

SIG API Machinery Subproject Meeting - Declarative APIs and Linters: Agenda & Notes

Zoom Information:

Join Zoom Meeting

<https://zoom.us/j/91574890163?pwd=1uQn4MIKhatwUjk5ATti5b6l8Ylll3.1>

Meeting ID: 915 7489 0163

Passcode: 77777

One tap mobile

+16699006833,,91574890163# US (San Jose)

+16694449171,,91574890163# US

Join instructions

<https://zoom.us/meetings/91574890163/invitations?signature=emlgJJNCFwLZadJQNdt5Aclu5EhlUrWMXMsKdzOa1-g>

We'll meet every other Tuesday at 9AM, Pacific Time. We'll keep the meeting to 30 minutes unless there's a good reason to go longer.

This meeting agenda gets sent to the SIG each Monday afternoon for people to propose agenda topics, and each Tuesday afternoon for visibility of the upcoming agenda.

When adding items:

- If your item is about a design/pr/issue, give some indication of what you're hoping to achieve: is it informational? To identify blockers? To get advice / help / mentorship? To get a decision?

Key Objectives

1. **Simplify API Development:** Replace complex handwritten Go code with declarative expressions
2. **Improve Review Process:** Reduce cognitive load on API reviewers through automated validation
3. **Unify Experience:** Bridge the gap between Kubernetes native types and CRDs

4. **Enable Automation:** Facilitate automated API quality checks and documentation generation

Slack Chatroom

- For discussion topics related to this meeting, we use the
 - [# sig-api-machinery-dev-tools slack channel](#)




Youtube Recordings

- Playlist of all meeting recordings -
<https://www.youtube.com/playlist?list=PL--jPUI8HA10losS6xSbUstXIG05liHkz>

Agenda Backlog (please add your name, email or slack when adding an item)

-

Dec 2, 2025

- Meeting host
 - Aaron Prindle
- Upcoming meeting dates
 - Dec 16th , Dec 30th 
- [joelspeed] Optional+required - kubecon discussions & takeaways
 - Consensus on structs
 - Non-pointer structs, came up w/ rules that are ok
 - Linter rules agreed upon 
 - “Any non-pointer struct w/ a required is considered required”
 - Any non-pointer struct w/o required -> optional
 - Structured and unstructured clients
 - Spec is required vs spec.replicas is required
 - Get error either way, not concerned about identical
 - Joelspeed needs review on PR!
 - <https://github.com/kubernetes-sigs/kube-api-linter/pull/189>
 - [bryce] On my list
 - Mark all structs
 - In future - want empty slice
 - Use pointer, linters says this
 - Unstructured client, might still matter
 - Import into CRDs, still need markers
 - DV
 - Follow rules, error if not applied

- [discussion] Thoughts on getting API reviewers involved with this meeting to discuss API standards related to Declarative APIs?
 - Common issue around DV, kube-api-linter, etc. is that the projects require standards and case-by-case
 - None exist as the implementation, etc. is bespoke/different across k/k
 - There are common patterns used across implementations but it is unclear if they are the preferred patterns
 - Tim spoke w/ Bryce and Joel about being API reviewers
 - Bryce pursuing this 😊
 - JoelSpeed
 - 2 avenues in calls
 - Implementation
 - Coordination (CRDs and native types)
 - Defining conventions
 - Tim, David, etc. very busy
 - ARE IN SIG ARCH call
 - Maybe we distill here, get agenda and take it to sig arch call
 - End up with more opinions from the sig arch meeting
 - API best practices sources of truth
 - Relevant for DV, kube-api-linter, and possibly AI assisted API review
 - [Kubernetes API Conventions](#)
 - ???
 - Project News/Updates
 - Declarative Validation
 - Kube-api-linter
 - TODO
 - Get this meeting added to k8s calendar proper
 - PRs
 - @itzPranshul - [Enables Declarative validation for ClusterRoleBinding #135050](#)
 - @priyansh3006 - [feat: wired coordination group for declarative validation and k8s:minimum in Lease.LeaseTransitions and Lease.LeaseDurationSeconds #135043](#)
 - @itzPranshul - [rbac: migrate PolicyRule.verbs to Declarative Validation #135028](#)
 - @priyansh3006 - [Mark spec.selector as required in ReplicationController and add tests #135010](#)
 - @darshansreenivas - [feat: wire cisinode group for declarative validation #134983](#)
 - ~~[MERGED] @bryce - +maxProperties PR in dev branch~~

Nov 18, 2025

- Meeting host
 - Aaron Prindle
- Kubecon

- Bryce and Joel: Simplifying API Review: A Tool Based Approach for Kubernetes and Beyond
 - <tbd-recording-link>
 - Lots of interest here w/ kube-api-linter. Excited about best practice standards
 - Certify - how to make sure changes we are making are compatible with users
- Yongrui: Taming the Beast: The Move To Declarative API Validation in Kubernetes
 - <tbd-recording-link>
 -
- Optional + required discussion
 - Unconference related to this
 - Landed on: for non-pointer types, if there is a required field in that - means the type is also required FOR BUILT-INS. Status-quo for CRDs
 - @joelspeed updating conventions related to this
 - Maybe add a new linter s.t. A non-pointer type is marked as required if a non-pointer type is there within it
- Other discussion/themes
 - ...
-
- Currently in seeing external contributor migrations there are a number of cases where an empty string returns an invalid error going straight to a format validation instead of distinguishing empty string as required error and invalid error for invalid format cases.
 - Example - (returns Invalid for "")
 - <https://github.com/kubernetes/kubernetes/blob/master/pkg/apis/storage/validation/validation.go#L202-L209>
 - Opinion on saying "required" for these, to be consistent.
 - Required and optional mean something different in openapi, and the goal is, I think, to get as close to each other as possible
- lalitchauhan@google.com - optional vs required open-api/CRD vs DV
 - Open api - key present in schema
 - Works at a diff layer in the json schema
 - DV - present AND non-zero
 - Path forward ongoing
 - @bryce - consensus around error handling in CRDs and built-in types
 - OK for some differences, some things in open-api can't be represented in non-pinter types in built-in
 - @joelspeed has convention takeaways in notes
 - Can circle back w/ Tim and Jordan to make sure we get the takeaways here.
 - OK for differences in handling this
 - <https://github.com/kubernetes-sigs/kube-api-linter/pull/189>
- Project News/Updates

- Declarative Validation
 - V1.35 complete
 - v1.36
- Kube-api-linter
 - Looking at cutting a versioned release soon 🏆
 - Merged a # of new linters
- PRs
 - @itzPranshul - [Enables Declarative validation for ClusterRoleBinding #135050](#)
 - @priyansh3006 - [feat: wired coordination group for declarative validation and k8s:minimum in Lease.LeaseTransitions and Lease.LeaseDurationSeconds #135043](#)
 - @itzPranshul - [rbac: migrate PolicyRule.verbs to Declarative Validation #135028](#)
 - @priyansh3006 - [Mark spec.selector as required in ReplicationController and add tests #135010](#)
 - @darshansreenivas - [feat: wire cisinode group for declarative validation #134983](#)
 - @bryce - +maxProperties PR in dev branch

Nov 4, 2025

- Meeting host
 - Aaron Prindle
- K8s News
 - Code freeze 11/6 🚧
 - Kubecon ~11/10
- [Joel] What do **optional** and **required** actually mean?
 - In CRDs, they mean “is this key present”
 - Key present in serialize obj. Field foo, is it there?
 - Could we achieve the same for built-ins? What are the technical limitations of this?
 - Today we have only the decoded request at validation time?
 - Go type
 - Doesn't say if key exists
 - At X point we lose serialized data
 - 3 types - CRD, Open API, DV tags
 - Kube openapi tags not playing role in validation
 - Only CRD validation
 - DV work on go object. Doesn't know serialization
 - For DV, required means object non-zero value
 - Struct, non-empty
 - Pointer - non-null
 - In golang, can't “not set” non-pointer struct
 - Can't have same semantics as at diff layers
 - Joel - can we fix this @ API layer for proper optional/required semantics

- If required string, need min length 1
 - Required doesn't mean non-empty
 - 2 opts
 - 2x tags (DV + Open API)
 - Overload DV tags
 - No DV tag on non-pointer struct
 - Joel - want one set of tags ideally. +k8s:[optional|required] ideal
 - BUT want it to behave 1:1 w/ CRDs -> odd if embed upstream core type into CRD. Won't behave 1:1 as DV tag use 😞
 - Q's - what do optional/required mean? What is possible?
 - Could we also have a way to detect key presence?
- Context, in debate with thockin on what optional and required mean
 - See [comment](#)
 - Document [K8s: optional and required vs. struct-type fields](#)
 - Lists in proto, can't have nil list
 - If need min len list in CRD -> easy
 - NOT WORK w/ BUILT-IN
 - Ex: Status is marked optional BUT not a pointer
 - Opts:
 - Could have exception on non-pointer structs
- Aaron [noted](#) that this is forbidden today in the code generator
- Kubecon
 - Talk w/ @thockin and @yongruilin & @jpbetz (at maintainer summit?)
- **[Aaron] Short-circuit DV tags**
 - Declarative validation tags that short-circuit currently are
 - +k8s:optional, +k8s:required, +k8s:immutable, +k8s:max[Items|Properties]
 - Short-circuit tags must be well documented as if a user wants to migrate field Z to use DV, instead of being able to do it in an isolated way they must look at the fields path spec.X.Y.Z and see if X or Y are short-circuit tags.
 - This is done by looking at the [validation.go](#) and seeing if those fields have logic associated with any of the short-circuit and migrating those as well
 - Q: should union tags be short-circuit as well?
 - apiserver/Kubebuilder short-circuit
 - Don't fire CEL validations if other validations fire first
 - Marke field required but frozen other won't
 - **More research on how this is implemented until we get more storage information**
- **[Aaron] Common patterns found during DV migrations where there is additional friction for migration (sometimes hand-written code being changed is preferred)**
 - Hand-written code outputs duplicate errors
 - Short-circuit error ordering
 - Eg: DV has immutable/maxItems short-circuit but hand-written doesn't

- Fields were moved across API versions
 - DV has a supported way to let our error matcher know this but not well documented currently
- Shared validation method that has N validation errors but DV tag added to only one of the types that use the shared method -> requires "MarkCoveredByDeclarative" on the method but need a way for caller to specify it's error should be marked for one of the errors
 - Requires plumbing options to the hand-written shared validation method
- +required field does not have an explicit hand-written Required error output for that field
- [Lalit/Joel] **DV vs kubebuilder type vs local validations** - In DV today if a type has a validation and a field has a related validation (ex: both have +k8s:maxLength) both validations would be run (eg: +k8s:maxLength=10 and +k8s:maxLength=12) but for kubebuilder tags the field/"local" validation is able to override the type tags for these cases.
 - We can revert the local override feature in controller-gen.
 - It is confusing, people will ask the questions.
 - <https://github.com/kubernetes-sigs/controller-tools/pull/1270>
 -
 - +k8s:opaque can help here
 - Hard one for controller-tools
 - "I added tag here BUT doesn't work 😞"
 - Revert recent Joel PR but fix bug w/o local override and then implement opaque for local override
- Recent convos/info
 - **Field map key representation** - Why did we decide to represent a proper map key field path in DV? Today we are using a field path up to the map name, not the key name. Should we show it with the key? DRA is using up to the key.
 - Support map in CRD for native types
 - Q: what field path want to use for key? What to use for values?
 - In DV - we use up to map name. spec.Map1, spec.Map1[key]
 - For CRD can't write validation for key BUT once KEP goes in will be possible
 - Joel - currently can't do key validation in Map
 - BUT 3.1 can - So lets grab that in 3.0
 - listType=map
 - Can't write validations for idx BUT how field path?
 - Use idx #
 - spec.ListMap1[1]....
 - Joel - list w/ 3 obj
 -

```
// +k8s:item(type:
"Approved")=+k8s:zeroOrOneOfMember
// +k8s:item(type:
```

"Denied")=+k8s:zeroOrOneOfMember

- +k8s:eachKey=+k8s....
 - Joe Betz Related KEP - [CRD Map Key Validation #5667](#)
 - Lalit Chauhan - merged one PR using path field path name. Errors not part of API, can change
- Project News/Updates
 - Declarative Validation
 - @darshansreenivas PR merged 🎉 (2nd new DV contributor PR!)
 - [feat: wire storage group for declarative validation and +k8s:required to StorageClass.Provisioner #134796](#)
 - [\[Umbrella\]\[Declarative Validation\] Enable Declarative Validation for APIs #134280](#)
 - Kube-api-linter
 - k/k PRs ongoing for getting +optional & +required lint rules
 - [Tracking: Enable Kube-API-Linter rules #134671](#)
 - Specifically
<https://github.com/kubernetes/kubernetes/issues/134671#issuecomment-3415486017>
 - Flurry of activity last week
 - Handful of new linters being worked on (noreferences, arrayofstruct, defaultorrequired)
 - Some bugs being fixed
 - Working on changing output messages to `<struct>.<field>`
 - Working towards v0.1.0 release soon (setting up goreleaser)
 - [Docs for all current linters](#)
- Kubecon
 - Bryce and Joel
 - Kube-api-linter talk!
 - [Sunday morning](#)
 - Yongrui Lin
 - Declarative Validation
 - <https://maintainersummitna2025.sched.com/event/28aDM/taming-the-beast-the-move-to-declarative-api-validation-in-kubernetes-yongrui-lin-google>
 -
- PRs to review (not v1.35 code freeze blocking)
 - Declarative Validation
 - @itzPranshul - [Enables Declarative validation for ClusterRoleBinding #135050](#)
 - @priyansh3006 - [feat: wired coordination group for declarative validation and k8s:minimum in Lease.LeaseTransitions and Lease.LeaseDurationSeconds #135043](#)
 - @itzPranshul - [rbac: migrate PolicyRule.verbs to Declarative Validation #135028](#)

- @priyansh3006 - [Mark spec.selector as required in ReplicationController and add tests #135010](#)
 - @darshansreenivas - [feat: wire cisinode group for declarative validation #134983](#)
 - Kube-api-linter
- Als
 - Aaron Prindle - remove positional args from DV tags
 - <https://github.com/jpbetz/validation-gen/issues/217>

Oct 21, 2025 (Next Meeting)

- Meeting host
 - Aaron Prindle
- [chrischdi]: Markers for custom-resource metrics:
 - Related Controller-tools PR:
 - <https://github.com/kubernetes-sigs/controller-tools/pull/1043>
 - Note: the PR description has examples for the markers.
 - Question: Is there precedence for similar markers / patterns so we stay consistent?
 - ===
 - Review around defining markers - @JoelSpeed
 - Kube-state metrics - allows kube state metrics to scrape CRDs
 - Usually only for core types
 - Stuff exists already but getting to PR
 - Status.Replicas -> goes to a metric
 - Infer path from marker location
 - Labels on this config
 - Info metrics via labels
 - Counters like replicas, available metrics
 - State set metrics - conditions
 - Recommendations
 - Use a controller-gen specific prefix
 - Metrics on namespace might not be ideal
 - "Kubebuilder:"
 - Controller-gen and kubebuilder confusion
 - Controller-gen -> controller-tools 😞
 - K8s: and controller-gen
 - Controller-tools is repo name vs controller-gen
 - Lowercase
 - DV grammar:
 - +k8s:item(name)=+k8s:minimum
 - Joe Betz - have multiple types of tags w/ diff grammars
 - Kubebuilder has nice format w/ :, etc.
 - Important thing is prefix means you are an authority on the rest of that tag

- Kubebuilder is authority on how tag works
 - K8s: k8s is authority on how tag works
 - Agree on - what prefix means and the tag authority
 - ^ worried a bit about "Metric" currently
- [Joel] Local overrides for markers
 - We recently changed the behaviour in controller-tools (see <https://github.com/kubernetes-sigs/controller-tools/pull/1270>)
 - It looks like this might not align with DV
 - Raising questions about the behaviour of the unique markers rule, <https://kubernetes.slack.com/archives/C08MNH7PH9P/p1760702576853699>
 - Unique - will validate maxLength on field and type alias on field also maxLenght -> issue 🤔
 - In controller tools, fixed - re-use type, apply validation on field. Controller-gen apply both -> restricted by stricter rule. If want to have type w/ looser validation not possible. NOW - local override rule wins
 - DV - similar to what used to do
 - DV run both validations on types and fields
 - No issue if validation stricter
 - stricter take effect
 - JoelSpeed - folks expect overrides to be possible
 - Add value to enum
 - Length
 - Not obvious both validations apply
 - Yongrui Lin - author of root type want specific validation
 - If someone use it, they add validation -> lint takes effect and root type validation known to user
 - If override local field, does root type look ability to be authoritative
 - @joelspeed - if import a type, looks up validation for type and reuse
 - In CRD schema, whole new schema each time
 - Maybe ok to be different?
 - "If CRD, don't do this" for linting
 - Lalit Chauhan - type transposed from one field to another
 - Transpose will fail in some cases
 - All of one only
 - Honor root field validation critical here
 - @joelspeed - use coretype for podspec, makes sense
 - In controller-gen, needs extension use case
 - X person doing Y thing
 - Go struct or type alias to string
 - Folks re-using smaller struct
 - Lalit Chauhan - add new value to enum, should be strict
 - CRD wants override for everything

- Stefan - on field validation we only do field validation (local validation)
 - Lalit Chauhan - local maybe more lenient. Copy of Deployment, want to create pod, in this case root validation should be authoritative
 - Stefan - APIs for native might be more closed down.
 - Kubebuilder can maintain this, not in k/k/ currently
 - Want dv tag in k/k
 - If kubebuilder tag has local override, make kubebuilder tag have priority
 - Can still use core type BUT also keep
 - @joelspeed if I use a field validation as a type alias DV tag is overridden
 - Bryce - if CRD handled differently. Why can't controller-gen handle different than k/k?
 - Unique linter - won't trace back to everything
 - ^ will linter chase back all alias and forbid overwrite OR we can just do same thing as kubebuilder
 - @joelspeed - currently type aware and show dupes. BUT suggestion is maybe to add type aware collection -> compatible w/ controller-gen
 - Kubebuilder vs DV tags different
 - Differentiate by tag type?
 - Hoping long term we stop kubebuilder tags for DV tags
 - @joelspeed - in DV reused, in CRD we re-gen schema for type
 - Stefan - DV marker and kubebuilder marker over long term
 - Core type w/ N markers, now need to use allOf
 - Now embed in CRD
 - Controller-gen doesn't know where type is coming from (core or NOT core)
 - allOf in coretype, embed in CRD. then want to keep allOf
 - Don't want entire struct to behave diff
 - @joelspeed - adding unique markers in k/k prevents this
 - Summary
 - Aaron Prindle - 2 patterns
 - Yongrui Lin + @joelspeed - linting
- [jpbetz] [OpenAPI 3.0 vs. 3.1](#)
 - Horrifically backward incompatible
 - A full upgrade would be large, disruptive. Not convinced it's worth it.
 - Why should we care?
 - **propertyNames** generally useful as a way to attach validations to map keys
 - CEL needs to know map key max lengths for estimated cost
 - ===
 - Labels in k8s - map of string string BUT validation on both values

- CRD - kubebuilder type import another type
 - Can't validation keys, not sure if valid labels (can't val keys atm 😞)
 - Also allows setting maxLength and get cost here
 - Can't handle labels atm well
 - Can we do an x-kubernetes-??? And stay on 3.0
 - Semver broken promises 😞
 - Open API 3 schema
 - @joelspeed - does this preclude CEL usage
 - Joe Betz - not exactly
 - [Bryce] A couple questions related to DV tag parsing in kube-api-linter. I started working on this and have WIP PR: <https://github.com/kubernetes-sigs/kube-api-linter/pull/152>
 - Has there been a decision on <https://github.com/jpbetz/validation-gen/issues/85> ? I think it would be helpful for tooling like kube-api-linter to be able to have a guarantee that DV tags will always be parsed in a way where we can know the "name" of all arguments passed in.
 - Wanting to make sure I have a good understanding of how "conditional" tags are expected to be used long term. Do we ever envision something like `//+k8s:ifEnabled("my-feature")=+k8s:required,+k8s:maxLength=10` being valid?
 - ===
 - Long term - maybe rework on how to store validation for api-linter
 - No concept of nested tag payloads for controller-gen markers
 - DV tags have unnamed and named tags
 - Bryce - requiring names might be helpful here for other tooling
 - Positional only args difficult to parse
 - For WIP PR, we have "unnamed" name for handling this
 - Joe Betz - if more than 1, need all to be named
 - Maybe we should name
 - Bryce - DV tag parsing logic, it is represented as X named arg
 - Joe Betz - if we want to extend this, then we need to re-key first item. So maybe better to not have this? Always call this X value
- ```
+k8s:example(1)
+k8s:example(min: 1)
+k8s:example(min: 1, max: 2)
```
- Joe Betz - don't allow positional args
      - Second param no place to do this
      - Aaron Prindle can do this ^^
  - Kube-api-linter rules likely need to parse conditions
    - Bryce - is allowed to pass multiple tags
      - Joe Betz - one tag as payload in the grammar. E.g you would use N lines

- Yongrui Lin - PR open for grouping tags. Currently can't change multiple validation under one conditional BUT can separately
    - Currently trying to group under <yongrui-pr-link>
- ===
- Topics from last meeting:
  - Openshift feature gating // matrix of N schemas
  - Optional + required in k/k
    - Tracking issue for K/K KAL integration in <https://github.com/kubernetes/kubernetes/issues/134671>
- Help wanted issues related to Declarative APIs Subprojects
  - Declarative Validation
    - [\[Umbrella\]\[Declarative Validation\] Enable Declarative Validation for APIs #134280](#)
  - Kubernetes-api-linter
    - [Tracking: Enable Kube-API-Linter rules #134671](#)
    - Open issues for all ongoing lint rules for k/k
- Declarative Validation Updates
  - DRA Migration ongoing, see [staging/src/k8s.io/api/resource/v1/types.go](https://staging/src.k8s.io/api/resource/v1/types.go) for example of DV tags in action
    - ~40 DV tags added (not including +k8s:optional) using ~12 unique DV tags (splitting out +k8s:format).
  - v1.36 Planning
    - “Cross-field” category of validation is a planned item. Examples
      - Mutually required
      - Mutually exclusive
      - Validation of field X depending on the value of another field
      - +k8s:minimum(spec.minAllowedConfig)
      - ^ any early thoughts on cross-field validation generally in other projects as we plan? (controller-tools, etc.)
- Kube-api-linter Updates
  - ...
- Declarative Validation PRs in Queue From Contributors
  - @priyansh3006 - [Enable Declarative Validation for ConfigMap #134603](#)
  - @itzPranshu - [\[RBAC\] Enable Declarative Validation\(DV\) support for ClusterRole and RoleBinding #134537](#)
  - @darshansreenivas - [Enable Declarative Validation for storage #134653](#)

## Oct 7, 2025 (Next Meeting)

- Meeting host
  - Aaron Prindle
- Go through linting wishlist

- Go into k/k
  - Joel: [K/K API linter enablement](#)
    - Barrier to enable is breaking issue - required golangci tagging across current code
    - 3 tags have more gaps
      - Maxlength
      - Required
        - "If 0 value required, doesn't need to be a pointer".
        - IS struct 0 value value, ... -> recommend field should be empty, pointer, etc.
        - <td-link>
        - <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#serialization-of-optionalrequired-fields>
  - ^ Currently implemented
  - ~17 known future rules WIP
- Depend on DV for k/k
- From @stts on Slack

- Be explicit about optional and required.
  - Joel: optionalorrequired rule
  - ^ need feedback for reviewers as there are ~900 exceptions (no +optional OR required 😞)
- For enums, document the possible values.
  - Joel: In the godoc? Not really a linter thing IMO, better suited to AI
    - WIP [Claude.md](#) AI review tool
    - [OpenShift working on Claude for this](#)
    - Walter Fender - PR out starting convos here.
  - ML hint files on k8s
  - gke-labs - <https://github.com/gke-labs>
  - DV review bandwidth constraint
    - AI could help!
  - Lint focus for now, AI future
- Document defaults.
  - Joel: As above with Enums
- MinLength on non-optional strings.
  - Joel: Should also be optional, we can tell omitted from set to zero length string so why not required a non-zero length string when set
- Make the descriptions start with the JSON field name, not with the Go field name.

- Joel: jsontags rule
- Pointers for optional fields.
- Joel: optionalfields rule

- Immutability, ratcheting, and feature gating

- Ratcheting

- Summary:

- Default ratcheting is added to all DV validated fields

- 

```
// don't revalidate unchanged data
if op.Type == operation.Update && (obj == oldObj || (obj
!= nil && oldObj != nil && *obj == *oldObj)) {
 return nil
}
```

- 

- Lists have special ratcheting semantics based on listType=...

- ...

- KEP: <https://github.com/kubernetes/enhancements/pull/5292>

- validation-gen discussions/issues:

<https://github.com/jpbetz/validation-gen/issues?q=is%3Aissue%20ratcheting>

- Recent ratcheting discussion

- Slack:

<https://kubernetes.slack.com/archives/C0EG7JC6T/p1759530446177879>

- Related k/k issue:

<https://github.com/kubernetes/kubernetes/issues/113482>

- Immutability

- Summary:

- +k8s:immutable

- Non-lists -> +k8s:update=NoSet+NoModify+NoClear

- lists -> +k8s:update=NoSet+NoAddItem+NoRemoveItem +  
+k8s:eachVal=+k8s:immutable

- +k8s:update=[NoSet|NoModify|NoClear] (non-lists)


- +k8s:update=[NoSet|NoAddItem|NoRemoveItem] (lists)

- Leverages ratcheting for equality checking // identifying when a change has occurred

- validation-gen Discussion:

<https://github.com/jpbetz/validation-gen/issues/63>

- Design document:

 [PUBLIC] Declarative Validation: Immutability States, Transitions, a...

- KEP Update PR (unmerged):

<https://github.com/kubernetes/enhancements/pull/5614>

- Feature Gating
  - validation-gen discussion:
    - <https://github.com/jpbetz/validation-gen/discussions/149>
  - if[Enabled|Disabled] format - DV vs CRD usage
  - No feature gating in CRDs
    - Built on top by having multiple version. Preview and stable version. Operator framework, etc. have similar
    - Markers w/ optional enabled. Include/exclude part of schema based on marker
    - Require a payload - wouldn't work w/ CRDs, wouldn't exclude a schema
    - Upstream type embedded in CRDs
    - Feature gate in built-in type.
    - ifDisabled()=+k8s:forbidden
    - ifEnabled() [CRD]
      - tooling , if enabled include schema OR completely exclude
      - Enum
      - Max items in a list
        - Specific marker in controller-tools. Merging logic in schema.
      - CEL vary w/ gating
  - Summary:
    - GOAL: CRDs and built-ins to use same markers
    - [native] ifEnabled built-in has payload
    - [CRD] ifEnabled optional payload
    - Call them out as separate, make differences explicit
    - For CRD, ifEnabled no payload OR optional?
      - [CRD] ifEnabled w/ no payload goes for exclusion
      - BUT - [CRD] ifEnabled()=+k8s:maxLength -> works similar to DV
      - Controller-tools build on openshift current work
        - included/not-included
      - Bryce - openshift maybe upstream how we inject handling partial schemas.
        - "This is alpha, this is beta" - 3 schemas. Promote feature -> find marker and upgrade
        - Openshift -> feature gates -> matrix of N schemas
          - Change options
- How to contribute to DV
  -

How to mark an issue for DV:

- [Declarative Validation] in title (in non feature-branch)
- Help-wanted tag
- Good-first-issue (if applicable)

[WIP] README.md file - [\[WIP\] README.md file containing an overview of the feature-branch and high level overview of how to do common tasks](#)

Sample PRs:

- Creating a new tag
  - [feat: add +k8s:neg=<disallowed-comparable-value> validation tag #94](#)
- Enabling Declarative Validation For an API
  - [Enable Declarative Validation for resource.k8s.io v1/v1beta1/v1beta2 #134072](#)
- Migrating a validation from hand-written to Declarative Validation
  - [Add declarative validation +k8s:maxItems tag to ResourceClaim#134211](#)

k/k

- [\[Umbrella\]\[Declarative Validation\] Enable Declarative Validation for APIs](#)
  - @itzPranshul has a PR out doing a task from this umbrella issues - enabling DV for ClusterRole 🚀
- TBD - [Declarative Validation] Migrate specific validation to use Declarative Validation for APIs
  - ...
- New tags?
  - Currently these issues are tracked in the validation-gen feature branch as we try to only upstream a tag when it is used in a migration
- Related issues spawned from DV:
  - [Verify if calling both ValidateObjectMeta and ValidateObjectMetaUpdate for update validation is necessary and update callsites if not necessary #134444](#)

[validation-gen feature branch:](#)

- Build out OpenAPI tags for DV w/ known semantics
  - [Add +k8s:maxProperties tag #208](#)
  - [Add +k8s:minItems tag #205](#)

controller-gen:

- Add DV parser (code tags)
- Start with +k8s:optional and +k8s:required

kube-openapi:

- All migrated tags, translate to openapi schema
- maxItems
- Format (some cases)

===

Higher level project ideas

- 1) generating better field level docs based on annotations
- 2) shift left some of the validation -> IDE validation
- 3) ML + annotations -> improved k8s usage for AI

- Kube-api-linter

- 

Enabling linters in k/k

- Best place is to look @ the table above
- Turn it on, run golang ci linting, see the issues
  - ^ exceptions are likely largest workstream piece
  - Look for JoelSpeed PRs in k/k
  - Breakdown into different API groups

- 

- Parsing DV tags (chaining syntax, etc.)
  - Bryce interested in this for kube-api-linter
  - DV markers - k/k validations, need DV marker support
- Early DV v1.36 planning
  - Cross-field validation
  - DV vs CRDs
  - “DV Only”
- Project Updates
  - Declarative Validation
    - DRA migrations
  - Kube-api-linter
    - Next steps - PR runnable bin
    - Go releases
    - Help ena
- Next meeting
  - Openshift feature gating // matrix of N schemas
  - Optional + required in k/k/

Sep 23, 2025

- Meeting Host
  - Aaron Prindle
- Agenda
  - Declarative Validation Overview and Roadmap
    - Slides
      - [PUBLIC] New Kubernetes Feature - Declarative Validation /...
      - v1.33
        - validation-gen framework core completed

- Early set of base tags developed in a dev-branch
  - One validation migration - ReplicationController.Replicas
    - +k8s:minimum tag
- v1.34
  - Validation-gen framework complex cases workstream
  - Tags extended to include +k8s:unionMember and +k8s:zeroOrOneOfMember
  - Ratcheting implementation
  - One validation migration
    - CertificateSigningRequestStatus.Conditions
      - Validation: at most one of terminal states Approved OR Denied
- validation-gen dev-branch - <http://github.com/jpbetz/validation-gen>
  - ^ for getting involved and getting into discussion there
  - **Aaron Prindle to tag help-wanted**
  - **Joe Betz - do folks want to hear about immutability, ratcheting or feature gating here**
    - [joel.speed@hotmail.co.uk](mailto:joel.speed@hotmail.co.uk) +1 🙋
    - [sbueringer@gmail.com](mailto:sbueringer@gmail.com) +1
- Joel Speed - stability levels?
  - We have a graduation process for tags where tags graduate across stability levels
  - KEP update incoming around this
  - @jpbetz - discussion around go/no-go. We have informal agreement to go forward here and support DV Only validations
- Current state of project
  - KEP:
    - <https://github.com/kubernetes/enhancements/tree/master/keps/sig-api-machinery/5073-declarative-validation-with-validation-gen>
- Docs
  - <https://kubernetes.io/docs/reference/using-api/declarative-validation/>
  - [https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api\\_changes.md](https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api_changes.md)
  - <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md>
- Codifying validation best practices
  - Recent docs PR:
    - <https://github.com/kubernetes/community/pull/8616>
- Migration Plan and Call To Action
  - Asking folks that are working on new APIs with net new validation code to use Declarative Validation where possible
  - Also asking folks to begin migrating simple tags
- Kube api linter

- Recent updates
  - Linter aimed at API types and easing API review
  - 22 rule implemented, 15 open issues for other rules (as of 23rd Sep)
  - Enabled both ssatags (listType) and conditions linters on K/K
  - Need to prio:
    - Updating docs for golanci-lint v2
    - Creating an entrypoint that we can build standalone
      - Module & plugin atm
    - Publishing a v0.1.0
    - Investigating additional +k8s marker implications
  - Usage: Cluster api, openshift, k/k
  - @jpbetz - +1 for optional and default linting 🤖. Difficult case for reviewers
- Ongoing API Conventions and Validation Best Practices Discussions
  - Immutability
  - Ratcheting
- Recent conversations:
  - IDL Tag Migration Strategy: future of existing markers like +optional and whether they'll be deprecated in favor of new declarative validation markers
  - Controller-gen Integration & Standardization: Discuss feature parity between the new k8s IDL tags, what's currently available to CRD developers through kubebuilder, and standardizing IDL tag semantics between k/k and controller-gen
- joel.speed@hotmail.co.uk - core API types discussion
  - Missing opinion of CRD authors
  - Stefan, Bryce, Joel have experience here
  - Ex: ifEnabled - w/ 4 types need payload
    - In CRD doesn't make sense, gateway API. identical validation in open-shift
    - Q: w/ all these discussion, how do we make sure the markers work for types BUT we consider CRDs
    - "This is how it works" -> 🤔 w/ CRDs
      - Maybe identical is not feasible but close as possible
    - Joe Betz - some tags are easy, 1:1 w/ Open API
      - Enum is similar
      - Union -> oneOf is similar
      - ifEnabled and feature gates
    - Joe Betz - in apiserver doesn't go if feature -> then
      - We do two passes
        - In use detection
        - If feature gate is on OR in use -> can use feature
        - Gate in the sense obj is in once in

- ^ complex mechanism
- Yongrui Lin - kube-apilinter SSA tags in k/k
  - DV has own SSA tags w/ k8s: prefix
    - Both exist long-term
    - Opinion, support both in kube-apilinter
  - Eventually we want to deprecate old tags
  - Kube-apilinter - don't use kubebuilder, use upstream
    - End goal is upstream versions (+\* -> +k8s:\*)
    - Don't currently have marker duplication
  - Better to support both now, we are about to open gate for DV Only
    - New API use +k8s:\* prefix tags, if we have linter that covers this - need to extend linter
  - Bryce - can forbid certain markers
    - Maybe exclude linting existing APIs and enforce rule where not +k8s: this is forbidden
  - joel.speed@hotmail.co.uk - tag chaining is new, kube-apilinter parsing doesn't handle this new chaining currently
    - How will this work?
      - +k8s:ifEnabled(=)+k8s:listType....
        - ^ SSA tag linting
  - Yongrui Lin - different parsing that controller-tools w/ chaining
    - Gengo parse chaining
    - Controller-tools is different
    - Kube-apilinter similar to controller-tools
    - Joe Betz - for DV we have a tag w/ a consistent style
      - Not mixing w/ other tags and structure
      - Maybe extend to kubebuilder?
      - In gengo we have a parser for this
      - Lalit Chauhan - use gengo to parse chained tag (new concept). Syntax understood by gengo mainly atm. Payload application. Existing one is key value pair
      - Bryce - parsing logic in gengo, we should be able to implement tag parsing in this case
        - Look at suffix, we can do DV tag parsing
          - If not: controller-tools, kubebuilder tags
      - Joe Betz - let us know how that impl goes, we have namespacing to make a first pass attempt this
      - joel.speed@hotmail.co.uk - chaining
        - **eachKey (delta)**
        - eachVal
        - **ifEnabled|Disabled (some diffs)**
        - +k8s:subfield
        - +k8s:item

- CEL
- 
- Bryce - wishlist items around CRD type linting rules
  - Go through wishlist and see what could be done for both
  - Get linters in place, align DV work
- Where to chat async after this meeting
  - Slack
  - SIG API Machinery dev tools
- Abdul - docs, how to get started here
  - Chat async around work here in sig api machinery dev tools
- ASYNC CHAT LOCATION
  - [# sig-api-machinery-dev-tools Slack Channel](#)
- Als
  - Go through linting wishlist
    - Go into k/k
    - Depend on DV for k/k
    - upcoming
  - How to contribute to DV
  - ifEnabled format
  - Parsing DV tags