

前端模板

adminLTE:

- <https://adminlte.io/themes/v3/index.html>
- <https://adminlte.io/themes/AdminLTE/index2.html>

我们copy的是static/AdminLTE-master/starter.index
修改静态文件的引用方式

modelform

it表

法1

```
# 来自张子俊

from django.forms import ModelForm
from django import forms
from django.forms import widgets as wid
from app01 import models

class ItModelForm(ModelForm):
    class Meta:
        model = models.It
        fields = "__all__"
        bootstrapClass_filter = ['it_start_time', 'it_end_time']
        it_start_time = forms.DateField(label="开始时间", widget=wid.DateInput(attrs=
{"class": "form-control", 'type': "date"}))
        it_end_time = forms.DateField(label="结束时间", widget=wid.DateInput(attrs=
{"class": "form-control", 'type': "date"}))
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        for name, field in self.fields.items():
            if name in self.bootstrapClass_filter:
                continue
            old_class = field.widget.attrs.get('class', "")
            field.widget.attrs['class'] = '{} form-control'.format(old_class)
            field.widget.attrs['placeholder'] = '请输入%s' % (field.label,)
```

法2

```
from django.forms import ModelForm
from django import forms
from django.forms import widgets as wid
from app01 import models

class ItModelForm(ModelForm):
    class Meta:
        model = models.It
```

```

fields = "__all__"
# 法2
labels = {
    "it_name": "项目名称",
    "it_desc": "项目描述",
    "it_start_tile": "项目开始时间",
    "it_end_tile": "项目结束时间",
}
error_messages = {
    "it_name": {"required": "不能为空"},
    "it_desc": {"required": "不能为空"},
    "it_start_tile": {"required": "不能为空"},
    "it_end_tile": {"required": "不能为空"},
}
widgets = {
    "it_name": wid.Input(attrs={"class": "form-control", "placeholder":
"输入项目名称"}),
    "it_desc": wid.Textarea(attrs={"class": "form-control",
"placeholder": "输入项目名称"}),
    "it_start_time": wid.DateInput(attrs={"class": "form-control",
'type': "date"}),
    "it_end_time": wid.DateInput(attrs={"class": "form-control", 'type':
"date"}),
}

```

模板语言

```

<!-- 切片， 参数必须是 str -->
<td>{{ foo.api_url | slice:"10" }}</td>
<!-- 截取指定长度字符，后续以点代替， 参数必须是int -->
<td title="{{ foo.api_url }}">{{ foo.api_url | truncatechars:10 }}</td>

```

django的下载逻辑

```

from django.http import FileResponse
from django.http import StreamingHttpResponse

```

参考: <https://www.cnblogs.com/Neeo/articles/11021972.html>

关于复选框的操作

获取所有的选中状态的复选框:

```

$("#chk1").find('input:checkbox').each(function() { //遍历所有复选框

    if ($(this).prop('checked') == true) {

        console.log($(this).val()); //打印当前选中的复选框的值
    }
}

```

```

    }

});

function getCheckBoxVal(){ //jquery获取所有选中的复选框的值

    var chk_value =[];

    $("#chk1").find('input[name="test"]:checked').each(function(){ //遍历，将所有
    选中的值放到数组中

        chk_value.push($(this).val());

    });

    alert(chk_value.length==0 ?'你还没有选择任何内容!':chk_value);

}

```

或者：

```

$("#sure").click(function () {
    var arr = new Array();
    $.each($(".p1"), function (index, item) {
        // console.log(index, item)
        if ($(item).get(0).checked) {
            arr.push($(item).val())
        }
    });
    if (arr.length == 0) {
        // 说明用户未选中用例，需要给提示
        // console.log(2222222, "未选中", arr);
        $("#errorMsg").html("请勾选至少一个用例！");

    } else {
        // 编写后续的操作
    }
});

```

ajax如何处理跨域问题？

有以下几种办法：

- 装饰器

```

# views中
from django.views.decorators.csrf import csrf_exempt

@csrf_exempt
def run_case(request, pk=0):
    pass

```

其他形式参考：<https://www.cnblogs.com/Neeo/articles/11455271.html>

前端序列化与反序列化

```
// 序列化
JSON.stringify(['A', 'B'])
// 反序列化
JSON.parse()
```

io

在内存中创建一个文件句柄

```
from io import BytesIO
from io import StringIO

f = open('a.html', 'wb') # BytesIO
f1 = open('a.html', 'w', encoding='utf-8') # StringIO
```

可视化

- hightcharts
- echarts

前端从后端获取必要的的数据，进行渲染展示

- pyecharts： 百度开源的，后台自己处理数据，自己生成html页面

pyecharts的使用

```
pip install wxpy # 备用下载地址: pip install -i
https://pypi.tuna.tsinghua.edu.cn/simple wxpy
pip install pyecharts==0.5.11 # 备用下载地址: pip install -i
https://pypi.tuna.tsinghua.edu.cn/simple pyecharts==0.5.11
pip install pyecharts_snapshot # 备用下载地址: pip install -i
https://pypi.tuna.tsinghua.edu.cn/simple pyecharts_snapshot
```

示例：

```
import wxpy
import webbrowser
from pyecharts import Pie
# 1. 登录
bot = wxpy.Bot() # cache_path=True 可以不填，但每次都要重新扫描
# 2. 获取所有的朋友对象，放到列表中
friends = bot.friends()
attr = ['男朋友', '女朋友', '性别不详']
value = [0, 0, 0]
for friend in friends:
```

```

print(friend.name, friend.sex)
if friend.sex == 1:      # 1代表男性
    value[0] += 1
elif friend.sex == 2:    # 2代表女性
    value[1] += 1
else:      # 未指定性别的
    value[2] += 1
# 3. 处理为图像
pie = Pie('%s 朋友圈性别比例图' % bot.self.name)
pie.add('', attr, value, is_label_show=True) # 图表名称str, 属性名称list, 属性所对应的值list, is_label_show是否显示标签
pie.render('sex.html') # 生成html页面
# 4. 打开浏览器展示
webbrowser.open('sex.html')

```

示例参考:

<https://www.cnblogs.com/Neeo/articles/10454764.html>

<https://www.cnblogs.com/Neeo/articles/10454677.html>

echarts的使用

参考: <https://echarts.apache.org/zh/index.html>

使用:

1. 绑定一个标签, 生成一个echarts对象
2. 配置数据和参数
3. setoption生成图表

定时任务

常见用于定时任务的:

- crontab
- django的定时任务
- [APScheduler](#)
- celery: <http://docs.jinkan.org/docs/celery/getting-started/introduction.html>

APScheduler

```
pip install -i https://pypi.doubanio.com/simple/ apscheduler
```

使用

指定时间执行一次

```
import datetime
from apscheduler.schedulers.blocking import BlockingScheduler
def job2(text):
    print('job2', datetime.datetime.now(), text)
scheduler = BlockingScheduler()
scheduler.add_job(job2, 'date', run_date=datetime.datetime(2019, 2, 25, 19, 5, 6), args=['text'], id='job2')
scheduler.start()
```

每天指定时间执行一次

```
# ----- 每天指定时间 执行一次 -----

from apscheduler.schedulers.blocking import BlockingScheduler # 后台运行

sc = BlockingScheduler()
f = open('t1.txt', 'a', encoding='utf8')

@sc.scheduled_job('cron', day_of_week='*', hour=11, minute='56', second='2') #
    每天11点56分02秒执行一次
def check_db():
    print(111111111111)

if __name__ == '__main__':
    try:
        sc.start()
        f.write('定时任务成功执行')
    except Exception as e:
        sc.shutdown()
        f.write('定时任务执行失败')
    finally:
        f.close()
```

django发邮件

参考: <https://www.cnblogs.com/Neeo/articles/11199085.html>

简单配置发邮件

settings.py配置:

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_USE_TLS = True # 是否使用TLS安全传输协议(用于在两个通信应用程序之间提供保密性和数据完整性。)
EMAIL_USE_SSL = False # 是否使用SSL加密, qq企业邮箱要求使用
EMAIL_HOST = 'smtp.163.com' # 发送邮件的邮箱的SMTP服务器, 这里用了163邮箱
EMAIL_PORT = 25 # 发件箱的SMTP服务器端口

# 上面配置可以不动, 下面配置可以修改
EMAIL_HOST_USER = '邮箱@163.com' # 发送邮件的邮箱地址
EMAIL_HOST_PASSWORD = '你的授权码' # 发送邮件的邮箱密码(这里使用的是授权码)
EMAIL_TO_USER_LIST = ['接收人@qq.com'] # 此字段是可选的, 用来配置收件人列表
```

views.py配置：

```
from django.http import HttpResponse
from dengxin import settings
from django.core.mail import send_mail

def send_email(request):
    send_mail(
        subject='这里是邮件标题',
        message='这里是邮件内容',
        from_email='tingyuweilou@163.com',
        recipient_list=settings.EMAIL_TO_USER_LIST,
        fail_silently=False
    )
    return HttpResponse('OK')
```

发送待附件的示例

views.py中：

```
from django.http import HttpResponse
from dengxin import settings
from django.core.mail import send_mail, EmailMessage

def send_email(request):

    # 发送带附件的邮件
    msg = EmailMessage(
        subject='这是带附件的邮件标题',
        body='这是带附件的邮件内容',
        from_email=settings.EMAIL_HOST_USER, # 也可以从settings中获取
        to=settings.EMAIL_TO_USER_LIST
    )
    msg.attach_file(r'D:\video\s28-testing-day16-接口自动化平台-实现-3\note\dengxin\sex.html')
    msg.send(fail_silently=False)
    return HttpResponse('OK')
```

可以把发邮件功能和定时任务结合起来。

文件上传

有两种常用的文件上传方式：

- ajax上传
- form表单上传

ajax上传

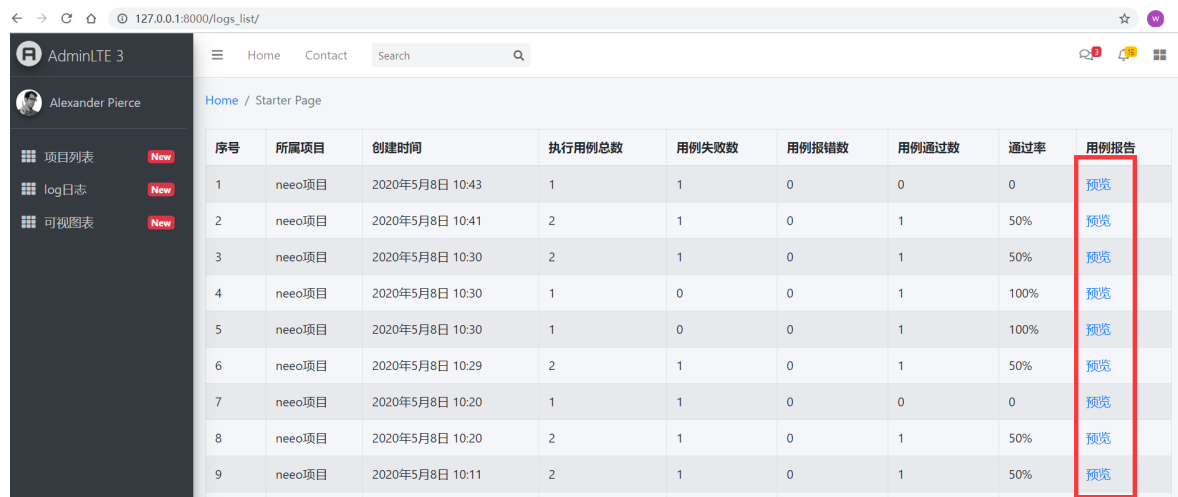
参考：<https://www.cnblogs.com/Neeo/articles/11021972.html>

周末作业

日志表相关的作业

把预览完成

- 点击预览，跳转到一个新的页面中，在新的页面中，有返回上一页和下载按钮
 - 点击返回上一页，跳转到logs_list页面
 - 点击下载，就把当前的报告下载到本地

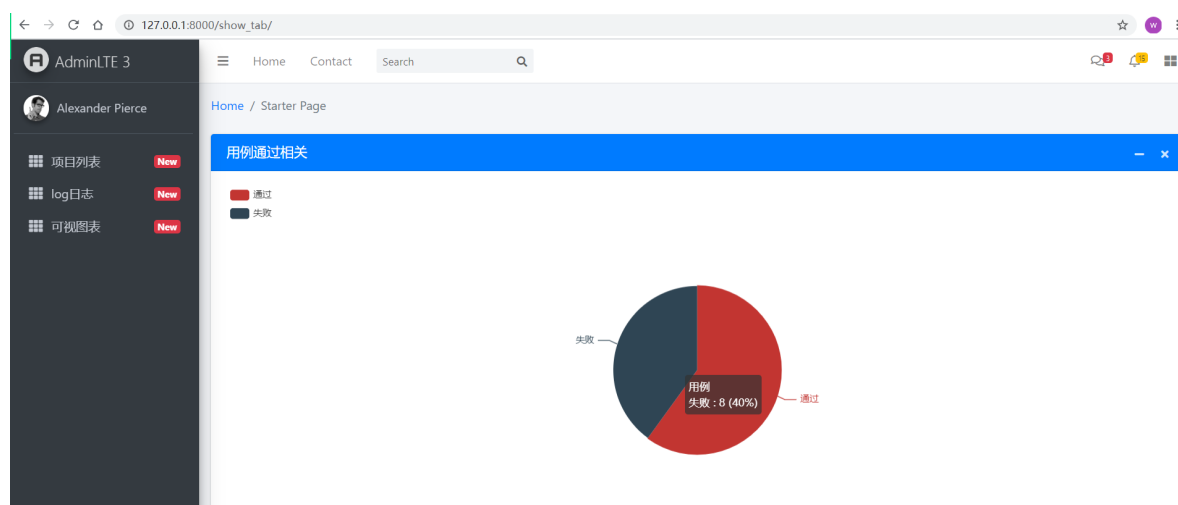


序号	所属项目	创建时间	执行用例总数	用例失败数	用例报错数	用例通过数	通过率	用例报告
1	neeo项目	2020年5月8日 10:43	1	1	0	0	0	预览
2	neeo项目	2020年5月8日 10:41	2	1	0	1	50%	预览
3	neeo项目	2020年5月8日 10:30	2	1	0	1	50%	预览
4	neeo项目	2020年5月8日 10:30	1	0	0	1	100%	预览
5	neeo项目	2020年5月8日 10:30	1	0	0	1	100%	预览
6	neeo项目	2020年5月8日 10:29	2	1	0	1	50%	预览
7	neeo项目	2020年5月8日 10:20	1	1	0	0	0	预览
8	neeo项目	2020年5月8日 10:20	2	1	0	1	50%	预览
9	neeo项目	2020年5月8日 10:11	2	1	0	1	50%	预览

可视化

使用echarts展示三个图：

- 用例通过/失败（饼图）
- 用例执行/未执行（饼图）
- 展示最近一年的每个月的项目创建数量（折线图），注意，提前准备数据



定时任务

每天凌晨1点30分50秒，检查it表，it_end_time是当前日期的接口项目，提取出来，获取该项目下所有的用例，批量执行一次，并生成log日志。

扩展：使用django发邮件，附件是批量执行的报告

建议：

多线程执行该任务