FHIR Shorthand Thoughts

Sydney Connectathon

2/3/2020

# 1) Adding an extension slice (nickjpgeorge@gmail.com)

For an extension named FishSpecies, this is currently this is done as:

|  |
| --- |
| \* extension[FishSpecies] |

This implies that the extension slice will be named as FishSpecies. Two issues with this approach:

1. The profile has no control over the name of the slice (e.g., UsCore Patient uses race) as name of extension, not UsCorePatientRace
2. The slice name is non-standard case (CapitalCase instead of camelCase)

**Proposal**: specify slice name separate from type.

|  |
| --- |
| \* extension[species] FishSpecies |

Since adding extensions the most common kind of slicing, and almost never includes any additional slicing metadata, some additional sugar could be added so that the `extension contain` stanza could be elided:

|  |
| --- |
| \* extension[species] 0..1 FishSpecies |

**Thought:** Referring to extensions by name makes me a little uncomfortable, because it could be ambiguous, e.g., if two Extensions share a name, or Extension has a name that is a Resource type.

**Suggestion:** I would advise my team to only refer by name to things within the current package, otherwise use URL.

# 2) ComplexExtension keyword (nickjpgeorge@gmail.com)

Consider USCoreEthnicityExtension

|  |
| --- |
| Extension: USCoreEthnicityExtension \* extension contains ombCategory 0..1 MS and detailed 0..\* and text 1..1 MS \* extension[ombCategory].value[x] only Coding \* extension[ombCategory].valueCoding from OmbEthnicityCategories (required) \* extension[detailed].value[x] only Coding \* extension[detailed].valueCoding from DetailedEthnicity (required) \* extension[text].value[x] only string |

This feels a bit verbose. If we make a specialized grammar for keyword **ComplexExtensions** we could express this all as:

|  |
| --- |
| ComplexExtension: USCoreEthnicityExtension \* ombCategoryCoding 0..1 MS Coding from OmbEthnicityCategories (required) \* detailed 0..\* Coding from DetailedEthnicity (required) \* text 1..1 MS string |

#

# 3) Nested Complex Extensions (nickjpgeorge@gmail.com)

If I understand correctly, it’s not currently possible to **define** a multi-level complex extensions via FSH

E.g., something where

|  |
| --- |
| MyComplexExtension.extension[complexField].extension[intField] = 5MyComplexExtension.extension[complexField].extension[stringField] = "foo"MyComplexExtension.extension[simpleField] = "bar" |

Proposed grammer, building on the previous proposal

|  |
| --- |
| ComplexExtension: MyComplex \* complexField 0..1 ComplexExtension \* complexField.intField 1..1 integer \* complexField.stringField 0..\* string \* simpleField 0..1 string |

# 4) Ginzu/auto slicing on Coding (nickjpgeorge@gmail.com)

In CodeableConcepts, it is very useful to slice on system value. I’m not sure of any other slicing that would be useful. I propose that if a codeable concept has slices, we assume the following:

|  |
| --- |
|  \* $CODEABLE\_CONCEPT.coding ^slicing.discriminator.path = "system" \* $CODEABLE\_CONCEPT.coding ^slicing.discriminator.type = #value |

# 5) Alias feel a bit ambiguous (nickjpgeorge@gmail.com)

It’s not obvious to me when something should be substituted out for an alias. Pure string matching feels brittle.

**Suggestion**: Adopt the common “**$VARIABLE**” pattern.

E.g.,

Instead of

|  |
| --- |
| \* codes from system http://snomed.info/sct where code is-a SCT#90580008 "Fish (organism)" |

Use

|  |
| --- |
| \* codes from system http://snomed.info/sct where code is-a $SCT#90580008 "Fish (organism)" |

This will make it easier to visually scan for variable substitutions, as well as presumably easier to do typo checking and syntax highlighting.