

Trifolia-on-FHIR

Table of contents

Introduction	4
Welcome	4
What's New	4
Navigation	6
Technical Details	7
System Requirements	8
REST API	8
Security and Permissions	9
Access and Permissions	10
Accounts and Login	11
FHIR Versions	12
Resource Permissions	12
Authoring	13
Getting Started	13
Implementation Guides	16
Narrative Pages	17
Markdown Syntax	18
JIRA Spec Generation	20
Structure Definitions/Profiles	21
Binding values to elements in a profile	23
Value sets	23
Code Systems	24
Slicing Elements	24
Elements	25
Publishing	26
Pre-Publishing Errors	27
IG Version	28
Generating Snapshot	29
IG Package ID	31
Terminology Server Connection	31
Dependency	32
Export/Import	33
Export	33
Import	34
GitHub Integration	35
Validation	35
Walk-through	36

Glossary 37

FAQ 37

Introduction

Documentation Formats

The help documentation is these formats:

- CHM
- DOCX
- PDF
- EPub

Welcome

Trifolia-on-FHIR (ToF) is a FHIR® implementation guide publishing tool.

Core Features

- Edit conformance resource types:
 - ImplementationGuide
 - StructureDefinition(Profiles/Extensions)
 - ValueSet
 - CodeSystem
 - CapabilityStatement
 - OperationDefinition
 - Questionnaire
- Import and view any resource in the FHIR® specification (e.g., Observation, MedicationStatement).
- Validate resources while authoring (uses [FHIR.js](#) for validation)
- Export:
 - Implementation Guides and associated resources
 - Bundles
 - FHIR IG Publisher packages
- Publish - Integrates with the FHIR IG Publisher
- Import individual resources and/or batch/transaction bundles

Support Request

Before requesting support, we suggest you review the topic in our [Help](#) Documentation.

- Support requests are captured using JIRA Service Desk, located [here](#). Requests require an Atlassian account.
- The FHIR Zulip chat has a channel dedicated to ToF publishing questions and announcements, located on chat.fhir.org in the [#trifolia-on-fhir](#) channel.

What's New

Release 3.0.4

Initial release on 5/31/2023, patched on 6/2/2023, 6/6/2023, 6/7/2023, 6/14/2023

Proprietary persistence layer

We have transitioned away from using one or more FHIR servers to support the persistence of data, and are now using a Mongo (document-based) database. The Mongo database consists of multiple collections (similar to tables) that represent the data persisted in ToF. A migration tool has been created to assist existing installations in migrating from FHIR servers to the Mongo database.

Making this change will allow ToF to:

- Import and export additional types of data
- Preserve the content of the resources as they are defined by the user, rather than amending the resources with extensions that are needed for ToF to function
- Integrate more easily (in the future) with tools such as GitHub and the FHIR IG Publisher

This transition minimally impacts the user interface. In some cases, the user interface is simplified because there is no need to distinguish between versions of FHIR and the FHIR server that supports the version of the IG.

One notable difference is that there is now an official concept of a "Project" that is tightly coupled with an ImplementationGuide. Now, when you open a project, it will show you metadata about the project that is not contained within the ImplementationGuide resource, and allows you to edit the ImplementationGuide resource separately.

We are *considering* allowing a single "project" to have multiple ImplementationGuide resources, share permissions, and other common functionality.

CDA Example Import Support

ToF's functionality for importing has been enhanced to support the import of CDA examples. When an XML file is being imported into a project that has a dependency on the CDA implementation guide, it automatically detects if the XML file is *not* a valid FHIR resource, and treats it as an arbitrary example. This supports CDA implementation guides having examples such as "Component" or "ClinicalDocument" that are not valid FHIR resources.

Profile editing improvements

In parallel to the very significant foundational changes for persistence, we are focusing on improving profile editing, specifically around the "Elements" tab. A critical bug has been fixed related to the sorting of elements within the profile's differential list. We will continue to prioritize profile editing enhancements and bug fixes in future sprints, as we see this as crucial functionality for the product as a whole.

GitHub functionality removed (for now)

Functionality for importing/exporting to/from GitHub has been temporarily removed. We are in the process of re-imagining how the GitHub functionality should be implemented given our significant changes to persistence.

Development Log

3.0.0

- [TOFDEV-644](#) - Story - Replace FHIR server(s) persistence with MongoDB
 - Including 35 sub-tasks that are related to this effort
- [TOFDEV-668](#) - Story - Support for importing CDA examples
 - Including 1 sub-task related to this effort
- [TOFDEV-634](#) - Defect - Adding a Must Support to a previously unconstrained element causes it to be added out of order in the XML/JSON/Raw
- [TOFDEV-714](#) - Defect - Package ID field allows spaces during initial auto-generation and errors out on Edit IG page

3.0.1

- [TOFDEV-734](#) - Improvement - Consistently pre-populate base URL, FHIR version and validating required fields across several resource pages (StructureDefinition, CodeSystem, ValueSet, etc.)
- [TOFDEV-731](#) - Improvement - FHIR version of every new StructureDefinition should be pre-populated with FHIR version # of IG
- [TOFDEV-751](#) - Defect - Element Definition editor window doesn't show options for narratives anymore
- [TOFDEV-750](#) - Defect - Re-add support for importing Bundle resources
- [TOFDEV-724](#) - Defect - Capability Statement auto-populated date needs to be in correct format

3.0.2

- [TOFDEV-768](#) - Defect - Searching for a Value Set doesn't do anything
- [TOFDEV-769](#) - Defect - Getting error Error parsing JSON when publishing
- [TOFDEV-773](#) - Defect - Missing part of URL for some resources
- [TOFDEV-774](#) - Defect - Getting wrong file extension downloading Examples

3.0.3

- [TOFDEV-779](#) - Defect - Changes to a page's content are lost under certain circumstances
- [TOFDEV-775](#) - Defect - Example disappears from Example Page after editing

3.0.4

- [TOFDEV-780](#) - Defect - Importing ImplementationGuide for the opened project does not persist changes
- [TOFDEV-735](#) - Defect - Changing Structure Definitions Name from Profiles/Extensions doesn't change Structure Definitions Name in Resources
- [TOFDEV-790](#) - Defect - Selects multiple checkboxes when selecting resource from Select a resource section

Navigation

Main Navigation

Note: Some menu items are hidden pending login. Menus are dependent on the user's permissions to the application.

File

- **Home** - This is the first screen users see after login. It lists the Core Features, links to request support, and new features and improvements in the latest Trifolia-on-Fire (ToF) release.
- **New Project** - Start a new implementation guide (IG).
- **Open project** - Continue working on an existing IG.
- **Open from computer** - Open a resource directly from their computers, either as an XML file or JSON file, and edit the resource in ToF without saving the resource to the FHIR server. When saving, the browser prompts users to re-download the updated resource as an XML or JSON file depending on the format when opened.

Help

- **Documentation** - Opens this help documentation in HTML format.

- **Request Support** - Opens the support page for ToF, where users can submit support requests (defects, new ideas for features/improvements and general questions).
- **Settings** - Change FHIR server.

Browse/Edit - Search, select, delete, and create new resources depending on the resource type selected in the sub-menu.

- Implementation Guides
- Profiles
- Capability Statements
- Operation Definitions
- Value Sets
- Code Systems
- Questionnaires

Import - Import resources from other locations into ToF.

- Export - Export implementation guides from ToF in various formats (i.e., bundles, FHIR IG Publisher package, GitHub, etc).
- Publish - Publish your implementation guide using the FHIR IG Publisher.

Help-Access the Help documentation and request support.

FHIR Server


ToF supports multiple versions of the FHIR standard. ToF currently supports STU3 and R4. Users can select a FHIR server at the top right of every screen.

For more information, see [FHIR Versions](#).

Login

Log into your account or update your profile. For more information, see [Accounts and Login](#).

Information

Click the  (information) icon in the top-right corner to learn key points of interest on the screen.

Note: Not all screens support this functionality, i.e. the Home screen.

Technical Details

Overview

- Open source

- Web-based
- Supports multiple versions of FHIR
- Single interface for CDA templating and FHIR profiling
- Integrated ImplementationGuide editor--supports most common profiling tasks
- Exports a complete FHIR build package (zip) or standalone web-based implementation guide (IG)
- FHIR API support--Includes FHIR API endpoints for each supported FHIR version
- Integrated value set authoring and VSAC integration
 - Supports importing value sets from VSAC
 - Can re-use value sets created in other specifications, such as C-CDA, US-Core, etc.
- Limited API support for conformance resources
- Supports creating enumerated value sets manually in Trifolia-on-FHIR

System Requirements

The only requirement for using Trifolia-on-FHIR (ToF) is a browser. Organizations may want to install ToF to keep data in their own database or to keep their users separate from a publicly available tool.

Browser

ToF requires Chrome, Firefox, Internet Explorer, Microsoft Edge, Safari, or another modern browser.

Installation

To install ToF on your individual servers, administrators must have the following requirements :

- **Windows** or **Linux**
- **Java 8+** (to execute the publish process using the FHIR IG Publisher)
- **Jekyll** (to successfully complete the publish process using the FHIR IG Publisher)
- **FHIR Server** (STU3 or R4) must support:
 - Creating resources via a PUT with an ID ([Update as Create](#))
 - The [\\$validate operation](#)
 - The [\\$meta-delete operation](#)
 - ImplementationGuide search query parameters:
 - resource
 - global
 - [_has](#) (reverse chaining) search criteria. For example: GET /StructureDefinition?_has:ImplementationGuide:resource:_id=<IG_ID>
 - [_include](#) search criteria to get a list of all resources related to an implementation guide. For example: GET /ImplementationGuide?_id=some-ig-id&_include=ImplementationGuide:resource&ImplementationGuide:global
 - [PATCH](#) operation is required for bulk editing

REST API

Use REST API to automate interactions with Trifolia-on-FHIR (ToF). For example, a REST API can export your implementation guide (IG) on a regular schedule through another program.

Multiple FHIR servers

- The default FHIR server for this proxy endpoint is the first FHIR server available in the drop-down of FHIR

servers in the settings screen of the application.

- If the ToF installation is configured to support multiple FHIR servers, the first FHIR server is the default in the REST API.
- To perform REST API operations on a different FHIR server, specify a `fhirserver` header in each request. The value of the `fhirserver` header must align with the ID of one of the FHIR servers returned by `/api/config`.

For more information, see [FHIR Versions](#).

FHIR Server Proxy

An `/api/fhir` end-point in the API represents a "proxy" to the FHIR server(s) available within the ToF installation. The proxy endpoint supports GET, PUT, POST, DELETE requests for individual resources, as well as [batches](#). Transactions are *not* supported by this proxy.

Authentication

Authentication is required to access the REST API. To get an authorization code to use the REST API:

1. Log into the ToF web application and open the settings screen where the Authorization Code is displayed.
2. Attach the authorization code in an Authorization header prefixed with Bearer.

Example: "Authorization: Bearer XXXX" where XXXX is your authorization code.

Trifolia-on-FHIR Native API

ToF's REST API is documented using Swagger. The publicly available installation exposes custom API documentation at <https://trifolia-fhir.lantanagroup.com/api-docs>. The API described by `/api-docs` is the same API the web application (i.e., user interface) uses.

Security and Permissions

Permissions for resources are stored in `Resource.meta.security`. A custom code is created for three types of permissions:

- Everyone - Anyone with a user account in the installation.
- Group - One or more users (Practitioners) represented as a single group. Use a group to represent a team of users. To create a group, see [Accounts and Login](#).
- User (Practitioner) - A single person with access to the installation. Trifolia-on-FHIR (ToF) requires every user to create a Practitioner that represents their user login to open ToF to a specific FHIR server for the first time.

Two Levels of Permissions:

- Read - Allows the user to search/view the resource
- Write - Allows the user to update/delete the resource

With these concepts combined, the resource may have several security codes. For example:

```
{
  resourceType: "ImplementationGuide",
  meta: {
    security: [
      // Everyone has access to read/write
      { system: "https://trifolia-fhir.../security", code: "everyone^read" },
      { system: "https://trifolia-fhir.../security", code: "everyone^write" },
      // Members of group test-group-id have access to read/write
      { system: "https://trifolia-fhir.../security", code: "group^test-group-id^read" },
      { system: "https://trifolia-fhir.../security", code: "group^test-group-id^write" },
      // A specific user (Practitioner) with id test-practitioner-id has access to read/write
    ]
  }
}
```

```

    { system: "https://trifolia-fhir.../security", code: "user^test-practitioner-id^read" },
    { system: "https://trifolia-fhir.../security", code: "user^test-practitioner-id^write" }
  ]
}
}

```

When a user searches for ImplementationGuide resources, ToF sends a search request to the FHIR server that includes a `_security` parameter with all possible variations applicable to the active user. For example:

```

// un-encoded for readability
https://some-fhir-server.com/fhir/ImplementationGuide?_security=<system>|
everyone^read,<system>|group^test-group-id^read,<system>|user^test-practitioner-id^read
// encoded
https://some-fhir-server.com/fhir/ImplementationGuide?_security=https%3A%2F%2Ftrifolia-fhir...%
2Fsecurity%7Ceveryone%5Eread%2Chttps%3A%2F%2Ftrifolia-fhir...%2Fsecurity%7Cgroup%
5Etest-group-id%5Eread%2Chttps%3A%2F%2Ftrifolia-fhir...%2Fsecurity%7Cuser%5Etest-
practitioner-id%5Eread

```

When a user clicks the **Edit** button on a resource, ToF retrieves a single/specific resource. The ToF server verifies whether the persisted resource grants the active user permissions to view the resource before sending the resource to the user's browser for viewing.

Similarly, when a user clicks **Save** or **Delete**, the ToF server retrieves the instance of the resource persisted on the FHIR server, verifies whether the user has permissions to modify the resource, and rejects the request with a 401 Unauthorized response if the user does not have permissions. Otherwise, the resource is updated on the FHIR server according to the request.

Access and Permissions

Authentication

ToF is designed with minimum requirements for user authentication to access the data stored on the FHIR servers ToF is configured to use. Additional permissions may be required depending on the configuration of the ToF installation.

Permissions

If the ToF installation is configured to require permissions, only data that the user has been permitted to view/edit will be access to them in the user interface. The remainder of this section presumes that permissions are enabled in the installation.

Permissions are maintained for each individual resource in the system. For example, permissions may be different for an instance of an ImplementationGuide compared to a StructureDefinition that the implementation guide references.

Each edit screen contains a "Permissions" tab which allows the user to define the permissions for the resource. The user may search for users and groups, and add read and/or write permissions to the resource for the selected users/groups.

The user may select a different resource to copy permissions, either:

1. Select a resource type and type search criteria in the text field. Suggestions will present below the text field. Select one of the suggestions and press the "Copy" button.
2. Click the "Search" button next to the text field to select a resource using the advanced search pop-up window. Click the "Copy" button after you identify and select a resource.

If you have permissions to a resource via a group and that resource has other associated groups and you *aren't* a member, the name of the group will not appear and the "Permissions" tab will only display the ID of those other groups.

If you do not have permissions to edit a resource, you cannot click the "Edit" button on the resource from the browse screen. Future enhancements may allow the user to access the "Edit" screen in a disabled state when the user doesn't have edit permissions to the resource.

Managing Groups

All users can create/manage their groups. A group may only have one manager.

To create/edit/delete groups, click your name in the top-right of ToF, and select the "Groups" tab. Changes to the "Groups" tab are persisted immediately. The "Save" button only applies to editing information for your profile.

When you create a group, you are automatically added as a member to the group. You cannot remove yourself as a member from the group.

Importing Resources

When importing *new* resources, the permissions for those new resources default to allow the user performing the import view/edit access. To allow additional permissions, edit each resource and grant additional permissions.

Implementation Guide Permissions

You may copy permissions for an implementation guide to resources within the implementation guide. This functionality is primarily for scenarios where the permissions to the implementation guide have changed, and those changes need to be propagated down to the resources within the implementation guide.

Note: You cannot change permissions for a resource you don't have access. If you don't have access to one or more of the resources within the IG, you cannot copy permissions from the IG to that/those resources.

To do this:

1. Open a project.
2. Select Browse > "Edit ImplementationGuide."
3. Select the "Permissions" tab.
4. Click the "Copy" button in the "Copy IG Permissions" panel.
5. After ToF copies the permissions, you will be prompted to indicate the number of resources changed as part of the request.

Accounts and Login

Users

1. On the top right side of the screen, click the Login button. If you do not have an account, you may register via this screen. Trifolia-on-FHIR (ToF) re-directs users to the identity provider to either register or login.

Note: After you have registered and logged-in with the identity provider, your browser will redirect to the ToF homepage.

2. If this is your first time logging-in to Trifolia-on-FHIR (ToF), you must create a profile (represented by a FHIR [Practitioner](#) resource in ToF's FHIR server). If you configure ToF with multiple FHIR servers, you may need to create a new profile for each FHIR server.

Note: ToF identifies you as the author of resources and associates your profile with audit records when changing resources.

3. Click your name at the top right side of the screen to edit your profile for the selected FHIR server.

Groups

Create a group to share resources and grant permissions to members. All users can create/manage their groups. A group may only have one manager. When you create a group, you are automatically added as a member to the group. You cannot remove yourself as a member from the group.

Managing Groups

1. Click your name in the top-right of ToF
2. In the Edit/User Person screen, select the Groups tab.
3. Under the Managing Groups section, click the plus icon to create a group.
4. Enter a name for the group.
5. Search for group members by name or email. Click the plus to add them to the group.
6. Click **Save**.

Note: Changes to the "Groups" tab are persisted immediately. The "Save" button only applies to editing information for your profile.

FHIR Versions

Trifolia-on-FHIR (ToF) supports multiple versions of the FHIR standard: STU3 and R4. It defaults to the FHIR server selected during previous login.

To select a different FHIR server:

1. Click the FHIR server button to display the Edit Settings screen.
2. In the **Selected FHIR Server** field, click drop down menu and select the FHIR server.

FHIR Versions in the UI

Depending on the FHIR server and the version it supports, the screens for editing resources may appear (e.g., STU3 vs. R4). The screens in ToF reflect the changes between STU3 and R4 resources. In STU3, for example, ImplementationGuide has "packages" with "resources" inside each package. In R4, ImplementationGuide has "resources" parallel to "packages," and each resource references a package.

Resource Permissions

If the Trifolia-on-FHIR (ToF) installation is configured to require permissions, only data that the user has been permitted to view/edit will be accessible to them in the user interface. The remainder of this section presumes that permissions are enabled in the installation.

Permissions are maintained for each individual resource in the system. For example, permissions may be different for an instance of an ImplementationGuide compared to a StructureDefinition that the implementation guide references.

If you have permissions to a resource via a group and that resource has other associated groups and you *aren't* a member, the name of the group will not appear and the "Permissions" tab will only display the ID of those other groups.

If you do not have permissions to edit a resource, you cannot click the "Edit" button on the resource from the browse screen. Future enhancements may allow the user to access the "Edit" screen in a disabled state when the user doesn't have edit permissions to the resource.

Implementation Guide Permissions

You may copy permissions for an implementation guide to resources within the implementation guide. This functionality is primarily for scenarios where the permissions to the implementation guide have changed, and those changes need to be propagated down to the resources within the implementation guide.

Note: You cannot change permissions for a resource you don't have access. If you don't have access to one or more of the resources within the IG, you cannot copy permissions from the IG to that/those resources.

To do this:

1. Open a project.
2. Select **Browse > Edit ImplementationGuide**.
3. Select the **Permissions** tab.
4. In the **Copy IG Permissions** panel, click the **Copy** button.

5. After ToF copies the permissions, you will be prompted to indicate the number of resources changed as part of the request.

Add or Copy Permissions from Another Resource

Each edit screen contains a "Permissions" tab which allows the user to define the permissions for the resource. The user may search for users and groups, and add read and/or write permissions to the resource for the selected users/groups.

To copy the permissions from one resource to another, in the **Permissions** tab:

1. Select a resource type and type search criteria in the text field. Suggestions will present below the text field.
2. Select one of the suggestions and press the **Copy** button.

OR

3. Click the **Search** button next to the text field to select a resource using the advanced search pop-up window.
4. Click the **Copy** button after you identify and select a resource.

Getting Started

IG Design Overview

1. Create/Open an ImplementationGuide (Project).
2. Create ValueSets and CodeSystems in the UI.
3. Create profiles (StructureDefinition resources) in UI. Create elements and constrain them to value sets created in step #2.
4. Import images into the IG to insert into narrative.
5. Create narrative pages for the IG.
6. Mark example resources as "example" with optional subject/context.
7. Publish the IG.
8. Review the published IG.
9. Review the QA results and resolve errors.
10. Export as HTML package and store in GitHub.

Create Implementation Guide

1. Open and log into Trifolia-on-FHIR.
2. Select File > New Project.
3. Specify an ID that is unique to your IG (e.g., test-ig).
4. Specify a URL unique to your IG (e.g., http://test.com/fhir/implementationguide/test-ig). The last leaf/level of the URL should match the ID to prevent errors during publishing.
5. Specify a name for your IG. This is the computable format of the name, used similarly (but not the same) as the ID.
6. Specify a title for your IG. This is what is displayed to readers of the IG.
7. Specify a Package ID. If developing an HL7 IG, you will receive this information pending PSS approval.
8. Define permissions using the "Permissions" tab.
9. Save the IG. It is now set as the project you are working on.

Create Narrative Pages

1. Click Browse/Edit > Edit Implementation Guide.
2. Select the "Narrative/Pages" tab.

3. Provide a high-level description of the implementation guide, if desired. If no narrative pages are specified (see Step 4), then the IG's home page will use this description.
4. Scroll to the bottom and click the "Add Narrative Page" button. This will create an index page that represents the home page for your IG. When you edit this page, ToF can automatically generate the content for the home page (index) based on the description of the IG. You can also specify custom content displayed on the home page.
5. Click the "+" on the index/home page to create child pages.
6. The "Downloads" template is available to add to your IG and customize, which will provide default content for a "Downloads" page.
7. Select "Show on top navigation bar" and specify a navigation menu name for the pages you want to appear on the top menu of the published IG.
8. When editing a page, you can click the icon that represents multiple images to insert imports.

Import Images

1. Click the "Import" menu from the top navigation menu.
2. Drag-and-drop image files from your computer to the import window.
3. Click "Import."

Identify Examples

1. Click Browse/Edit > Edit Implementation Guide.
2. Select the "Resources" tab.
3. Click "Edit" next to an example resource.
4. Select "Example = Yes" or select a profile to set an example.
5. Click "Done."
6. "Save" the IG after updating resources.

Publish

1. Click the "Publish" menu from the top navigation menu.
2. Select the options for your publication.
3. HL7 requires the latest IG publisher for ballots. The IG publisher updates occasionally include bugs, which leads to errors. You can select "No" pending resolution.
4. Selecting "No" for "Use terminology server" can quicken the publish execution, assuming you're not worried about terminology.
5. Indicating "Yes" for "Download?" will offer the entire published package as a download to your computer when complete.
6. Click "Publish." This process can be time-consuming, depending on the size of your IG. Please be patient. Once published, you can view the IG with the link on the status screen. Alternatively, you can select Browse/Edit > View Implementation Guide.

Create Terminology

Create a new value set or code system:

1. Select Browse/Edit > Value Sets and/or Code Systems.
2. Select the blue "+" button in the top-right to create a new value set or code system.
3. Specify the details of the value set and/or code system.
4. Save the value set or code system. The value set or code system is automatically associated with your IG.

Locate existing value sets or code systems:

1. Select Browse/Edit > Edit Implementation Guide.

2. Select the “Resources” tab.
3. Select the “hand/pointer” icon in the top-right to add an existing resource.
4. Un-check “Showing resources for the “Test IG” implementation guide.”
5. Filter/Find then select the value set or code system.

Import a ValueSet from VSAC:

1. Select “Import” on the top navigation bar
2. Select the “VSAC” tab
3. Specify your VSAC credentials
Your credentials are never sent to ToF servers, they are used only by your browser to communicate with VSAC; they do not leave your computer, aside from being sent to VSAC for authentication.
4. Specify the OID for the value set you would like to import.
5. Click “Import”.
6. Assuming the value set is found, it will be imported and automatically associated with your IG.

Create Profiles

1. Select Browse/Edit > Profiles/Extensions.
2. Click the blue “+” icon in the top-right of the screen.
3. Indicate the URL for the profile. The beginning of the URL should match the IG’s URL (e.g., <http://test.com/fhir/structureddefinition/my-profile> if the IG is <http://test.com/fhir/implementationguide/test-ig>).
4. Verify the ID matches the end of the URL. This should auto-populate when you enter the URL (e.g., “my-profile” for the above URL example)
5. Specify a name and title, which the published IG will display.
6. Indicate the “type” of resource this profile constrains (e.g., “Patient”).
7. You may want to build your profile based on another profile. Specify a “Base Profile” that exists in the system.
8. Click “Save.”
9. Click the “Elements” tab.
10. Select an element to constrain.
11. Click “Constrain Element” in the right panel.
12. Re-define the element in the right panel to specify the difference between the element and the base definition.
13. Repeat for all constraints.
14. Define a description for the profile, which the published IG will display to the reader.
15. Save the profile.
- 16.

Reverting to a previous version of a resource in Trifolia on FHIR

1. In the resource(IG, Profile, Value Set, etc) Editor, navigate to the History tab
2. Find the version of the resource you wish to restore
3. Select the “Revert” option to the right of the “Save” button to undo any unsaved changes
4. Select the “Restore” option by clicking the up arrow to the right of “Comparing version E to version #” to review the history of a resource and restore a previously saved version.
5. Save the IG in order for the restored version to persist

Implementation Guides

Guidelines and Best Practices

Trifolia-on-FHIR has the functionality to allow users to completely customize resources. By following these guidelines, users can ensure the FHIR publisher successfully processes the Implementation Guide.

Implementation Guide

An implementation guide (IG) is a set of rules about how FHIR resources are used (or should be used) to solve a particular problem, with associated documentation to support and clarify the usage. Classically, FHIR implementation guides are published on the web after they are generated using the FHIR Implementation Guide Publisher.

The ImplementationGuide resource is a single resource that defines the logical content of the IG, along with the important entry pages into the publication, so that the logical package that the IG represents, so that the contents are computable. In particular, validators are able to use the ImplementationGuide resource to validate content against the implementation guide as a whole. The significant conformance expectation introduced by the ImplementationGuide resource is the idea of Default Profiles. Implementations may conform to multiple implementation guides at once, but this requires that the implementation guides are compatible.

- The URL of the Implementation Guide must be in the format of `http[s]://xxx.yyy/zzz/aaa/ImplementationGuide/my-ig-id`. For example:
`http://myproject.com/someRoot/ImplementationGuide/myproject-ig`
- The "id" of the implementation guide must align with the URL of the implementation guide. For example: If the URL of your implementation guide is `http://myproject.com/someRoot/ImplementationGuide/myproject-ig`, the id must be "myproject-ig". Users can select the "Change this resource's ID" button on the "Browse Implementation Guide" screen.
- The Implementation Guide should have a description. The main screen of the FHIR IG Publisher export displays the description.
- All contacts in the Implementation Guide appear as authors in the FHIR IG Publisher export.
- ToF only exports resources referenced directly within the Implementation Guide resource. Confirm the Implementation Guide resource references all resources.

Resources

The FHIR specification defines a set of resources, and an infrastructure for handling resources. In order to use FHIR to create solutions for integration requirements, implementers must map their problems to resources and their content.

- All resources within an Implementation Guide need URLs in one format. Based on the example above, if your implementation guide's URL is `http://myproject.com/someRoot/ImplementationGuide/myproject-ig` then all profiles (StructureDefinition resources) within the Implementation Guide must have URLs that start with <http://myproject.com/someRoot/StructureDefinition/>.

Creating an Implementation Guide in Trifolia on FHIR

1. Open and log into Trifolia-on-FHIR.
2. Select File > New Project.
3. Specify an ID that is unique to your IG (e.g., test-ig).
4. Specify a URL unique to your IG (e.g., `http://test.com/fhir/implementationguide/test-ig`). The last

leaf/level of the URL should match the ID to prevent errors during publishing.

5. Specify a name for your IG. This is the computable format of the name, used similarly (but not the same) as the ID. A natural language name identifying the implementation guide. This name should be usable as an identifier for the module by machine processing applications such as code generation.
6. Specify a title for your IG. A short, descriptive, user-friendly title for the implementation guide. This is what is displayed to readers of the IG.
7. Specify a Package ID. If developing an HL7 IG, you will receive this information pending PSS approval.
8. Define permissions using the "Permissions" tab.
9. Save the IG. It is now set as the project you are working on.

Narrative Pages

An implementation guide can (and should) contain narrative pages that explain the use-case for the implementation guide as well as special requirements. You can add these narrative pages in the "Edit Implementation Guide" screen under the "Narrative/Pages" tab.

The content of the pages is markdown. The ToF user interface provides a WYSIWYG editor based on markdown syntax. The WYSIWYG editor supports basic markdown syntax, such as bold, italics, underline, and headings. More advanced markdown functionality (e.g., tables) may not display correctly in the editor, but will be supported during the publish process. The WYSIWYG editor doesn't show the tables; however, they may appear as tables after you publish your implementation guide.

Navigation Menus

You can associate each page in the implementation guide with a top-level navigation menu. Edit the page and specify a name for the top-level navigation menu. When you edit additional pages and begin typing the same menu name, the existing menu name will appear as a suggestion. After publishing the implementation guide, each unique page name will appear as a top-level navigation menu with all pages associated with that menu appearing as options within that menu.

The top navigation menu *does not* reflect the hierarchy of the pages. The table of contents *does* reflect the hierarchy of the pages. The table of contents is automatically generated during the publish process.

Images

To add one or more images to pages in your implementation guide:

1. Import your images via the "Import" screen. You may drag-and-drop the images into the "File" tab. ToF will import these images as "Media" resources. The "id" of the Media resource will be based on the filename of the image. The exact file name will be stored as an "identifier" in the Media.
2. Ensure your newly imported Media resources are added to the IG's "resources". Make sure they are *not* marked as an example. Leave the "Example" field either "Undefined" or "No". Otherwise, your Media resources will be an example preserved during the export process, which may produce errors during final publication.
3. Open the page(s) where you want the image to appear and place your cursor where you want the image. Select the "Insert image from predefined list" option in the Markdown editor.
4. Select the image.
5. Text will appear at your cursor for the image you selected.

Note: ToF supports the following image types:

- .JPG
- .GIF
- .PNG
- .BMP

Technical Notes

During the IG export and publish processes, the pages in the IG have to be in a specific order. The first/root page of the IG *has* to be an index.html file. If the first page in an IG is *not* an index.html file, then ToF automatically adds one and makes the root page of the IG a child page of the new index.html page.

If the ImplementationGuide resource does not have properties for storing the content of each page, ToF creates contained Binary resources within the ImplementationGuide resource.

Markdown Syntax

Headers

H1

H2

H3

H4

H5

H6

Alternatively, for H1 and H2, an underline-ish style:

Alt-H1

=====

Alt-H2

Emphasis

Emphasis, aka italics, with *asterisks* or underscores.

Strong emphasis, aka bold, with **asterisks** or underscores.

Combined emphasis with **asterisks and underscores**.

Strikethrough uses two tildes. ~~Scratch this.~~

Lists

1. First ordered list item

2. Another item

* Unordered sub-list.

1. Actual numbers don't matter, just that it's a number

1. Ordered sub-list

4. And another item.

To have a line break without a paragraph, you will need to use two trailing spaces

* Unordered list can use asterisks

- Or minuses

+ Or pluses

Links

There are two ways to create links.

[I'm an inline-style link](https://www.google.com)

[I'm an inline-style link with title](https://www.google.com "Google's Homepage")

[I'm a reference-style link][Arbitrary case-insensitive reference text]

[I'm a relative reference to a repository file](../blob/master/LICENSE)

[You can use numbers for reference-style link definitions][1]

Or leave it empty and use the [link text itself].

URLs and URLs in angle brackets will automatically get turned into links.

<http://www.example.com> or [<http://www.example.com>](http://www.example.com) and sometimes [example.com](http://www.example.com) (but not on Github, for example).

Some text to show that the reference links can follow later.

[arbitrary case-insensitive reference text]: <https://www.mozilla.org>

[1]: <http://slashdot.org>

[link text itself]: <http://www.reddit.com>

Images

Here's the markdown component's logo (hover to see the title text):

Inline-style:

![alt text](https://github.com/adam-p/markdown-here/raw/master/src/common/images/icon48.png "Logo Title Text 1")

Reference-style:

![alt text][logo]

[logo]: <https://github.com/adam-p/markdown-here/raw/master/src/common/images/icon48.png> "Logo Title Text 2"

Code and Syntax Highlighting

Inline ``code`` has ``back-ticks around`` it.

Specify the multi-line code with the language:

```
```javascript
var s = "JavaScript syntax highlighting";
alert(s);
...`
```

No language indicated, so no syntax highlighting:

```
...`
```

But let's throw in a `<b>tag</b>`.

```
...`
```

## Blockquotes

> Blockquotes are very handy in email to emulate reply text.

> This line is part of the same quote.

Quote break.

> This is a very long line that will still be quoted properly when it wraps. Oh boy let's keep writing to make sure this is long enough to actually wrap for everyone. Oh, you can `*put*` `**Markdown**` into a blockquote.

## Horizontal Rule

Three or more...

---

Hyphens

\*\*\*

Asterisks

—

Underscores

## Tables

Colons can be used to align columns.

```
| Tables | Are | Cool |
| :----- | :-----: | :----:|
| col 3 is | right-aligned | $1600 |
| col 2 is | centered | $12 |
| zebra stripes | are neat | $1 |
```

There must be at least 3 dashes separating each header cell.

The outer pipes (|) are optional, and you don't need to make the raw Markdown line up prettily. You can also use inline Markdown.

Markdown | Less | Pretty

```
--- | --- | ---
```

```
Still | `renders` | **nicely**
```

```
1 | 2 | 3
```

```
123123123
```

```
[](http://)
```

## JIRA Spec Generation

The "Edit Implementation Guide" screen has functionality to generate a JIRA Spec file that is required by the HL7 ballot process. You can read more information on the purpose of the JIRA Spec file [here](#).

ToF provides basic support for generating the JIRA Spec file that is needed for HL7 ballots. The first time the user clicks generate, it creates a template of the JIRA Spec file with as many defaults filled in as possible. Some of the information will have to be manually entered by the user because the information is not available automatically to Trifolia-on-FHIR, such as the "ballot url", the "git url", the "ci url", etc.

ToF automatically populates the "artifact" and "page" elements in the JIRA Spec based on the resources and pages in the IG.

If the text field for the JIRA Spec is empty when the user clicks "Generate", the JIRA Spec is generated brand new, with defaults for fields such as "ballot url", "git url", etc. If the text field has a JIRA Spec already defined in it when the user clicks "Generate", the existing JIRA Spec's artifacts and pages are updated based on the current resources and pages defined in the IG. When re-generating, ToF will flag old artifacts and pages as deprecated if the resource/page is not found in the current IG with a matching "key".

The process a user should follow for working with the ToF-generated JIRA Spec is:

1. Create an IG, define resources and pages within the IG.
2. Click "Generate" for the JIRA Spec.
3. Specify the correct values for default properties such as "ballot url", "git url", etc.
4. Save the IG to persist the JIRA Spec within the IG.
5. Follow the process defined by HL7 for creating a pull request for the JIRA Spec.

6. Over time, the IG is updated with new resources/pages...

1. The user clicks "Generate" for the JIRA Spec and the artifacts/pages are updated based on the current state of the IG.
2. Save the IG to persist the JIRA Spec's updates within the IG.
3. Update the JIRA Spec within HL7's GIT repository via a new pull request.

## Structure Definitions/Profiles

---

### Structure Definition

A definition of a FHIR structure. This resource is used to describe the underlying resources, data types defined in FHIR, and also for describing extensions and constraints on resources and data types.

The StructureDefinition resource describes a structure - a set of data element definitions, and their associated rules of usage. These structure definitions are used to describe both the content defined in the FHIR specification itself - Resources, data types, the underlying infrastructural types, and also are used to describe how these structures are used in implementations. This allows the definitions of the structures to be shared and published through repositories of structure definitions, compared with each other, and used as the basis for code, report and UI generation. For more information on Structure Definitions, see <https://www.hl7.org/fhir/structuredefinition.html>.

### Profiling FHIR

A profile is defined as a A set of constraints on a resource represented as a structure definition with kind = **constraint**

The base FHIR specification describes a set of base resources, frameworks and APIs that are used in many different contexts in healthcare. However, there is wide variability between jurisdictions and across the healthcare ecosystem around practices, requirements, regulations, education and what actions are feasible and/or beneficial.

For this reason, the FHIR specification is a "platform specification" - it creates a common platform or foundation on which a variety of different solutions are implemented. As a consequence, this specification usually requires further adaptation to particular contexts of use. Typically, these adaptations specify:

- Rules about which resource elements are or are not used, and what additional elements are added that are not part of the base specification
- Rules about which API features are used, and how
- Rules about which terminologies are used in particular elements
- Descriptions of how the Resource elements and API features map to local requirements and/or implementations

Note that because of the nature of the healthcare ecosystem, there may be multiple overlapping sets of adaptations - by healthcare domain, by country, by institution, and/or by vendor/implementation. For more information on FHIR Profiles, please visit <https://www.hl7.org/fhir/profiling.html>.

Note: Trifolia-on-FHIR (ToF) forces users to create *structurally* correct FHIR but does not enforce that the content of the resources meet the requirements of all business rules in FHIR. An example of this, is that the structure of an Extension may be correct, but slices within the Extension may not be correct. Over time, as FHIR continues to mature and those business rules are solidified, ToF will be updated to enforce them.

## ***Creating/Editing Profiles in Trifolia on FHIR***

1. Select Browse/Edit > Profiles/Extensions.
2. Click the blue “+” icon in the top-right of the screen to create a new profile or the edit icon to edit an existing profile.
3. Indicate the URL for the profile. The beginning of the URL should match the IG’s URL (e.g., <http://test.com/fhir/structuredefinition/my-profile> if the IG is <http://test.com/fhir/implementationguide/test-ig>).
4. Verify the ID matches the end of the URL. This should auto-populate when you enter the URL (e.g., “my-profile” for the above URL example)
5. Specify/update a name and title, which the published IG will display.
6. Add/update the “type” of resource this profile constrains (e.g., “Patient”).
7. You may want to build your profile based on another profile. Specify/update a “Base Profile” that exists in the system.
8. Click “Save.”
9. Click the “Elements” tab.
10. Select an element to constrain or update
11. Click “Constrain Element” in the right panel for new elements, or click on the element to update an existing element.
12. Re-define the element in the right panel to specify the difference between the element and the base definition.
13. Repeat for all constraints.
14. Define a description for the profile, which the published IG will display to the reader.
15. Save the profile.

## ***Extensions***

All extensions used in resources require a formal published definition that can be used by application developers or the applications themselves, to help integrate extensions into the healthcare process they support.

Every extension in a resource refers directly to its definition, which is made available as a [StructureDefinition](#). A resource can be [profiled](#) to specify where particular extensions are required or expected.

For more information on creating/incorporating extensions into existing resources, please visit the “Defining Extensions” page at <https://www.hl7.org/fhir/defining-extensions.html>

## Binding values to elements in a profile

### ***To create a fixed binding for an element in a profile:***

Starting from the profile editor's "Elements" tab:

1. Select the element you want to bind to the fixed value.
2. Select the "Binding" tab in the properties of the element on the right side of the screen.
3. Check the checkbox next to "Fixed."
4. Select the associated "type" for the element (e.g., "CodeableConcept" or "string," depending on the data type of the element).
5. Edit the value of the fixed binding according to the instruction for the implementer.  
In some cases, more complex data types (e.g., CodeableConcept) will show an "Edit" (pencil) icon next to the type you selected in Step 4, which opens a dialog box to enter the information.  
Types that are simple (e.g., "code" or "string") simply show a text field to enter the value.

## Value sets

A ValueSet resource instance specifies a set of codes drawn from one or more code systems, intended for use in a particular context. Value sets link between CodeSystem definitions and their use in [coded elements](terminologies.html). When using value sets, proper differentiation between a code system and a value set is important. This is one very common area where significant clinical safety risks occur in practice.

- Value sets used by an implementation guide *should* have a "compose" defined which asserts either the enumerated codes that should be included in the value sets, or asserts other value sets that should be included (making the value set a "wrapper" of sorts).
- Enumerated codes are shown/edited in the "Compose" tab's "Concepts" section. This section is paged, showing five (5) codes at a time. You may search for a concept by either the code or display values by entering text in the "Code (search)" and "Display (search)" fields shown at the top of the table.
- Additional fields (such as the "Designations") may be modified for each concept by clicking the "Edit" button.

Below the table of (at most) five (5) concepts is a set of buttons which allow you to control which page you are viewing/editing.

- The << button returns you to the first page of concepts
- The >> button moves you to the last page of concepts
- The < button moves you one page backward
- The > button moves you one page forward
- Selecting a number will bring you to that specific page number

The "Value Sets" section allows you to indicate what value sets should be included in this value set. Each entry in the "Value Sets" section represents the canonical URL of the value set (ValueSet.url).

When publishing an implementation guide which has a value set that references other value sets, those *other* value sets must be available to the FHIR IG Publisher via one of the following methods:

- Included in the implementation guide itself (via a resource referenced in the ImplementationGuide resource)
- The ValueSet is publicly available by the URL of the value set (e.x. putting the URL of the value set in a browser should return the ValueSet in either XML or JSON format)
- The terminology server used by the FHIR IG Publisher (tx.fhir.org) has the value set pre-loaded

## Code Systems

The CodeSystem resource is used to declare the existence of and describe a code system or code system supplement and its key properties, and optionally define a part or all of its content.

- Code systems define which codes (symbols and/or expressions) exist, and how they are understood. Value sets select a set of codes from one or more code systems to specify which codes can be used in a particular context.
- The CodeSystem resource may list some or all of the concepts in the code system, along with their basic properties (code, display, definition), designations, and additional properties.
- Code System resources may also be used to define supplements, that extend an existing code system with additional designations and properties.

## Slicing Elements

Slicing is only available/required on repeating elements. After a repeating element has been initially constrained, you can slice the element by clicking the scissors icon in the top-right like so:

The screenshot shows the 'Element Definition' interface for 'Age.extension'. A red arrow points to the 'Slicing' button (represented by a scissors icon) in the top right corner of the definition panel. Below the button is a tooltip that says 'Slice this element'. The interface also shows tabs for 'General', 'Narrative', 'Binding', and 'JSON', and input fields for 'ID/Path' (Age.extension) and 'Min/Max' cardinality (1).

This will create a new slice for the repeating element, and auto-generate the slice name. You can change the slice name to be more descriptive of what the slice represents. After creating a slice of the repeating element, you *should* go back to the top-level repeating element and specify discriminators for it. A new tab will appear when you have selected the the top-level repeating element that indicates “Slicing”, which allows you to specify details such as the “discriminator”.

The screenshot shows the 'Elements' table with columns for 'Element/Attribute', 'Flags', 'Cardinality', 'Type', and 'Description & Constraints'. The table lists 'Age' as the root repeating element, and 'extension' as a repeating element. A new slice, 'extensionslice1693', has been created. Red arrows point to the 'Root repeating element' and the 'New slice with auto-generated slice name'. On the right, the 'Slicing' configuration panel is shown, with a red arrow pointing to the 'Discriminator' checkbox and another pointing to the 'Type' and 'Path' fields.



## Elements

---

### ***Elements***

As the base type for all elements included in a resource, `Element` is an important structural element of FHIR. Even the primitive types inherit the base features and representation rules that apply to the `Element` type.

content:

- Extensions
- An internal id

There are 3 kinds of descendant types that specialize `Element`:

- Primitive data types, that add a primitive value property of the specified type
- Complex data types, that add their own children (all of which are also elements)
- `BackboneElement`, A specialization that adds `modifierExtension`, which is the super-type of all the element types defined in resource definitions (e.g. `Patient.contact`)

Note that resources themselves all specialize the base type `Resource`. For more information on the backbone element, visit <https://www.hl7.org/fhir/backboneelement.html>

### **Element Definition in Trifolia on FHIR**

- The profile editor loads the elements from the snapshot of the “base definition” profile.
  - If the base definition represents a profile that does not yet have a snapshot, it attempts to create a snapshot for the base definition profile.
  - If the base definition has other base definition profiles that are not in the system, the snapshot it returns will simply be the core FHIR profile for the “type” selected (ex: the underlying profile for “Observation”)
  - The “Base Definition” field should allow you to select pre-existing profiles that are based on a matching “type”, or the core FHIR specification’s profile *for* the underlying type.
- Cardinality
  - Users can set the cardinality of an element to an invalid value, but a Warning will appear in the Element Definition form and a Validation Error appears in the Validation tab if the user.
    - selects a min cardinality less than the base element's min cardinality.
    - selects a max cardinality greater than the base element's max cardinality.
- Maximum Field Length
  - `maxLength` can only be entered when the `ElementDefinition.type` is a primitive type, with the exception of “boolean” (instant time date dateTime decimal integer string uri base64Binary code id oid unsignedInt positiveInt markdown url canonical uuid).

- Narrative
  - **Short:** A concise description of what this element means (e.g. for use in autogenerated summaries).
    - Text entered in "Short" will be displayed in the "Description & Constraints" column of the Differential Table for the profile.
  - **Definition:** Provides a complete explanation of the meaning of the data element for human readability. For the case of elements derived from existing elements (e.g. constraints), the definition SHALL be consistent with the base definition, but convey the meaning of the element in the particular context of use of the resource.
    - Text entered in the "Definition" field will be displayed in the "Detailed Descriptions" page for the specific element.

## Publishing

An implementation guide (IG) is a set of rules on how FHIR resources are used to solve a particular problem, with associated documentation to support and clarify the usage. The FHIR Implementation Guide Publisher generates an IG for online publication.

The HL7 IG Publisher tool is developed by HL7 and is not part of Trifolia-on-FHIR (ToF). Questions related to the execution of the HL7 IG Publisher should be directed to [chat.fhir.org](https://chat.fhir.org).

The main screen of the FHIR IG Publisher export displays the description of the Implementation Guide. The main page of the IG publisher executes the HL7 IG Publisher.

- The **Options** tab allows users to specify options you would like to use for the export.
- The **Validation** tab lists any validation issues returned by the FHIR server's \$validate operation. These validation issues may vary depending on the implementation of the FHIR server.
- The **Status** tab allows users to monitor the status of the publication process (the FHIR IG Publisher)

To publish your IG, follow these steps:

1. From the top navigation menu, click **Publish**.
2. Select the options for your publication.

The screenshot shows the 'Publish' page in the Trifolia-on-FHIR application. The 'Options' tab is selected, displaying various configuration options for the FHIR IG Publisher. The options include:

- Use latest FHIR IG Publisher?**: Set to 'Yes'.
- Use terminology server?**: Set to 'Yes'.
- Download?**: Set to 'No'.
- Output Format**: Set to 'JSON'.
- Template Type**: Set to 'Official'.
- Template**: Set to 'HL7 FHIR Template'.
- Version**: Set to 'current'.

- a. In the **Use latest FHIR IG Publisher** field, select **Yes**.

**Note:** HL7 requires the latest IG publisher for ballots. The IG publisher updates occasionally include bugs, resulting in errors. You can select **No** pending resolution of the bugs.

- b. In the **Use terminology server** field,
  - Select **No** if the IG does not require connection to the terminology server. This speeds up the publishing process.
  - Select **Yes** if the IG needs to connect to the terminology server to check terminology bindings.
- c. In the **Download?** field, select **Yes** to download the entire published package to your computer when complete.

3. Click **Publish**.

**Note:** This process can be time-consuming, depending on the size of your IG. Once published, click the link on the status screen to view the IG or Select **Browse/Edit** then select **View Implementation Guide**.

## Pre-Publishing Errors

### *Common Pre-publishing Errors*

A pre-publishing error has occurred when the string of codes results in IG Publisher finished with code 1. This means that there's an error in your implementation guide (IG) that needs to be resolved before Trifolia-on-FHIR can publish the IG. This guide provides the steps for resolving these pre-publishing error examples:

- IG Version
- Generating Snapshot
- IG Package ID
- Terminology server connection
- Dependency

For other pre-publishing errors, go to [chat.fhir.org](https://chat.fhir.org).

### *Identifying the Error*

To identify the type of error, follow these steps:

1. In the Publishing Status text, find the phrase, "Publishing Content Failed."

## Publish

This page executes the HL7 IG Publisher. The HL7 IG Publisher tool is not developed as part of Trifolia-on-FHIR; it is developed by HL7. Questions related to the execution of the HL7 IG Publisher should be directed to [chat.fhir.org](https://chat.fhir.org).

Options Validation (71) Status

```
--ig
XXXXXX-1ke7RiURop091/ig.ini
dir = /ToF/apps/server, path = /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
Run time = Wednesday, May 6, 2020 3:50:17 PM GMT (2020-05-06T15:50:17Z)
Package Cache: XXXX/.fhir/packages (00:00.0080)
Load Template from hl7.fhir.template@current (00:01.0279)
Load Template from hl7.base.template@current (00:02.0305)
Load Template from fhir.base.template@current (00:02.0343)
onLoad.findSheets:
onLoad.updateIg:
[xml] Processing XXXXXX-1ke7RiURop091/template/onload-ig-working.xml to /tmp/tmp-1ke7RiURop091/template/onload-ig-updated.xml
[xml] Loading stylesheet XXXXXX-1ke7RiURop091/template/scripts/onload.xslt
ImplementationGuide.version must be specified
[xml] XXXXXX-1ke7RiURop091/template/scripts/onload.xslt:15: Fatal Error! Processing terminated by xsl:message at line 15 in onload.xslt
Publishing Content Failed: Fatal error during transformation using XXXXXX-1ke7RiURop091/template/scripts/onload.xslt: Processing terminated by xsl:message at line 15 in onload.xslt; SystemID: file:/tmp/tmp-1ke7RiURop091/template/scripts/onload.xslt; Line#: 15; Column#: -1 (00:05.0391)
(00:05.0392)
This error was created by the template (00:05.0393)
IG Publisher finished with code 1
The HL7 IG Publisher failed. The HL7 IG Publisher tool is not developed as part of Trifolia-on-FHIR; it is developed by HL7. If you are still having issues after having reviewed and addressed ToF errors reported in the log above, go to chat.fhir.org to get more information on how to proceed.
```

## 2. Look at the error message that follows this phrase to determine the type of pre-publishing error.

## Publish

This page executes the HL7 IG Publisher. The HL7 IG Publisher tool is not developed as part of Trifolia-on-FHIR; it is developed by HL7. Questions related to the execution of the HL7 IG Publisher should be directed to [chat.fhir.org](https://chat.fhir.org).

Options Validation (71) Status

```
--ig
XXXXXX-1ke7RiURop091/ig.ini
dir = /ToF/apps/server, path = /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
Run time = Wednesday, May 6, 2020 3:50:17 PM GMT (2020-05-06T15:50:17Z)
Package Cache: XXXX/.fhir/packages (00:00.0080)
Load Template from hl7.fhir.template@current (00:01.0279)
Load Template from hl7.base.template@current (00:02.0305)
Load Template from fhir.base.template@current (00:02.0343)
onLoad.findSheets:
onLoad.updateIg:
[xml] Processing XXXXXX-1ke7RiURop091/template/onload-ig-working.xml to /tmp/tmp-1ke7RiURop091/template/onload-ig-updated.xml
[xml] Loading stylesheet XXXXXX-1ke7RiURop091/template/scripts/onload.xslt
ImplementationGuide.version must be specified
[xml] XXXXXX-1ke7RiURop091/template/scripts/onload.xslt:15: Fatal Error! Processing terminated by xsl:message at line 15 in onload.xslt
Publishing Content Failed: Fatal error during transformation using XXXXXX-1ke7RiURop091/template/scripts/onload.xslt: Processing terminated by xsl:message at line 15 in onload.xslt; SystemID: file:/tmp/tmp-1ke7RiURop091/template/scripts/onload.xslt; Line#: 15; Column#: -1 (00:05.0391)
(00:05.0392)
This error was created by the template (00:05.0393)
IG Publisher finished with code 1
The HL7 IG Publisher failed. The HL7 IG Publisher tool is not developed as part of Trifolia-on-FHIR; it is developed by HL7. If you are still having issues after having reviewed and addressed ToF errors reported in the log above, go to chat.fhir.org to get more information on how to proceed.
```

**Note:** IG Version and IG Package ID errors have similar error codes. Look at the numbers and letters bolded for emphasis to determine which error is occurring.

Error Message	Type of Pre-publishing Error
Fatal error during transformation using XXXXXX- <b>1t1UQG0X5t3Cm</b>	<a href="#">IG Version</a>
Error generating snapshot	<a href="#">Generating Snapshot</a>
Fatal error during transformation using XXXXX- <b>11BB6wlpjUFIB</b>	<a href="#">IG Package ID</a>
Unable to connect to terminology server	<a href="#">Terminology Server Connection</a>
The package “hl7.fhir.us” has no entry on the current build server	<a href="#">Dependency</a>

## IG Version

### IG Version Error

To resolve an IG Version error, follow these steps:

1. Open the IG in Trifolia-on-FHIR IG Editor
2. In the **General** tab, in the **Version** field, enter the IG version, e.g. 1.0.0.

**Implementation Guide (R4)**  
**CCDaonFHIRR4\_IG\_TEST**

General | Narrative/Pages | Resources | Other | Permissions | Validation | RAW | History

ID: FHIRR4TEST Change URL: http://www.test.com/implementationGuide/FHIRR4TEST

Name: CCDaonFHIRR4\_IG\_TEST Title: C-CDA on FHIR R4 Test

Publisher: Lantana Consulting Group Version: 1.0.0 Package ID: hl7.fhir.us.ccdar4test1

FHIR Version: 4.0.1 Status: Draft HL7 Work Group: Publishing

3. Click **Save** and then publish.

## Generating Snapshot

### Generating Snapshot Errors

Generating Snapshot is a category of errors that have different root causes. This section provides the steps to resolve two types of Generating Snapshot errors: Base Definition and Authorization.

To determine the type of Generating Snapshot error, look at code next to the phrase, “Publishing Content Failed.”

- **Base Definition error code:** Cannot find or generate snapshot for base definition
- **Authorization error code:** Unable to generate snapshot for [resource URL]  
 Example resource URL: <https://trifolia-fhir-dev.lantanagroup.com/structure-definition/ccda-authorization-extension>

### Base Definition

To resolve a Base Definition error, follow these steps:

1. In the Publishing Status text, look at the code to identify the resource causing the error. The screenshot shows an example of a resource that might be causing the error.

**Publish**

This page executes the HL7 IG Publisher. The HL7 IG Publisher tool is not developed as part of Trifolia-on-FHIR; it is developed by HL7. Questions related to the execution of the HL7 IG Publisher should be directed to [chat.fhir.org](https://chat.fhir.org).

Options | Validation (216) | Status

Initialization complete (00:16.0233)  
 Load Content (00:16.0234)  
 Processing Conformance Resources (00:16.0392)  
 Publishing Content failed: Error generating snapshot for elTSS Person Logical Model(elTSSPersonModel): Cannot find or generate snapshot for base definition (null from http://hl7.org/fhir/us/elTSS/StructureDefinition/elTSSPersonModel) (00:16.0488)  
 Use -? to get command line help (00:16.0481)  
 Stack Dump (for debugging): (00:16.0481)  
 java.lang.Exception: Error generating snapshot for elTSS Person Logical Model(elTSSPersonModel): Cannot find or generate snapshot for base definition (null from http://hl7.org/fhir/us/elTSS/StructureDefinition/elTSSPersonModel)  
 at org.hl7.fhir.igtools.publisher.Publisher.generateSnapshots(Publisher.java:4181)  
 at org.hl7.fhir.igtools.publisher.Publisher.loadConformance(Publisher.java:3563)  
 at org.hl7.fhir.igtools.publisher.Publisher.createIg(Publisher.java:826)  
 at org.hl7.fhir.igtools.publisher.Publisher.execute(Publisher.java:688)  
 at org.hl7.fhir.igtools.publisher.Publisher.main(Publisher.java:7310)  
 Caused by: java.lang.Exception: Cannot find or generate snapshot for base definition (null from http://hl7.org/fhir/us/elTSS/StructureDefinition/elTSSPersonModel)  
 at org.hl7.fhir.igtools.publisher.Publisher.generateSnapshot(Publisher.java:4204)  
 at org.hl7.fhir.igtools.publisher.Publisher.generateSnapshots(Publisher.java:4179)  
 ... 4 more  
 IG Publisher finished with code 1  
 The HL7 IG Publisher tool is not developed as part of Trifolia-on-FHIR; it is developed by HL7. Questions related to the execution of the HL7 IG Publisher should be directed to <https://chat.fhir.org/narrow/stream/179252-IG-creation>.

☒ Automatically scroll to bottom when status is updated

2. In Browse-Profiles, check that a profile exists for the resource in the error.

**Structure Definition**

Test\_Extension

General | Narrative | Additional | Mapping(s) | Elements | Permissions | Validation (0) | RAW | History

URL: https://trifolia-on-fhir.lantanagroup.com/StructureDefinition/extension-ig-page-content Status: Draft ID: extension-ig-page-content

Name: Test\_Extension Version: Type: Extension

Title: Test Implementation Guide Page Content Extension FHIR Version: Base Definition: http://hl7.org/fhir/StructureDefinition/Extension

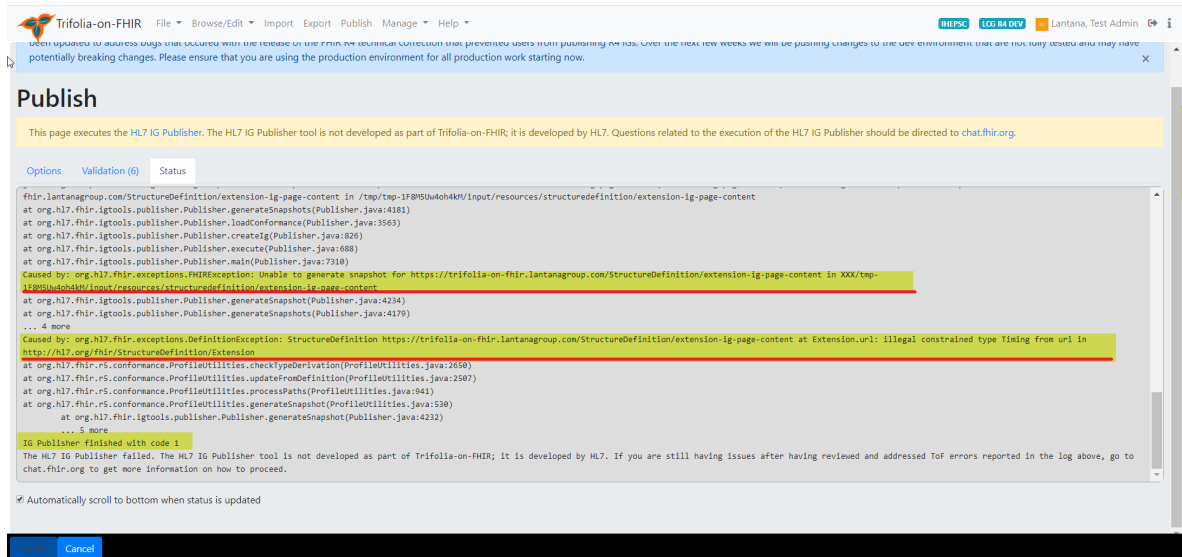
3. In the [HL7 resource list](#), search for the correct base definition URL.

4. In the **Base Definition** field, paste the correct base definition URL.
5. Click **Save** and then publish.

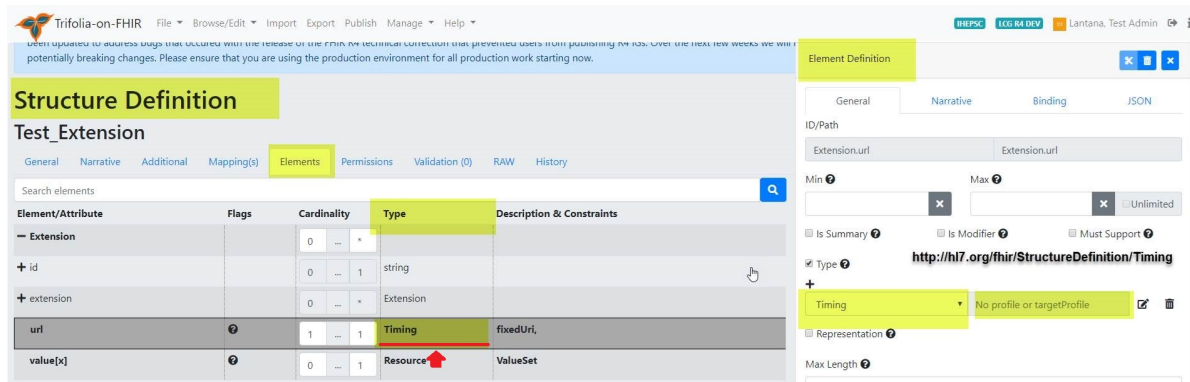
## Authorization

To resolve an Authorization error, follow these steps:

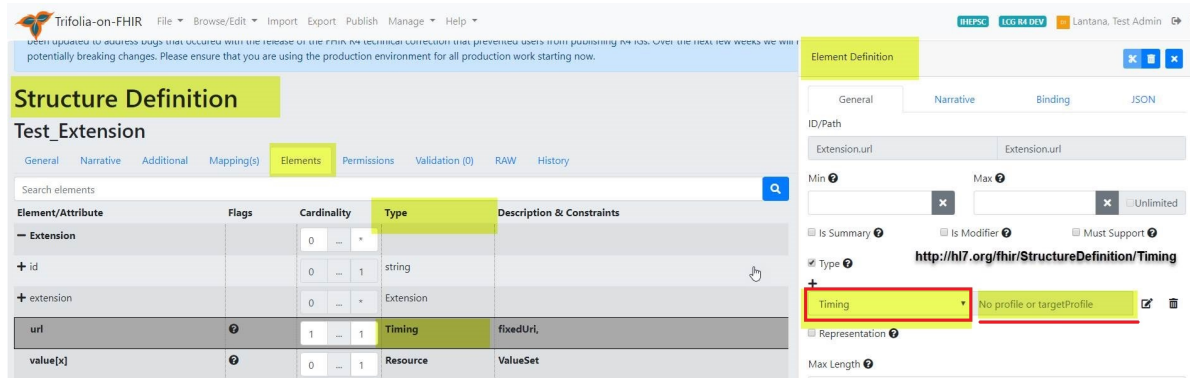
1. Find the profile causing the error by looking in the Publishing Status text for the phrase “Caused by.”



2. Using the Profile Editor, open the profile causing the error identified in step 1.
3. In the **Elements** tab, confirm the **Type** for each constrained element is correct. If not, correct any errors.



4. For each constrained element, confirm the Type profile URL is correct. If not, correct any errors.

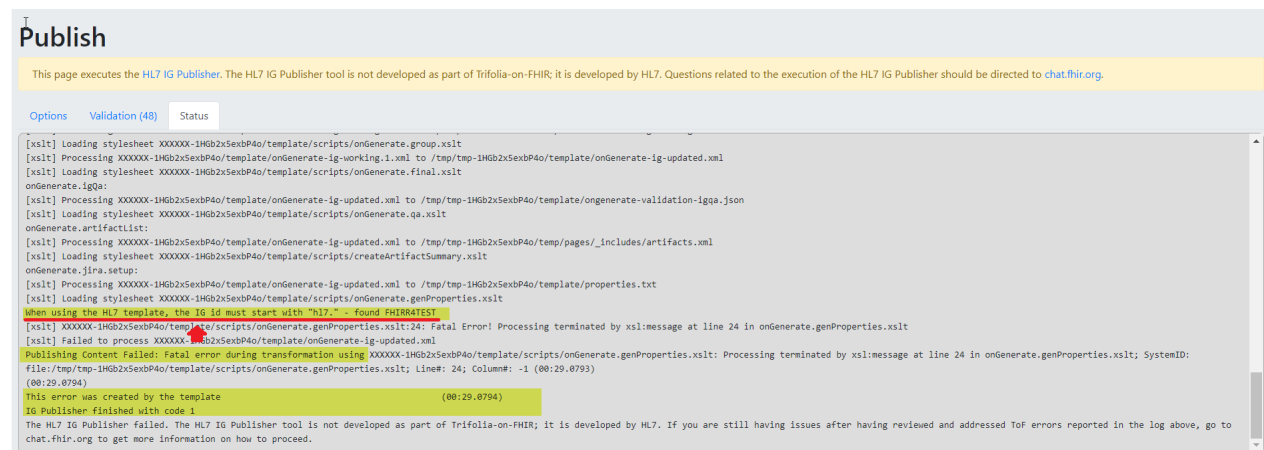


5. In the [HL7 resource list](#), search for the correct element type URL.
6. In the **Element Definition** form, in the **Type URL** field, paste the correct element type URL.
7. Click **Save** and then publish.

## IG Package ID

### IG Package ID Errors

This example shows the text indicating the cause of this IG Package ID error: the ID doesn't begin with hl7.



To resolve the IG Package ID error shown in this example, follow these steps:

1. In the IG Editor, open the IG.
2. In the General tab, in the **Package ID** field, enter a package ID that begins with “hl7.” (e.g., hl7.fhir.us.mytestig).

3. Click **Save** and then publish.

## Terminology Server Connection

### Terminology Server Connection Error

The Terminology Server Connection error means that the VSAC terminology server is temporarily down.

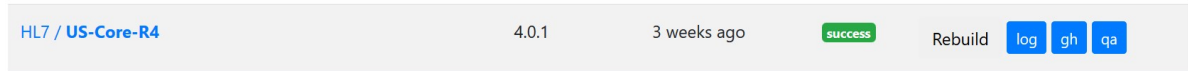
If:	Then:
The IG requires a VSAC check,	Wait and try to publish again in 1 to 2 hours.
The IG does not require a VSAC check,	In the <b>Use Terminology Server</b> option, select <b>No</b> .

## Dependency

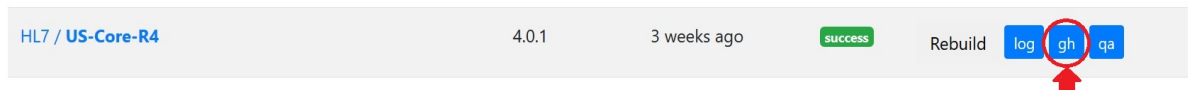
### Dependency Error

To resolve a Dependency error, follow these steps:

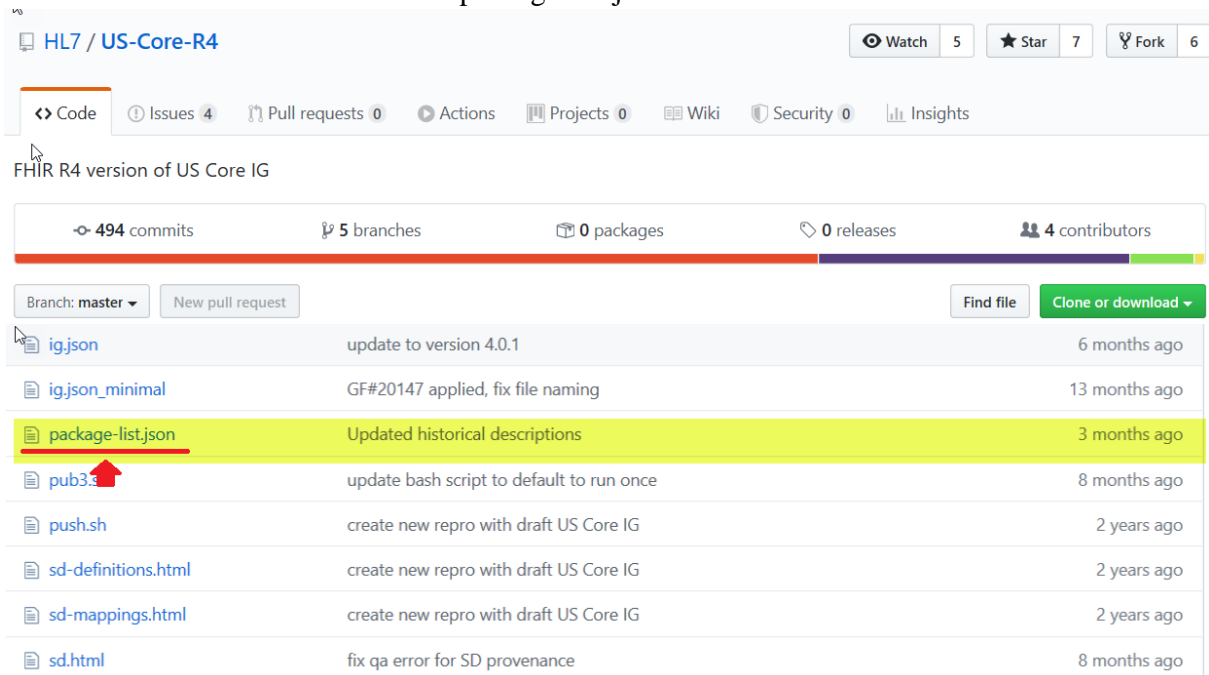
1. Go to <https://fhir.github.io/auto-ig-builder/builds.html>.
2. Search for the HL7 / US-Core-R4 in the list of IGs.



3. Click on the **gh** option.

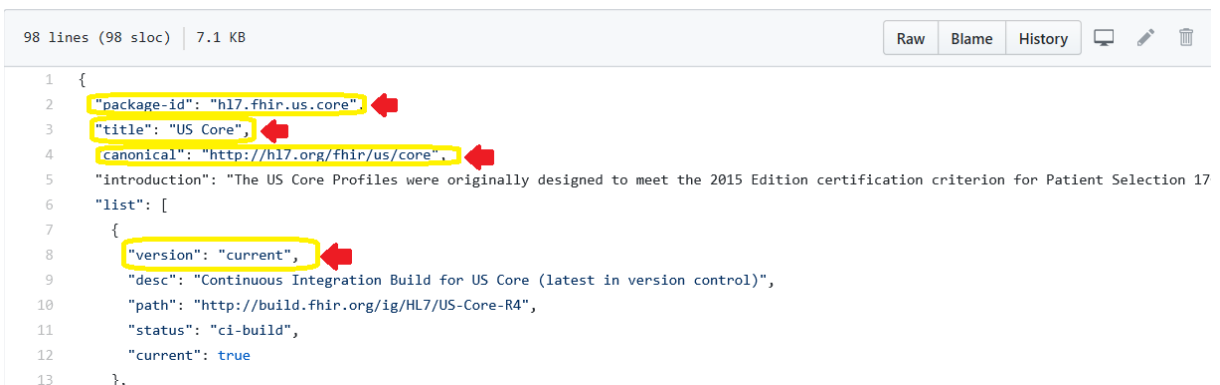


4. Double-click on the US-Core-R4/package-list.json file.



You will need the information from these fields:

- Canonical value
- Package-id value
- Title value
- Version value



5. In the IG Editor, open the IG.
6. In the **General** tab, scroll down to the **Dependencies** section
7. In the **URI** field, enter the canonical value from the package-list.json.



8. In the **Package ID** field, enter the package-id value from the package-list.json file.
9. In the **Name** field, enter the title value from the package-list.json.
10. In the **Version** field, enter the version value from the package-list.json.

11. Click **Save** and then publish.

## Export

## Export

Select Export in the tabbed tool bar on the top of the screen.

In order to export artifacts, you must first select the Implementation Guide(IG).

The Export page contains form fields that allow users to specify the details of their exports. Users can export the Implementation Guides (IGs) saved under the Browse/Edit tab at the top right side of the screen. Users can export IGs as bundles or HTML with the IG Publisher. Once the form fields are complete, select the Export button on the left side of the scrolling tab at the bottom of the screen.

1. The "IG Publisher Package" generates a package using the published version of the IG to be used with the FHIR IG Publisher.
  - Users can select either JSON or XML format for the output
  - The "FHIR IG Publisher JAR" can be included in the export package
  - The Export tool will take a few minutes to process. The length of time is correlated with the size of the export.
  - After completing the process, the export will automatically download to users' computers in a compressed folder, and the user is prompted to download a ZIP package of all files necessary to execute the FHIR IG publisher.
  - When the IG Publisher is executed, the output from the IG Publisher is copied to a public location in Trifolia-on-FHIR for preview.
  - HTML exports produce a package (ZIP file) for use with the FHIR [IG Publisher](#).
2. The "Bundle" export format is a Bundle of all resources referenced by the selected implementation guide, including the implementation guide resource
  - Bundle exports produce a single download (pretty quickly) as a single XML file. This XML file is a FHIR [Bundle](#) that can be used to import the resources for the implementation guide in another FHIR environment.
  - Users can select the type of output, JSON or XML, the bundle will be exported in.
3. The "GitHub" export format places all the resources within the IG into the specified GitHub repository/branch
  - Only resources that have a repository, branch and path will be exported to GitHub. At least one resource must have GitHub location information specified to export
  - If the file already exists in GitHub, it will be completely overwritten by this export
  - When this tab is selected from the Export page, the user will be prompted to enter their GitHub credentials. If the user does not log into GitHub initially, the "Login to GitHub" button can be selected before the export is commenced.
  - After a user has selected an Implementation Guide, the user must enter a commit message that will be associated with the IG on GitHub.

## Import

### Import

ToF allows users to import files, text, VSAC and GitHub content.

- **File** imports allow users to drag-and-drop FHIR resources (e.g., StructureDefinitions, ValueSets, CodeSystems) and Excel-based value sets directly from your computer to ToF.
  - Users can drag-and-drop files from their computers File Explorer directly into the Import screen, or select the "Click to Select" link to select the files they wish to import.
  - Once the user has selected the files in Explorer, the list of files to be imported will be displayed in the Import screen.
  - Users can click the **trash** icon to remove file(s) from the list of files to be imported.
  - Imported resources are sent directly to the ToF server as a transaction bundle. ToF displays an excerpt of the JSON or XML bundle of the file prior to uploading.
  - The **Results** tab displays the outcome of importing from the **Files** tab
- **Text** imports allows users to copy/paste JSON or XML content directly into Trifolia-on-FHIR to have it imported.
- **VSAC** imports allows users to import value sets and code systems directly from VSAC into ToF.
  - Imported value sets and code systems can be referenced by Structure Definition resources.
  - The VSAC imports require users' VSAC credentials, which are not persisted on the ToF server. If users select 'Remember VSAC Credentials,' the tool will store this information as cookies in the users' browser.
- The **GitHub** tab allows users to import resources directly from GitHub into ToF.
  - After selecting the "GitHub" tab, the user will be prompted to login to GitHub using their GitHub credentials.

**Note:** Users who upload more than 20 resources at once may experience a timeout error notification. In the event of a timeout error notification, users should reduce the size of the resource import. Users can edit resource numbers based on individual needs.

### Excel Value Sets

In the "Files" tab you may import excel spreadsheets (XLSX) that represent value sets. The spreadsheet document must have these columns, in this specific order (note that that row 1 is expected to contain the below column headers, not the first value in the ValueSet):

- **ID** - This is the ID of the value set. This value in this column is repeated for each code/row. The value must be formatted as a [valid ID](#). Ex: 2.16.840.1.113883.42.3.1
- **Name** - The name of the value set. This value in this column is repeated for each code/row.
- **URL** - The URL of the value set. This value in this column is repeated for each code/row. Ex: `http://some.com/test/fhir/valueset`
- **Code** - The code representing the concept. Ex: 9000000000000073002
- **Display** - The display representing the concept. Ex: "Sufficiently defined concept definition status"
- **System** - The system URL that owns/maintains the code. Ex: `http://snomed.info/sct`

## GitHub Integration

### Authentication

The import and export screens both contain options for GitHub. As soon as the GitHub option is selected in either screen, you are prompted to login with your GitHub credentials. Once logged in, your GitHub authentication token is stored in cookies so that you do not have to login every time you select "GitHub" under the import/export screens.

After you have logged into GitHub, a GitHub icon appears in the top-right corner of the all screens. When clicked, this icon logs you out of GitHub within ToF.

ToF uses a pop-up window to authenticate with GitHub. If your browser blocks the pop-up window, ToF will not be able to authenticate with GitHub and you will receive an error.

*NOTE: If you select "Remember Me" when logging into GitHub, and want to change to a different GitHub user, you will need to open [github.com](https://github.com) in a separate window and logout of your GitHub account, then have ToF forget the GitHub login using the GitHub icon mentioned above. Once this is done, when you attempt to login to GitHub via ToF, it will ask you which account you would like to use with GitHub.*

### Work Flow

The work flow within ToF for GitHub is to

1. Import resources from a GitHub repository into the selected FHIR server
2. Edit the resources using ToF
3. Export the resources back to the GitHub repository after they have the desired changes

### Extensions

When importing resources, two extensions are added to each resource representing the location within GitHub for where the resource came from. This enables ToF to know where in GitHub to export the resources back to.

If you are exporting *new* resources to GitHub, these extensions will not yet exist and you will need to specify where the resources should be stored during the export (which will create the two extensions on the resource).

### Limitations

- Trifolia only exports the individual resources associated with the implementation guide, and does not include the entire IG Publication package. For example, the "framework" (html templates) folder is not included in the export.
- Trifolia only allows importing FHIR resources. Trifolia-on-FHIR allows the user to select any JSON or XML file from GitHub. If the user selects an XML or JSON file that is not a FHIR resource, the import will fail.
- GitHub does not allow retrieving/updating very large files. For example, if attempting to import/export a large ValueSet resource, GitHub may fail with a "Payload too large" error.

### Signing Out

When you sign out of GitHub within Trifolia, this clears your GitHub session only within Trifolia. GitHub maintains its own session within your browser. To sign out of GitHub entirely, you will need to go to [github.com](https://github.com) and click "Sign out".

## Validation

### Validation

Trifolia-on-FHIR uses three validation methods to provide as much feedback to IG authors as possible:

1. **Real-time UI Validation**

This validation checks the base FHIR specification requirements (i.e., cardinality, terminology bindings, value set requirements). This validation occurs as each field in ToF is changed to update and render to the user in real-time.

2. **Pre-publish Validation**

When publishing an implementation guide from the "Publish" screen, the FHIR server's [\\$validate operation](#) is executed for each resource in the implementation guide. This is specific to the FHIR server in Trifolia-on-FHIR for this IG (e.g., HAPI, the FHIR server instance default).

Please see the [FAQ](#) page, *Publishing with the FHIR IG Publisher* section for examples of errors and steps to resolve.

3. **HL7 IG Publisher Validation**

The FHIR IG Publisher executes this validation step automatically *during* the publishing process. Validation checks for relationships between all resources and pages within this IG package. This includes all applicable IG resources, profiles, extensions, value sets, etc. FHIR IG Publisher validation also validates HTML links within the package. You cannot execute this validation step externally/independently of the publish process for an entire IG.

## Walk-through

---

The purpose of this page is to guide new users through Trifolia-on-FHIR:

1. Create account and Login
2. Select FHIR Release version (gear icon in top right)
3. Create new Implementation Guide
  - Option A: Create IG from scratch. Navigate to Browse Implementation Guides > click the "plus" + button at top IG list/table.
  - Option B: Import IG from a file. Import IG.xml from your computer ("Import" button at top and either drag-and-drop the IG.xml file into the "Files" tab or copy/paste the contents of IG.xml into the second tab).
4. Modify IG. Be sure to always Save (bottom left)
 

**NOTE:** When creating and ordering pages, you can only pivot a page's position with pages that are siblings with one another. Pages that are children of the page you're trying to move or children of a page that's not the parent of the page being moved can not be swapped.
5. Create/import additional templates/profiles
  - Option A: Create Profile from scratch. Navigate to Browse Templates/Profiles > click the "plus" + button at top of Profile list/table.
  - Option B: Import profiles from directories on computer
6. Modify and constrain the templates/profiles to use case
7. Resolve all Validation errors and warnings on Validation (tab) within each profile
8. Export selected IG package. Suggested settings for initial export:
  1. Export Format: HTML (IG publisher)
  2. Run the IG Publisher: Yes
  3. Run the latest version of the IG Publisher: No
  4. Use terminology server: Yes/No (Suggest No if IG uses large standard codesets)  
Selecting Yes will verify applicable value sets and code systems externally
  5. Download: Yes
  6. Output format: XML
9. Confirm build logs against CI-publisher on Zulip > Notifications

## Glossary

Acronym	Definition
ToF	Trifolia-on-FHIR
FHIR	Fast Healthcare Interoperability Resources
VSAC	Value Set Authority Center
IG	Implementation Guide
VS	Value Set
CS	Code System

## FAQ

### The published implementation guide shows a lot of messages about link's not having a href value

The official HL7 templates use the first contact in the implementation guide as the contact for the Copyright information in the footer. Ensure you have at least one contact in your implementation guide to stop seeing these errors.

### My implementation guide has pages, but the table of contents is empty?

On the **Table of Contents** page, in the **Auto Generate Table of Contents?** field, select **Yes**.

### Why is the Home Screen stuck on Loading?

Based on your web browser storage settings, Trifolia-on-FHIR (ToF) may "hang" or "stall" at the Loading page with the orange circle. Clearing browser history typically resolves this issue.

Follow these steps based on the web browser you use to access ToF; if clearing your browser history does not resolve the issue, submit a ticket and specify the browser you use.

#### Chrome:

1. Select **Options**, the three vertical dots to right of the URL address bar within Chrome, and select **History**.
2. From the menu, select **History**.
3. On the **History** page, from the left pane, select **Clear browsing data**.
4. Refresh the page.

#### Internet Explorer:

1. Select the gear icon (to the right of the URL address bar within IE) and select **Internet Options**.
2. On the **General** tab, under **Browsing History**, click **Delete**.
3. Click **Apply**.

#### Safari:

1. Select **Develop** and then select **Show JavaScript Console** (or Option-Command-C).
2. Select the **Storage** tab.
3. Highlight Local Storage - trifolia-fhir.lantanagroup.com and Application Cache.

4. Select each of the key values, and click **Delete** to remove each of them.
5. After deleting each key/value pair, refresh the page.

**Microsoft Edge:**

1. Select the three horizontal dots (to the right of the URL address bar within Edge).
2. Select **History** and select **Clear Browsing Data** (Ctrl + Shift + Delete).