HL7® FHIR® DevDays

JUNE | 2024

The Future of CQL

Bryn Rhodes – Smile Digital Health

HL7 FHIR DevDays 2024 | Minneapolis, MN | June 10-13, 2024 | @HL7 | @FirelyTeam | #fhirdevdays | www.devdays.com

ORGANIZED BY

Who am I?

- Bryn Rhodes
- Director of Standards Strategy, Smile Digital Health

History of CQL

A Stroll Down Memory Lane...

- 2011-2012: Real-time Clinical Decision Support
- Rapidly changing rules
  - Authoring and maintenance
  - Governance
  - Effective periods
- Real-time evaluation
  - What data was needed
  - Initial vs subsequent data needs
- Variability based on setting and provider preference
  - Selective evaluation
  - Configuration

Solution

- Context, platform, and model independent representation for expression logic
- Supported static analysis to determine data requirements
- Reasoning on data requirements

Why Not HQMF?

- HQMF – Health Quality Measures Format
  - Quality Measures Specification Format in use since 2009/2010
- Various reasons
  - Expression language and model were baked into the same structure
  - Insufficient expressive power
  - Inadequate for data requirements analysis

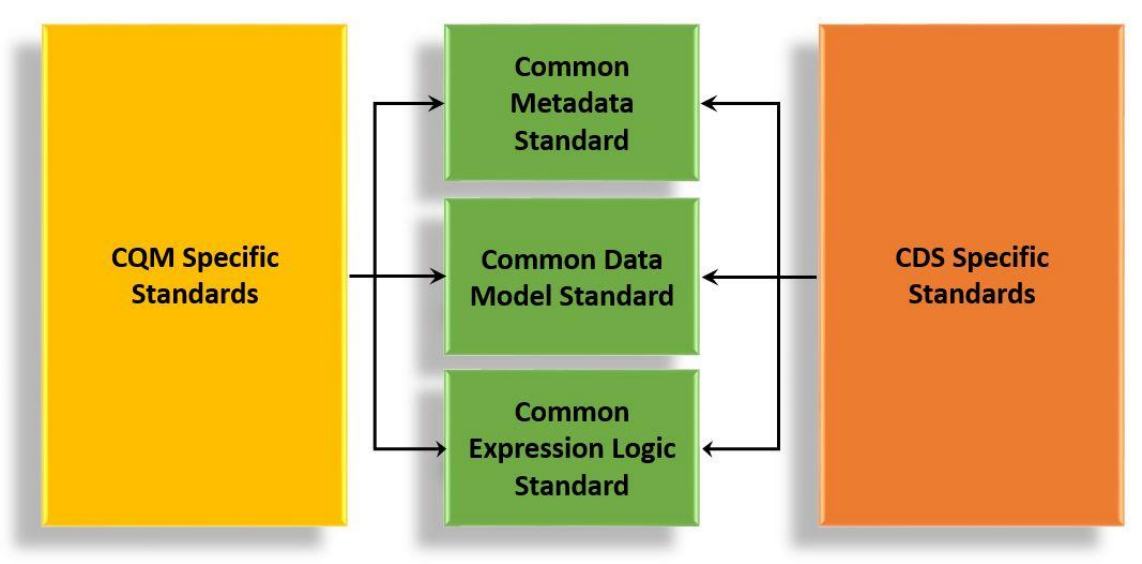
Health eDecisions

- CDS Knowledge Artifact Specification
- Virtual Medical Record (vMR)
- Decision Support Service (DSS)

Clinical Quality Framework (CQF)

Clinical Quality Measurement (CQM) and Clinical Decision Support (CDS) use different representations for the same concepts

Clinical Quality Framework initiative focused on the alignment/harmonization of standards between the two domains



Clinical Quality Framework

- Harmonization of Health Quality Artifact Expression Logic
  - https://www.hl7.org/implement/standards/product_brief.cfm?product_id=359
- Clinical Quality Common Metadata Conceptual Model
  - https://www.hl7.org/implement/standards/product_brief.cfm?product_id=391
- HL7 DAM: Health Quality Improvement
  - https://www.hl7.org/implement/standards/product_brief.cfm?product_id=378
- Characteristics of a Value Set Definition (Not CQF, but super important!!)
  - https://www.hl7.org/implement/standards/product_brief.cfm?product_id=437

Clinical Reasoning Module

Home | Getting Started | Documentation | Data Types | ***Resource Types*** | Terminologies | Artifacts | Implementation Guides

Clinical Reasoning

This page is part of the FHIR Specification (v5.0.0: R5 - STU). This is the current published version. For a full list of available versions, see the Directory of published versions. Page versions: **R5** R4B R4 R3

Work Group Clinical Decision Support & Clinical Quality Information	Standards Status: Informative

14.0 Clinical Reasoning

14.0.1 Introduction

The Clinical Reasoning module provides resources and operations to enable the representation, distribution, and evaluation of clinical knowledge artifacts such as clinical decision support rules, quality measures, public health indicators, order sets, clinical protocols, and evidence summaries. In addition, the module describes how expression languages can be used throughout the specification to provide dynamic capabilities.

Clinical Reasoning involves the ability to represent and encode clinical knowledge in a very broad sense so that it can be integrated into clinical systems. This encoding may be as simple as controlling whether or not a particular section of an order set appears based on the conditions that a patient has, or it may be as complex as representing the care pathway for a patient with multiple conditions.

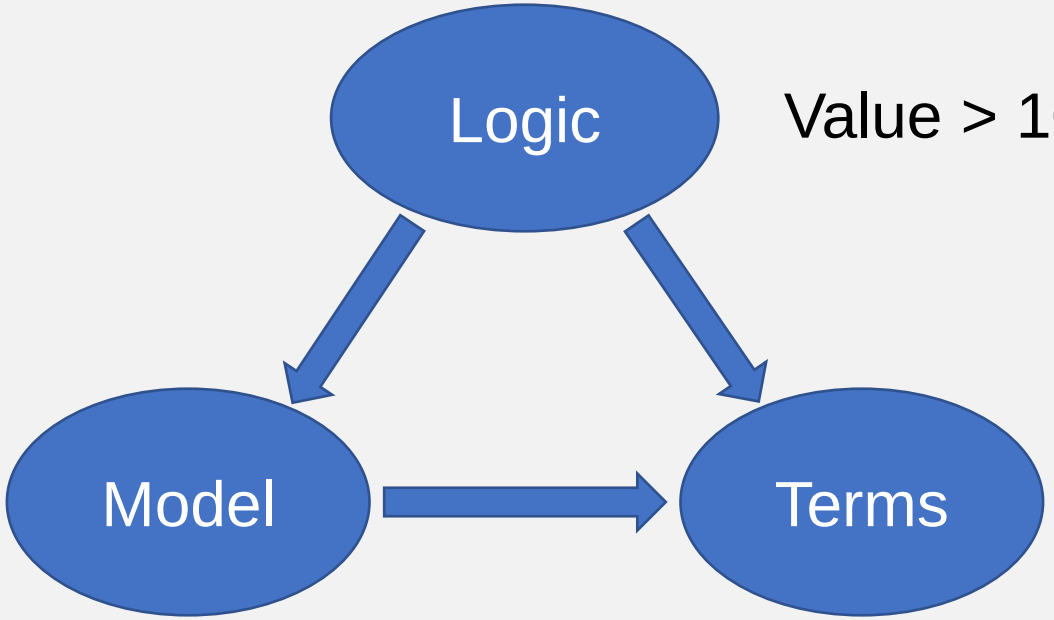
https://hl7.org/fhir/clinicalreasoning-module.html

MetadataResource

This resource interface extends the CanonicalResource interface. The following resources implement this interface:

- ActivityDefinition
- ChargeItemDefinition
- Citation
- CodeSystem
- ConceptMap
- ConditionDefinition
- EventDefinition
- Evidence
- EvidenceReport
- EvidenceVariable
- Library
- Measure
- NamingSystem
- ObservationDefinition
- PlanDefinition
- Questionnaire
- SpecimenDefinition
- ValueSet

Components of Sharing Logic



Value > 100

Encounter
Medication
Observation

SNOMED-CT
LOINC
RxNorm

Definitions:
SNOMED CT – Systematized Nomenclature of Medicine – Clinical Terms
LOINC – Logical Observation Identifiers Names and Codes

What is Clinical Quality Language?

- A domain-specific query language designed to represent and share clinical logic
- It is:
  - Clinically focused
  - Platform/technology independent
  - Computationally complete
  - Functional

What is Clinical Quality Language NOT?

- A rule language
- A measure language
- A guideline language
- A terminology definition language

Specification

CQL Clinical Quality Language Release 1

HL7 International

Documentation | Author's Guide | Developer's Guide | CQL Reference | Grammar | ELM | Examples | Tests | Downloads | Version History

This page is part of the Clinical Quality Language Specification (v1.5.2: Normative - Normative) based on FHIR R4. This is the current published version. For a full list of available versions, see the Directory of published versions

Clinical Decision Support Work Group	Maturity Level: N	Standards Status: Normative

Clinical Quality Language (CQL)

HL7 Standard: Clinical Quality Language Specification, Release 1 Mixed Normative/Trial-Use (CQL 1.5)

HL7 Mixed Normative/STU Specification

Clinical Quality Language (CQL) is a high-level, domain-specific language focused on clinical quality and targeted at measure and decision support artifact authors.

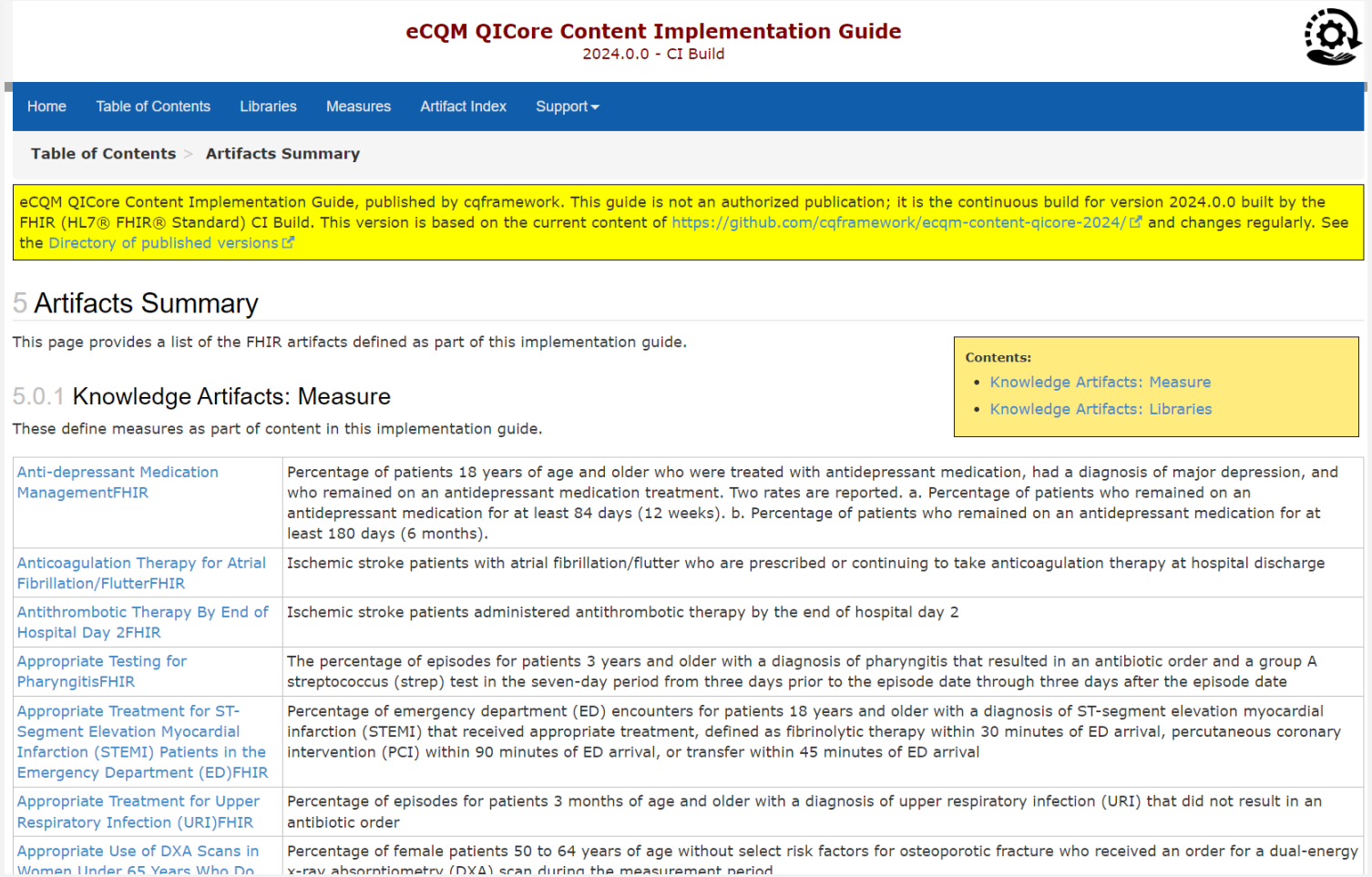
In addition, this specification describes a machine-readable canonical representation called Expression Logical Model (ELM) targeted at implementations and designed to enable sharing of clinical knowledge.

ANSI

http://cql.hl7.org

Current Projects

Quality Measurement and Reporting



eCQM QICore Content Implementation Guide
2024.0.0 - CI Build

Home | Table of Contents | Libraries | Measures | Artifact Index | Support

Table of Contents > **Artifacts Summary**

eCQM QICore Content Implementation Guide, published by cqframework. This guide is not an authorized publication; it is the continuous build for version 2024.0.0 built by the FHIR (HL7® FHIR® Standard) CI Build. This version is based on the current content of https://github.com/cqframework/ecqm-content-qicore-2024/ and changes regularly. See the Directory of published versions

5 Artifacts Summary

This page provides a list of the FHIR artifacts defined as part of this implementation guide.

Contents:
- Knowledge Artifacts: Measure
- Knowledge Artifacts: Libraries

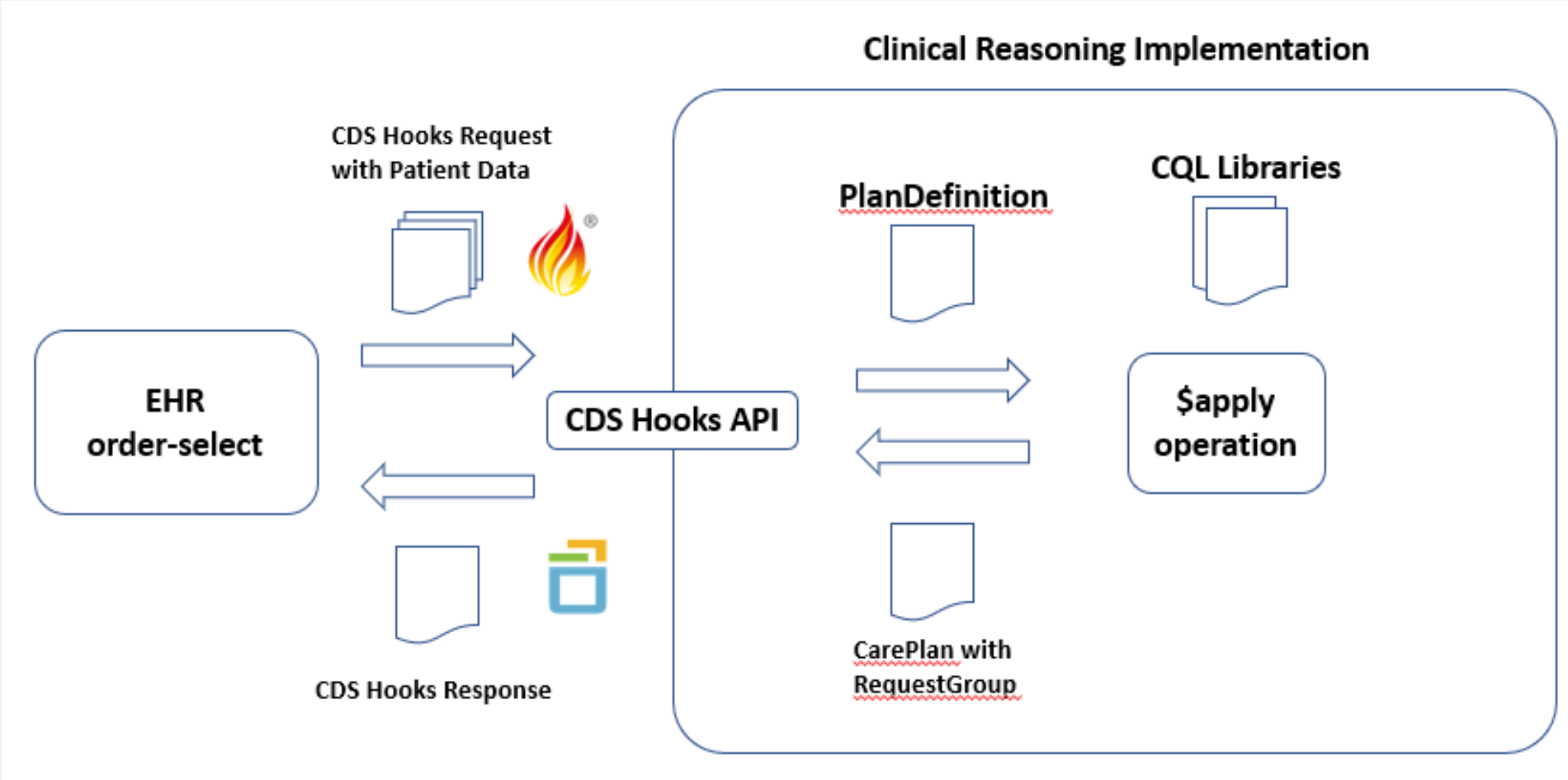
5.0.1 Knowledge Artifacts: Measure

These define measures as part of content in this implementation guide.

Anti-depressant Medication ManagementFHIR	Percentage of patients 18 years of age and older who were treated with antidepressant medication, had a diagnosis of major depression, and who remained on an antidepressant medication treatment. Two rates are reported. a. Percentage of patients who remained on an antidepressant medication for at least 84 days (12 weeks). b. Percentage of patients who remained on an antidepressant medication for at least 180 days (6 months).
Anticoagulation Therapy for Atrial Fibrillation/FlutterFHIR	Ischemic stroke patients with atrial fibrillation/flutter who are prescribed or continuing to take anticoagulation therapy at hospital discharge
Antithrombotic Therapy By End of Hospital Day 2FHIR	Ischemic stroke patients administered antithrombotic therapy by the end of hospital day 2
Appropriate Testing for PharyngitisFHIR	The percentage of episodes for patients 3 years and older with a diagnosis of pharyngitis that resulted in an antibiotic order and a group A streptococcus (strep) test in the seven-day period from three days prior to the episode date through three days after the episode date
Appropriate Treatment for ST-Segment Elevation Myocardial Infarction (STEMI) Patients in the Emergency Department (ED)FHIR	Percentage of emergency department (ED) encounters for patients 18 years and older with a diagnosis of ST-segment elevation myocardial infarction (STEMI) that received appropriate treatment, defined as fibrinolytic therapy within 30 minutes of ED arrival, percutaneous coronary intervention (PCI) within 90 minutes of ED arrival, or transfer within 45 minutes of ED arrival
Appropriate Treatment for Upper Respiratory Infection (URI)FHIR	Percentage of episodes for patients 3 months of age and older with a diagnosis of upper respiratory infection (URI) that did not result in an antibiotic order
Appropriate Use of DXA Scans in Women Under 65 Years Who Do	Percentage of female patients 50 to 64 years of age without select risk factors for osteoporotic fracture who received an order for a dual-energy x-ray absorptiometry (DXA) scan during the measurement period

https://build.fhir.org/ig/cqframework/ecqm-content-qicore-2024/artifacts.html

Decision Support



http://hl7.org/fhir/uv/cpg/documentation-approach-10-mechanisms-of-integration.html

Clinical Practice Guidelines

DRAFT DRAFT DRAFT DRAF **WHO SMART Guidelines - Measles Immunization** DRAFT DRAFT DRAFT

0.1.0 - ci-build

Table of Contents > **Indices** > **Artifact Index**

WHO SMART Guidelines - Measles Immunization, published by WHO. This guide is not an authorized publication; it is the continuous build for version 0.1.0 built by the FHIR (HL7® FHIR® Standard) CI Build. This version is based on the current content of https://github.com/WorldHealthOrganization/smart-immunizations-measles/ and changes regularly. See the Directory of published versions

5.1 Artifact Index

This page provides a list of the FHIR artifacts defined as part of this implementation guide.

Contents:

5.1.0.1 Knowledge Artifacts: Activity Definitions

These define activities that can be performed as part of content in this implementation guide.

IMMZ.D2.DT.Measles.Contraindication	Provide measles immunization
IMMZ.D2.DT.Measles.Eval	Provide measles immunization
IMMZ.D2.DT.Measles.MR	Provide measles immunization

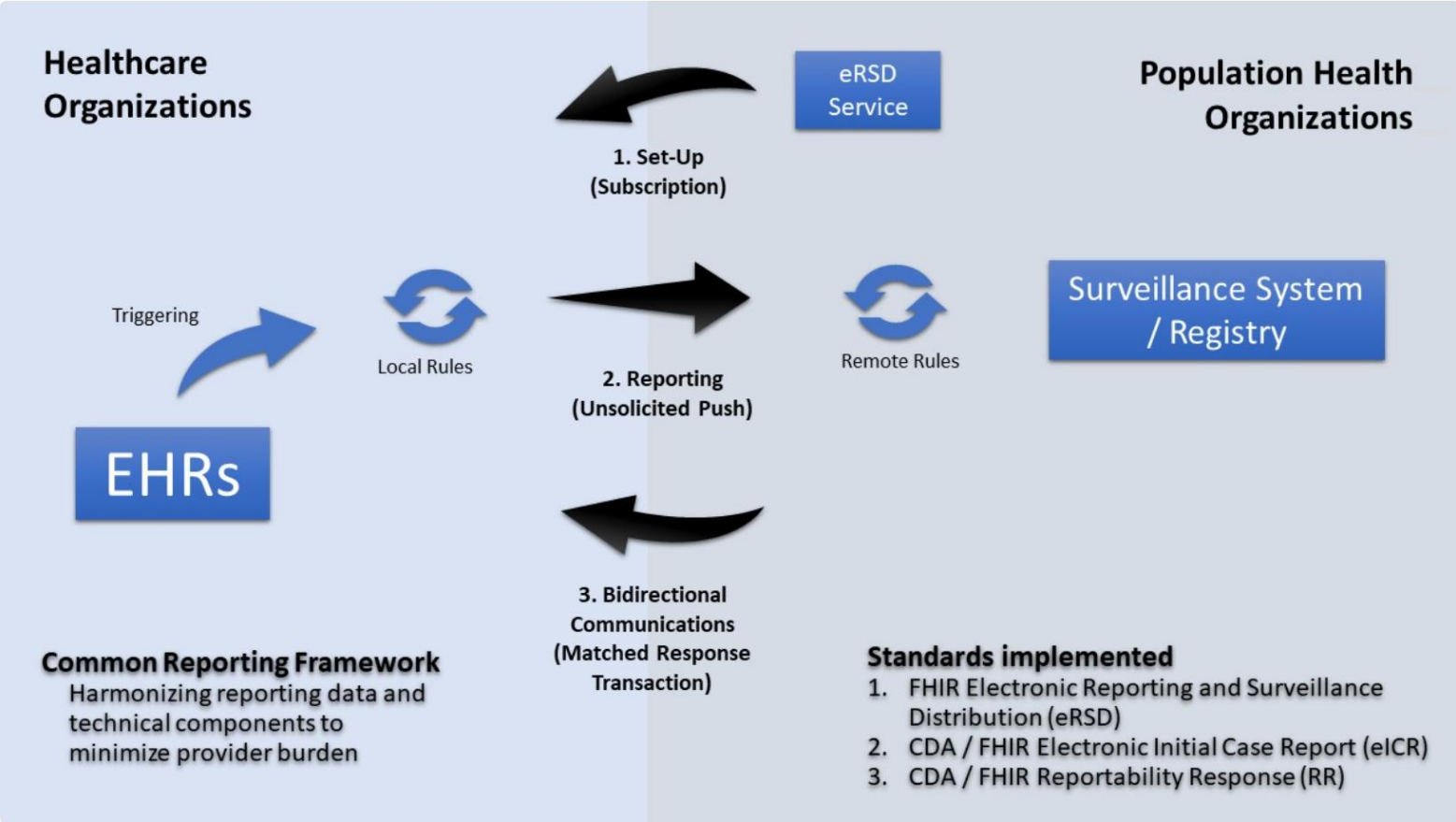
5.1.0.2 Knowledge Artifacts: Plan Definitions

These define workflows, rules, strategies, or protocols as part of content in this implementation guide.

IMMZ.D2.DT.Measles Low Transmission	If the child or patient has not been given MCV1 (at 12 months) and MCV2 (between 15-18 months) vaccination
IMMZ.D2.DT.Measles MCV Dose 0	If the child or patient has not been given MCV0 between 6 and 9 months
IMMZ.D2.DT.Measles MCV Dose 0	If the child or patient has not been given a supplementary dose
IMMZ.D2.DT.Measles Ongoing Transmission	If the child or patient has not been given MCV1 (at 9 months) and MCV2 (between 15-18 months) vaccination
IMMZ.D5.DT.Measles.Contraindication	Check contraindications for Measles vaccine
IMMZ.DT.Immunization Strategy	Provide vaccinations according to the recommended schedule

https://build.fhir.org/ig/WorldHealthOrganization/smart-immunizations-measles/artifacts.html

Public Health Reporting (for Local Rules)



Prior Authorization Questionnaires

Clinical Quality Framework Common FHIR Assets (US-Based)

0.1.0 - CI Build USA

IG Home | Artifact Index | Support

Table of Contents > Artifacts Summary

Clinical Quality Framework Common FHIR Assets (US-Based), published by Clinical Quality Framework. This guide is not an authorized publication; it is the continuous build for version 0.1.0 built by the FHIR (HL7® FHIR® Standard) CI Build. This version is based on the current content of https://github.com/cqframework/cqf-us/ and changes regularly. See the Directory of published versions

4 Artifacts Summary

This page provides a list of the FHIR artifacts defined as part of this implementation guide.

Contents:

- Knowledge Artifacts: Libraries
- Structures: Questionnaires
- Other

4.0.1 Knowledge Artifacts: Libraries

These define logic, asset collections and other libraries as part of content in this implementation guide.

Library/CumulativeMedicationDuration
Library/FHIRHelpers
Library/GMTPInitialExpressions
Library/MBODAInitialExpressions
Library/UPPARFInitialExpressions
Library/USCoreCommon
Library/USCoreElements
Library/USCoreTests

https://build.fhir.org/ig/cqframework/cqf-us/artifacts.html

Implementations

Open Source Implementations

- Java Translator and Engine
  - https://github.com/cqframework/clinical_quality_language
- JavaScript Engine
  - https://github.com/cqframework/cql-execution
- Community Projects
  - https://github.com/cqframework/clinical_quality_language/wiki/Community-Projects

Digital Quality Implementers Community

- Current focus on demonstrating engine conformance
- Starting point – CQL Test Suite from the spec
  - https://cql.hl7.org/tests.html
- Tests Runner
  - https://github.com/cqframework/cql-tests-runner
- Community for engine implementation teams to collaborate
- Identify and correct
  - Engine disparities
  - Testing coverage gaps
  - Test/specification discrepancies

https://www.ncqa.org/digital-quality-implementers-community/

vNext

Clinical Quality Language vNext

- https://confluence.hl7.org/display/CDS/Clinical+Quality+Language
- Themes
  - Conformance
  - Feedback and Features
  - STU to Normative Candidates

What We Will NOT Do

- No Breaking Changes
- CQL has been a normative specification for almost 5 years
- Multiple production engines and systems rely on this stability

Conformance

- Need for engines to demonstrate conformance
- Specification defines a set of capabilities
  - Formalize this description (it's narrative right now)
    - Looking at making use of "Application Feature Framework"
- So that...
  - Engines can advertise what capabilities they *support*
  - Artifacts can advertise what capabilities they *require*
  - Potentially support capability *negotiation*

Feedback and Uplift

- Publication uplift
- Errata and clarifications
  - definitions for compile-time vs run-time
  - clarify runtime type testing semantics for choice types
  - clarify implicit conversion semantics
- Formalize error reporting
- Formalize JSON serialization of ELM

Promote Conformance Language from Using CQL

HL7 International

Using CQL with FHIR
1.0.0 - STU1

HL7 FHIR

Home | Authoring | Integrating | FHIR Artifacts | Downloads | Version History

Table of Contents > **Home**

This page is part of the Using CQL with FHIR Implementation Guide (v1.0.0: STU1) based on FHIR (HL7® FHIR® Standard) R4. This is the current published version. For a full list of available versions, see the Directory of published versions

1 Home

Official URL: **http://hl7.org/fhir/uv/cql/ImplementationGuide/hl7.fhir.uv.cql**	*Version*: **1.0.0**
Active as of 2024-04-20	*Computable Name*: CQL
Other Identifiers: OID:2.16.840.1.113883.4.642.40.37	

1.1 Summary

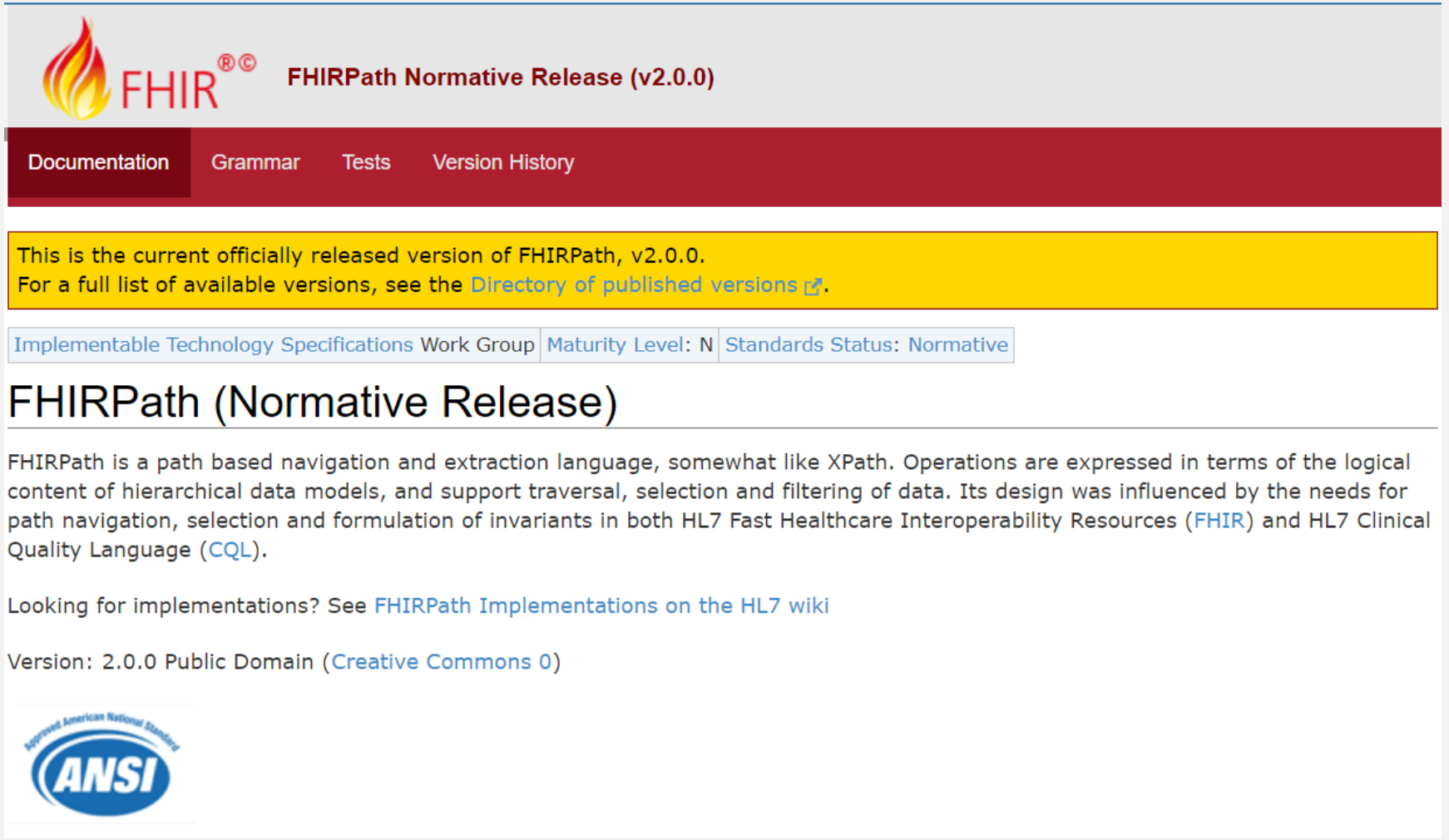
Clinical Quality Language (CQL) is a clinically-focused query language that can be used to express logic in a broad range of healthcare use cases, including clinical decision and cognitive support, public health and quality reporting, computable clinical guidelines, research trial eligibility, and many others. Several implementation guides have been published that include conformance criteria related to the use of CQL in these contexts. This implementation guide is the result of extracting common aspects of that support to a universal realm, broadly applicable implementation guide that supports the use of CQL with Fast Healthcare Interoperability Resources (FHIR). Future versions of those implementation guides should consider referencing the conformance and guidance established here.

Note that although this is a first release of this implementation guide, the content has been balloted, published, reviewed, implemented, and refined over many years as part of the Quality Measure Implementation Guide, FHIR Clinical Guidelines, Quality Improvement Profile (QI-Core), and the Canonical Resource Management Infrastructure IG.

- Summary
- Scope of Use
- How to read this Guide
- Acknowledgements
- References
- Dependencies
- Cross Version Analysis
- Global Profiles
- IP Statements

https://hl7.org/fhir/uv/cql/

Incorporate FHIRPath updates



https://hl7.org/fhirpath/

Default Comparison Precision

- Default comparison precision
  - Currently defaults to `millisecond` if unspecified

```
define "Right Mastectomy Procedure":
  [Procedure: "Unilateral Mastectomy Right"] P
    where P.performed.toInterval() ends on or before end of "Measurement Period"
```

- If data coming in is specified to a different precision than the Measurement Period, edge cases result in unknown due to the comparison running down to the millisecond
- Specifying comparison precision addresses this, however, would be nice to be able to specify a default comparison precision (say `minutes`)

Additional Terminology Operators

- Can currently say `x in ValueSet`, but not `ValueSet contains x`
- Proposing `~in` and `~contains` to address this
  - `in` and `contains` but with *equivalent* rather than *equality* semantics

Better Support For Profile-Informed Authoring

- “Pure-FHIR” approach
  - `using FHIR`
  - `Patient.extension(‘ethnicity’).value`
- “Profile-informed” approach
  - `using QICore`
  - `Patient.ethnicity`
  - relies on “mappings” inferred from StructureDefinitions and expanded by the translator

Considering “Accessors”

```
define fluent function birthsex(patient Patient):
  (singleton from (
    patient.extension E where E.url = 'http://hl7.org/fhir/us/core/StructureDefinition/us-core-birthsex'
  )).value as FHIR.code

define "Patient Birth Sex Is Male (refined)":
  Patient P
    where P.birthsex() = 'M'
```

```
define accessor birthsex(patient Patient):
  (singleton from (
    patient.extension E where E.url = 'http://hl7.org/fhir/us/core/StructureDefinition/us-core-birthsex'
  )).value as FHIR.code

define "Patient Birth Sex Is Male (accessors)":
  Patient P
    where P.birthsex = 'M'
```

Considering Type Definitions

- CQL does not currently have a mechanism to define types
- Relies exclusively on ModelInfo to supply type information
- Considering a `define type` statement
  - Would eliminate the difference between “models” and “libraries”
    - i.e. `include QICore` instead of `using QICore`
  - Would significantly simplify implementation of profile-informed authoring
  - Would better enable multi-version support

Implementer Support

- Considering approaches to simplifying implementation
  - Opportunities to reduce the number of types of ELM nodes?
  - Opportunities to simplify representation of ELM
    - e.g. currently multiple ways to represent system function calls
  - Opportunities to use capabilities to facilitate implementation
    - e.g. authors could restrict their use of features to the set of features expected in a target implementation
  - Opportunities to move some “system-defined” functions into “standard libraries”
    - i.e. move some things out of the spec and into standard libraries that are themselves written in CQL

Considering a Function Type

- ELM already has this capability
  - it's used throughout the iteration operations such as .Filter, .Return, etc.
- No facility in the language to accept functions as an argument
- Considering a `function` type to support this

Improved Support for Library Organization and Usage

- Allowing multiple versions of the same library to be included
  - so long as they have different local identifiers
- Allowing aliasing of included declarations
  - i.e. “renaming” a declaration from an included library
- Allowing binding of parameters in included libraries
  - i.e. using the value of a parameter in my library as the value of another parameter in a library I’ve included

Balloting Plan

- Phase I – Publishing uplift
  - ASCIIDoc to Markdown
  - Side-by-side review to ensure no inadvertent changes due to uplift
- Phase II – Errata and clarifications
  - Apply all outstanding errata and clarifications
  - Considering a snapshot at this point
- Phase III – Additional features
  - STU Ballot in January 2025
- Phase IV – Normative Edition 2
  - Possibly January 2026

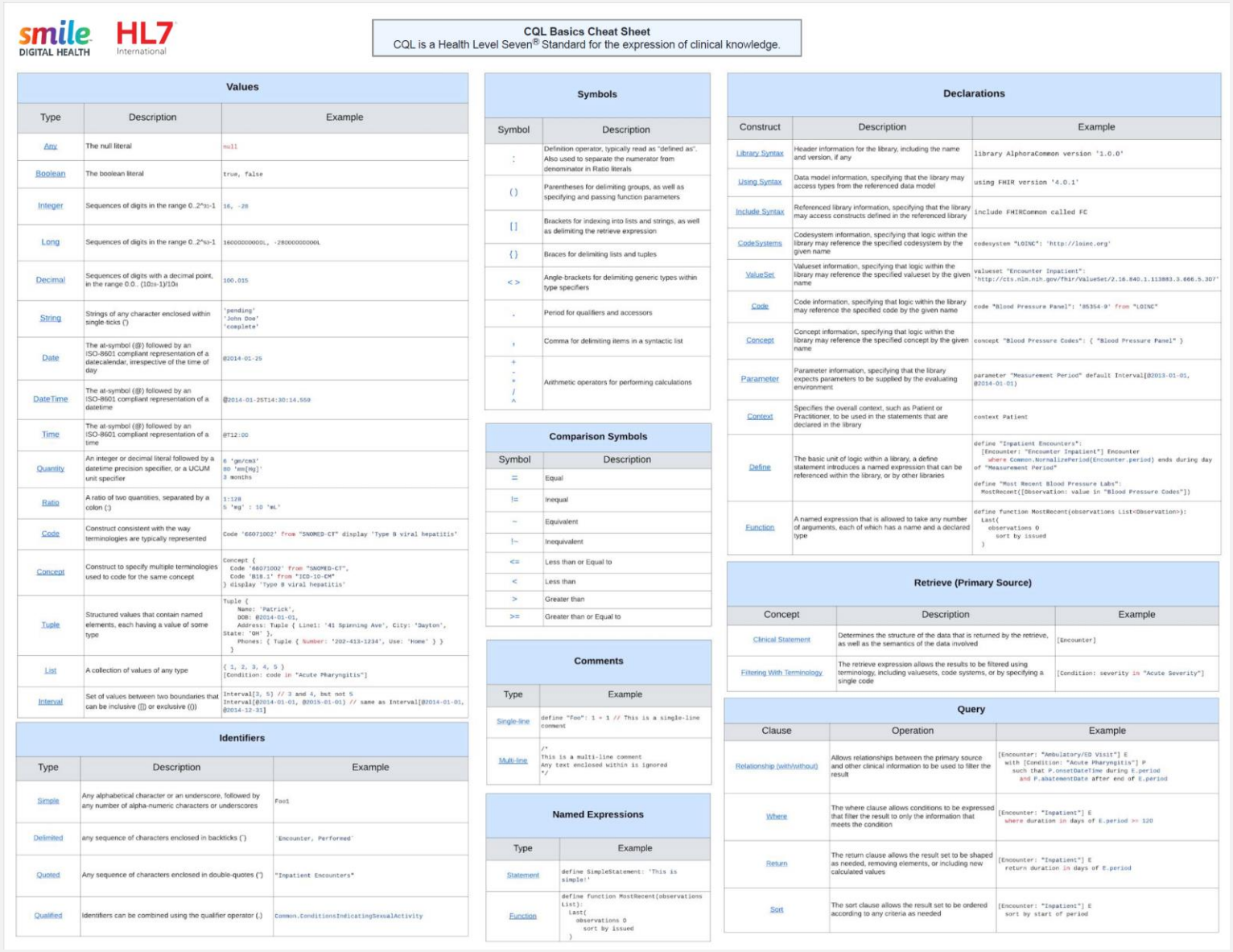
Community

Community Support

- Education and Capacity
  - Considering a “CQL Certified” for authors
- Tooling
  - VSCode Plugin Enhancements
    - Debugger
    - Code completion
    - Goto definition/references
    - Visualization tools
  - Better support for CQL in JavaScript environments
- Knowledge Repositories

CQL Cheat Sheet

https://bit.ly/cql-cheat-sheet



smile DIGITAL HEALTH HL7 International

CQL Basics Cheat Sheet
CQL is a Health Level Seven® Standard for the expression of clinical knowledge.

Values

Type	Description	Example
Any	The null literal	`null`
Boolean	The boolean literal	`true, false`
Integer	Sequences of digits in the range 0..2^31-1	`16, -28`
Long	Sequences of digits in the range 0..2^63-1	`1600000000L, -2800000000L`
Decimal	Sequences of digits with a decimal point, in the range 0.0..(10^31-1)/10^8	`100.015`

String	Strings of any character enclosed within single ticks (')	`'pending'`
		`'John Doe'`
		`'complete'`

Date	The at-symbol (@) followed by an ISO-8601 compliant representation of a date/calendar, irrespective of the time of day	`@2014-01-25`
DateTime	The at-symbol (@) followed by an ISO-8601 compliant representation of a date/time	`@2014-01-25T14:30:14.559`
Time	The at-symbol (@) followed by an ISO-8601 compliant representation of a time	`@T12:00`

Quantity	An integer or decimal literal followed by a date/time precision specifier, or a UCUM unit specifier	`6 'gm/cm3'`
		`80 'mm[Hg]'`
		`3 months`

Ratio	A ratio of two quantities, separated by a colon (:)	`1:128` `5 'mg' : 10 'mL'`
Code	Construct consistent with the way terminologies are typically represented	`Code '66071002' from "SNOMED-CT" display 'Type B viral hepatitis'`

Concept	Construct to specify multiple terminologies used to code for the same concept	`Concept {`
		`Code '66071002' from "SNOMED-CT",`
		`Code 'B18.1' from "ICD-10-CM"`
		`} display 'Type B viral hepatitis'`

Tuple	Structured values that contain named elements, each having a value of some type	`Tuple {`
		`Name: 'Patrick',`
		`DOB: @2014-01-01,`
		`Address: Tuple { Line1: '41 Spinning Ave', City: 'Dayton', State: 'OH' },`
		`Phones: { Tuple { Number: '202-413-1234', Use: 'Home' } }`
		`}`

List	A collection of values of any type	`{ 1, 2, 3, 4, 5 }` `[Condition: code in "Acute Pharyngitis"]`
Interval	Set of values between two boundaries that can be inclusive ([]) or exclusive (())	`Interval[3, 5] // 3 and 4, but not 5` `Interval[@2014-01-01, @2015-01-01) // same as Interval[@2014-01-01, @2014-12-31]`

Identifiers

Type	Description	Example
Simple	Any alphabetical character or an underscore, followed by any number of alpha-numeric characters or underscores	`Foo1`
Delimited	any sequence of characters enclosed in backticks (`)	`` `Encounter, Performed` ``
Quoted	Any sequence of characters enclosed in double-quotes (")	`"Inpatient Encounters"`
Qualified	Identifiers can be combined using the qualifier operator (.)	`Common.ConditionsIndicatingSexualActivity`

Symbols

Symbol	Description
:	Definition operator, typically read as "defined as". Also used to separate the numerator from denominator in Ratio literals
()	Parentheses for delimiting groups, as well as specifying and passing function parameters
[]	Brackets for indexing into lists and strings, as well as delimiting the retrieve expression
{}	Braces for delimiting lists and tuples
< >	Angle-brackets for delimiting generic types within type specifiers
.	Period for qualifiers and accessors
,	Comma for delimiting items in a syntactic list
+ - * / ^	Arithmetic operators for performing calculations

Comparison Symbols

Symbol	Description
=	Equal
!=	Inequal
~	Equivalent
!~	Inequivalent
<=	Less than or Equal to
<	Less than
>	Greater than
>=	Greater than or Equal to

Comments

Type	Example
Single-line	`define "Foo": 1 + 1 // This is a single-line comment`

Multi-line	`/*`
	`This is a multi-line comment`
	`Any text enclosed within is ignored`
	`*/`

Named Expressions

Type	Example
Statement	`define SimpleStatement: 'This is simple!'`

Function	`define function MostRecent(observations List<Observation>):`
	`Last(`
	`observations O`
	`sort by issued`
	`)`

Declarations

Construct	Description	Example
Library Syntax	Header information for the library, including the name and version, if any	`library AlphoraCommon version '1.0.0'`
Using Syntax	Data model information, specifying that the library may access types from the referenced data model	`using FHIR version '4.0.1'`
Include Syntax	Referenced library information, specifying that the library may access constructs defined in the referenced library	`include FHIRCommon called FC`
CodeSystems	Codesystem information, specifying that logic within the library may reference the specified codesystem by the given name	`codesystem "LOINC": 'http://loinc.org'`
ValueSet	Valueset information, specifying that logic within the library may reference the specified valueset by the given name	`valueset "Encounter Inpatient": 'http://cts.nlm.nih.gov/fhir/ValueSet/2.16.840.1.113883.3.666.5.307'`
Code	Code information, specifying that logic within the library may reference the specified code by the given name	`code "Blood Pressure Panel": '85354-9' from "LOINC"`
Concept	Concept information, specifying that logic within the library may reference the specified concept by the given name	`concept "Blood Pressure Codes": { "Blood Pressure Panel" }`
Parameter	Parameter information, specifying that the library expects parameters to be supplied by the evaluating environment	`parameter "Measurement Period" default Interval[@2013-01-01, @2014-01-01)`
Context	Specifies the overall context, such as Patient or Practitioner, to be used in the statements that are declared in the library	`context Patient`

Define	The basic unit of logic within a library, a define statement introduces a named expression that can be referenced within the library, or by other libraries	`define "Inpatient Encounters":`
		`[Encounter: "Encounter Inpatient"] Encounter`
		`where Common.NormalizePeriod(Encounter.period) ends during day of "Measurement Period"`
		`define "Most Recent Blood Pressure Labs":`
		`MostRecent([Observation: value in "Blood Pressure Codes"])`

Function	A named expression that is allowed to take any number of arguments, each of which has a name and a declared type	`define function MostRecent(observations List<Observation>):`
		`Last(`
		`observations O`
		`sort by issued`
		`)`

Retrieve (Primary Source)

Concept	Description	Example
Clinical Statement	Determines the structure of the data that is returned by the retrieve, as well as the semantics of the data involved	`[Encounter]`
Filtering With Terminology	The retrieve expression allows the results to be filtered using terminology, including valuesets, code systems, or by specifying a single code	`[Condition: severity in "Acute Severity"]`

Query

Clause	Operation	Example

Relationship (with/without)	Allows relationships between the primary source and other clinical information to be used to filter the result	`[Encounter: "Ambulatory/ED Visit"] E`
		`with [Condition: "Acute Pharyngitis"] P`
		`such that P.onsetDateTime during E.period`
		`and P.abatementDate after end of E.period`

Where	The where clause allows conditions to be expressed that filter the result to only the information that meets the condition	`[Encounter: "Inpatient"] E` `where duration in days of E.period >= 120`
Return	The return clause allows the result set to be shaped as needed, removing elements, or including new calculated values	`[Encounter: "Inpatient"] E` `return duration in days of E.period`
Sort	The sort clause allows the result set to be ordered according to any criteria as needed	`[Encounter: "Inpatient"] E` `sort by start of period`

Learn More/Get Involved

- VSCode Plugin User's Guide
  - https://github.com/cqframework/vscode-cql/wiki/User-Guide
- Cooking with CQL Monthly Sessions
  - https://ecqi.healthit.gov/events/4206
- Zulip CQL Stream
  - https://chat.fhir.org/#narrow/stream/179220-cql
- CQL vNext Project
  - https://confluence.hl7.org/display/CDS/Clinical+Quality+Language

Contact

- You can find / reach me here:
  - bryn@smiledigitalhealth.com
  - https://www.linkedin.com/in/brynrhodes/
  - https://github.com/brynrhodes

Q&A

Feedback

- Include link and/or QR code to collect feedback on your talk and DevDays

https://www.devdays.com/feedback/

ORGANIZED BY