ChatGPT

Helm Mastery Guide with Connected Exercises

This complete guide is designed to teach you Helm from the ground up in a structured, connected way — where each exercise builds upon the previous one. By the end, you'll have mastered Helm concepts and will have built a production-ready Helm chart with advanced practices.

Phase 1: Get Started with Helm and Your First App

Step 1: Setup Local Environment

Objective: Prepare your system with Minikube and Helm.

```
minikube start
curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
```

Step 2: Explore a Real Helm Chart

Objective: Use a production-grade chart to see Helm in action.

```
helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
helm install my-nginx bitnami/nginx
minikube service my-nginx
```

This lets you explore how Helm simplifies app deployment using reusable charts.

Step 3: Inspect the Deployed Chart

```
helm get all my-nginx
```

Look at the resources, values, hooks, etc., and understand what was deployed.

📦 Phase 2: Create and Customize Your Own Chart

Step 4: Scaffold Your Own Chart

```
helm create mychart
cd mychart
```

> 🔍 You'll now use this chart throughout the rest of the exercises, modifying it step by step.

Step 5: Deploy the Chart Locally

```
helm install myapp ./
```

Use this to validate your environment is working.

Step 6: Customize Values

Edit `values.yaml`:

```
replicaCount: 2
image:
  repository: nginx
  tag: 1.21.6
  pullPolicy: IfNotPresent
```

Then apply the changes:

```
helm upgrade myapp ./
```

Phase 3: Make the Chart Dynamic

Step 7: Use Conditionals and Loops

Update `values.yaml`:

```
config:
  enabled: true
  data:
    LOG_LEVEL: debug
    FEATURE_FLAG: true
```

Create `templates/configmap.yaml`:

```
{{- if .Values.config.enabled }}
apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ include "mychart.fullname" . }}-config
  labels:
    app: {{ include "mychart.name" . }}
data:
  {{- range $key, $value := .Values.config.data }}
  {{ $key }}: "{{ $value }}"
  {{- end }}
{{- end }}
```

Run:

```
helm upgrade myapp ./
```

Phase 4: Add Dependencies

Step 8: Add Redis as a Subchart

In `Chart.yaml`:

```
dependencies:
  - name: redis
    version: "17.0.0"
    repository: "https://charts.bitnami.com/bitnami"
```

Then:

```
helm dependency update
helm upgrade myapp ./
```

You now have an app with Redis bundled in a single chart.

⛓ Phase 5: Add a Hook

Step 9: Add Pre-install Job

Create `templates/init-job.yaml`:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: {{ include "mychart.fullname" . }}-init
  annotations:
    "helm.sh/hook": pre-install
spec:
  template:
    spec:
      containers:
        - name: init
          image: busybox
          command: ['sh', '-c', 'echo Pre-Installation Script']
      restartPolicy: Never
```

Run:

```
helm upgrade --install myapp ./
```

> 📦 This job runs automatically before installing the app.

Phase 6: Upgrade, Rollback & Testing

Step 10: Upgrade

Change something in `values.yaml` (e.g., `replicaCount: 3`) and run:

```
helm upgrade myapp ./
```

Step 11: Rollback

```
helm rollback myapp 1
```

> ⏪ Rollbacks restore the previous working version.

Step 12: Add Helm Tests

Create `templates/tests/test-connection.yaml`:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: "{{ include \"mychart.fullname\" . }}-test"
  annotations:
    "helm.sh/hook": test
spec:
  containers:
    - name: test
      image: busybox
      command: ['wget', 'http://myapp']
  restartPolicy: Never
```

Run:

```
helm test myapp
```

📤 Phase 7: Package and Distribute

Step 13: Lint & Package

```
helm lint ./
helm package ./
```

Step 14: Serve and Consume

```
helm repo index .
python3 -m http.server 8080
helm repo add myrepo http://localhost:8080
helm install myapp myrepo/mychart
```

Phase 8: Handle Secrets Securely

Step 15: Store Encoded Secrets

In `values.yaml`:

```
secret: cGFzc3dvcmQ=
```

Create `templates/secret.yaml`:

```
apiVersion: v1
kind: Secret
metadata:
  name: {{ include "mychart.fullname" . }}-secret
  type: Opaque
  data:
    password: {{ .Values.secret }}
```

Phase 9: CI/CD with GitHub Actions

Step 16: Automate with GitHub Actions

Create `.github/workflows/helm-deploy.yaml`:

```
name: Helm Deploy
on: [push]
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Setup Helm
        uses: azure/setup-helm@v3
      - name: Deploy
        run: |
          helm upgrade --install myapp ./mychart \
            --kubeconfig ${{ secrets.KUBECONFIG }}
```

🎓 Final Goal

By completing all these connected steps, you've:

- Built a complete Helm chart
- Used values, templates, conditionals, dependencies
- Tested, upgraded, rolled back, and packaged it
- Integrated CI/CD automation

You now have Helm mastery. Want to go further with Helm plugins, OCI registries, or multi-env support?