All   MM/DD/YYYY  h:mm:ss A

When_a_resource_event_occurs 

Callback URL [POST]

https://mylogicapp222222222.azurewebsites.net:443/runtime/webhooks/workflow/scaleUnits/prod-00/workflows/b7db658f007543b8929315d9c0a87d56/triggers/When_a_resource_even... 

Identifier	Status ↑↓	Start time	Fired
12345678901234567890	 Succeeded	5/31/2022, 8:35 PM	Fired

Next steps

For more information about FHIR events, see

[What are Events?](#)

FHIR® is a registered trademark of Health Level Seven International, registered in the U.S. Trademark Office and is used with their permission.

Recommended content

- **[Supported FHIR features in FHIR service](#)**

This article explains which features of the FHIR specification that are implemented in Azure Health Data Services
- **[What is the MedTech service? - Azure Health Data Services](#)**

In this article, you'll learn about the MedTech service, its features, functions, integrations, and next steps.
- **[FHIR REST API capabilities for Azure Health Data Services FHIR service](#)**

This article describes the RESTful interactions and capabilities for Azure Health Data Services FHIR service.
- **[Data conversion for Azure API for FHIR](#)**

Use the \$convert-data endpoint and customize-converter templates to convert data in Azure API for FHIR.