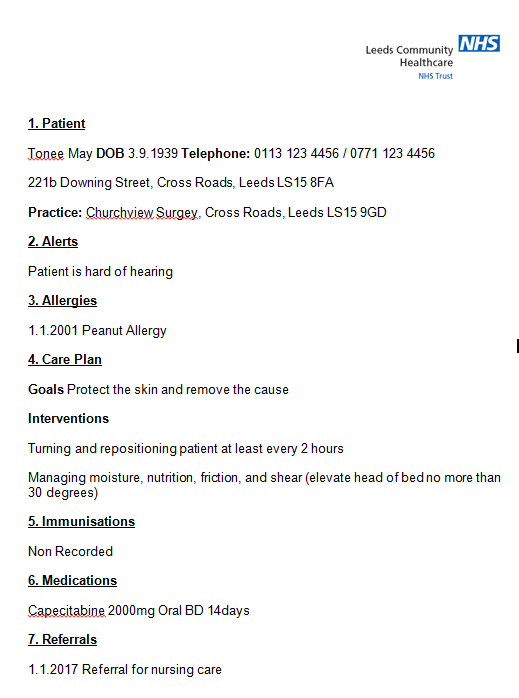
# Creating a Composition Profile

Date: 10th March 2017

Author: Kevin Mayfield

Ref: CareConnect FHIR Profiles: <http://www.interopen.org/candidate-profiles/care-connect/>

This tutorial will use this sample letter as a base for the composition. It is loosely based on the data interchanges between health and social care providers who contribute to the Leeds Care Record.

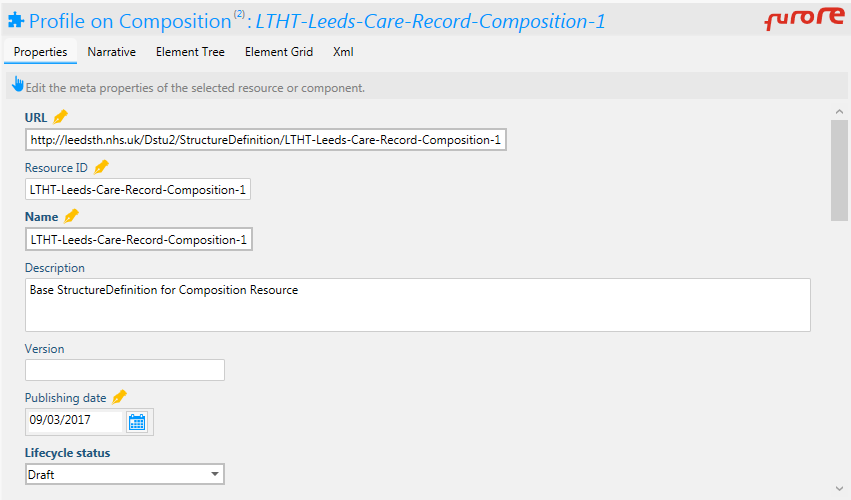


The letter consists of seven sections which may realise correspond to FHIR Resource’s; we will come back to this latter in this tutorial.

## Properties

In Furore Forge create a new profile based on Composition. On the properties tab:

* Add the url for the resource. E.g. <https://fhir.leedsth.nhs.uk/Dstu2/StructureDefinition/LTHT-Leeds-Care-Record-Composition-1>
  + Dstu2 shows the FHIR version which was used for the profile
  + Number at the end of the url allows for multiple versions of the profile.
* Add a Resource Id and Name. In this example we have used **LTHT-Leeds-Care-Record-Composition-1** which matches the url of the resource.
* Add copyright and publisher



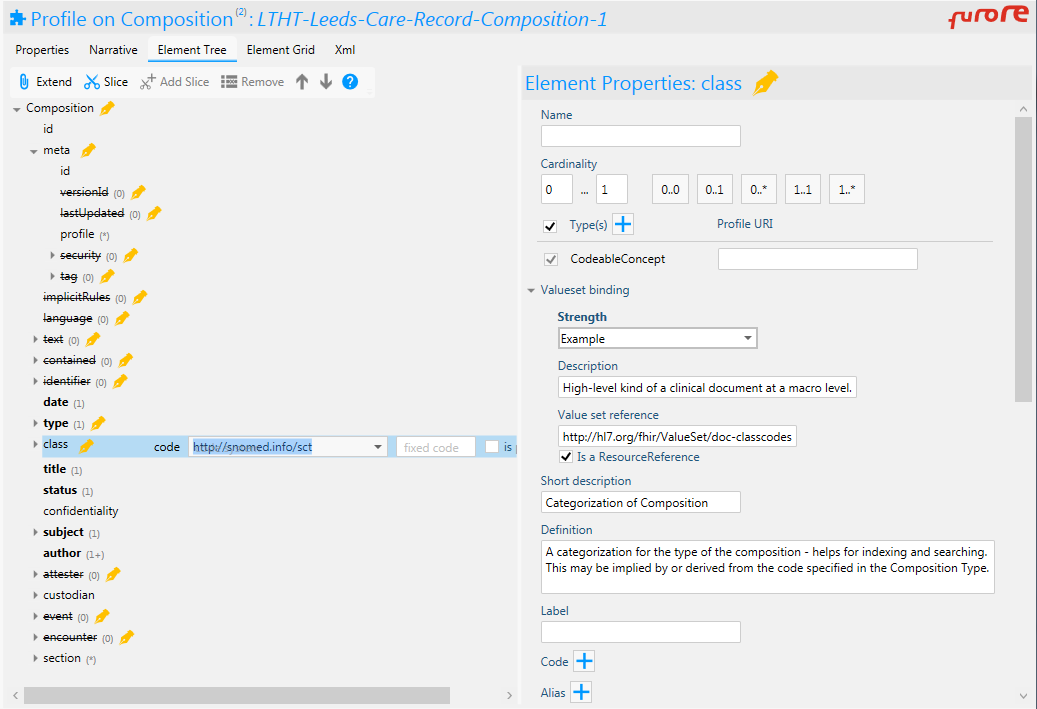
## Constrain the Profile

Next we need to constrain properties we don’t need in the profile. Set the cardinality to 0..0 for:

* implicitRules
* language
* test
* contained
* identifier
* attester
* event
* encounter

Similarly for the type and class entries set the code type to **http://snomed.info/sct** and the cardinality to 0..0 for

* id
* text
* coding
  + id
  + version
  + userSelected



The constraints we’ve done so far match the GP Connect Composition profile (<https://fhir.nhs.net/StructureDefinition/gpconnect-carerecord-composition-1> ) and we will continue this and reuse the same ValueSets fopr type and class. These are:

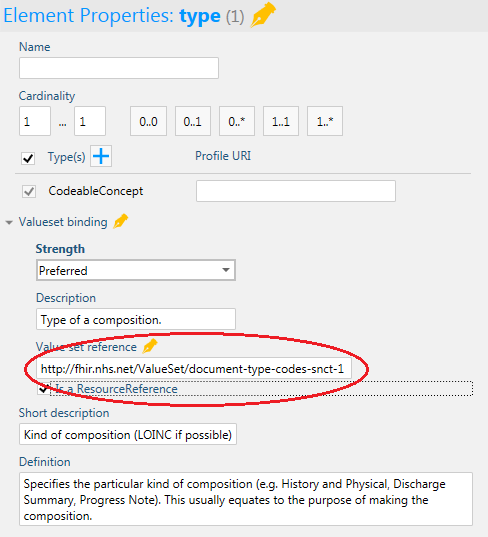
type: <https://fhir.nhs.net/ValueSet/document-type-codes-snct-1>

([SNOMED RefSet=999000391000000109](http://browser.ihtsdotools.org/?perspective=full&conceptId1=999000391000000109&edition=uk-edition&release=v20161001&server=https://prod-browser-exten.ihtsdotools.org/api/snomed&langRefset=900000000000508004))

class: <https://fhir.nhs.net/ValueSet/care-setting-codes-snct-1>

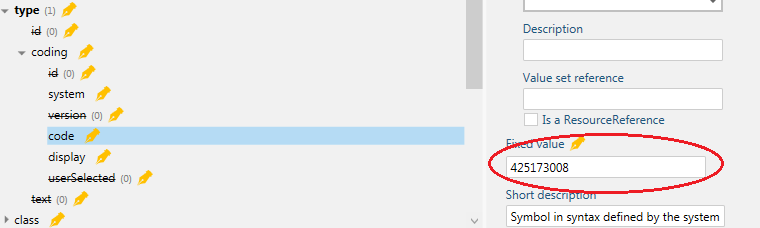
([SNOMED RefSet=999000381000000107](http://browser.ihtsdotools.org/?perspective=full&conceptId1=999000381000000107&edition=uk-edition&release=v20161001&server=https://prod-browser-exten.ihtsdotools.org/api/snomed&langRefset=900000000000508004))

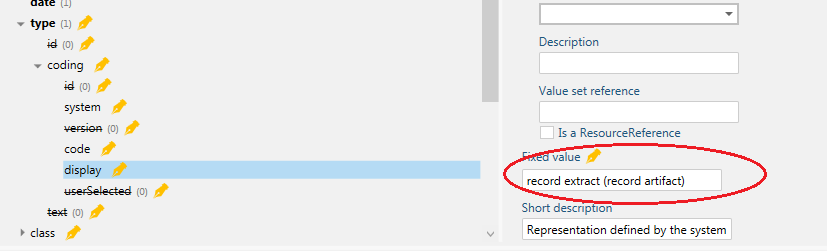
Add these valueset to the properties of type and class



The contents of these value sets can be found in ITK Reference Set documentation, GP Connect documentation or use the SNOMED links above.

The type code in this instance will be *425173008 record extract (record artifact)*, these can be added to the type.





The code and display value will change depending on which organisation creates the composition, e.g. For Mental Health this could be *708168004 Mental health service (qualifier value).*

## Section - Slicing

Like CDA, FHIR Documents ideally contain both a human readable and structured information. The human readable portion is normally coded as html within the **Composition** with links to structured versions of the same information.

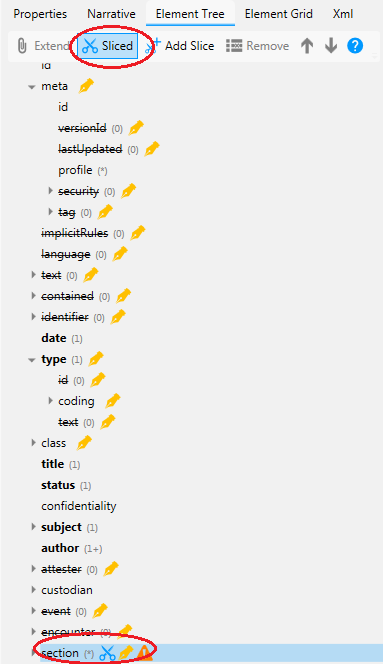
Firstly we will split up the letter into sections and assign a discriminator to differentiate each section. The sections are:

|  |  |
| --- | --- |
| Section Title | SNOMED CT Code and Display Term |
| Patient | 886731000000109 Patient Demographics |
| Alerts | 886931000000107 Safety Alerts |
| Allergies | 886921000000105 Allergies and adverse reactions |
| Care Plan | 887201000000105 Plan and Requested Actions |
| Immunisations | Nil - see note below |
| Medications | 933361000000108 Medications and Devices |
| Referrals | 886721000000107 Referral Details |

SNOMED Codes have been taken from RefSet: *999001721000000100 Standards for the clinical structure and content of patient records*

This reference set doesn’t have a suitable entry for immunisations for the purposes of this tutorial *304250009 Immunization Finding* will be used.

To split up the section entry of the profile, select *section* and then click on the *slice* button. The button should then change to *sliced* and section will have a pair of scissors which indicates it is sliced.



Next we need to specify how the sections are broken up. We’ve assigned a code to each section and so this can be used.



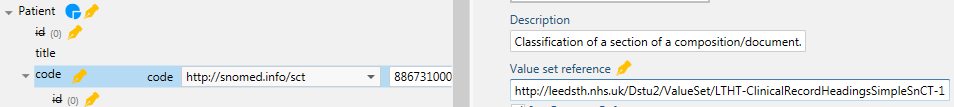
Ensure section is highlighted and then click on the Add Slice button. In the name box add Patient.

Remove the following sections by changing cardinality to 0..0

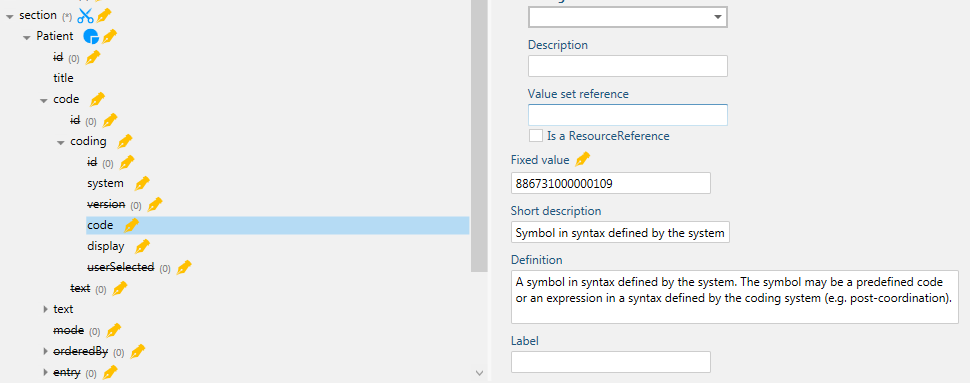
* Id
* code.coding.id
* code.coding.version
* code.coding.userSelected
* code.text
* mode
* orderdedBy
* entry
* emptyReason
* section

Change coding cardinality to 0..1 and set the code system to SNOMED

Change the Value set reference to one pointing to the *999001721000000100 Standards for the clinical structure and content of patient records.* In the below this is <http://leedsth.nhs.uk/Dstu2/ValueSet/LTHT-ClinicalRecordHeadingsSimpleSnCT-1>



Set the coding.code to have a fixed value of *886731000000109* and coding.display to *Patient Demographics*

**

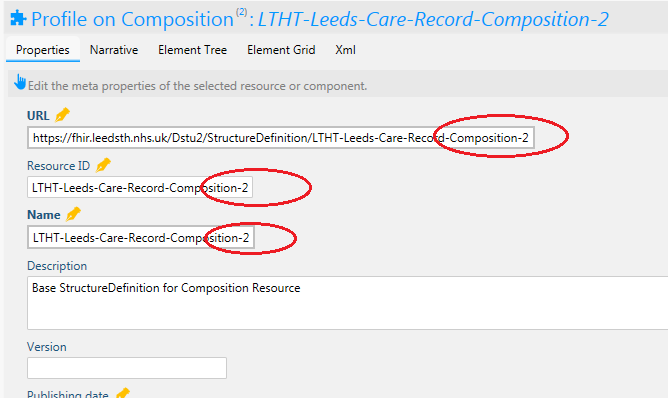
That’s it… or that’s how to do it for the next sections.

## Adding Structure

What we’ve done so far is very similar to GP Connect, we’ve defined a resource to return a patient summary record in html, split into sections. (The text node of the section contains the html). This is useful for displaying the context in a browser but parsing html is not ideal for information systems. As mentioned earlier each section corresponds to a FHIR resource type, the table below shows the CareConnect profile or FHIR resource mapping to type.

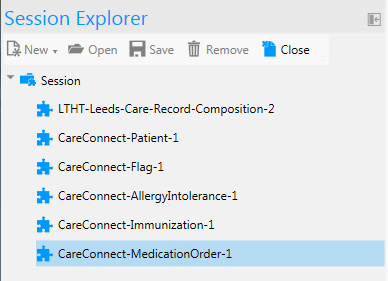
|  |  |
| --- | --- |
| Section Title | CareConnect (or FHIR resource) |
| Patient | https://fhir.hl7.org.uk/CareConnect-Patient-1.structuredefinition.xml |
| Alerts | https://fhir.hl7.org.uk/CareConnect-Flag-1.structuredefinition.xml |
| Allergies | https://fhir.hl7.org.uk/CareConnect-AllergyIntolerance-1.structuredefinition.xml |
| Care Plan | Care Plan |
| Immunisations | https://fhir.hl7.org.uk/CareConnect-Immunization-1.structuredefinition.xml |
| Medications | https://fhir.hl7.org.uk/CareConnect-MedicationOrder-1.structuredefinition.xml |
| Referrals | ReferralRequest |

To add links to these resources to our Composition we will create a new version of our composition. Change the version number to 2.

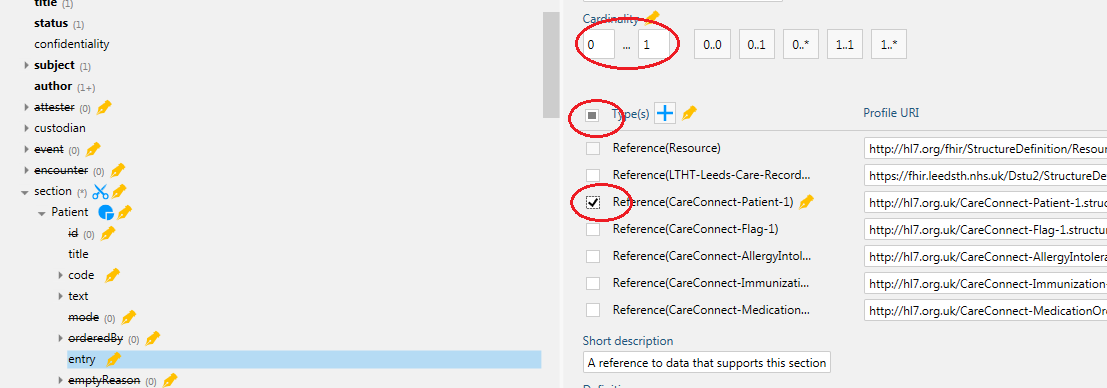


Save the profile with a new file name.

Within forge open the CareConnect profiles mentioned in the table above, this makes adding the resources easier.



Firstly we will configure the patient section of the composition. Find the entry node and change the cardinality to 0..1, uncheck the Type text box and select the CareConnect-Patient-1 type.



Adding the CareConnect profiles to the current session added them to the type picklist.

Similarly for the alerts section, change the cardinality to 0..\* (a patient can have many alerts) and link it to CareConnect-Flag-1. Repeat this for the remaining sections.

Referrals and Care Plan won’t be linked to a CareConnect profile, if you wish you can link them to your own profile if you have created one.

That’s it, we’ve now extended the Composition profile to handle structured resources but you may be asking where do I put the actual resources?

They are placed with the Composition within a FHIR Bundle.

## 

## Composition-Bundle

The source code can be found here: <https://github.com/KevinMayfield/FHIRTest/tree/master/Compositions>

We would create a profile on FHIR Bundle and this can be found in the github project but as we’ve covered slicing earlier we will concentrate on how we build the bundle in code. HAPI FHIR and Java is used in this section as it is the most mature implementation and shows the key concepts. We also use Apache Camel testing components, Camel is an open source integration tool based on known integration patterns which supports many transport patterns out of the box.

First we will build a basic Composition:

*uk.nhs.jorvik.fhirTest.careConnectCompositionTest.CareConnectExamples.java*



This is fairly explanatory, we have made some shortcuts with organisation and patient references which assume an endpoint exists which will serve these resources. At the time of writing these are virtual references.

Code in other languages such as C# or Cache should look similar.

A simple test called testSendCompositionV1 can now use this function to check it functions correctly.

*uk.nhs.jorvik.fhirTest.careConnectCompositionTest.CareConnectCompositionTest.java*

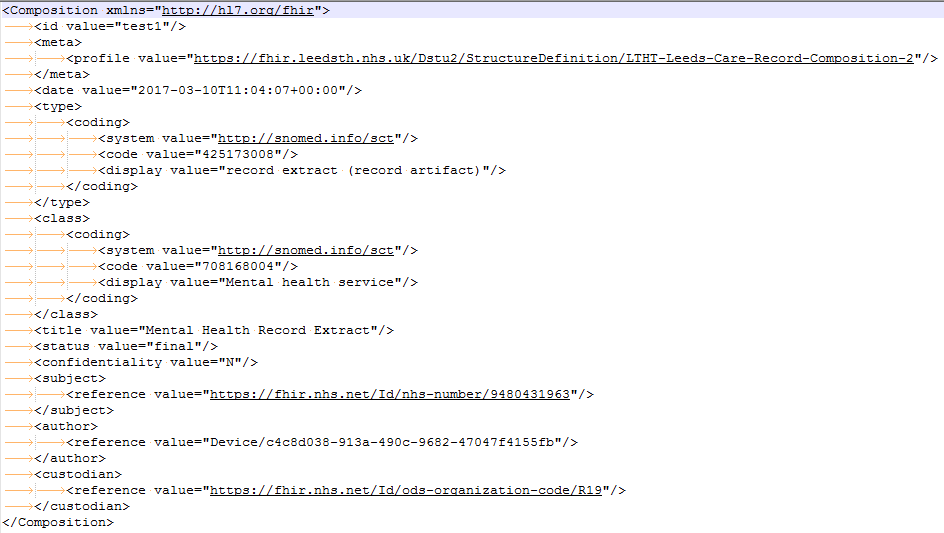


The test uses Apache Camel to send the resource to a file and a reference server via a simple camel route:

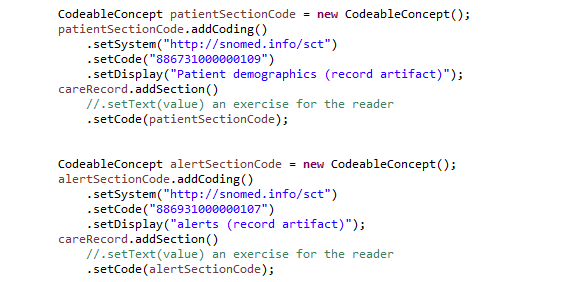
*uk.nhs.jorvik.fhirTest.javaconfig.FHIRRoute.java*



The processor section adds the headers required for the REST API. The test will now generate a file containing the sample Composition resource.



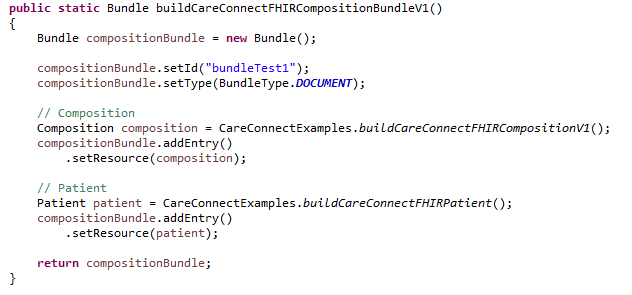
We’ve not included any code for the sections at this point. This is because we will go straight on to a Composition with structured data. The code below shows how sections could be coded.



## Composition Bundle

Next we need to start adding the sections to resource starting with Patient. To do this we need to bundle together the Composition and Patient resources. To do this we use a FHIR Bundle, the main resource (composition) must come first.

A simple view on how a bundle is built is done is shown below



This creates a bundle and adds two example resources but the resources aren’t linked to each other and the composition doesn’t contain a patient section. The Bundle is set to type Document which is used to denote this is a FHIR Document with Composition as the first resource.

From the composition profile we need to add a reference to the resource and coding for the section. The previous code sample now becomes:



Note how we have changed the id for the patient and used this as the reference within the patient section of the composition.

The route use for this test is similar but uses a Bundle endpoint.

The source code contains sections for Allergies and Alerts. If maven is installed correctly, the samples should run via the **mvn test** command and output data to *C:\test\Composition* and *C:\test\Composition\Output* folders. If you don’t wish to run the tests, sample output can be found in the example folder in the github project.

The tests are minimal, enough to demonstrate how to test.