🔹 Toutes les Propriétés et Méthodes des Class-Based Views (CBV) dans Django REST Framework (DRF)

Voici un récapitulatif des **différentes méthodes** qu'on peut définir dans les vues basées sur les classes en fonction de chaque type de vue.

1️⃣ APIView (Base des CBV dans DRF)

Classe de base qui offre une structure pour manipuler les requêtes HTTP.

🔹 Méthodes principales

Méthode	Description
`get(self, request, *args, **kwargs)`	Gère une requête GET
`post(self, request, *args, **kwargs)`	Gère une requête POST
`put(self, request, *args, **kwargs)`	Gère une requête PUT
`patch(self, request, *args, **kwargs)`	Gère une requête PATCH
`delete(self, request, *args, **kwargs)`	Gère une requête DELETE

🔹 Méthodes utilitaires

Méthode	Description
`dispatch(self, request, *args, **kwargs)`	Redirige vers la bonne méthode selon HTTP method
`get_authenticators(self)`	Retourne les classes d'authentification
`get_permissions(self)`	Retourne les classes de permissions
`get_parsers(self)`	Retourne les classes de parsing
`get_renderers(self)`	Retourne les classes de rendu (JSON, XML, etc.)
`handle_exception(self, exc)`	Gère les erreurs et exceptions

💡 **Exemple :**

```
from rest_framework.views import APIView
from rest_framework.response import Response

class ExampleView(APIView):
```

```
    def get(self, request):
        return Response({"message": "GET request successful"})

    def post(self, request):
        return Response({"message": "POST request successful"})
```

2️⃣ GenericAPIView (Base des vues génériques)

Hérite de `APIView` et ajoute des fonctionnalités pour les modèles Django.

🔹 Méthodes spécifiques

Méthode	Description
`get_queryset(self)`	Retourne l'ensemble des objets manipulés
`get_serializer_class(self)`	Retourne le serializer utilisé
`get_serializer(self, *args, **kwargs)`	Instancie un serializer
`get_object(self)`	Retourne un objet spécifique en fonction de `lookup_field`

💡 **Exemple :**

```
from rest_framework.generics import GenericAPIView
from rest_framework.response import Response
from myapp.models import Snippet
from myapp.serializers import SnippetSerializer

class SnippetDetail(GenericAPIView):
    queryset = Snippet.objects.all()
    serializer_class = SnippetSerializer

    def get(self, request, pk):
        snippet = self.get_object()
        serializer = self.get_serializer(snippet)
        return Response(serializer.data)
```

3️⃣ Mixins (Ajout de comportements spécifiques)

Les **mixins** permettent d'ajouter des comportements spécifiques à `GenericAPIView`.

🔹 Méthodes disponibles dans chaque mixin

Mixin	Méthodes
`CreateModelMixin`	`create(self, request, *args, **kwargs)`
`ListModelMixin`	`list(self, request, *args, **kwargs)`
`RetrieveModelMixin`	`retrieve(self, request, *args, **kwargs)`
`UpdateModelMixin`	`update(self, request, *args, **kwargs)`, `partial_update(self, request, *args, **kwargs)`
`DestroyModelMixin`	`destroy(self, request, *args, **kwargs)`

💡 **Exemple :**

```
from rest_framework import mixins, generics
from myapp.models import Snippet
from myapp.serializers import SnippetSerializer

class SnippetDetail(mixins.RetrieveModelMixin,
                    mixins.UpdateModelMixin,
                    mixins.DestroyModelMixin,
                    generics.GenericAPIView):
    queryset = Snippet.objects.all()
    serializer_class = SnippetSerializer

    def get(self, request, *args, **kwargs):
        return self.retrieve(request, *args, **kwargs)

    def put(self, request, *args, **kwargs):
        return self.update(request, *args, **kwargs)

    def delete(self, request, *args, **kwargs):
        return self.destroy(request, *args, **kwargs)
```

4️⃣ Vues Génériques (CRUD simplifié)

DRF propose des **vues génériques** qui combinent `GenericAPIView` et les mixins.

🔹 Méthodes intégrées dans chaque vue

Vue générique	Méthodes intégrées
`ListAPIView`	`get()`
`RetrieveAPIView`	`get()`
`CreateAPIView`	`post()`
`UpdateAPIView`	`put()`, `patch()`

Vue générique	Méthodes intégrées
`DestroyAPIView`	`delete()`
`ListCreateAPIView`	`get()`, `post()`
`RetrieveUpdateAPIView`	`get()`, `put()`, `patch()`
`RetrieveDestroyAPIView`	`get()`, `delete()`
`RetrieveUpdateDestroyAPIView`	`get()`, `put()`, `patch()`, `delete()`

💡 **Exemple avec `RetrieveUpdateDestroyAPIView` :**

```
from rest_framework.generics import RetrieveUpdateDestroyAPIView
from myapp.models import Snippet
from myapp.serializers import SnippetSerializer

class SnippetDetail(RetrieveUpdateDestroyAPIView):
    queryset = Snippet.objects.all()
    serializer_class = SnippetSerializer
```

👉 **Les méthodes `get()`, `put()`, `patch()` et `delete()` sont déjà définies !**

5️⃣ ViewSets (Regroupe les opérations CRUD)

Les `ViewSets` permettent de regrouper les opérations CRUD dans une seule classe.

🔹 Méthodes principales

Méthode	Description
`list(self, request, *args, **kwargs)`	Retourne une liste d'objets (GET)
`retrieve(self, request, *args, **kwargs)`	Retourne un objet unique (GET)
`create(self, request, *args, **kwargs)`	Crée un objet (POST)
`update(self, request, *args, **kwargs)`	Met à jour un objet (PUT)
`partial_update(self, request, *args, **kwargs)`	Met à jour partiellement un objet (PATCH)
`destroy(self, request, *args, **kwargs)`	Supprime un objet (DELETE)

🔹 Différents types de ViewSets

Type de ViewSet	Méthodes prises en charge

Type de ViewSet	Méthodes prises en charge
`ViewSet`	Doit définir explicitement `list()`, `retrieve()`, `create()`, `update()`, `destroy()`
`ModelViewSet`	Fournit toutes les méthodes CRUD automatiquement
`ReadOnlyModelViewSet`	Fournit uniquement `list()` et `retrieve()`

💡 **Exemple avec `ModelViewSet` :**

```
from rest_framework.viewsets import ModelViewSet
from myapp.models import Snippet
from myapp.serializers import SnippetSerializer

class SnippetViewSet(ModelViewSet):
    queryset = Snippet.objects.all()
    serializer_class = SnippetSerializer
```

👉 **Les routes suivantes sont gérées automatiquement :**

- `GET /snippets/` → Liste
- `POST /snippets/` → Création
- `GET /snippets/{id}/` → Détail
- `PUT /snippets/{id}/` → Modification
- `DELETE /snippets/{id}/` → Suppression

🎯 Conclusion

Type de Vue	Méthodes principales
`APIView`	`get()`, `post()`, `put()`, `patch()`, `delete()`
`GenericAPIView`	`get_queryset()`, `get_serializer()`, `get_object()`
`Mixins`	`list()`, `retrieve()`, `create()`, `update()`, `partial_update()`, `destroy()`
`Vues Génériques`	Méthodes CRUD intégrées selon le type de vue
`ViewSets`	`list()`, `retrieve()`, `create()`, `update()`, `partial_update()`, `destroy()`

👉 **Plus on va vers `ViewSets`, plus on a d'automatisation et moins on écrit de code ! 🚀**