# publish.json Data Format

All settings in the second version of publish\_dataset are stored in files named ‘publish.json’. These files are serialized JSON objects containing several sub objects and arrays.

## properties

This is simply a dictionary of values which are used for generating the dataset’s catalogue entry and publishing the dataset. Valid keys are as follows, and all are optional:

title: The title of the dataset (string literal)

description: You’ve guessed it (string literal)

stars: The quality of the data according to the 5-star Linked Open Data scale (integer)

endpoint: The SPARQL endpoint used to get at the data (uri)

license: The license by which the data is released (uri)

accessurl: A URL by which the data can be downloaded (uri)

creatorname: The name of the data’s creator (string literal)

creatoruri: A URI for the creator (uri)

publishername: The name of the data’s publisher (string literal)

publisheruri: A URI for the publisher (uri)

homepage: The homepage of the data, if appropriate (uri)

authority: Whether the data is authorative or non-authorative (boolean)

seealso: Generic rdfs:seeAlso command (uri)

dump: A URI which resolves to a raw dump of the data (uri)

contact: A contact email (string literal)

corrections: An email address or uri for corrections (string literal / uri)

check\_hashes: If true, the dataset is only published if it has changed. If false, it is published regardless (boolean)

import\_file: The name of the file to import into the triplestore after all processing has completed (filename)

NB ‘corrections’ looks for an @ symbol, treats as an email address if present, or a uri if not.

## additional\_triples

This is an optional array of strings, each of which is a triple in N-Triple format. They are used during the creation of the data catalogue and will be simply appended to the output, no questions asked.

## downloads

This is an array of dictionaries. Each dictionary should contain two items, ‘localfile’ and ‘download’, and each are strings. Prior to processing and importing the dataset, these files are downloaded (using WGET) from the URL specified in ‘download’ to the local location specified in ‘localfile’. If a download fails, the publish will not proceed further.

## tools

This is an array of strings, each representing a tool that must be copied from the Grinder/bin directory prior to pre-processing.

## files

This is an array of strings, each representing a file that is required for the publish to succeed. It is checked during publish and if a file is mentioned but not present, the publish will not go ahead. If this object is absent or empty, publish will go ahead regardless. This is useful for requiring manually generated files, such as XSL stylesheets or Grinder configuration files.

## retention

This is an object defining the retention policy for the dataset. The object must contain the key ‘type’, and depending on the value of ‘type’, other values. ‘Type’ can be one of the following:

keep - Old versions of the data are never deleted. No additional parameters needed.

Future versions will support

keeplast - The last [n] versions will be kept. ‘cutoff’ defines [n], ‘action’ can be ‘delete’ or ‘zip’, for old data

keeprecent - Data from the latest [n] days will be kept. ‘cutoff’ defines [n], ‘action’ can be ‘delete’ or ‘zip’.

## commands

This is a dictionary which contains up to three arrays of strings which are passed to the shell in order. The first array is called ‘prepare’ and the second ‘import’. The difference is that the commands in ‘prepare’ are called before the hash check, the commands in ‘import’ are only called if everything else has succeeded and the data is actually to be imported. The ‘import’ instructions, when complete, should produce a file in some RDF format, referred to by the ‘import\_file’ property. The ‘completed’ commands are run if the dataset is successfully published.