

سلط بر پاسخ‌ها در DRF

کاوش عمیق و کامل در آبجکت `rest_framework.response.Response`

چرا `Response` فراتر از `HttpResponse` استاندارد جنگو



پاسخ استاندارد جنگو

- محتوای رندر شده و نهایی را به عنوان ورودی می‌پذیرد (مثلاً یک رشته JSON از پیش ساخته شده).
- نوع محتوا (Content-Type) باید به صورت دستی و صریح مشخص شود.
- انعطاف‌پذیری محدودی برای پشتیبانی از فرمتهای خروجی متعدد دارد.



پاسخ قدرتمند DRF

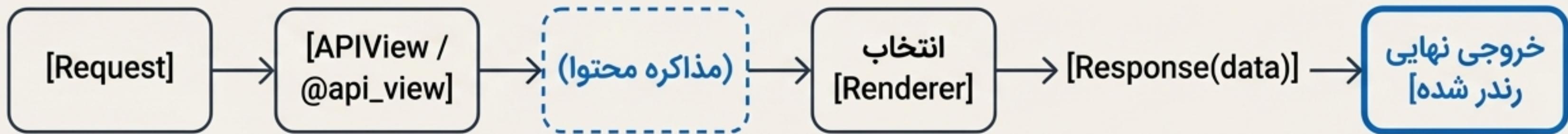
- داده‌های خام و سریالایزنده پایتون را می‌پذیرد (مانند دیکشنری‌ها و لیست‌ها).
- به طور خودکار از مذاکره محتوا (Content Negotiation) پشتیبانی می‌کند.
- به کلاینت اجازه می‌دهد تا فرمات پاسخ مورد نظر خود را (مانند JSON یا XML) درخواست کند و DRF به طور خودکار پاسخ را در آن فرمات رندر می‌کند.

معماری پاسخ: `Response` در اکوسیستم DRF کار می‌کند

« کلاس `Response` از کلاس `SimpleTemplateResponse` جنگو ارث بری می‌کند. »

« این به این معنی است که جزئیات مربوط به context در آبجکت باقی می‌ماند و رندر نهایی محتوا تا آخرین لحظه در چرخه پاسخ به تعویق می‌افتد. »

نکته حیاتی: « همیشه از `APIView` یا دکوریتور `@api_view` برای viewهایی که آبجکت بازمی‌گردانند، استفاده کنید. این کار تضمین می‌کند که view می‌تواند مذاکره محتوا را انجام دهد و مناسب را برای پاسخ انتخاب کند، پیش از آنکه پاسخ از view بازگردانده شود. »

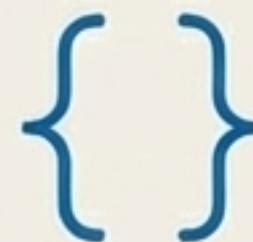


ساخت یک پاسخ: معرفی سازنده `Response()`

```
Response(data, status=None, template_name=None,  
         headers=None, content_type=None)
```

«برخلاف `HttpResponse`، شما آبجکت `Response` را با محتوای رندر شده ایجاد نمی کنید. به جای آن، داده های رندر نشده را که می توانند شامل هر نوع داده اولیه پایتون باشد، به آن پاس پاس می دهید.»

کالبدشکافی سازنده `Response(): آرگومان‌های اصلی



data

داده‌های سریالایز شده برای پاسخ. این تنها آرگومان ضروری است.



status

کد وضعیت HTTP برای پاسخ. مقدار پیش‌فرض آن OK است.

`status=status.HTTP_201_CREATED`



template_name

نام تمپلیتی که در صورت انتخاب HTMLRenderer برای رندر کردن پاسخ استفاده می‌شود.

کالبدشکافی سازنده(`Response): آرگومان‌های پیشرفتی

headers

یک دیکشنری از هدرهای HTTP که باید در پاسخ استفاده شوند.

content_type

نوع محتوای پاسخ. به طور معمول، این مقدار به صورت خودکار توسط Renderer و بر اساس نتیجه مذاکره محتوا تعیین می‌شود، اما مواردی وجود دارد که شاید لازم باشد آن را به صراحت مشخص کنید.

مثال کاربردی

```
# An example of returning a 201 Created response with a Location header.  
headers = {'Location': '/api/items/123/'}  
  
return Response(serializer.data,  
                status=status.HTTP_201_CREATED,  
                headers=headers)
```

پیش‌نیاز اصلی: داده‌ها باید سریالایز شوند



هشدار مهم: «`Renderer`‌های استفاده شده توسط کلاس `Response` به طور ذاتی نمی‌توانند با انواع داده‌های پیچیده مانند نمونه‌های مدل جنگو (Model instances) کار کنند.»

راه حل

«شما باید قبل از ایجاد آبجکت `Response`، داده‌ها را به انواع داده اولیه (primitive datatypes) پایتون سریالایز کنید.»

ابزار پیشنهادی

«برای انجام این کار می‌توانید از کلاس‌های `Serializer` در DRF یا روش سریالایز کردن سفارشی خود استفاده کنید.»



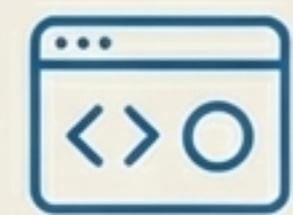
[Django Model Instance]



[Serializer]

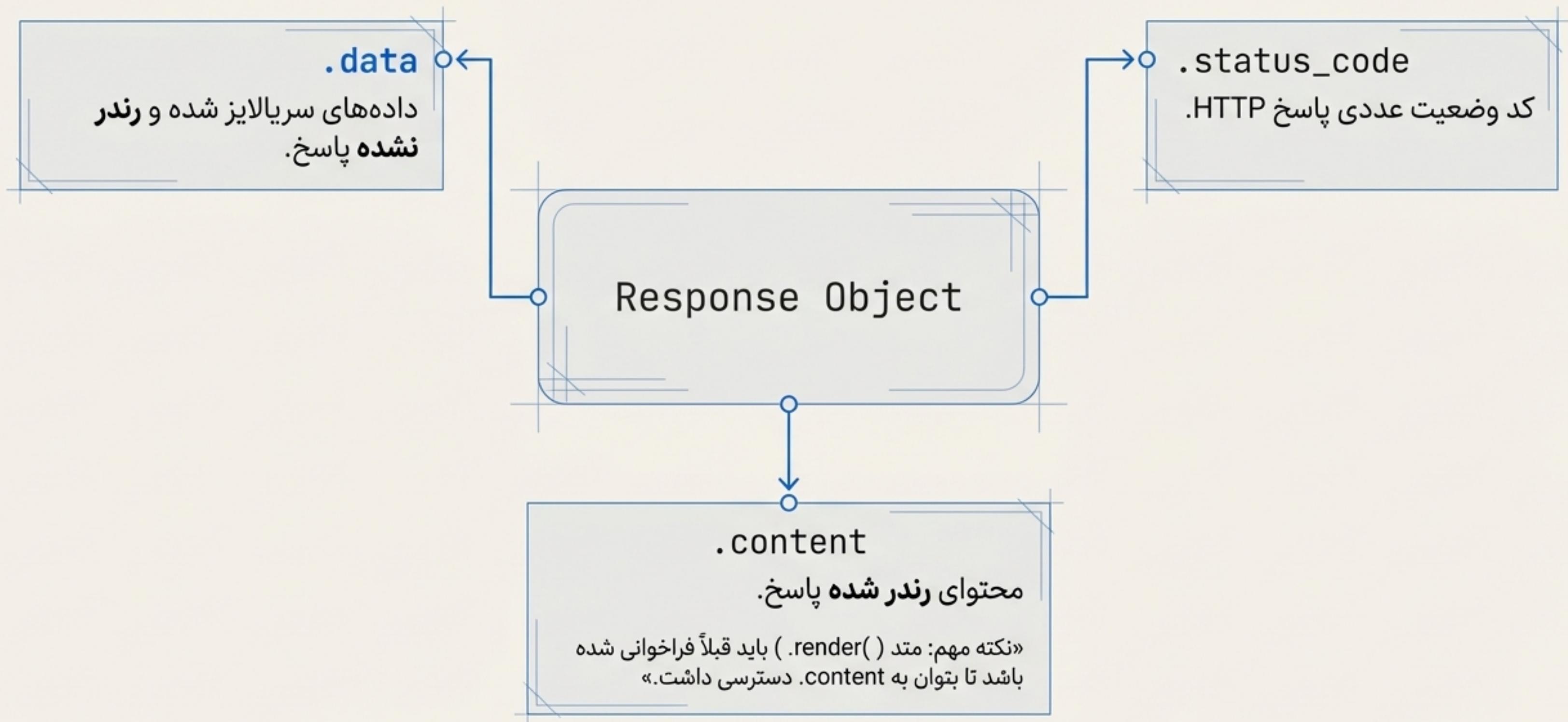


[Python Dictionary/List]



[Response()]

آناتومی یک `Response`: ویژگی‌های اصلی



آناتومی یک `Response`: ویژگی‌های مربوط به Renderer

template_name	نام تمپلیت، در صورت ارائه.
accepted_renderer	نمونه‌ای (instance) از کلاس Renderer که برای رندر کردن پاسخ پاسخ استفاده خواهد شد.
accepted_media_type	نوع رسانه‌ای (Media Type) که در مرحله مذاکره محتوا انتخاب شده است (مثلًاً (application/json)
renderer_context	دیکشنری از اطلاعات اضافی که به متدها render() ارسال می‌شود.

نکته کلیدی «این ویژگی‌ها به طور خودکار توسط `APIView` یا `@api_view` APIView قبل از بازگشت پاسخ از view، تنظیم می‌شوند.»

به ارث بردن قدرت: ویژگی‌های استاندارد `HttpResponse`

«کلاس `Response` از `SimpleTemplateResponse` ارث بری می‌کند، بنابراین تمام ویژگی‌ها و متدهای معمول در دسترس هستند.»

«برای مثال، شما می‌توانید هدرهای پاسخ را به روش استاندارد جنگو تنظیم کنید.»

```
response = Response()  
response['Cache-Control'] = 'no-  
cache'
```



نگاهی به زیر کاپوت: متدهای .render()

امضا

.render()

وظیفه

«مانند هر دیگری، این متدهای سریالایز شده به محتوای نهایی پاسخ فراخوانی می‌شود.»

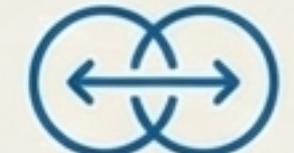
نحوه عملکرد

«هنگام فراخوانی `().render(data, render_context)`، محتوای پاسخ برابر با نتیجه فراخوانی متدهای .accepted_renderer(`.`)، .accepted_media_type(`.`)، .render_context(`.`) بر روی نمونه قرار می‌گیرد.»

نکته بسیار مهم

«شما به طور معمول نیازی به فراخوانی مستقیم `render()` ندارید، زیرا این کار توسط چرخه پاسخ استاندارد جنگو مدیریت می‌شود.»

جمع‌بندی: نکات کلیدی برای تسلط بر `Response`

- همیشه داده‌های سریالایز شده و خام پایتون را به `Response` پاس دهید، نه محتوای از پیش رندر شده.
- از قدرت مذاکره محتوا (Content Negotiation) برای پشتیبانی از فرمتهای مختلف خروجی بدون کدنویسی اضافی بهره ببرید.
- همیشه از `@api_view` یا `APIView` استفاده کنید تا فرآیند انتخاب Renderer را تنظیم کنید تا فرآیند انتخاب `context` به صورت خودکار انجام شود.
- به یاد داشته باشید که می‌توانید هدرها و سایر ویژگی‌های استاندارد `HttpResponse` را مستقیماً روی آبجکت `Response` تنظیم کنید.