Profile Tooling

This page gathers links for tools that help with development of FHIR resource, including Profiles/StructureDefinitions. These tools are categorized into several functional categories:

- **1 Validating Resources against the specification and Profiles**
  - 1.1 Validator Status
  - 1.2 Marking that a resource conforms to a profile
- **2 Editing / Authoring Profiles**
- **3 Publishing Profiles**
  - 3.1 Finding Packages and Implementation Guides
  - 3.2 Creating and publishing Packages and Implementation Guides

## Validating Resources against the specification and Profiles

You can validate a resource against a profile in several different ways.

- **On the web**
  - You can use validator.fhir.org - this is the same validator as the command line tool (Java engine)
  - On the Validation page of Simplifier.net you can, after creating a free account, validate any resource against the core FHIR specification or any published FHIR package. (DotNet engine)
    - Simplifier.net also allows you to store and validate one FHIR project for free. Other features require a paid subscription.
  - Also see Public FHIR Validation Services.
- **Via API**
  - You can use the [validation] interaction on any of the Public Test Servers or your own FHIR server instance.
    - use a RESTful client like [POSTman], or write code to do this in the background
    - post the resource as described by the validation operation, and tag it with the profile you wish it to be tested against (see below)
    - most servers require the profile to be hosted on the server itself
- **On the command line**
  - You can use the Java validator package. See [Using the FHIR Validator]. (Java engine)
  - You can install the Firely Terminal command line tool. Validation of individual resources is free; some other features require a paid subscription. (DotNet engine)
- **In code**
  - You can use the Java classes directly
    - see the java package org.hl7.fhir.instance.validation in the [FHIR GitHub]
  - Or use the corresponding .NET from the Firely .NET SDK.

### Validator Status

All the options above package one of three different validation engines:

<table>
<thead>
<tr>
<th>Engine</th>
<th>Source</th>
<th>Status</th>
<th>Passes Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java</td>
<td><a href="https://github.com/hapifhir/org.hl7.fhir.core">https://github.com/hapifhir/org.hl7.fhir.core</a></td>
<td>Fully maintained</td>
<td>Yes</td>
</tr>
<tr>
<td>C#/DotNet</td>
<td><a href="https://github.com/FirelyTeam/firely-net-sdk">https://github.com/FirelyTeam/firely-net-sdk</a></td>
<td>Fully maintained</td>
<td>No</td>
</tr>
<tr>
<td>Pascal</td>
<td><a href="https://github.com/HealthIntersections/fhirserver">https://github.com/HealthIntersections/fhirserver</a></td>
<td>Not current; future undecided</td>
<td>No</td>
</tr>
</tbody>
</table>

HL7 regards any validator that passes the validator test cases as an HL7 approved validator.

### Marking that a resource conforms to a profile

See the explanation of `meta.profile` in the section on resource metadata. Note that this is only a conformance claim and no guarantee that the resource actually conforms. Every server can test these claims, test resources against any other profile and can update the values in `meta.profile` accordingly.

## Editing / Authoring Profiles

The following choices exist for editing profiles

- **By hand**
  - Use a schema drive editor against Profile.xsd in the FHIR schemas
  - start with an existing profile (look for the link under the structure definition on the page for the resource)
• edit the profile as you see fit
• Note: profiles are full rich description of the content, and so they're pretty hard to edit by hand. You can do this but it's not the intention
• Generic XML tools
  • XmlSpy
  • Oxygen
  • Saxon XSLT2

• From spreadsheets
  • this is how some of the profiles in the specification are produced
  • use the FHIR Build Process following the method used here: FHIR Profile authoring

• Using the desktop user interface tool Forge
  • see the Firely website

• Using Trifolia
  • see the Lantana Trifolia website

• Using FHIR Shorthand
  • for FHIR Shorthand specification standard, see the FHIR Shorthand website
  • for documentation and tools, visit the FSH School

• Kodjin FHIR Profiler
  • The tool is available as a web-based version and a plugin to Visual Studio Code (see extensions market)
  • Kodjin FHIR Profiler: https://profiler.kodjin.com
  • Instructions: https://kodjin.com/fhir-profiler-instructions

Use of spreadsheets is no longer recommended (except for certain HL7-internal profiles) and hand editing is tedious and error prone. Generally, choose one of the other approaches.

Editing/testing profile invariants can be done with the "fhirpath-lab" it provides syntax highlighting and advanced testing capabilities

Publishing Profiles

FHIR resources can be published as a stable, coherent set called a FHIR Package (see the NPM Package Specification).

This should generally be accompanied by documentation, called an Implementation Guide.

Finding Packages and Implementation Guides

• FHIR Package Registry - All publicly available FHIR packages (and their guides) are available in https://registry.fhir.org. For more information see FHIR Package Registry User Documentation.
• Implementation Guide Registry (fhir.org) - A curated list of Implementation Guides.
• Simplifier.net - A registry containing all public profiles (StructureDefinition, ValueSet, Conformances) and a platform for private development with advanced search functionality

Creating and publishing Packages and Implementation Guides

• On the command line
  • The Implementation Guide Publishing Tool, see IG Publisher Documentation
    • Creating an Implementation Guide with the IG publisher will run extensive validation and create a documentation page, plus a FHIR package. See the IG Publisher Documentation.
    • The Implementation Guide can be self-hosted or, in case of official HL7 projects, hosted on by HL7 following the FHIR Implementation Guide Process Flow.
    • Packages published with HL7.org or FHIR.org will automatically be published in the FHIR Registry. Others can read in the FHIR Package Registry User Documentation how to do this.

• On the web
  • Simplifier.net Implementation Guide editor - Graphical editor for FHIR Implementation Guides
    • See the documentation on Creating packages. Packages will automatically be added to the FHIR Registry on publishing.
    • See the documentation on Creating Implementation Guides. Implementation guides can be hosted or exported and self-hosted (paid).
  • Trifolia on FHIR - FHIR Implementation Guide creation tool with support for editing profiles, value sets, narrative pages, and other content. Integrated with the FHIR IG Publisher

• Other
  • Create a custom website describing your FHIR resources.
  • Create a Implementation Guide resource and include it in your package.
  • You can still publish your FHIR resources as a FHIR package as described in the FHIR Package Registry User Documentation.