

MADIE TEST CASE JSON GUIDE

Version 1.2.0

Measure Authoring Development
integrated Environment




TABLE OF CONTENTS

..... 1

Table of Contents..... 2

Revisions..... 4

Introduction to Javascript object notation (JSON) 4

 About MADiE Test Cases 4

JSON Syntax..... 5

 JSON Datatypes and Rules..... 5

 Example -The FHIR Patient Resource in JSON 6

 Complex Datatypes and Arrays..... 6

Constructing FHIR Bundles in JSON 7

 Overview..... 7

 Structure..... 7

 Collection Bundles 7

 Example of a Measure Denominator Test Case Bundle 8

 Example of a Measure Denominator Exclusion Test Case Bundle 11

 Transaction Bundles 13

Steps for Creating a new Test Case in MADiE..... 13

Exporting Test Cases from BONNIE FHIR 18

Tips For Using QI-CORE Profile Structure Definitions 20

 Identifying the QI-Core Profile Official URL for use in the resource meta..... 20

 Profile Content 22

 Cardinality 23

DATA Type	24
Example Test Case BUNDLE: 'QI-COREV4.1.1 EXAMPLE BUNDLE VERSION 2'	24
Description	24
Instructions for Use	25

REVISIONS

Table 1: Revision History

Version	Author	Revisions
1.0.0	ICF	Initial MADiE Test Case JSON Guide
1.1.0	ICF	Added description and instruction for use of the QI-Corev4.1.1 Test Case Template and test case QI-Corev4.1.1 Example Bundle version 2
1.2.0	ICF	Added instructions for inclusion of meta.profile in bundle resources using QI-Core profiles

INTRODUCTION TO JAVASCRIPT OBJECT NOTATION (JSON)

JavaScript Object Notation (JSON) is a lightweight, text-based language for the purpose of storage, representation and exchange of structured data within and between applications. Because JSON is simple to read, compact and is supportive of commonly used data types and can support hierarchical structures, JSON has become a standard for representing and exchanging data in web-based applications and APIs. It is also a supported standard for representing FHIR data and metadata.

ABOUT MADIE TEST CASES

Measure Authoring Development Integrated Environment (MADiE) test cases are constructed using JSON for representing measures and test case bundles. In MADiE, bundles are JSON files, each representing a measure test case, typically for a population used in the measure such as Initial Population, Denominator and Numerator with criteria for meeting success or failure. For example, an Initial Patient resource may contain patient and encounter data or a procedure.

Test case bundles can be created directly within the MADiE tooling environment which offers a window to paste JSON and validate and includes validation errors and highlighting of the code segment that fails validation.

At this time all synthetic test cases must be constructed in JSON. The JSON test cases may be constructed within the MADiE application, or they may be constructed in an external editor and pasted into the MADiE JSON panel. When test cases are developed externally, it is recommended that use of an editing tool supporting JSON validation is employed. JSON test cases in MADiE must be syntactically correct and must be mapped correctly to QI-Core profiles to support the data model and correctly meet the intent of the measure. MADiE offers functionality to assist the developer in identifying JSON errors with additional features to help evaluate test case alignment with measure intent.

JSON SYNTAX

JSON structure at simplest form includes paired data identifiers and values separated by tokens in the form of a name followed by a colon literal and a value with comma separation within an object. For example, “first name”: “Anna”. The following rules apply to JSON syntax:

JSON DATATYPES AND RULES

- JSON is case sensitive.
- Supported data types include string, number, Boolean, null, object and array.
- Numbers can be integers or floating point.
- At the lowest level of granularity, data is specified as name and value pairs separated by a colon “:” symbol. For example: “first name”: “Jonathan”
- Data elements are separated by commas.
- Objects or complex data types are enclosed within curly braces {}.
- Array data is enclosed within square brackets [].
- Quotation marks are used to surround names, data when not numeric or Boolean.

EXAMPLE -THE FHIR PATIENT RESOURCE IN JSON

Below is a snippet from the FHIR v4.0.1 Patient Resource in JSON format.

```
{
  "resource": {
    "resourceType": "Patient",
    "id": "ipl-pass",
    "identifier": [
      {
        "use": "usual",
        "type": {
          "coding": [
            {
              "system": "http://terminology.hl7.org/CodeSystem/v2-0203",
              "code": "MR",
              "display": "Medical Record Number"
            }
          ]
        },
        "system": "http://hospital.smarthealthit.org",
        "value": "9999999911"
      }
    ],
    "name": [
      {
        "family": "Pass",
        "given": [
          "John"
        ]
      }
    ],
    "gender": "male",
    "birthDate": "1964-06-30"
  }
}
```

Figure 1: FHIR Patient Resource

The JSON file contains a single patient.

Pairing names are always quoted and commas are used to delineate data elements. Unless the value is numeric or Boolean, the pairing value is also quoted. The cardinality of the element will determine if it needs to be enclosed in square brackets indicative of array values, or elements which can have more than one value. Complex datatype elements will be enclosed in curly braces.

COMPLEX DATATYPES AND ARRAYS

Using the [HL7 FHIR Resource structure definition for the Patient resource](#), the cardinality and complexity of each object can be obtained. The datatype is hyperlinked In the HL7 FHIR

Resource or Profile structure definition. Name and Coding are complex datatypes in this example. Additionally, a given name may have more than one value and is also an array as denoted below with square brackets.

CONSTRUCTING FHIR BUNDLES IN JSON

OVERVIEW

MADiE test cases in JSON follow the structure of a [FHIR Bundle](#). A bundle is another resource in FHIR. For simplicity, JSON bundles may be thought of as means of combining a group of resources for the server in a transaction which reduces roundtrips to the server and mitigates loss of referential integrity. MADiE test file bundles can be thought of as the resources necessary to execute the test case and meet its intended results. In most cases it will include a patient resource and may include encounters, conditions, procedures, observations, and other resources.

MADiE test case bundles may be of type collection (value='collection') or transaction (value='transaction'). Special rules in FHIR apply to different kinds of bundles. For example, the resources in bundles of type transaction will also contain a FHIR operation such as POST or PUT. For more information on Bundle resources and what rules are applicable to each type of bundle, refer to the HL7 [Bundle](#) resource.

STRUCTURE

Test case bundles must contain a resource type of 'Bundle', a string identifier which may be generated or named for the test case. The Bundle type may be a collection or a transaction. This is followed by an array of resource objects needed for the test case.

COLLECTION BUNDLES

The bundle begins with a Resource type of 'Bundle' and a bundle id and type of value collection. This is followed by an entry array consisting of the test case resources. In this case, there are two resources included for patient and procedure. Each resource must include a unique

identifier pairing and a fullUrl pairing. The id value must be unique for each resource in the JSON bundle however it may also be used in reference elements referring to another resource of a resource may be used in subsequent resource references such as an encounter reference to a patient within the bundle.

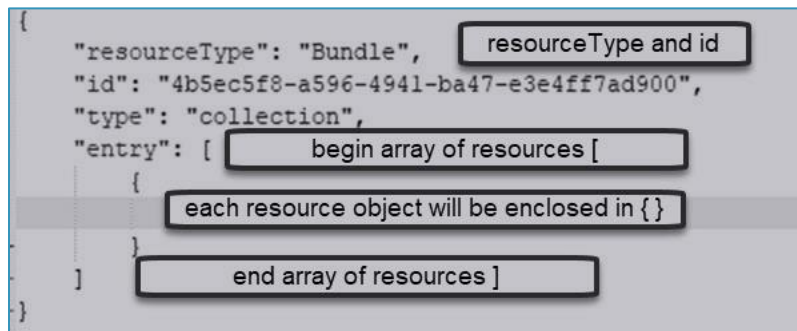


Figure 2: Bundle Structure

When the measure model is QI-Core, the best practice is to use QI-CORE profiles for bundle resources to avoid unexpected results. The HL7 FHIR QI-Core Implementation Guide can be followed for guidance in populating test case resource entries in JSON bundles.

Note: It may be helpful to use a UUID generator for assignment of id elements within bundles.

Steps to populate resource entries which may be helpful are:

1. Identify data elements needed for your measure
2. Map additional data elements needed for the measure to the corresponding profile requirement.
3. Reference the QI-Core Profiles to determine additional data elements necessary as determined by the element cardinality
4. Note the datatype of the data element on the QI-Core profile structure for non-primitive data types.

EXAMPLE OF A MEASURE DENOMINATOR TEST CASE BUNDLE

Consider a JSON bundle for a test case meeting a denominator population for a measure requiring a patient aged between 18 and 65 and a cataract surgical procedure during the

measurement year. The JSON bundle will require a patient and procedure resource and will reference [the QI-Core version 4.1.1 Profiles](#) for [QICorePatient](#) and [QICoreProcedure](#).

The JSON can be started with the collection bundle structure as illustrated in Figure 1 and replacing the bundle id with another UUID.

Profile	Data elements needed for measure	Data Elements required by Profile
QICorePatient	birthDate	identifier, name, gender

Examining the QICorePatient profile structure definition note the profile requires that at least one identifier data element of data type Identifier, a complex datatype. It also requires at least one name of datatype HumanName, also a complex datatype. Cardinality mandates a gender is also entered. Although it is not required by the profile, the measure requirement needs a birthdate data element.

Enter the patient entry as an object within the collection array. The fullUrl and id are each unique for each entry. Notably this includes the ids of resources which have more than one instance in the bundle such as two encounters. Meta is populated to reflect the correct QI-Core profile being used. A medical record number is included to fulfill the requirement for an identifier of data type [Identifier](#) in the profile structure definition. Additionally, Name is included conforming to FHIR data type [HumanName](#) and gender is added conforming to the profile definition (datatype [code](#)) and binding to [AdministrativeGender](#). Last a birthdate element is added to the entry object. The profile for QI-Core Patient also indicates the [datatype for birthdate](#).

The object is closed with a curly brace} and an added comma to indicate another entry follows.

```

{
  "resourceType": "Bundle",
  "id": "QICoreExamples-1",
  "type": "collection",
  "entry": [
    {
      "fullUrl": "f6fdb684-9acd-11ed-a23b-af8d50340832",
      "resource": {
        "id": "Patient-1",
        "meta": {
          "profile": [
            "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-patient"
          ]
        },
        "resourceType": "Patient",
        "extension": [
          {
            "url": "http://hl7.org/fhir/us/core/StructureDefinition/us-core-race",
            "extension": [
              {
                "url": "ombCategory",
                "valueCoding": {
                  "system": "urn:oid:2.16.840.1.113883.6.238",
                  "code": "2106-3",
                  "display": "White"
                }
              },
              {
                "url": "text",
                "valueString": "White"
              }
            ]
          },
          {
            "url": "http://hl7.org/fhir/us/core/StructureDefinition/us-core-ethnicity",
            "extension": [
              {
                "url": "ombCategory",
                "valueCoding": {
                  "code": "2135-2",
                  "system": "urn:oid:2.16.840.1.113883.6.238",
                  "display": "Hispanic or Latino"
                }
              },
              {
                "url": "text",
                "valueString": "Hispanic or Latino"
              }
            ]
          }
        ]
      }
    }
  ]
}

```

Figure 3: JSON Bundle with QICorePatient Entry for Denominator Test Case

The same steps can be followed to add the procedure with reference to the [QICoreProcedure](#) profile structure definition. The denominator requires that the patient have a cataract procedure in a VSAC valueset which is completed and is performed during the measurement period. The procedure will also need to reference the patient in the first entry. Note that for QI-Core, the resource element includes a [meta](#) entry identifying the QI-Core profile.

Referencing the QICoreProcedure profile it can be determined that the patient is mapped to the subject data element referencing the QICorePatient id. The status data element has a required binding for value. Performed is a choice data element which can be populated either performedDateTime or performedPeriod which are FHIR datatypes dateTime and Period respectively. Code for the procedure is a complex datatype of type [CodeableConcept](#) and this datatype must be referenced to construct it correctly. The completed entry for the cataract procedure

```

{
  "fullUrl": "ed92ded2-b936-4257-9d49-0f2d8100dcde",
  "resource": {
    "resourceType": "Procedure",
    "id": "97527ae8-3314-4ee0-bc65-3a500596b450",
    "meta": {
      "profile": [
        "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-procedure"
      ]
    },
    "extension": [
      {
        "url": "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-recorded",
        "valueDateTime": "2021-04-05T12:35:00.000Z"
      }
    ],
    "status": "completed",
    "code": {
      "coding": [
        {
          "system": "http://snomed.info/sct",
          "code": "35717002",
          "display": "Discission of congenital cataract (procedure)"
        }
      ]
    },
    "text": "Discission of congenital cataract (procedure)"
  },
  "subject": {
    "reference": "Patient/e8b218bb-8a60-45a5-a73e-12bcc3976d0f"
  },
  "performedPeriod": {
    "start": "2022-06-05T10:00:00.000Z",
    "end": "2022-06-05T12:00:00.000Z"
  }
}

```

Figure 4: Completed Procedure Object

Since this test case bundle needs only the patient and procedure the JSON object array can be closed with a square bracket and the bundle object is closed with a curly bracket.

EXAMPLE OF A MEASURE DENOMINATOR EXCLUSION TEST CASE BUNDLE

The measure may include denominator exclusions. In this example, exclusions include valuesets of conditions that direct the test case into the denominator exclusion population so long as the condition is active, the code is a member of an exclusionary valueset for the member and the onset of the condition must predate the denominator procedure. Again, reference to the [QICore Condition profile](#) is helpful to determine the mapping, data types and cardinality.

```

{
  "fullUrl": "33e3e44c-9ace-11ed-9919-f340a01f594c",
  "resource": {
    "resourceType": "Condition",
    "id": "Condition-2",
    "meta": {
      "profile": [
        "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-condition"
      ]
    },
    "clinicalStatus": {
      "coding": [
        {
          "system": "http://terminology.hl7.org/CodeSystem/condition-clinical",
          "code": "active"
        }
      ]
    },
    "verificationStatus": {
      "coding": [
        {
          "system": "http://terminology.hl7.org/CodeSystem/condition-ver-status",
          "code": "confirmed"
        }
      ]
    },
    "category": [
      {
        "coding": [
          {
            "system": "http://terminology.hl7.org/CodeSystem/condition-category",
            "code": "encounter-diagnosis",
            "display": "Encounter Diagnosis"
          }
        ]
      }
    ],
    "severity": {
      "coding": [
        {
          "system": "http://snomed.info/sct",
          "code": "24484000",
          "display": "Severe (severity modifier)"
        }
      ]
    },
    "code": {
      "coding": [
        {
          "system": "http://hl7.org/fhir/sid/icd-10-cm",
          "code": "I25.5",
          "display": "Ischemic cardiomyopathy"
        }
      ]
    },
    "subject": {
      "reference": "Patient/Patient-1"
    },
    "onsetDateTime": "2022-01-02"
  }
},

```

Figure 5: Adding a QICore Condition Entry to the Bundle

TRANSACTION BUNDLES

Transaction bundles in FHIR are most useful when specific supporting operations are desired. For transaction bundles the type element is set to 'transaction'. An additional element must follow each resource with instruction to POST/PUT the resource.

```

1 {
2   "resourceType": "Bundle",
3   "id": "tests-ip-pos-CohortEpisodeProcedure-bundle",
4   "type": "transaction",
5   "entry": [
6     {
7       "fullUrl": "http://local/Procedure",
8       "resource": {
9         "resourceType": "Procedure",
10        "id": "CohortEpisodeProcedure-1",
11        "meta": {
12          "profile": [
13            "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-procedure"
14          ]
15        },
16        "extension": [
17          {
18            "url": "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-recorded",
19            "valueDateTime": "2021-04-05T12:35:00.000Z"
20          }
21        ],
22        "status": "completed",
23        "code": {
24          "coding": [
25            {
26              "system": "http://snomed.info/sct",
27              "code": "35717002",
28              "display": "Discission of congenital cataract (procedure)"
29            }
30          ],
31          "text": "Discission of congenital cataract (procedure)"
32        },
33        "subject": {
34          "reference": "Patient/CohortEpisodeProcedure"
35        },
36        "performedPeriod": {
37          "start": "2022-06-05T10:00:00.000Z",
38          "end": "2022-06-05T12:00:00.000Z"
39        }
40      },
41      "request": {
42        "method": "PUT",
43        "url": "Procedure/CohortEpisodeProcedure-1"
44      }
45    }
46  ]
47 }

```

Figure 6: Snippet of a Transaction Bundle

STEPS FOR CREATING A NEW TEST CASE IN MADIE

The following example details the construction of a FHIR bundle of type collection to be used in a MADiE test case for a Measure Denominator.

1. Sign into MADiE and select the intended measure to add test cases to with the View/Edit Control button.

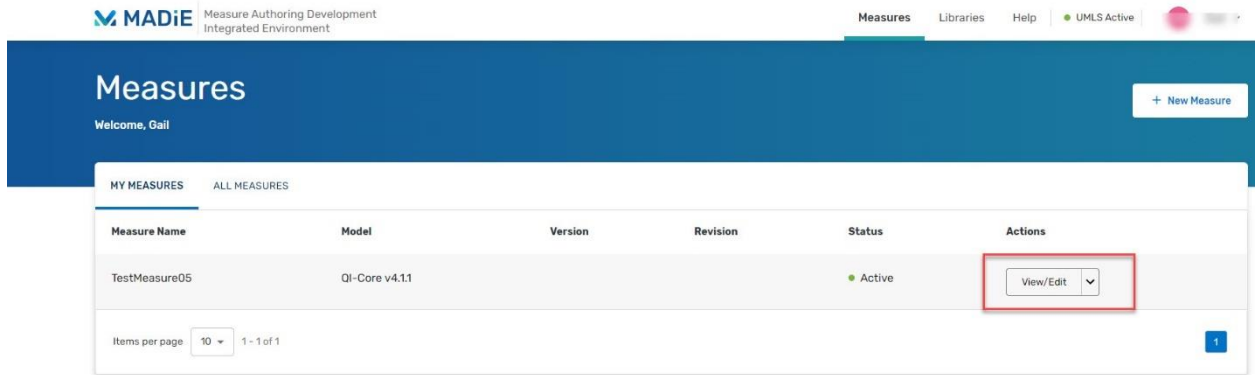


Figure 7: MADiE Measures Panel

2. With the intended measure open for edit, navigate to the Test Cases tab.

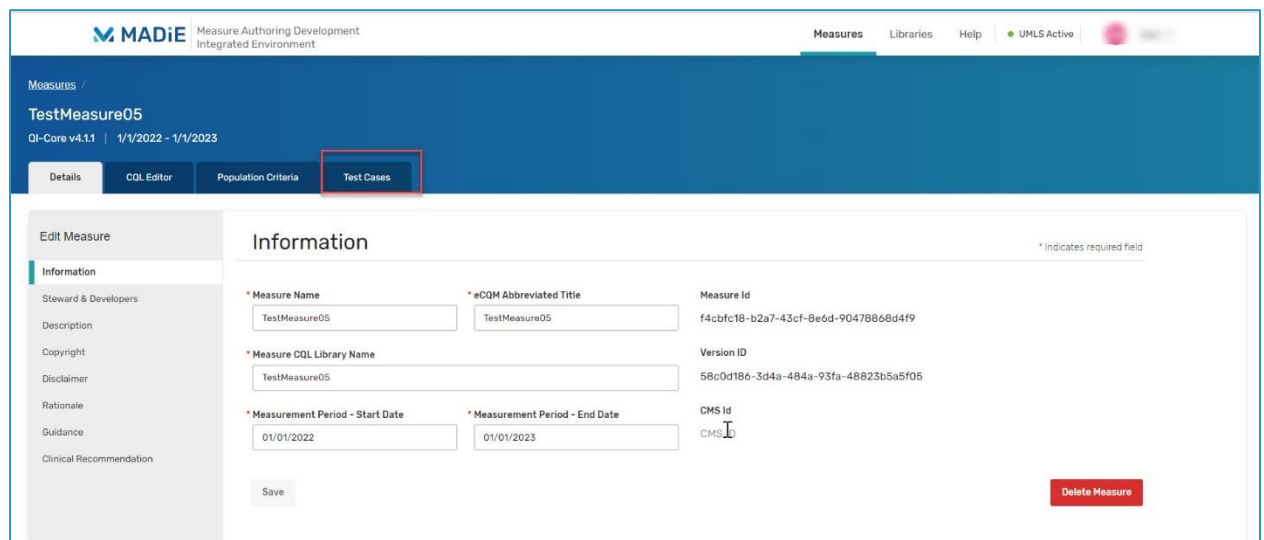


Figure 8 Measure Test Cases Tab Navigates to the Primary Test Case Window

3. The main Test Case window will open. Click the “+ New Test Case” control button to open the window. Enter the name and a description of the test case and click the Save control button.

Create Test Case [X]

*** Title**
DENOM-POS

Description
Positive denominator test case

Test Case Group
Start typing or select

Cancel Save

Figure 9: Creation of a New Test Case

4. Note that until JSON is entered and passes validation Status will display as Invalid.

MADiE Measure Authoring Development Integrated Environment

Measures Libraries Help UMLS Active GW Gail

Measures / TestMeasure05
QI-Core v4.1.1 | 1/1/2022 - 1/1/2023

Details CQL Editor Population Criteria **Test Cases**

An error occurred, please try again. If the error persists, please contact the help desk.

Passing Coverage + New Test Case Execute Test Cases

TITLE	SERIES	STATUS	
IP-POS		Invalid	Edit
DENOM-POS		Invalid	Edit

Terms of Use Privacy Policy Rules of Behavior

Figure 10: The Status of the Test Case is Invalid Until JSON is Entered

5. Enter the test case editor by clicking on the Edit button to the right of the status to enter the test case editor. Note the measure appears in the right window on the screen which may be useful for guidance. The left pane is a blank editor. This is where you can enter or paste your test case.

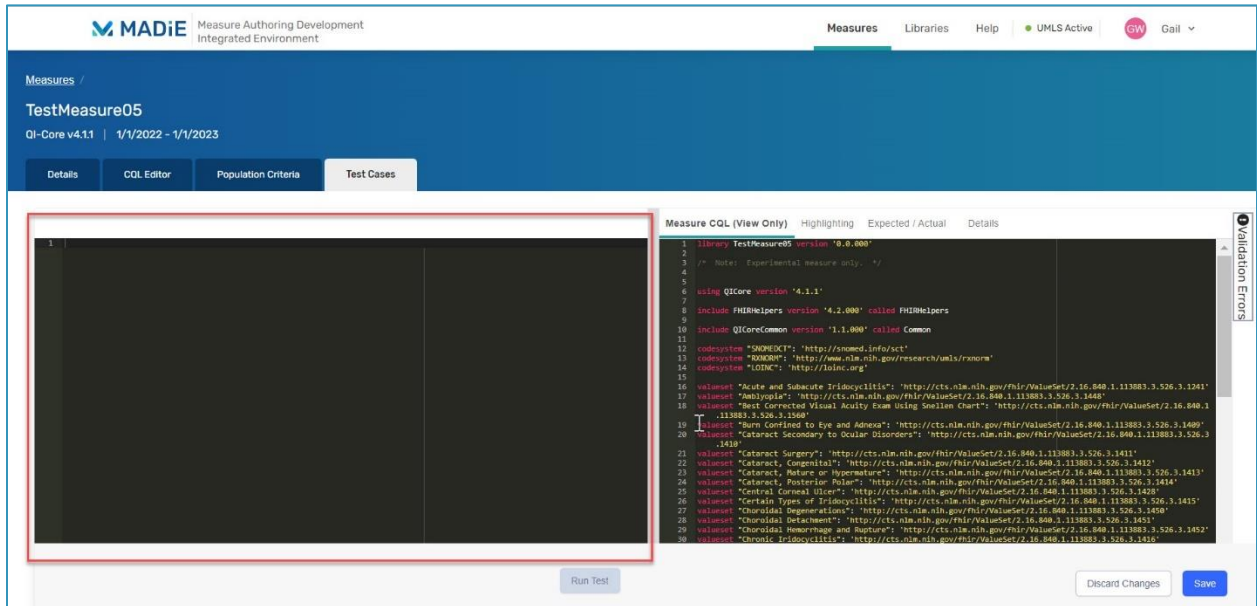


Figure 11: MADiE Test Case Editor

6. When the test case JSON is completed use the Save button on the bottom right of the window to save the changes. Note the Validation Errors icon to the top right of the measure pane. If there are errors, this will be red and will continue until the JSON contains no errors where it will revert to black and white. At that time, the Status of the Test Case will no longer be set to Invalid

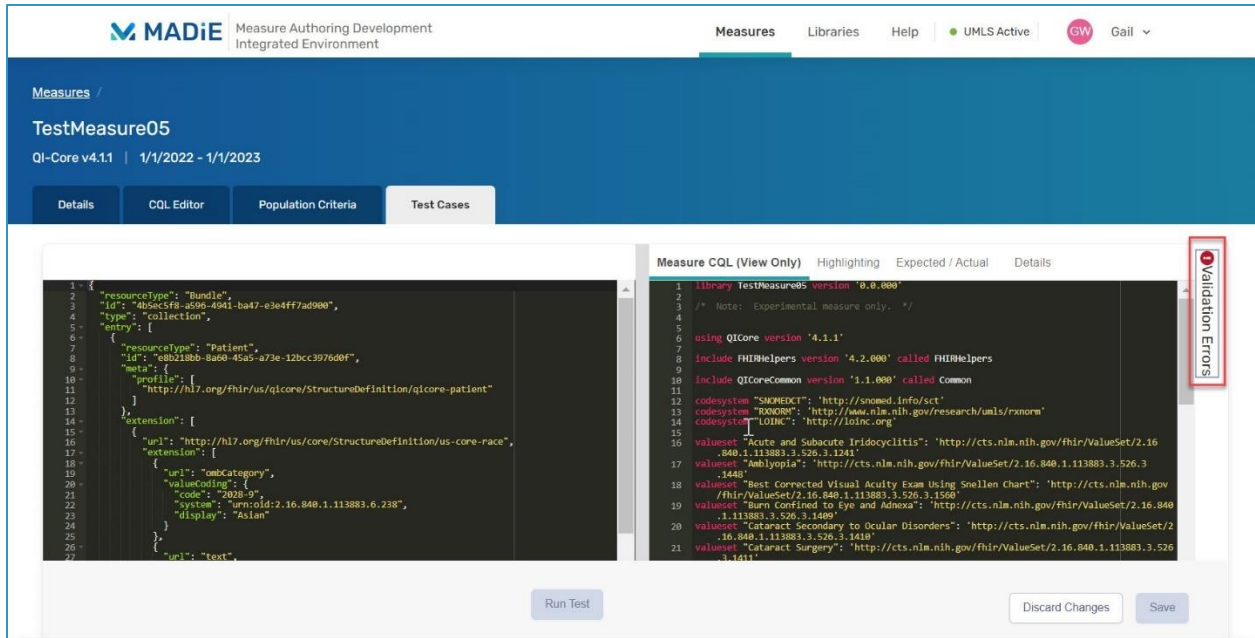


Figure 12: JSON Contains Errors and Does Not Pass Validation

7. Clicking on the Validation Errors icon will show the JSON errors which are preventing the test case from passing validation and need to be corrected. Each error must be corrected. Once they have click save and if there are no more errors the Validation icon will revert to black and white.

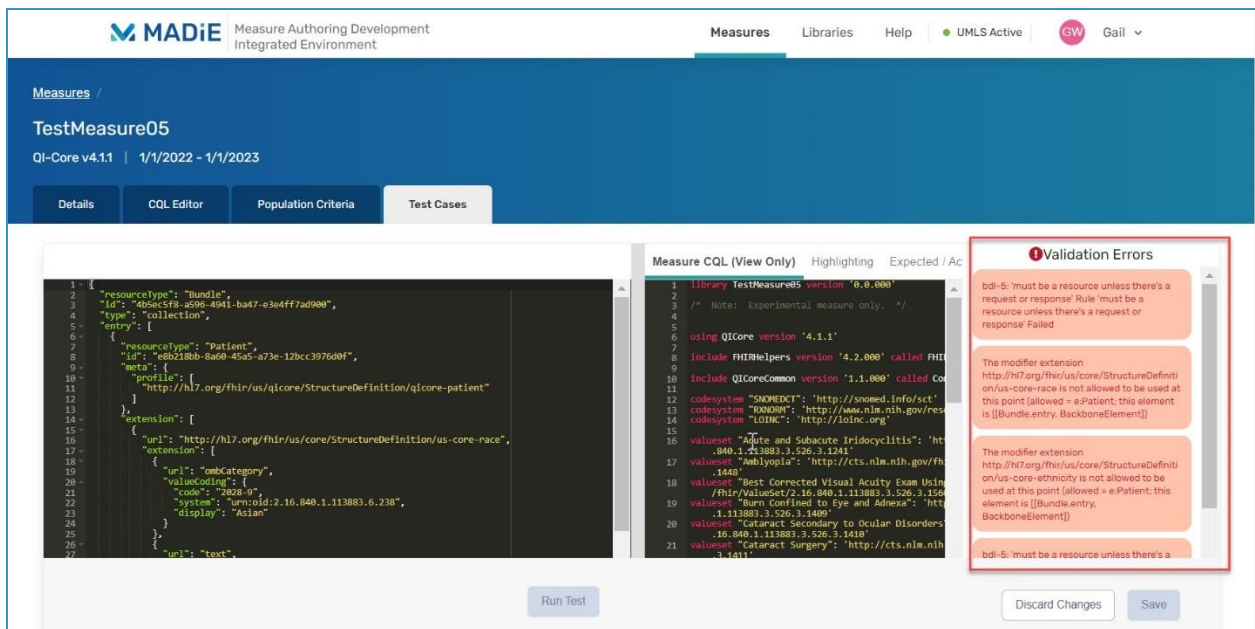


Figure 13: Validation Error Details Display by Clicking the Validation Icon

- When the JSON is error free, navigate to the Expected/Actual tab by clicking on it and enter the Expected results for the measure. Then click on the 'Run Test' control button in the bottom center of the window.

Note: Population values for the measure may vary depending upon the measure and its population definition and measure groups. The Actual results will be populated adjacent to the Expected values when the Test Case is run.

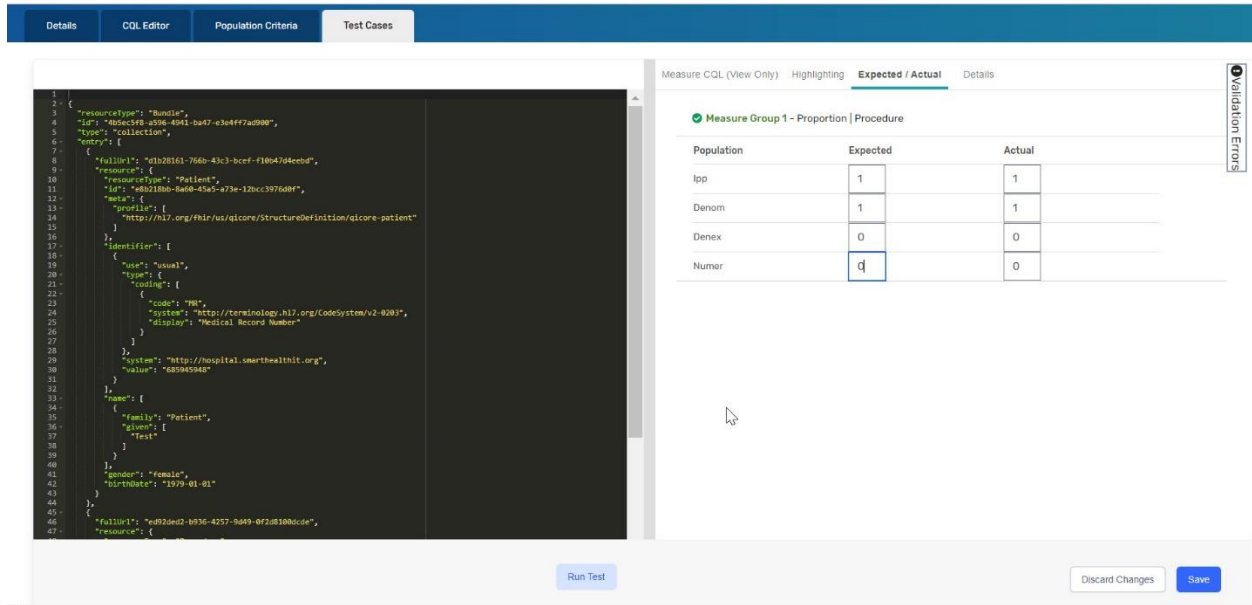


Figure 14: MADiE Test Case Expected and Actual Results

- Repeat Steps 1 through 6 to create additional test cases.

EXPORTING TEST CASES FROM BONNIE FHIR

Note: It is important to understand that test case exports from Bonnie FHIR use base FHIR resources and require updates for QI-Core profiles using the [QI-Core Implementation Guide](#) version 4.1.1. A future release of Bonnie FHIR is planned which will adapt the Patient resource in test cases for the QI-Core Patient profile. Currently Patient and other test case FHIR resources need manual adaptation to QI-Core profiles. Test cases left using base FHIR v4.0.1 resources may not meet the intent of measure.

- Log into Bonnie FHIR (<https://bonnie-fhir.healthit.gov/>).

2. Locate and open the measure for which test cases are wanted for export.
3. Identify the gear icon adjacent to measure on the right side of the screen to invoke the control buttons for Importing and Exporting.



Figure 15: BonnieFHIR Gear Icon

4. Click the Export Control Button once. Files will be saved to a local Download directory.



Figure 16: BonnieFHIR Export Control Button

5. Open the Download Directory with Windows Explorer and Extract the compressed export to location of choice while making note of location you have extracted to.

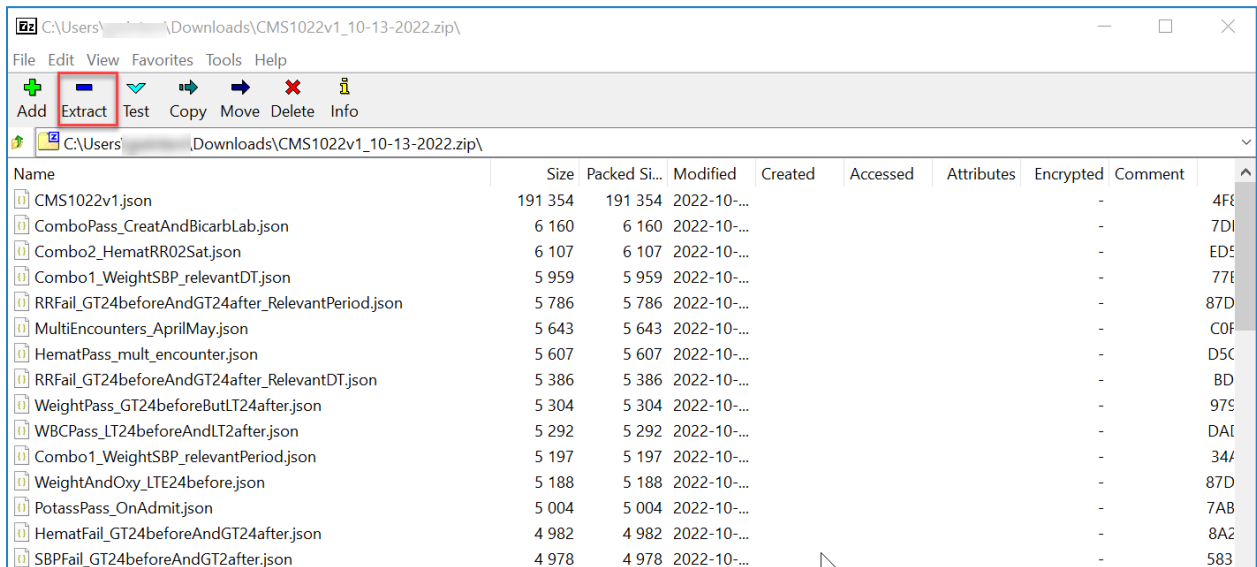


Figure 17: Downloaded BonnieFHIR.json Files

The extract consists of a JSON file for each test case. A larger, full measure bundle is also extracted (CMS1022v1.json in Figure 15); however, it is not needed for import to MADiE. Only the test case JSON files are useful for the purpose of importing test cases to MADiE.

6. Open the JSON file for the test case desired to be added to MADiE in a text editor such as Notepad++. Select the entire file and copy it to your windows buffer with the Select All and Copy to Clipboard (Copy if using Notepad) selections from the Edit Menu.
7. Proceed to the Section of this document ‘Creating Test Cases in MADiE’ and follow steps to create and paste the JSON into MADiE.
8. Note: When exporting test cases from BonnieFHIR into MADiE adjustments are required for QI-Core. This includes but is not limited to adding the meta information for the resource to point to the correct QI-Core profile. Please refer to the section of this document ‘Identifying the QI-Core Profile Official URL for Use In The Resource Meta’.

TIPS FOR USING QI-CORE PROFILE STRUCTURE DEFINITIONS

This section will reference the [QI Core Implementation Guide Version 4.1.1](#) with focus on the [QICorePatient profile](#).

When constructing Test Cases in JSON it will be of benefit to be able to understand profile structure definitions to be able to correctly meet the profile requirements.

IDENTIFYING THE QI-CORE PROFILE OFFICIAL URL FOR USE IN THE RESOURCE META

The Official URL for the QI-Core profile is available here:

<http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-patient>

The URL for the QI-Core profile is needed in the bundle resource meta.profile element. Each resource entry will need this information added if it is not already included as with the patient resource. The official URL can be located on the QI-Core Implementation Guide in the [Profile Section](#). Click on the QI-Core Profile of interest to open the Profile details.

QI-Core Profile	US Core Profile	Base Profile
QICoreMedicationAdministrationNotDone		MedicationAdministration
QICoreMedicationDispense		MedicationDispense
QICoreMedicationDispenseNotDone		MedicationDispense
QICoreMedicationNotRequested	USCoreMedicationRequest	MedicationRequest
QICoreMedicationRequest	USCoreMedicationRequest	MedicationRequest
QICoreMedicationStatement		MedicationStatement
QICoreNutritionOrder		NutritionOrder
QICoreObservation		Observation
QICoreObservationNotDone		Observation
	FHIR Vital Signs	Observation
	USCore Smoking Status	Observation
	USCore Laboratory Result	Observation
	USCore Pediatric BMI for Age	Observation
	USCore Pediatric Weight for Height	Observation
	USCore Pulse Oximetry	Observation
QICoreOrganization	USCoreOrganization	Organization
QICorePatient	USCorePatient	Patient
QICorePractitioner	USCorePractitioner	Practitioner
QICorePractitionerRole	USCorePractitionerRole	PractitionerRole
QICoreProcedure	USCoreProcedure	Procedure
QICoreProcedureNotDone	USCoreProcedure	Procedure
QICoreRelatedPerson		RelatedPerson
QICoreServiceNotRequested		ServiceRequest
QICoreServiceRequest		ServiceRequest
QICoreSpecimen		Specimen
QICoreSubstance		Substance
QICoreTask		Task
QICoreTaskNotDone		Task

Figure 18: QI-Core Profile List

QI-Core Implementation Guide
4.1.1 - STU 4.1.1 US

HL7
International

Home | QI Core Profiles | Patterns | Extensions | Terminology | Examples | Downloads | QDM to QI Core

Table of Contents > Artifacts Summary > QICoreProcedure

This page is part of the Quality Improvement Core Framework (v4.1.1: STU 4) based on FHIR R4. This is the current published version. For a full list of available versions, see the [Directory of published versions of IG](#).

Content | Detailed Descriptions | Mappings

10.43.1 Resource Profile: QICoreProcedure

Official URL: http://hl7.org/fhir/us/qicore/structureDefinition/qicore-procedure	Version: 4.1.1
Draft as of 2018-08-22	Computable Name: QICoreProcedure

Profile of Procedure for decision support/quality metrics. Defines the core set of elements and extensions for quality rule and measure authors.

Usage
To create an expression specifically requesting information that a procedure intentionally did not occur for a medical, patient or system reason, use the profile [QICore-ProcedureNotDone](#).

The Procedure and ProcedureNotDone profiles represent the positive and negative statements for a procedure. To ensure instances retrieved meet positive intent, applications should check the status element as illustrated in this example:

```
define Procedure:
  [Procedure] P
  where P.status in ( 'preparation', 'in-progress', 'on-hold', 'completed' )
```

The following example illustrates the use of the Procedure profile:

```
define "Flexible Sigmoidoscopy Performed":
  [Procedure: "Flexible Sigmoidoscopy"] FlexibleSigmoidoscopy
  where FlexibleSigmoidoscopy.status = 'completed'
  and FlexibleSigmoidoscopy.performed.toInterval() ends 5 years or less on or before end of "Measurement Period"
```

Figure 19: Downloaded BonnieFHIR.json Files

Note the official URL on the profile details page. This will be used in the resource meta.profile entry in the bundle. Note that each resource will need the appropriate QI-Core profile official URL.

```

1 {
2   "resourceType": "Bundle",
3   "id": "tests-ip-pos-CohortEpisodeProcedure-bundle",
4   "type": "transaction",
5   "entry": [
6     {
7       "fullUrl": "http://local/Procedure",
8       "resource": {
9         "resourceType": "Procedure",
10        "id": "CohortEpisodeProcedure-1",
11        "meta": {
12          "profile": [
13            "http://hl7.org/fhir/us/qicore/StructureDefinition/qicore-procedure"
14          ]
15        }
16      }
17    ]
18 }

```

Figure 20: Profile URL Usage in Test Case Bundle

PROFILE CONTENT

On a profile structure definition, the snapshot view includes all properties which are in a base resource and added profile elements and may be preferable to the implementer creating the bundle resource (test case).

Text Summary					Differential Table					Snapshot Table					Snapshot Table (Must Support)					All				
Name	Flags	Card.	Type	Description & Constraints																				
Patient		0..*	USCorePatientProfile	Information about an individual or animal receiving health care services																				
id		Σ 0..1	string	Logical Id of this artifact																				
meta		Σ 0..1	Meta	Metadata about the resource																				
implicitRules	?! Σ	0..1	uri	A set of rules under which this content was created																				
language		0..1	code	Language of the resource content Binding: CommonLanguages (preferred) Max Binding: AllLanguages: A human language.																				
text		0..1	Narrative	Text summary of the resource, for human interpretation																				
contained		0..*	Resource	Contained, inline Resources																				
Slices for extension		0..*	Extension	Extension Slice: Unordered, Open by value:url																				
us-core-race	S	0..1	(Complex)	US Core Race Extension URL: http://hl7.org/fhir/us/core/StructureDefinition/us-core-race																				
us-core-ethnicity	S	0..1	(Complex)	US Core ethnicity Extension URL: http://hl7.org/fhir/us/core/StructureDefinition/us-core-ethnicity																				
us-core-birthsex	S	0..1	code	Extension URL: http://hl7.org/fhir/us/core/StructureDefinition/us-core-birthsex Binding: Birth Sex (required): Code for sex assigned at birth																				
patient-religion		0..1	CodeableConcept	The patient's professed religious affiliations URL: http://hl7.org/fhir/StructureDefinition/patient-religion Binding: ReligiousAffiliation (extensible)																				
patient-birthPlace		0..1	Address	Place of Birth for patient URL: http://hl7.org/fhir/StructureDefinition/patient-birthPlace																				
patient-disability		0..*	CodeableConcept	Condition(s) limiting movement, senses, or activities URL: http://hl7.org/fhir/StructureDefinition/patient-disability																				
patient-nationality		0..*	(Complex)	Nationality URL: http://hl7.org/fhir/StructureDefinition/patient-nationality																				

Figure 21: QI-Core Patient Snippet

Profile content will include information pertaining to property names and description, cardinality, and flags. The following [definitions](#) may be helpful.

Column	Content
Name	The name of the element in the resource (manifests as XML element name or JSON or RDF property name). Some names finish with [x] - the meaning of this is discussed below. In addition, this column contains an icon that denotes the underlying type of the content. The icons are described below
Flags	A set of information about the element that impacts how implementers handle them. The flags are described below
Card.	Cardinality: the lower and upper bounds on how many times this element is allowed to appear in the resource
Type	The type of the element (hyperlinked to the definition of the type). Note that the type of the element has one of two meanings, depending on whether the element has defined children. If the element has children, then the element has an anonymous type that specializes the given type. If the element has no children, then the element has properties and children as specified by the nominated type
Description & Constraints	A description of the element, and details about constraints that are applied to it. Particularly, for coded elements, information about which codes can be used. The description comes from ElementDefinition.short

Figure 22: Content of a Profile Structure Definition

[Flags](#) contain information about the element which impacts how implementers handle them.

Important flags include:

?! : [Modifier elements](#) – a modifier element can change the interpretation of the resource. For example, a verification status.

I : Element is affected by [Constraints](#). Also known as invariants. In FHIR this can change the meaning of the resource. For example, to indicate that a medication was not administered.

S: [Must Support](#) : implementations that produce or consume resources SHALL provide support for the element in some meaningful way.

CARDINALITY

Cardinality can be used to determine if an element is mandatory and needs to be populated in a resource within the test case bundle. Use the profile structure definition to determine if the element is required.

FHIR specification only defines four cardinalities: 0..1, 0..*, 1..1, 1..*.

0..1 and 0..* indicate the element is not required. It is left to the implementer to determine what is useful information. The element may have no or 1 value (0..1) or no and more than 1 values (0..*)

1..1 indicates the element is required and may only have one value

1..* indicates element is required and may have more than one value

DATA TYPE

Resource element datatype can be either complex or primitive. Complex data types are supertypes with additional elements within them. For example, an Address or a Codeable Concept. Primitive data types do not have additional elements within them.

This indicates to author the JSON it may require additional understanding of complex data types.

To do this, click the data type hyperlink on the profile structure definition. A deeper understanding of primitive and complex data types in FHIR can be found [Foundation data types](#).

EXAMPLE TEST CASE BUNDLE: 'QI-COREV4.1.1 EXAMPLE BUNDLE VERSION 2'

DESCRIPTION

A measure is available for all users to view in MADiE with an included test case JSON bundle. This test case contains entries for many QI-Core (v4.1.1) profiles. The purpose of this test case JSON bundle is to provide users the framework for commonly used QI-Core v4.1.1 profiles that can be leveraged with their own test case development. The measure (QI-Corev4.1.1 Test Case Template) is a blank measure (no CQL and no Population Criteria) with one test case (QI-Corev4.1.1 Example Bundle version 2).

The following profiles are included:

- Adverse Event
- Allergy Intolerance (medication)
- Condition (2: Problem List, Diagnosis)
- Coverage
- DiagnosticReportLab
- Encounter (2: Inpatient, Ambulatory)
- Immunization
- ImmunizationNotDone

- Location
- Medication
- MedicationAdministration
- MedicationRequest
- Observation
- Organization (2: Health Provider, Payer)
- Patient
- Practitioner
- PractitionerRole
- Procedure
- ServiceNotRequested
- ServiceRequest
- Task

The QI-Core profiles examples in the JSON bundle are for guidance only and are not intended to fulfill any measure or testing requirement. Updates to reflect the specific details of your test case will be necessary. For example, test case(s) may require a different medication than what is provided in the profile example for MedicationAdministration. Measure developers may copy any of the profiles from the 'QI-Corev4.1.1 Example Bundle version 2' test case directly to test case(s) or use a source code editor (e.g., Notepad++ and Visual Studio Code) and make any updates needed to meet testing needs.

INSTRUCTIONS FOR USE

1. Select the All Measures table list and search for the measure 'QI-Corev4.1.1 Test Case Template' and open the measure by clicking "View" in the "Action" column corresponding to the measure. Once opened, click the Test Cases tab. Open the test case 'QI-Corev4.1.1 Example Bundle version 2' by clicking on "Select" in the "Action" column and selecting "View".

2. While viewing the test case, it may be helpful to copy the test case bundle and save as a JSON file for use with Visual Studio Code, Notepad++ or another editor that supports JSON.
3. Determine the QI-Core profile(s) available in the example test case that are needed for the test cases you are creating or editing. For example, the test cases may need the QI-Core Patient, Encounter, Condition and Procedure profiles.
4. Search the test case editor in MADiE or the external editor in use to locate the Resource example.
5. Copy the resource block and paste it into the measure test case bundle for your measure, beginning with the “{” above the “fullUrl” pairing above the resource entry until the closing “}” before the beginning of the next resource.

Note: Test cases with multiple resources must separate each resource with a comma after the closing “}”.

6. Edit the resource block to update the test case as needed for the measure.

Note: Edited values need to be QI-Core v4.1.1 compliant. For example, codes may need to be from a particular value set indicated by binding requirements.

7. After making changes ensure the bundle is valid JSON by using the editor feature or by pasting and saving in MADiE as a test case. If working in Visual Studio Code or a local editor, it may be helpful to format the JSON file and ensure it is syntactically correct before adding it to the MADiE test case bundle.