|  |  |
| --- | --- |
| **DOCUMENT RULES:** | |
| **Task Number / Name:** | **K8s Trouble Shooting** |
| **Task name & column name should be written:** | **Bold (CTRL+B)** |
| **Commands should be written in the after # sign:** | *Italic (CTRL+I) #hostname* |
| **Output photo should be cropped or compressed:**  **Photo could be more than one:**  **If you need extra lines, add the line next after it:** | ***Description photo should be with title bar (CTRL + I + B)*** |
| **All other text should be written:** | Standard |
| **Font name and text size:** | Calibri and 9 |
| **Group name:** | Dev\_ops\_ |
| **Student name and surname:** | Murad Abbaszade |
| **E-mail:** | [muradabbaszade6@gmail.com](mailto:muradabbaszade6@gmail.com) |
| **WhatsApp number:** | **+994703664205** |

|  |  |
| --- | --- |
| 1.Automated text example |  |
| ## IP Address  echo [[HOST\_IP]]  ## URL  echo [[HOST\_SUBDOMAIN]]-80-[[KATACODA\_HOST]].[[KATACODA\_DOMAIN]] |  |
| uname -a |  |
| 2.Background script with explicit reference to host01 |  |
| This scenario contains an explicit reference to host01 in the background script (init-env.sh)  sleep 2  ssh root@host01 "chmod 755 /tmp/load-quiz.sh; /tmp/load-quiz.sh > /tmp/load-quiz.out" |  |
| 3.Clipboard integration |  |
| Within the Markdown step, you would write: | `docker` |
| This is created by embedded HTML into the Markdown. | <pre class="file" data-target="clipboard">  Copy Me To The Clipboard!!  </pre> |
| This was created by the HTML: | <pre data-target="clipboard">  Not a file  </pre> |
| 4.Credit where credit due |  |
| 5.Dashboard and terminal tabs |  |
| The index.json used to create the scenarios is shown below. It is important to remember to set the "showdashboard" variable.  "environment": {  "showdashboard": true,  "dashboards": [{"name": "URL", "href": "https://www.google.co.uk"},  {"name": "Port 80", "port": 80},  {"name": "Placeholder", "href": "https://[[HOST\_SUBDOMAIN]]-80-[[KATACODA\_HOST]].environments.katacoda.com"}]  } |  |
| The index.json supports the following syntax that allows you to define an additional terminal tab. | "environment": {  "terminals": [{"name": "Terminal 2", "target": "host01"}]  } |
| The command below will open a new Terminal tab and automatically run docker container stats command. | "environment": {  "terminals": [{"name": "Docker Stats", "command": "docker container stats", "target": "host01"}]  } |
|  | "assets": {  "host01": [  {"file": "scenario-setup-configuration.sh", "target": "/usr/local/bin", "chmod": "+x"}  ]  } |
| 6.Dashboard and terminal tabs iframe |  |
| This allows the user to keep a consistent view and allow the steps to explain actions that need to be performed within the UI. | "environment": {  "showdashboard": true,  "dashboards": [{"name": "URL", "href": "https://www.katacoda.com"},  {"name": "Port 80", "port": 80},  {"name": "Placeholder", "href": "https://[[HOST\_SUBDOMAIN]]-80-[[KATACODA\_HOST]].environments.katacoda.com"}],  "uilayout": "terminal-iframe"  } |
| 7.Creating training Pathway.json file |  |
| git clone https://github.com/katacoda/scenario-examples.git katacoda-scenario-examples |  |
| katacoda-scenario-examples/training/training-example-pathway.json |  |
| 8.Display images |  |
| In this scenario, within the Assets directory, a logo-text-with-head.png file exists. Only images within the Assets directory are available. | ![Katacoda Logo](./assets/logo-text-with-head.png) |
| 9.Displaying progress spinner |  |
|  |  |
| cat /usr/local/bin/wait.sh |  |
| 10.Displaying Web UI |  |
| docker container run -p 80:80 -d katacoda/docker-http-server |  |
| curl localhost |  |
| Generated Web Link  https://[[HOST\_SUBDOMAIN]]-80-[[KATACODA\_HOST]].environments.katacoda.com | https://[[HOST\_SUBDOMAIN]]-80-[[KATACODA\_HOST]].environments.katacoda.com |
| 11.Embed videos into Scenarios |  |
| Example of embedding a video on the intro screen.  Here is the HTML example: | <iframe style="width: 700px;height: 400px;" src="https://www.youtube-nocookie.com/embed/KeJJ34BvA7Q" frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe> |
| Within the Markdown content for Katacoda you can embed any HTML element, this means you would have the ability to embed YouTube videos, such as the example below.  The HTML included is: | <iframe width="560" height="315" src="https://www.youtube-nocookie.com/embed/KeJJ34BvA7Q" frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe> |
| 12.Environment Usages  Kubernetes |  |
| kubeadm init --kubernetes-version $(kubeadm version -o short) |  |
| kubectl get nodes |  |
| Start using the Multi-node Kubernetes environment for your content by setting the imageid to kubernetes-cluster:1.18 . | "environment": {  "uilayout": "terminal"  },  "backend": {  "imageid": "kubernetes-cluster:1.18"  } |
| 13.Kubernetes |  |
| You can get with kubectl get nodes |  |
| Start using the Multi-node Kubernetes environment for your content by setting the imageid to kubernetes-cluster-running:1.18 . | "environment": {  "uilayout": "terminal"  },  "backend": {  "imageid": "kubernetes-cluster-running:1.18"  } |
| 14.Minikube |  |
| The cluster can be started with the command minikube start . |  |
| Start using the Minikube environment for your content by setting the imageid to minikube . | "environment": {  "uilayout": "terminal"  },  "backend": {  "imageid": "minikube"  } |
| 15.Ubuntu 18.04 Instance |  |
| Start using the Ubuntu environment for your content by setting the imageid to ubuntu:1804 . | "backend": {  "imageid": "ubuntu:1804"  } |
| 16.Ubuntu 20.04 instance |  |
| Start using the Ubuntu environment for your content by setting the imageid to ubuntu:2004 . | "backend": {  "imageid": "ubuntu:2004"  } |
| 17.Get the IP or DNS name for the environment |  |
| IP Address | echo [[HOST\_IP]] |
| URL | echo [[HOST\_SUBDOMAIN]]-80-[[KATACODA\_HOST]].[[KATACODA\_DOMAIN]] |
| 18.Opening multiple terminals |  |
| echo "Run in T1" |  |
| echo "Run in T2" |  |
| echo "No Terminal Defined" |  |
| 19.Run commands automatically |  |
| The index.json file contains the scenario structure.  Within the intro block and for each step, two files can be defined under courseData and code .  courseData defines a script which runs in the background.  code defines the commands to run in the foreground.  "intro": {  "text": "intro.md",  "courseData": "background.sh",  "code": "foreground.sh",  "credits": ""  } |  |
| By having scripts run for each step it's possible to simulate activities and actions under the covers. Such as modifying settings or causing additional situations for the user to complete.  The syntax is the same as when defining scripts for intro. | {  "title": "Scripts Example",  "courseData": "step2-background.sh",  "code": "step2-foreground.sh",  "text": "step2.md"  } |
| echo "This is a background script with a long running process"  sleep 10  echo "done" >> /opt/.backgroundfinished |  |
| echo "Waiting to complete"; while [ ! -f /opt/.backgroundfinished ] ; do sleep 2; done; echo "Done" |  |
| 20.UILayouts  Scenario with Terminal and VS Code UI as tab |  |
| The Katacoda terminal layout provides a full Terminal experience. This can be extended to include a full IDE experience as a separate tab by including showide within the environment section of the index.json . For example:  "environment": {  "showide": true  } |  |
| With the IDE, you can open files certain files from Markdown - newFile.js{{open}}  You can copy, extend or replace text from UI helpers.  var http = require('http');  var requestListener = function (req, res) {  res.writeHead(200);  res.end('Hello, World!');  }  var server = http.createServer(requestListener);  server.listen(3000, function() { console.log("Listening on port 3000")}); |  |
| Within the Markdown, include: | <pre class="file" data-filename="app.js" data-target="prepend">console.log("Starting...")  </pre> |
| Within the Markdown, include: | <pre class="file" data-filename="index.js" data-target="replace">console.log("Index.js here...")  </pre> |
| 21.Scenario with terminal IFRAME split UI |  |
| The Katacoda terminal-iframe-split UI Layout provides a full Terminal experience with the ability to include webpages as iframes alongside the content.  # Index.json  Example:  "environment": {  "uilayout": "terminal-iframe-split"  },  "backend": {  "imageid": "ubuntu",  "port": 80  } |  |
| docker container run -p 80:80 -d katacoda/docker-http-server |  |
| 22.Scenario with terminal iframe ui |  |
| The Katacoda iframe UI Layout provides a full webpage experience.  # Index.json  Example:  "environment": {  "uilayout": "iframe"  },  "backend": {  "url": "https://httpbin.org"  } |  |
| 23.Upload file assets |  |
| The options are:   1. file : The name of the file within the Assets directory. 2. target : The directory where the file should be uploaded to. 3. chmod : Optional, automatically set a chmod flag after uploading the file.   The following is an example from the index.json .  "details": {  "assets": {  "host01": [  {"file": "wait.sh", "target": "/usr/local/bin/", "chmod": "+x"},  {"file": "deploy.sh", "target": "/usr/local/bin/", "chmod": "+x"}  ]  }  }, |  |
| cat /usr/local/bin/wait.sh |  |
| cat /usr/local/bin/deploy.sh |  |
| 24.Verifying user actions during scenario |  |
| For the user to be allowed to proceed, the script should output the text "done". For example, the following verify script checks to make sure the user has run git init before they can proceed.  [ -d /home/scrapbook/tutorial/.git ] && echo "done" |  |
| The index.json example is:  "details": {  "steps": [  {  "title": "Verify",  "text": "step1.md",  "verify": "step1-verify.sh"  }  ]  } |  |
| 25.Visual studio code install extension |  |
| To install a VSCode extension execute the following in a background script:  The use of bsdtar is required due to the packaging format of VS Extensions.  echo 'debconf debconf/frontend select Noninteractive' | sudo debconf-set-selections |  |
| cd /tmp && apt-get update -y && apt install libarchive-tools -y # install bsdtar |  |
| curl -L https://marketplace.visualstudio.com/\_apis/public/gallery/publishers/humao/vsextensions/rest-client/0.24.3/vspackage | bsdtar -xvf - extension  mv extension /opt/.katacodacode/extensions/humao.rest-client-0.24.3 |  |
| 26.Visualise Environment tab |  |
| Katacoda has integrated [Weave Scope](https://weave.works/scope) to help visualise Docker and Kubernetes based deployments.  The aim of integrating Scope is to help people understand what has been deployed and as changes new aspects are deployed to automatically see the impact.  To launch Scope, click the Tab. |  |
| Index.json  To add the visualise tab to your scenarios, include the following JSON snippet within the environments node. This will automatically add the tab to the scenarios. When the tab is pressed, the command to launch the Scope container will run.  "showvisualise": true,  "scope": "docker run --name=scope -d --net=host --pid=host --privileged -v /var/run/docker.sock:/var/run/docker.sock:rw weaveworks/scope:1.9.1 --probe.docker=true",  "scopePort": 4040, |  |
| 27.Writing files |  |
| You can write files in your scenario using a bash command:  cat << EOF > /tmp/storageos-secret.yaml  apiVersion: v1  kind: Secret  metadata:  name: "storageos-api"  namespace: "storageos-operator"  labels:  app: "storageos"  type: "kubernetes.io/storageos"  EOF |  |
| cat /tmp/storageos-secret.yaml |  |