## **Return Values**

A job may return some values to Zuul to affect its behavior and for use by dependent jobs. To return a value, use the zuul\_return Ansible module in a job playbook. For example:

**tasks:**

**- zuul\_return:**

**data:**

**foo: bar**

Will return the dictionary {'foo': 'bar'} to Zuul.

Optionally, if you have a large supply of data to return, you may specify the path to a JSON-formatted file with that data. For example:

**tasks:**

**- zuul\_return:**

**file: /path/to/data.json**

Normally returned data are provided to dependent jobs in the inventory file, which may end up in the log archive of a job. In the case where sensitive data must be provided to dependent jobs, the secret\_data attribute may be used instead, and the data will be provided via the same mechanism as job secrets, where the data are not written to disk in the work directory. Care must still be taken to avoid displaying or storing sensitive data within the job. For example:

**tasks:**

**- zuul\_return:**

**secret\_data:**

**password: foobar**

### **Returning the log url**

To set the log URL for a build, use zuul\_return to set the **zuul.log\_url** value. For example:

**tasks:**

**- zuul\_return:**

**data:**

**zuul:**

**log\_url: http://logs.example.com/path/to/build/logs**

### **Returning artifact URLs**

If a build produces artifacts, any number of URLs may be returned to Zuul and stored in the SQL database. These will then be available via the web interface and subsequent jobs.

To provide artifact URLs for a build, use zuul\_return to set keys under the zuul.artifacts dictionary. For example:

**tasks:**

**- zuul\_return:**

**data:**

**zuul:**

**artifacts:**

**- name: tarball**

**url: http://example.com/path/to/package.tar.gz**

**metadata:**

**version: 3.0**

**- name: docs**

**url: build/docs/**

If the value of **url** is a relative URL, it will be combined with the **zuul.log\_url** value if set to create an absolute URL. The **metadata** key is optional; if it is provided, it must be a dictionary; its keys and values may be anything.

### **Skipping dependent jobs**

To skip a dependent job for the current build, use zuul\_return to set the zuul.child\_jobs value. For example:

**tasks:**

**- zuul\_return:**

**data:**

**zuul:**

**child\_jobs:**

**- dependent\_jobA**

**- dependent\_jobC**

Will tell zuul to only run the dependent\_jobA and dependent\_jobC for pre-configured dependent jobs. If dependent\_jobB was configured, it would be now marked as SKIPPED. If zuul.child\_jobs is empty, all jobs will be marked as SKIPPED. Invalid dependent jobs are stripped and ignored, if only invalid jobs are listed it is the same as providing an empty list to zuul.child\_jobs.

### **Leaving warnings**

A job can leave warnings that will be appended to the comment zuul leaves on the change. Use zuul\_return to add a list of warnings. For example:

**tasks:**

**- zuul\_return:**

**data:**

**zuul:**

**warnings:**

**- This warning will be posted on the change.**

If zuul\_return is invoked multiple times (e.g., via multiple playbooks), then the elements of **zuul.warnings** from each invocation will be appended.

### **Leaving file comments**

To instruct the reporters to leave line comments on files in the change, set the **zuul.file\_comments** value. For example:

**tasks:**

**- zuul\_return:**

**data:**

**zuul:**

**file\_comments:**

**path/to/file.py:**

**- line: 42**

**message: "Line too long"**

**level: info**

**- line: 82**

**message: "Line too short"**

**- line: 119**

**message: "This block is indented too far."**

**level: warning**

If zuul\_return is invoked multiple times (e.g., via multiple playbooks), then the elements of zuul.file\_comments from each invocation will be appended.

### **Pausing the job**

A job can be paused after the run phase by notifing zuul during the run phase. In this case the dependent jobs can start and the prior job stays paused until all dependent jobs are finished. This for example can be useful to start a docker registry in a job that will be used by the dependent job. To indicate that the job should be paused use zuul\_return to set the **zuul.pause** value. You still can at the same time supply any arbitrary data to the dependent jobs. For example:

**tasks:**

**- zuul\_return:**

**data:**

**zuul:**

**pause: true**

**registry\_ip\_address: "{{ hostvars[groups.all[0]].ansible\_host }}"**

### **Build Status**

A job build may have the following status:

**SUCCESS**

Nominal job execution.

**FAILURE**

Job executed correctly, but exited with a failure.

**RETRY**

The pre-run playbook failed and the job will be retried.

**RETRY\_LIMIT**

The pre-run playbook failed more than the maximum number of retry attempts.

**POST\_FAILURE**

The post-run playbook failed.

**SKIPPED**

One of the build dependencies failed and this job was not executed.

**NODE\_FAILURE**

The test instance provider was unable to fullfill the nodeset request. This can happen if Nodepool is unable to provide the requested node(s) for the request.

Detailed Document: