

---

# biowdl-input-converter

*Release 0.2.1*

Nov 20, 2019



---

## Contents

---

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Installation</b>	<b>5</b>
<b>3</b>	<b>Usage</b>	<b>7</b>
3.1	Positional Arguments . . . . .	7
3.2	Named Arguments . . . . .	7
<b>4</b>	<b>Samplesheet</b>	<b>9</b>
4.1	CSV/TSV Format . . . . .	9
4.2	Creating comma-delimited files . . . . .	10
4.3	YAML format . . . . .	10
<b>5</b>	<b>Changelog</b>	<b>13</b>
5.1	0.2.1 . . . . .	13
5.2	0.2.0 . . . . .	13
5.3	0.1.0 . . . . .	13



## Table of contents

- *biowdl-input-converter*
- *Introduction*
- *Installation*
- *Usage*
  - *Positional Arguments*
  - *Named Arguments*
- *Samplesheet*
  - *CSV/TSV Format*
  - *Creating comma-delimited files*
  - *YAML format*
- *Changelog*
  - *0.2.1*
  - *0.2.0*
  - *0.1.0*



# CHAPTER 1

---

## Introduction

---

biowdl-input-converter converts human-readable samplesheets into a format that can be easily processed by BioWDL pipelines.

For more information on BioWDL check out the documentation on <https://biowdl.github.io>.



## CHAPTER 2

---

### Installation

---

- Create a new virtualenv
- `run pip install biowdl-input-converter`



---

## Usage

---

Parse samplesheets for BioWDL pipelines.

```
usage: biowdl-input-converter [-h] [-o OUTPUT] [--validate] [--old]
                             [--skip-file-check] [--check-file-md5sums]
                             samplesheet
```

### 3.1 Positional Arguments

<b>samplesheet</b>	The input samplesheet. Format will be automatically detected.
--------------------	---

### 3.2 Named Arguments

<b>-o, --output</b>	The output file to which the json is written. Default: stdout
<b>--validate</b>	Do not generate output but only validate the samplesheet. Default: False
<b>--old</b>	Output old style JSON as used in BioWDL germline-DNA and RNA-seq version 1 pipelines Default: False
<b>--skip-file-check</b>	Skip the checking if files in the samplesheet are present. Default: False
<b>--check-file-md5sums</b>	Do a md5sum check for reads which have md5sums added in the samplesheet. Default: False



# CHAPTER 4

## Samplesheet

A samplesheet provides information about fastq files.

- Sample name
- Library name (for each sample usually one library is used to prepare the sample for sequencing)
- Readgroup name (which lane on the sequencer was used)
- Location of the fastq file containing forward reads (R1) on the filesystem
- Forward reads fastq (R1) md5sum
- Location of the fastq file containing reverse reads (R2) on the filesystem
- Reverse reads fastq (R2) md5sum
- additional properties (if necessary)

### 4.1 CSV/TSV Format

A samplesheet can be a comma- or tab-delimited file. An example looks like this

```
"sample","library","readgroup","R1","R1_md5","R2","R2_md5"  
"s1","lib1","rg1","r1_1.fq","181a657e3f9c3cde2d3bb14ee7e894a3","r1_2.fq",  
↪ "ebe473b62926dcf6b38548851715820e"  
"s2","lib1","rg1","r2_1.fq","7e79b87d95573b06ff2c5e49508e9dbf","r2_2.fq",  
↪ "dc2776dc3a07c4f468455bae1a8ff872"
```

The md5sum fields and the R2 field are optional and can be empty:

```
"sample","library","readgroup","R1","R1_md5","R2","R2_md5"  
"s1","lib1","rg1","r1_1.fq",,"r1_2.fq",  
"s2","lib1","rg1","r2_1.fq",,"r2_2.fq",
```

The R1\_md5, R2 and R2\_md5 columns are optional and can be left out entirely.

```
"sample", "library", "readgroup", "R1"
"s1", "lib1", "rg1", "r1_1.fq"
"s2", "lib1", "rg1", "r2_1.fq"
```

Additional properties at the sample level can be set using additional columns:

```
"sample", "library", "readgroup", "R1", "R1_md5", "R2", "R2_md5", "HiSeq4000", "other_property"
↪
"s1", "lib1", "rg1", "r1_1.fq", "r1_2.fq", "yes", "pizza"
"s2", "lib1", "rg1", "r2_1.fq", "r2_2.fq", "no", "broccoli"
```

Additional properties for the same sample only have to be defined in one line. This saves a lot of duplication for samples with a high readgroup or library count and makes it easier to read the file.

```
"sample", "library", "readgroup", "R1", "R1_md5", "R2", "R2_md5", "HiSeq4000", "other_property"
↪
"s1", "lib1", "rg1", "r1_1.fq", "r1_2.fq", "yes", "pizza"
"s1", "lib1", "rg2", "r1_1.fq", "r1_2.fq", "yes", "pizza"
"s1", "lib2", "rg1", "r1_1.fq", "r1_2.fq", "yes", "pizza"
"s2", "lib1", "rg1", "r2_1.fq", "r2_2.fq", "no", "broccoli"
"s2", "lib1", "rg2", "r2_1.fq", "r2_2.fq", "no", "broccoli"
"s2", "lib1", "rg3", "r2_1.fq", "r2_2.fq", "no", "broccoli"
```

If an additional column is filled with two conflicting values for the same sample an error will be thrown.

## 4.2 Creating comma-delimited files

These files can be easily generated using a spreadsheet program (such as Microsoft Excel or LibreOffice Calc).

Create a table:

sample	library	read-group	R1	R1_md5	R2	R2_md5	HiSeq4000	other_property
s1	lib1	rg1	r1_1.fq	181a657e3f9c3cde2d3bb14e72894a3	r1_2.fq	dc2776dc3a07c4f468455baada8ff872	yes	pizza
s2	lib1	rg1	r2_1.fq		r2_2.fq			

**Note:** Optional fields can be left blank.

And save the table as CSV or TSV format from your spreadsheet program.

## 4.3 YAML format

Alternatively a YAML format can be used

```
samples:
- id: s1
  libraries:
  - id: lib1
    readgroups:
    - id: rg1
```

(continues on next page)

(continued from previous page)

```

      reads:
        R1: r1_1.fq
        R1_md5: 181a657e3f9c3cde2d3bb14ee7e894a3
        R2: r1_2.fq
        R2_md5: ebe473b62926dcf6b38548851715820e
- id: s2
  libraries:
    - id: lib1
      readgroups:
        - id: rg1
          reads:
            R1: r2_1.fq
            R1_md5: 7e79b87d95573b06ff2c5e49508e9dbf
            R2: r2_2.fq
            R2_md5: dc2776dc3a07c4f468455bae1a8ff872

```

Optional fields can be omitted and extra properties can be added:

```

samples:
- id: s1
  HiSeq4000: no
  libraries:
    - id: lib1
      readgroups:
        - id: rg1
          reads:
            R1: r1_1.fq
            R1_md5: 181a657e3f9c3cde2d3bb14ee7e894a3
            R2: r1_2.fq
- id: s2
  HiSeq4000: yes
  libraries:
    - id: lib1
      readgroups:
        - id: rg1
          reads:
            R1: r2_1.fq
            R2: r2_2.fq

```



### 5.1 0.2.1

- Bugfix: R1\_md5 and R2\_md5 columns are not required to be defined anymore in a csv file.

### 5.2 0.2.0

- Make sure only one line of additional properties per sample is need in a csv file.
- Fix a bug where an empty field for an additional property in a csv samplesheet would be defined as "" instead of None.

### 5.3 0.1.0

- Added documentation and readthedocs page
- Added changelog and release procedures
- Added test suite with coverage metrics, enabled CI
- Add validate flag to allow users to validate files
- Added command line interface with ability to write to stdout and files
- Added ability to check files for presence and md5sum checking
- Added sample group -> old style JSON/YAML conversion
- Added sample group -> new style JSON/YAML conversion
- Added yaml -> sample group conversion
- Reworked csv conversion by @DavyCats to fit the new sample group structure

- Added sample group structure to enable any-to-any conversions