django

el curso

Día 5 – Parte 2

Día 5 – Contenido

- HttpRequest y HttpResponse
- *Middleware*
- Sesiones
- Señales
- Misc.

Día 5 – Contenido

- HttpRequest y HttpResponse
- *Middleware*
- Sesiones
- Señales
- Misc.

HttpRequest y HttpResponse

- Clases Python
- `HttpRequest`: Instancias creadas por Django y pasada a nuestras vistas
- `HttpResponse`: Instancias creadas y retornadas por nuestras vistas

HttpRequest

Atributos

- Sólo lectura (excepto `.session`)
- Cadenas: `.path`, `.method` (“GET”, “POST”)
- Diccionarios: .META (cabeceras HTTP: `CONTENT_*`, `HTTP_*`, `REMOTE_*`, ...)
- *Dictionary-like* (`QueryDict`, múltiples valores para la misma clave): `.GET`, `.POST`, `.COOKIES`, `.FILES` (valores: Instancias de `UploadedFile`)
- Instancias de dtas .clases: `.user` (`django.contrib.auth.models.user`) cuando está activa la app auth, `.session` (objeto sesión de `django.contrib.sessions`, tambien dict-like) cuando está activa la app sessions

HttpRequest
Métodos

- Métodos estándar para implementar protocolo de emulación de un dict ()
- `.is_secure()` - HTTPS
- `.is_ajax()` - Cabecera HTTP `HTTP_X_REQUESTED_WITH` envida por librerías JS populares en requests vía `XMLHttpRequest`
- `.get_full_path()` - Retorna path absoluto con query string incluída
- `.build_absolute_uri(loc)` - genera URL completa con esquema, host, puerto, full path y query string

HttpResponse

- Híbrido
- Contenido: Cadenas en constructor o método `.write()` (file-like)
- Encabezados: Sintaxis dictionary-like
- Cookies: Métodos `set_cookie()` y `.delete_cookie()`

```
resp1 = HttpResponse("Hola mundo")
resp2 = HttpResponse()
resp2.write("<li>Linea1</li>")
resp2.write("<li>Linea2</li>")
resp['Content-Disposition'] = 'attachment; filename=page.pdf'
```

- Sub-clases: Representan códigos de estado HTTP:
  `HttpResponseRedirect` (302), `HttpResponseBadRequest` (400), `HttpResponseNotFound` (404), `HttpResponsePermanentRedirect` (301), `HHttpResponseServerError` (500), `HttpResponseForbidden` (403), `HHttpResponseNotModified` (304)...

Día 5 – Contenido

- HttpRequest y HttpResponse
- *Middleware*
- Sesiones
- Señales
- Misc.

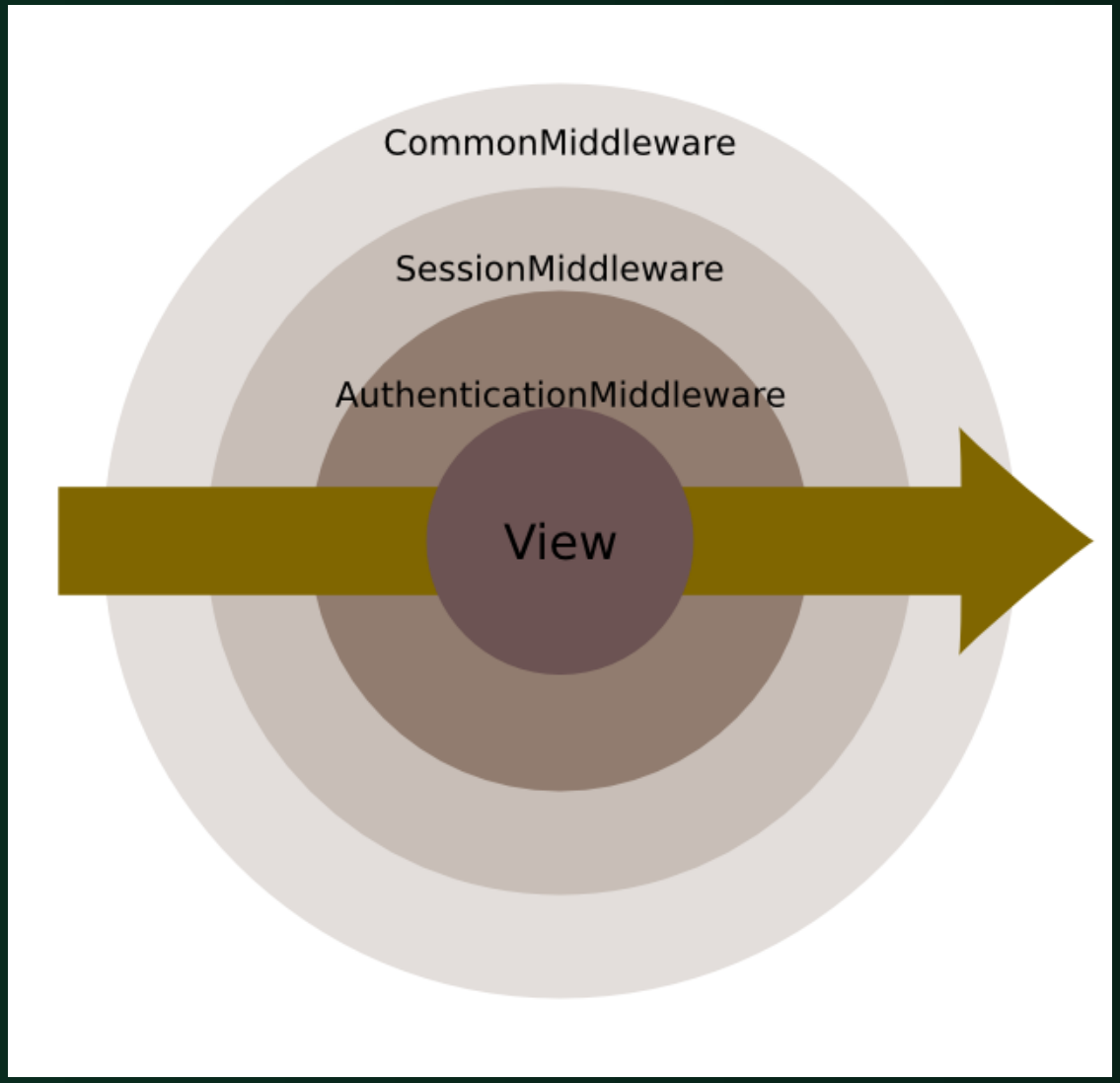
Middleware

- Sistema de acceso al proceso de cada request/response
- Clases Python, sin clase base específica
- Se habilitan y enumeran en el setting `MIDDLEWARE_CLASSES`, el orden es importante

```
MIDDLEWARE_CLASSES = (
    'django.middleware.common.CommonMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.middleware.doc.XViewMiddleware', # No existe mas
)
```

- Cuatro puntos de acceso al ciclo de vida del proceso de request/response...

Middleware

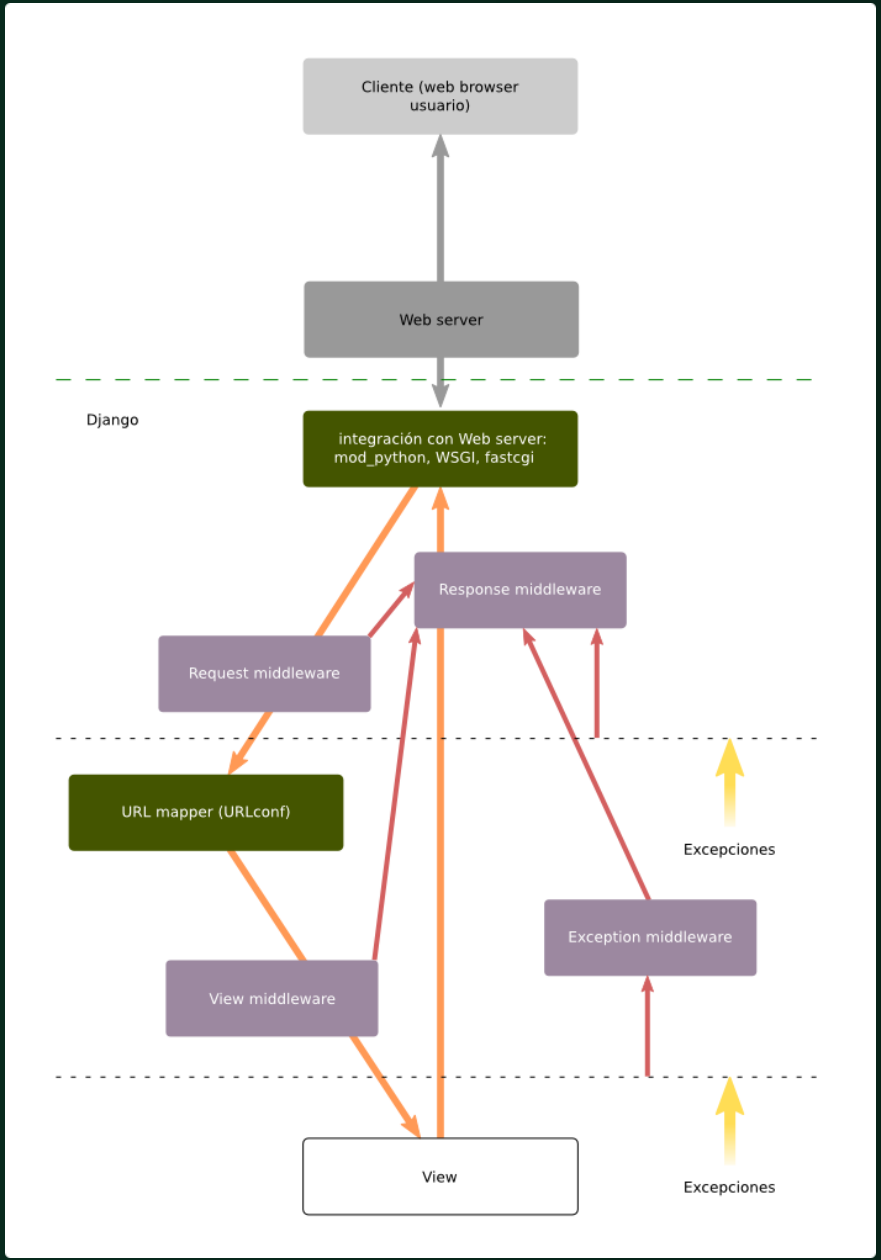

CommonMiddleware
SessionMiddleware
AuthenticationMiddleware
View

Middleware

Un middleware hace uso de esos puntos de acceso vía cuatro métodos, los valores de retorno de los misos controlan el flujo del proceso. `None` (continuar con proceso normal) o una `HttpResponse` (*cortocircuito*: Se obvia la invocación del resto del proceso, excepto `.process_response`):

- Antes de decidir qué vista invocar: Método `.process_request(...)`
- Antes de invocar la vista: Método `.process_view(...)`
- Luego de invocar a la vista: Método `.process_response(...)` - No es *cortocircuitable*
- Si la vista genera una excepción: Método `.process_exception(...)`

Middleware


Cliente (web browser
usuario)
Web server
Django
integración con Web server:
mod_python, WSGI, fastcgi
Response middleware
Request middleware
URL mapper (URLconf)
Excepciones
Exception middleware
View middleware
View
Excepciones

Middleware

Incluídos en Django

- “Common” - Tiene en cuenta settings DISALLOWED_USER_AGENTS, APPEND_SLASH, PREPEND_WWW, USE_ETAGS
- Authentication – Agrega el atributo `user` al `HttpRequest`
- Session (`django.contrib.sessions.middleware.SessionMiddleware`) – Parte del framework de sesiones
- gzip
- Cache – Dos componentes middleware
- Conditional GET – Encabezados `Last-Modified, If-Modified-Since`, clase `HttpNotModified`
- Locale, CSRF protection, transaction, reverse proxy

(cont)

Día 5 – Contenido

- HttpRequest y HttpResponse
- Middleware
- Sesiones
- Señales
- Misc.

Sesiones

Útiles para mantener el estado del lado del servidor ante la característica *state-less* del protocolo HTTP.

- Puede implementarse aún cuando el usuario no se ha autenticado (o sea no se está usando `django.contrib.auth`)
- Usa *cookies*
- No usa ni degrada a la *query string* (`PHPSESSID`)
- La solución está formada por dos partes: Un middleware (`SessionMiddleware`) y una aplicación Django (`django.contrib.sessions`, sólo cuando el backend es BD)
- Puede usar uno de tres *backends* de almacenamiento de la información de las sesiones: BD, archivos y cache. Se configura con el setting `SESSION_ENGINE`

Sesiones

Objetos Session

`SessionMiddleware` agrega una instancia de esta clase como el atributo `.session` al `HttpRequest` que reciben las vistas.

- Métodos de manejo de ciclo de vida de sesión: `.set_expiry(x)` (0 -> cuando se cierra el browser), `.flush()` (limpieza de sesión en backend, logout)
  - Settings asociados: `SESSION_EXPIRE_AT_BROWSER_CLOSE` (como `.set_expiry()` pero a nivel global), `SESSION_SAVE_EVERY_REQUEST`
- Métodos manejo de cookies: `.set_test_cookie()`, `test_cookie_worked()`, `delete_test_cookie()`
  - Settings asociados: `SESSION_COOKIE_NAME`, `SESSION_COOKIE_DOMAIN`, `SESSION_COOKIE_SECURE`, `SESSION_COOKIE_AGE`

Día 5 – Contenido

- HttpRequest y HttpResponse
- Middleware
- Sesiones
- Señales
- Misc.

Señales

Llamadas *intra-proceso* disparados cuando suceden ciertos eventos

- Son instancias de `Signal`. Existen algunos predefinidos disparados desde código de Django, pueden crearse los propios
- Se registran *callbacks* que son llamados por el framework en el momento adecuado, signature: `mi_callback(sender, **kwargs)`
- Pueden registrarse más de un *callback* con un evento determinado. la registración se hace usando el método `.connect()` de Signal
- Los *callbacks* pueden registrarse con eventos genéricos o asociados a (enviados por) una instancia en particular

Señales

Señales incluídas en Django:

- Ciclo de vida de modelos: `pre_init`, `post_init`, `pre_save`, `post_save`, `pre_delete`, `post_delete`, `class_prepared` (clase, no instancia)
- Ciclo de vida request/response: `request_started`, `request_finished`, `got_request_exception`
- Management: `post_syncdb`

Día 5 - Contenido

- HttpRequest y HttpResponse
- Middleware
- Sesiones
- Señales
- Misc.

Misc.

¿Qué le falta a Django? (07/11/2008)

- Validación en capa M - Ticket #6845
- *Aggregation* en el ORM - Ticket #3566 (GSOC 2008)
- Identity map - Ticket #17
- `SELECT de solo algunos campos` en SQL - Ticket #5420
- `SELECT FOR UPDATE` en SQL - Ticket #2705
- Soporte de conexiones a múltiples DB (¿de diferentes backends?)
- Streaming de información al cliente - Ticket #7581
- Schema evolution: Proyectos externos: dmigrations, south, django-evolution (y dbmigrations, deseb)
- Templates: Consistencia (manejo de espacios), thread safety, eficiencia

(cont)

Misc.

Política de releases

- A.B.C
  - A: Mayor, cambios *backwards*-incompatibles
  - B: Menor, cambios *backwards*-compatibles con releases con el mismo A
  - C: *Patch level* – siempre, 100% *backwards*-compatible con releases con los mismos A y B
- Timed releases: Releases menores cada ~6 meses
- 1.0: 2/9/2008
- 1.1: Marzo 2009
- 1.0.1: 14/11/208
- Soporte de seguridad: trunk SVN y dos releases menores previos

Misc.

Tintero

- CSRF (Cross site request forgery)
- Cache
- GeoDjango
- i18n/l10n
- Auto-escaping en templates
- File uploading and storage [backends]
- Envío de e-mail
- Pagination
- Comments
- Form sets, form widgets, form wizards

Ejercicio.
Ticketok

- Sistema de venta de entradas para eventos (deportivos, espectáculo,...)
- Diseñar modelos y relaciones con modelos *obvios* y adicionalmente...
- Los usuarios se registran en el sitio
- Medios de pago
- Socios en cadena de ventas
- Cada socio es dueño de uno o mas puntos de venta
- Un punto de venta puede ser propio (no de un socio)
- *Locations* en las que se realizan eventos (Luna Park, Orfeo,...)
- Zonas de butacas (ubicaciones en *locations*, dtos. precios)

Ejercicio.

Ticketok

- Los tickets pueden venderse por Internet o en puntos de venta
- En caso de venta por Internet, cada usuario tiene una dirección de cobro/entrega del ticket
- ...
- Profit!