Python

базовый тренинг

Рузин Алексей
ruzin@me.com

Django - Views

- urls.py

```
urlpatterns = [
    url(r’^view/$', views.view, name='view'),
]
```

- views.py

```
def view(request):
   return HttpResponse(‘html’)
```

Django - Views

- urls.py

```
urlpatterns = [
    url(r’^view/(?<param>.*)/$’, views.view,
name='view'),
]
```

- views.py

```
def view(request, param):
   return HttpResponse(‘html’)
```

Django - Response

- def view(request, **param**):
  return …

- HttpResponse(‘data’, content_type=None, status=None, charset=None)

- HttpResponseNotFound() # …BadRequest,

- HttpResponseRedirect(reverse(‘app:url_name’, params))

Django - Templates

- <app>/templates/<app>/template_name.html

-
  ```
  from django.template import loader

  template = loader.get_template(‘<app>/template_name.html’)
  …
  return HttpResponse(template.render(context, request))
  ```

- render(request, ‘template.html’, context)

Django - Templates

-
  ```
  <html>
  …
  {{ data }}
  {{ my_list.0 }} {{ my_list_of_dicts.0.key }}
  {{ data.keys|sort }}  {{ today|date:«Y-m-d» }}

  {% url «shop:url_name» param %}
  {% if data == 1 %} … {% else %} … {% endif %}
  {% for variable in data %} … {{ variable }} … {% endfor %}
  ```

Django - Template tags

-
  ```
  {% load humanize %}
  {{ 45000|intcomma }}
  ```

- templatetags/custom_tags.py

  ```
  from django import template
  register = template.Library()
  @register.filter(is_safe=True)
  def do_something(value, param=' '):
      return …
  ```

-
  ```
  {% load custom_tags %}
  …
  {{ some_variable|do_something:«XYZ» }}
  ```

Django - Template blocks

- templates/base.html

```
<html>
…
{% block my_name %}
… default html …
{% endblock %}
```

- templates/xyz.html

```
{% extends «base.html» %}
{% block my_name %}
… override default html …
{% endblock %}
```

- {% include "subtemplate.html" with param=value %}

Django - generic Views

- urls.py

  ```
  urlpatterns = [
      url(r’^item/(?<pk>.*)/$’, views.ItemView.as_view(), name='item'),
  ]
  ```

- views.py

  ```
  class ItemView(generic.DetailView):
      model = Item
  class ItemListView(generic.ListView):
      model = Item
      context_object_name = 'categories'
  ```

- templates/item_detail.html
  templates/item_list.html

Django - Class Views

-
  ```
  class ItemView(django.views.View):
      def get(self, request):
          …
      def post(self, request):
          …
      def head(self, request):
          …
      def delete(self, request):
          …
      def put(self, request):
          …
  ```

Django - Forms

-
  ```
  from django.forms import ModelForm
  from shop.models import Item
  class ItemForm(ModelForm):
      class Meta:
          model = Item
          fields = ['title', 'price', 'category']
  class ItemFormView(View):
      template_name = 'shop/item_detail.html'
      def get(self, request, pk, *args, **kwargs):
          item = Item.objects.get(pk=pk)
          form = ItemForm(instance=item)
          return render(request, self.template_name, {'form': form, 'item': item})
      def post(self, request, pk, *args, **kwargs):
          item = Item.objects.get(pk=pk)
          form = ItemForm(request.POST)
          if form.is_valid():
              item.title = form.instance.title
              item.price = form.instance.price
              item.category = form.instance.category
              item.save()
          return render(request, self.template_name, {'form': form, 'item': item})
  ```