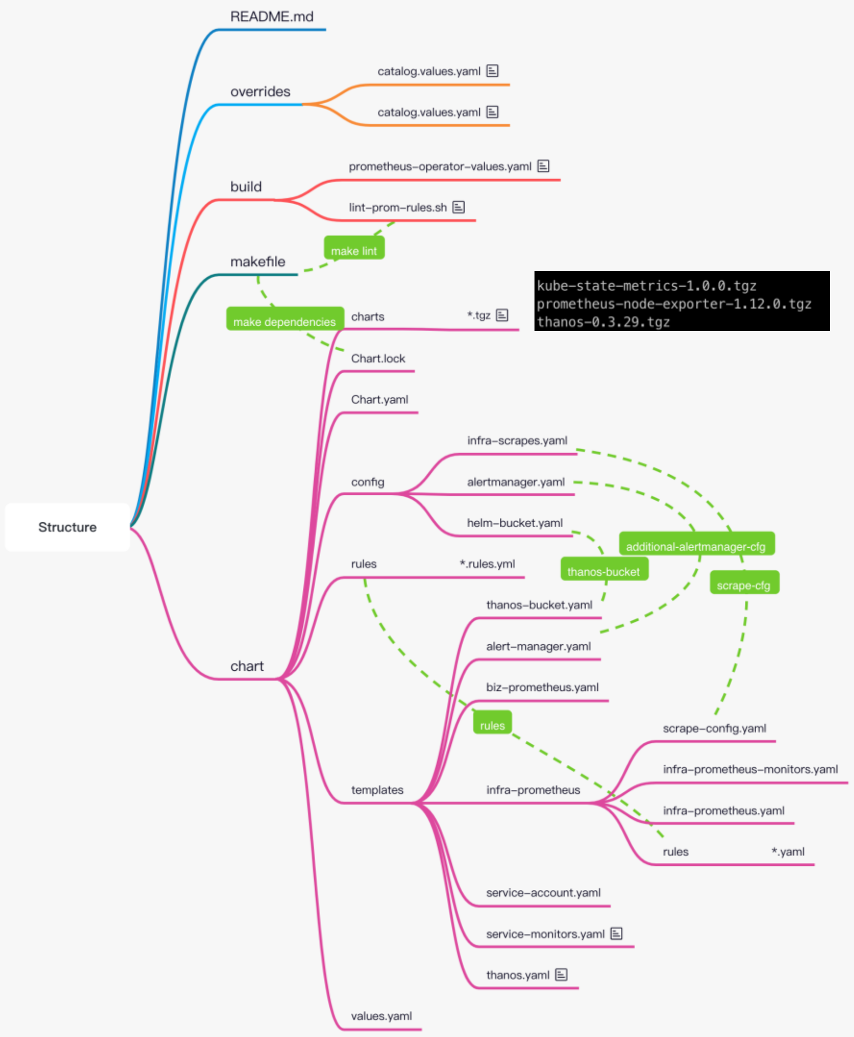
Prometheus Chart

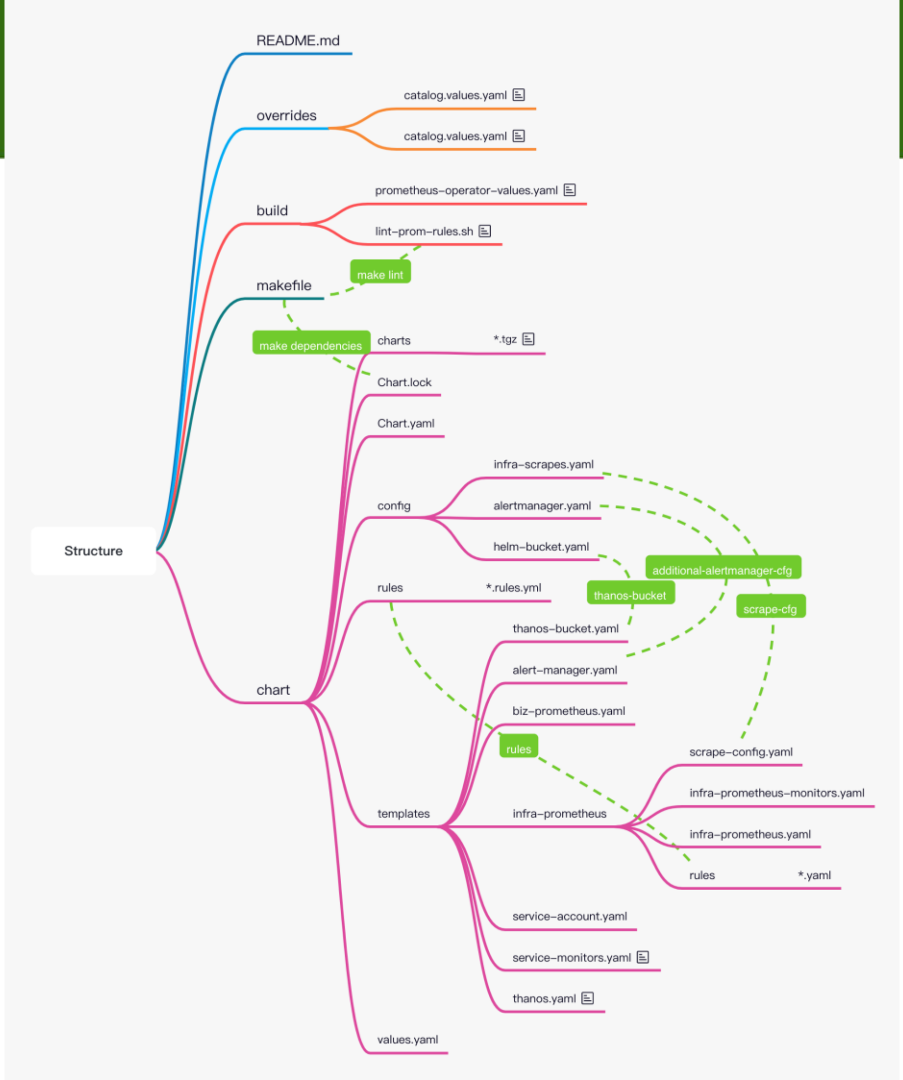
Prometheus

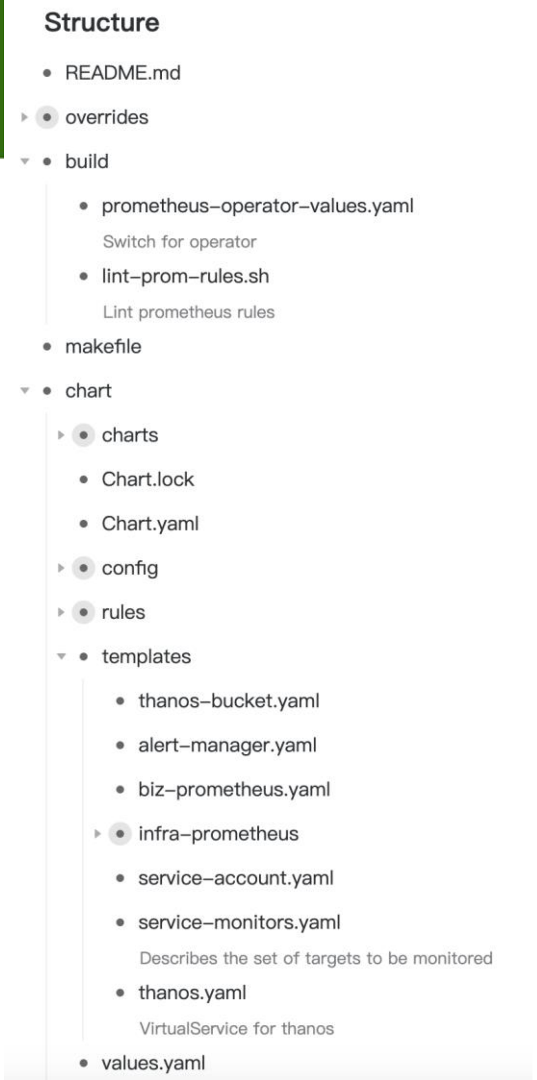
https://bitbucket.org/iherbllc/infra.prometheus.git

```
Purpose:
  Create all the ones we need in one chart.


1. Two prometheus
infra has all the kubernetes related metrics
biz has all the metrics from the apps

2. Prometheus and prometheus operator

First turn off the default prometheus install.
Then install the operator separately from the official
chart.

3. Arguments can be passed from the pipeline

4. Structure is clear

5. Familiar with dependences is required
```



```
kube-state-metrics-1.0.0.tgz
prometheus-node-exporter-1.12.0.tgz
thanos-0.3.29.tgz
```





Structure

- README.md
- overrides
- build
  - prometheus–operator–values.yaml

    Switch for operator
  - lint–prom–rules.sh

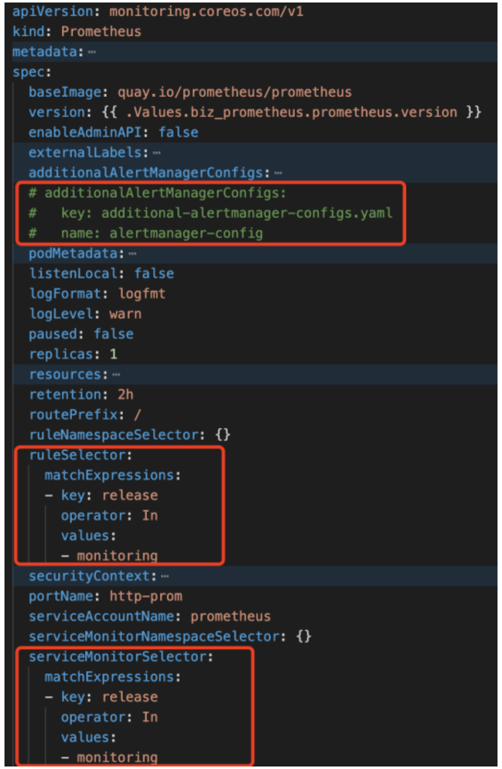
    Lint prometheus rules
- makefile
- chart
  - charts
  - Chart.lock
  - Chart.yaml
  - config
  - rules
  - templates
    - thanos–bucket.yaml
    - alert–manager.yaml
    - biz–prometheus.yaml
    - infra–prometheus
    - service–account.yaml
    - service–monitors.yaml

      Describes the set of targets to be monitored
    - thanos.yaml

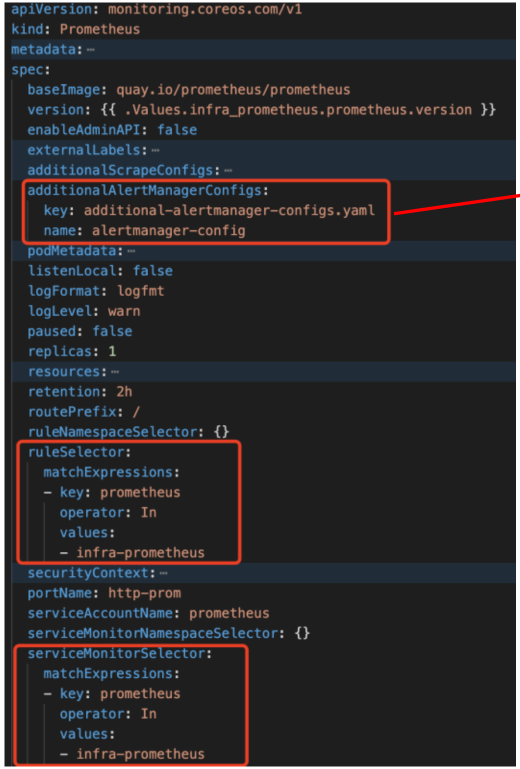
      VirtualService for thanos
  - values.yaml

infra vs biz

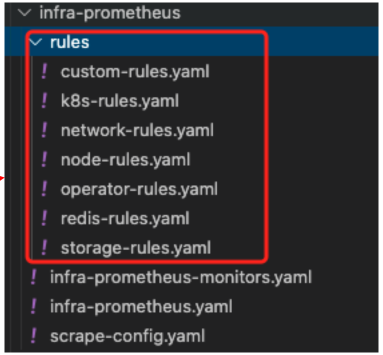
```
infra has all the kubernetes related metrics
biz has all the metrics from the apps
```



```yaml
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:⋯
spec:
  baseImage: quay.io/prometheus/prometheus
  version: {{ .Values.biz_prometheus.prometheus.version }}
  enableAdminAPI: false
  externalLabels:⋯
  additionalAlertManagerConfigs:⋯
  # additionalAlertManagerConfigs:
  #   key: additional-alertmanager-configs.yaml
  #   name: alertmanager-config
  podMetadata:⋯
  listenLocal: false
  logFormat: logfmt
  logLevel: warn
  paused: false
  replicas: 1
  resources:⋯
  retention: 2h
  routePrefix: /
  ruleNamespaceSelector: {}
  ruleSelector:
    matchExpressions:
    - key: release
      operator: In
      values:
      - monitoring
  securityContext:⋯
  portName: http-prom
  serviceAccountName: prometheus
  serviceMonitorNamespaceSelector: {}
  serviceMonitorSelector:
    matchExpressions:
    - key: release
      operator: In
      values:
      - monitoring
```

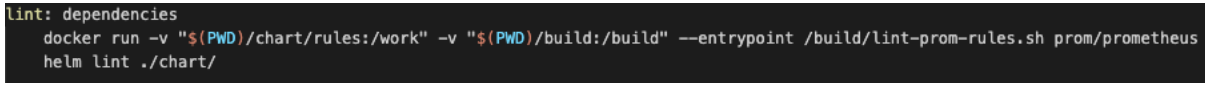


```yaml
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:⋯
spec:
  baseImage: quay.io/prometheus/prometheus
  version: {{ .Values.infra_prometheus.prometheus.version }}
  enableAdminAPI: false
  externalLabels:⋯
  additionalScrapeConfigs:⋯
  additionalAlertManagerConfigs:
    key: additional-alertmanager-configs.yaml
    name: alertmanager-config
  podMetadata:⋯
  listenLocal: false
  logFormat: logfmt
  logLevel: warn
  paused: false
  replicas: 1
  resources:⋯
  retention: 2h
  routePrefix: /
  ruleNamespaceSelector: {}
  ruleSelector:
    matchExpressions:
    - key: prometheus
      operator: In
      values:
      - infra-prometheus
  securityContext:⋯
  portName: http-prom
  serviceAccountName: prometheus
  serviceMonitorNamespaceSelector: {}
  serviceMonitorSelector:
    matchExpressions:
    - key: prometheus
      operator: In
      values:
      - infra-prometheus
```

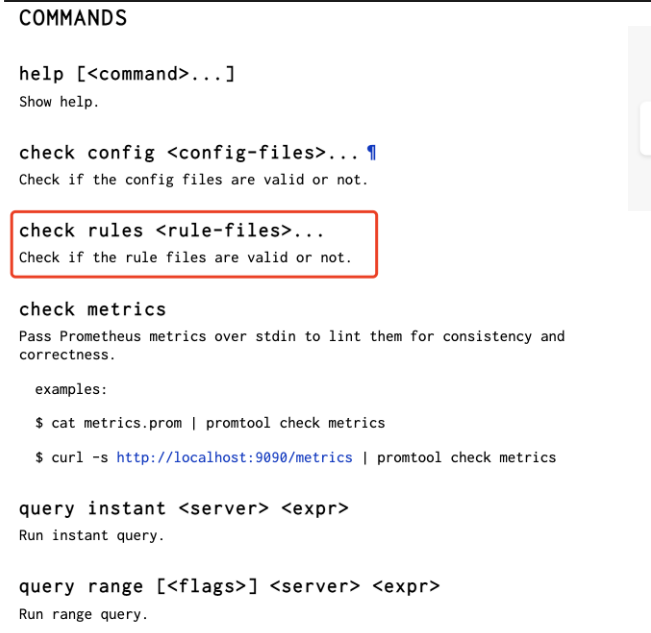


```
infra-prometheus
  rules
    custom-rules.yaml
    k8s-rules.yaml
    network-rules.yaml
    node-rules.yaml
    operator-rules.yaml
    redis-rules.yaml
    storage-rules.yaml
  infra-prometheus-monitors.yaml
  infra-prometheus.yaml
  scrape-config.yaml
```

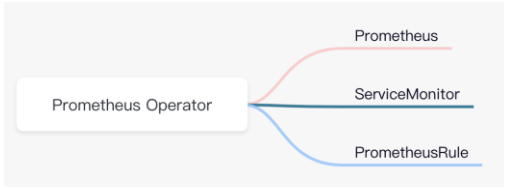
Makefile



```
lint: dependencies
	docker run -v "$(PWD)/chart/rules:/work" -v "$(PWD)/build:/build" --entrypoint /build/lint-prom-rules.sh prom/prometheus
	helm lint ./chart/
```



```
COMMANDS

help [<command>...]
Show help.

check config <config-files>... ¶
Check if the config files are valid or not.

check rules <rule-files>...
Check if the rule files are valid or not.

check metrics
Pass Prometheus metrics over stdin to lint them for consistency and
correctness.

  examples:

  $ cat metrics.prom | promtool check metrics

  $ curl -s http://localhost:9090/metrics | promtool check metrics

query instant <server> <expr>
Run instant query.

query range [<flags>] <server> <expr>
Run range query.
```

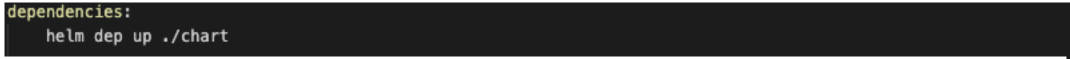


→ api objects are schemas defined in th
CRD
API server do **verify**

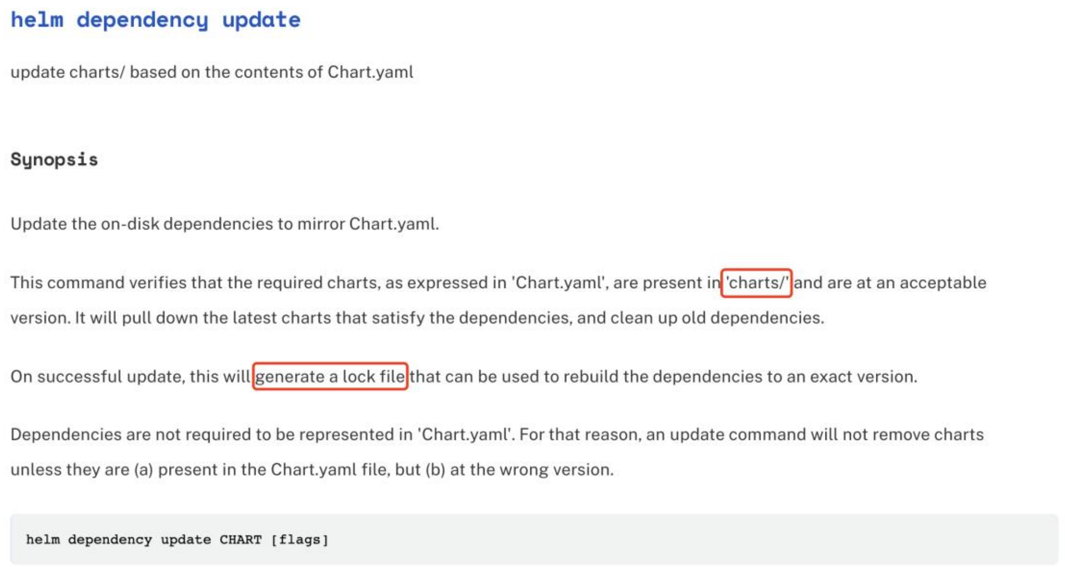
- ☐ ez maintenance of rules
- ☐ apply → verify → reduce risk
- ☐ configmap hard to debug

promtool usage:
https://manpages.debian.org/testing/prometheus/promtool.1.en.html

Makefile



```
dependencies:
	helm dep up ./chart
```



helm dependency update

update charts/ based on the contents of Chart.yaml

Synopsis

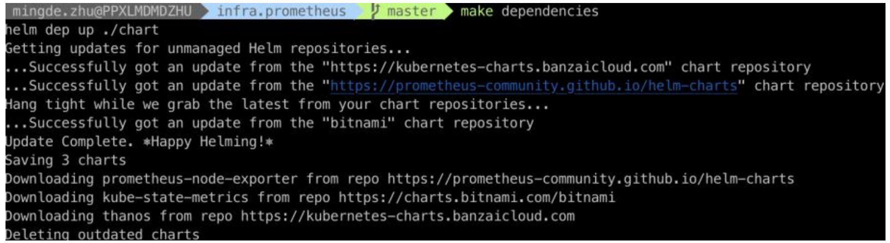
Update the on-disk dependencies to mirror Chart.yaml.

This command verifies that the required charts, as expressed in 'Chart.yaml', are present in 'charts/' and are at an acceptable version. It will pull down the latest charts that satisfy the dependencies, and clean up old dependencies.

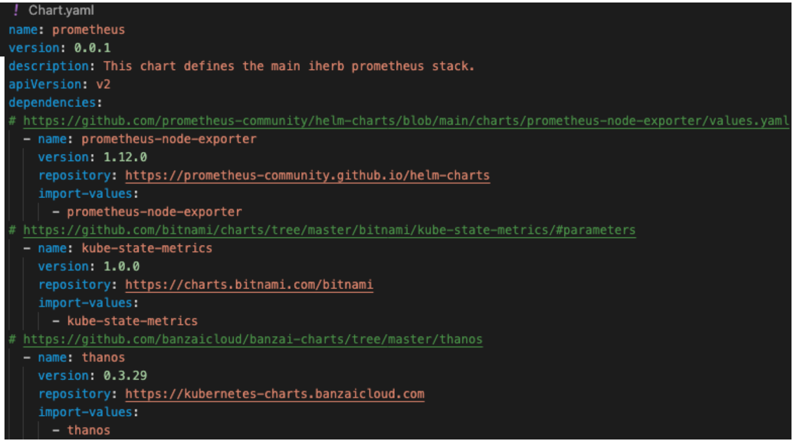
On successful update, this will generate a lock file that can be used to rebuild the dependencies to an exact version.

Dependencies are not required to be represented in 'Chart.yaml'. For that reason, an update command will not remove charts unless they are (a) present in the Chart.yaml file, but (b) at the wrong version.

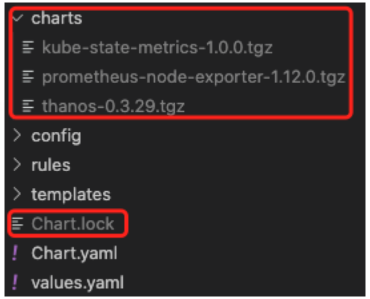
```
helm dependency update CHART [flags]
```



```
mingde.zhu@PPXLMDMDZHU  infra.prometheus  master  make dependencies
helm dep up ./chart
Getting updates for unmanaged Helm repositories...
...Successfully got an update from the "https://kubernetes-charts.banzaicloud.com" chart repository
...Successfully got an update from the "https://prometheus-community.github.io/helm-charts" chart repository
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "bitnami" chart repository
Update Complete. *Happy Helming!*
Saving 3 charts
Downloading prometheus-node-exporter from repo https://prometheus-community.github.io/helm-charts
Downloading kube-state-metrics from repo https://charts.bitnami.com/bitnami
Downloading thanos from repo https://kubernetes-charts.banzaicloud.com
Deleting outdated charts
```

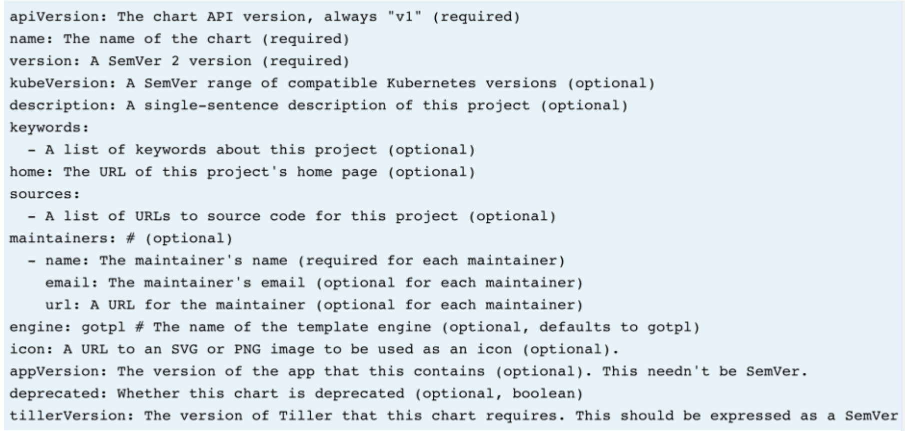


```
! Chart.yaml
name: prometheus
version: 0.0.1
description: This chart defines the main iherb prometheus stack.
apiVersion: v2
dependencies:
# https://github.com/prometheus-community/helm-charts/blob/main/charts/prometheus-node-exporter/values.yaml
  - name: prometheus-node-exporter
    version: 1.12.0
    repository: https://prometheus-community.github.io/helm-charts
    import-values:
      - prometheus-node-exporter
# https://github.com/bitnami/charts/tree/master/bitnami/kube-state-metrics/#parameters
  - name: kube-state-metrics
    version: 1.0.0
    repository: https://charts.bitnami.com/bitnami
    import-values:
      - kube-state-metrics
# https://github.com/banzaicloud/banzai-charts/tree/master/thanos
  - name: thanos
    version: 0.3.29
    repository: https://kubernetes-charts.banzaicloud.com
    import-values:
      - thanos
```

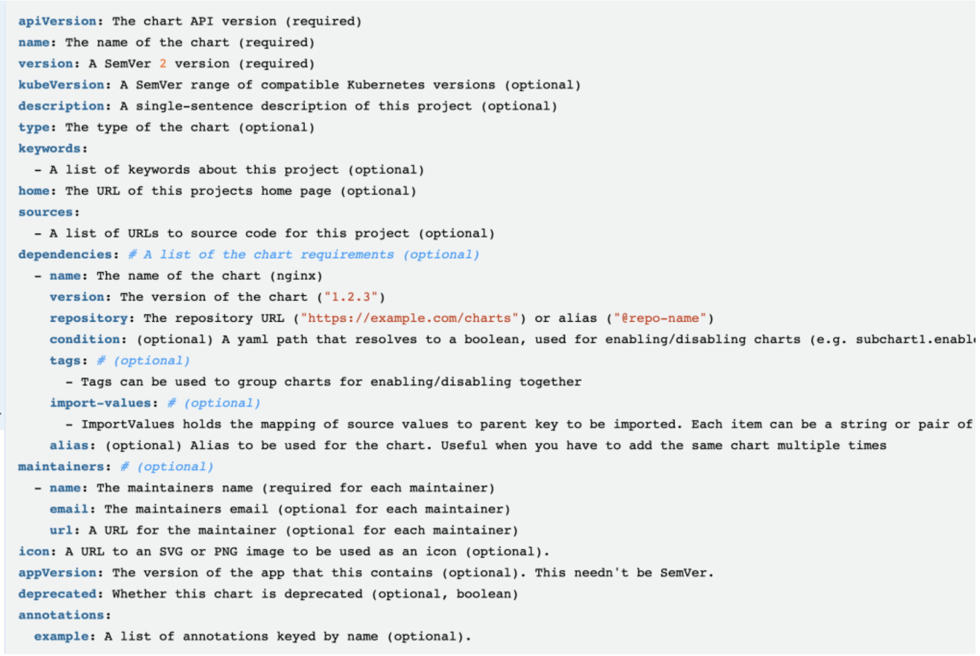


```
charts
  kube-state-metrics-1.0.0.tgz
  prometheus-node-exporter-1.12.0.tgz
  thanos-0.3.29.tgz
> config
> rules
> templates
Chart.lock
! Chart.yaml
! values.yaml
```

apiVersion : v1 vs v2



```
apiVersion: The chart API version, always "v1" (required)
name: The name of the chart (required)
version: A SemVer 2 version (required)
kubeVersion: A SemVer range of compatible Kubernetes versions (optional)
description: A single-sentence description of this project (optional)
keywords:
  - A list of keywords about this project (optional)
home: The URL of this project's home page (optional)
sources:
  - A list of URLs to source code for this project (optional)
maintainers: # (optional)
  - name: The maintainer's name (required for each maintainer)
    email: The maintainer's email (optional for each maintainer)
    url: A URL for the maintainer (optional for each maintainer)
engine: gotpl # The name of the template engine (optional, defaults to gotpl)
icon: A URL to an SVG or PNG image to be used as an icon (optional).
appVersion: The version of the app that this contains (optional). This needn't be SemVer.
deprecated: Whether this chart is deprecated (optional, boolean)
tillerVersion: The version of Tiller that this chart requires. This should be expressed as a SemVer
```



```
apiVersion: The chart API version (required)
name: The name of the chart (required)
version: A SemVer 2 version (required)
kubeVersion: A SemVer range of compatible Kubernetes versions (optional)
description: A single-sentence description of this project (optional)
type: The type of the chart (optional)
keywords:
  - A list of keywords about this project (optional)
home: The URL of this projects home page (optional)
sources:
  - A list of URLs to source code for this project (optional)
dependencies: # A list of the chart requirements (optional)
  - name: The name of the chart (nginx)
    version: The version of the chart ("1.2.3")
    repository: The repository URL ("https://example.com/charts") or alias ("@repo-name")
    condition: (optional) A yaml path that resolves to a boolean, used for enabling/disabling charts (e.g. subchart1.enabl
    tags: # (optional)
      - Tags can be used to group charts for enabling/disabling together
    import-values: # (optional)
      - ImportValues holds the mapping of source values to parent key to be imported. Each item can be a string or pair of
    alias: (optional) Alias to be used for the chart. Useful when you have to add the same chart multiple times
maintainers: # (optional)
  - name: The maintainers name (required for each maintainer)
    email: The maintainers email (optional for each maintainer)
    url: A URL for the maintainer (optional for each maintainer)
icon: A URL to an SVG or PNG image to be used as an icon (optional).
appVersion: The version of the app that this contains (optional). This needn't be SemVer.
deprecated: Whether this chart is deprecated (optional, boolean)
annotations:
  example: A list of annotations keyed by name (optional).
```

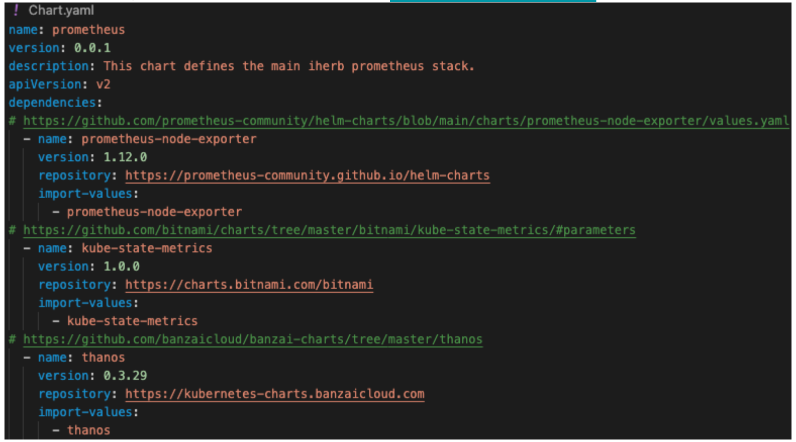
apiVersion : v1 vs v2

The apiVersion Field

The apiVersion field should be v2 for Helm charts that require at least Helm 3. Charts supporting previous Helm versions have an apiVersion set to v1 and are still installable by Helm 3.

Changes from v1 to v2:

- A dependencies field defining chart dependencies, which were located in a separate requirements.yaml file for v1 charts (see [Chart Dependencies]).
- The type field, discriminating application and library charts (see [Chart Types]).

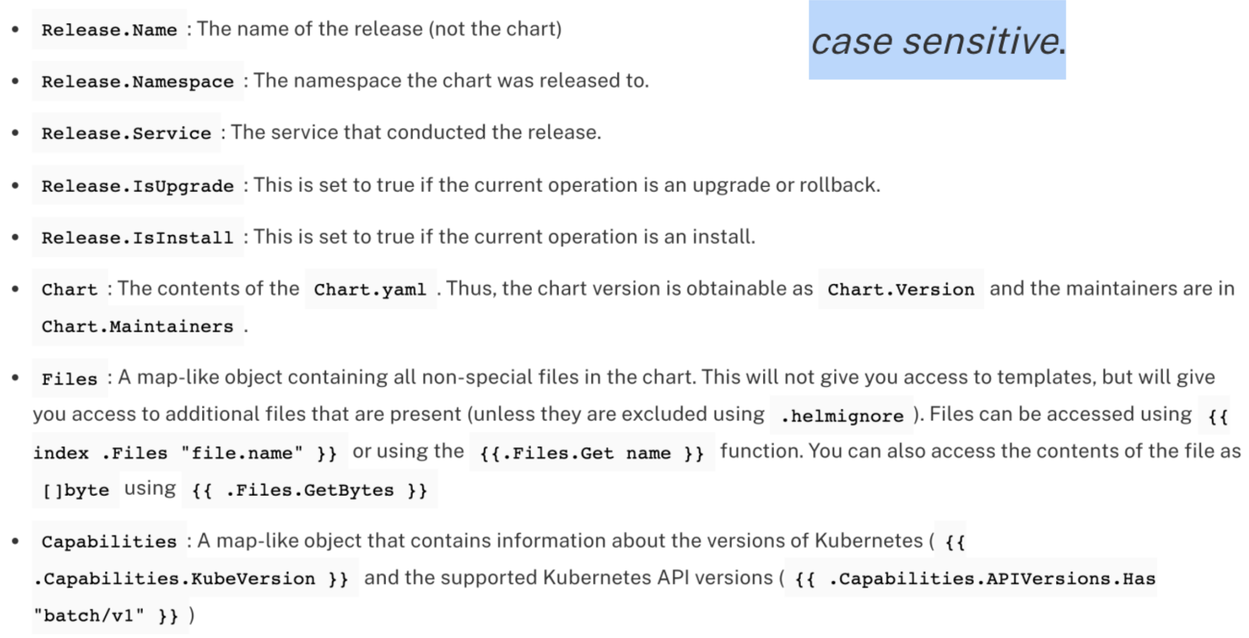


! Chart.yaml

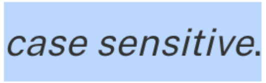
```yaml
name: prometheus
version: 0.0.1
description: This chart defines the main iherb prometheus stack.
apiVersion: v2
dependencies:
# https://github.com/prometheus-community/helm-charts/blob/main/charts/prometheus-node-exporter/values.yaml
  - name: prometheus-node-exporter
    version: 1.12.0
    repository: https://prometheus-community.github.io/helm-charts
    import-values:
      - prometheus-node-exporter
# https://github.com/bitnami/charts/tree/master/bitnami/kube-state-metrics/#parameters
  - name: kube-state-metrics
    version: 1.0.0
    repository: https://charts.bitnami.com/bitnami
    import-values:
      - kube-state-metrics
# https://github.com/banzaicloud/banzai-charts/tree/master/thanos
  - name: thanos
    version: 0.3.29
    repository: https://kubernetes-charts.banzaicloud.com
    import-values:
      - thanos
```

https://helm.sh/docs/topics/charts/#the-apiversion-field

Predefined Values



- `Release.Name` : The name of the release (not the chart)
- `Release.Namespace` : The namespace the chart was released to.
- `Release.Service` : The service that conducted the release.
- `Release.IsUpgrade` : This is set to true if the current operation is an upgrade or rollback.
- `Release.IsInstall` : This is set to true if the current operation is an install.
- `Chart` : The contents of the `Chart.yaml` . Thus, the chart version is obtainable as `Chart.Version` and the maintainers are in `Chart.Maintainers` .
- `Files` : A map-like object containing all non-special files in the chart. This will not give you access to templates, but will give you access to additional files that are present (unless they are excluded using `.helmignore`). Files can be accessed using `{{ index .Files "file.name" }}` or using the `{{.Files.Get name }}` function. You can also access the contents of the file as `[]byte` using `{{ .Files.GetBytes }}`
- `Capabilities` : A map-like object that contains information about the versions of Kubernetes (`{{ .Capabilities.KubeVersion }}` and the supported Kubernetes API versions (`{{ .Capabilities.APIVersions.Has "batch/v1" }}`)

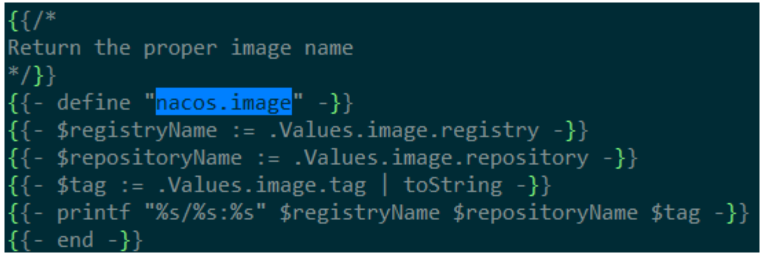


case sensitive.

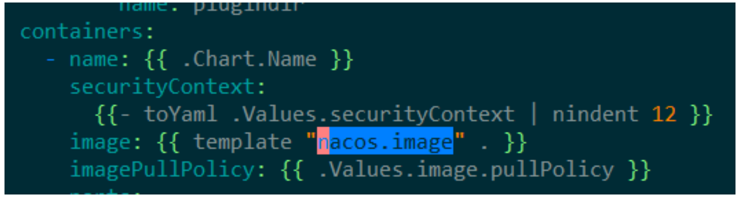
https://helm.sh/docs/topics/charts/#the-apiversion-field

Chart

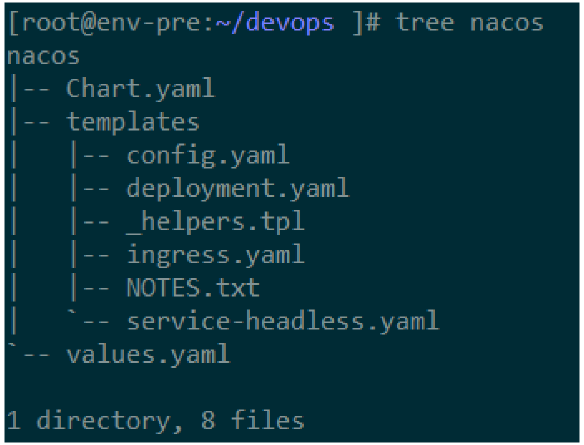
- `_helpers.tpl` : A place to put template helpers that you can re-use throughout the chart



```
{{/*
Return the proper image name
*/}}
{{- define "nacos.image" -}}
{{- $registryName := .Values.image.registry -}}
{{- $repositoryName := .Values.image.repository -}}
{{- $tag := .Values.image.tag | toString -}}
{{- printf "%s/%s:%s" $registryName $repositoryName $tag -}}
{{- end -}}
```



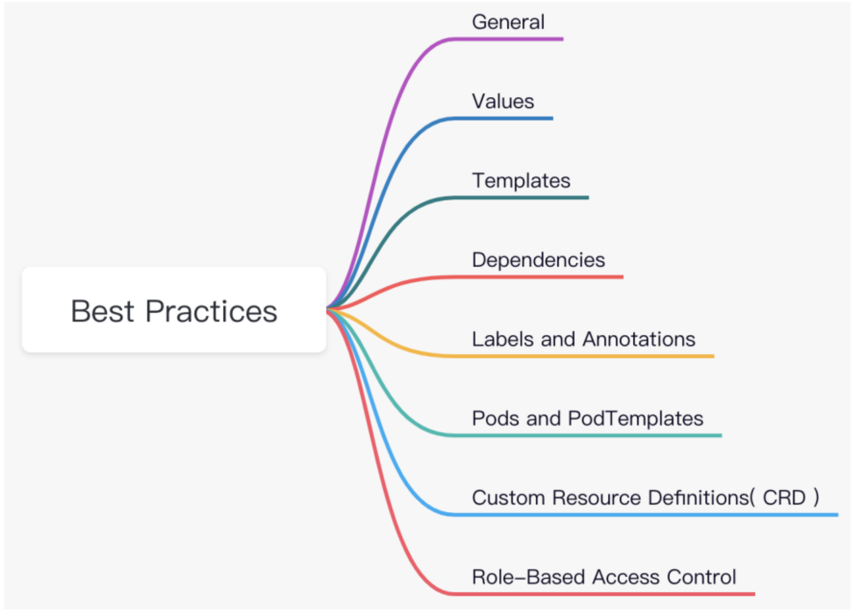
```
        name: plugindir
containers:
  - name: {{ .Chart.Name }}
    securityContext:
      {{- toYaml .Values.securityContext | nindent 12 }}
    image: {{ template "nacos.image" . }}
    imagePullPolicy: {{ .Values.image.pullPolicy }}
```



```
[root@env-pre:~/devops ]# tree nacos
nacos
|-- Chart.yaml
|-- templates
|   |-- config.yaml
|   |-- deployment.yaml
|   |-- _helpers.tpl
|   |-- ingress.yaml
|   |-- NOTES.txt
|   `-- service-headless.yaml
`-- values.yaml

1 directory, 8 files
```

https://helm.sh/docs/chart_template_guide/named_templates/#partials-and-_-files

Best Practices



https://helm.sh/docs/chart_best_practices/conventions/

Thank You!