Ask Company

리액트와 함께 장고 시작하기 / 장고 Views

장고 기본 CBV API (Base views)

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

Built-in CBV API

Base views

View, TemplateView, RedirectView

Generic display views

DetailView, ListView

Generic date views

ArchiveIndexView, YearArchiveView, MonthArchiveView, WeekArchiveView, DayArchiveView, TodayArchiveView, DateDetailView

Generic editing views

FormView, CreateView, UpdateView, DeleteView

https://docs.djangoproject.com/en/2.1/ref/class-based-views/

Base Views

django/views/generic/base.py

View

TemplateView

- ← TemplateResponseMixin
- ← ContextMixin
- ← View

RedirectView

- ← View

View

모든 CBV의 모체

이 CBV를 직접 쓸 일은 거의 X

http method별로 지정 이름의 멤버함수를 호출토록 구현

CBV.as_view(**initkwargs)

initkwargs인자는 그대로 CBV 생성자로 전달

```
def __init__(self, **kwargs):
    for key, value in kwargs.items():
        setattr(self, key, value)
```

```
class View:
    def __init__(self, **kwargs):
        # ...

    @classonlymethod
    def as_view(cls, **initkwargs):
        # ...
        def view(request, *args, **kwargs):
            # ...
            return self.dispatch(request, *args, **kwargs)
        # ...
        return view

    def dispatch(self, request, *args, **kwargs):
        # ...
        # request.method.lower() 이름의 멤버함수를 호출
        handler = getattr(self, request.method.lower(),
                          self.http_method_not_allowed)
        return handler(request, *args, **kwargs)

    def http_method_not_allowed(self, request, *args, **kwargs):
        # ...
        return HttpResponseNotAllowed(self._allowed_methods())

    def options(self, request, *args, **kwargs):
        # ...
        return response

    def _allowed_methods(self):
        return [m.upper() for m in self.http_method_names
                if hasattr(self, m)]
```

https://github.com/django/django/blob/2.1/django/views/generic/base.py#L49

TemplateView

```python
class ContextMixin:
    extra_context = None

    def get_context_data(self, **kwargs):
        kwargs.setdefault('view', self)
        if self.extra_context is not None:
            kwargs.update(self.extra_context)
        return kwargs
```

```python
class TemplateResponseMixin:
    template_name = None
    template_engine = None
    response_class = TemplateResponse
    content_type = None

    def render_to_response(self, context, **response_kwargs):
        response_kwargs.setdefault('content_type', self.content_type)
        return self.response_class(
            request=self.request,
            template=self.get_template_names(),
            context=context,
            using=self.template_engine,
            **response_kwargs)

    def get_template_names(self):
        if self.template_name is None:
            raise ImproperlyConfigured(
                "TemplateResponseMixin requires either a definition of "
                "'template_name' or an implementation of 'get_template_names()'")
        else:
            return [self.template_name]
```

```python
class TemplateView(TemplateResponseMixin, ContextMixin, View):
    def get(self, request, *args, **kwargs):
        context = self.get_context_data(**kwargs)
        return self.render_to_response(context)
```

RedirectView

permanent (디폴트: False)

True : 301 응답 (영구적인 이동) - 검색엔진에 영향

False : 302 응답 (임시 이동)

url = None

URL 문자열

pattern_name = None

URL Reverse를 수행할 문자열

query_string = False

QueryString을 그대로 넘길 것인지 여부

```python
class RedirectView(View):
    permanent = False
    url = None
    pattern_name = None
    query_string = False

    def get_redirect_url(self, *args, **kwargs):
        if self.url:
            url = self.url % kwargs
        elif self.pattern_name:
            url = reverse(self.pattern_name, args=args, kwargs=kwargs)
        else:
            return None

        args = self.request.META.get('QUERY_STRING', '')
        if args and self.query_string:
            url = "%s?%s" % (url, args)
        return url

    def get(self, request, *args, **kwargs):
        url = self.get_redirect_url(*args, **kwargs)
        if url:
            if self.permanent:
                return HttpResponsePermanentRedirect(url)
            else:
                return HttpResponseRedirect(url)
        else:
            logger.warning(
                'Gone: %s', request.path,
                extra={'status_code': 410, 'request': request}
            )
            return HttpResponseGone()

    # head, post, options, delete, put, patch 모두 같은 구현
    def head(self, request, *args, **kwargs):
        return self.get(request, *args, **kwargs)
```

인생은 짧습니다.
파이썬/장고를 쓰세요.

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

-

Ask Company