

K8s ConfigMap 更新策略

AUTHOR: 彭玲 TIME: 2022/1/26

K8s ConfigMap 更新策略

前言

创建 ConfigMap 应用

创建 ConfigMap

创建 Pod

验证 环境变量

修改 ConfigMap

修改 ConfigMap

重启 Pod

验证 环境变量

K8s 资源清理

前言

为了说明 ConfigMap 更新策略，本文以 ConfigMap data 中的 `player_initial_lives` 键为例，测试对其修改前后的 Pod 应用情况。

创建 ConfigMap 应用

本节将创建一个 ConfigMap 和一个 `alpine` Pod，并在 Pod 中使用 ConfigMap。

创建 ConfigMap

```
1 anxin@node38:~/pengling/k8s/configmap-demo$ vi configmap-demo.yaml
2
3 apiVersion: v1
4 kind: ConfigMap
5 metadata:
6   name: game-demo
7   namespace: julin
8 data:
9   player_initial_lives: "3"
10  ui_properties_file_name: "user-interface.properties"
11
12  game.properties: |-
13    enemy.types=aliens,monsters
14    player.maximum-lives=5
15  user-interface.properties: |
16    color.good=purple
17    color.bad=yellow
18    allow.textmode=true
```

应用 ConfigMap 清单:

```
1 anxin@node38:~/pengling/k8s/configmap-demo$ kubectl apply -f configmap-  
demo.yaml  
2 configmap/game-demo created
```

创建 Pod

```
1 anxin@node38:~/pengling/k8s/configmap-demo$ vi pod-demo-using-configmap.yaml  
2  
3 apiVersion: v1  
4 kind: Pod  
5 metadata:  
6   name: configmap-demo-pod  
7   namespace: julin  
8 spec:  
9   containers:  
10    - name: demo  
11      image: alpine  
12      command: ["sleep", "3600"]  
13      env:  
14        - name: PLAYER_INITIAL_LIVES # 请注意这里和 ConfigMap 中的键名是不一样的  
15          valueFrom:  
16            configMapKeyRef:  
17              name: game-demo # 这个值来自 ConfigMap  
18              key: player_initial_lives # 需要取值的键  
19        - name: UI_PROPERTIES_FILE_NAME  
20          valueFrom:  
21            configMapKeyRef:  
22              name: game-demo  
23              key: ui_properties_file_name  
24      volumeMounts:  
25        - name: config  
26          mountPath: "/config"  
27          readOnly: false  
28      volumes:  
29        - name: config  
30          configMap:  
31            name: game-demo  
32            # 来自 ConfigMap 的一组键, 将被创建为文件  
33            items:  
34              - key: "game.properties"  
35                path: "game_p.properties"  
36              - key: "user-interface.properties"  
37                path: "user-interface_p.properties"
```

应用 Pod 清单:

```
1 anxin@node38:~/pengling/k8s/configmap-demo$ kubectl apply -f pod-demo-using-  
configmap.yaml  
2 pod/configmap-demo-pod created
```

验证环境变量

查看 `alpine` 容器中的 `PLAYER_INITIAL_LIVES` 环境变量：

```
1 anxin@node38:~$ kubectl exec -it configmap-demo-pod -n julin -- sh
2 / # echo $PLAYER_INITIAL_LIVES
3 3
```

修改 ConfigMap

本节修改 ConfigMap，并验证其更新策略。

修改 ConfigMap

将 ConfigMap `data` 中的 `player_initial_lives: "3"` 键值修改为 `"6"`。

```
1 anxin@node38:~/pengling/k8s/configmap-demo$ sed -i 's/player_initial_lives:
  "3"/player_initial_lives: "6"/g' configmap-demo.yaml
```

应用 ConfigMap 清单：

```
1 anxin@node38:~/pengling/k8s/configmap-demo$ kubectl apply -f configmap-
  demo.yaml
2 configmap/game-demo configured
```

重启 Pod

通过 `kubectl exec` 再次检查 `alpine` Pod，查看是否已应用配置。结果：`PLAYER_INITIAL_LIVES` 仍为 `3`（并未改变）。

配置值未更改，因为 **需要重新启动 Pod** 才能从关联的 ConfigMap 中获取更新的值。

让我们删除并重新创建 Pod：

```
1 anxin@node38:~$ kubectl delete pod configmap-demo-pod -n julin
2 pod "configmap-demo-pod" deleted
3
4 anxin@node38:~/pengling/k8s/configmap-demo$ kubectl apply -f pod-demo-using-
  configmap.yaml
5 pod/configmap-demo-pod created
```

验证环境变量

重新检查 `alpine` 容器中的 `PLAYER_INITIAL_LIVES` 环境变量：结果说明修改后的 ConfigMap 已生效。

```
1 anxin@node38:~$ kubectl exec -it configmap-demo-pod -n julin -- sh
2 / # echo $PLAYER_INITIAL_LIVES
3 6
```

K8s 资源清理

最后，清理上面创建的 ConfigMap 和 Pod。

```
1 # 删除 ConfigMap
2 anxin@node38:~/pengling/k8s/configmap-demo$ kubectl delete -f configmap-
  demo.yaml
3 configmap "game-demo" deleted
4 # 删除 Pod
5 anxin@node38:~/pengling/k8s/configmap-demo$ kubectl delete -f pod-demo-using-
  configmap.yaml
6 pod "configmap-demo-pod" deleted
```