

ViewSet

Django REST framework allows you to combine the logic for a set of related views in a single class, called a ViewSet.

There are two main advantages of using a ViewSet over using a View class.

- Repeated logic can be combined into a single class.
- By using routers, we no longer need to deal with wiring up the URL conf ourselves.

ViewSet Class

A ViewSet class is simply a type of class-based View, that does not provide any method handlers such as `get()` or `post()`, and instead provides actions such as `list()` and `create()`.

- `list()` – Get All Records.
- `retrieve()` – Get Single Record
- `create()` – Create/Insert Record
- `update()` – Update Record Completely
- `partial_update()` – Update Record Partially
- `destroy()` – Delete Record

ViewSet Class

```
from rest_framework import viewsets
class StudentViewSet(viewsets.ViewSet):
    def list(self, request): .....
    def create(self, request): .....
    def retrieve(self, request, pk=None): .....
    def update(self, request, pk=None): .....
    def partial_update(self, request, pk=None): .....
    def destroy(self, request, pk=None): .....
```

ViewSet Class

During dispatch, the following attributes are available on the ViewSet:-

- `basename` - the base to use for the URL names that are created.
- `action` - the name of the current action (e.g., `list`, `create`).
- `detail` - boolean indicating if the current action is configured for a list or detail view.
- `suffix` - the display suffix for the viewset type - mirrors the `detail` attribute.
- `name` - the display name for the viewset. This argument is mutually exclusive to `suffix`.
- `description` - the display description for the individual view of a viewset.

ViewSet – URL Config


```
from django.urls import path, include
```

```
from api import views
```

```
from rest_framework.routers import DefaultRouter
```

```
router = DefaultRouter()
```

Creating Default Router Object




```
router.register('studentapi', views.StudentViewSet, basename='student')
```

Register StudentViewSet with Router



```
urlpatterns = [  
    path("", include(router.urls)),  
]
```

The API URLs are now determined automatically by the router.



ModelViewSet Class

The ModelViewSet class inherits from GenericAPIView and includes implementations for various actions, by mixing in the behavior of the various mixin classes.

The actions provided by the ModelViewSet class are list(), retrieve(), create(), update(), partial_update(), and destroy(). You can use any of the standard attributes or method overrides provided by GenericAPIView

```
class StudentModelViewSet(viewsets.ModelViewSet):
```

```
    queryset = Student.objects.all()
```

```
    serializer_class = StudentSerializer
```

ReadOnlyModelViewSet Class

The ReadOnlyModelViewSet class also inherits from GenericAPIView. As with ModelViewSet it also includes implementations for various actions, but unlike ModelViewSet only provides the 'read-only' actions, list() and retrieve(). You can use any of the standard attributes and method overrides available to GenericAPIView

```
class StudentReadOnlyModelViewSet(viewsets.ReadOnlyModelViewSet):  
    queryset = Student.objects.all()  
    serializer_class = StudentSerializer
```