

# نظام القواعد في Django

تكليف عملي لمقرر هندسة البرمجيات

إعداد الطالب طارق العمري

# مقدمة

يهدف هذا التكليف العملي إلى دراسة إمكانيات وقوف نظام القوالب في Django، مع تسليط الضوء على المقارنة مع محركات القوالب الأخرى، وفهم منظومة الفلاتر، وتنفيذ فلتر عمل مخصص. يمثل نظام القوالب في Django أحد أهم المكونات في هذا الإطار، حيث يوفر طريقة مرنة وقوية لعرض البيانات على واجهة المستخدم.

هذا التكليف العملي يغطي مقارنة بين محركات القوالب المختلفة، وأنواع الفلاتر المدعومة في نظام Django مع أمثلة عملية، بالإضافة إلى تنفيذ فلتر مخصص يخدم غرضًا محدداً وهو حساب وقت القراءة التقريري للمحتوى.

إشراف: م. هالك المصنف

# فهرس المحتوى

عرض تدريسي شامل عن نظام القوالب في Django مع التركيز على مقارنة المحركات، استخدام الفلاتر، وإنشاء فلاتر مخصصة

أنواع الفلاتر

مقارنة محركات القوالب

مقدمة عن نظام القوالب

فلاتر HTML وأخرى

فلاتر القوائم والأرقام

فلاتر النصوص

الخلاصة والتوصيات

خطوات التنفيذ

إنشاء فلتر مخصص

# مقارنة محرّكات القوالب

مقارنة بين محرّكات القوالب المختلفة المتواقة مع Django لمساعدتك في اختيار المحرّك المناسب لمشروعك

Mako	الصيغة
أقرب إلى بايثون النقي داخل HTML	
الأداء	الأداء
يسهل كتابة كود بايثون كامل داخل القالب	يسهل كتابة كود بايثون كامل داخل القالب
الأمان	الأمان
يحتاج ضبط يدوي لـ Escaping	يحتاج ضبط يدوي لـ Escaping
التكامل	التكامل
يحتاج إعدادات إضافية للتكامل مع Django	يحتاج إعدادات إضافية للتكامل مع Django

Jinja2	الصيغة
مشابهة لـ DTL مع ميزات بايثونية إضافية	
الأداء	الأداء
سرير جدًا (أسرع بـ 4.5 مرة من DTL)	سرير جدًا (أسرع بـ 4.5 مرة من DTL)
الميزات المتقدمة	الميزات المتقدمة
يدعم استدعاء الدوال بوسائل داخل القالب	يدعم استدعاء الدوال بوسائل داخل القالب
الأمان	الأمان
Escaping تلقائي للحماية من XSS	Escaping تلقائي للحماية من XSS
التكامل	التكامل
تكامل جيد مع Django منذ الإصدار 1.8	تكامل جيد مع Django منذ الإصدار 1.8

Django Template Language	الصيغة
% {{ variable }}	صيغة مبسطة
{% logic %}	صيغة مبسطة
الأداء	الأداء
متوسط ولكن كافي لمعظم المشاريع	متوسط ولكن كافي لمعظم المشاريع
الميزات المتقدمة	الميزات المتقدمة
محدودة نسبياً لتشجيع فصل المنطق عن العرض	محدودة نسبياً لتشجيع فصل المنطق عن العرض
الأمان	الأمان
XSS تلقائي للحماية من	XSS تلقائي للحماية من
التكامل	التكامل
Django افتراضي ممتاز مع	Django افتراضي ممتاز مع

# أنواع الفلاتر في Django

تُستخدم الفلاتر في نظام قوالب Django لتنسيق وتعديل البيانات قبل عرضها للمستخدم. توفر هذه الأدوات طرحاً مرنة وفعالة لمعالجة النصوص والقوائم والأرقام وعناصر HTML دون الحاجة إلى كتابة شيفرة بايثون في القوالب.

## فلاتر النصوص

مجموعة من الأدوات لمعالجة وتنسيق النصوص، مثل اختصار النص، تحويل الأدרכ، تحويل النص إلى تنسيق URL، وتنظيف المحتوى النصي من الأدרכ غير المرغوب فيها.

## فلاتر القوائم والمصفوفات

أدوات للتعامل مع القوائم والمجموعات، بما في ذلك دمج العناصر، فرز القواميس، عكس الترتيب، وعرض عناصر محددة ضمن نطاق معين.

## فلاتر الأرقام والتاريخ

فلاتر متخصصة لتنسيق الأرقام والقيم العددية والتاريخ، مثل تحديد عدد الخانات العشرية، تنسيق العملات، وعرض التواريخ بصيغ مختلفة.

## فلاتر HTML والأمان

فلاتر خاصة بالتعامل مع شيفرة HTML، بما في ذلك تأمين المحتوى ضد هجمات XSS، والسماح بعرض HTML آمن عند الحاجة.

# أمثلة فلتر النصوص

فلاتر النصوص في Django تقدم أدوات قوية لتنسيق وتحصيص عرض البيانات النصية داخل القوالب، مما يسمح بعمليات معالجة نصية معقّدة بسطّر واحد من الكود.

## slugify فلتر

يدخل النص إلى صيغة URL مناسبة عن طريق استبدال المسافات بشرطات (-)، وإزالة الأدרכ الفاصلة، وتحويل الأدרכ إلى أدرف صغيرة.

```
{{ title|slugify }}
```

## truncatewords فلتر

يفعّم هذا الفلتر باختصار النص بعد عدد معين من الكلمات مع إضافة (...) للإشارة إلى أن النص مقطّع. يُفيد في العناوين وملخصات المحتوى.

```
{{ summary|truncatewords:7 }}
```

# أمثلة فلاتر القوائم والأرقام

تساعدنا الفلاتر في Django على معالجة وعرض البيانات بطريقة مزنة وفعالة، خاصة عند التعامل مع القوائم والأرقام في واجهات المستخدم.

## فلاتر الأرقام

**floatformat:** يتحكم في عدد الخانات العشرية للأرقام، مفيد لعرض الأسعار والقيم الرقمية بتنسيق موحد.

```
{{ price|floatformat:2 }}
```

**divisibleby:** يتتحقق ما إذا كان الرقم قابل للقسمة على قيمة معينة، مفيد في العرض الشرطي للبيانات.

```
الرقم قابل {{ num|divisibleby:"3" }} للفحص على 3
{{ num|divisibleby:"3" }}: {{ num }} %
```

## فلاتر القوائم

**join:** يدمج عناصر القائمة بفواصل محددة، مما يسهل عرض قائمة من العناصر بتنسيق أنيق.

```
{{ tags|join:", " }}
```

**dictsort:** يرتيب قائمة من القواميس حسب مفتاح معين، مفيد جدًا عند عرض بيانات مجدولة.

```
{{ for student in students|dictsort:"grade" }}
{{ student.name }}: {{ student.grade }}
{{ endfor }}
```

# أمثلة فلتر HTML وأخر

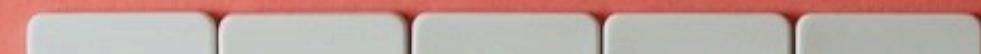
تساعد فلتر HTML في Django على التعامل مع محتوى HTML بأمان، حيث تقوم بحماية التطبيق من هجمات XSS وتحمّن المطوريين مرونة في عرض المحتوى الديناميكي

escape فلتر

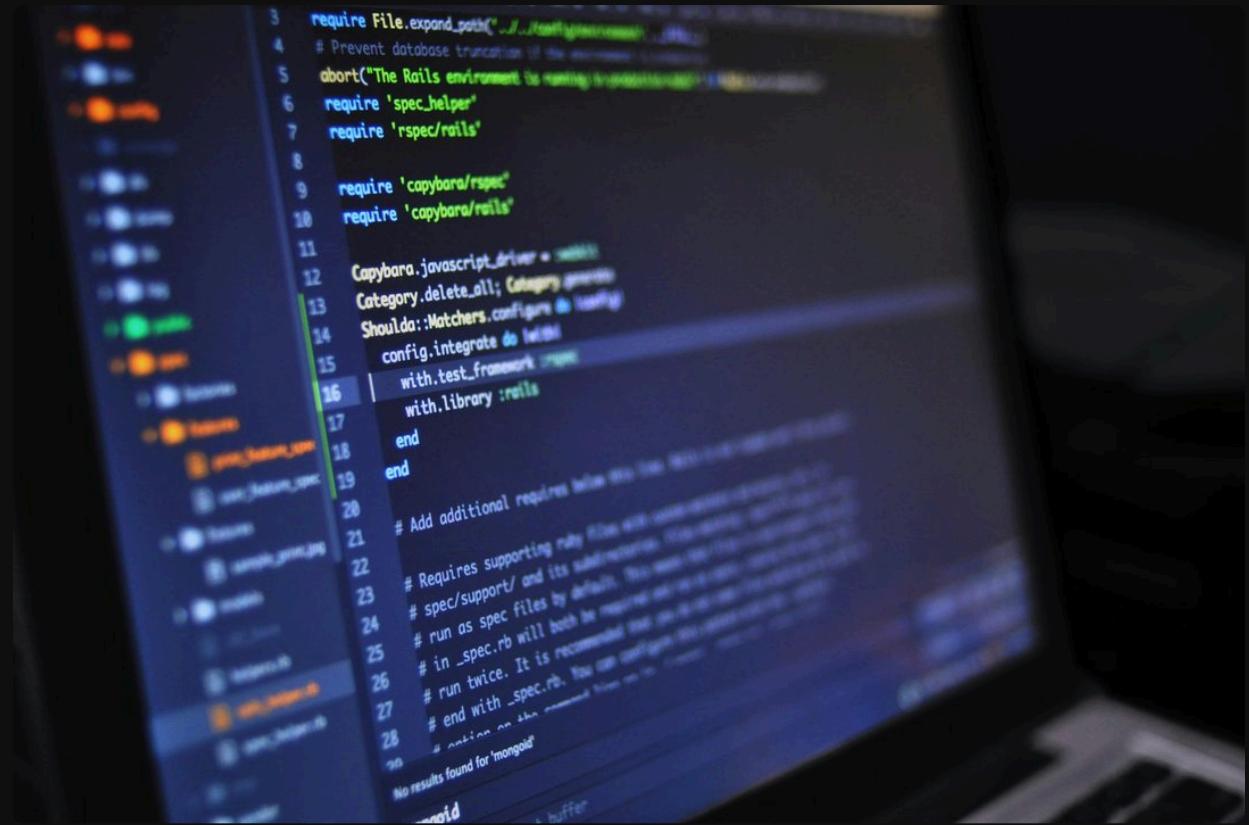
```
{{ html_content|escape }}
```

safe فلتر

```
{{ html_content|safe }}
```



# إنشاء فلتر مخصص



تتيح Django إمكانية إنشاء فلتر مخصص لتلبية احتياجات محددة في معالجة وعرض البيانات بطرق لا توفرها الفلتر الافتراضية، مما يجعل تطوير واجهات المستخدم أكثر مرونة وتحفيزاً.

## I إنشاء مجلد templatetags

يجب إنشاء مجلد باسم `templatetags` داخل تطبيق Django الخاص بك، مع إضافة ملف `__init__.py` فارغ لجعله حزمة بايثون صالحة. هذا المجلد هو المكان المحدد الذي تبحث فيه Django عن الفلتر المخصص.

## II إنشاء ملف Python للفلتر

إنشاء ملف جديد داخل المجلد (مثلاً `app_tags.py`), وفيه نقوم بتسجيل الفلتر المخصص باستخدام الديكوريتور `register.filter`: register.filter:

```
from django import template
register = template.Library()

@register.filter(name='custom_filter')
def custom_filter_function(value):
    # معالجة القيمة
    return processed_value
```

## III استخدام الفلتر في القالب

لستخدام الفلتر المخصص في قالب HTML، يجب أولاً تحميل ملف الفلتر المخصص ثم استخدام الفلتر مع القيمة المطلوبة:

```
{% load app_tags %}
{{ my_value|custom_filter }}
```

# مثال عمل: فلتر حساب وقت القراءة

فلتر يقوم بحساب الوقت التقديري لقراءة محتوى نصي بافتراض معدل قراءة 200 كلمة في الدقيقة

```
# app_tags.py داخل مجلد templatetags
from django import template
import math

register = template.Library()

@register.filter(name='readtime')
def calculate_read_time(text):
    if not isinstance(text, str):
        return 0
    word_count = len(text.split())
    minutes = math.ceil(word_count / 200.0)
    return f"{minutes} دقيقة قراءة"
```

```
# استخدم الفلتر في قالب HTML
{% load app_tags %}
وقت القراءة التقريبي: {{ article.content|readtime }}
```

## خطوات تنفيذ الفلتر المخصص

### إنشاء مجلد templatetags

يتم إنشاء مجلد باسم templatetags داخل التطبيق مع إضافة ملف \_\_init\_\_.py فارغ، ثم إنشاء ملف باسم app\_tags.py لتعريف الفلتر

### تسجيل المكتبة والفلتر

تعريف المكتبة باستخدام register = template.Library() وتسجيل الفلتر بواسطه register.filter@القوالب

### استخدام الفلتر في القالب

تحميل الفلتر باستخدام {% load app\_tags %}، ثم استدعاء الفلتر من خلال وضع علامة | مع اسم الفلتر بعد المتغير: {{ article|readtime }}

### كتابة منطق حساب وقت القراءة

حساب عدد الكلمات في النص باستخدام split() ثم قسمة العدد على 200 (معدل القراءة) وتقريب الناتج لأعلى رقم صحيح



# الخدمة

أظهر هذا التكليف مرونة Django في دعم محركات قوالب متعددة، وقوة نظام الفلاتر في تخصيص عرض البيانات، سواء باستخدام الفلاتر الجاهزة أو بإنشاء فلاتر مخصصة.

هذه المعرفة توفر للمطور أدوات قوية لتحسين الأداء، وتوسيع إمكانيات العرض، وتسهيل بناء واجهات تفاعلية ومتکاملة مع الحفاظ على الأمان والكفاءة.

# شكراً لبسن استماعكم!

إعداد: طارق العمري