Service Binding feature comparison

	Service Binding Spec	**Service Binding Operator**
Service to application binding	One-to-one	One-to-many

Environment Variables	Field **spec.env**
	
	Sets which keys from a provisioned service secret should be added to deployment as environment variables
	`env:`
	`- name: ACCOUNT_SERVICE_HOST`
	`  key: host`
	`- name: ACCOUNT_SERVICE_USERNAME`
	`  key: username`
	`- name: ACCOUNT_SERVICE_PASSWORD`
	`  key: password`
	`- name: ACCOUNT_SERVICE_URI`
	`  key: accountServiceUri`
	
	The name is variable name inside a pod, key is a field to map to inside the binding secret
		Environment variables are added always to workload from the secret
		Additionally has **spec.envVarPrefix** setting to add to each environment variable
		
		Can generate environment variables from service workload or CR using **spec.CustomEnvVar**
		`- name: DB_PASSWORD`
		`  value: '{{ .postgresDB.status.dbCredentials.password }}'`

Variable Mapping	spec.mappings (Go templates)
	The executed output of the template MUST be added to the Secret exposed to the resource represented by application as the key specified by the name of the Mapping
	`mappings:`
	`- name: accountServiceUri`
	`  value: https://{{ urlquery .username }}:{{ urlquery .password }}@{{ .host }}:{{ .port }}`
		N/A

Application Selector	Label Selector or a selector that has apiVersion (group + version), Kind and a name
	
	`application: # ObjectReference-like`
	`apiVersion: # string`
	`kind: # string`
	`name: # string, mutually exclusive with selector`
	`selector: # metav1.LabelSelector, mutually exclusive with name`
	`containers: # []intstr.IntOrString, optional`
	
	Example;
	`application:`
	`apiVersion: apps/v1`
	`kind: Deployment`
	`name: online-banking`
	
	Can specify a container to be mounted to using **containers** array
		Label Selector or a selector that has resourceRef (name), group, version and resource (kind)
		
		Example:
		`applicationSelector:`
		`resourceRef: node-todo-git`
		`group: apps`
		`version: v1`
		`resource: deployments`
		
		bindingPath object can be used to customize where in application workload schema the binging will be performed. Can be used specify path to Container spec within workload

Service Selector	Single service selector
	`service:`
	`apiVersion: com.example/v1alpha1`
	`kind: AccountService`
	`name: prod-account-service`
	
	Selector has apiVersion (group + version), kind and a name
		Array of service selectors:
		`backingServiceSelectors:`
		`- group: postgresql.baiju.dev`
		`version: v1alpha1`
		`kind: Database`
		`resourceRef: db-demo`
		`- group: etcd.database.coreos.com`
		`version: v1beta2`
		`kind: EtcdCluster`
		`resourceRef: etcd-demo`
		
		Each selector has name (resourceRef) , group, version and kind
		
		Allows binding services across namespaces using namespace field

Provisioned Service Secret	Secret must be volume mounted	Adds binding information as environment variables by default, Secret could be volume mounted through use of annotations or x-descriptor

Volume Mounting	Must be mounted at $SERVICE_BINDING_ROOT/<binding-name>/	**spec.volumeMountPrefix** is used to add a prefix to a volume mount location path

Spec Extensions:

Custom application projection:

When **projection.service.binding/type** annotations set to **Custom**
The projection of secret into the application is not handles by service binding implementation but is handled by the consumer through ServiceBindingProjection CRD