Django路由Path方法

在《Django路由系统精讲》一节，我们详细了解了 Django 1.x 版本中 url 方法匹配路由的规则，在本节我们将讲解 Django 2.x 版本中 path 方法，希望对大家掌握路由系统的相关知识能够有所帮助。

1.初识path()方法

Django 2.0 可谓是 Django 的里程碑版本，它于 2017 年 12 月正式发布。它移除了对 Python2.7 的支持，最少需要 3.4 以上版本，它增加以一些 Django 1.x 版本不具有的新特性其中就包括，更简单 URL 路由方法。它主要应用于动态路由的定义上，主要变化是新增了 path 函数来进行路由的匹配，可通过以下方式进行导入：

```
#新的2.x版本导入path，导入简化
from django.urls import path
#原来的1.x版本url方式，conf子包
from django.conf.urls import url
```

但是之前的 url 模块并没有废止，只是 Django 强烈建议我们使用新模块 path 进行路由的匹配。从对比可以看出 Django 2.0 简化了路由 path 的导入方法。

1) path()方法函数定义

path 函数在 Django中的的定义如下所示：

path(route,view,kwargs,name)

它可以接收 4 个参数，其中前两个是必填参数后两个为可选参数。参数解析如下：

1. route

route 是一个匹配 URL 的准则（类似正则表达式）。当 Django 响应一个请求时，它会从 urlpatterns 的第一项开始，按顺序依次匹配列表中的项，直到找到匹配的项，然后执行该项映射的视图函数或者 include 函数分发的下级路由，因此，url 路由的编写在 Django中十分的重要！

2. view

view 指的是处理当前 url 请求的视图函数。当 Django 匹配到某个路由条目时，自动将封装的 HttpRequest 对象作为第一个参数，被“捕获”的参数以关键字参数的形式，传递给该条目指定的视图函数。

3. kwargs

kwargs 指使用字典关键字传参的形式给关联的目标视图函数传递参数。

4. name

name 给 URL 起个别名，常用于 url 的反向解析，避免在模板中适应硬编码的方式使用嵌入 url，在后续章节会进行详细讲解。

当使用 path 方法关联视图函数时与 url 方法相比更为简化，也更容易让初学者理解，path 方法引入了类型转化器（converter type）的概念，以此省去了较为复杂的正则表达式匹配路由的方法，实例说明如下：

```
#1.x url方法
url(r'^test/(?P<year>[0-9]{4})/$', views.year_test),
#2.x path方法
path('test/<int:year>/', views.year_test),
```

int 支持整数类型的转化，在上述的例子中， year_test 函数接收到的 year 参数就变成整数而不是字符串，从而避免在视图中使用 year=int(year)。

path 函数定义的 <int:year> 规则会捕获到 URL 中的值，映射给视图中的同名参数 year ，并根据转换器将参数值转换为指定的类型，这里对应 int 大于等于 0 的整数。之所以使用转化器，有以下两个原因：

- 第一是可以将捕获到的字符值转换为对应的类型；
- 第二是对 URL 中传值的一种限制，避免视图处理出错

在使用 url 函数时候，我们遇上这样一种情景：即不同的视图函数使用相同的字段作为参数，那么 url 函数也会使用相同正则表达式，只是它们关联的视图函数不同，但是当这个被关联的字段更改的是后，那么可想而知，我们也需要修改所有的正则表达式，重新匹配它，举例如下：

```
#views.py 视图函数
def year_test(request, year):
    year = int(year) # 转换整形
def num1_view(request, id):
    pass
def num2_view(request, id):
    pass
def num3_view(request, id):
    pass
```

urls.py 中配置路由如下所示：

```
from django.conf.urls import url #引入url方法
urlpatterns = [
    url('test/(?P<year>[0-9]{4})/', year_tst),
    url('test/(?P<id>[a-zA-Z0-9]+)/num/', num_view),
```

```
    url('test/(?P<id>[a-zA-Z0-9]+)/num1/', num1_view),
    url('test/(?P<id>[a-zA-Z0-9]+)/num2/', num2_view),
]
```

利用 path 方法中提供的类型转换器就很好的解决了这一问题。

2. path方法类型转化器

Django 默认支持 5 个类型转换器，在大多数情况下，绝对可以满足我们的正常业务需求，如果不能，Django 同样提供了自定义转换器。下面介绍 Django 默认支持的转换器，如下所示：

- str，匹配除了路径分隔符（/）之外的非空字符串，这是默认的形式；
- int，匹配正整数，包含0；
- slug，匹配字母、数字以及横杠、下划线组成的字符串；
- uuid，匹配格式化的 uuid，如 075194d3-6885-417e-a8a8-6c931e272f00；
- path，匹配任何非空字符串，包含了路径分隔符。

3. re_path正则表达式匹配

如果上述的 paths 和 converters 还是无法满足需求，Django 2.x 也支持我们使用正则表达式来捕获值，在这里需要使用 re_path()，而不是前面介绍的 path()。我们使用带命名的正则表达式分组，语法如下：

(?P<name>pattern)

其中，尖括号里的name为分组名，pattern为正则表达式。re_path()同样包含于django.urls模块中，所以同样使用如下方式进行导入。

示例如下：

```
from django.urls import path, re_path  #导入re_path
from . import views
urlpatterns = [
    re_path('test/(?P<year>[0-9]{4})/', views.year_test),
    re_path('test/(?P<year>[0-9]{4})/(?P<month>[0-9]{2})/', views.month_test),
    re_path('test/(?P<year>[0-9]{4})/(?P<month>[0-9]{2})/(?P<slug>[^/]+)/', views.article_test),
```

re_path 其实相当于 Django 1.x 中的 url 方法。它们两的用法是一致的，所以在这里就不多加赘述了。

4. 总结归纳

Django 2.0 和之前相比多了变量类型转化这一步骤。目前路由（url）到视图（View）的流程可以大致分为四个步骤：

- url 匹配
- 正则表达式捕获
- 变量类型转化
- 视图函数调用

新增的 path 方法可以帮助我们解决以下几个问题：

- 类型自动转化问题，可以使用类型转换器完成；
- 公用正则表达式，牵一发而动全身，使用类型转换器也可以规避这个问题。

分布式路由使用到的 include 函数同样使用 from django.urls import include 来引入