Python Valencia

Quique Porta

develapps

Desarrollo de apps nativas para IOS y Android

Django REST Framework

...

A powerful and flexible toolkit for building Web APIs
... with deadlines.

REST (REpresentational State Transfer)

Es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP.

http://dominio.com/clientes/

http://dominio.com/clientes/15/

http://dominio.com/clientes/15/facturas/

http://dominio.com/facturas/8/

http://dominio.com/facturas/?from_year=2015

OPERACIONES HTTP

GET
POST
PUT / PATCH
DELETE

CÓDIGOS DE ESTADO

1xx Respuesta informativa

2xx Peticiones correctas

3xx Redirecciones

4xx Errores del cliente

5xx Errores de servidor

HYPERMEDIA

{
“id”: 15,
“nombre”: “Tomas”,
“edad”: 25,
“facturas”: “http://api.com/user/15/facturas/”,
“url”: “http://api.com/user/15/”
}

No guarda estado

Elementos básicos de Django REST Framework

Request

Hereda de HttpRequest de Django

.data - POST o FILES
.query_params - GET
.user - Django User (django.contrib.auth.models.User)
.method - “GET”, “POST”, “PUT” ...

Serializers

Son como los formularios de Django

.is_valid() - Valida los datos pasados
.save() - Crea o actualiza el objeto asociado
ModelSerializer - Igual que un ModelForm
HyperlinkedModelSerializer

```
class EventSerializer(serializers.Serializer):
    description = serializers.CharField(max_length=100)
    start = serializers.DateTimeField()
    finish = serializers.DateTimeField()

    def validate(self, data):
        """
        Check that the start is before the stop.
        """
        if data['start'] > data['finish']:
            raise serializers.ValidationError("finish must occur after start")
        return data
```

```
class AccountSerializer(serializers.ModelSerializer):
    class Meta:
        model = Account
        fields = ('id', 'account_name', 'users', 'created')
        read_only_fields = ('account_name',)
```

Views

Django Rest nos provee de muchos tipos de clases de vistas. (http://www.cdrf.co/)

APIView, ViewSet, ModelViewSet, ListCreateAPIView, RetrieveDestroyAPIView, Mixins, etc..

```python
class UserViewSet(viewsets.ViewSet):
    """
    A simple ViewSet for listing or retrieving users.
    """
    def list(self, request):
        queryset = User.objects.all()
        serializer = UserSerializer(queryset, many=True)
        return Response(serializer.data)

    def retrieve(self, request, pk=None):
        queryset = User.objects.all()
        user = get_object_or_404(queryset, pk=pk)
        serializer = UserSerializer(user)
        return Response(serializer.data)
```

```python
class UserList(generics.ListCreateAPIView):
    queryset = User.objects.all()
    serializer_class = UserSerializer
    permission_classes = (IsAdminUser,)
    paginate_by = 100
```

Response

Permite devolver al cliente una respuesta con los datos y con un estado.

.data - Los datos serializados para la respuesta
.status - Código de estado de la respuesta

Routers

Permite definir las URL’s de acceso a nuestros recursos.

```
router = routers.SimpleRouter()
router.register(r'users', UserViewSet)
router.register(r'accounts', AccountViewSet)

urlpatterns = [
    url(r'^forgot-password/$', ForgotPasswordFormView.as_view()),
]

urlpatterns += router.urls
```

Ejemplo

http://www.django-rest-framework.org/

http://www.cdrf.co/

@quiqueportac

Gracias.