مقارنة محركات القوالب ومُرشحات لغة قوالب جانغو

دراسة تحليلية

الطالب: **فتح الرحمن الياسري**

الدكتور: **مالك المصنف**

التقنية: **معلومات**

الجامعة/الكلية: **جامعة إب / كلية العلوم**

المستوى: **رابع**

مقدمة نظرية عن محركات القوالب

الأهمية والتطور في تطوير الويب

</> ما هي محركات القوالب؟

أنظمة برمجية تسمح بدمج البيانات مع قوالب HTML لإنشاء مستندات ديناميكية، مع الفصل بين منطق التطبيق وعرض البيانات.

- **الفصل بين المنطق والعرض** - تطبيق نمط MVC/MVT لتحسين تنظيم الكود
- **إعادة استخدام الكود** - تقليل التكرار عبر الوراثة والكتل القابلة لإعادة الاستخدام
- **الأمان** - الحماية التلقائية من هجمات حقن الكود (XSS, CSRF)
- **تسهيل الصيانة** - فصل المسؤوليات وتسهيل التعديل على واجهة المستخدم

التطور التاريخي وأهميتها في جانغو

2005 إطلاق إطار عمل جانغو مع محرك القوالب المدمج (DTL)

2008 ظهور Jinja2 كبديل أسرع وأكثر مرونة لـ DTL

2010 تطور Mako و Genshi كخيارات متخصصة للأداء العالي

2015 دمج دعم أفضل لمحركات القوالب البديلة في جانغو

حالياً تنوع الخيارات مع تركيز على الأمان والأداء والمرونة

محركات القوالب في بايثون

نظرة عامة على المحركات المختلفة

محركات القوالب في بايثون توفر حلولاً متنوعة لفصل العرض عن المنطق البرمجي، مع اختلافات في الأداء والمرونة والأمان. إليك أهم المحركات المستخدمة:

Jinja2

الظهور: 2008

المطور: Armin Ronacher

الرخصة: BSD

المميزات: **آمن**، **سريع**، صيغة مشابهة لـ DTL، يدعم تعابير Python

Mako

الظهور: 2007

المطور: Mike Bayer

الرخصة: MIT

المميزات: **سريع جداً**، وراثة ديناميكية، يدعم كتل Python

Genshi

الظهور: 2006

المطور: Edgewall Software

الرخصة: BSD-like

المميزات: **محرك XML**، دعم معالجة التدفق، حماية تلقائية من XSS

Cheetah

الظهور: 2005

المطور: Tavis Rudd

الرخصة: MIT

المميزات: **سهل التعلم**، صيغة مشابهة لـ Python، أقل مرونة من المحركات الأخرى

التكامل مع إطار عمل جانغو

- **DTL** - المحرك الافتراضي في جانغو
- **Jinja2** - تكامل سهل عبر حزم خارجية
- **المحركات الأخرى** - تتطلب إعدادات إضافية

مقارنة محركات القوالب

Jinja2 و Mako

Jinja2

الأداء

- جيد
- أبطأ من Mako
- استهلاك ذاكرة **متوسط**

الأمان

- **آمن جداً**
- حماية تلقائية من XSS
- وضع الحماية التلقائي

المرونة

- دعم تعابير Python محدود
- وراثة القوالب **قوية**
- توافقية عالية مع جانغو

مثال

```
{% extends "base.html" %}
{% block content %}
  <h1>{{ title }}</h1>
  <p>{{ content|safe }}</p>
{% endblock %}
```

VS

Mako

الأداء

- **ممتاز**
- سريع جداً
- استهلاك ذاكرة **منخفض**

الأمان

- أقل أماناً من Jinja2
- حماية من XSS اختيارية
- يتطلب إعدادات أمان إضافية

المرونة

- دعم كامل لتعابير Python
- **وراثة ديناميكية**
- يتطلب إعدادات إضافية لجانغو

مثال

```
<%inherit file="base.html"/>
<%block name="content">
  <h1>${title}</h1>
  <p>${content|n}</p>
```

مقارنة محركات القوالب

Cheetah و Genshi

Genshi

الأداء

- **متوسط**
- بطيء نسبياً
- استهلاك ذاكرة **مرتفع**

الأمان

- **آمن**
- حماية تلقائية من XSS
- معالجة XML آمنة

المرونة

- **محرك XML** متخصص
- دعم معالجة التدفق
- يتطلب معرفة XML

مثال

```
<html
xmlns:py="http://genshi.edgewall.org/">
  <head>
    <title
py:content="title">عنوان</title>
  </head>
  <body>
    <div py:content="content">محتوى</div>
  </body>
</html>
```

VS

Cheetah

الأداء

- **جيد**
- سريع نسبياً
- استهلاك ذاكرة **منخفض**

الأمان

- متوسط الأمان
- حماية محدودة من XSS
- يتطلب إعدادات أمان يدوية

المرونة

- **سهل التعلم**
- صيغة مشابهة لـ Python
- أقل مرونة من المحركات الأخرى

مثال

```
#extends "base.html"

#block content
  <h1>$title</h1>
  <p>$content</p>
#end block
```

مُرشحات لغة قوالب جانغو (DTL)

تحليل أكاديمي

مُرشحات نصية

capfirst	تكبير الحرف الأول من النص
lower	تحويل النص إلى أحرف صغيرة
upper	تحويل النص إلى أحرف كبيرة
title	تحويل النص إلى حالة العنوان

مُرشحات تاريخ

date	تنسيق التاريخ حسب الصيغة المعطاة
time	تنسيق الوقت حسب الصيغة المعطاة
timesince	تنسيق التاريخ كالمنقضي منذ ذلك التاريخ
timeuntil	تنسيق التاريخ كالمتبقي حتى ذلك التاريخ

مُرشحات قوائم

first	إرجاع العنصر الأول في القائمة
last	إرجاع العنصر الأخير في القائمة
length	إرجاع طول القيمة
join	ضم قائمة بسلسلة نصية

مُرشحات أمان

escape	تهريب أحرف HTML في سلسلة
safe	تعطيل التهريب التلقائي للHTML
addslashes	إضافة شرطات قبل الاقتباسات
slugify	تحويل النص إلى صيغة URL آمنة

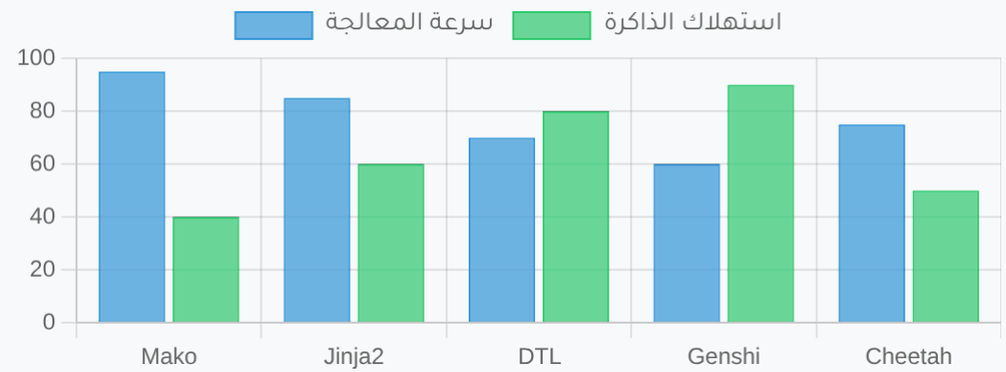
أفضل الممارسات في استخدام المُرشحات

- **الأمان أولاً** - استخدم escape دائماً للبيانات الخارجية
- **التسلسل** - يمكن سلسلة مُرشحات متعددة ({{ value|lower|capfirst }})
- **المُرشحات المخصصة** - إنشاء مُرشحات خاصة عند الحاجة

تحليل الأداء والأمان

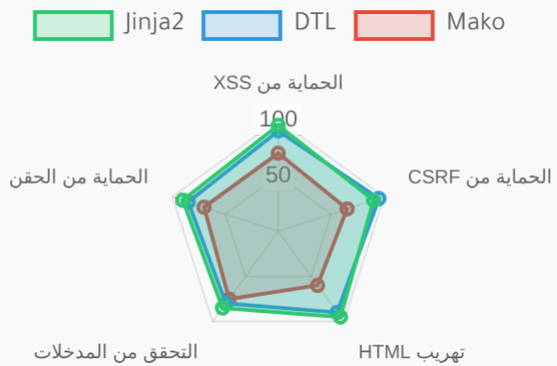
مقارنة معمقة بين محركات القوالب

الأداء



- **Mako** - الأسرع في الظروف العادية
- **Jinja2** - أداء متفوق مع البيانات الكبيرة
- **DTL** - استهلاك ذاكرة مرتفع مع التحميل العالي

الأمان



- **Jinja2** - حماية تلقائية متقدمة من XSS
- **DTL** - آمن بشكل افتراضي
- **Mako** - يتطلب إعدادات أمان يدوية

توصيات لاختيار المحرك المناسب

- **التطبيقات الصغيرة** - Jinja2 أو DTL
- **التطبيقات الكبيرة** - Mako أو Jinja2
- **المتطلبات الأمنية العالية** - Jinja2

الخلاصة والمراجع

نتائج الدراسة ومصادر البحث

النتائج الرئيسية

الأداء

Mako الأسرع في المعالجة
Jinja2 أفضل مع البيانات الكبيرة
DTL استهلاك ذاكرة أعلى

الأمان

Jinja2 حماية متقدمة من XSS
DTL آمن بشكل افتراضي
Mako يتطلب إعدادات يدوية

الاتجاهات المستقبلية

الأداء المحسن

تحسين سرعة المعالجة
تقليل استهلاك الذاكرة
دعم أفضل للتخزين المؤقت

تعزيز الأمان

حماية متقدمة من الثغرات
تحقق تلقائي من المدخلات
إعدادات أمان مبسطة

التكامل

توافق أفضل مع أطر العمل
دعم للتنسيقات الحديثة
واجهات برمجية موحدة

التوصيات

التطبيقات الصغيرة

- DTL - سهولة التكامل
- Jinja2 - مرونة وأمان

التطبيقات الكبيرة

- Jinja2 - أداء متوازن
- Mako - سرعة عالية

المتطلبات الأمنية

- Jinja2 - حماية متقدمة
- DTL - آمن افتراضياً

المراجع الأكاديمية

Ronacher, A. (2008). **Jinja2: Template Engine for Python**. Pocoo Team.

Bayer, M. (2007). **Mako Templates for Python**. SQLAlchemy Project.

Edgewall Software. (2006). **Genshi: Python Toolkit for Generation of Markup**.

Rudd, T. (2005). **Cheetah: Python-Powered Template Engine**.

Django Software Foundation. (2005). **Django Template Language**. Django Project.

Dyspatch Team. (2020). **Python Templating Performance: Django vs Jinja**. Dyspatch Blog.

zzzeek, M. (2010). **Quick Mako vs. Jinja Speed Test**. TechSpot.

OpenSource.com. (2021). **3 Python Template Libraries Compared**.