kubebuilder 项目创建

AUTHOR: 彭玲 TIME: 2021/9/29

创建一个项目

kubebuilder(`v2.3.1`) 安装成功后，创建 kubebuilder 项目`guestbook`：

```
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ kubebuilder init --domain my.domain --repo my.domain/guestbook
Writing scaffold for you to edit...
Get controller runtime:
$ go get sigs.k8s.io/controller-runtime@v0.5.0
Update go.mod:
$ go mod tidy
Running make:
$ make
go: creating new go.mod: module tmp
go get: added sigs.k8s.io/controller-tools v0.2.5
/home/pengling/go/bin/controller-gen object:headerFile="hack/boilerplate.go.txt" paths="./..."
go fmt ./...
go vet ./...
go build -o bin/manager main.go
Next: define a resource with:
$ kubebuilder create api
```

`guestbook`项目目录

```
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ ll
total 88
drwxrwxr-x 5 pengling pengling  4096    Sep 13 16:30    ./
drwxrwxr-x 5 pengling pengling  4096    Sep 13 16:30    ../
-rw------- 1 pengling pengling   357    Sep 13 16:30    .gitignore
-rw------- 1 pengling pengling   795    Sep 13 16:30    Dockerfile
-rw------- 1 pengling pengling  2069    Sep 13 16:30    Makefile
-rw------- 1 pengling pengling   57     Sep 13 16:30    PROJECT
drwxrwxr-x 2 pengling pengling  4096    Sep 13 16:30    bin/
drwx------ 8 pengling pengling  4096    Sep 13 16:30    config/
```

```
-rw------- 1 pengling pengling   143    Sep 13 16:30    go.mod
-rw-rw-r-- 1 pengling pengling 44135     Sep 13 16:30    go.sum
drwx------ 2 pengling pengling  4096    Sep 13 16:30    hack/
-rw------- 1 pengling pengling  1966    Sep 13 16:30    main.go
```

`boilerplate.go.txt`文件

```
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.
*/
```

创建一个 API

定义一个资源

运行下面的命令，创建一个新的 API (`组/版本`) 为 “webapp/v1”，并在上面创建新的 Kind (CRD) “Guestbook”。

如果在`Create Resource [y/n]`和`Create Controller [y/n]`中按`y`，将会创建两个文件：

- `api/v1/guestbook_types.go`文件：该文件中定义相关 API 。
- `controller/guestbook_controller.go`文件：实现`Guestbook`这一 Kind (CRD) 的`reconciliation`(对账) 业务逻辑。

```
pengling@FSZJ-PENGLING:~/myprojects/guestbook$ kubebuilder create api --group webapp --version v1 --kind Guestbook
Create Resource [y/n]
y
Create Controller [y/n]
y
Writing scaffold for you to edit...
api/v1/guestbook_types.go
controllers/guestbook_controller.go
Running make:
$ make
go: creating new go.mod: module tmp
go get: added sigs.k8s.io/controller-tools v0.2.5
/home/pengling/go/bin/controller-gen object:headerFile="hack/boilerplate.go.txt" paths="./..."
```

```
Error: go [list -e -json -compiled=true -test=false -export=false -deps=true -find=false -tags ignore_autogenerated -- ./...]: exit status 1: go: my.domain/guestbook/controllers: package github.com/go-logr/logr imported from implicitly required module; to add missing requirements, run:
    go get github.com/go-logr/logr@v0.1.0

Usage:
  controller-gen [flags]

Examples:
    # Generate RBAC manifests and crds for all types under apis/,
    # outputting crds to /tmp/crds and everything else to stdout
    controller-gen rbac:roleName=<role name> crd paths=./apis/... output:crd:dir=/tmp/crds output:stdout

    # Generate deepcopy/runtime.Object implementations for a particular file
    controller-gen object paths=./apis/v1beta1/some_types.go

    # Generate OpenAPI v3 schemas for API packages and merge them into existing CRD manifests
    controller-gen schemapatch:manifests=./manifests output:dir=./manifests paths=./pkg/apis/...

    # Run all the generators for a given project
    controller-gen paths=./apis/...

    # Explain the markers for generating CRDs, and their arguments
    controller-gen crd -ww


Flags:
  -h, --detailed-help count   print out more detailed help
                              (up to -hhh for the most detailed output, or -hhhh for json output)
      --help                  print out usage and a summary of options
      --version               show version
  -w, --which-markers count   print out all markers available with the requested generators
                              (up to -www for the most detailed output, or -wwww for json output)


Options


generators

+webhook
                             package  generates (partial) {Mutating,Validating}WebhookConfiguration objects.
+schemapatch:manifests=<string>[,maxDescLen=<int>]
                             package  patches existing CRDs with new schemata.
+rbac:roleName=<string>
                              package  generates ClusterRole objects.
```

```
+object[:headerFile=<string>][,year=<string>]                              package  generates code containing DeepCopy, DeepCopyInto, and DeepCopyObject method implementations.
+crd[:crdVersions=<[]string>][,maxDescLen=<int>][,preserveUnknownFields=<bool>][,trivialVersions=<bool>]  package  generates CustomResourceDefinition objects.



generic

+paths=<[]string>  package  represents paths and go-style path patterns to use as package roots.


output rules (optionally as output:<generator>:...)

+output:artifacts[:code=<string>],config=<string>  package  outputs artifacts to different locations, depending on whether they're package-associated or not.
+output:dir=<string>                               package  outputs each artifact to the given directory, regardless of if it's package-associated or not.
+output:none                                       package  skips outputting anything.
+output:stdout                                     package  outputs everything to standard-out, with no separation.

run `controller-gen object:headerFile=hack/boilerplate.go.txt paths=./... -w` to see all available markers, or `controller-gen object:headerFile=hack/boilerplate.go.txt paths=./... -h` for usage
make: *** [Makefile:55: generate] Error 1
2021/09/13 09:12:59 failed to create API: exit status 2
```

项目变化

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。

- `guestbook-init`目录：`kubebuilder init --domain my.domain --repo my.domain/guestbook`初始化后的结果。
- `guestbook`目录：`kubebuilder create api --group webapp --version v1 --kind Guestbook`创建 API 后的结果。

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

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

查看`hack`子目录差别：

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