

Django基本功

URL调度器

View 视图

Model 模型

Template 模板

底层如何实现

URL调度器的path和自定义函数的区别在哪里

View视图函数怎样只能支持类方法

Model模型怎么动态创建类

Template模板怎样实现内置类型自带的方法

URL调度器

```
path("", views.index),
```

```
path = partial(_path, Pattern=RoutePattern)
```

偏函数partial()

偏函数是functools模块的功能
通过固定参数，降低函数调用的难点
比函数的默认参数简单、而且灵活

偏函数partial()

```
from functools import partial
```

```
# 二进制 111 转成十进制
```

```
# >>> int('111',base=2)
```

```
base2 = partial(int, base=2)
```

```
print(base2('111'))
```

偏函数partial()

```
def partial(func, *args, **keywords):  
    def newfunc(*fargs, **fkeywords):  
        newkeywords = keywords.copy()  
        newkeywords.update(fkeywords)  
        return func(*args, *fargs, **newkeywords)
```

```
newfunc.func = func  
newfunc.args = args  
newfunc.keywords = keywords  
return newfunc
```

view视图

```
frameworks > Python.framework > Versions > 3.7 > lib > python3.7 > site-packages > django > views > generic > base.py > View

29
30 class View:
31     """
32     Intentionally simple parent class for all views. Only implements
33     dispatch-by-method and simple sanity checking.
34     """
35
36     http_method_names = ['get', 'post', 'put', 'patch', 'delete', 'head', 'options', 't
37
38 > def __init__(self, **kwargs): ...
47
48 @classmethod
49 def as_view(cls, **initkwargs):
50     """Main entry point for a request-response process."""
51     for key in initkwargs:
52         if key in cls.http_method_names:
53             raise TypeError("You tried to pass in the %s method name as a "
54                             "keyword argument to %s(). Don't do that."
55                             % (key, cls.__name__))
56         if not hasattr(cls, key):
57             raise TypeError("%s() received an invalid keyword %r. as_view "
58                             "only accepts arguments that are already "
59                             "attributes of the class." % (cls.__name__, key))
60
```

site-packages/django/views/generic/base.py

非数据描述符

site-packages/django/utils/decorators.py

```
class classonlymethod(classmethod):
    def __get__(self, instance, cls=None):
        if instance is not None:
            raise AttributeError("This method is available
only on the class, not on instances.")
        return super().__get__(instance, cls)
```


ORM

site-packages/django/db/models/base.py

```
class ModelBase(type):  
    def __new__(cls, name, bases, attrs, **kwargs):  
        super_new = super().__new__  
        ...  
        return super_new(cls, name, bases, attrs)  
        new_class = super_new(cls, name, bases, new_attrs,  
**kwargs)
```

Template

```
class Template:
    def __init__(self, template_string, origin=None,
name=None, engine=None):

... ..
def __iter__(self):
    for node in self.nodelist:
        yield from node
```

THANKS! |  极客大学