

DaemonSet, StatefulSet, Job, CronJob

Слайд 1. Задача контроллеров

Факты:

- Каждый контроллер отвечает за определённый способ размещения и жизненный цикл Pod'ов.
- Нужен для повторяемого и предсказуемого поведения приложений.

Слайд 2. Когда использовать DaemonSet

Факты:

- Один Pod на каждой ноде (или на выбранных нодах).
- Полезно для агентов, лог-сборщиков, мониторинга.

Конкретные примеры:

- node-exporter для Prometheus.
- filebeat/fluentd для логов.
- cni plugins.

Слайд 3. Когда использовать StatefulSet

Факты:

- Предсказуемые имена Pod'ов (pod-0, pod-1...).
- Стабильный сетевой идентификатор.
- Привязка PV → Pod.

Использование:

- Базы данных (PostgreSQL, MongoDB).
- Кластеры Kafka, Zookeeper.
- Любые приложения, которым нужно состояние.

Слайд 4. Когда использовать Job

Факты:

- Одноразовая задача.
- Контроллер следит, что задача закончилась успешно.
- Можно задавать количество параллельных попыток.

Использование:

- Миграция базы.
- Генерация отчёта.
- Запуск скрипта по требованию.

Слайд 5. Когда использовать CronJob

Факты:

- Job + расписание.
- Работает как cron на уровне кластера.
- Контролирует историю запусков.

Использование:

- Бэкапы.
- Очистка временных файлов.
- Регулярные ETL-задачи.

Слайд 6. Пример: CronJob для бэкапа

Цель: раз в сутки сохранять данные из PVC в удалённое хранилище (пример — S3).

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: backup-job
spec:
  schedule: "0 2 * * *" # каждый день в 02:00
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: backup
              image: alpine
              command: ["sh", "-c", "tar czf /backup/data.tar.gz /data && echo done"]
              volumeMounts:
                - name: data
                  mountPath: /data
                - name: backup
                  mountPath: /backup
          restartPolicy: OnFailure
          volumes:
            - name: data
              persistentVolumeClaim:
                claimName: app-data
            - name: backup
```

Слайд 7. Что важно проверить в CronJob

Факты:

- Расписание (валидность cron).
- Сколько Job сохранять (`successfulJobsHistoryLimit`).
- Ограничение параллельного запуска (`concurrencyPolicy`).
- Контейнер должен *реально завершаться*.

Слайд 8. Пример: DaemonSet с node-exporter

Цель: развернуть node-exporter на всех нодах.

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: node-exporter
spec:
  selector:
    matchLabels:
      app: node-exporter
  template:
    metadata:
      labels:
        app: node-exporter
    spec:
      hostPID: true
      containers:
        - name: node-exporter
          image: prom/node-exporter:v1.7.0
          ports:
            - containerPort: 9100
```

Слайд 9. Что важно для DaemonSet

Факты:

- Проверить taints/tolerations, если нужны все ноды.
- Убедиться, что порты не конфликтуют.
- Если нужен доступ к хосту — включить hostPID/hostNetwork.