This document is coupled with the Django presentation. Since outputting code in Power Point does not look good, the presentation includes many image screens from the Django tutorial. In order to support easy copy & paste of code to be used in the interactive lessons, this document provides the necessary code for each relevant slide.

**Slide #7**:

To define the Python path properly on *cmd*, run the following commands: (assuming Python is installed on C:\Python34\).

set PYTHONPATH=%PYTHONPATH%;C:\Python34\;C:\Python34\Scripts\

set PATH=%PATH%;C:\Python34\;C:\Python34\Scripts\

To check whether the path was defined run **echo %PATH%** and look for the C:\Python34\;C:\Python34\Scripts\ directories.

Now check if you can run the Python interpreter by running the command **python** from any directory. Exit the interpreter using *exit()*.

To install Python we will use pip, which is located at the Scripts\ directory. To check if pip is accessible run ***pip help*** which will show its menu and tell us that it is accessible.

**Slide #19**:

from django.db import models

class Question(models.Model):

question\_text = models.CharField(max\_length=200)

pub\_date = models.DateTimeField('date published')

class Choice(models.Model):

question = models.ForeignKey(Question)

choice\_text = models.CharField(max\_length=200)

votes = models.IntegerField(default=0)

**Slide #28**:

from polls.models import Question, Choice

Question.objects.all()

from django.utils import timezone

q = Question(question\_text="What's new?", pub\_date=timezone.now())

q.save()

q.id

**Slide #29**:

q.question\_text

q.pub\_date

q.question\_text = "What's up?"

q.save()

Question.objects.all()

**Slide #30**:

from django.db import models

class Question(models.Model):

# ...

def \_\_str\_\_(self):

return self.question\_text

class Choice(models.Model):

# ...

def \_\_str\_\_(self):

return self.choice\_text

**Slide #31**:

import datetime

from django.db import models

from django.utils import timezone

class Question(models.Model):

# ...

def was\_published\_recently(self):

return self.pub\_date >= timezone.now() - datetime.timedelta(days=1)

**Slide #32**:

from polls.models import Question, Choice

Question.objects.all()

Question.objects.filter(id=1)

Question.objects.filter(question\_text\_\_startswith='What')

from django.utils import timezone

current\_year = timezone.now().year

Question.objects.get(pub\_date\_\_year=current\_year)

Question.objects.get(id=2)

**Slide #33**:

q = Question.objects.get(pk=1)

q.was\_published\_recently()

q = Question.objects.get(pk=1)

q.choice\_set.all()

**Slide #34**:

q.choice\_set.create(choice\_text='Not much', votes=0)

q.choice\_set.create(choice\_text='The sky', votes=0)

c = q.choice\_set.create(choice\_text='Just hacking again', votes=0)

c.question

q.choice\_set.all()

q.choice\_set.count()

Choice.objects.filter(question\_\_pub\_date\_\_year=current\_year)

c = q.choice\_set.filter(choice\_text\_\_startswith='Just hacking')

c.delete()

**Slide #37**:

from django.contrib import admin

from .models import Question

admin.site.register(Question)

**Slide #43**:

class ChoiceInline(admin.StackedInline):

model = Choice

extra = 3

class QuestionAdmin(admin.ModelAdmin):

inlines = [ChoiceInline]

admin.site.register(Question, QuestionAdmin)

**Slide #50**:

**Views.py**:

from django.http import HttpResponse

def index(request):

return HttpResponse("Hello, world. You're at the polls index.")

**urls.py**:

from django.conf.urls import url

from . import views

urlpatterns = [

url(r'^$', views.index, name='index'),

]

**Slide #54**:

def detail(request, question\_id):

return HttpResponse("You're looking at question %s." % question\_id)

def results(request, question\_id):

response = "You're looking at the results of question %s."

return HttpResponse(response % question\_id)

def vote(request, question\_id):

return HttpResponse("You're voting on question %s." % question\_id)

**Slide #55**:

from django.conf.urls import url

from . import views

urlpatterns = [

# ex: /polls/

url(r'^$', views.index, name='index'),

# ex: /polls/5/

url(r'^(?P<question\_id>[0-9]+)/$', views.detail, name='detail'),

# ex: /polls/5/results/

url(r'^(?P<question\_id>[0-9]+)/results/$', views.results, name='results'),

# ex: /polls/5/vote/

url(r'^(?P<question\_id>[0-9]+)/vote/$', views.vote, name='vote'),

]

**Slide #59**:

from .models import Question

def index(request):

latest\_question\_list = Question.objects.order\_by('-pub\_date')[:5]

output = ', '.join([p.question\_text for p in latest\_question\_list])

return HttpResponse(output)

**Slide #62**:

{% if latest\_question\_list %}

<ul>

{% for question in latest\_question\_list %}

<li><a href="/polls/{{ question.id }}/">{{ question.question\_text }}</a></li>

{% endfor %}

</ul>

{% else %}

<p>No polls are available.</p>

{% endif %}

**Slide #63**:

from django.http import HttpResponse

from django.template import RequestContext, loader

from .models import Question

def index(request):

latest\_question\_list = Question.objects.order\_by('-pub\_date')[:5]

template = loader.get\_template('polls/index.html')

context = RequestContext(request, {

'latest\_question\_list': latest\_question\_list,

})

return HttpResponse(template.render(context))

**Slide #64**:

from django.shortcuts import render

from .models import Question

def index(request):

latest\_question\_list = Question.objects.order\_by('-pub\_date')[:5]

context = {'latest\_question\_list': latest\_question\_list}

return render(request, 'polls/index.html', context)

**Slide #65**:

<h1>{{ question.question\_text }}</h1>

<ul>

{% for choice in question.choice\_set.all %}

<li>{{ choice.choice\_text }}</li>

{% endfor %}

</ul>

**Slide #66**:

from django.http import Http404

from django.shortcuts import render

from .models import Question

# ...

def detail(request, question\_id):

try:

question = Question.objects.get(pk=question\_id)

except Question.DoesNotExist:

raise Http404("Question does not exist")

return render(request, 'polls/detail.html', {'question': question})

**Slide #67**:

from django.shortcuts import get\_object\_or\_404, render

from .models import Question

# ...

def detail(request, question\_id):

question = get\_object\_or\_404(Question, pk=question\_id)

return render(request, 'polls/detail.html', {'question': question})

**Slide #68**:

<li><a href="{% url 'detail' question.id %}">{{ question.question\_text }}</a></li>

**Slide #70**:

**urls.py**:

url(r'^polls/', include('polls.urls', namespace="polls")),

**index.html**:

<li><a href="{% url 'polls:detail' question.id %}">{{ question.question\_text }}</a></li>

**Slide #72**:

<h1>{{ question.question\_text }}</h1>

{% if error\_message %}<p><strong>{{ error\_message }}</strong></p>{% endif %}

<form action="{% url 'polls:vote' question.id %}" method="post">

{% csrf\_token %}

{% for choice in question.choice\_set.all %}

<input type="radio" name="choice" id="choice{{ forloop.counter }}" value="{{ choice.id }}" />

<label for="choice{{ forloop.counter }}">{{ choice.choice\_text }}</label><br />

{% endfor %}

<input type="submit" value="Vote" />

</form>

**Slide #74**:

from django.shortcuts import get\_object\_or\_404, render

from django.http import HttpResponseRedirect, HttpResponse

from django.core.urlresolvers import reverse

from .models import Choice, Question

# ...

def vote(request, question\_id):

p = get\_object\_or\_404(Question, pk=question\_id)

try:

selected\_choice = p.choice\_set.get(pk=request.POST['choice'])

except (KeyError, Choice.DoesNotExist):

# Redisplay the question voting form.

return render(request, 'polls/detail.html', {

'question': p,

'error\_message': "You didn't select a choice.",

})

else:

selected\_choice.votes += 1

selected\_choice.save()

# Always return an HttpResponseRedirect after successfully dealing

# with POST data. This prevents data from being posted twice if a

# user hits the Back button.

return HttpResponseRedirect(reverse('polls:results', args=(p.id,)))

**Slide #76**:

**views.py**:

from django.shortcuts import get\_object\_or\_404, render

def results(request, question\_id):

question = get\_object\_or\_404(Question, pk=question\_id)

return render(request, 'polls/results.html', {'question': question})

**results.html**:

<h1>{{ question.question\_text }}</h1>

<ul>

{% for choice in question.choice\_set.all %}

<li>{{ choice.choice\_text }} -- {{ choice.votes }} vote{{ choice.votes|pluralize }}</li>

{% endfor %}

</ul>

<a href="{% url 'polls:detail' question.id %}">Vote again?</a>

**Slide #78**:

from django.conf.urls import url

from . import views

urlpatterns = [

url(r'^$', views.IndexView.as\_view(), name='index'),

url(r'^(?P<pk>[0-9]+)/$', views.DetailView.as\_view(), name='detail'),

url(r'^(?P<pk>[0-9]+)/results/$', views.ResultsView.as\_view(), name='results'),

url(r'^(?P<question\_id>[0-9]+)/vote/$', views.vote, name='vote'),

]

**Slide #79**:

from django.views import generic

class IndexView(generic.ListView):

template\_name = 'polls/index.html'

context\_object\_name = 'latest\_question\_list'

def get\_queryset(self):

"""Return the last five published questions."""

return Question.objects.order\_by('-pub\_date')[:5]

class DetailView(generic.DetailView):

model = Question

template\_name = 'polls/detail.html'

class ResultsView(generic.DetailView):

model = Question

template\_name = 'polls/results.html'

**Slide #87**:

<**html**>  
{% load staticfiles %}  
<**link rel="stylesheet" type="text/css" href="{% static 'polls/style.css' %}"** />  
  
<**head**>  
 <**title**>  
 {% block title %} Polls App {% endblock %}  
 </**title**>  
</**head**>  
  
<**body**>  
  
<**div id="header"**>  
 {% block header %}  
 <**h1**>This is the coolest voting app!</**h1**>  
 {% endblock %}  
</**div**>  
<**div id="content"**>  
 {% block content %}  
 {% endblock %}  
</**div**>  
<**div id="footer"**>  
 {% block footer %}  
 <**h3**>Caution: Excessive voting can cause severe brain damage!</**h3**>  
 {% endblock %}  
</**div**>  
  
</**body**>  
</**html**>