Describiendo un Pod

Es posible crear un Pod directamente (lo que se denomina utilización imperativa) mediante kubectl:

kubectl run pod-nginx --image=nginx

De esta forma se crea un Pod con un contenedor que utiliza la imagen nginx:latest (no hemos especificado una versión) del registro que esté definido por defecto en el cluster de Kubernetes, se asigna una dirección IP y se lanza en uno de los nodos del cluster.

Un Pod tiene otros muchos parámetros asociados, que en este caso quedarán sin definir o Kubernetes asumirá los valores por defecto. Sin embargo es mucho más habitual trabajar con los objetos de Kubernetes de manera declarativa, definiendo los objetos de forma detallada a través de un fichero en formato YAML. De esta forma tenemos un fichero con la definición del objeto que hemos lanzado y podemos utilizar en otro momento exactamente la misma definición o podemos ir modificándola y aplicando los cambios cuando sea conveniente.

Un ejemplo podría ser el contenido del fichero pod.yaml:

```
apiVersion: v1 # required
kind: Pod # required
metadata: # required
 name: pod-nginx # required
 labels:
  app: nginx
  service: web
spec: # required
 containers:
  - image: nginx:1.16
    name: contenedor-nginx
    imagePullPolicy: Always
```

Veamos cada uno de los parámetros que hemos definido:

- apiVersion: v1: La versión de la API que vamos a usar.
- kind: Pod: La clase de recurso que estamos definiendo.

- metadata: Información que nos permite identificar unívocamente el recurso:
    - name: Nombre del pod
    - labels: Las Labels nos permiten etiquetar los recursos de Kubernetes (por ejemplo un pod) con información del tipo clave/valor.
- spec: Definimos las características del recurso. En el caso de un Pod indicamos los contenedores que van a formar el Pod (sección containers), en este caso sólo uno.
    - image: La imagen desde la que se va a crear el contenedor
    - name: Nombre del contenedor.
    - imagePullPolicy: Las imágenes se guardan en un registro interno. Se pueden utilizar registros públicos (google o docker hub son los más usados) y registros privados. La política por defecto es IfNotPresent, que se baja la imagen si no está en el registro interno. Si queremos forzar la descarga desde el repositorio externo, tendremos que indicar imagePullPolicy:Always.