

# CTENETのBlueKing运维笔记

Written For Centos Operating System

## 前端环境

### 配置指定版本nodejs

这里以配置Node.js V10.15.1为例介绍

1. 打开终端并以root用户身份登录。
2. 安装Node.js的版本管理工具nvm (Node Version Manager) 。  
运行以下命令来下载和安装nvm:

```
1 | curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
```

3. 运行 `source` 命令以使nvm生效:

```
1 | source ~/.bashrc
```

4. 安装Node.js V10.15.1 运行以下命令

```
1 | nvm install 10.15.1
```

5. 验证是否安装成功

```
1 | node -v
```

```
(base) [root@VM-24-4-centos ~]# source ~/.bashrc
(base) [root@VM-24-4-centos ~]# nvm install 10.15.1
Downloading and installing node v10.15.1...
Downloading https://nodejs.org/dist/v10.15.1/node-v10.15.1-linux-x64.tar.xz...
##### 100.0%
Computing checksum with sha256sum
Checksums matched!
Now using node v10.15.1 (npm v6.4.1)
Creating default alias: default -> 10.15.1 (-> v10.15.1)
(base) [root@VM-24-4-centos ~]# node -v
v10.15.1
(base) [root@VM-24-4-centos ~]# node -v
v10.15.1
```

# 安装vue并创建项目测试

此前安装好了Nodejs和npm 测试版本号

```
1 node -v
2 npm -v
```

## 1. 设置镜像并测试

```
1 npm config set registry
  https://registry.npm.taobao.org
2 npm config get registry
```

## 2. 安装2.0脚手架，建立软链接，测试

```
1 npm install -g vue-cli
2 ln -s /usr/local/software/node/nodejs/bin/vue
  /usr/bin/vue
3 vue -V
```

## 创建项目并测试

### 1. 创建Vue应用程序

```
1 vue init webpack my-vue-app
```

该命令将创建一个基础的Vue项目，并将其安装在“my-vue-app”目录下。你可以将“my-vue-app”替换为你想要的目录名称。

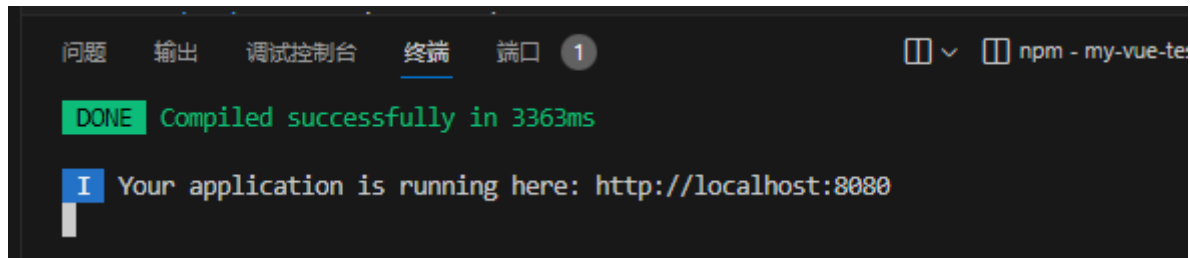
安装程序会提示你输入一些信息，例如应用程序名称、描述、作者等。你可以按照提示键入相应的信息。在安装过程中，你还需要选择一些Vue配置选项，例如是否使用ESLint、CSS预编译器等。

### 2. 进入目录并安装依赖项

```
1 cd my-vue-app
2 npm install
```

### 3. 启动服务并测试

```
1 | npm run dev
```



## Welcome to Your Vue.js App

### Essential Links

[Core Docs](#)

[Forum](#)

[Community Chat](#)

[Twitter](#)

[Docs for This Template](#)

### Ecosystem

[vue-router](#)

[vuex](#)

[vue-loader](#)

[awesome-vue](#)

## 后端环境

### 配置miniconda作为Python环境管理器

在Linux的下载命令：

```
1 | wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

## 安装

```
1 sh Miniconda3-latest-Linux-x86_64.sh
```

随后按照提示按回车即可 不用更改任何设置 一路回车直到以下问题:

安装完后, 会问: `Do you wish the installer to initialize Miniconda3`

这里输入yes即可

随后输入 `source ~/.bashrc` 激活miniconda

## 创建Python3.70环境并安装Django2.2.12相关模块

- 创建虚拟环境并指定Python版本号 `conda create -n py36 python=3.6`
- 安装Django2 `pip3 install Django==2.2.12`

```
(py36) (base) [root@VM-24-4-centos BlueKingSRE]# pwd
/Projects/BlueKingSRE
(py36) (base) [root@VM-24-4-centos BlueKingSRE]# cd ~/miniconda3/
(py36) (base) [root@VM-24-4-centos miniconda3]# ls
bin  compiler_compat  conda-meta  etc  lib  man  sbin  shell  x86_64-conda_cos7-linux-gnu
cmake  condabin  envs  include  LICENSE.txt  pkgs  share  ssl  x86_64-conda-linux-gnu
(py36) (base) [root@VM-24-4-centos miniconda3]# cd envs/
(py36) (base) [root@VM-24-4-centos envs]# ls
InitialPy  py36
(py36) (base) [root@VM-24-4-centos envs]# cd py36/
(py36) (base) [root@VM-24-4-centos py36]# ls
bin  compiler_compat  conda-meta  include  lib  share  ssl  x86_64-conda_cos7-linux-gnu  x86_64-conda-linux-gnu
(py36) (base) [root@VM-24-4-centos py36]# cd bin/
(py36) (base) [root@VM-24-4-centos bin]# ls
2to3      lzcmp      pip        python3-config  tput      xzdec
2to3-3.6  lzdiff     pip3       pyvenv          tset      xzdiff
captoinfo lzgrep     __pycache__ pyenv-3.6       unlzma    xzegrep
clear     lzfgrep    pydoc      reset           unxz      xzfgrep
```

- 检查

```
1 import django
2
3 print(django.get_version()) # 查看当前Django版本
4 print(django.VERSION) # 查看当前Django版本
5
6 # 查看当前使用的Python版本及conda环境
7 import sys
8 print(sys.version)
9 print(sys.version_info)
```

```
10 print(sys.executable)
11 # 查看conda环境
12 # conda env list
13 # conda 创建环境并指定python版本号为3.6
14 # conda create -n py36 python=3.6
```

```
(py36) (base) [root@VM-24-4-centos bin]# pwd
/root/miniconda3/envs/py36/bin
(py36) (base) [root@VM-24-4-centos bin]# python -u "/Projects/BlueKingSRE/hello.py"
2.2.12
(2, 2, 12, 'final', 0)
3.6.13 |Anaconda, Inc.| (default, Jun  4 2021, 14:25:59)
[GCC 7.5.0]
sys.version_info(major=3, minor=6, micro=13, releaselevel='final', serial=0)
/root/miniconda3/envs/py36/bin/python
(py36) (base) [root@VM-24-4-centos bin]#
```

## Django报错相关

### 1045, "Access denied for user xxx"

```
py", line 353, in __init__
    self.connect()
  File "/root/miniconda3/envs/py36/lib/python3.6/site-packages/pymysql/connections.
py", line 633, in connect
    self._request_authentication()
  File "/root/miniconda3/envs/py36/lib/python3.6/site-packages/pymysql/connections.
py", line 907, in _request_authentication
    auth_packet = self._read_packet()
  File "/root/miniconda3/envs/py36/lib/python3.6/site-packages/pymysql/connections.
py", line 725, in _read_packet
    packet.raise_for_error()
  File "/root/miniconda3/envs/py36/lib/python3.6/site-packages/pymysql/protocol.py"
, line 221, in raise_for_error
    err.raise_mysql_exception(self._data)
  File "/root/miniconda3/envs/py36/lib/python3.6/site-packages/pymysql/err.py", lin
e 143, in raise_mysql_exception
    raise errorclass(errno, errval)
django.db.utils.OperationalError: (1045, "Access denied for user 'papertest'@'local
host' (using password: YES)")
```

原因：未授权的数据库访问操作

解决方法：在Django应用的settings.py文件中 修改数据库配置 将用户修改为root，并保证密码的正确即可

# AttributeError: 'str' object has no attribute 'decode'

```
1  Traceback (most recent call last):
2  File "manage.py", line 15, in <module>
3  execute_from_command_line(sys.argv)
4  File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/core/management/init.py", line 381, in
   execute_from_command_line
5  utility.execute()
6  File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/core/management/init.py", line 375, in
   execute
7  self.fetch_command(subcommand).run_from_argv(self.argv)
8  File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/core/management/base.py", line 323, in
   run_from_argv
9  self.execute(*args, **cmd_options)
10 File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/core/management/base.py", line 364, in
   execute
11 output = self.handle(*args, **options)
12 File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/core/management/base.py", line 83, in
   wrapped
13 res = handle_func(*args, **kwargs)
14 File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/core/management/commands/makemigrations
   .py", line 101, in handle
15 loader.check_consistent_history(connection)
16 File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/db/migrations/loader.py", line 283, in
   check_consistent_history
17 applied = recorder.applied_migrations()
18 File "/root/miniconda3/envs/py37/lib/python3.7/site-
   packages/django/db/migrations/recorder.py", line 73, in
   applied_migrations
19 if self.has_table():
```

```
20 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/migrations/recorder.py", line 56, in
    has_table
21 return self.Migration._meta.db_table in
    self.connection.introspection.table_names(self.connecti
    on.cursor())
22 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/base/base.py", line 256, in
    cursor
23 return self._cursor()
24 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/base/base.py", line 233, in
    _cursor
25 self.ensure_connection()
26 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/base/base.py", line 217, in
    ensure_connection
27 self.connect()
28 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/base/base.py", line 197, in
    connect
29 self.init_connection_state()
30 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/mysql/base.py", line 231,
    in init_connection_state
31 if self.features.is_sql_auto_is_null_enabled:
32 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/utils/functional.py", line 80, in get
33 res = instance.dict[self.name] = self.func(instance)
34 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/mysql/features.py", line
    82, in is_sql_auto_is_null_enabled
35 cursor.execute('SELECT @@SQL_AUTO_IS_NULL')
36 File "/root/miniconda3/envs/py37/lib/python3.7/site-
    packages/django/db/backends/utils.py", line 103, in
    execute
37 sql = self.db.ops.last_executed_query(self.cursor, sql,
    params)
```

```
38 File "/root/miniconda3/envs/py37/lib/python3.7/site-  
packages/django/db/backends/mysql/operations.py", line  
146, in last_executed_query  
39 query = query.decode(errors='replace')  
40 AttributeError: 'str' object has no attribute 'decode'
```

这个报错是在 Django 的数据库迁移过程中出现的错误。根据报错信息，在执行迁移命令时，Django 尝试执行了一条查询语句 `SELECT @@SQL_AUTO_IS_NULL`，但是在查询结果的处理过程中出现了问题。

从报错信息中可以看出，问题发生在 `django/db/backends/mysql/operations.py` 文件的 `last_executed_query` 方法中的 `query.decode(errors='replace')` 这一行代码。根据错误信息，`query` 被当作字符串处理，但是字符串对象没有 `decode` 方法，导致了这个错误。

这个问题的产生可能与 Python 版本和 Django 版本相关。在 Python 3 中，字符串对象已经是 Unicode 对象，不再需要进行解码操作。而在早期的 Django 版本中，可能还使用了 Python 2 的字符串处理方式，导致了这个错误。

解决这个问题的方式是升级 Django 版本。确保你使用的是兼容 Python 3 的最新版本的 Django。另外，也可以检查一下你的 Python 环境是否正确配置，并且所有依赖包的版本是否兼容。

如果你已经使用了最新版本的 Django，仍然遇到这个问题，可以尝试在你的代码中查找是否有对 `last_executed_query` 方法的调用，并检查是否有对查询结果进行解码的操作，如果有的话，可以尝试将其移除或修改为适用于 Python 3 的方式。

## 不升级版本 通过本地修改 Django 源码解决这个问题

如果你不想升级 Django 版本，可以尝试下面的解决方法：

根据错误信息，问题发生在 `django/db/backends/mysql/operations.py` 文件的 `last_executed_query` 方法中的 `query.decode(errors='replace')` 这一行代码。这里的 `decode` 方法是用于将字符串解码为 Unicode 对象，然而在 Python 3 中，字符串已经是 Unicode 对象，不再需要解码。



你可以尝试修改 `django/db/backends/mysql/operations.py` 文件，将 `query.decode(errors='replace')` 这一行代码修改为 `query`，即直接返回查询语句的字符串。

具体做法如下：

1. 打开文件 `django/db/backends/mysql/operations.py`。
2. 找到 `last_executed_query` 方法，定位到 `query.decode(errors='replace')` 这一行代码。
3. 将这一行代码修改为 `return query`。
4. 保存文件并重新运行你的 Django 项目，看是否问题解决。

这样修改后，应该可以解决 `'str' object has no attribute 'decode'` 的错误。但是请注意，这只是一种临时解决方法，可能会导致其他潜在问题。建议在有机会时尽快升级 Django 版本，以获得更好的兼容性和稳定性。

文件路径：

```
1 /root/miniconda3/envs/py37/lib/python3.7/site-packages/django/db/backends/mysql
```

```
operations.py x
format_str)~
123 ~ .....else:~
124 ~ .....return "TIME(%s)" %
125 ~
126 ~ .....def date_interval_sql(self, timedelta):~
127 ~ .....return 'INTERVAL %s MICROSECOND' % duration_microseconds(timedelta)~
128 ~
129 ~ .....def format_for_duration_arithmetic(self, sql):~
130 ~ .....return 'INTERVAL %s MICROSECOND' % sql~
131 ~
132 ~ .....def force_no_ordering(self):~
133 ~ ....."""~
134 ~ .....    "ORDER BY NULL" prevents MySQL from implicitly ordering by grouped~
135 ~ .....    columns. If no ordering would otherwise be applied, we don't want any~
136 ~ .....    implicit sorting going on.~
137 ~ .....    """~
138 ~ .....    return [(None, ("NULL", [], False))]~
139 ~
140 ~ .....def last_executed_query(self, cursor, sql, params):~
141 ~ .....    # With MySQLdb, cursor objects have an (undocumented) "_executed"~
142 ~ .....    # attribute where the exact query sent to the database is saved.~
143 ~ .....    # See MySQLdb/cursors.py in the source distribution.~
144 ~ .....    query = getattr(cursor, '_executed', None)~
145 ~ .....    # if query is not None:~
146 ~ .....    # .....query = query.decode(errors='replace')~
147 ~ .....    return query~
148 ~
149 ~ .....def no_limit_value(self):~
150 ~ .....    # 2**64 - 1, as recommended by the MySQL documentation~
151 ~ .....    return 18446744073709551615~
152 ~
153 ~ .....def quote_name(self, name):~
154 ~ .....    if name.startswith("`") and name.endswith("`"):~
155 ~ .....        return name # Quoting once is enough.~
156 ~ .....    return "`%s`" % name~
157 ~
158 ~ .....def random_function_sql(self):~
159 ~ .....    return 'RAND()'~
```