



شیء `HttpRequest` در DRF: فراتر از یک `Request` معمولی

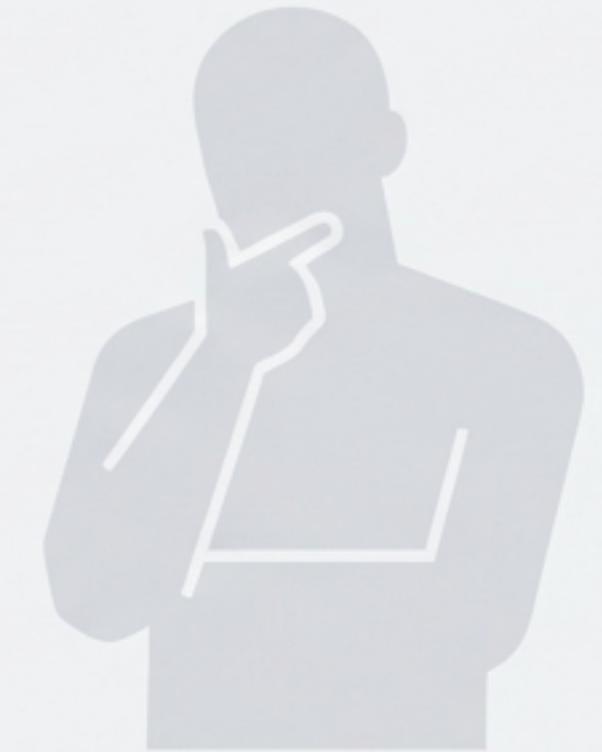
مروری عمیق بر ابزاری که DRF برای تحلیل، احراز هویت و مدیریت درخواست‌های API در اختیار شما قرار می‌دهد.

نقطه شروع: چرا باید `request.POST` را نادیده بگیریم؟

“

اگر در حال کار روی سرویس‌های وب، مبتنی بر REST هستید... باید "request.POST" را نادیده بگیرید.

– گروه توسعه‌دهندگان جنگو – Malcom Tredinnick,

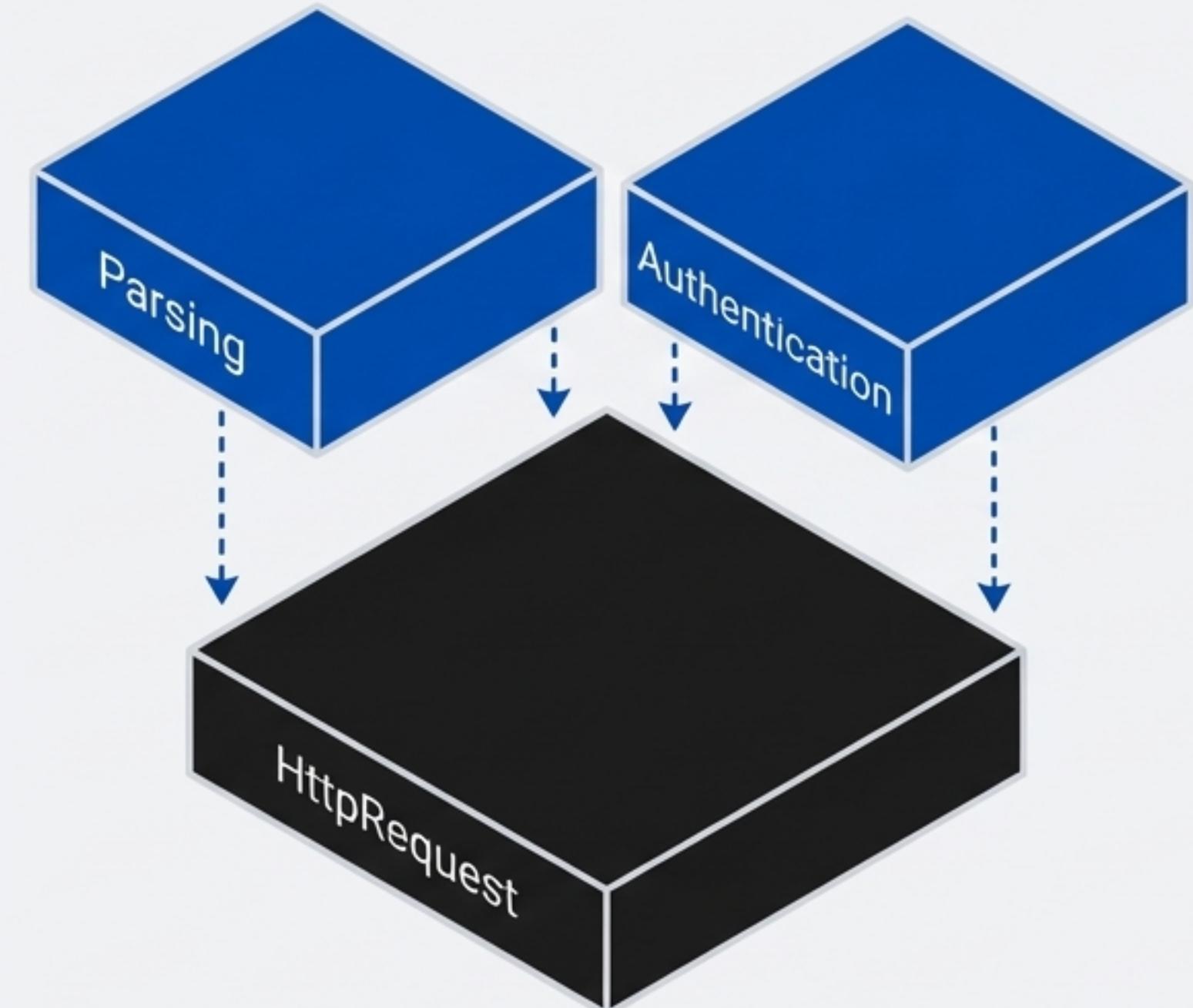


برای API‌های مدرن که با داده‌های JSON، متدهای PUT و PATCH و انواع رسانه‌های مختلف سروکار دارند، ابزارهای سنتی جنگو منه مانند `request.POST` محدودیت‌هایی دارند.

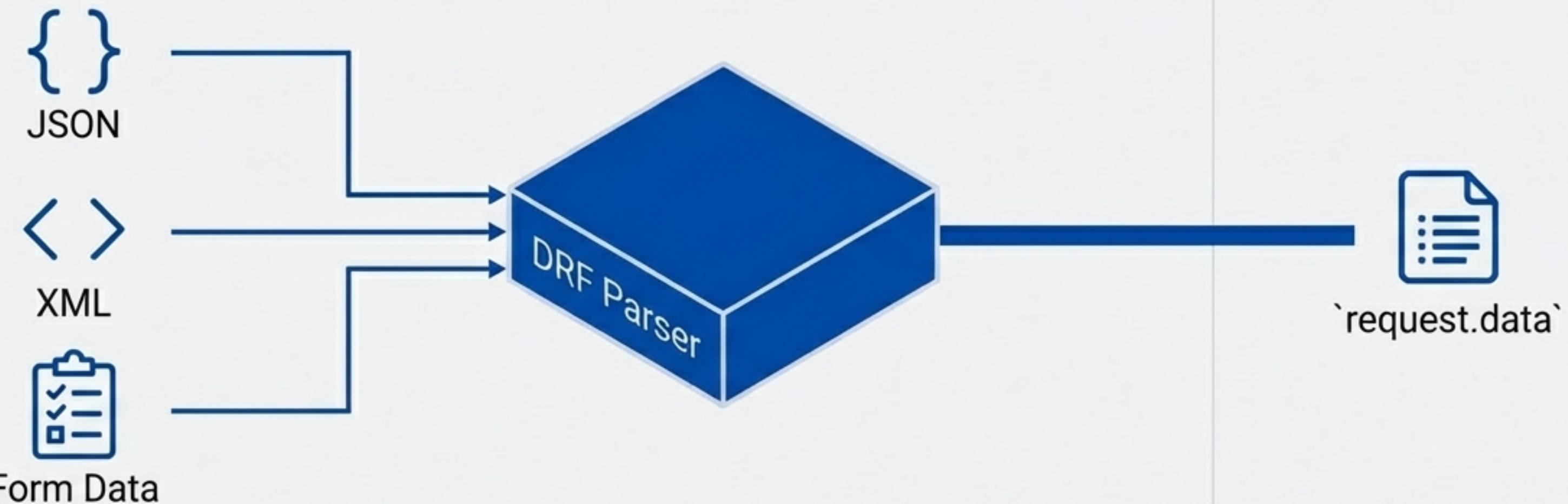
معرفی راه حل: شیء `Request` در DRF

کلاس `Request` در فریمورک REST، کلاس استاندارد `HttpRequest` را گسترش می‌دهد و قابلیت‌های جدیدی برای دو حوزه کلیدی اضافه می‌کند:

1. تحلیل انعطاف‌پذیر درخواست (Flexible Request Parsing)
2. احراز هویت درخواست (Request Authentication)



ارتقاء اول: تحلیل انعطاف‌پذیر درخواست (Request Parsing)



اشیاء `Request` در DRF به شما این امکان را می‌دهند که با درخواست‌های حاوی داده‌های JSON یا دیگر انواع رسانه، دقیقاً همانند داده‌های فرم (form data) رفتار کنید. این پیچیدگی تحلیل دستی محتوای درخواست را از بین می‌برد.

محتوای یکپارچه بدن درخواست: `request.data`

معادل قدرتمند در DRF	ویژگی در جنگو
`request.POST` + `request.FILES`	<code>request.data</code>

محتوای تحلیل شده (parsed) بدن درخواست را بر می گرداند. این ویژگی شبیه به `request.FILES` و `request.POST` است، با سه تفاوت کلیدی:

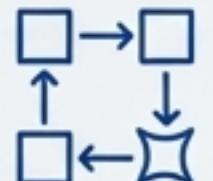
شامل تمام محتوا: ورودی های فایلی و غیر فایلی را با هم در بر می گیرد.



پشتیبانی از متدهای مختلف: محتوای متدهای `PUT` و `PATCH` را نیز در دسترس قرار می دهد.



تحلیل انعطاف پذیر: علاوه بر داده های فرم، فرمتهایی مانند JSON را نیز به طور خودکار تحلیل می کند.



نامی: `request.query_params` نامی دقیق‌تر برای پارامترهای URL

مترا遁ی با نام صحیح‌تر `request.query_params` برای `request.GET` است.

چرا استفاده کنیم؟

استفاده از آن کد شما را صحیح‌تر و واضح‌تر می‌کند. هر نوع هر نوع متده HTTP (نه فقط `GET`) می‌تواند پارامترهای کوئری (query parameters) داشته باشد.

روش استاندارد جنگو

```
روش استاندارد جنگو  
user_id = request.GET.get('user_id')
```

روش توصیه‌شده و واضح‌تر در DRF

```
# DRF روشن توصیه شده و واضح‌تر در  
user_id = request.query_params.get('user_id')
```

Parsing هوشمند خطاها

به طور پیشفرض خطاها رایج در تحلیل محتوا درخواست را مدیریت کرده و پاسخهای مناسبی برمی‌گرداند.



محتوا ناقص یا اشتباه `ParseError`:

شرط: اگر کلاینت محتوا ناقص یا با فرمت اشتباه ارسال کند.

نتیجه: دسترسی به `request.data` یک استثناء `UnsupportedMediaType` ایجاد می‌کند. DRF این خط را گرفته و یک پاسخ `400 Bad Request` برمی‌گرداند.



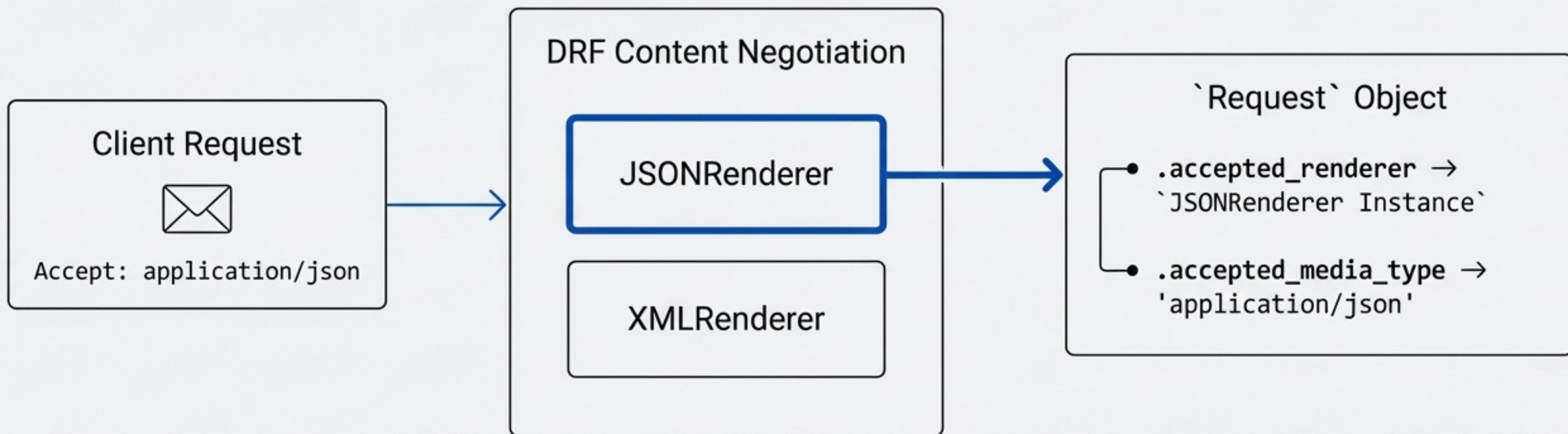
نوع محتوا پشتیبانی نشده `UnsupportedMediaType`:

شرط: اگر کلاینت درخواستی با `Content-Type` ارسال کند که قابل تحلیل نباشد.

نتیجه: یک استثناء `UnsupportedMediaType` ایجاد شده و پاسخ `415 Unsupported Media Type` برگردانده می‌شود.

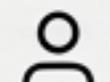
ارتقاء دوم: مذاکره بر سر محتوا (Content Negotiation)

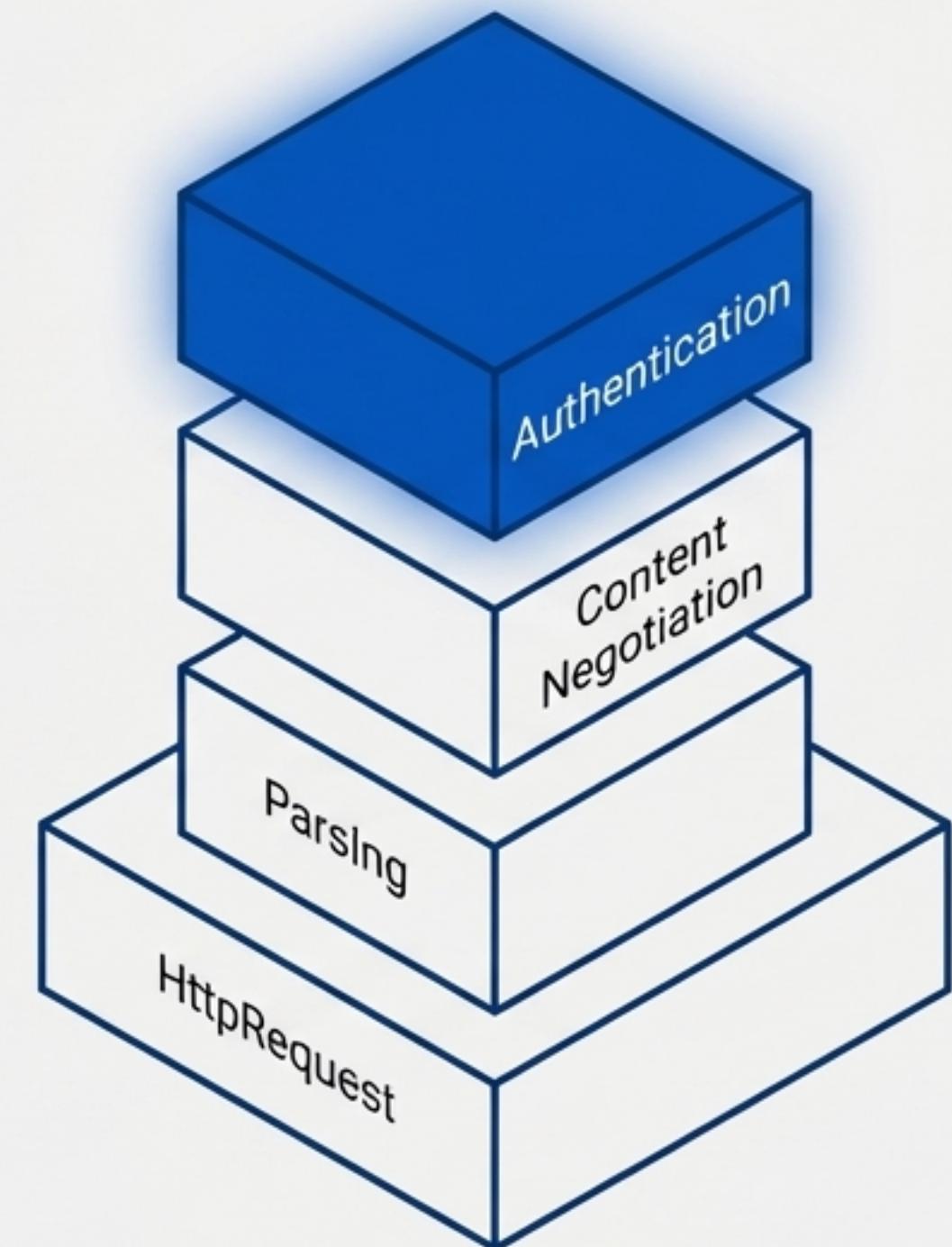
شیء `Request` نتایج مرحله "مذاکره بر سر محتوا" را در خود ذخیره می‌کند. این به شما اجازه می‌دهد تا رفتارهای متفاوتی را پیاده‌سازی کنید، مانند انتخاب یک طرح سریال‌سازی (serialization) متفاوت برای انواع رسانه‌های مختلف.



(Authentication) احراز هویت انعطاف‌پذیر

یک سیستم احراز هویت انعطاف‌پذیر و به ازای هر درخواست (per-request) ارائه می‌دهد که به شما این توانایی‌ها را می‌دهد:

- استفاده از سیاست‌های احراز هویت متفاوت برای بخش‌های مختلف API. 
- پشتیبانی از چندین سیاست احراز هویت به طور همزمان. 
- ارائه اطلاعات کاربر و توکن (token) مرتبط با درخواست ورودی. 



هويت کاربر و توکن: `request.auth` و `request.user`



`request.user`

معمولًاً يك نمونه از `django.contrib.auth.models.User` برمیگرداند (بسته به سیاست احراز هويت).

درخواست‌های احراز هويت نشده: مقدار پيش‌فرض آن يك نمونه از `django.contrib.auth.models.AnonymousUser` است.



`request.auth`

هرگونه زمينه (context) اضافي احراز هويت را برمي‌گرداند. معمولًاً نمونه‌اي از توکنی است که درخواست با آن احراز هويت شده.

درخواست‌های احراز هويت نشده: مقدار پيش‌فرض آن None است.

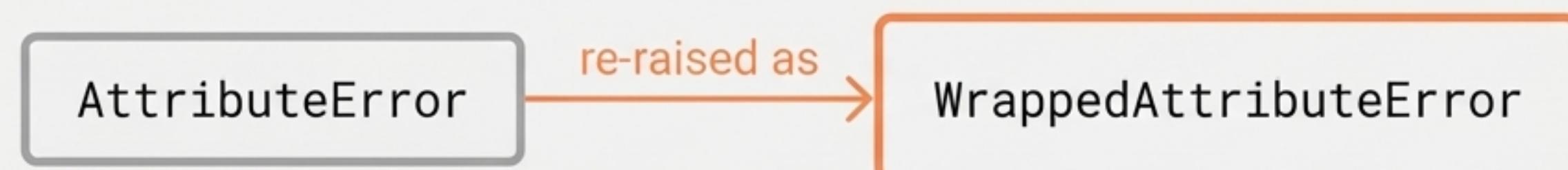
`WrappedAttributeError` خطای مهم



نکته عمیق (Deep Dive)

مشکل: هنگام فراخوانی `WrappedAttributeError` مواجه شوید.

علت: این خطا در اصل یک `authenticator` استاندارد است که از درون یک `AttributeError` رخدیده است. آن را به نوع دیگری تغییر می‌دهد تا توسط مفسر پایتون نادیده گرفته نشود.



راه حل: اگر این خطا را مشاهده کردید، مشکل از شیء `Request` نیست. باید `authenticator` خود را اصلاح کنید.

ارتقاء چهارم: بهبودهای ویژه مرورگر (Browser Enhancements)

از چند بهبود برای مرورگرها پشتیبانی می‌کند، مانند فرم‌های مبتنی بر مرورگر برای متدهای `PUT`، `PATCH` و `DELETE`.

	.method	همیشه متد HTTP درخواست را به صورت یک رشته با حروف بزرگ برمی‌گرداند و به طور شفاف از متدهای فوق در فرم در فرم‌های مرورگر پشتیبانی می‌کند.
	.content_type	نوع رسانه بدن درخواست را برمی‌گرداند و به طور شفاف محتوای غیر فرمی (non-form) مبتنی بر مرورگر را پشتیبانی می‌کند.
	.stream	یک استریم (stream) از محتوای بدن درخواست را برمی‌گرداند.

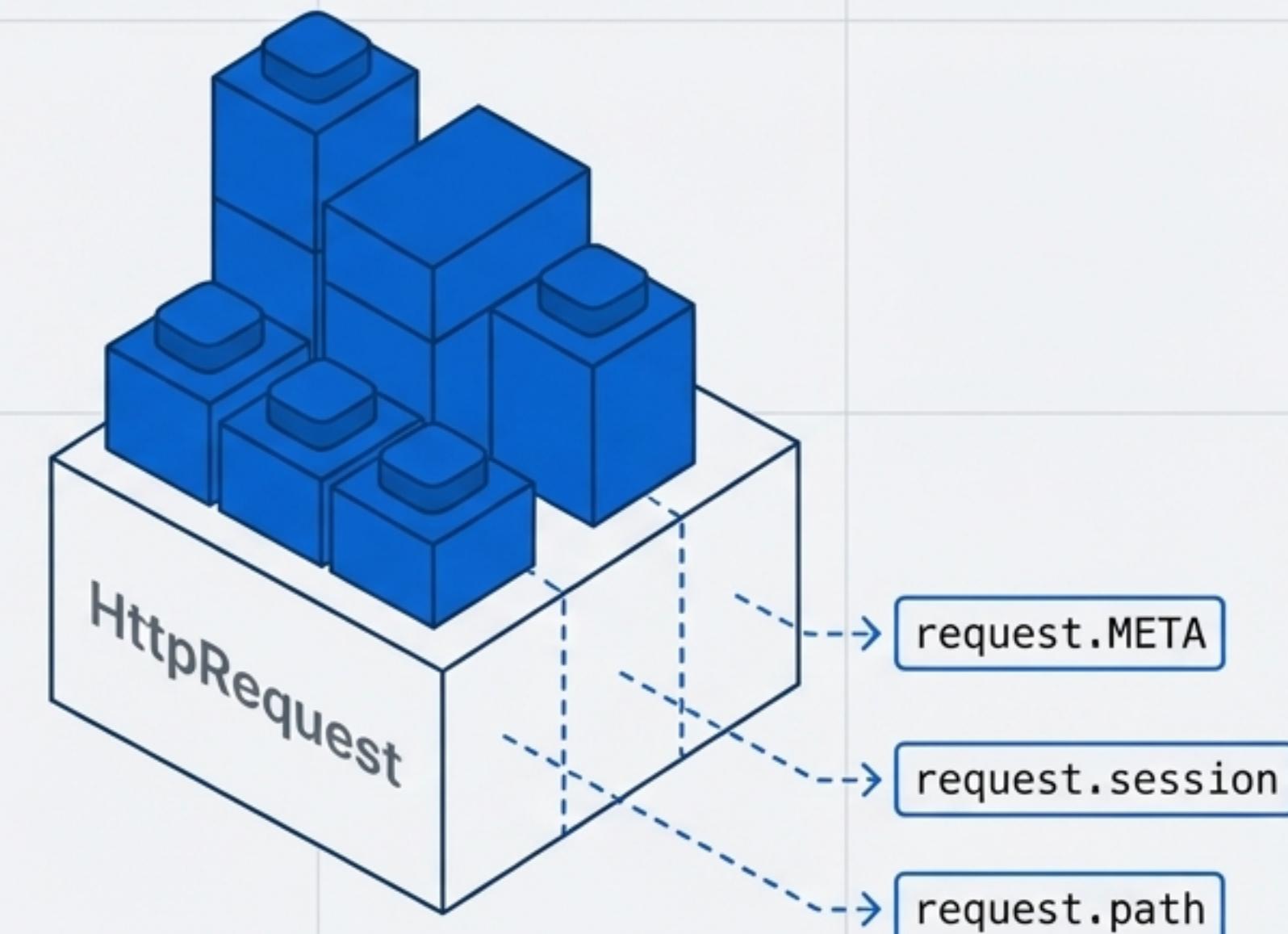
بنیاد همچنان پابرجاست: دسترسی به `HttpRequest`‌های تمام ویژگی‌های

از آنجایی که `HttpRequest` در DRF کلاس `Request` در گسترش می‌دهد، تمام ویژگی‌ها و متدهای استاندارد دیگر نیز در دسترس هستند.

مثال‌ها: شما همچنان می‌توانید به‌طور معمول از موارد زیر استفاده کنید:

- `request.META`
- `request.session`
- `request.path`

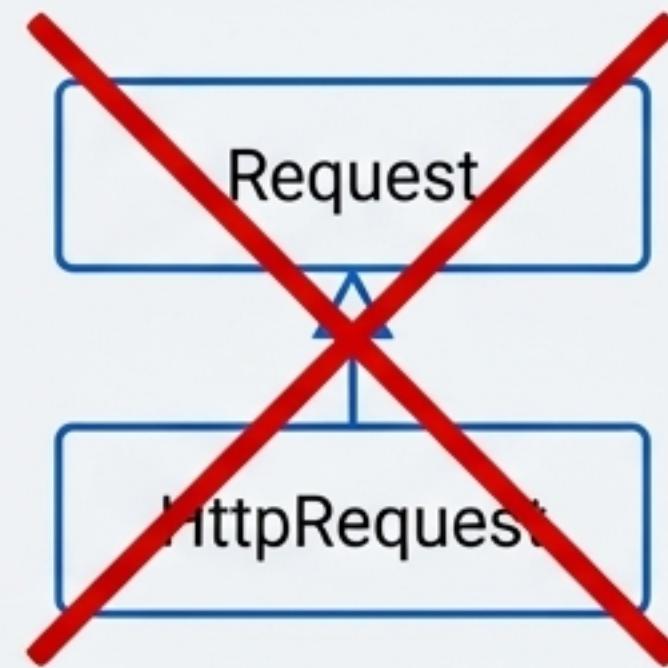
و سایر ویژگی‌های استاندارد...



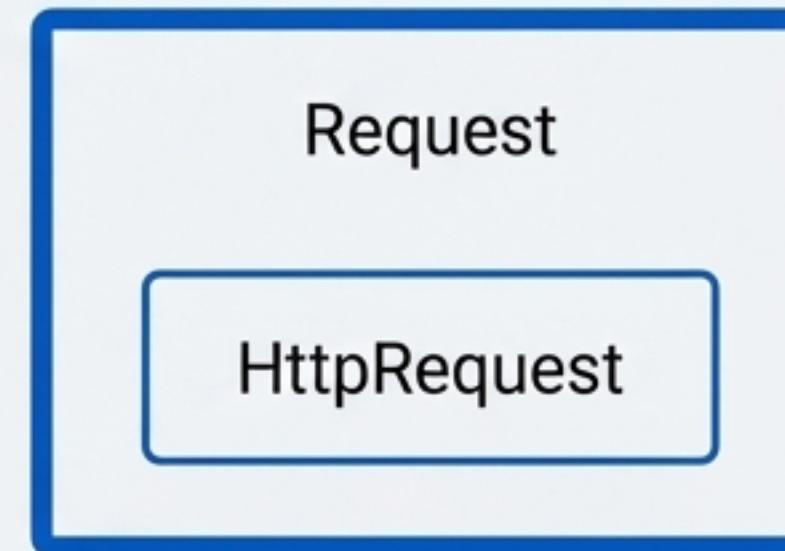
(Inheritance vs. Composition) : وراثت یا ترکیب؟

”توجه داشته باشید که به دلایل پیاده‌سازی، کلاس `HttpRequest` از کلاس `Request` ارث بری نمی‌کند، بلکه با استفاده از ترکیب (composition) آن را گسترش می‌دهد.“

Inheritance (Incorrect)



Composition (Correct)



این بدان معناست که `HttpRequest` یک 'پوشش' (wrapper) هوشمند در اطراف `Request` است و تمام ویژگی‌های آن را به را به صورت پروکسی در دسترس قرار می‌دهد، در حالی که قابلیت‌های جدیدی به آن اضافه می‌کند.

جمع‌بندی: قدرت ووضوح با شیء `Request` در DRF

نتیجه	`Request` DRF را حل در	چالش در جنگو استاندارد
کد یکپارچه برای تحلیل انواع محتوا	request.data	مدیریت XML، JSON و ...
پشتیبانی کامل از متدهای RESTful	request.data	دسترسی به بدن `PUT/PATCH`
کد خواناتر و صحیح‌تر	request.query_params	خوانایی پارامترهای URL
سیستم احراز هویت انعطاف‌پذیر و قدرتمند	request.user و request.auth	احراز هویت پیچیده
کنترل کامل بر فرمت پاسخ API	accepted_renderer و accepted_media_type	مذاکره بر سر محتوا

شیء `Request` یک ابزار بنیادی در DRF است که به شما امکان می‌دهد کدی تمیزتر، قوی‌تر قوی‌تر و خواناتر برای API‌های خود بنویسید.