1、JS 正则

test - 判断字符串是否符合规定的正则

```
rep = /\d+/;
rep.test("asdfoiklfasdf89asdfasdf")
# true

rep = /^\d+$/;
rep.test("asdfoiklfasdf89asdfasdf")
# true
```

exec - 获取匹配的数据

```
rep = /\d+/;
str = "wangshen_67_houyafa_20"
rep.exec(str)
# ["67"]

JavaScript is more fun than Java or JavaBeans!
var pattern = /\bJava(\w*)\b/;
# ["JavaScript", "Script"]


JavaScript is more fun than Java or JavaBeans!
var pattern = /\bJava\w*\b/g;
# ["JavaScript"]
# ["Java"]
# ["JavaBeans"]
# null

JavaScript is more fun than Java or JavaBeans!
var pattern = /\bJava(\w*)\b/g;
# ["JavaScript",'Script']
# ["Java", ""]
# ["JavaBeans", "Beans"]
# null
```

多行匹配:

```
默认就是多行匹配
^$
```

- 登录注册验证

```
默认事件先执行:
    checkbox
自定义先执行
    a
    submit
    ...
<form>
```

```
        <input type='type' />
        <input type='password' />
        <input type='submit' />

    </form>

    $(':submit').click(function(){

        $(':text,:password').each(function(){
            ...
            return false;
        })
        return false;
    })

    input,checbox
```

================================== 验证 ==================================

JS: 验证

  各种验证

```
        $(':submit').click(function(){

            $(':text,:password').each(function(){
                ...
                return false;
            })
            return false;
        })
```

后端：python 实现

业务处理

....

2、组件

BootStrap

- css
- js

学习 BootStrap 规则

一、响应式

@media

二、图标、字体
@font-face

三、基本使用

========》 后台管理

jQueryUI *
- css
- js
学习 jQueryUI 规则

EasyUI
- css
- js

学习 jQueryUI 规则
========== Ajax 操作 ==========

3、WEB 框架

MVC
Model View Controller
数据库 模板文件 业务处理

MTV
Model Template View
数据库 模板文件 业务处理

########## WEB：MVC、MTV

4、Django

pip3 install django
C:\Python35\Scripts // 将这个路径添加到环境变量（替换为自己的 python 路径）

创建 Django 工程
django-admin startproject 【工程名称】

mysite
- mysite # 对整个程序进行配置
- init

- settings # 配置文件
- url # URL 对应关系
- wsgi # 遵循 WSGI 规范，uwsgi + nginx

- manage.py # 管理 Django 程序:
  - python manage.py
  - python manage.py startapp xx
  - python manage.py makemigrations
  - python manage.py migrate

运行 Django 功能
python manage.py runserver 127.0.0.1:8001

chouti
- chouti
  - 配置
- 主站 app
- 后台管理 app

创建 app
python manage.py startapp cmdb
python manage.py startapp openstack
python manage.py startapp xxoo....

app:

migrations	数据修改表结构
admin	Django 为我们提供的后台管理
apps	配置当前 app
models	ORM,写指定的类　通过命令可以创建数据库结构
tests	单元测试
views	业务代码

1、配置模板的路径

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
```

```
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

2、配置静态目录

static

```
STATICFILES_DIRS = (
    os.path.join(BASE_DIR, 'static'),
)
```

```
<link rel="stylesheet" href="/static/commons.css" />
```

内容整理

1. 创建 Django 工程

   django-admin startproject 工程名

2. 创建 APP

   cd 工程名

   python manage.py startapp cmdb

3、静态文件

project.settings.py

```
STATICFILES_DIRS = (
    os.path.join(BASE_DIR, "static"),
)
```

4、模板路径

DIRS ==> [os.path.join(BASE_DIR,'templates'),]

5、settings 中

middlerware

注释 csrf

6、定义路由规则

url.py

"login" --> 函数名

7、定义视图函数

app 下 views.py

```
def func(request):
    # request.method      GET / POST

    # http://127.0.0.1:8009/home?nid=123&name=alex
    # request.GET.get('',None)     #  获取请求发来的而数据

    # request.POST.get('',None)



    # return HttpResponse("字符串")
    # return render(request, "HTML 模板的路径")
    # return redirect('/只能填 URL')
```

8、模板渲染

特殊的模板语言

-- {{ 变量名 }}

```
    def func(request):
        return render(request, "index.html", {'current_user': "alex"})


    index.html

    <html>
    ..
        <body>
            <div>{{current_user}}</div>
        </body>

    </html>

    =====>  最后生成的字符串

    <html>
    ..
        <body>
            <div>alex</div>
        </body>

    </html>
```

-- For 循环

```
    def func(request):
        return render(request, "index.html", {'current_user': "alex", 'user_list': ['alex','eric']})
```

index.html

```
<html>
..
    <body>
        <div>{{current_user}}</div>

        <ul>
            {% for row in user_list %}

                {% if row == "alex" %}
                    <li>{{ row }}</li>
                {% endif %}

            {% endfor %}
        </ul>

    </body>

</html>
```

#####索引#################

```
def func(request):
    return render(request, "index.html", {
                    'current_user': "alex",
                    'user_list': ['alex','eric'],
                    'user_dict': {'k1': 'v1', 'k2': 'v2'}})
```

index.html

```
<html>
..
    <body>
        <div>{{current_user}}</div>

        <a> {{ user_list.1 }} </a>
        <a> {{ user_dict.k1 }} </a>
        <a> {{ user_dict.k2 }} </a>

    </body>

</html>
```

条件

```
def func(request):
    return render(request, "index.html", {
                    'current_user': "alex",
```

```
                                    "age": 18,
                                    'user_list': ['alex','eric'],
                                    'user_dict': {'k1': 'v1', 'k2': 'v2'}})


            index.html

            <html>
            ..
                <body>
                    <div>{{current_user}}</div>

                    <a> {{ user_list.1 }} </a>
                    <a> {{ user_dict.k1 }} </a>
                    <a> {{ user_dict.k2 }} </a>

                    {% if age %}
                        <a>有年龄</a>
                        {% if age > 16 %}
                            <a>老男人</a>
                        {% else %}
                            <a>小鲜肉</a>
                        {% endif %}
                    {% else %}
                        <a>无年龄</a>
                    {% endif %}
                </body>

            </html>
```

XXOO 管理:

- MySQL
- SQLAlchemy
- 主机管理（8 列）:
  - IP
  - 端口
  - 业务线
  - ...
- 用户表:
  - 用户名
  - 密码
- 功能:
  - 1、 登录
  - 2、主机管理页面
    - 查看所有的主机信息（4 列）

- 增加主机信息（8 列） ** 模态对话框

3、查看详细

url:

"detail" -> detail

```
def detail(reqeust):
    nid = request.GET.get("nid")
    v = select * from tb where id = nid
    ...
```

4、删除

del_host -> delete_host

```
def delete_host(request):
    nid = request.POST.get('nid')
    delete from tb where id = nid
    return redirect('/home')
```