A graphic on the left side of the slide features four stacked, 3D rectangular blocks in purple, orange, yellow, and blue. The orange block is the largest and has a white arrow pointing to the right. The text 'Agencia de Aprendizaje a lo largo de la vida' is written across the blocks in white.

Agencia de  
Aprendizaje  
a lo largo  
de la vida

# DJANGO

## Clase 11

Django: Views - 2

# Les damos la bienvenida

Vamos a comenzar a grabar la clase

## Clase 11

### Django: Views - 2

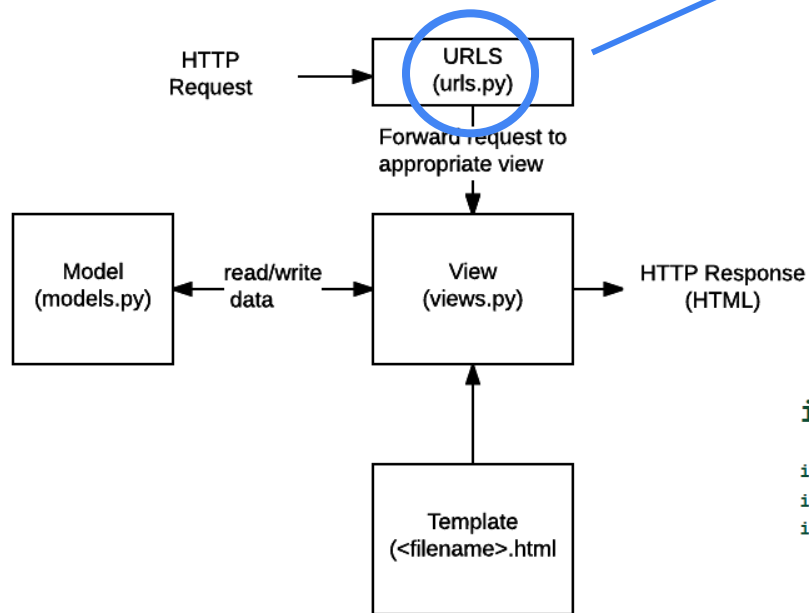
- Url y vistas parametrizadas
- Analizando el HttpRequest
- Tipos de HttpResponse
- Html desde Django

## Clase 12

### Django: Templates - 1

- Template backend
- Template loader
- Configuración
- Integración con Vistas
- Contexto
- Variables

# Vistas Parametrizadas



## django.urls:

### path()

`path(ruta, vista, kwargs = Ninguno, nombre = Ninguno)`

El argumento **ruta** debe ser una cadena de texto que contenga un patrón de URL.

### re\_path()

`re_path(ruta, vista, kwargs = Ninguno, nombre = Ninguno)`

El argumento **ruta** debe ser una cadena de texto que contenga un **expresión regular** compatible con el módulo **re** de Python.

### include()

`include(módulo, espacio de nombres = Ninguno) [fuente]`

`include(lista_patrones)`

`include((lista_de_patrones, espacio_de_nombres de la aplicación), espacio de nombres=Ninguno)`

## path()

```
path(ruta, vista, kwargs = Ninguno, nombre = Ninguno)
```

El argumento **ruta** debe ser una cadena de texto que contenga un patrón de URL.

## Disponibles por defecto

	str	int
path		
	slug	uuid

# path()

```
from django.urls import path

from . import views

urlpatterns = [
    path('articles/2003/', views.special_case_2003),
    path('articles/<int:year>', views.year_archive),
    path('articles/<int:year>/<int:month>', views.month_archive),
    path('articles/<int:year>/<int:month>/<slug:slug>', views.article_detail),
]
```

/articles/2003/

views.special\_case\_2003(request)

/articles/2005/03/

views.month\_archive(request, year=2005, month=3)

/articles/2003 ❌

/articles/2003/03/building-a-django-site/

views.article\_detail(request, year=2003, month=3, slug="building-a-django-site")

## re\_path()

`re_path(ruta, vista, kwargs = Ninguno, nombre = Ninguno)`

El argumento **ruta** debe ser una cadena de texto que contenga un **expresión regular** compatible con el módulo **re** de Python.

Símbolo	Coincide con
.	Cualquier carácter
\d	Cualquier dígito
[A-Z]	Cualquier carácter, A-Z (mayúsculas)
[a-z]	Cualquier carácter, a-z (minúsculas)
[A-Za-z]	Cualquier carácter, a-z (no distingue entre mayúscula y minúscula)
+	Una o más ocurrencias de la expresión anterior (ejemplo, \d+ coincidirá con uno o más dígitos)
[^/]+	Todos los caracteres excepto la barra
*	Cero o más ocurrencias de la expresión anterior (ejemplo, \d* coincidirá con cero o más dígitos)
{1,3}	Entre una y tres (inclusive) ocurrencias de la expresión anterior

## re\_path()

`re_path(ruta, vista, kwargs = Ninguno, nombre = Ninguno)`

```
from django.urls import path, re_path

from . import views

urlpatterns = [
    path('articles/2003/', views.special_case_2003),
    re_path(r'^articles/(?P<year>[0-9]{4})/$', views.year_archive),
    re_path(r'^articles/(?P<year>[0-9]{4})/(?P<month>[0-9]{2})/$', views.month_archive),
    re_path(r'^articles/(?P<year>[0-9]{4})/(?P<month>[0-9]{2})/(?P<slug>[w-]+)/$', views.article_detail),
]
```



# Opciones adicionales

```
from django.urls import path
from . import views

urlpatterns = [
    path('blog/<int:year>/', views.year_archive, {'foo': 'bar'}),
]
```

/blog/2005/

`views.year_archive(request, year=2005, foo='bar')`

# Nombrando patrones de URL

```
from django.urls import path

from . import views

urlpatterns = [
    #...
    path('articles/<int:year>/', views.year_archive, name='news-year-archive'),
    #...
]
```

Se usa en  
Templates también

```
from django.http import HttpResponseRedirect
from django.urls import reverse

def redirect_to_year(request):
    # ...
    year = 2006
    # ...
    return HttpResponseRedirect(reverse('news-year-archive', args=(year,)))
```

# Analizando el HttpRequest

## HttpRequest.method

```
if request.method == 'GET':  
    hacer_algo()  
  
elif request.method == 'POST':  
    hacer_otra_cosa()
```

## HttpRequest.GET

Diccionario que contiene todos los parámetros HTTP GET dados

## HttpRequest.POST

Diccionario que contiene todos los parámetros HTTP POST dados, siempre que la solicitud contenga datos de formulario

# Códigos de Respuesta HTTP

```
from django.http import HttpResponse, HttpResponseNotFound
```

```
def my_view(request):
```

```
    # ...
```

```
    if foo:
```

```
        return HttpResponseNotFound('<h1>Page not found</h1>')
```

```
    else:
```

```
        return HttpResponse('<h1>Page was found</h1>')
```

Subclase de HttpResponse

```
raise Http404("La entidad no existe")
```

```
from django.http import HttpResponse
```

```
def my_view(request):
```

```
    # ...
```

```
    # Return a "created" (201) response code.
```

```
    return HttpResponse(status=201)
```

No están todas  
las subclases

# Respuesta con HTML

```
from django.http import HttpResponse

# Create your views here.
def index(request):
    return HttpResponse("<h1 style=\"color:blue\">Hola gente</h1>")
```

```
from django.http import HttpResponse
from django.shortcuts import render

def index(request):
    return render(request, "index.html")
```

```
▼ CAC2022
  > cac2022
  ▼ hola_mundo
    > __pycache__
    > migrations
    ▼ templates
      <> index.html
      📄 __init__.py
      📄 admin.py
      📄 apps.py
      📄 models.py
      📄 tests.py
      📄 urls.py
      📄 views.py
      📄 manage.py
```

**No te olvides de completar la  
asistencia y consultar dudas**

# Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.

**TODO EN EL AULA VIRTUAL**