GoLand 开发项目 (`kubebuilder_3.1.0`)

AUTHOR: 彭玲 TIME: 2021/10/28

创建项目

通过 `kubebuilder_3.1.0` 创建 `kbexample` 项目，GoLand 打开项目后会根据 `go.mod` 文件自动下载并安装项目依赖的包。

```
kubebuilder init --domain my.domain --repo my.domain/kbexample
kubebuilder create api --group webapp --version v1 --kind Kbexample
```

`go.mod` 文件如下：

```
module my.domain/kbexample

go 1.16

require (
    github.com/onsi/ginkgo v1.14.1
    github.com/onsi/gomega v1.10.2
    k8s.io/apimachinery v0.20.2
    k8s.io/client-go v0.20.2
    sigs.k8s.io/controller-runtime v0.8.3
)

```

代码修改

1. 增加业务逻辑

修改 CRD 的数据结构，并在 controller 中增加一些日志输出。

修改 CRD

修改使用 kubebuilder 命令生成的默认 CRD 配置，在 CRD 中增加 `FirstName`、`LastName` 和 `Status` 字段。

修改后的代码比原先使用 kubebuilder 生成的默认代码增加了以下内容：

```
FirstName string `json:"firstname"`
LastName  string `json:"lastname"`
Status string `json:"Status"`
```

下面是修改后的 `api/v1/guestbook_types.go` 文件的内容：

```
// KbexampleSpec defines the desired state of Kbexample
type KbexampleSpec struct {
    // INSERT ADDITIONAL SPEC FIELDS - desired state of cluster
    // Important: Run "make" to regenerate code after modifying this file

    // Foo is an example field of Kbexample. Edit kbexample_types.go to
remove/update
    Foo string `json:"foo,omitempty"`

    // 添加两个新的字段
    FirstName string `json:"firstname"`
    LastName  string `json:"lastname"`
}

// KbexampleStatus defines the observed state of Kbexample
type KbexampleStatus struct {
    // INSERT ADDITIONAL STATUS FIELD - define observed state of cluster
    // Important: Run "make" to regenerate code after modifying this file

    // 添加 Status 字段
    Status string `json:"Status"`
}
```

修改 Reconcile()

Reconcile 函数是`控制器`的核心逻辑，其业务逻辑都位于`controllers/guestbook_controller.go`文件的`Reconcile()`函数中。

```go
// Reconcile is part of the main kubernetes reconciliation loop which aims to
// move the current state of the cluster closer to the desired state.
// TODO(user): Modify the Reconcile function to compare the state specified by
// the Kbexample object against the actual cluster state, and then
// perform operations to make the cluster state reflect the state specified by
// the user.
//
// For more details, check Reconcile and its Result here:
// - https://pkg.go.dev/sigs.k8s.io/controller-runtime@v0.8.3/pkg/reconcile
func (r *KbexampleReconciler) Reconcile(ctx context.Context, req ctrl.Request) (ctrl.Result, error) {
    _ = log.FromContext(ctx)

    // your logic here

    // 获取当前的 CR， 并打印
    obj := &webappv1.Kbexample{}
    if err := r.Get(ctx, req.NamespacedName, obj); err != nil {
        _log.Println(err, "Unable to fetch object")
    } else {
        _log.Println("Geeting from Kubebuilder to", obj.Spec.FirstName, obj.Spec.LastName)
    }

    // 初始化 CR 的 Status 为 Running
    obj.Status.Status = "Running"
    if err := r.Status().Update(ctx, obj); err != nil {
        _log.Println(err, "unable to update status")
    }

    return ctrl.Result{}, nil
}
```

修改 CR

修改`config/samples/webapp_v1_guestbook.yaml`文件中的配置。

```yaml
apiVersion: webapp.my.domain/v1
kind: Kbexample
metadata:
  name: kbexample-sample
spec:
  # Add fields here
  foo: bar
  firstname: julin # 新增 firstname 和 lastname 两个字段
  lastname: peng
```

2. 运行测试

部署 CRD

```
anxin@node38:~/pengling/k8s/kubebuilder/kbexample$ make install
...
/home/anxin/pengling/k8s/kubebuilder/kbexample/bin/kustomize build config/crd | kubectl apply -f -
customresourcedefinition.apiextensions.k8s.io/kbexamples.webapp.my.domain created

# 查看 api 资源
anxin@node38:~/pengling/k8s/kubebuilder/kbexample$ kubectl api-resources
NAME              SHORTNAMES   APIGROUP               NAMESPACED   KIND
...
kbexamples                     webapp.my.domain       true         Kbexample
```

运行 Controller

```
anxin@node38:~/pengling/k8s/kubebuilder/kbexample$ make run
...
go run ./main.go
I1026 10:26:57.260714   19276 request.go:655] Throttling request took 1.030182502s, request: GET:https://10.8.30.38:6443/apis/tenant.kubesphere.io/v1alpha2?timeout=32s
2021-10-26T10:26:57.641+0800    INFO    controller-runtime.metrics  metrics server is starting to listen    {"addr": ":8080"}
2021-10-26T10:26:57.642+0800    INFO    setup   starting manager
2021-10-26T10:26:57.642+0800    INFO    controller-runtime.manager  starting metrics server {"path": "/metrics"}
2021-10-26T10:26:57.642+0800    INFO    controller-runtime.manager.controller.kbexample Starting EventSource    {"reconciler group": "webapp.my.domain", "reconciler kind": "Kbexample", "source": "kind source: /, Kind="}
2021-10-26T10:26:57.743+0800    INFO    controller-runtime.manager.controller.kbexample Starting Controller {"reconciler group": "webapp.my.domain", "reconciler kind": "Kbexample"}
2021-10-26T10:26:57.743+0800    INFO    controller-runtime.manager.controller.kbexample Starting workers    {"reconciler group": "webapp.my.domain", "reconciler kind": "Kbexample", "worker count": 1}
2021/10/26 10:29:45 Geeting from Kubebuilder to julin peng # 从日志中可以看到这条输出，这正是在 Reconcile 函数中的输出。
```

部署 CR

```
anxin@node38:~/pengling/k8s/kubebuilder/kbexample$ kubectl apply -f config/samples
kbexample.webapp.my.domain/kbexample-sample created
```

3. 获取 CR

输出的最后部分，正是我们在 CRD 里定义的字段。

```
spec:
  firstname: julin
  foo: bar
  lastname: peng
status:
  Status: Running
```

使用下面的命令获取当前的 CR。

```
anxin@node38:~$ kubectl get kbexamples.webapp.my.domain kbexample-sample -o yaml
apiVersion: webapp.my.domain/v1
kind: Kbexample
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"webapp.my.domain/v1","kind":"Kbexample","metadata":{"annotations":{},"name":"kbexample-sample","namespace":"default"},"spec":{"firstname":"julin","foo":"bar","lastname":"peng"}}
  creationTimestamp: "2021-10-26T02:29:27Z"
  generation: 1
  managedFields:
  - apiVersion: webapp.my.domain/v1
    fieldsType: FieldsV1
    fieldsV1:
      f:metadata:
        f:annotations:
          .: {}
          f:kubectl.kubernetes.io/last-applied-configuration: {}
      f:spec:
        .: {}
        f:firstname: {}
        f:foo: {}
        f:lastname: {}
    manager: kubectl
    operation: Update
    time: "2021-10-26T02:29:27Z"
  - apiVersion: webapp.my.domain/v1
    fieldsType: FieldsV1
    fieldsV1:
      f:status:
        .: {}
        f:Status: {}
    manager: main
    operation: Update
    time: "2021-10-26T02:29:45Z"
  name: kbexample-sample
  namespace: default
  resourceVersion: "134203417"
  selfLink: /apis/webapp.my.domain/v1/namespaces/default/kbexamples/kbexample-sample
  uid: 55115d51-039e-4b45-add0-d51b64144a2d
```

```
spec:
  firstname: julin
  foo: bar
  lastname: peng
status:
  Status: Running
```

`kubebuilder init`生成的文件

执行`kubebuilder init`初始化一个项目，会在本地生成一些文件。比如：`controller-gen`可执行文件、`kubebuilder 项目`下的`bin/manager`可执行文件、`样板文件`等。

`~/go/bin/controller-gen`可执行文件

`kubebuilder init`创建项目的过程中(`go get: added sigs.k8s.io/controller-tools v0.2.5`之后)会在本地生成`controller-gen`可执行文件。

```
pengling@FSZJ-PENGLING:~$ ll /home/pengling/go/bin
total 18268
drwxrwxr-x 2 pengling pengling     4096 Sep 10 10:04    ./
drwxrwxr-x 4 pengling pengling     4096 Sep 10 10:04    ../
-rwxrwxr-x 1 pengling pengling  18694569 Sep 13 10:27    controller-gen*
```

`<kbproject>/bin/manager`可执行文件

`kubebuilder init`创建项目的过程中`go build -o bin/manager main.go`的结果：

```
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ ll bin
total 37456
drwxrwxr-x 2 pengling pengling     4096 Sep 13 09:09    ./
drwxrwxr-x 7 pengling pengling     4096 Sep 13 10:28    ../
-rwxrwxr-x 1 pengling pengling  38346014 Sep 13 09:09    manager*
```

`boilerplate.go.txt`(样板文件)

该文件作为`.go`源码的`文件注释头`。

```
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ cat hack/boilerplate.go.txt
/*


Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

```
16 */
```

`kubebuilder create api` 文件变化

项目差异比较

执行 `kubebuilder create api` 创建 API 之后，除了新增了 `api` 和 `controllers` 两个目录，项目文件也有变化。使用 `diff` 工具进行差异比较。

- `guestbook-init` 目录，为项目初始化后的结果。
- `guestbook` 目录，为创建 API 后的结果。

```
1  pengling@FSZJ-PENGLING:~/myprojects$ diff -qr guestbook-init guestbook
2  Files guestbook-init/PROJECT and guestbook/PROJECT differ # 3. PROJECT 文件有
   差异
3  Only in guestbook: api # 1. 新增 api 目录
4  Files guestbook-init/bin/manager and guestbook/bin/manager differ # 5.
   bin/manager 文件有差异
5  Only in guestbook/config: crd # 6. config 目录有差异 -- 新增 crd 子目录
6  Files guestbook-init/config/default/kustomization.yaml and
   guestbook/config/default/kustomization.yaml differ # 6. config/default 子目录
   有差异
7  Only in guestbook/config/rbac: guestbook_editor_role.yaml # 6. config/rbac 子
   目录有差异
8  Only in guestbook/config/rbac: guestbook_viewer_role.yaml # 6. config/rbac 子
   目录有差异
9  Only in guestbook/config: samples # 6. config 目录有差异 -- 新增 samples 子目录
10 Only in guestbook: controllers # 2. 新增 controllers 目录
11 Files guestbook-init/main.go and guestbook/main.go differ # 4. main.go 文件有
   差异
```

查看 `bin` 子目录差别：

```
1  pengling@FSZJ-PENGLING:~/myprojects$ diff -q guestbook-init/bin guestbook/bin
2  Files guestbook-init/bin/manager and guestbook/bin/manager differ # manager 文
   件有差别
```

查看 `hack` 子目录差别：

```
1  pengling@FSZJ-PENGLING:~/myprojects$ diff -q guestbook-init/hack
   guestbook/hack
2  pengling@FSZJ-PENGLING:~/myprojects$ # 无差别
```

1. added files

`api/v1/guestbook_types.go`

该文件由 `kubebuilder create api` 定义一个资源的过程中，在 `Create Resource [y/n]` 交互步骤选择 `y` 生成。

```
1  /* http://www.apache.org/licenses/LICENSE-2.0 */
2
```

```go
package v1

import (
    metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
)

// EDIT THIS FILE!  THIS IS SCAFFOLDING FOR YOU TO OWN!
// NOTE: json tags are required.  Any new fields you add must have json tags for the fields to be serialized.

// GuestbookSpec defines the desired state of Guestbook
type GuestbookSpec struct {
    // INSERT ADDITIONAL SPEC FIELDS - desired state of cluster
    // Important: Run "make" to regenerate code after modifying this file

    // Foo is an example field of Guestbook. Edit Guestbook_types.go to remove/update
    Foo string `json:"foo,omitempty"`
}

// GuestbookStatus defines the observed state of Guestbook
type GuestbookStatus struct {
    // INSERT ADDITIONAL STATUS FIELD - define observed state of cluster
    // Important: Run "make" to regenerate code after modifying this file
}

// +kubebuilder:object:root=true

// Guestbook is the Schema for the guestbooks API
type Guestbook struct {
    metav1.TypeMeta   `json:",inline"`
    metav1.ObjectMeta `json:"metadata,omitempty"`

    Spec   GuestbookSpec   `json:"spec,omitempty"`
    Status GuestbookStatus `json:"status,omitempty"`
}

// +kubebuilder:object:root=true

// GuestbookList contains a list of Guestbook
type GuestbookList struct {
    metav1.TypeMeta `json:",inline"`
    metav1.ListMeta `json:"metadata,omitempty"`
    Items           []Guestbook `json:"items"`
}

func init() {
    SchemeBuilder.Register(&Guestbook{}, &GuestbookList{})
}
```

`controller/guestbook_controller.go`

该文件由 `kubebuilder create api` 定义一个资源的过程中，在 `Create Controller [y/n]` 交互步骤选择 `y` 生成。

```go
/* http://www.apache.org/licenses/LICENSE-2.0 */

```

```go
package controllers

import (
    "context"

    "github.com/go-logr/logr"
    "k8s.io/apimachinery/pkg/runtime"
    ctrl "sigs.k8s.io/controller-runtime"
    "sigs.k8s.io/controller-runtime/pkg/client"

    webappv1 "my.domain/guestbook/api/v1"
)

// GuestbookReconciler reconciles a Guestbook object
type GuestbookReconciler struct {
    client.Client
    Log    logr.Logger
    Scheme *runtime.Scheme
}

// +kubebuilder:rbac:groups=webapp.my.domain,resources=guestbooks,verbs=get;list;watch;create;update;patch;delete
// +kubebuilder:rbac:groups=webapp.my.domain,resources=guestbooks/status,verbs=get;update;patch

func (r *GuestbookReconciler) Reconcile(req ctrl.Request) (ctrl.Result, error) {
    _ = context.Background()
    _ = r.Log.WithValues("guestbook", req.NamespacedName)

    // your logic here

    return ctrl.Result{}, nil
}

func (r *GuestbookReconciler) SetupWithManager(mgr ctrl.Manager) error {
    return ctrl.NewControllerManagedBy(mgr).
        For(&webappv1.Guestbook{}).
        Complete(r)
}
```

`config/rbac/` 新增文件

`config/rbac/` 子目录中新增了 2 个文件：`guestbook_editor_role.yaml` 和 `guestbook_viewer_role.yaml`。

guestbook_editor_role.yaml

```yaml
# permissions for end users to edit guestbooks.
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: guestbook-editor-role
rules:
- apiGroups:
```

```
  - webapp.my.domain
  resources:
  - guestbooks
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - webapp.my.domain
  resources:
  - guestbooks/status
  verbs:
  - get
```

guestbook_viewer_role.yaml

```
# permissions for end users to view guestbooks.
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: guestbook-viewer-role
rules:
- apiGroups:
  - webapp.my.domain
  resources:
  - guestbooks
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - webapp.my.domain
  resources:
  - guestbooks/status
  verbs:
  - get
```

2. added directories

`config` 目录下，新增了 2 个文件夹：`crd/` 和 `samples/`。

`config/crd/`

该目录用来部署 CRD 用。

```
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ tree config/crd
config/crd
├── kustomization.yaml
├── kustomizeconfig.yaml
└── patches
    ├── cainjection_in_guestbooks.yaml
    └── webhook_in_guestbooks.yaml

1 directory, 4 files
```

`config/samples/`

该目录给出了 CR 对象示例。

```
# alias lll='ls -lAF'
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ lll config/samples/
total 4
-rw------- 1 pengling pengling 120 Sep 13 16:32 webapp_v1_guestbook.yaml
```

3. changed files

`main.go` 文件差异

```
/* http://www.apache.org/licenses/LICENSE-2.0 */

package main

import (
    "flag"
    "os"

    "k8s.io/apimachinery/pkg/runtime"
    clientgoscheme "k8s.io/client-go/kubernetes/scheme"
    _ "k8s.io/client-go/plugin/pkg/client/auth/gcp"
    ctrl "sigs.k8s.io/controller-runtime"
    "sigs.k8s.io/controller-runtime/pkg/log/zap"

   webappv1 "my.domain/guestbook/api/v1" // @JULIN: (+) 新增代码
    "my.domain/guestbook/controllers" // @JULIN: (+) 新增代码
    // +kubebuilder:scaffold:imports
)

var (
    scheme   = runtime.NewScheme()
    setupLog = ctrl.Log.WithName("setup")
)

func init() {
    _ = clientgoscheme.AddToScheme(scheme)

    _ = webappv1.AddToScheme(scheme) // @JULIN: (+) 新增代码
    // +kubebuilder:scaffold:scheme
}

func main() {
```

```
    var metricsAddr string
    var enableLeaderElection bool
    flag.StringVar(&metricsAddr, "metrics-addr", ":8080", "The address the metric endpoint binds to.")
    flag.BoolVar(&enableLeaderElection, "enable-leader-election", false,
        "Enable leader election for controller manager. "+
            "Enabling this will ensure there is only one active controller manager.")
    flag.Parse()

    ctrl.SetLogger(zap.New(zap.UseDevMode(true)))

    mgr, err := ctrl.NewManager(ctrl.GetConfigOrDie(), ctrl.Options{
        Scheme:             scheme,
        MetricsBindAddress: metricsAddr,
        Port:               9443,
        LeaderElection:     enableLeaderElection,
        LeaderElectionID:   "ecaf1259.my.domain",
    })
    if err != nil {
        setupLog.Error(err, "unable to start manager")
        os.Exit(1)
    }

   // @JULIN: (+) 新增代码
    if err = (&controllers.GuestbookReconciler{
        Client: mgr.GetClient(),
        Log:    ctrl.Log.WithName("controllers").WithName("Guestbook"),
        Scheme: mgr.GetScheme(),
    }).SetupWithManager(mgr); err != nil {
        setupLog.Error(err, "unable to create controller", "controller", "Guestbook")
        os.Exit(1)
    }
    // +kubebuilder:scaffold:builder

    setupLog.Info("starting manager")
    if err := mgr.Start(ctrl.SetupSignalHandler()); err != nil {
        setupLog.Error(err, "problem running manager")
        os.Exit(1)
    }
}
```

PROJECT 文件

该文件记录了项目的基本信息。

```
domain: my.domain
repo: my.domain/guestbook
resources:
- group: webapp
  kind: Guestbook
  version: v1
version: "2"
```

`bin/` manager 可执行文件

```
# guestbook-init/bin/manager 文件大小：38345998 字节
pengling@FSZJ-PENGLING:~/myprojects$ ls guestbook-init/bin -lAF
total 37448
-rwxrwxr-x 1 pengling pengling 38345998 Sep 13 16:23 manager*
# guestbook/bin/manager 文件大小：38346014 字节
pengling@FSZJ-PENGLING:~/myprojects$ ls guestbook/bin -lAF
total 37448
-rwxrwxr-x 1 pengling pengling 38346014 Sep 13 16:30 manager*
```

`config/default/` kustomization.yaml

该文件为 kustomize 重要的配置文件。

```
# Adds namespace to all resources.
namespace: guestbook-system

# Value of this field is prepended to the
# names of all resources, e.g. a deployment named
# "wordpress" becomes "alices-wordpress".
# Note that it should also match with the prefix (text before '-') of the namespace
# field above.
namePrefix: guestbook-

# Labels to add to all resources and selectors.
#commonLabels:
#  someName: someValue

bases:
- ../crd
- ../rbac
- ../manager
# [WEBHOOK] To enable webhook, uncomment all the sections with [WEBHOOK] prefix including the one in
# crd/kustomization.yaml
#- ../webhook
# [CERTMANAGER] To enable cert-manager, uncomment all sections with 'CERTMANAGER'. 'WEBHOOK' components are required.
#- ../certmanager
# [PROMETHEUS] To enable prometheus monitor, uncomment all sections with 'PROMETHEUS'.
#- ../prometheus

patchesStrategicMerge:
  # Protect the /metrics endpoint by putting it behind auth.
  # If you want your controller-manager to expose the /metrics
  # endpoint w/o any authn/z, please comment the following line.
- manager_auth_proxy_patch.yaml

# [WEBHOOK] To enable webhook, uncomment all the sections with [WEBHOOK] prefix including the one in
# crd/kustomization.yaml
#- manager_webhook_patch.yaml

```

```
# [CERTMANAGER] To enable cert-manager, uncomment all sections with 'CERTMANAGER'.
# Uncomment 'CERTMANAGER' sections in crd/kustomization.yaml to enable the CA injection in the admission webhooks.
# 'CERTMANAGER' needs to be enabled to use ca injection
#- webhookcainjection_patch.yaml

# the following config is for teaching kustomize how to do var substitution
vars:
# [CERTMANAGER] To enable cert-manager, uncomment all sections with 'CERTMANAGER' prefix.
#- name: CERTIFICATE_NAMESPACE # namespace of the certificate CR
#  objref:
#    kind: Certificate
#    group: cert-manager.io
#    version: v1alpha2
#    name: serving-cert # this name should match the one in certificate.yaml
#  fieldref:
#    fieldpath: metadata.namespace
#- name: CERTIFICATE_NAME
#  objref:
#    kind: Certificate
#    group: cert-manager.io
#    version: v1alpha2
#    name: serving-cert # this name should match the one in certificate.yaml
#- name: SERVICE_NAMESPACE # namespace of the service
#  objref:
#    kind: Service
#    version: v1
#    name: webhook-service
#  fieldref:
#    fieldpath: metadata.namespace
#- name: SERVICE_NAME
#  objref:
#    kind: Service
#    version: v1
#    name: webhook-service
```

`kustomization.yaml` 差异比较

```
pengling@FSZJ-PENGLING:~/myprojects$ diff guestbook-init/config/default/kustomization.yaml guestbook/config/default/kustomization.yaml
2c2
< namespace: guestbook-init-system
---
> namespace: guestbook-system
9c9
< namePrefix: guestbook-init-
---
> namePrefix: guestbook-
```