FHIR Connectathon - mCODE Track Starter Guide

# Summary of Useful Links

|  |  |
| --- | --- |
|  |  |
| mCODE Connectathon Track Page | [**https://confluence.hl7.org/display/FHIR/2019-09+mCODE+Cancer+Interoperability+Track**](https://confluence.hl7.org/display/FHIR/2019-09+mCODE+Cancer+Interoperability+Track) |
| mCODE HSPC Sandbox FHIR Endpoint | [**https://api.logicahealth.org/mCODE/open**](https://api.logicahealth.org/mCODE/open) |
| mCODE Track Issues and Group Notes | [**https://confluence.hl7.org/display/FHIR/2019-09+mCODE+Track+Issues+and+Group+Notes**](https://confluence.hl7.org/display/FHIR/2019-09+mCODE+Track+Issues+and+Group+Notes) |
| mCODE FHIR chat stream | [**https://chat.fhir.org/#narrow/stream/179234-Cancer-Interoperability/topic/Connectathon.20track.20Atlanta**](https://chat.fhir.org/#narrow/stream/179234-Cancer-Interoperability/topic/Connectathon.20track.20Atlanta) |
|  |  |

# Summary of REST Calls for mCODE

1. **Search for all mCODE patients:**

HTTP GET

[*https://api.logicahealth.org/mCODE/open/Patient?\_profile=https://api.logicahealth.org/mCODE/open/StructureDefinition/obf-Patient*](https://api.logicahealth.org/mCODE/open/Patient?_profile=https://api.logicahealth.org/mCODE/open/StructureDefinition/obf-Patient)

1. **Search for the PrimaryCancerCondition for a given patient:**

Pre-requisite: the Patient FHIR ID is known.

In the example below, we have an mCODE patient with FHIR ID = 17041.

HTTP GET

[*https://api.logicahealth.org/mCODE/open/Condition?subject=Patient/17041*](https://api.logicahealth.org/mCODE/open/Condition?subject=Patient/17041)

1. **Search for the CancerStageGroup for a given patient:**

Pre-requisite: the PrimaryCancerCondition is known.

In the example below, we completed REST query #2 and identified the relevant primary cancer condition with FHIR ID = 17043

HTTP GET

[*https://api.logicahealth.org/mCODE/open/Condition?subject=Patient/17041*](https://api.logicahealth.org/mCODE/open/Condition?subject=Patient/17041)

1. Create a new mCODE Patient:

HTTP POST

[*https://api.logicahealth.org/mCODE/open/Patient*](https://api.logicahealth.org/mCODE/open/Patient)

Place the following in key and value data in the header info:

|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+json |
| Accept | application/fhir+json |

If your example is in XML format then change the header info to the following:

|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+xml |
| Accept | application/fhir+xml |

If you're creating a new patient, omit the id parameter from your example. The FHIR server will assign and return one to you in the response.

1. Update an existing mCODE Patient:

HTTP PUT

[*https://api.logicahealth.org/mCODE/open/Patient/<Patient FHIR ID>*](https://api.logicahealth.org/mCODE/open/Patient/%3cPatient%20FHIR%20ID%3e)

Place the following in key and value data in the header info:

|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+json |
| Accept | application/fhir+json |

If your example is in XML format then change the header info to the following:

|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+xml |
| Accept | application/fhir+xml |

Note: in the example that you're updating, make sure that the FHIR id is referenced in the body of the FHIR instance. For example, if I were to update mCODE patient with FHIR ID 17041:

{

"resourceType": "Patient",

"id": "17041",

"meta": {

"profile": [

"https://api.logicahealth.org/mCODE/open/StructureDefinition/obf-Patient"

]

},

*…etc…*

1. Validate an mCODE Patient

Pre-requisite: the Patient FHIR ID is known.

In the example below, we have an mCODE patient with FHIR ID = 17041.

HTTP GET

[*https://api.logicahealth.org/mCODE/open/Condition?subject=Patient/17041$validate*](https://api.logicahealth.org/mCODE/open/Condition?subject=Patient/17041$validate)

# Appendix A: Using Postman for Testing mCODE

This section is for users that are new to querying a FHIR server. It will give you just enough information to get one started with setting up an environment to post, retrieve, and update mCODE FHIR examples.

[Postman](https://www.getpostman.com/) is an application programming interface (API) development environment which includes client support for testing Representational State Transfer (REST) calls. The HL7 FHIR community oftentimes uses Postman to query a FHIR server.

Firely has created a good general tutorial for using Postman for FHIR here: <https://github.com/FirelyTeam/fhirstarters/tree/master/postman/crud>

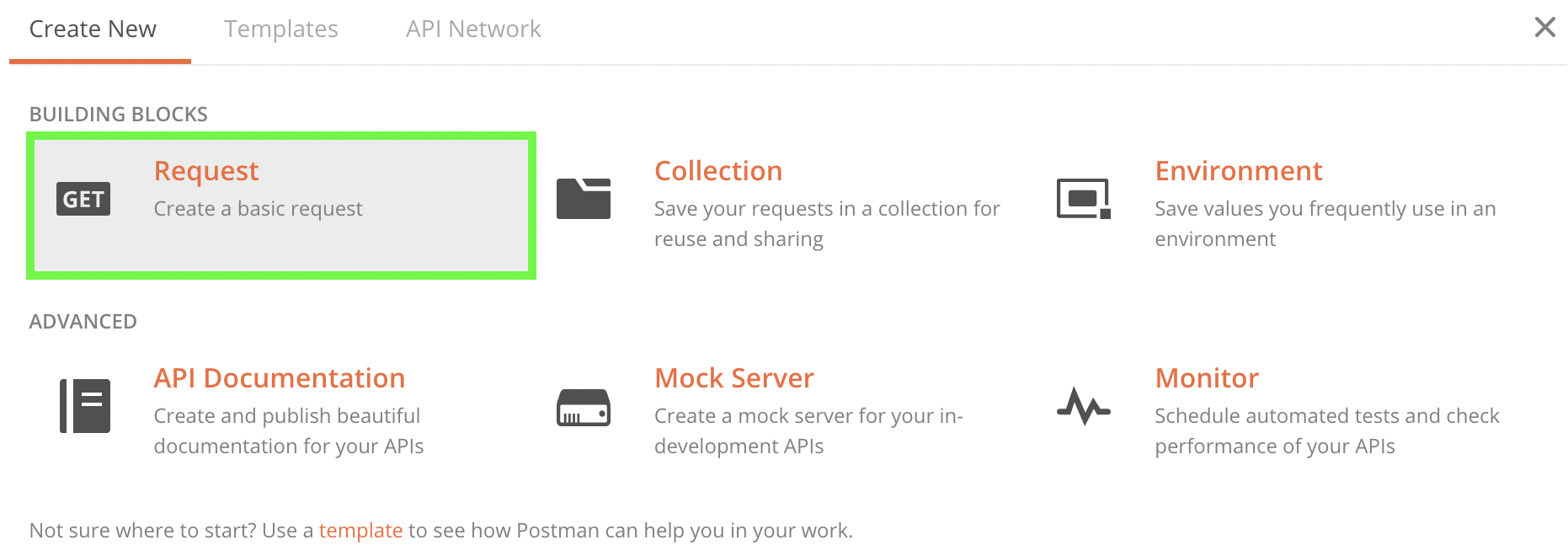
We'll be customizing that tutorial for the mCODE HSPC Sandbox FHIR server in the steps that follow.

## Installing Postman

Download Postman here: <https://www.getpostman.com/downloads/>.

Postman supports both Windows and OS X environments.

When you launch Postman you'll see the following modal window. Select *Request*.



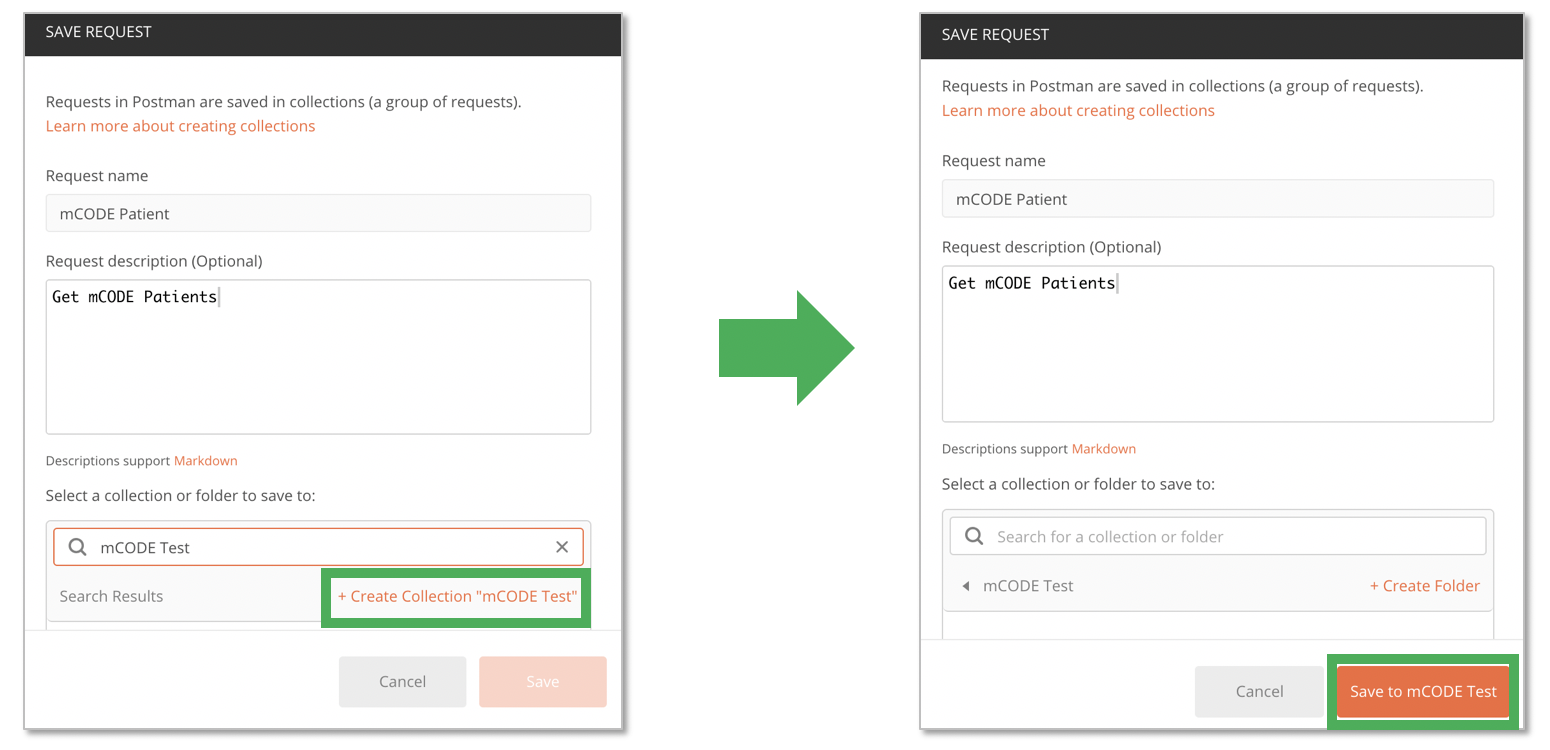
On the *Save Request* modal window, fill in the following fields:

* **Request name**
* **Requestion description** (optional field)
* **Select or create a version to save to** – a Postman Collection is like a folder which will contain all of your FHIR Rest Calls. As a good best practice, I like to create one collection per FHIR endpoint that I'm testing. E.g.: "mCODE Test"

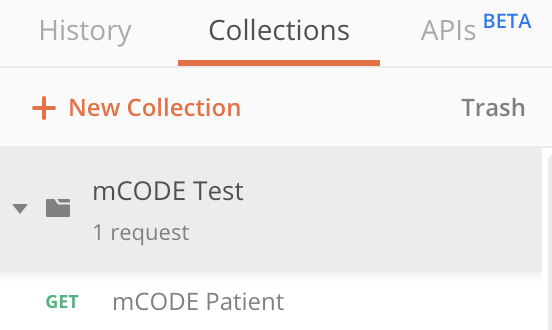
Click on "*+ Create Connection <Collection name>*" to create the collection

Now click on "*Save to <Collection name>*"

Screenshots on these steps are shown in the figure below:



One the left hand pane, you will see your created collection and the new request.



A new Untitled Request tab will now appear.

## Creating Your First Request

The HSPC Sandbox has been pre-populated with examples for the following FHIR R4 Base Resources:

* Patient
* Practitioner

Let's do a simple search to retrieve FHIR Patient resources.

HTTP **GET** is the default request that shows. In the field to the right of that call, type in the following:

<https://api.logicahealth.org/mCODE/open/Patient>

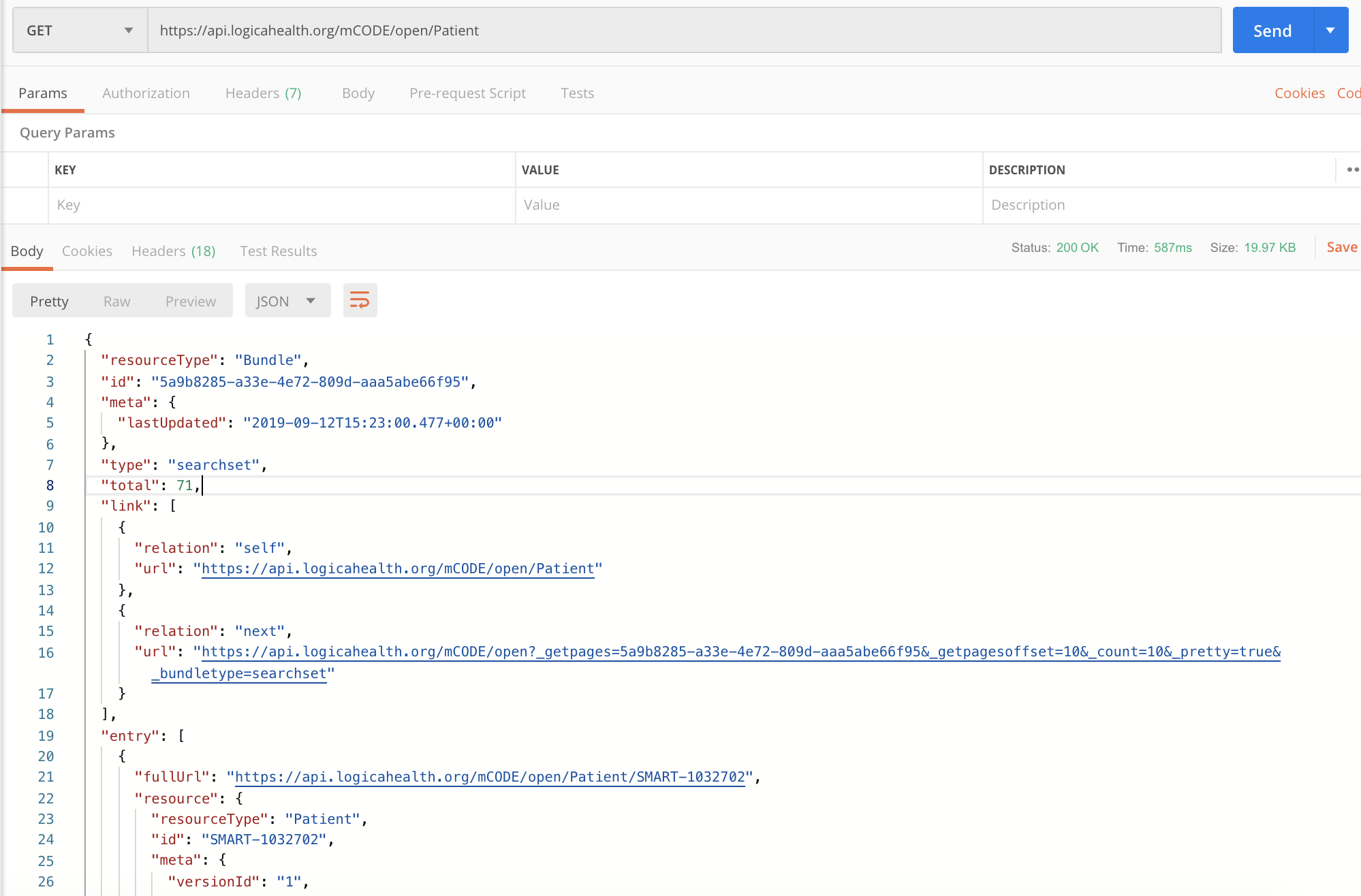
Click the **Send** Button.

The FHIR server will return a FHIR Bundle containing the FHIR instances for your Patient query.

Let's now query for an example mCODE conformant Patient resource. The sandbox contains one mCODE example patient named "John Anyperson".

Open up a new request by clicking on the "**+**" tab in the main menu bar and enter the following in the field:

<https://api.logicahealth.org/mCODE/open/Patient/?family=anyperson>



**Note:** The HSPC Sandbox limits the number of returns to the first 50 patients. This is not a limitation of FHIR, but by the HSPC implementation which was configured to conserve memory and processing in a shared environment of many implementations. You can use smarter search parameters to retrieve your patients of interest.

## Retrieve (GET) an mCODE example patient

Open a new request, select GET for the HTTP message and enter the following URL in the field:

<https://api.logicahealth.org/mCODE/open/Patient?_profile=https://api.logicahealth.org/mCODE/open/StructureDefinition/obf-Patient>

The FHIR server will return a FHIR Bundle with all Patient instances referencing the [mCODE-specified Patient profile](http://hl7.org/fhir/us/mcode/2019Sep/StructureDefinition-obf-Patient.html).

Retrieve the FHIR ID for the mCODE Patient instance that you'll want to further explore. For our example, we'll use FHIR ID 17041.

### Create (POST) a new mCODE example patient

Open a new request, select POST for the HTTP message type and enter the following URL in the field:

[*https://api.logicahealth.org/mCODE/open/Patient*](https://api.logicahealth.org/mCODE/open/Patient)

Click on the "Headers" tab in Postman and enter the following into the key and value fields:

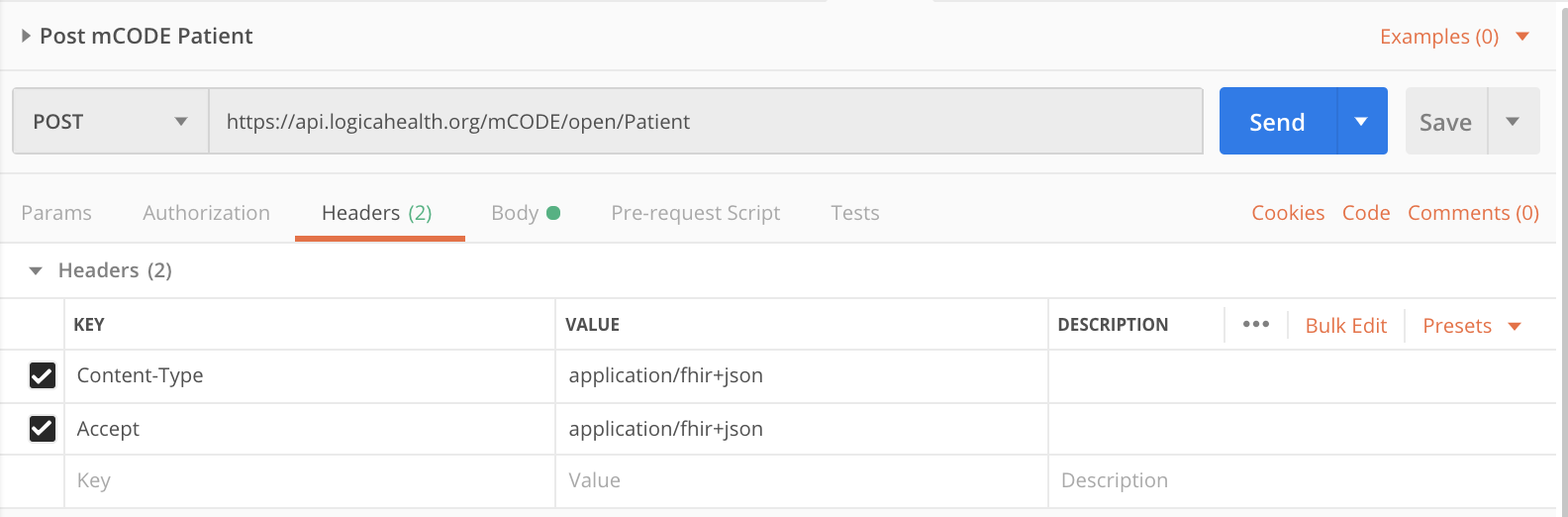
|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+json |
| Accept | application/fhir+json |

If your example is in XML format then change the header info to the following:

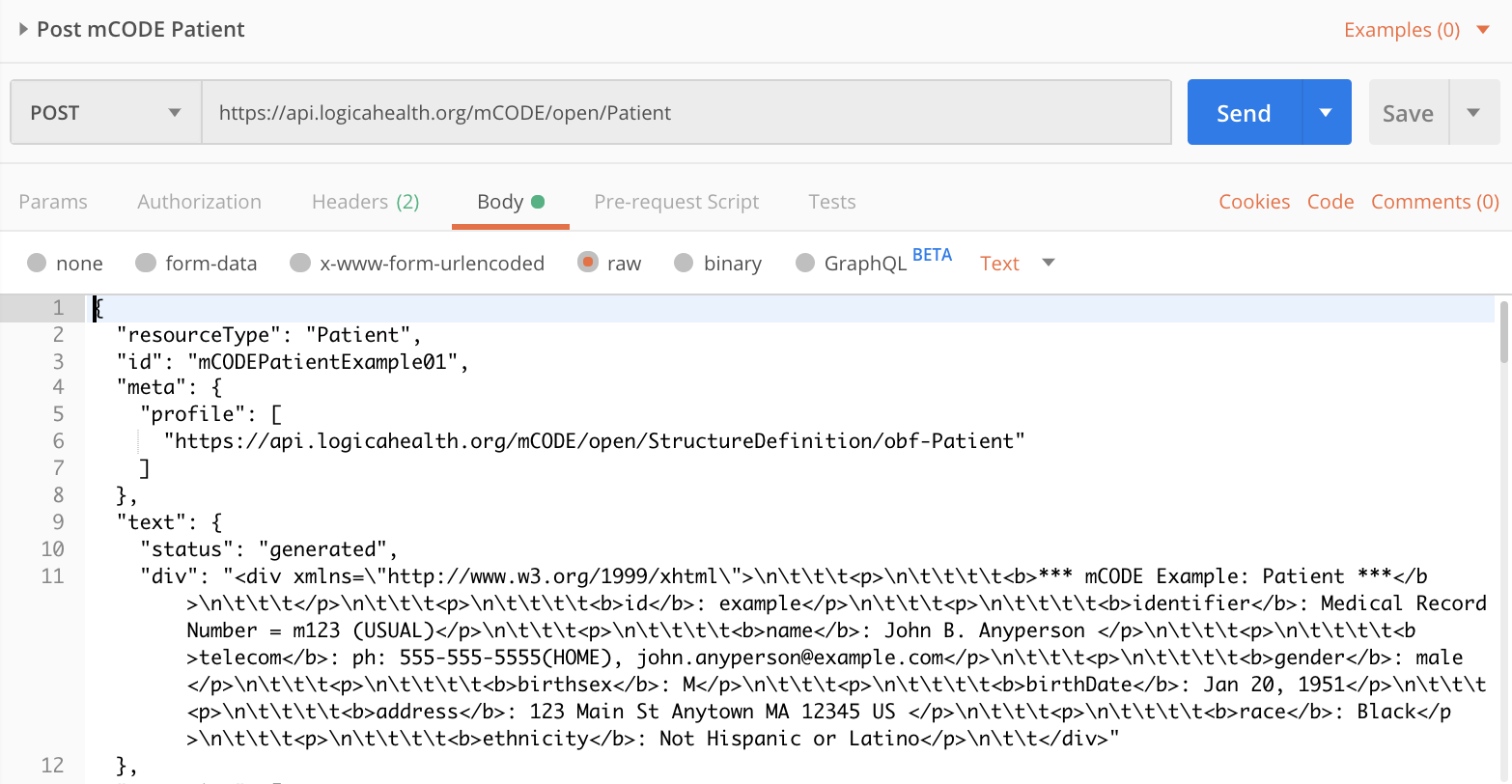
|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+xml |
| Accept | application/fhir+xml |

If you're creating a new patient, omit the id parameter from your example. The FHIR server will assign and return one to you in the response.

The figure below shows how this looks in the UI:



Click on the "Body" tab in Postman, select the "Raw" radio button and paste in your FHIR example data in the body.



Click the "Send" button.

If it's successful, you'll see an HTTP response of Status = 200 OK and a FHIR Bundle in the body of the response which contains your example as an entry within the Bundle.

## Update (PUT) the record for an existing mCODE patient

The HTTP PUT message is used to update a FHIR instance which has already been created. If you use HTTP POST (even if the message looks identical), the FHIR server will assign a new FHIR ID.

[*https://api.logicahealth.org/mCODE/open/Patient/<Patient FHIR ID>*](https://api.logicahealth.org/mCODE/open/Patient/%3cPatient%20FHIR%20ID%3e)

Place the following in key and value data in the header info:

|  |  |
| --- | --- |
| **Key** | **Value** |
| Content-Type | application/fhir+json |
| Accept | application/fhir+json |

Note: in the example that you're updating, make sure that the FHIR id is referenced in the body of the FHIR instance. For example, if I were to update mCODE patient with FHIR ID 17041:

{

"resourceType": "Patient",

"id": "17041",

"meta": {

"profile": [

"https://api.logicahealth.org/mCODE/open/StructureDefinition/obf-Patient"

]

},

*…etc…*

If the call is successful, you will get an HTTP Response of Status = 200 OK with the FHIR instance returned and containing the same FHIR ID but a versionId incremented by 1.

The figure below shows an example of a PUT message and its response.

