**Environment Variables**

Kubernetes environment variables by example

You can set environment variables for containers running in a pod and in addition, Kubernetes exposes certain runtime infos via environment variables automatically.

Let’s launch a [pod](https://github.com/openshift-evangelists/kbe/blob/master/specs/envs/pod.yaml) that we pass an environment variable SIMPLE\_SERVICE\_VERSION with the value 1.0:

$ kubectl apply -f https://raw.githubusercontent.com/openshift-evangelists/kbe/master/specs/envs/pod.yaml

$ kubectl describe pod envs | grep IP:

IP: 172.17.0.3

Now, let’s verify from within the cluster if the application running in the pod has picked up the environment variable SIMPLE\_SERVICE\_VERSION:

[cluster] $ curl 172.17.0.3:9876/info

{"host": "172.17.0.3:9876", "version": "1.0", "from": "172.17.0.1"}

And indeed it has picked up the user-provided environment variable since the default response would be "version": "0.5.0".

You can check what environment variables Kubernetes itself provides automatically (from within the cluster, using a dedicated endpoint that the [app](https://github.com/mhausenblas/simpleservice) exposes):

[cluster] $ curl 172.17.0.3:9876/env

{"version": "1.0", "env": "{'HOSTNAME': 'envs', 'DOCKER\_REGISTRY\_SERVICE\_PORT': '5000', 'KUBERNETES\_PORT\_443\_TCP\_ADDR': '172.30.0.1', 'ROUTER\_PORT\_80\_TCP\_PROTO': 'tcp', 'KUBERNETES\_PORT\_53\_UDP\_PROTO': 'udp', 'ROUTER\_SERVICE\_HOST': '172.30.246.127', 'ROUTER\_PORT\_1936\_TCP\_PROTO': 'tcp', 'KUBERNETES\_SERVICE\_PORT\_DNS': '53', 'DOCKER\_REGISTRY\_PORT\_5000\_TCP\_PORT': '5000', 'PATH': '/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin', 'ROUTER\_SERVICE\_PORT\_443\_TCP': '443', 'KUBERNETES\_PORT\_53\_TCP': 'tcp://172.30.0.1:53', 'KUBERNETES\_SERVICE\_PORT': '443', 'ROUTER\_PORT\_80\_TCP\_ADDR': '172.30.246.127', 'LANG': 'C.UTF-8', 'KUBERNETES\_PORT\_53\_TCP\_ADDR': '172.30.0.1', 'PYTHON\_VERSION': '2.7.13', 'KUBERNETES\_SERVICE\_HOST': '172.30.0.1', 'PYTHON\_PIP\_VERSION': '9.0.1', 'DOCKER\_REGISTRY\_PORT\_5000\_TCP\_PROTO': 'tcp', 'REFRESHED\_AT': '2017-04-24T13:50', 'ROUTER\_PORT\_1936\_TCP': 'tcp://172.30.246.127:1936', 'KUBERNETES\_PORT\_53\_TCP\_PROTO': 'tcp', 'KUBERNETES\_PORT\_53\_TCP\_PORT': '53', 'HOME': '/root', 'DOCKER\_REGISTRY\_SERVICE\_HOST': '172.30.1.1', 'GPG\_KEY': 'C01E1CAD5EA2C4F0B8E3571504C367C218ADD4FF', 'ROUTER\_SERVICE\_PORT\_80\_TCP': '80', 'ROUTER\_PORT\_443\_TCP\_ADDR': '172.30.246.127', 'ROUTER\_PORT\_1936\_TCP\_ADDR': '172.30.246.127', 'ROUTER\_SERVICE\_PORT': '80', 'ROUTER\_PORT\_443\_TCP\_PORT': '443', 'KUBERNETES\_SERVICE\_PORT\_DNS\_TCP': '53', 'KUBERNETES\_PORT\_53\_UDP\_ADDR': '172.30.0.1', 'KUBERNETES\_PORT\_53\_UDP': 'udp://172.30.0.1:53', 'KUBERNETES\_PORT': 'tcp://172.30.0.1:443', 'ROUTER\_PORT\_1936\_TCP\_PORT': '1936', 'ROUTER\_PORT\_80\_TCP': 'tcp://172.30.246.127:80', 'KUBERNETES\_SERVICE\_PORT\_HTTPS': '443', 'KUBERNETES\_PORT\_53\_UDP\_PORT': '53', 'ROUTER\_PORT\_80\_TCP\_PORT': '80', 'ROUTER\_PORT': 'tcp://172.30.246.127:80', 'ROUTER\_PORT\_443\_TCP': 'tcp://172.30.246.127:443', 'SIMPLE\_SERVICE\_VERSION': '1.0', 'ROUTER\_PORT\_443\_TCP\_PROTO': 'tcp', 'KUBERNETES\_PORT\_443\_TCP': 'tcp://172.30.0.1:443', 'DOCKER\_REGISTRY\_PORT\_5000\_TCP': 'tcp://172.30.1.1:5000', 'DOCKER\_REGISTRY\_PORT': 'tcp://172.30.1.1:5000', 'KUBERNETES\_PORT\_443\_TCP\_PORT': '443', 'ROUTER\_SERVICE\_PORT\_1936\_TCP': '1936', 'DOCKER\_REGISTRY\_PORT\_5000\_TCP\_ADDR': '172.30.1.1', 'DOCKER\_REGISTRY\_SERVICE\_PORT\_5000\_TCP': '5000', 'KUBERNETES\_PORT\_443\_TCP\_PROTO': 'tcp'}"}

Alternatively, you can also use kubectl exec to connect to the container and list the environment variables directly, there:

$ kubectl **exec** envs -- printenv

PATH=/usr/**local**/bin:/usr/**local**/sbin:/usr/**local**/bin:/usr/sbin:/usr/bin:/sbin:/bin

HOSTNAME=envs

SIMPLE\_SERVICE\_VERSION=1.0

KUBERNETES\_PORT\_53\_UDP\_ADDR=172.30.0.1

KUBERNETES\_PORT\_53\_TCP\_PORT=53

ROUTER\_PORT\_443\_TCP\_PROTO=tcp

DOCKER\_REGISTRY\_PORT\_5000\_TCP\_ADDR=172.30.1.1

KUBERNETES\_SERVICE\_PORT\_DNS\_TCP=53

ROUTER\_PORT=tcp://172.30.246.127:80

...

You can destroy the created pod with:

$ kubectl delete pod/envs

In addition to the above provided environment variables, you can expose more using the [downward API](https://kubernetes.io/docs/user-guide/downward-api/).