

---

# **Galaxy Tool Generator Documentation**

**Ming Chen, Bradford Condon**

**Nov 16, 2018**



---

## Contents:

---

<b>1</b>	<b>Quick Start Guide</b>	<b>1</b>
1.1	Launch GTG . . . . .	1
<b>2</b>	<b>User's Guide</b>	<b>3</b>
2.1	Launching GTG . . . . .	3
2.2	Build Tool XML . . . . .	3
2.3	More examples . . . . .	20
<b>3</b>	<b>Developer Guide</b>	<b>21</b>
<b>4</b>	<b>What is Galaxy Tool Generator (GTG)?</b>	<b>23</b>



# CHAPTER 1

---

## Quick Start Guide

---

This repository builds a Docker image that can be used to quickly launch the GTG web application for Galaxy tool development.

To get necessary docker images:

```
docker pull mingchen0919/gtgdocker
docker pull bgruening/galaxy-stable:17.09
```

**Warning:** This documentation is under active construction and should be completed by November 17th 2018, so please check back then if you can't find what you need.

## 1.1 Launch GTG

```
wget https://raw.githubusercontent.com/MingChen0919/gtgdocker/master/launch_dev_env.sh
sh launch_dev_env.sh
```

This script will launch a docker container running the GTG app and another container running a Galaxy instance. Login to the Galaxy instance with username **admin** and password **admin** so that you can install tools from tool shed.



## 2.1 Launching GTG

### 2.1.1 Docker

```
` wget https://raw.githubusercontent.com/MingChen0919/gtgdocker/master/
launch_dev_env.sh sh launch_dev_env.sh `
```

This script will launch a docker container running the GTG app and another container running a Galaxy instance. Login to the Galaxy instance with username **admin** and password **admin** so that you can install tools from tool shed.

After running this script, you should see the following directories in your current directory:

```
` gtg_dev_dir/ |— database |— galaxy_tool_repository |— shed_tools `
```

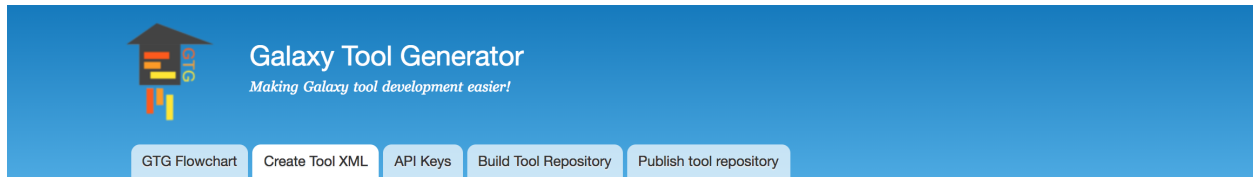
### 2.1.2 Drupal Site

If you want to add the galaxy tool generator to an existing Drupal site...

## 2.2 Build Tool XML

GTG provides three ways to build a Galaxy XML file:

- Aurora Galaxy Tool: this option starts with a template file for developing an Aurora Galaxy Tool.
- Uploaded XML: starts with an uploaded XML.
- From scratch: builds XML from scratch.



### Create Tool XML

**XML file name \***

Galaxy tool XML file name with extension, e.g. [hisat2.xml](#), [bowtie2.xml](#).

**Tool description**

Detailed information about this tool

**Start with a template XML**

☒ Aurora Galaxy Tool

☐ Uploaded XML

☐ From scratch

#### 2.2.1 Start from scratch

For comparison with another software for Galaxy tool development [planemo](https://planemo.readthedocs.io/en/latest/) [<https://planemo.readthedocs.io/en/latest/>](https://planemo.readthedocs.io/en/latest/) ‘\_, I am going to use ‘an example from the planemo use cases. In this example we are going to use GTG to build this `seqtk_seq_2.xml` file.

#### 0. Initialize an XML

- Click **Create Tool XML**
- Enter `seqtk_seq_2.xml` into **XML file name**
- Select **From scratch** and click **Save**



GTG Flowchart

Create Tool XML

API Keys

Build Tool Repository

Publish tool repository

# Create Tool XML

## XML file name \*

seqtk\_seq\_2.xml

Galaxy tool XML file name with extension, e.g. [hisat2.xml](#), [bowtie2.xml](#).

If you are creating an Aurora Galaxy Tool. The XML file name should be [rmarkdown\\_report.xml](#).

## Tool description

Detailed information about this tool

## Start with a template XML

- ☐ Aurora Galaxy Tool  
☐ Uploaded XML  
☒ From scratch

## 1. Create tool component, which is the root component.

```
<tool id="seqtk_seq" name="Convert to FASTA (seqtk)" version="0.1.0">`
```

seqtk\_seq\_2.xml

View/Update XML

XML components

Label	Type	Operations
No Components, add a component below.		
+	tool	tool <input type="button" value="Add"/>

Edit tool component attributes

Attributes

Tag sets for this element.

Tool ID \*

seqtk\_seq

Must be unique across all tools; should be lowercase and contain only letters, numbers, and underscores. It allows for tool versioning and metrics of the number of times a tool is used, among other things. Find the Intergalactic Utilities Commission suggested best practices for this element [here](#)

Name \*

Convert to FASTA (seqtk)

This string is what is displayed as a hyperlink in the tool menu. Find the Intergalactic Utilities Commission suggested best practices for this element [here](#)

Version

0.1.0

This string defaults to `1.0.0` if it is not included in the tag. It allows for tool versioning and should be increased with each new version of the tool. Find the Intergalactic Utilities Commission suggested best practices for this element [here](#)

Hidden

☐ False
 ☐ True

Allows for tools to be loaded upon server startup, but not displayed in the tool menu. This attribute should be applied in the toolbox configuration instead and so should be considered deprecated.

Display interface

☐ False
 ☐ True

Disable the display the tool's graphical tool form by setting this to `false`.

Tool type

☐ data\_source
 ☐ manage\_data

Allows for certain framework functionality to be performed on certain types of tools. Normal tools that execute typical command-line jobs do not need to specify this, special kinds of tools such as [Data Source](#) and [Data Manager](#) tools should set this to have values such as `data_source` or `manage_data`.

Profile

This string specified the minimum Galaxy version that should be required to run this tool. Certain legacy behaviors such as using standard error content to detect errors instead of exit code are disabled automatically if profile is set to any version newer than `16.01`, such as `16.04`.

Workflow compatible

☐ True
 ☐ False

This attribute indicates if this tool is usable within a workflow (defaults to `true` for normal tools and `false` for data sources).

URL method

☐ get
 ☐ put

Only used if `tool_type` attribute value is `data_source` - this attribute defines the HTTP request method to use when communicating with an external data source application (the default is `get`).

Save component

## 2. Create tool->requirements component.

```
<requirements>
  <requirement type="package" version="1.2">seqtk</requirement>
</requirements>
```

### Add tool->requirements component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements				

requirements

tool->requirements

Add

Drag to place requirements under tool

Save

Edit **tool->requirements** component attributes. However, this component does not have any attributes.

Attributes

Tag sets for this element. This component does not have any attributes

Save component

### Add tool->requirements->requirement component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk				

seqtk

tool->requirements->requirement

Add

This component is under requirements

Save

Edit **tool->requirements->requirement** component attributes.

Attributes

Tag sets for this element.

**Type \***  
  
 This value defines the which type of the 3rd party module required by this tool.

**Version**  
  
 For package type requires this value defines a specific version of the tool dependency.

**Package name \***  
  
 A package from the bioconda repository.

Save component

### 3. Create tool->command component

```
<command detect_errors="exit_code"><![CDATA[
    seqtk seq -a '$input1' > '$output1'
]]></command>
```

Add **tool->command** component

Label	Type	Operations
tool	tool	Edit Clone Delete
requirements	tool->requirements	Edit Clone Delete
seqtk	tool->requirements->requirement	Edit Clone Delete
command		tool->command Add

command and requirements components are at the same level

Save

Edit **tool->command** component attributes.

Attributes

Tag sets for this element.

**Detect errors**

- default
- ☒ exit\_code
- aggressive

**Strict**

- True
- False

This boolean forces the `set -e` directive on in shell scripts - so that in a multi-part command if any part fails the job exits with a non-zero exit code. This is enabled by default for tools with `profile=>16.04` and disabled on legacy tools.

**Interpreter**

Older tools may define an `interpreter` attribute on the command, but this is deprecated and using the `$_tool_directory__` variable is superior.

**XML value \*** Enter shell script that goes into the command section

seqtk seq -a '\$input1' > '\$output1'

There are two ways that you can add your shell script to the command section: 1) copy and paste your script to the text area above. 2) create a `seqtk_seq_2.sh` file in the `galaxy_tool_repository` folder and then click the [VIEW/UPDATE XML](#) button twice.

Save component

The **XML value** field in the above web form is used to collect the shell script for the command section. However, there is an easier way to input shell script into the tool XML file. Go to the `gtg_dev_dir/galaxy_tool_repository` and create a `.sh` file. Put the shell script into this file, the content will be automatically integrated into the web form field when the XML webform page is being viewed (see the image below). The `.sh` file should have exact the same base name as the XML file. For example, in this example, the XML file is `seqtk_seq_2.xml`, then the `.sh` file should be `seqtk_seq_2.sh`.

## seqtk\_seq\_2.xml

View/Update XML

XML components

Click this button twice to integrate the content from  
seqtk\_seq\_2.sh into the command section field

Submitted by Anonymous (not verified) on Fri, 11/09/2018 - 21:10

Start

Complete

## ▼ tool

```
<tool id="seqtk_seq" name="Convert to FASTA (seqtk)" version="0.1.0" ></tool>
```

## ▼ tool &gt; requirements

```
<requirements ></requirements>
```

## ▼ tool &gt; requirements &gt; requirement

```
<requirement type="package" version="1.2" >seqtk</requirement>
```

## ▼ tool &gt; command

```
<command detect_errors="exit_code" >![CDATA[
```

```
seqtk seq -a '$input1' > '$output1'
```

```
]]></command>
```

## 4. Create tool-&gt;inputs component

```
<inputs>
  <param type="data" name="input1" format="fastq" />
</inputs>
```

## Add tool-&gt;inputs component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs		tool->inputs	Add	

Save

## Edit tool-&gt;inputs component attributes

In this example, we don't need to edit any attributes for this component.

Attributes
Tag sets for this element.

Action
URL used by data source tools.

Check values
☐ True  
☐ False  
Set to **false** to disable parameter checking in data source tools.

Method
☐ get  
☐ put  
Data source HTTP action (e.g. **get** or **put**) to use.

Target
UI link target to use for data source tools (e.g. **\_top**).

Nginx upload
☐ True  
☐ False  
This boolean indicates if this is an upload tool or not.

Save component

## Add tool->inputs->param(type: data) component

Label	Type	Operations
+ tool	tool	<a href="#">Edit</a> <a href="#">Clone</a> <a href="#">Delete</a>
+ requirements	tool->requirements	<a href="#">Edit</a> <a href="#">Clone</a> <a href="#">Delete</a>
+ seqtk	tool->requirements->requirement	<a href="#">Edit</a> <a href="#">Clone</a> <a href="#">Delete</a>
+ command	tool->command	<a href="#">Edit</a> <a href="#">Clone</a> <a href="#">Delete</a>
+ inputs	tool->inputs	<a href="#">Edit</a> <a href="#">Clone</a> <a href="#">Delete</a>

Save

## Edit tool->inputs->param(type: data) component attributes

Attributes
Tag sets for this element.

Name \*  

Name for this element. This **name** is used as the Cheetah variable containing the user-supplied parameter name in **command** and **configfile** elements. The name should not contain pipes or periods (e.g. **.**). Some "reserved" names are **REDIRECT\_URL**, **DATA\_URL**, **GALAXY\_URL**.

Argument  

If the parameter reflects just one command line argument of a certain tool, this tag should be set to that particular argument. It is rendered in parenthesis after the help section, and it will create the name attribute from the argument attribute by stripping the dashes (e.g. if **argument**="**--sensitive**" then **name**="**sensitive**" is implicit).

Label  

The attribute value will be displayed on the tool page as the label of the form field (label="**Sort Query**").

Help  
Short bit of text, rendered on the tool form just below the associated field to provide information about the field. Find the Intergalactic Utilities Commission suggested best practices for this element here.

Optional
☐ False  
☐ True  
If **false**, parameter must have a value. Defaults to "false".

Refresh on change
☐ False  
☐ True  
Force a reload of the tool panel when the value of this parameter changes to allow **code** file processing. See deprecation-like notice for **code** blocks.

Format  

Only if **type** attribute value is **data** or **data\_collection** - the list of supported data formats is contained in the **/config/datatypes\_conf.xml.sample** file. Use the file extension.

Multiple
☐ True  
☐ False  
Allow multiple values to be selected. Valid with **data** and **select** parameters.

Save component

## 5. Create tool->outputs component

```
<outputs>
  <data name="output1" format="fasta" />
</outputs>
```

### Add tool->outputs component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs			

tool->outputs

Save

### Edit tool->outputs component attributes

In this example, we don't need to edit any attributes for this component.

Attributes

Tag sets for this element.

Provided metadata style

Style used for tool provided metadata file (i.e. galaxy.json) - this can be either "legacy" or "default". The default of tools with a profile of 17.09 or newer are "default", and "legacy" for older and tools and tools without a specified profile. A discussion of the differences between the styles can be found at <https://github.com/galaxyproject/galaxy/pull/4437>.

Provided metadata file

Path relative to tool's working directory to load tool provided metadata from. This metadata can describe dynamic datasets to load, dynamic collection contents, as well as simple metadata (e.g. name, dbkey, etc...) and datatype specific metadata for declared outputs. More information can be found [here](#). The default is galaxy.json.

Save component

## 6. Create tool->tests component

```
<tests>
  <test>
    <param name="input1" value="2.fastq"/>
    <output name="output1" file="2.fasta"/>
  </test>
</tests>
```

### Add tool->tests component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs	Edit	Clone	Delete
+ tests	tool->tests			

Save

### Edit tool->tests component attributes

This component does not have attributes

Attributes
Tag sets for this element.

Save component

### Add tool->tests->test component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs	Edit	Clone	Delete
+ tests	tool->tests	Edit	Clone	Delete
+ test	tool->tests->test			

Save

### Edit tool->tests->test component attributes

This component does not have attributes

Attributes
Tag sets for this element.

Save component

### Add tool->tests->test->param component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs	Edit	Clone	Delete
+ tests	tool->tests	Edit	Clone	Delete
+ test	tool->tests->test	Edit	Clone	Delete
+ test	tool->tests->test->param			

Save

### Edit tool->tests->test->param component attributes

Attributes
Tag sets for this element.

Name \*
2.fastq
This value must match the name of the associated input parameter (param).

Value

File type

This attribute name should be included only with parameters of type data for the tool. If this attribute name is not included, the functional test framework will attempt to determine the data type for the input dataset using the data type sniffers.

dbkey

Specifies a dbkey value for the referenced input dataset. This is only valid if the corresponding parameter is of type data.

Save component

### Add tool->tests->test-output component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs	Edit	Clone	Delete
+ tests	tool->tests	Edit	Clone	Delete
+ test	tool->tests->test	Edit	Clone	Delete
+ input1	tool->tests->test->param	Edit	Clone	Delete

## Edit tool->tests->test-output component attributes

Attributes

Tag sets for this element.

Name

This value is the same as the value of the **name** attribute of the **data** tag set contained within the tool's **outputs** tag set.

File

If specified, this value is the name of the output file stored in the target **test-data** directory which will be used to compare the results of executing the tool via the functional test framework.

ftype

If specified, this value will be checked against the corresponding output's data type. If these do not match, the test will fail.

Sort

☐ true
☒ false

This flag causes the lines of the output to be sorted before they are compared to the expected output. This could be useful for non-deterministic output.

Value

An alias for **file**

md5

If specified, the target output's MD5 hash should match the value specified here. For large static files it may be inconvenient to upload the entry file and this can be used instead.

checksum

If specified, the target output's checksum should match the value specified here. This value should have the form **hash\_type:hash\_value** (e.g. **sha1:8136d7ca0f46ed7abac98f82e36cfaddb2ac041**). For large static files it may be inconvenient to upload the entry file and this can be used instead.

Compare

☒ diff

## 7. Create tool->help component

```
<help><![CDATA[

Usage:  seqtk seq [options] <in.fq>|<in.fa>
Options: -q INT      mask bases with quality lower than INT [0]
         -X INT      mask bases with quality higher than INT [255]
         -n CHAR     masked bases converted to CHAR; 0 for lowercase [0]
         -l INT      number of residues per line; 0 for 2^32-1 [0]
         -Q INT      quality shift: ASCII-INT gives base quality [33]
         -s INT      random seed (effective with -f) [11]
         -f FLOAT     sample FLOAT fraction of sequences [1]
         -M FILE     mask regions in BED or name list FILE [null]
         -L INT      drop sequences with length shorter than INT [0]
         -c          mask complement region (effective with -M)
         -r          reverse complement
         -A          force FASTA output (discard quality)
         -C          drop comments at the header lines
         -N          drop sequences containing ambiguous bases
         -1          output the 2n-1 reads only
         -2          output the 2n reads only


```

(continues on next page)



(continued from previous page)

```

-V      shift quality by '(-Q) - 33'
-U      convert all bases to uppercases
-S      strip of white spaces in sequences
]]></help>

```

### Add tool->help component

Label	Type	Operations		
tool	tool	Edit	Clone	Delete
requirements	tool->requirements	Edit	Clone	Delete
seqtk	tool->requirements->requirement	Edit	Clone	Delete
command	tool->command	Edit	Clone	Delete
inputs	tool->inputs	Edit	Clone	Delete
input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
outputs	tool->outputs	Edit	Clone	Delete
tests	tool->tests	Edit	Clone	Delete
test	tool->tests->test	Edit	Clone	Delete
input1	tool->tests->test->param	Edit	Clone	Delete
output1	tool->tests->test->output	Edit	Clone	Delete

### Edit tool->help component attributes

Attributes

Tag sets for this element.

**XML value \***

```

Usage: seqtk seq [options] <in.fq>|<in.fa>
Options: -q INT    mask bases with quality lower than INT [0]
        -x INT    mask bases with quality higher than INT [255]
        -n CHAR    masked bases converted to CHAR; 0 for lowercase [0]
        -l INT    number of residues per line; 0 for 2^32-1 [0]
        -Q INT    quality shift: ASCII-INT gives base quality [33]
        -s INT    random seed (effective with -f) [11]
        -f FLOAT   sample FLOAT fraction of sequences [1]
        -M FILE    mask regions in BED or name list FILE [null]
        -L INT    drop sequences with length shorter than INT [0]
        -c         mask complement region (effective with -M)
        -r         reverse complement
        -A         force FASTA output (discard quality)
        -C         drop comments at the header lines
        -N         drop sequences containing ambiguous bases
        -1         output the 2n-1 reads only
        -2         output the 2n reads only
        -V         shift quality by '(-Q) - 33'
        -U         convert all bases to uppercases
        -S         strip of white spaces in sequences

```

This tag set includes all of the necessary details of how to use the tool. This tag set should be included as the next to the last tag set, before citations, in the tool config. Tool help is written in **reStructuredText**. Included here is only an overview of a subset of features. For more information see <http://docutils.sourceforge.net/docs/ref/rst/restructuredtext.html>.

## 8. Create tool->citations component

```

<citations>
  <citation type="bibtex">
@misc{githubseqtk,
  author = {LastTODO, FirstTODO},
  year = {TODO},
  title = {seqtk},
  publisher = {GitHub},
  journal = {GitHub repository},
  url = {https://github.com/lh3/seqtk},
}</citation>
</citations>

```

### Add tool->citations component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs	Edit	Clone	Delete
+ tests	tool->tests	Edit	Clone	Delete
+ test	tool->tests->test	Edit	Clone	Delete
+ input1	tool->tests->test->param	Edit	Clone	Delete
+ output1	tool->tests->test->output	Edit	Clone	Delete
+ help	tool->help	Edit	Clone	Delete

citations

tool-> citations

Add

Save

Edit **tool->citations** component attributes

This component does not have attributes

Attributes

Tag sets for this element.

Save component

Add **tool->citations->citation** component

Label	Type	Operations		
+ tool	tool	Edit	Clone	Delete
+ requirements	tool->requirements	Edit	Clone	Delete
+ seqtk	tool->requirements->requirement	Edit	Clone	Delete
+ command	tool->command	Edit	Clone	Delete
+ inputs	tool->inputs	Edit	Clone	Delete
+ input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
+ outputs	tool->outputs	Edit	Clone	Delete
+ tests	tool->tests	Edit	Clone	Delete
+ test	tool->tests->test	Edit	Clone	Delete
+ input1	tool->tests->test->param	Edit	Clone	Delete
+ output1	tool->tests->test->output	Edit	Clone	Delete
+ help	tool->help	Edit	Clone	Delete
+ citations	tool->citations	Edit	Clone	Delete

citation githubseqtk

tool-> citations-> citation

Add

Save

Edit **tool->citations->citation** component attributes

Attributes

Tag sets for this element.

Title

doi

bibtex

Type of citation - currently **doi** and **bibtex** are the only supported options.

Citation \*

@misc{githubseqtk,  
 author = {Last{TODO}, First{TODO}},  
 year = {TODO},  
 title = {seqtk},  
 publisher = {GitHub},  
 journal = {GitHub repository},  
 url = {https://github.com/lh3/seqtk},  
}

Citation in **doi** and **bibtex** format

Save component

## 9. View the complete XML file

Now you have created all the components for building the `seqtk_seq_2.xml` file, you can view the XML page to see how it look like on GTG. Of course, you can view the XML page any time you want. It doesn't have to be after you have added all the components.

The screenshot shows the Galaxy Tool Generator interface for the file `seqtk_seq_2.xml`. The main area displays a table of XML components. A red arrow points to the **VIEW/UPDATE XML** button in the top right corner. Below the table, there is a form to add new components and a **Save** button.

Label	Type	Operations		
tool	tool	Edit	Clone	Delete
requirements	tool->requirements	Edit	Clone	Delete
seqtk	tool->requirements->requirement	Edit	Clone	Delete
command	tool->command	Edit	Clone	Delete
inputs	tool->inputs	Edit	Clone	Delete
input_data	tool->inputs->param(type: data)	Edit	Clone	Delete
outputs	tool->outputs	Edit	Clone	Delete
tests	tool->tests	Edit	Clone	Delete
test	tool->tests->test	Edit	Clone	Delete
input1	tool->tests->test->param	Edit	Clone	Delete
output1	tool->tests->test->output	Edit	Clone	Delete
help	tool->help	Edit	Clone	Delete
citations	tool->citations	Edit	Clone	Delete
citation githubseqtk	tool->citations->citation	Edit	Clone	Delete

Below the table, there is a form to add new components:

New component name:  tool

Below is the XML page.

## seqtk\_seq\_2.xml

[View/Update XML](#)[XML components](#)

Submitted by Anonymous (not verified) on Fri, 11/09/2018 - 21:10

Start

Complete

## ▼ tool

`<tool id="seqtk_seq" name="Convert to FASTA (seqtk)" version="0.1.0" ></tool>`

## ▼ tool &gt; requirements

`<requirements ></requirements>`

## ▼ tool &gt; requirements &gt; requirement

`<requirement type="package" version="1.2" >seqtk</requirement>`

## ▼ tool &gt; command

`<command detect_errors="exit_code" >![CDATA[``seqtk seq -a '$input1' > '$output1'``]]></command>`

## ▼ tool &gt; inputs

`<inputs ></inputs>`

## ▼ tool &gt; inputs &gt; param (type: data)

`<param type="data" name="input1" optional="False" format="fasta" ></param>`

## ▼ tool &gt; outputs

`<outputs ></outputs>`

## ▼ tool &gt; tests

`<tests ></tests>`

## ▼ tool &gt; tests &gt; test

`<test ></test>`

## ▼ tool &gt; tests &gt; test &gt; param

`<param name="2.fastq" ></param>`

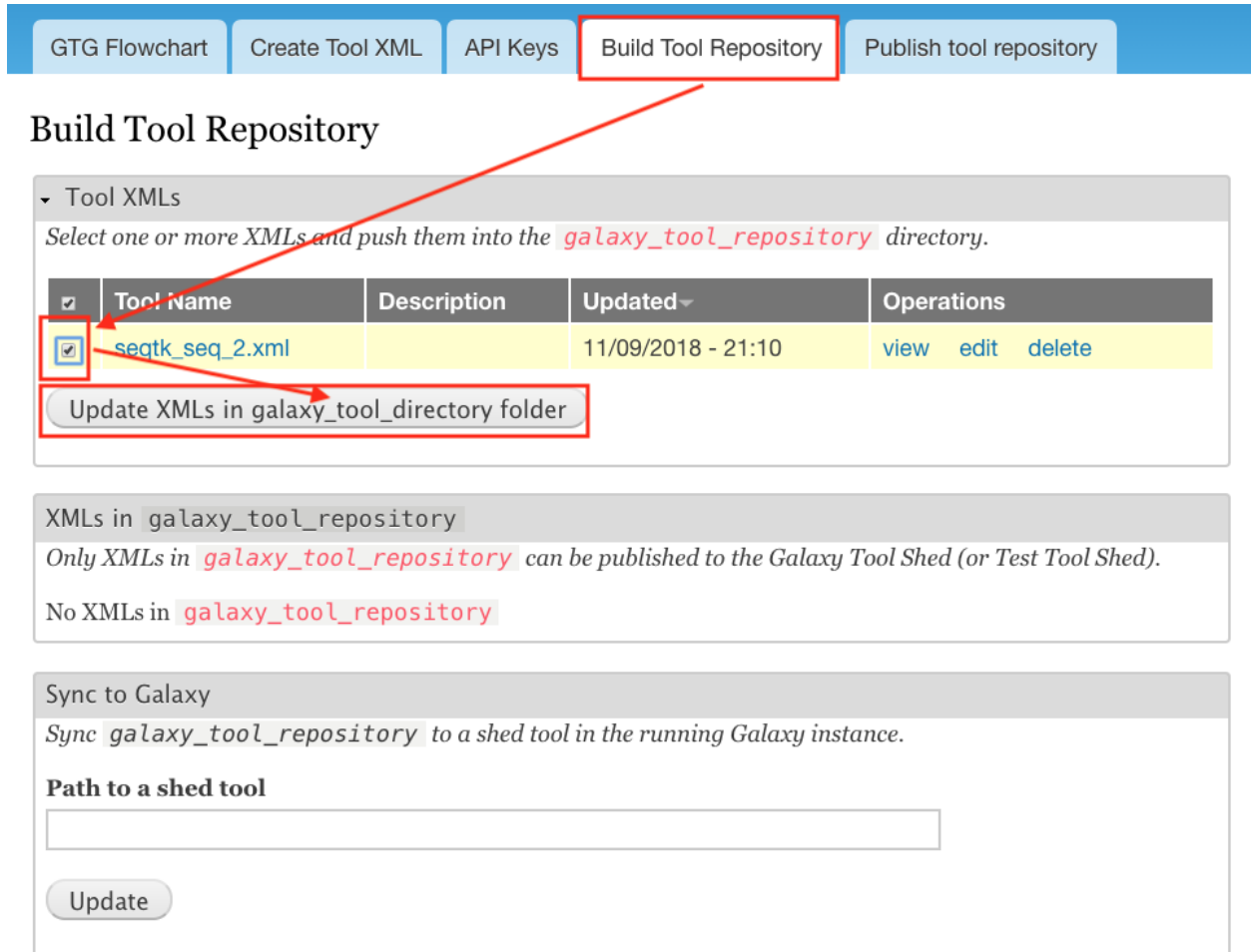
## ▼ tool &gt; tests &gt; test &gt; output

`<output name="output1" file="2.fasta" compare="diff" ></output>``<help >![CDATA[``Usage: seqtk seq [options] |`

## Build tool repository

You have just created the `seqtk_seq_2.xml` file in GTG. However, this file is not in the `gtg_dev_dir/galaxy_tool_repository` directory yet. We need to the XML file into it, and any other non-XML files if there is any.

Click the **Build Tool Repository** tab and select any XML files that you want to add to the `gtg_dev_dir/galaxy_tool_repository` directory. And then click the *Update XMLs in galaxy\_tool\_directory folder*. **This is also the button that you use to add an updated XML to the directory.**



GTG Flowchart   Create Tool XML   API Keys   **Build Tool Repository**   Publish tool repository

### Build Tool Repository

▼ Tool XMLs

Select one or more XMLs and push them into the `galaxy_tool_repository` directory.

<input checked="" type="checkbox"/>	Tool Name	Description	Updated▼	Operations
<input checked="" type="checkbox"/>	seqtk_seq_2.xml		11/09/2018 - 21:10	<a href="#">view</a> <a href="#">edit</a> <a href="#">delete</a>

**Update XMLs in galaxy\_tool\_directory folder**

---

XMLs in `galaxy_tool_repository`

Only XMLs in `galaxy_tool_repository` can be published to the Galaxy Tool Shed (or Test Tool Shed).

No XMLs in `galaxy_tool_repository`

---

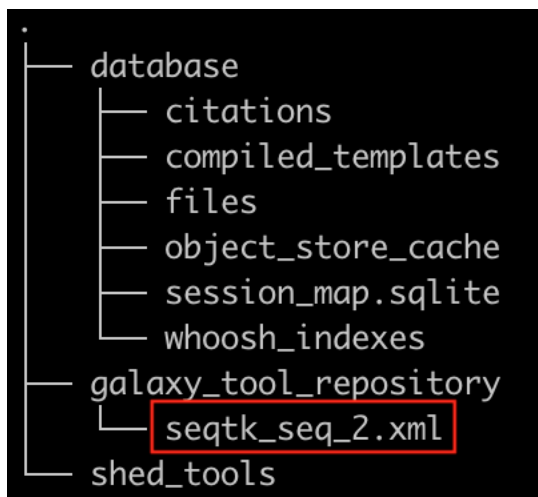
Sync to Galaxy

Sync `galaxy_tool_repository` to a shed tool in the running Galaxy instance.

**Path to a shed tool**

**Update**

You should be able to see the `seqtk_seq_2.xml` file in the `gtg_dev_dir` directory.



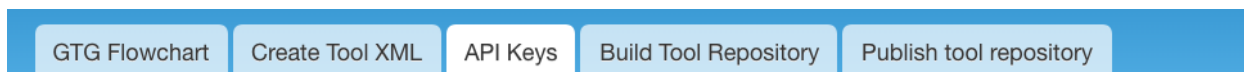
### Add non-XML files

If this tool requires any other non-XML files (for example, test files, scripts, etc.), you can add them directory to the *gtg\_dev\_dir/galaxy\_tool\_repository* directory.

### Publish tool to Test ToolShed

Once we have the XML file(s) and all other non-XML files in the *gtg\_dev\_dir/galaxy\_tool\_repository*, we can publish the tool to Test ToolShed or ToolShed with GTG.

First, we need to add the API key.



## API Keys

### Platform

☒ Test Tool Shed

Select a platform

☐ Tool Shed

Select a platform that you API key is associated.

Test Tool Shed Account Info

Username

mingchen0919

Your username

The public username of your Test Tool Shed account

API Key

.....

Your API key

The associated API key of your Test Tool Shed account

Submit

Then we can publish the tool through the interface below.

GTG Flowchart

Create Tool XML

API Keys

Build Tool Repository

Publish tool repository

## Publish tool repository

The form below creates a `.planemo.yml` file and then use the `planemo` tool to publish the repository to **Tool Shed** or **Test Tool Shed**.

It uses the following command to create and/or update tool repository:

- Create repository in test tool shed: `planemo shed_create --shed_target testtoolshed`
- Publish repository in test tool shed: `planemo shed_update --check_diff --shed_target testtoolshed`
- Create repository in tool shed: `planemo shed_create --shed_target toolshed`
- Publish repository in tool shed: `planemo shed_update --check_diff --shed_target toolshed`

### Repository metadata

#### Name \*

Tool repository name.

#### Synopsis \*

A short description for the tool.

#### Detailed description

### ### Install and test Tool in Galaxy

The next step would be to install and test the tool in the connected Galaxy instance. If the tool needs more work, you can use GTG to update the XML file.

The following interface is used to link the tool in GTG with the same tool installed in Galaxy so that the update will be automatically synced to Galaxy for testing.

GTG Flowchart

Create Tool XML

API Keys

Build Tool Repository

Publish tool repository

## Build Tool Repository

### ▼ Tool XMLs

Select one or more XMLs and push them into the `galaxy_tool_repository` directory.

<input type="checkbox"/>	Tool Name	Description	Updated▼	Operations
<input type="checkbox"/>	<a href="#">seqtk_seq_2.xml</a>		11/09/2018 - 21:10	<a href="#">view</a> <a href="#">edit</a> <a href="#">delete</a>

Update XMLs in galaxy\_tool\_directory folder

### XMLs in galaxy\_tool\_repository

Only XMLs in `galaxy_tool_repository` can be published to the Galaxy Tool Shed (or Test Tool Shed).

- [seqtk\\_seq\\_2.xml](#)

### Sync to Galaxy

Sync `galaxy_tool_repository` to a shed tool in the running Galaxy instance.

Path to a shed tool

Update

Enter path to the tool in Galaxy. The path should be relative to the shed\_tools directory

Everytime you update XML file in Galaxy, you will need to restart Galaxy to integrate the updates. Below is the command to restart Galaxy.

```
docker exec -it gtg_galaxy sh -c 'supervisorctl restart galaxy:'
```

You expect to see the following stdout.

```
galaxy:galaxy_nodejs_proxy: stopped
galaxy:handler0: stopped
galaxy:handler1: stopped
galaxy:galaxy_web: stopped
galaxy:galaxy_nodejs_proxy: started
galaxy:galaxy_web: started
galaxy:handler0: started
galaxy:handler1: started
```

## 2.3 More examples

- [findSSRs tool]([https://github.com/MingChen0919/gtgdocker/blob/master/example\\_tools/findSSRs/findSSRs.md](https://github.com/MingChen0919/gtgdocker/blob/master/example_tools/findSSRs/findSSRs.md)): an example for developing [Aurora Galaxy Tools](<https://github.com/statonlab/aurora-galaxy-tools>).



## CHAPTER 3

---

### Developer Guide

---

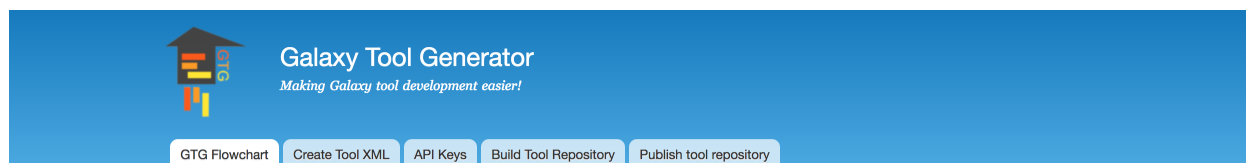


---

### What is Galaxy Tool Generator (GTG)?

---

GTG is a [Drupal](#) based web application which enables developing and publishing Galaxy tools through web interfaces. This web application consists of two Drupal modules: [galaxy\\_tool\\_generator\\_ui](#) and [galaxy\\_tool\\_generator](#), and depends on the [Drupal webform](#) module.



## GTG Flowchart

